Project Plan

Biomembrane Team - Nick Klose, Richard Leung, Cameron Mann, Glen Nelson, Omar Qadri, James Wang

Introduction

This document provides an overview of our team's project plan for the Biomembrane project of CMPUT 401 at the University of Alberta during the Winter 2013 term.

Preliminary Requirements

Consult the Requirements Document for information.

Risks

Technical

- Some group members are not familiar with Python or Matlab, so may require support from other group members who are knowledgeable on the subjects.
- There may not be libraries available for multidimensional arrays and visualizations.
- Github may become temporarily unavailable.

Personnel

- Numerical functions may be difficult for our team to code due to lack of background knowledge and/or experience.
- A member of the group may try to sabotage the project.
- A member of the group may drop CMPUT 401 or withdraw from university.
- A member of the group may become ill or otherwise unable to fulfil their duties.
- A member of the group may be too busy with other courses, or unwilling to dedicate the time necessary to adequately complete the project.
- A member of the group may lack background knowledge and require training such that they hinder the group rather than helping.

Schedule

- Group members may be too busy with courses to adequately contribute to the projec
- It may be difficult to schedule meetings which all group members, the client, and our TA are all able to attend.

Client

- There may be difficulties in communication between the group members and client.

Other

- The undergraduate lab servers may become unavailable.
- The server hosting the web component of our project may experience failure.

Resources Required

- Adequate time dedicated by group members for meetings, planning, and implementation.
- A server to host the web component of our project.
- Access to the previous version(s) of the software.
- Support from our instructor and TA.

Dependencies

Each sprint will consist of a coding and testing phase for each component. The testing is done by the person who implemented the code and can also be done by other team members as needed. Any member of the development team may review, test, or improve any part of the code at any time.

Development Tasks

The *product backlog*, including features to be implemented, bug fixes, outstanding documentation, or other elements needed for the product will be separated into a series of *sprint backlogs*. Each sprint backlog lasts the duration of one sprint (two weeks) and contains a list of tasks to be completed during the course of that sprint.

Each sprint's tasks fall under one or more of the three product components: backend, local GUI, and web interface.

Risk Assessments

Scope

Risk: we may take on too many different tasks
Result: we may fail to deliver everything we indicated
Potential impact: low, since we would at least deliver the core system
Likelihood: low, since our client has indicated that these additional parts exist but that he would only consider attempting them once the core part is complete
Indicators: new tasks may pile up faster than we can close old ones

Technology

Risk: we may not be able to find libraries for multi-dimensional arrays and visualizations
Result: we may have to write these libraries on our own

Potential impact: medium, since we may not have enough time to create these libraries to high quality standards
Likelihood: low, since we are using Python, which is a well-known language for which many libraries are available
Plans: use Python 2.7 instead of the newest version 3.3, since 2.7 is more stable and has a wider range of libraries available
Indicators: few results may show up on google searches and on the Python package index for the type of library we are seeking

Personnel

Risk: we may have difficulty coding the numerical functions used in the original program
Result: we may fail to all the required functionality
Potential impact: high, since these functions are at the core of the analysis performed
Likelihood: high, since none of us has an advanced numerical analysis background
Plans: use Python libraries that implement these numerical functions, such as numpy and scipy
Indicators: we may spend an unreasonable amount of time implementing a function that requires only a few lines of code but a lot of background knowledge

Risk: a member of the group may try to sabotage the project
Result: the system may function incorrectly or not at all
Potential impact: high, since the program may not even be able to run
Likelihood: low, since the project is a large part of our grade, and it is in no group member's best interest to reduce their own grade
Plans: revert back to before the malicious changes, reduce the saboteur's git repository privileges if possible; inform TA if necessary
Indicators: intentional ruining of large parts of previously working code

Schedule

Risk: we may be too busy with other courses to have enough time to work on the project
Result: the project may be incomplete when the deadline comes
Potential impact: high, since the system would not be able to function properly
Likelihood: high, since each of us is taking four other courses in addition to this course
Plans: use time wisely, work on the project when you would otherwise be procrastinating and keep some hours each week free to work on the project.
Indicators: the progress of the project, as measured by the lines of code written or the number of tasks completed, may stall

Client

Risk: we may have difficulty communicating with the client

Result: we may deliver something other that what the client wanted
Potential impact: high, since the client would not be satisfied with the system
Likelihood: high, since the client is going to Europe in March, so we will not be able to meet him face-to-face
Plans: use Skype and the mailing list to communicate with the client while he is away
Indicators: a sense of vagueness may remain about certain sections of the specification, with no clarification as time progresses

Other

Risk: the servers running the computer labs may fail or be scheduled for maintenance
Result: our files on the lab machines may be lost or we may not be able to use the lab machines for development for some time, resulting in delays to the project schedule
Potential impact: low, since it is convenient but not necessary to work on the project in the labs
Likelihood: medium, since the servers are well-maintained by the system administrators, which means less chance for failure but more chance for maintenance downtime
Plans: in case of such events, we may pull the code from the Github repository onto our own computers and develop from there, so as not to lose any time
Indicators: a message may be sent from helpdesk to our emails regarding the date and time of any scheduled maintenance

Risk: the server to which we deploy the backend of the web component of our system may fail
Result: the web interface may be unusable during the period of time in which the server is down, resulting in loss of access for clients who don't have a local copy of the system
Potential impact: high, since the remote clients would not be able to get their analysis done
Likelihood: low if the server is properly deployed and well-maintained, otherwise medium
Plans: have a backup server if possible, and route requests to that server if the main one fails
Indicators: the web browser may show a timeout error or a server error or a page-not-found error when trying to access the web interface

Project Macro-Structure

Overview

This project follows the Scrum project macro-structure. As such, this section borrows content from the Scrum Guide.

- No daily stand-up meetings due to time constraints.
    - Replaced by a weekly meeting, lasting one hour, and frequent correspondence through GitHub, Google Drive, and the project mailing list.
    - Using the above, the team should keep track of what is being done each day and

what must be done by the next day, as well as any obstacles.
- Transparency: our code, wiki, and issue tracker on GitHub are available to those invested in the project.
- Inspection: each component is reviewed by another team member and will be inspected by Dr. Eleni Stroulia and Dr. Nils Petersen.
- Opportunities for inspection and adaptation:
    - Sprint planning meeting: every two weeks, with time set aside to fix issues which are not yet known but will possibly arise later.
    - Sprint review
        - Held at the end of a sprint to inspect the Increment and adapt the backlog.
        - Overview of what is done and not done.
        - Product Owner discusses the state of the backlog and projects likely completion dates.
    - Sprint retrospective
        - Create a plan for improvements to be enacted during the next Sprint.
        - Inspect how the previous sprint went regarding people, process, tools.
        - Identify major items that went well
        - Plan improvements to the Scrum Team for the following Sprint

Scrum Team

Our team will be primarily self-organizing rather than being directed by others outside the team. Some things to focus on are *flexibility, creativity, productivity,* and *maximizing opportunities for feedback*.

Product Owner (Glen):

- Responsible for maximizing the value of the product and the work of the dev team.
- Accountable for the product backlog.
- In this project, is also a member of the development team.

Development Team:

- Converts the product backlog into potentially releasable functionality.
- Each team member does not have any additional titles other than Developer. However, every member of the development team is responsible for either one or two of the three product components (backend, web interface, and local GUI).
    - This is to enforce that while team members will spend most of their time working on one or two of the project components, they are also welcome to review, discuss, test, and implement components to which they are not officially assigned. Using this model will help ensure cohesion between the web interface and local GUI, as well as providing a rudimentary checks-and-balances system within the team.

- Each developer should maintain a personal log, and ensure the Glossary is kept up-to-date with terminology used.

Scrum Master (Nick):

- In this project, is also a member of the development team.
- Responsible for ensuring Scrum is understood and enforced.
- Helps to determine which interactions are helpful and which aren't, then helps to change the interactions to maximize value created by the team.
- Finds ways to manage the product backlog.
- Communicates with the development team regarding goals and backlog.
- Facilitates Scrum events as needed.

Individual Developer Responsibilities

Cameron

- Responsible both for the backend and web interface.
- Other responsibilities:
    - Requirements section of requirements document.
    - Django Models of the web interface.
    - Queue-and-run functionality of the web interface.
    - Batch processing for the backend.
    - Refinement of the web interface, including compatibility testing.

Glen

- Product owner and management lead, responsible for management of the product backlog and sprint backlog.
- Developer responsibilities primarily related to the backend.
- Helps to ensure the team is being productive and efficient (along with the Scrum Master).
- Other responsibilities:
    - Command line interface
    - Acceptance and integration tests
    - Code license
    - Batch processing for the backend
    - Project plan - major phases and milestones

James

- Responsible mainly for the web interface.
- Other responsibilities:
    - Resources section of requirements document
    - Project plan - major phases and milestones

    - Display of results on the web interface.
    - Batch processing for the web interface.
    - Refinement of the web interface, including compatibility testing.

Nick

- Scrum master, responsible for ensuring the team is being productive and efficient and following the Scrum methodology.
- Primarily responsible for the local GUI.
- Works throughout the project to keep documentation up-to-date.
- Creates burn-down charts and other project monitoring tools for each sprint.
- Other responsibilities:
    - Storyboard section of requirements document
    - Project plan - introduction and project macrostructure
    - Creating the basic GUI for the local interface, and calling the backend with it.
    - Batch processing for the local GUI.
    - Refinement of the local GUI.

Omar

- Technical lead developer for the backend, including profiling, optimization, refactoring, and Python conversion.
- Also responsible for the local GUI.
- Other responsibilities:
    - Backend unit tests
    - Backend image format support, including channel-separated images
    - Batch processing for the local GUI.

Richard

- Responsible for both the local GUI and web interface, and ensuring consistency between them.
- Other responsibilities:
    - Overview for requirements document.
    - Web interface upload form.
    - Loading images into the local GUI.
    - Displaying results on the local GUI.
    - Batch processing for the web interface.
    - Refinement of the web interface, including compatibility testing.

Sprints

- Two weeks in length.
- Working prototype delivered at the end of each sprint.
- Only at the beginning of a new sprint can the sprint goal be modified.

- After each sprint, the scope is clarified and re-negotiated.
- The Product Owner (Glen) may cancel a sprint in progress if the sprint goal becomes obsolete.
    - For example, if the project goal changes or market or technology conditions change.
    - If a sprint is canceled, any completed backlog items are reviewed to see if any of them are releasable. Incomplete backlog items are re-estimated and put back in the backlog.

Sprint Planning Meetings

- Held at the beginning of each sprint to determine the sprint goal and plan.
- Can last between one and four hours for our project.
- First half of the meeting:
    - "What will be delivered in the Increment resulting from the upcoming sprint?"
    - Forecast functionality developed during the sprint.
- Second half of the meeting:
    - "How will the work needed to deliver the increment be achieved?"
    - Sprint Backlog is created/reviewed.

Product Backlog

- Ordered list of requirements and the sole source of requirements for changes.
- Product Owner (Glen) is responsible for its content, availability, and ordering.
- Backlog is never complete; it evolves as the product and environment evolve.
- Dynamic and constantly changing, and exists as long as a product exists.
- Lists all features, functions, requirements, enhancements, and fixes.
- Consists of descriptions, order, and estimates.
- Higher-ordered backlog items are clear, detailed, and important.

Sprint Backlog

- Subset of Product Backlog items in a given sprint.
- Has enough detail that daily progress can be tracked using it.

Project Monitoring

The Scrum Master (Nick) will be primarily responsible for collecting and reporting data related to project monitoring. This will include:

- Burn-down charts to show expected versus actual progress with the sprint and product backlogs.
- Charts depicting velocity overall and across sprints.
- Difference tables showing which elements of sprint backlogs were completed and deferred.

Major Phases and Milestones

There are three sets of tasks within each sprint: Backend, Web, and GUI. Backend tasks refer to tasks relating to the actual computational model (“ICS”). Web tasks are tasks to support a Web Page frontend, allowing the program to be deployed as a service. Finally, GUI tasks focus on providing a local GUI allowing for creation of the data.

Each task should include the time to: develop, unit test, document, and deploy. Additional testing will be done in unique tasks. All changes are to be reviewed, including documents. These reviews aren’t tasks. Tasks with no owners will have to be covered by the team. Finally, the creation of documentation for the 401 course will be considered as tasks. A Gantt chart will be created for the purposed task structure.

All dates are in DD/MM/YY. Much of the first Sprint is untracked, as it includes time to get used to tools, etc, and it wasn’t clear that time should be closely monitored.

Sprint	**Planned/**	**Estimated**		**Actual**		
	Start Date	End Date	Effort (person hours)	Start Date	End Date	Effort (person hours)
Prototype	25/01/13	08/02/13	32	25/01/13	08/02/13	Untracked
Webpage	08/02/13	22/02/13	30	08/02/13	22/02/13	TBD
Batch	22/02/13	08/03/13	30	22/02/13	08/03/13	TBD
Profiling	08/03/13	22/03/13	34	08/03/13	22/03/13	TBD
Completion	22/03/13	01/04/13	20	22/03/13	05/01/13	TBD
Docs and Demo	01/04/13	08/04/13	17	01/04/13	08/04/13	TBD

Prototype Sprint

The first Sprint “Prototype” focuses on creating an initial prototype in this time. The Monday after this sprint, two documents are due.

Task	**Start Date**	**End Date**	**Story Points**	**Assigned to**
Backend - Python Conversion - Non Triple Correlation	25/01/13	08/02/13	3	Omar

Backend - Python Conversion - Triple Correlation	25/01/13	08/02/13	3	Omar
Backend - Command Line Interface	25/01/13	05/02/13	0.5	Glen
Acceptance Tests - Part 1	04/02/13	08/02/13	5	Glen
Requirements Document - Resources	25/02/13	08/02/13	1	James
Requirements Document - Glossary	25/02/13	08/02/13	0.5	Everyone
Requirements Document - IP License	25/02/13	08/02/13	1	Glen
Requirements Document - Storyboard	25/02/13	08/02/13	3	Nick
Requirements Document - Requirements	25/02/13	08/02/13	1	Cameron
Requirements Document - Overview	25/02/13	08/02/13	3	Richard
Project Plan - Personal logs	25/02/13	08/02/13	0.5	Everyone
Project Plan - Major Phases / Milestones	25/02/13	08/02/13	3	Glen, James
Project Plan - Project Macrostructure	25/02/13	08/02/13	3	Nick
Project Plan - Risk Assessment	25/02/13	08/02/13	1	Omar
Project Plan - Introduction	25/02/13	08/02/13	1	Nick

Total Points in this sprint: 29.5
Team Capacity: 36 points (6 points per team member)
Overbudget: Omar (8 points), Glen (9 points), Nick (8 Points)
Underbudget: James (3.5), Cameron (2 points), Richard (4 points)

Webpage Sprint

The core of this sprint is setting up the web service aspect of the project. The monday after this sprint, the Requirements review document is due.

Task	**Start Date**	**End Date**	**Story Points**	**Assigned to**

Web - Django Models	08/02/13	13/02/13	3	Cameron
Web - Upload Form	13/02/13	22/02/13	1	Richard
Web - Queue and Run task	13/02/13	18/02/13	3	Cameron
Web - Display Results	16/02/13	22/02/13	3	James
Backend - Refactor Initial Prototype / Proof of Concept	08/02/13	11/02/13	0.5	Omar
Backend - Add Unit Tests	11/02/13	22/02/13	3	Omar, Glen
Backend - Add Image Formats	11/02/13	18/02/13	3	Omar, Glen
Backend - Add support for channel separated images	11/02/13	22/02/13	0.5	Cameron
Local GUI - Load images	13/02/13	15/02/13	1	Richard
Local GUI - Basic gui in place	08/02/13	13/02/13	3	Nick
Local GUI - Call the backend	13/02/13	22/02/13	1	Nick
Local GUI - Display results	15/02/13	22/02/13	1	Richard
Integration Tests Part 1	15/02/13	22/02/13	3	Glen
Requirements Technical Review - Findings Report	20/02/13	25/02/13	5	All team members
Requirements Technical Review - 25 minute meetings	20/02/13	25/02/13	1	All team members
Update Documents	22/02/13	22/02/13	1	Nick

Total Points in this sprint: 33
Team Capacity: 34 points (5-6 points per team member)
Overbudget: Cameron (7.5), Glen (7)
On Budget: Nick(6)
Underbudget: Richard (4), James (4), Omar (4.5)

Batch Sprint

The core of this sprint is adding in batch processing. Additional breakdown of the tasks of this sprint is required.

Task	**Start Date**	**End Date**	**Story Points**	**Assigned to**

Backend - Batch processing	22/02/13	08/03/13	8	Glen, Cameron
Web - Batch Processing View	22/02/13	08/03/13	8	James, Richard
GUI - Batch processing	22/02/13	08/03/13	8	Omar, Nick
Acceptance Tests Part 2	22/02/13	08/03/13	5	Glen
Requirements Technical Review - Repair	25/02/13	01/03/13	3	Depends on Repair work
Update Documents	08/03/13	08/03/13	1	Nick

Total Points in this sprint: 33
Team Capacity: 34 points (5-6 points per team member)
Overbudget: Glen (9)
On Budget: Nick (5)
Underbudget: Omar (4), James(4), Cameron (4), Richard (4)

Profiling Sprint

The core of this sprint is profiling and optimizing the code. This sprint is vague, due to the cone of uncertainty.

Task	**Start Date**	**End Date**	**Story Points**	**Assigned to**
Backend - Profile	08/03/13	22/03/13	5	Omar
Web - Better Batch upload methods	08/03/13	22/03/13	5	James
GUI - Refinement	08/03/13	22/03/13	5	Nick
Web - Refinement	08/03/13	22/03/13	13	Cameron, James, Richard
Final Integration Tests	08/03/13	22/03/13	8	Glen
Update Documents	22/03/13	22/03/13	1	Nick

Total Points in this sprint: 37
Team Capacity: 38 points (6 points per team member)
Overbudget: Glen (8), James(9.33)
On Budget: Nick(6)
Underbudget: Omar (5), Cameron (4.33), Richard (4.33)

Completion Sprint

This sprint is the rush to completion. This sprint is cut short due to the due date. The Code, the Acceptance Tests, and the User Manual are due on this date.

Task	Start Date	End Date	Story Points	Assigned to
Backend - Complete	22/03/13	25/03/13	3	Whoever has work left
Web - Complete	22/03/13	25/03/13	3	Whoever has work left
GUI - Complete	22/03/13	25/03/13	3	Whoever has work left
Acceptance Test Document - Introduction	25/03/13	01/04/13	1	Glen
Acceptance Test Document - Installation Document	25/03/13	01/04/13	3	Glen
Acceptance Test Document - System Tests	25/03/13	01/04/13	3	Glen
Acceptance Test Document - Support Contact	25/03/13	01/04/13	1	Glen
User Manual - Online Manual	22/03/13	01/04/13	3	Omar, Richard, Nick
User Manual - Offline Manual	22/03/13	01/04/13	3	Richard, Cameron, James
Presentation - 3 to 5 minute demo video	22/03/13	01/04/13	1	Cameron
Presentation - Product Overview	22/03/13	01/04/13	0.5	Richard
Presentation - Example User Scenarios	22/03/13	01/04/13	0.5	Nick
Presentation - Chosen Architecture	22/03/13	01/04/13	0.5	James
Presentation - Implementation Challenges	22/03/13	01/04/13	0.5	James

Presentation - Management Challenges	22/03/13	01/04/13	0.5	Omar
Presentation - Lesson Learned	22/03/13	01/04/13	0.5	Omar
Future Work Document - Deferred Features	22/03/13	01/04/13	0.5	Omar
Future Work Document - Dropped Features	22/03/13	01/04/13	0.5	James
Future Work Document - Non Functionality (if any)	22/03/13	01/04/13	1	Nick
Future Work Document - Lessons Learned Section	22/03/13	01/04/13	0.5	Cameron

Total Points in this sprint: 29.5
Team Capacity: 23 points (4-5 points a sprint per team member)
Overbudget: Glen (8)
Underbudget: Cameron (2.5), Nick (2.5), James (2.5), Omar (2.5), Richard (2.5)

Docs and Demo Sprint

This sprint is for final documentation considerations, and preparing the demo. The complete documentation is due with this sprint.

Task	Start Date	End Date	Story Points	Assigned to
Architecture document - software subsystems	01/04/13	08/04/13	3	Richard, Cameron
Architecture document - logical model	01/04/13	08/04/13	5	Omar, Nick, James
Architecture document - dynamic behaviors	01/04/13	08/04/13	0.5	Glen

Total Points in this sprint: 8.5
Team Capacity: 19 points (3-4 points a sprint per team member)

Personal Logs

The personal logs of each team member are available on the project wiki.

1. Nick Klose: https://github.com/UniversityOfAlberta/BioMembrane/wiki/Nick's-Diary
2. Cameron Mann:

https://github.com/UniversityOfAlberta/BioMembrane/wiki/Cameron's-Diary

3. Glen Nelson: https://github.com/UniversityOfAlberta/BioMembrane/wiki/Glen's-Diary
4. Omar Qadri: https://github.com/UniversityOfAlberta/BioMembrane/wiki/Omar's-Diary
5. James Wang: https://github.com/UniversityOfAlberta/BioMembrane/wiki/James'-Diary