Me, you and software development

While factors like processing power has been the main obstacle when it comes to software development the last few decades, we are now, more than ever, limited to creations that are within our abilities. We ever increasingly stumble upon a ‘language barrier’ between human and computer. Even when we continue to create programming languages that are evermore powerful and easy to understand, we still limit ourselves to whatever was imagined at the point of the languages creation. We might see the programming languages as nothing but a tool for creation, but this tool was still engineered by people who had some expectation or plan for whatever it would be used for. While this is good because it promotes and helps the development of software within the intended use case, it also halts innovation in others. The question is then how we make software development accessible to all people without sacrificing complexity, neutrality and ease of learning?

In the exam paper, formulated as a contribution to a future edition of the Software Studies Lexicon, I want to dive into the future of human-computer interaction(HCI). The chapter will primarily focus on the mediums or types of encoding involved in software development. This will include traditional programming languages and how they get formed in the context in which they are used, but also explore how other means of communication might come to use in the future.

I would like to examine trends in the evolution of vocabularies in different human languages and then comparing it to syntaxes in programming languages through time. My hypothesis is that both types of languages expands and advances when the need for differentiating or complexity is met amongst the people who intend to use it. An example of this would be a theory about language evolution published in 1969 by Paul Kay and Brent Berlin in their book *‘Basic Color Terms: Their Universality and Evolution’*. The theory is that languages follow a distinct pattern in what colors they choose to name. An explanation for this would be that the civilizations behind the languages had no need to distinct og specify different colors. Instead they would be unified under meanings or categories. As the civilization then evolved it grew a need to describe colors in fields like literature, production and art. This closely resembles the creation and need for multiple colors models as RGB, HSL, HSV, CMYK and Pantone in computer related fields. By using examples like this we might get an get an idea of how specifying complex ideas and sets of actions, into abbreviations, might become the gateway to complex yet minimalistic communication with computers.

While I think complexity might play a huge role in the future of software development, I also think that accessibility might have even more focus. My reason for thinking this is that I realised the popularity behind DIY Webpage and app programs like Squarespace, Wordpress and the Adobe programs Spark and Muse. While these are all good, they still limit you to specific styles, layouts and mindsets. This kind of ‘generic’ look is of course always avoided in the eyes of bigger companies. That’s why it would be both time- and money saving to have designers being able to develop themselves instead of being

dependent on software engineers. This I think can go multiple ways and it will be a thing I want to talk/reflect more about in the paper.

We are currently heading towards making programming languages that look more and more like ‘human languages’ like English. So there might be a lot of strategic reasons behind converging even further to a point where they become almost impossible to tell apart. This require computers to learn a deeper understanding of language, semantics and grammar. It might even include abstract, from a computers perspective, concepts like irony, metaphors, symbolism and context. But while these seem too complicated to teach computers to recognize I think we are a lot closer than you would otherwise think. This could even be done with the following option for the future of human-computer interaction.

The other path that HCI might go is that of some level of artificial intelligence. By artifical intelligence I mean computers who are able to do certain actions ‘requiring intelligens’ by themselves. The result of this might, or might not, be predictable or comparable to the way a human would solve a problem. My particular interest within this field is the use of artificial neural networks(ANNs) and evolutionary computation. ANNs work by simulating neurons in a brain to analyze patterns in things like, but not limited to, human behavior(on- and offline), pictures and speech recognition. The idea behind evolutionary computation is then to let the patterns determine the next ‘generation’ of product. While this sounds complicated and fancy, it looks a lot of humans make different versions of product and the patterns we see might be the problem that other people find with the product. By using these methods of computation we might leave a lot of programming and research behind. As my example for this I have chosen ‘MarI/O’ created by software engineer and speedrunner, Seth ‘Sethbling’ in 2015. Here he studies a computer learning to play the side scrolling game ‘Super Mario World’. It has no context about the game, rules or goal of the game. The only thing that Seth taught it was the controls and its goal was to get ‘fitness’ points the longer it got in the game. By the generation 32 of its runs through the game it learned to complete a level without dying. I think this way of software development have a lot of breakthrough possibilities.

The last thing I wanted to speculate about in the paper is the notion of neutrality, especially in the use with ANNs or other kind of AI. I would like to reflect on the implications and advantages of both neutrality and whether or not AI provides a kind neutrality we would desire. I would imagine a perfectly modeled ruleset to understanding humans would provide us with tools and programs beyond our imagination. But I could also imagine that a computer would have a hard time understanding the sometimes impulsive human behavior. And furthermore by trying to map or analyze it would end in undesirable results. Even if a computer was to have a complete understanding of humans and designing things for us, would we then miss the incompleteness and imperfections that is connected with products made by humans?