

Lecture 8: Designing complex systems as interaction spaces

Case studies on applying theory to hard HCI problems

Overview of the course

- Theory driven approaches to HCI
- Design of visual displays
- Goal-oriented interaction
- Designing efficient systems
- Designing smart systems
- Designing meaningful systems (guest lecturer)
- Evaluating interactive system designs
- **Designing complex systems**

What are some things that make designs complex?

- How complex is the domain?
- How many different tasks might a user perform?
- How well defined are the outcomes? (Wicked problems, L3)
- How easy is it to understand each part?
- When the parts are put together how easy is to guess the behaviour?
- Does the system do things when the user isn't there? (Attention Investment from L3)

Designing tasks vs interaction spaces

Consider a (slightly silly) API for sending a message:

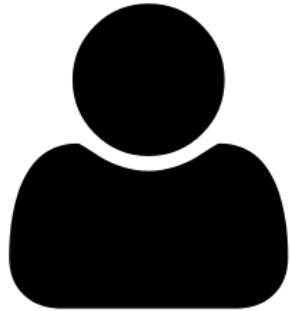
- (1) **sendTheRightMessage()**
- (2) **sendMessage(Enum message)**
- (3) **sendMessage(String message, Urgency status)**

- Naive design would produce (1). Complex systems tend to be built out of reusable components that the users configure (2,3)
- Building this kind of system involves discussing tradeoffs as well as detailed design decisions
- This is the kind of interaction space that most of you will build:
Programming languages, APIs, AI systems

Broad brush techniques

- Descriptions of specific actions result in a ‘death by detail’
- Don’t describe specific actions with an interface
 - Describe interaction with a level of *analytical distance* from the interface
 - Use an *analytical frame* which is a way of structuring a description of an interaction
 - The description can then be compared to an ideal for a domain to become a critical perspective (see Lecture 1)
- The right level of detail resembles an object-oriented design pattern, but for *human behaviour* rather than software.

Cognitive Dimensions of Notations (CDNs): Analytical Frame



A user

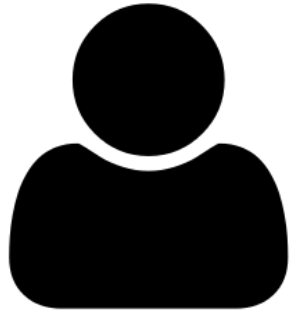


Performs an
activity

A screenshot of a code editor window titled 'min.dart' showing Dart code for a WebSocket client and server. The code includes imports for 'dart:io', 'dart:isolate', and 'dart:typed_data'. The server code defines a 'WebSocketChannel' class and a 'main' function that starts a server on port 9998. The client code defines a 'main' function that connects to the server. The editor shows a file explorer on the left with a project structure including 'pub', 'example', 'lib', 'src', 'test', and 'packages'. The bottom status bar shows 'example/min.dart 12:2 UTF-8 Dart 2 updates'.

Interface containing
notations, described
along a number of
dimensions

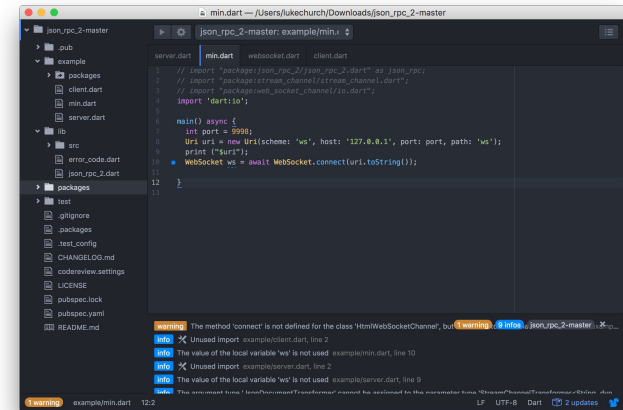
Cognitive Dimensions of Notations (CDNs): Analytical Frame



A user

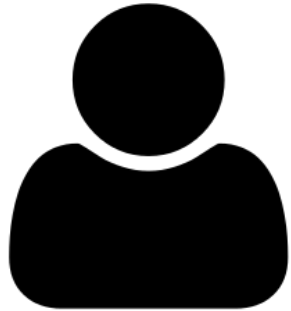


Performs an
activity



Interface containing
notations, described
along a number of
dimensions

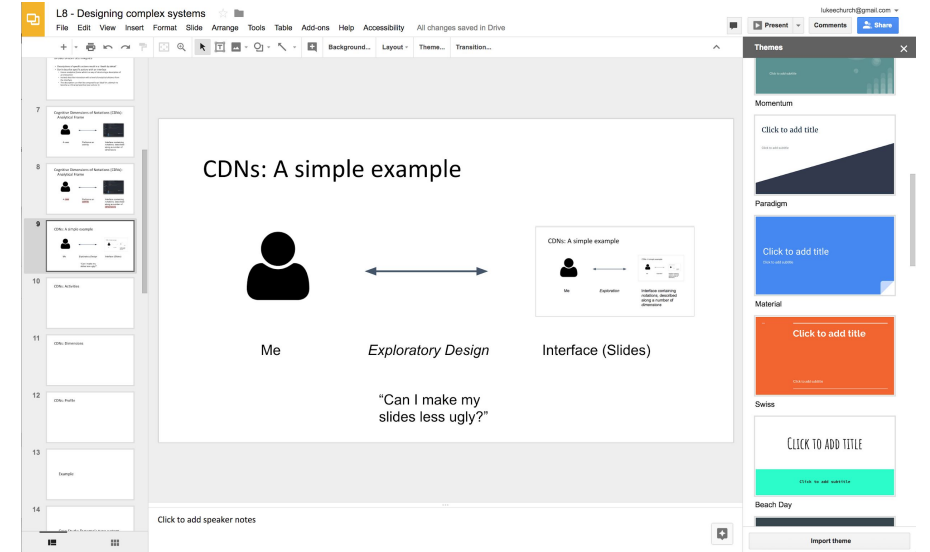
CDNs: A simple example



Me



Exploratory Design



Interface (Google Slides)

“Can I make my slides less ugly?”

CDNs: A simple example (Demo)

- One described change “Make the font of the headings Comic Sans”
 - Select the first slide, change the font
 - Select the second slide, change the font
 - Yawn.
- This is repetition *Viscosity*, many operations to perform one change
- Design maneuver: Introduce an *Abstraction* (master slide), decreases *Viscosity*, but increases *Premature Commitment*
- NB: CDNs analysis is meaningless independent of an interface.

CDNs: Activities



► EXPLORATION

Manipulating both information and structure

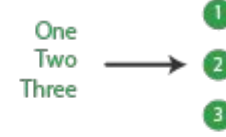
Exploration involves manipulating, and changing, both the content and the structure of the information



► MODIFICATION

Changing structure only

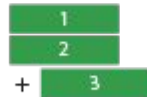
Modification is changing the structure of the information, but keeping the content the same. Also known as 'refactoring'



► TRANSCRIPTION

From one notation to another

Transcription involves copying information from one notational form to another, often between different media as well



► INCREMENTATION

Adding one more

Incrementation is adding new content, but leaving the information structure alone



► READING

Seeking information or gist

Reading doesn't involve changing the content or structure, but finding information either for detail or an overview

CDNs: Dimensions



▶ ROLE EXPRESSIVENESS

How much elements suggest their purpose
Some syntax suggests its purpose better than others, this can help learning a system but is difficult to achieve without accepting the limitations of existing conventions



▶ CONSISTENCY

Similar meanings, similar syntax
Internal consistency is important for understanding systems



▶ PREMATURE COMMITMENT

Constraints on the order of decisions
Constraints on order often involve people having to make decisions before they would normally do in the course of solving a problem. This pre-thinking obstructs exploration, but can cause useful forwardness



▶ VISCOSITY

Resistance to change
One change in the mind becomes many operations in the interface.
Viscosity is commonly exchanged for an abstraction.

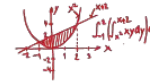
ADD AGE TO NEW_AGE GIVING NEW_AGE

vs

new_age += age

▶ DIFFUSENESS

The spread-out-ness of information
Information that is very diffuse can make it easier to see but constrains the amount that can be seen at any one time, decrease justaposition and can add to viscosity



▶ USEFUL AWKWARDNESS

Thinking hard is sometimes useful
Some notations cause the user to have to deeply consider their domain, which sometimes results in useful results. This is often caused by Premature Commitment.



▶ ABSTRACTION

Mechanisms for generality
Abstractions support operations over multiple objects, or when the user isn't present
Abstractions provide support for efficient use, but may increase cognitive load, user perception of risk, and premature commitment



▶ SYNOPSIS

Provides an understanding of the whole
Some notations provide a sense that you can step back and get a holistic impression, also described as the 'gastak view'



▶ HIDDEN DEPENDENCIES

Unexpected relationships
When one item is changed another, seemingly unconnected, item changes.
Harmful to exploration. Commonly reduced by making the dependencies visible, at the expense of diffuseness and viscosity



▶ HARD MENTAL OPERATIONS

Some things are just Hard
Some tasks that are known to be cognitively challenging, for example remembering lists of different possible branches



▶ PROVISIONALITY

Degree of commitment to marks
High provisionality supports exploratory strategies such as playing 'what-if' games but can increase viscosity by expecting users to state when they are easy to commit to their marks

<[^>]* >) / 1

▶ LEGIBILITY

Readability of the notation
Various factors affect a notations readability such as how distinguishable its characters are and how well it supports perceptual parsing. Often trades off with diffuseness



▶ CLOSENESS OF MAPPING

Correspondence to the domain being expressed
A close relationship between the notation and the domain that it models makes implementation easier but may result in inflexibility and diffuseness



▶ PROGRESSIVE EVALUATION

Feedback along the way
Progressive evaluation describes how much the system displays partial progress towards a goal, or mandates the whole thing to be finished before any feedback is given.



▶ SECONDARY NOTATION

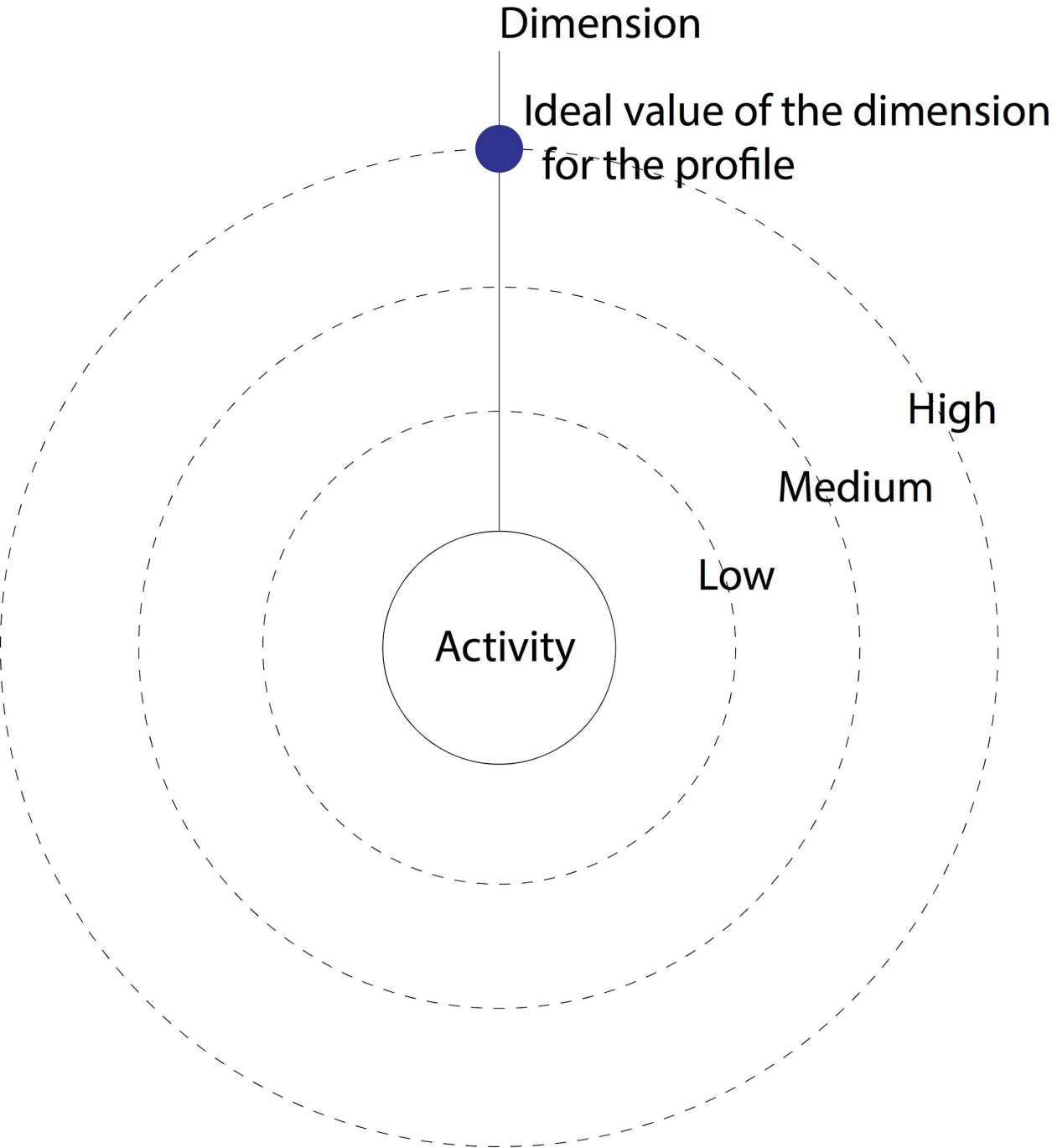
Escape from formality
Notation that is not formally interpreted. Comments, notes, break etc.
Often omitted from computer based systems



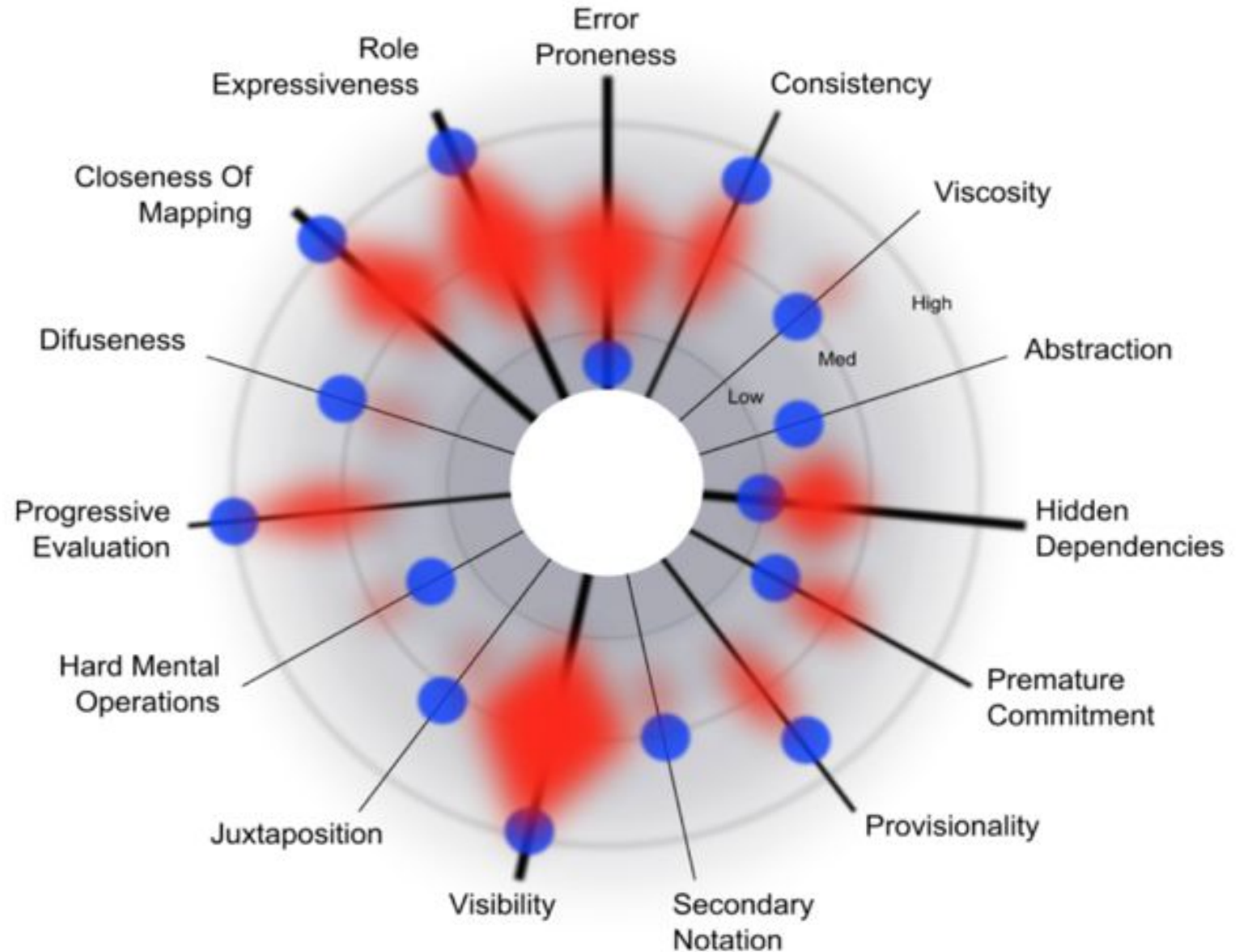
▶ JUXTAPOSITION

Simultaneous comparison
Simultaneous views of an information structure

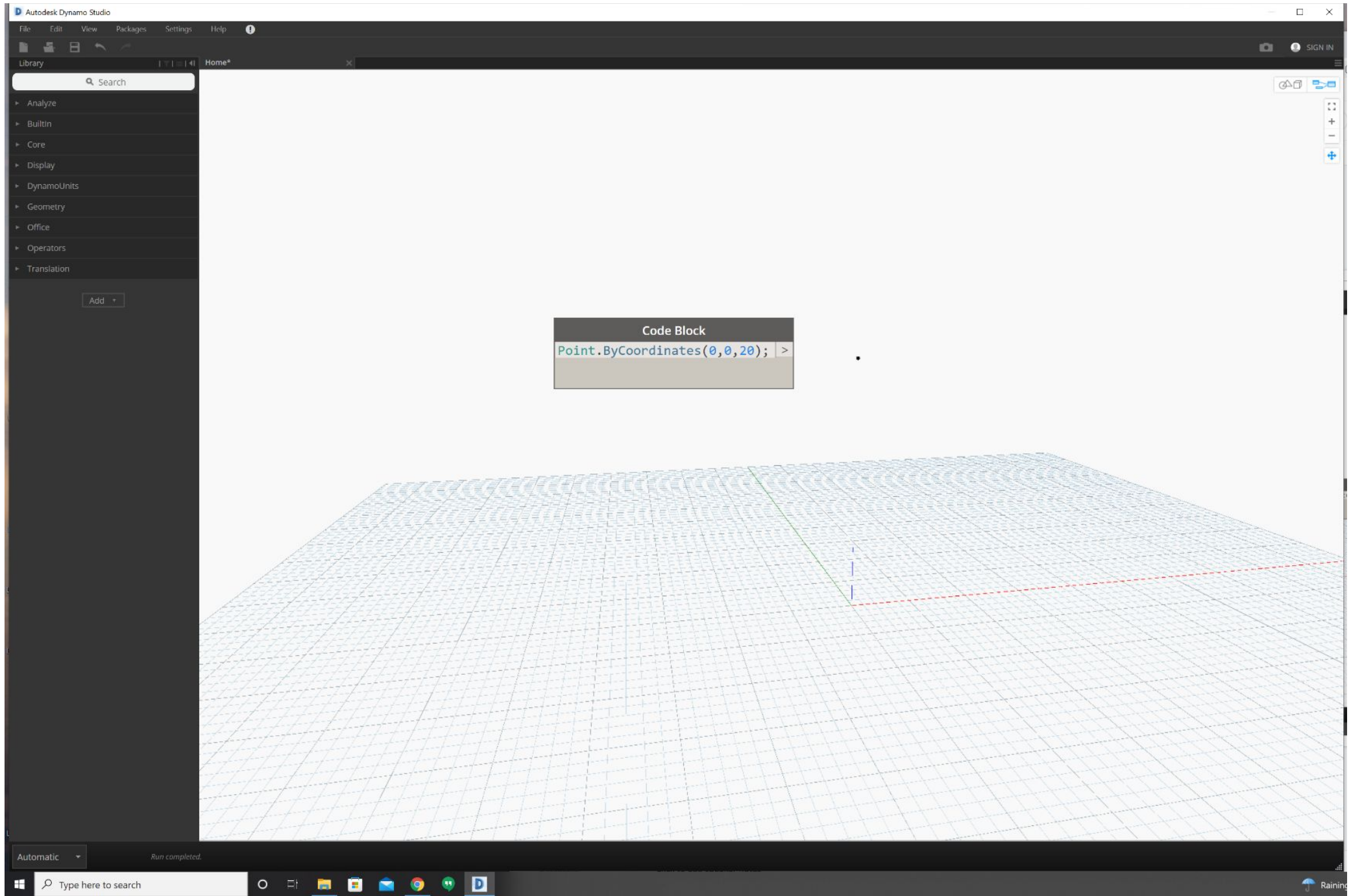
CDNs: Profile



CDNs: Profile



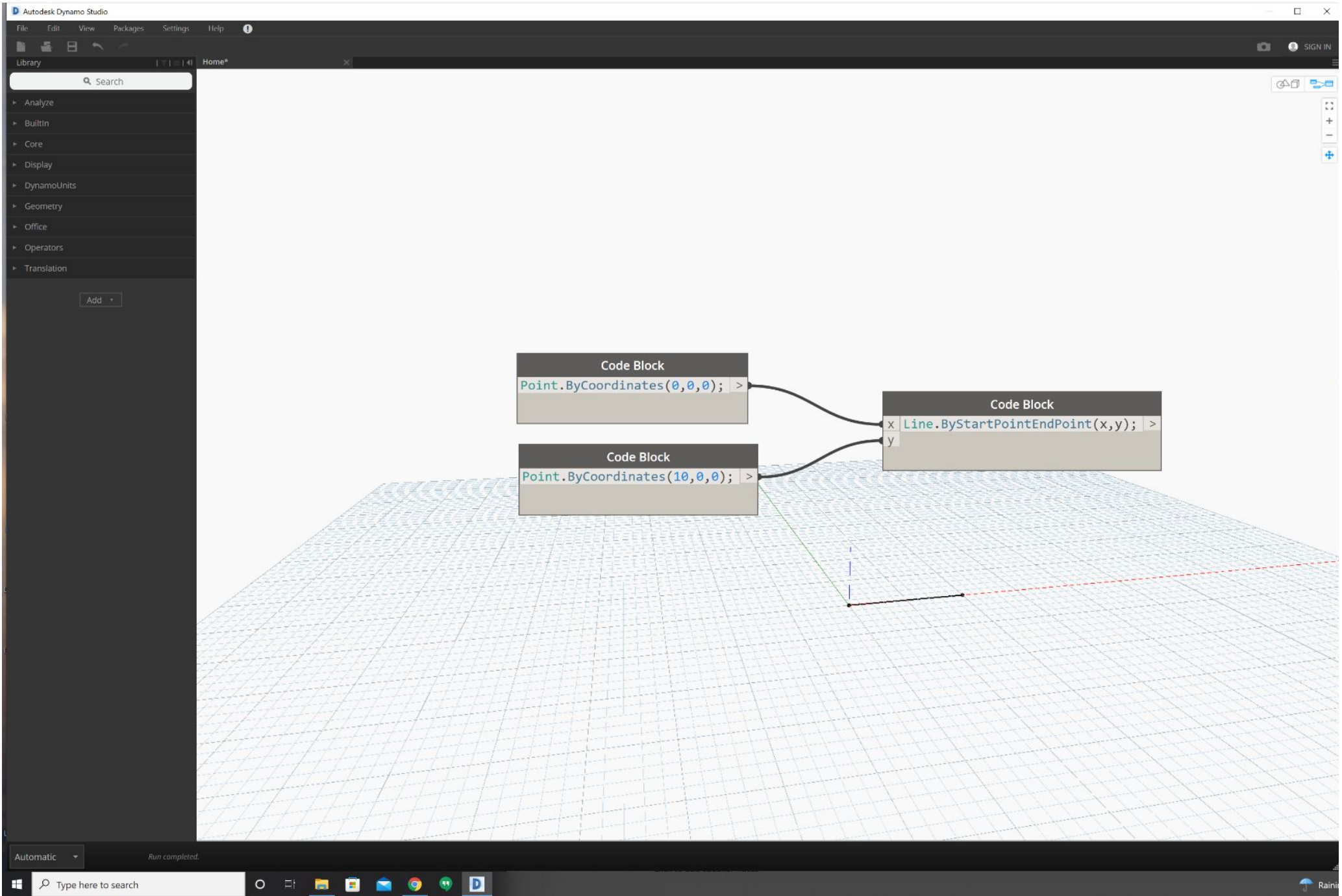
Case Study: Dynamo's type system



- Analyze
- BuiltIn
- Core
- Display
- DynamoUnits
- Geometry
- Office
- Operators
- Translation

Add

```
Code Block
Point.ByCoordinates(0,0,20); >
```



Dynamo

- Language for exploring building designs
- Includes a constructor **Point(x, y, z)** and array literal syntax [1,2]

Design question for discussion:

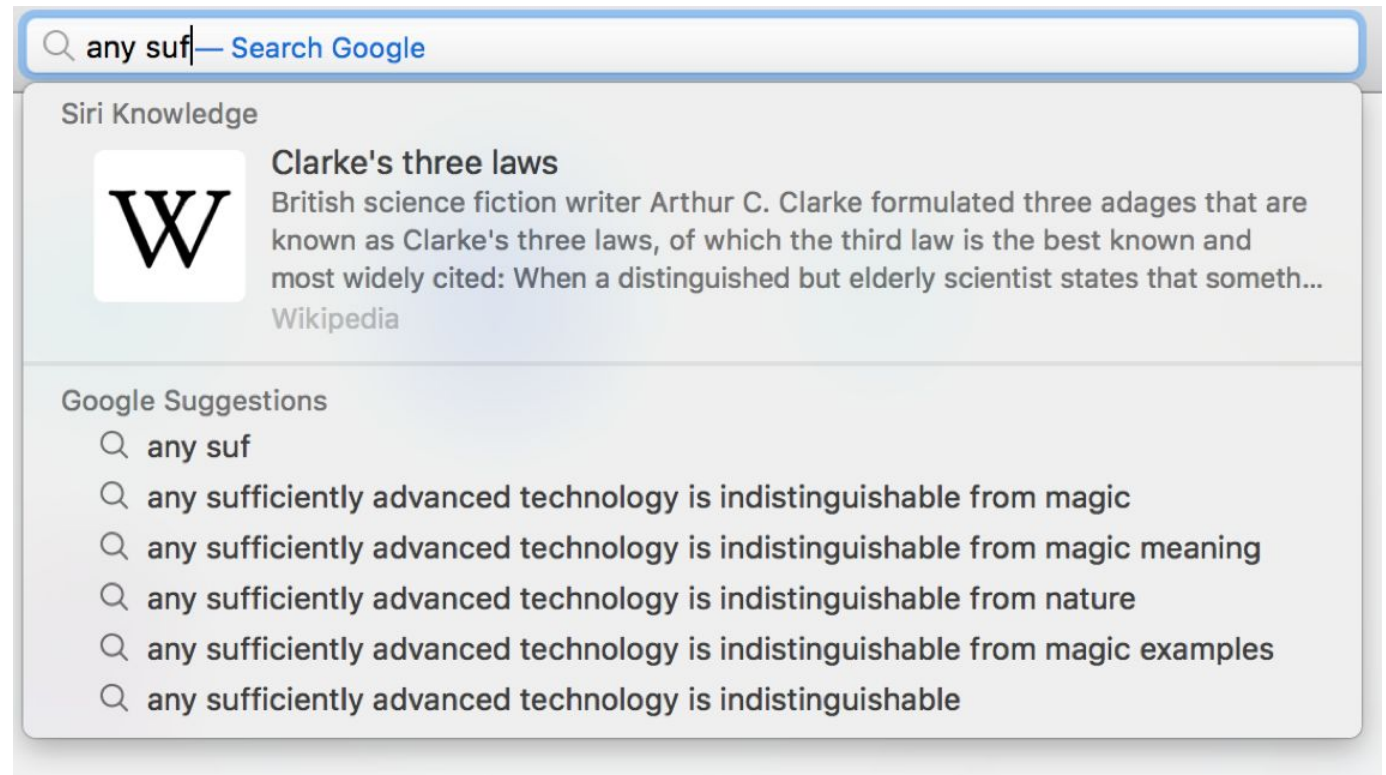
“What should **Point([0, 1, 2], 10, 10)** do?”

- What activities are important?
- How important: Viscosity? Premature commitment? Hidden Dependencies? Abstraction hunger?

Intelligent systems as interaction spaces


“Any sufficiently advanced technology will be indistinguishable from magic”

- Arthur C. Clarke, 1962



Q any suf — Search Google

Siri Knowledge

 **Clarke's three laws**
British science fiction writer Arthur C. Clarke formulated three adages that are known as Clarke's three laws, of which the third law is the best known and most widely cited: When a distinguished but elderly scientist states that someth...
Wikipedia

Google Suggestions

- Q any suf
- Q any sufficiently advanced technology is indistinguishable from magic
- Q any sufficiently advanced technology is indistinguishable from magic meaning
- Q any sufficiently advanced technology is indistinguishable from nature
- Q any sufficiently advanced technology is indistinguishable from magic examples
- Q any sufficiently advanced technology is indistinguishable

AI can feel like magic

Individual interactions with AI systems often seem quite trivial to perform, even if we don't know how we go about doing them.

Because we don't know how they work, if something goes wrong there's no obvious way of intervening



AI can feel like *scalable* magic

Having developed the capability to perform the task once, without human attention, we can now achieve results that would otherwise be impossible.

E.g. 'find all the cats on the internet'



AI can feel like scalable magic

Applying the same technique to animal behaviour analysis

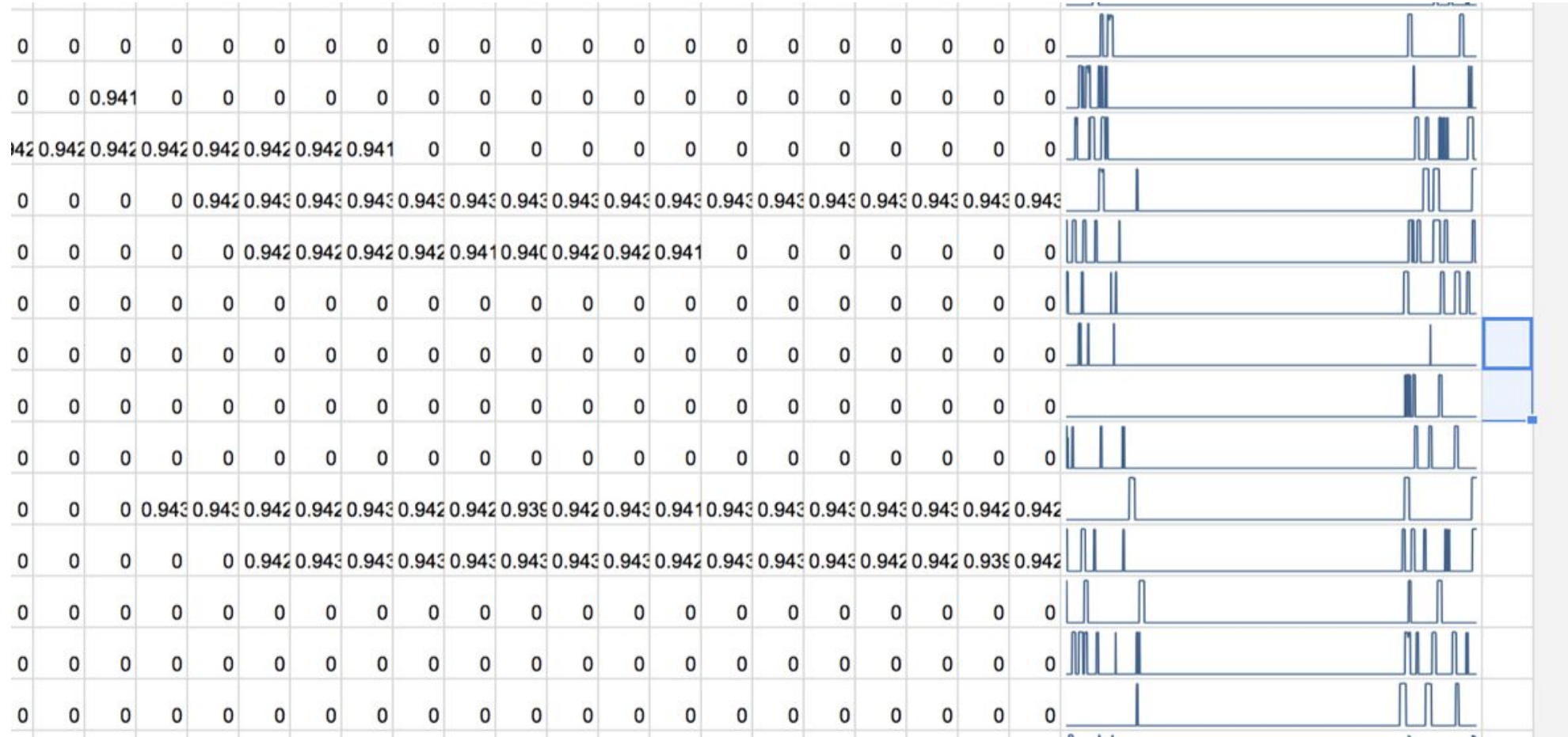


```
{
  "labelAnnotations": [
    {
      "description": "black",
      "score": 0.9555818
    },
    {
      "description": "black and white",
      "score": 0.9473888
    },
    {
      "description": "monochrome photography",
      "score": 0.90905684
    }
  ],
}
```

```
{
  "labelAnnotations": [
    {
      "description": "hedgehog",
      "score": 0.9818773
    },
    {
      "description": "erinaceidae",
      "score": 0.9702047
    },
    {
      "description": "domesticated hedgehog",
      "score": 0.9429917
    }
  ],
}
```

(Personal images)

AI can feel like scalable magic



How do you trust magic? How do you control it?

If this was the result you saw - would you believe it?



A simple example of programming with ML

- Research in 2011 by Sumit Gulwani at Microsoft Research
- “Synthesises a program from input-output examples”
 - How do you choose the examples? (Premature commitment?)
 - How do you know what will happen? (Progressive evaluation?)
-
- **Now Excel FlashFill**
 - Paste a list of semi-structured text data into the left column
 - Type an example transform result in top cell to the right, then <Enter>
 - Press <Ctrl+E>

Conversational agents

- Do they build a user model, goal model or task model?
- Will this be more or less complex than FlashFill?
- How can you see it the model?
 - i.e. what is the notation?
- How could you modify the model?
 - ... in response to errors (yours, or the system's)
 - ... if you change your goals?
- Does having a 'body' help?
 - (remember metaphor)



Human issues in machine learning

- Ethics and accountability
 - automating and/or justifying bias and prejudice
- Digital humanities
 - treating text and images as meaningful and sophisticated
 - (rather than just statistical fodder)
- Reward
 - who does the intellectual ‘work’ of providing training corpus content, data labelling, how are they paid, and where do the profits go?

Some [other] current research
problems

VR and AR still use visual correspondence



Programming, or direct manipulation?

- Many Internet of Things (IoT) devices have physical switches etc
 - But how do you define configuration, policy, future action?
 - Now we need a notation - or a programming language
- Remember behavioural economics and attention investment
 - Even around your house, bounded rationality happens



Global challenges

- Is knowledge infrastructure built to ...
 - ... prioritise low income populations
 - ... advance Sustainable Development Goals (human rights, education etc)?



Further interest...

- Part II: Project
- Part II/Part III Computer Music (not in 2022)
- Part II/Part III Advanced Graphics
- Part III: Interaction with Machine Learning
- Research Skills: Working with artists and designers; How to interpret experimental results; Introduction to qualitative research methods; How to design surveys; Assessing the quality of experience