

Distributed Development

Software Development Follows a Process

- What is this process ?
 - SDLC
- The coding aspect of the development is implemented following what ?
 - A Methodology
- How many methodologies are there ?
 - How high can you count ?

Okay we have the basics

- Typical methodology for AGILE software development ?
 - XP, RAD, RUP
- Methodologies important for you ?
 - SCRUM, LeSS, SAFe (maybe KanBan)

Elements covered so far

- Requirements Analysis consists of
 - User stories (functional requirement, non-functional requirements)
- A system's design can be constructed using
 - UML, Flowcharts, DFDs, Descriptions and scribbles
- Development process follows...
 - A methodology

Not all software is created equal

- Some software projects are large and complex and others are less so
- What are the implications of this ?
 - Larger teams
 - Multiple technologies
 - Multiple languages
 - Multiple interfaces

With regard to scrum

- What effect does a larger team have on development practice ?
- What effect does the use of multiple technologies have ?
- Does scrum handle the use of multiple languages well ?
- What type of software development is scrum particularly suited to ?

Small software projects and scrum

- Less issues with respect to identification of requirements
- Relatively less complex division of tasks
- Management of tasks become less difficulty
- Easier to monitor progress of a team

Large software projects and scrum

- Scaling of small teams to large projects
- Adaptation of scrum to form LeSS, SAFe
- Greater collaboration and communication required
- Possibly more than one developer working on a given task

So what have we learned from this ?

- Scrum is excellent at task division and providing responsibility
- Scrum is excellent for collaboration and involves the user (the end user may not be immediately obvious)
- Sometimes the management style provided by scrum can be a problem, particularly so with complex development tasks.

We have a neat development team

- How productive is this team ?
- How efficient is this team ?
- Is everyone doing their job ?
- How would we measure this ?

How to measure ?

- Start with what exactly you would like to know
- What is useful to the management style
- What is useful to the development
- What is useful to the end user

Larger teams means what ?

- Difficult to maintain code ?
- Communication is mandatory (Airbus)
- Many different developers editing code for a project means what ?
- How do we facilitate this ?

Which is more efficient ?

Team A



Team B



Consider the code contributed

- Team A

- Contributed 2000 lines of code
- 50 man hours writing code
- Three different features of the project contributed to

- Team B

- Contributed 200 lines of code
- 120 man hours editing code
- 1 feature of project contributed to

Not all is as it appears

- Just because a large amount of code was written does this mean that the team is better ?
- Just because more hours were worked does this make the team more productive ?
- What are our metrics ?

Metrics

- Time taken
 - Code contributions
 - Code commits
 - Features worked upon
 - Finger pointing
 - Ability to recover from erroneous code entry
-
- Entirely depends upon what is seen as important

How do we solve some of these issues ?

- We use version control
- What is version control ?

Examples of version control

- CVS – Concurrent Versioning System
 - Client-server based architecture
- SVN – Subversion
 - Client-Server based architecture
- Git –
 - Peer based architecture
- Mercurial –
 - Peer based architecture , multi-platform

So what does version control software do ?

- What is the primary function of version control ?
- What features does it afford development teams ?
- What processes do the features allow management to perform ?

How do these features help us ?

- They can elucidate the contributions to a feature by a developer
- Can aid in the identification of “slackers”
- Show the efficiency of team over time
- Can show the user/client statistics relevant to them

There are so many version control systems

- Similar to methodologies
- Which do we choose ?
 - Usually governed by the organisation

Subversion

- What specific features does SVN provide ?
- What means does it provide for identification of developers ?
- What features does it provide for recovery of code ?
- How does SVN represent the information that it contains ?
- What type of tools are there to facilitate this ?

Security

- What type of security is involved in accessing the details of a project within SVN ?
- What are the implications of this security ?