

**Faculty of Information Technology**

***FIT2044 advanced projects***

***Semester-2, 2007***

# **FINAL REPORT**

Project : Visualization and Exploration  
of Biological Sequence DNAGraphTool.

**Supervisors :**

**Minh Duc Cao**

Email: [MinhDuc.cao@infotech.monash.edu.au](mailto:MinhDuc.cao@infotech.monash.edu.au)

**Julie Bernal**

Email: [Julie.Bernal@infotech.monash.edu.au](mailto:Julie.Bernal@infotech.monash.edu.au)

**Student :**

**Hoang Anh Nguyen**

Email: [hangu4@student.monash.edu.au](mailto:hangu4@student.monash.edu.au)

## **Abstraction:**

DNAGraphTool was developed by Julie Bernal in 2005 in her Honour Project. It is used to display the linear structure of biological sequences and evaluate various compression models. The aims of this project is to add new functionalities into the existing platform to make it more useful and user friendly. The main works of this project relate to displaying biological sequences. During the project, Biojava which is a free open-source biological sequence analysis platform is merged to the DNAGraphTool.

## Table of contents.

<b>1.0 Introduction.....</b>	<b>4</b>
<b>2.0 Overview.....</b>	<b>4</b>
2.1 Biological Terminologies.....	4
2.2 Working approach.....	4
<b>3.0 Implementation.....</b>	<b>5</b>
3.1 Biojava.....	5
3.2 Copy and Paste DNA Sequence.....	5
3.3 Saving Functions.....	6
3.4 Displaying Amino Acids.....	7
3.5 Displaying Sequence Features.....	8
<b>4.0 Results.....</b>	<b>9</b>
5.1 Current Status and limitations .....	9
5.2 Future intentions.....	9
5.3 Personal Experience.....	9
<b>5.0 Conclusion .....</b>	<b>9</b>
<b>References.....</b>	<b>10</b>

## 1.0 Introduction:

In 2005, Julie Bernal developed a platform to support biological sequence analysis. In particular: It displays the linear structure of biological sequences and supports various compression models. It used some low level packages written by Dr David Powell. Other people added their compression models into the platform. This project's aim is to add new functionalities into the platform.

This report briefly describes what has been done in the project. It consists of 4 parts, after the introduction is the platform overview which shortly describes related biological terminologies and the working approach adopted in the project. The next part is the implementation part, it comes with details about implementation process and technical data. Before the conclusion is the results of the project which shows the current status, limitations and future intentions of the project.

## 2.0 Overview:

### 2.1 Biological Terminologies:

- **DNA sequence:** is a sequence of nucleic acid that contains the genetic instructions used in the development and functioning of all living organisms[1]. DNA looks like a very long twisted ladder which is called double helix shape. Each side of the ladder is made up of many repeating subunits called nucleotides or bases. There are 4 types of nucleotides: A, C, G, T. Rungs that connect two sides of the ladder are made up between 2 nucleotides. Because of the chemical nature, A only links with C and G only links with T in those rungs.
- **Amino Acid:** Each codon consists of three nucleotides, representing a single amino acid. In a single side of a DNA sequence, there are three possible reading frames correspond to three starting bases. Each reading frame represents an array of amino acids.
- **Feature:** is a special region in a DNA sequence. Feature can be gene, coding regions, exons, repeat regions, etc.

### 2.2 Working Approach:

Because there are no final goals for the project applied, the approach which resembles to agile model is adopted. Each week, the student and supervisors discuss about what would be done, how the changes affect the interfaces between modules so that changes do not affect the extensibility of the platform. In the project, on average, it took 2 weeks to add a functionality.

## 3.0 Implementation

### 3.1 Biojava

Biojava is an open-source project that provides a java framework for supporting biological sequence analysis. In this project, it is used as low level library like David Powell's packages. Merging biojava into the platform was probably one of the hardest task of this project. The reason was that biojava and DNAgraphTool used different data structures to hold the sequences' information. DNAgraphTools uses an array of characters as it considers DNasequences as arrays of characters. Meanwhile, biojava uses complicated data structure so that it can hold not only the characters in the sequence but also other information like annotation (name, type of the sequence for example). There are three possible solutions for this situation:

- Adopting the biojava data structure as the only data structure used to hold the sequence information as data structure in DNAgraphTool is just a subset of the biojava's data structure. However, if this one was used, the whole interfaces of the platform would have been affected. Therefore, this is not a realistic solution.
- Using array of characters and some information retrieved from biojava such as name, type, features of the sequence.
- Using both data structures.

The solution adopted in this project was to use both data structures, but the program just needs to read and parse the sequence one time by biojava funtions, then the array of character could be retrieved from biojava's data structure. In order to do that, the DNA class in common package (David Powell's package) was changed. It was seperated into 4 classes: DNA, FASTA, Genbank and Raw. The last three classes inherit from the first one. In FASTA and Genbank class, biojava functions are used to read the sequence in and the array of characters is retrieved after that also by biojava functions. The Raw sequence still uses the old parser. Thus, only sequences in FASTA and genbank file format has the data structure provided by biojava.

The following sections detail four new functionalities.

### 3.2 Copy and Paste a DNASequences

In the old version, DNasequences were only imported from hard disk. But in the real world, DNASequences can be imported from any source. Although the user can save DNasequences into the hard disk and import it, it is inconvenient and tedious and in

fact biologists prefer importing sequences by copy-and-paste. This function provides a way to import a sequence by copy-and-paste DNasequences.

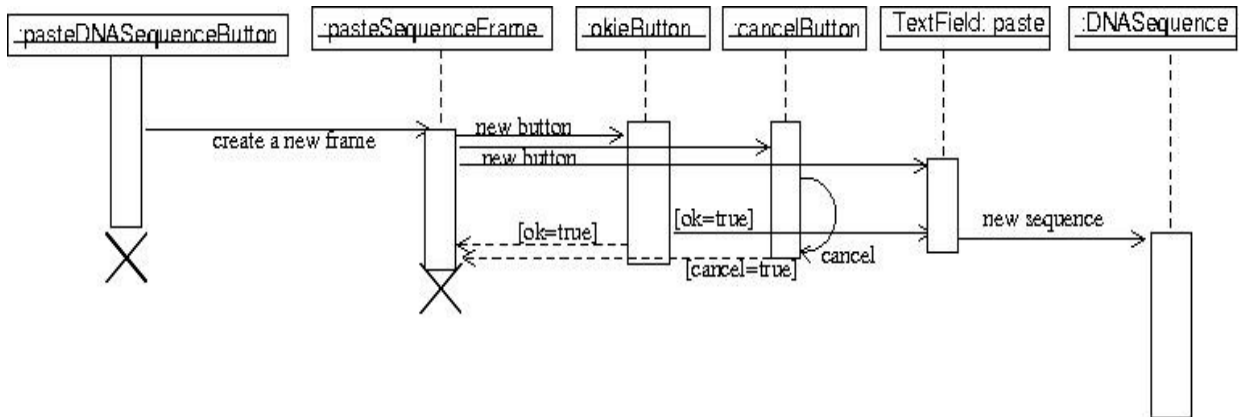


Figure 1: Copy and Paste Sequence Diagram

As can be seen from Figure 1, when the user presses `pasteDNASequenceButton`, a new `pasteSequenceFrame` which contains `okieButton`, `cancelButton` and `paste TextField` is created. When the user presses `okieButton` or `cancelButton`, the `pasteSequenceFrame` will be destroyed. If the user presses `okieButton`, the string in `paste TextField` will be used to create a `DNASequence`.

In order to implement this functionality, an attribute and a method were added in the `mainPanel` class:

- attribute : `pasteDNASequenceButton`
- method : `pasteSequence_actionPerformed(ActionEvent e)`

### 3.3 Save function

This functionality provides a way to save sequences into the hard disk. It supports several types of sequences, including DNA sequences (FASTA and genbank format), character sequences, information contents sequences.

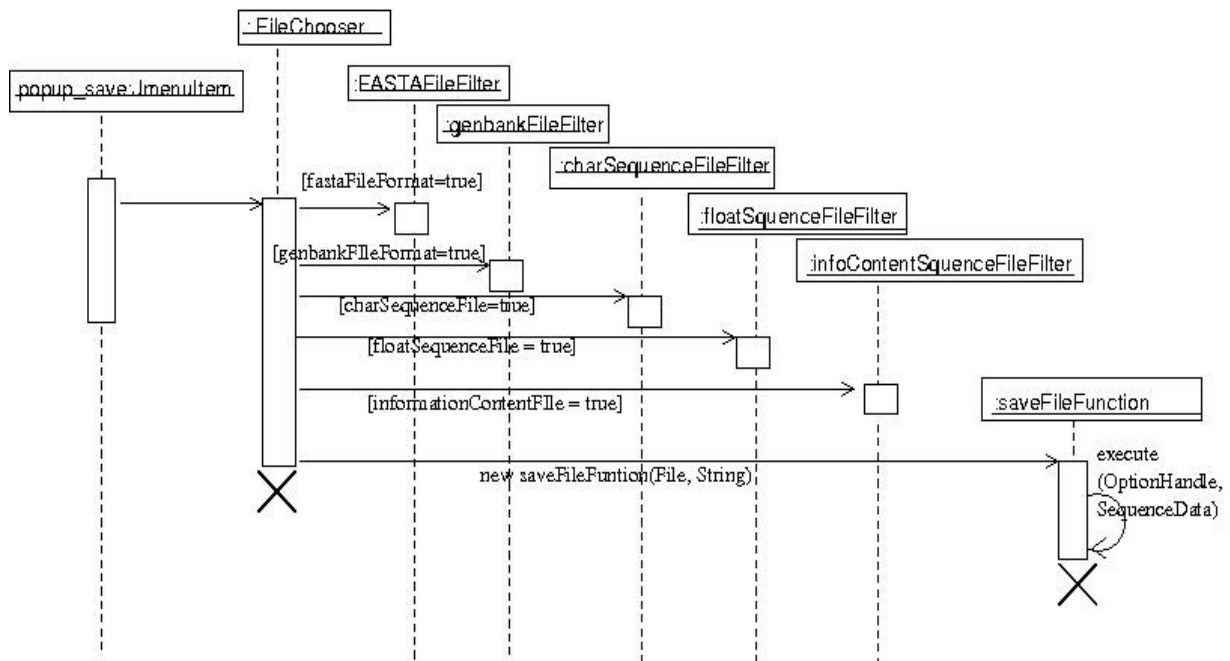


Figure 2: save file function sequence diagram

From the sequence diagram above, we can see that when save function is chosen (by choosing popup\_save JMenuItem), a new FileChooser is created. In the new FileChooser object, one of 5 filter classes will be created based on the type of the sequence and the user's choice (which format to save the sequence). After that, a new saveFileFunction object is created, it then invokes method "execute" to start the saving process.

### 3.4 Displaying Aminoacids

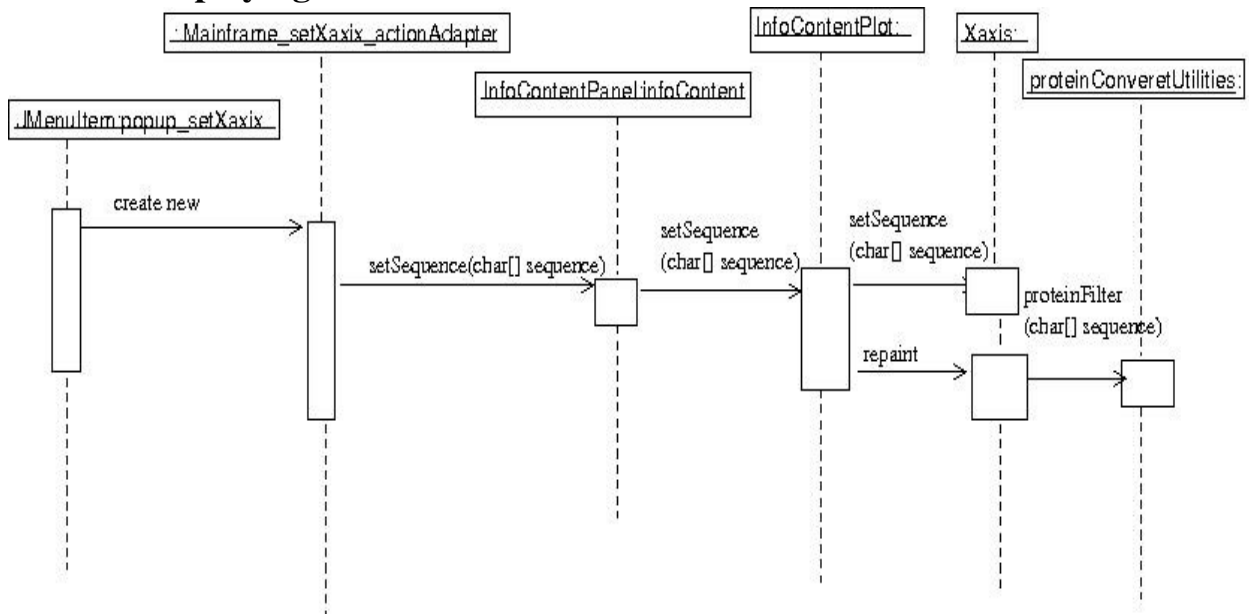


Figure 3: Displaying Amino Acids Sequence Diagram

### 3.5 Displaying sequences' features

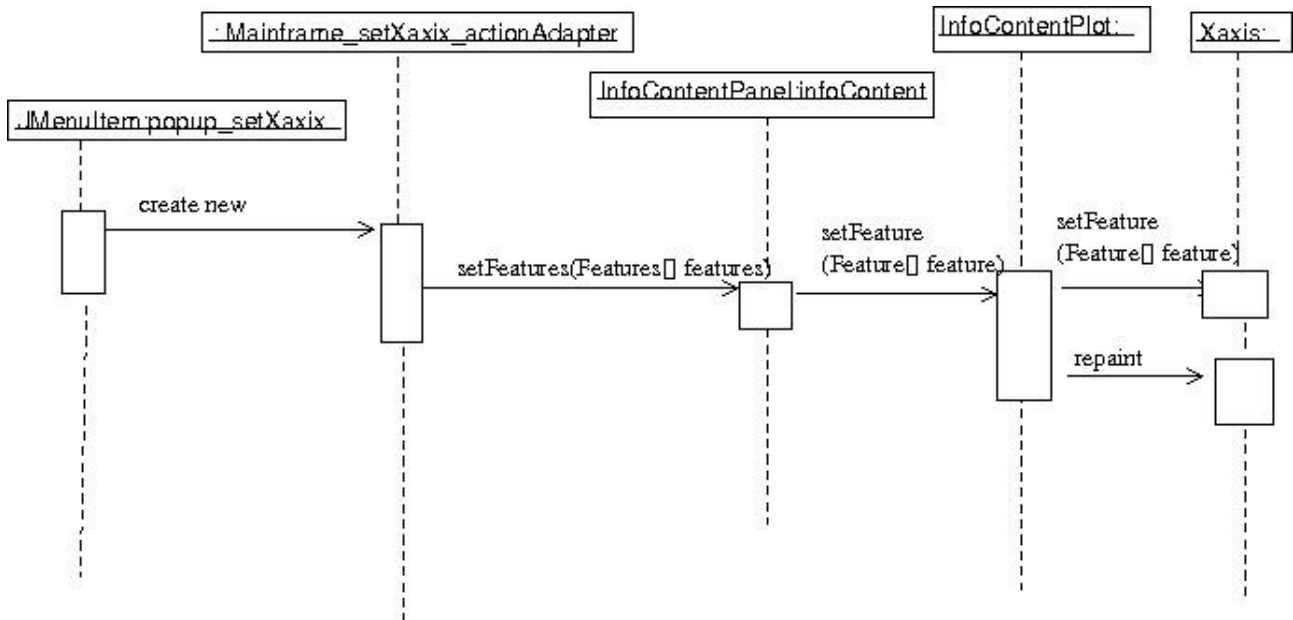


Figure 4: Displaying Features Sequence Diagram

One of the main problem of implementing this functionality was to draw features in the Xaxis. Because the user can navigate the window to any location of the sequence, the program has to search which features overlap the window. The solution adopted was to use a naïve linear search to find which are visible in the window. This algorithm works fine with short sequences with small number of features, but it would considerably decrease the performance of the program with long sequences.

## 4.0 Results

### 4.1 Current status and limitations:

During this project, four functionalities have been added to the platform. The DNAGraphTool can work as a standalone application or as an applet. Nevertheless, the platform has several limitations which could not have been solved in the project. Some of those limitations existed before the project, some of them were caused during the project:

- Lack of documentation
- Linear search is used to search and display sequence features which is fine for small sequences but becomes inefficient for long sequences.

## **4.2 Future Extentions:**

There are some suggestions about future improvements:

- Improve the algorithm used in searching and displaying sequence features so that the tool can be used effectively for long sequences.
- In the current version, there are only three different features to be displayed: genes, exons and coding regions. In the future, the program should let the user choose which features to be displayed.
- Importing sequences remotely from an open accessed sequences database like NCBI- National center for Biotechnology Information.

## **4.3 Personal Experience:**

This project has given me a good practice in Object Oriented programming especially GUI development. At the project, I had the chance to work with other developer's code which is rare in Universty. Once again, I understand the importance of the documentation in a software engineering project.

## **5.0 Conclusion**

The report has just described what was done in the project. At the end of the project, biojava was merged into the platform and 4 new functionalities were added. However, due to time constraints, some problems have not been solved. For future development, solving those problems should be solved before adding new functionalities. As pointed out earlier, this project has been a good programming practice, especially in graphical user interface programming which is my current interest. Experience gained from this project would be useful for future study and career.

## References

The following source of information is used as references for this report:

[1] <http://en.wikipedia.org/wiki/DNA>

[2] Julie Bernal . A platform for DNA Exploration and Visulization. October 2005