**Notes:**

Roughly 10 pages

* 2.5 pages per section
* max 15/20ish

By product - what does it mean to be a software engineer?

**15/11 Lecture Notes:**

**Format:**

1. Introduction
2. Software Engineering Process - sample layout from student
   1. Measurable Data
   2. Development Analysis Platforms
   3. Algorithmic Approaches
   4. Ethical Analysis
3. Conclusion

**Stephen’s Ideas:**

* Reading list
* Machine learning faults - e.g gender issues/biased data sets?
* Form your own opinion! As long as you back it up

**Measuring Data:**

* Start off that software engineering is measurable - that’s not the question
  + Can you measure it usefully?
  + Are the measurements trustworthy/fair?
  + Ambiguity in measurement is the problem - measured unfairly because some things are easy to collect - e.g lines of code, bug fixes
    - Can game it - forced to do the wrong thing - push quality down
    - Examples from other industries where taking the wrong measurements has had a negative effect - journalism (expected to write too many articles)
* Engineers are comfortable that the work they do might not be measurable - ego?
  + Can spot gaming
  + Significant difference of performance of engineers in teams
    - Not everyone can keep up
    - Engineers know that some are better than others - measuring performance makes people insecure?

**Development Analysis Platforms:**

* Many students talk about - personal software development process - engineer measures their own activity (framework?)
  + How do you combine measures together to build comprehensible platforms
* Another direction (more what they want) - if you have huge volumes of data and want to analyse cohort, where can you do that processing cost effectively and efficiently
  + How cheap it has become to do machine learning
  + Utility computing - precursor to all machine learning
  + Renting hardware to renting software
    - Iaas, PaaS, AaaS/Saas
* Their expectation - if I want to process this data where do I go
  + Google, Amazon, etc. offer utility computing that makes it doable
    - If I want to run this, where do I go?
  + But also analytics as a service products
  + Platform framework approach to analyse data - increasingly use AI

**Algorithmic Approaches:**

* Want to look at kind of algorithms used to measure underlying data sets in software engineering
  + Measure code they generate - functional complexity, is code overriding other code, is your code replaced, etc.
* Some students ask what do we mean by functional complexity and how can that be measured (field called software engineering metrics - GitPrime/PluralSight - Impact)
  + All these measures are contextual e.g not fair if you’re only asked to fix bugs and then are seen as less impactful
* Another direction - things like cyclomatic complexity and other measures and what can you do with those measures to see if an engineer is productive, etc.
  + What’s the nature of what we’re trying to measure?
  + What’s the appropriate algorithm to use for that?
  + Stephen’s view: machine learning is bad at this, ML tends to smooth out the noise and differences in data (what’s the middle ground?)
    - Human judgement needed? Context driven - leads to different viewpoints
    - e.g I judge code and think it’s idiomatically beautiful - then look at forums and see that it’s copied code from other parts, doesn’t solve bugs - always need to think about facts that aren’t visible
    - Abductive reasoning isn’t captured well by machine learning
    - Needs to be sensitive to nuanced differences in data - like humans
    - ML better with statistics, etc. to remove noise - here if you remove noise you could lose core value
* If Stephen was answering - Focus on often contradictory/confusing ways like cyclomatic complexity - no one actually thinks it’s measuring complexity - lots of different answers
  + No definitive way to capture complexity - context specific
  + Relative values can be handy
  + Using AI/ML to gather some data - is one engineer’s work at a similar level to another’s
  + Ultimately those kinds of techniques miss the point - can’t rely entirely on ML
    - Need to develop algorithms that can mimic human judgement - abductive reasoning - huge amount of work to do there
  + Best way - provide tools to help humans make judgement, e.g uncover facts and then humans decide

**29/11 Lecture Notes:**

**Ethics:**

Relating to the first 3 parts:

* 1. Metrics – lots of data
* 2. Platforms – cheap to process data
* 3. Algorithms – split into:
  + Machine Learning – easy to do – human behaviour is complex and subtle
    - Debateable whether this is good enough to determine if someone’s doing a good job
  + Expert Systems – sophisticated, hard to built

Judging human behaviour is a subtle problem – some of it is routine

* Ratio of subtle to routine activities in SE is greater – therefore ML struggles more

1. (Individual) Consequences – is the work you’re going to going to have negative impact on people – consequences of your measurement

2. Accuracy – what is it like to be measured?

* It needs to be accurate if you’re using it to judge people

3. Societal – is world better or worse from what you’re doing

* Need to measure well?
* If you don’t measure – you don’t know what’s going on

Would accurate measurement counteract biases?

* People tend to hire/promote people like them

Take your own view – reference material

**Four Sections:**

1. How one can **measure** engineering activity

* e.g look at structure of code and calculate complexity
* e.g frequency of code commits, try to decide how productive engineer is
* Explore and report findings on this
* What can you measure?

1. **Platforms/frameworks** on which one can gather and perform calculations over these data sets

* Important infrastructure here is emergence of utility computing which supports gathering of large volumes of data and processing of data via algorithms
* Infrastructure enables complex and computationally expensive calculations
* Explore what is possible here and what specialist infrastructure has emerged various kinds of data analysis
* How can you measure?

1. Various kinds of computation that could be done over software engineering data to provide profile of performance of software engineers - what **algorithms** can we use

* There are various techniques for data computation
  + e.g defeasible argumentation of the kind used in expert systems, simple counting, various software engineering algorithms measuring concepts like coal and complexity, and various machine learning approaches for plastering/analysing data sets
* Set out the landscape and consider strengths and weaknesses of various techniques
* AI/Machine Learning algorithms used to analyse - **fidelity** - how accurate are they?

1. **Ethics** and legal/moral issues surrounding the processing of this personal data

* Is it reasonable to perform analysis on the performance of software engineers as they do their work?
* Form an opinion that is informed by your prior research