Software Architecture

Lecture 13
Agile/Scrum/Project

Project Story

- Personally missing open bank project
- Something to base other projects on
- Concrete: demonstrating biometric authentication → no chance at “real” banks

Project

- We are building
  - a open source bank (transaction) server
  - some bank open source clients (Android + maybe Web)
- More specifically
  - transaction ledger
  - with corresponding clients
  - as much security as we can manage.

Initial requirements (will change)

- Tons of good scenarios (of course!)
  - Let's be able to play through the product early
  - Wireframes/Mockups where necessaruy
- Great architecture (but embracing change)
  - In addition to storyboards, object, and class diagrams...
  - … 1-2 component (or deployment/communication) diagrams, several sequence diagrams
- Use SDMLib and Story Driven Modeling
- Extensive tests
- Open Source development on github

More requirements

- It needs to be possible...
  - to create accounts from some kind of admin console
  - from the admin console to create and modify any transactions (maybe with a trace)
  - to make a seed transaction to create money
  - to make a transfer from admin console
  - to use mobile phones to transfer some virtual money (an android app for us)
  - transaction persistence
  - have an account snapshot cache to allow rollbacks
  - to survive crashes (have serial transaction log, consistent with snapshot)

Required parts

- Persistence layer/transaction log (very object based) + (secure) network interface (don’t use database in beginning)
- Security role model (who can do which transactions)/Account management
- Admin (several admin types?) console
- Mobile client which works fast in a p2p (two persons can exchange money as easy as cash – if they are connected to the Internet/database)

Not required, but maybe future addition

- Database adapter, let’s use in-memory object structure and files first
- Interest, money lending
- iOS client
- NFC transfer
- Blockchain support (but consider when designing architecture)
- Biometric authentication

Grading (to be refined)

- Git activity: 10 (personal for two phases: approx 700 lines of code, less for dedicated roles)
- Performance evaluation: 10
  - by team members and scrum maste, maybe in sprint reviews
- Sprint-presentations: 10
- Mutual review: 10 (maybe into performance eval)

Agile

- Watch movie and take notes
- Be able to discuss
  - What is the advantage of “Agile”?
  - What are the common missperceptions?
  - How does it fit with SDM?
- Post to slack #agile

How to develop in a team?

- Divide and conquer!
  - How do I know how to devide?
- Have some structure – flat hierarchies?
- Have team ethics
- Minimize bureaucracy, maximize programming!

Scrum

- Watch movies
- Take notes
  - What are main steps of Scrum development?
  - What are main roles?
  - What are main artifacts?
  - What does fit and not fit with SDM?
- Post to #scrum

Our Development Principles

- Discuss in current teams important guidelines for our project development
  - Use elements from SDM, Kanban, Scrum
  - Compile ten most important points
    - Standup is required (create strict late policy)
    - 2.5 class hours (x3) → approximately 7.5 extra hours of coding/management per week ← how do you guarantee and check this?
  - Post this to #sdm-scrum

Vote for team

- 2 competing teams (there will be approx two percentage points given to best team and not to weak team)
- Teams will be (kind of) random
- Send email to me (gmu address) with title "sa team vote"
  - Vote for one partner (if partner votes for you, you will end up in same team), only one vote partner vote allowed
  - Write two paragraph application for role (scrum master, product owner, software architect, developer)

Discussion

- Quickly check a definition of Kanban
- Compile list of our own (agile, scrum based rules) in current teams
- Post