DNA to Feature Models

Design Document

Senior Design Team 22
Professor Myra Cohen
Abdul Rahman Moughrabi: Developer/Documentation Management
Ahmad Nazar: Team Leader/Developer
Ahmed Alketbi: Developer/Debugger
Hyegeun Gug: Developer/Web Management
Prathik Nair: Debugger/Developer

Team Email
http://sddec20-22.sd.ece.iastate.edu/

Revised: 02.23.2020/Version1

Executive Summary

Development Standards & Practices Used

Following a set of standards ensures development of a product that is safe, adheres to the consumer preferences and expectations; while also ensuring a reliable, and organized workflow for the engineers and the consumer. The standards used in this engineering standards used in this project follow the guidelines of:

- IEEE Engineering Standards
- IEEE Software Engineering Standards

Summary of Requirements

- Biobrick repository
- Extending plugin to support bio bricks
- Web crawling
- Software product line engineering
- Translation of features to be compatible with Feature IDE
- Creating a system architecture

Applicable Courses from Iowa State University Curriculum

- Com S 228: Introduction to Data Structures
- Com S 309: Software Development Practices
- Com S 311: Design and Analysis of Algorithms
- CPR E 308: Introduction to Operating Systems
- E E 230: Electronic Circuits and Systems

New Skills/Knowledge acquired that was not taught in courses

- Background on BioBrick parts that are used in biological living cells building.
- Feature Modeling Concept and application
- FeatureIDE Eclipse Plugin
- Effective Team Coordination
- Effective Client Communication

Table of Contents

List of figures/tables/symbols/definitions

- **BioBrick Parts:** a standard for interchangeable parts, developed with a view to building biological systems in living cells.
- **Software Product Line:** Software engineering methods, tools and techniques for creating a collection of similar software systems from a shared set of software assets using a common means of production
- **Feature Model:** a compact representation of all the products of the Software Product Line in terms of features.
- **FeatureIDE:** an Eclipse-based IDE that supports all phases of feature-oriented software development for the development of SPLs: domain analysis, domain design, domain implementation, requirements analysis, software generation, and quality assurance. Different SPL implementation techniques are integrated such as feature-oriented programming (FOP), aspect-oriented programming (AOP), preprocessors, and plug-ins.

1 Introduction

1.1 Acknowledgement

We would like to thank Dr.Myra Cohen (Iowa State University), and Mikaela Cashman (Iowa State University) for providing the technical knowledge and guidance needed to be successful in the completion of our project. We would also like to thank our course supervisors, and anyone else who has provided mentorship during the course of our project.

1.2 Problem and Project Statement

Feature modeling is an organization tool that allows an engineer to represent features in a tree of hierarchies. It is a unique and efficient way of modeling feature rich systems.

In our instance, an open source repository called BioBricks allows scientists to view various DNA models. While this tool is useful, it does not implement the feature model organization method; which could reveal new things about these DNA models that one could not see before.

Our goal over the course of one year is to create an eclipse plugin that creates feature models based on existing models found in an open-source repository called BioBricks. A successful implementation of this could allow biologists and scientists to view various models from BioBricks in an organized hierarchy.

1.3 Operational Environment

Our Project is software-based and uses Java and the FeatureIDE plugin for Eclipse .

1.4 Requirements

Access to the Biobrick repository with the extension plugin of Feature IDE. Through web crawling and scraping, gathered information is used to translate information and make it compatible with Feature IDE. This provides users to build software product lines of DNA Feature Models.

1.5 Intended Users and Uses

– The main intended users are biologists who build biological models of living organisms with specific desired properties . However, this project helps anyone who wants to try building feature models of dna without any restriction.

1.6 Assumptions and Limitations

Assumptions:

- Users with and without knowledge of feature models can build feature models of DNA.
- The end product provides access and can used anywhere with internet access to Biobrick repository

Limitations:

- The Biobricks Repository is the main source of information and users need internet access anytime they want to use the plugin.

1.7 Expected End Product and Deliverables

An expected end product is a FeatureIDE plugin that uses parts extracted from the BioBricks Repository.. The plugin includes up-to-date BioBrick parts organized in a feature modeling plugin in an organized categories with informative description for each part allowing users to construct models without the hustle of navigating BioBricks repository. Estimated Delivery Date: December 1st 2020

2. Specifications and Analysis

2.1 Proposed Approach

Our project can be tackled in various techniques and methods to be able to solve it and deliver a high-quality product. One method of approach can be dividing the project into two sections: theoretical and practical and for each section, assign two subsections: architecture and scope. By doing so, we can begin with the theoretical segment in which we understand every little aspect of the project enactment (theoretically) and composing the scope of the theory. After that, having a conceptual insight and obtaining all architectural designs will help ease the design application and the practical section. Another approach I deem vital, and best is approaching this project as a project manager working on a software application for a company. Our project is entirely coding and software design, so this method will be unparalleled. Devising such a mechanism will help produce an ideal product. Utilizing this method will commence several documents to aid in beginning the project: a business case, statement of requirements, a project timeline, risk assessment and mitigation, budget, and lastly, a communication plan. The last method of approach will be an agile approach. This approach will be promptly coordinated, vigorous, and nimble and adapting appropriately to varying settings. This will provide a fast-moving solution to the client, running editions of the software are consistently supplied and improved, and ultimately a

developmental team who cooperates throughout the advancement of the software procedure and are on similar stances. These methods of approach all follow IEEE standards with designing a software project and the standards regarding joint project work. So far, my team and I began with researching and analyzing several papers concerning everything we ought to know to be capable of compiling an architectural blueprint of the project and begin development. We started studying various research papers handed out to us from our client: DNA as features: Organic Software Product Lines and Principles of Feature Modeling. To start on the development of our plugin, we need to comprehend software product line engineering, the BioBricks repository, and Feature IDE (an eclipse plugin). The first few weeks began with grasping the core concept of the project by identifying and exploring the various aspects of implementation (mentioned above). In the following weeks, my team and I started developing a project scheme and strategy. We also needed to fuel ourselves with the practice of utilizing the many tools we will be using for this project. Those tools are web crawling, Java/XML programming, and working with Eclipse plugins. One of the team members is working on managing and editing the source code for the eclipse plugins to be able to grasp the idea of how they are implemented and built. The second member started practicing web crawling/ scraping random items from the internet and altering the scraped data into user readable code so we can add it to a plugin. We also began working on different documentation and the project schedule that will set the foundation on what we need to finish first, what we should work on next, and the expected risks with their mitigation.

2.2 Design Analysis

As discussed in the previous section, we went over the different tools necessary to begin our project and exactly how to use them. Those tools are web crawling, Java/XML programming, and working with Eclipse plugins. One of the team members is working on managing and editing the source code for the eclipse plugins to be able to grasp the idea of how they are implemented and built. The second member started practicing web crawling/ scraping random items from the internet and altering the scraped data into user readable code so we can add it to a plugin. We also began working on different documentation and the project schedule that will set the foundation on what we need to finish first, what we should work on next, and the expected risks with their mitigation. We finished a web crawling experiment, editing eclipse plugin code, a project proposal, and exploring the repository. Most of the tools and experiments we tried worked well; we had mostly successes but a single failure. Web crawling was a complicated task, but we managed to write code that scraped a simple, random website making our experiment successful. Another success was understanding and editing the source code for an eclipse plugin, which will help us in the future when we begin working on the project. After successfully scraping the data from the website and reading it, we wanted to translate it into XML code to be able to use it in a plugin of our choice. We had some trouble translating it to correct XML, and that was our only failure. Throughout the testing and experimenting session, we made a couple of observations on things we need to modify and tools we need to take advantage of. One observation was that data scraping would not give us what we want (XML code), and we should change that tool to a SQL server instead. Some recorded thoughts we had were that learning XML will help us a lot during the product, changing from web scraping to an SQL database, and understanding how additions to plugins are made. We also need to continue working on software design and how we need to approach the various elements of the project, so we minimize the failure rate of experiments and not feel overwhelmed.

2.3 Development Process

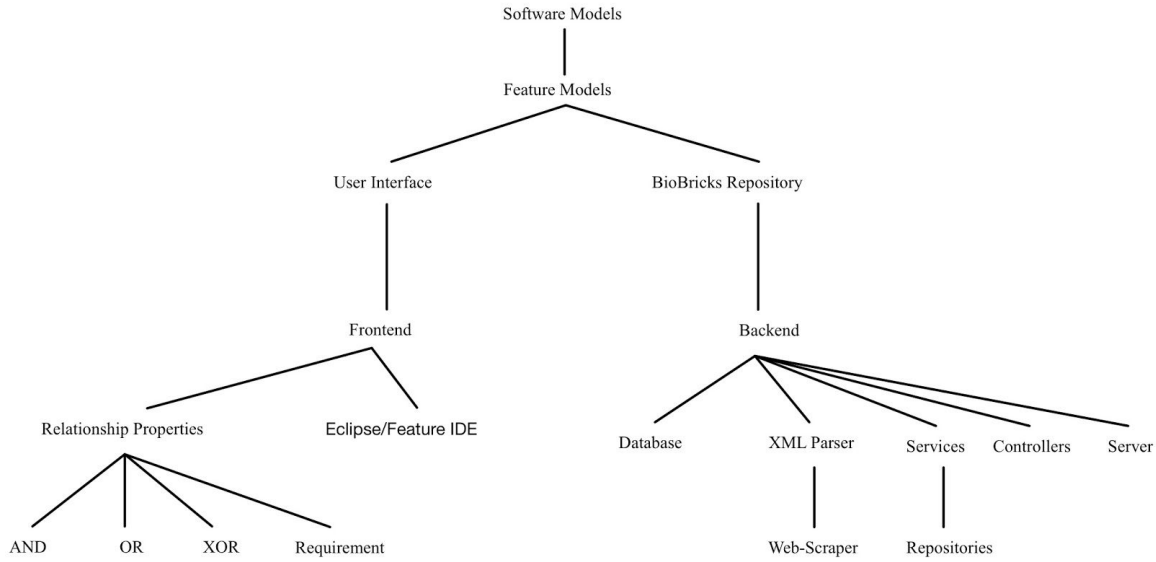
DNA to Feature Models follows Agile software development. Based on the nature of the project, Agile is most suitable due to project requirements and features evolving throughout the process of creation. The project has preliminary, required foundations but the building blocks and the materials built upon the foundations dynamically change with the project.

2.4 Conceptual Sketch

The conceptual sketch of the project is shown in Figure 1. The project involves utilizing Software Product Lines and Feature Models. To present Feature Models that make sense to a given user, a friendly user-interface is required. The user interface is provided through an Integrated Development Environment called FeatureIDE. This section is presented through the frontend aspect of the plugin. The frontend includes all formable relationships as defined by a feature model, and is built-upon Eclipse.

The next section talks about the backend aspect of the project. Parts from the BioBrick Repository will be extracted using a web-scraper and stored in a designated database. This database includes all information relevant parts used in a DNA model. Using the database, creation of models depend on an XML parser which organizes elements of a model according to a user and utilizes all properties of a subset of features with respect to a superset of features.

Figure 1: the flow of project requirements and dependencies. The figure is modelled in a similar to a feature model.



3. Statement of Work

3.1 Previous Work And Literature

Include relevant background/literature review for the project

– If similar products exist in the market, describe what has already been done

– If you are following previous work, cite that and discuss the **advantages/shortcomings**

– Note that while you are not expected to "compete" with other existing products / research groups, you should be able to differentiate your project from what is available

Detail any similar products or research done on this topic previously. Please cite your sources and include them in your references. All figures must be captioned and referenced in your text.

3.2 Technology Considerations

Highlight the strengths, weakness, and trade-offs made in technology available.

Discuss possible solutions and design alternatives

3.3 Task Decomposition

In order to solve the problem at hand, it helps to decompose it into multiple tasks and to understand interdependence among tasks.

3.4 Possible Risks And Risk Management

Include any concerns or details that may slow or hinder your plan as it is now. These may include anything to do with costs, materials, equipment, knowledge of area, accuracy issues, etc.

3.5 Project Proposed Milestones and Evaluation Criteria

What are some key milestones in your proposed project? Consider developing task-wise milestones. What tests will your group perform to confirm it works?

3.6 Project Tracking Procedures

What will your group use to track progress throughout the course of this and next semester?

3.7 Expected Results and Validation

What is the desired outcome?

How will you confirm that your solutions work at a **High level**?

4. Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

• A realistic, well-planned schedule is an essential component of every well-planned project

• Most scheduling errors occur as the result of either not properly identifying all of the necessary activities (tasks and/or subtasks) or not properly estimating the amount of effort required to correctly complete the activity

• A detailed schedule is needed as a part of the plan:

– Start with a Gantt chart showing the tasks (that you developed in 3.3) and associated subtasks versus the proposed project calendar. The Gantt chart shall be referenced and summarized in the text.

– Annotate the Gantt chart with when each project deliverable will be delivered

• Completely compatible with an Agile development cycle if that's your thing

How would you plan for the project to be completed in two semesters? Represent with appropriate charts and tables or other means.

Make sure to include at least a couple paragraphs discussing the timeline and why it is being proposed. Include details that distinguish between design details for present project version and later stages of project.

4.2 Feasibility Assessment

Realistic projection of what the project will be. State foreseen challenges of the project.

4.3 Personnel Effort Requirements

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be based on the projected effort required to perform the task correctly and not just "X" hours per week for the number of weeks that the task is active

4.4 Other Resource Requirements

Identify the other resources aside from financial, such as parts and materials that are required to conduct the project.

4.5 Financial Requirements

If relevant, include the total financial resources required to conduct the project.

5. Testing and Implementation

Testing is an **extremely** important component of most projects, whether it involves a circuit, a process, or a software library

Although the tooling is usually significantly different, the testing process is typically quite similar regardless of CprE, EE, or SE themed project:

1. Define the needed types of tests (unit testing for modules, integrity testing for interfaces, user-study for functional and non-functional requirements)
2. Define the individual items to be tested
3. Define, design, and develop the actual test cases
4. Determine the anticipated test results for each test case 5. Perform the actual tests
6. Evaluate the actual test results
7. Make the necessary changes to the product being tested 8. Perform any necessary retesting
9. Document the entire testing process and its results

Include Functional and Non-Functional Testing, Modeling and Simulations, challenges you've determined.

5.1 Interface Specifications

– Discuss any hardware/software interfacing that you are working on for testing your project

5.2 Hardware and software

- Indicate any hardware and/or software used in the testing phase
- Provide brief, simple introductions for each to explain the usefulness of each

5.3 Functional Testing

Examples include unit, integration, system, acceptance testing

5.4 Non-Functional Testing

Testing for performance, security, usability, compatibility

5.5 Process

– Explain how each method indicated in Section 2 was tested

– Flow diagram of the process if applicable (should be for most projects)

5.6 Results

– List and explain any and all results obtained so far during the testing phase

- – Include failures and successes
- – Explain what you learned and how you are planning to change it as you progress with your project
- – If you are including figures, please include captions and cite it in the text

• This part will likely need to be refined in your 492 semester where the majority of the implementation and testing work will take place

-**Modeling and Simulation**: This could be logic analyzation, waveform outputs, block testing. 3D model renders, modeling graphs.

-List the **implementation Issues and Challenges**.

6. Closing Material

6.1 Conclusion

Summarize the work you have done so far. Briefly re-iterate your goals. Then, re-iterate the best plan of action (or solution) to achieving your goals and indicate why this surpasses all other possible solutions tested.

6.2 References

This will likely be different than in project plan, since these will be technical references versus related work / market survey references. Do professional citation style(ex. IEEE).

6.3 Appendices

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar that does not directly pertain to the problem but helps support it, include that here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc. PCB testing issues etc. Software bugs etc.