Advanced Software Engineering Course

Increasing Productivity

Sadegh Sulaimany

info@Bioinfotmation.ir

دانشگاه کردستان
University of Kurdistan
زانکۆی کوردستان

SBNA
Social & Biological Network Analysis Laboratory

Initial assessment

1. What are this tools in software engineering used for:
    1. Cucumber
    2. Pivotal Tracker
    3. JIRA
    4. Makefile

Agenda

- Four productivity mechanisms in Software Engineering
  - Clarity via Conciseness
  - Synthesis
  - Reuse
  - Automation
- SaaS & SOA & Agile
  - Microservice

Productivity dependency of hardware progress

› Moore's Law
  – hardware resources have doubled every 18 months for nearly 50 years
  – faster computers with much larger memories could run much larger programs

› software engineers needed to improve their productivity

Fundamental mechanisms to improve productivity

1. Clarity via conciseness
2. Synthesis
3. Reuse
4. Automation via Tools

Mechanisms to improve productivity

- **Clarity via conciseness**
  - if programs are easier to understand,
    - they will have fewer bugs and will be easier to maintain
  - if the program is smaller
    - it's generally easier to understand

    1. **Offering syntax with fewer characters**
    2. **Raise the level of abstraction**

1. Clarity via conciseness

› Offering syntax with fewer characters

– Example

- `assert_greater_than_or_equal_to(a, b)`
- `expect(a).to be >= b`

› First Command
  – Confusion about the order of arguments in the first version
  – Higher cognitive load of reading twice as many characters
› Second Command (Ruby)
  – shorter and easier to read and understand,
  – will likely be easier to maintain

1. Clarity via conciseness

2. Raise the level of abstraction

- First languages
  - Assembly
- Higher-level programming languages
  - Fortran and COBOL
- Scripting languages like Python and Ruby
  - raised the level of abstraction even higher

1. Clarity via conciseness

2. Raise the level of abstraction

- Example
  - *reflection*
    - **Program can inspect, analyze, and modify itself.**
    - Ruby allows you to change the functionality of the language itself while running your own code
  - *higher order functions*
    - allows higher-level behaviors to be reused by
    - passing functions as arguments to other functions

Mechanisms to improve productivity

2. Synthesis

- refers to code that is generated automatically rather than created manually

- Example
  - **SWAP(A,B)**
  - puts the contents of variable A into variable B and vice versa

  - Rails framework makes extensive use of the Ruby language's
  - facilities for ***metaprogramming***
    allows Ruby programs to automatically synthesize code at runtime

offers us to write code that dynamically writes other code for us

Mechanisms to improve productivity

3. Reuse

- Using code from past, rather than writing from scratch
- software is even more likely than hardware to reuse

- Examples
  - **Structured Programming**
  - Procedures and functions
  - Standardized libraries for input/output and for mathematical functions

  - **OOP**
  - Inheritance
  - Design patterns

Mechanisms to improve productivity

3. Reuse

Dry = Reuse

- Ruby and JavaScript
  - › typical of modern scripting languages
    - automatic memory management
    - dynamic typing
    - support for higher-order functions
    - and various mechanisms for code reuse..

    - Ruby
    - supporting multiple programming paradigms
    - such as object-oriented and ***functional programming***

Mechanisms to improve productivity

4. Automation

- replacing tedious manual tasks with tools
  - to save time, improve accuracy, or both

- Examples
  - **Git**
  - Version control systems
  - **Cucumber**
  - helps automate turning user stories into acceptance tests
  - **Pivotal Tracker**
  - automatically measures Velocity, which is a measure of the rate of adding features to an application
  - **Rspec**
  - helps automate the unit testing process.

SaaS and Service Oriented Architecture

- Idea
  - Do not install software on user computer
  - run the software centrally on Internet-based servers,
    - and allow users to access it via a Web browser
- Salesforce
  - the first large company to fully embrace this new model
  - which was dubbed Software as a Service (SaaS)
- SaaS is popular in everyday use
  - searching, social networking, and watching videos

SaaS

› advantages for both users and developers

1. No installation:
   - Customer don't have to worry whether their hardware is the right brand or fast enough
   - Not dependency on OS
2. Data is kept with the service
   - No need to backup
3. Group level use
   - User interact
4. Centralization
   - data is large
   - data updated frequently
5. Only a single copy of the server software
   - runs in a uniform
   - User do not need upgrade
6. Cheaper service

SaaS

SaaS Programming Framework	*Programming Language*	*Introduced*
Active Server Pages (ASP.NET)	C#, VB.NET	1996
Enterprise Java Beans (EJB)	Java	1997
JavaServer Pages (JSP)	Java	1999
Spring	Java	2002
Rails	Ruby	2004
Django	Python	2005
Zend	PHP	2006
Sinatra	Ruby	2007

- Rails has embraced the Agile lifecycle
  - use the right tool for the job, even if it means learning a new tool or new language!

SaaS and Agile

- Agile fitness with SaaS
  - frequent upgrades of SaaS perfectly align with the Agile software lifecycle
    - Hence, Amazon, eBay, Facebook, Google, Microsoft and other SaaS providers all rely on the Agile lifecycle
    - The Agile process is an excellent match to the fast-changing nature of SaaS applications.
  - One pitfall
    - developers could not make extensive use of software third-party *libraries*
    - containing code to perform tasks common to many different applications. Because these libraries were often written by others
    - Missing the extensive access to hardware

SaaS and SOA

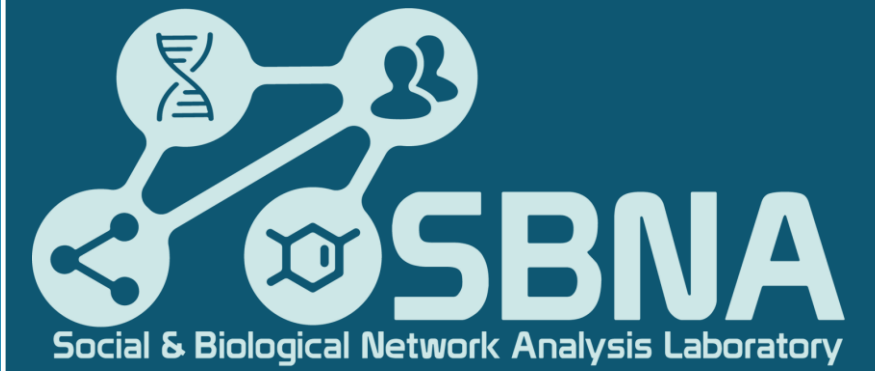
- SaaS needs using services built and maintained by other developers for common tasks

- *service-oriented architecture* (SOA)
  - in which a SaaS service could call upon other services

- Services that were highly specialized to a narrow range of tasks
  - *microservices*
  - Examples
    - credit card processing, search, driving directions
  -

Conclusion

- Higher and new programming languages and technologies
  - Speed up the software development process
- Currently SaaS is the dominant form of software use



Question?

Bioinformation.ir

info@Bioinformation.ir