

**CS3215 – Software Engineering Project**

17 April 2010

**CLAIM SHEET**

**Team 12**

# **Accomplishment**

* A quality product with not only basic features (basic sync: one way/two way) but also unique features: Right-click features (Smart Sync, Sync With), multiple jobs sync, drag-and-drop sync, IVLE Sync, many more minor features.
* A fault-tolerant, user-oriented, intuitive, simple, convenient, fewer-step GUI.
* Sync large files in a reasonable amount of time.
* A portable software without window installation.
* Extensible system design for future expansion and development. (The Logic can be built into .dll file to be used for different GUI and Storage, etc)

# **Applied techniques from CS2103/CS3215 Lectures**

* Separation of concerns principle: to design loose coupling and cohesive components. GUI, Logic and Storage can be implemented in parallel.
* Design patterns
  + MVC Pattern + Multilayer Architecture: GUI, Logic, Storage
  + Singleton Pattern: Storage
  + Façade Pattern: GUI 🡪 LogicFacade 🡪 Logic 🡪 StorageFacade 🡪 Storage
  + Observer Pattern: Logic notifies GUI whenever necessary
* Design approach: Design by contract + top down design
* Implementation:
  + Inheritance and Polymorphism
  + Substitutability principle
  + Aggressive refactoring
* Testing:
  + Exploratory Testing
  + Unit Testing + Integrated Testing + Regression Testing (using NUnit)
  + Test case design: grey box, equivalence partitioning, code-coverage
* Project management:
  + Process model: Spiral
  + Work breakdown structure

# **Apply programming techniques:**

* Multithreading
* Inter-process communication
* Context Menu Handler (right-click feature)
* Web Programming (IVLE)
* Windows Presentation Foundation (design GUI)

# **Challenges during project**

* *Problem:* Time constraint and pressure - team members had to deal wit pressure this module as well as other modules.

*Solve:* Team member had to pay extra efforts to fulfil the tasks. Sometimes we needed to work overnight to meet the time constraint.

* *Problem:* kita for the proposal. Quite a disappointment and discouragement.

*Solve:* Learnt from “failure”– worked harder for the rest of the project. In the end, our effort was paid off with NETS ☺

* *Problem*: Team lacked GUI design experience. Our WPF GUI was not finished in time for the release of version 2.0.

*Solve:* team decided to focus on developing more unique features instead of trying to design as flashy GUI as the selling point.

* *Problem:* Task division: most members wanted to do a same task (coding).

*Solve:* two members was chosen to be main coders, and others was assigned different tasks (testing, writing documentation, etc)

* *Problem:* different ideas and preference conflicts between different members

*Solve:* Analyze plus and minus points of each preferences to come up with the final decision.

*…*. and as many other problems as a coding team project may face ☺

# **Lessons and experience learned**

* Teamwork: Team bonding, team spirit and enthusiasm are factors of great importance to lead to success.
* Project Planning and Management:
  + A good plan and timing facilitates the later work of the project.
  + Time management: good time management leads to less stress and better productivity.
  + Meeting management:
    - have regular (weekly) team meeting sessions helps team members keep better track with the project, rather than rushing to meet only before the deadline to do all the work.
    - Meeting plan: have specific plan for each meeting (what is to be discuss, what needs to be done after this meeting) helps improving productivity of the meeting session.
    - Extra meeting sessions needed in emergent situations (e.g. new bugs detected just before deadline, etc.)
  + Toward-the-end task: All members in the team should focus on developing and polishing the software instead of trying to explore new things.
* Consultation from project advisors: is needed and is an important factor, helps the project progresses smoothly.
* Other lessons:
  + Spend more time on documentation besides coding and testing.
  + You can never build a bug-free software.

# **Usage of new tools**

* Visual Studio Team System 2008
  + Debugger
  + Refactor tool
  + Code metric calculator
  + NUnit Framework for Testing
  + Profiler
  + Ankh SVN
* Tortoise SVN
* Resharper
* DotTrace
* File Bot + FolderDiff
* DreamWeaver + Microsoft Expression Web + Photoshop CS4 to design Web

# **Ebook and resources**

* Resources provided by CS3215 Teaching Team
* Design Patterns: Elements of Reusable Object-Oriented Software (Gang of Four)
* Threading in C# (Joseph Albahari)
* C# 2008 (Wei-Meng Lee)
* Programmer Heaven C# School (Faraz Rasheed)
* Creating Context Menu Handlers: http://msdn.microsoft.com/en-us/library/cc144171%28VS.85%29.aspx