COM3523/COM6523 Individual Assignment:

Re-engineering the GraphStream Graph Visualisation Framework

1 Overview

GraphStream is a graph visualisation framework. It used to be quite popular, but development has slowed somewhat. It is your task to analyse and understand the framework, and to carry out a re-engineering task, perhaps to see whether you can rejuvenate it a bit.

As with the other assignments, we will use Github Classroom for this assignment.
Please clone your personal version of this repository here:

[check the PDF on Blackboard for the latest web-link]

We have created a base repository for you, which contains the following familiar directory structure:

- **subjectCode** This contains a snapshot of the repository. This contains the full subject system that you are expected to use as the basis for your analysis. It does not contain any git data pertaining to the version history.
- **submission** This is where you can store your report submission, any data files that are produced (e.g. CSV files with metrics, data traces, images, etc.)
- **analysisCode** This contains the latest snapshot of the analysis code, including code used for static / dynamic analysis, version repository analysis, and code clone analysis, etc.

2 Instructions

For this assignment the goal is to put all of the skills that we have learnt throughout the course into practice. Everything in your repository will count as part of your submission. The main assessed component (see below) will be the write-up, but any figures and committed code will be incorporated into assessment as well.

The report is to be split into the following three sections: (1) Analysis of the system, (2) assessment and critique, (3) reengineering. Their required contents are described in more detail below:

1. Analysis of the system

You should find out what you can about the system. What is the design of the system? What are the important elements (classes or methods)? Which classes or methods are particularly important for the functioning of the system? Which ones are particularly unusual?

Your analysis should involve the following:

- Use of Bash commands and scripts to analyse files and file structure.
- Use of static analysis to either:
  - extract a class diagram or …
  - … identify which methods might be called most frequently (via call graph analysis).
- Use of dynamic analysis to either:
  - identify which methods or classes are called most frequently for a given test execution or…
  - … to carry out a phase analysis to identify “phases” of method executions.
- One of the following:
  - Use of version repository analysis to identify frequently changing areas of code.
  - Code clone analysis to identify code duplication.

For this section you should present your techniques — detailing how you carried them out, highlighting any improvements you made to the approaches we covered in class (e.g. enhancements to analysis code, or enhancements to visualisations). You should present the results in terms of their direct outputs (the diagrams, tables, etc.), and highlight any notable features in the results.

If you analyse the version repository for GraphStream, you will need to clone the original repository into a separate directory, and analyse it from there (we have stripped out the gs-core Git data so that it would not conflict with the Git data for the assignment as a whole). For this you should run:

- git clone https://github.com/graphstream/gs-core.git
- git checkout 807bea7

  This latter command is just to ensure that, if any commits are made to the repository in the interim, you are working on the same version that we have used throughout.

Approximate word-length for COM3523: 700 words.
Approximate word-length for COM6523: 1000 words.

2. Assessment and critique

Use the results from the analysis of the system you conducted previously to highlight what you consider to be its key strengths and weaknesses. Justify your assessment(s) by

referring to the relevant underlying design principles and heuristics. Your assessment might draw coupling, cohesion, code reuse and code duplication.

Your writeup should be entirely based upon evidence,. For this you can point to specific results of analyses you conducted previously, or you can point to specific areas of the source code that you have identified by hand (perhaps following on from your analysis).

Approximate word-length for COM3523: 300 words.
Approximate word-length for COM6523: 500 words.

3. Reengineering the system

Pick one weakness in the system (that you identified in the previous section). It should be a weakness that is non-trivial in nature — not something that can be fixed by merely extracting a method, for example. It should be something more significant, such as a God Class, extensive coupling between one or more classes, a lack of cohesion, etc.

Describe the strategy that you would apply to address the weakness. Use class diagrams to illustrate your re-engineering approach (the diagrams only need to contain the classes and methods that are specifically relevant to the reengineering effort). Take care to mention any risks that might arise (e.g. where a re-engineering step might lead to a deterioration in a different aspect of the design). Also discuss how you would test the re-engineered system.

Implement your re-engineering effort as a change to the source code, and commit it. For all commits that are related to the re-engineering of the source code, add “REENG: “ to the beginning of the commit message.

Your attempt should demonstrate an understanding of the key steps in the re-engineering task. For some tasks, such as splitting up a big God Class, this will necessitate more effort than necessary for this assignment to complete in its entirety. For your re-engineering solution, it is simply important that it covers the essential activities in the strategy you’ve outlined. E.g. if your strategy suggests that you need to create a new class, and move 4 methods and 5 data members from the God class to that class, your applied effort would only need to implement this for a small sub-set of the methods and data members - you can indicate in your writeup that you would just replicate this process for the remaining elements.

Approximate word-length for COM3523: 400 words.
Approximate word-length for COM6523: 600 words.

Assessment

The assessment of your submission will be split into three parts: Analysis of the system (45%), Assessment and critique (10%), and Reengineering the system (25%).

The remaining 20% will be based upon an all-round assessment of the submission, analysing the coherence between the different sections, the presentation, innovation, insightfulness, the value and quality of any changes to the source code, etc.

Section	0-39%	40%-54%	55%-70%	70%-100%	Weighting
Analysis of the system	Incorrect use of techniques, poor presentation of outputs, no insights into the system.	Some basic commands correctly applied, poor presentation of results, few insights into the system.	Extensive application of techniques, well explained in documentation, with extensive presentation of the results, demonstrating insights.	Extensive application of the techniques, along with some novel enhancements. All justified and well explained in the documentation, demonstrating insights.	45%
Assessment and Critique	No coherent link is made between the results of the analysis and the key design principles.	Some basic points about design are drawn from the results, e.g. identification of God classes.	The analysis of the results touches upon several aspects of good and bad design, and is well justified in terms of the analysis results.	The assessment touches upon several aspects of good and bad design, is well justified. The justification includes helpful insights into the *causes* of the poor design - e.g. misuse of inheritance, poor interface use, etc.	10%
Reengineering the System	There is no sensible description of a reengineering strategy, and there are no commits to the source code that amount to a substantive reengineering effort.	There is a sensible description of a reengineering strategy, which is linked to specific aspects of the source code. There are some efforts to fulfil this description in the source code.	There is a sensible description of a reengineering strategy, linked to specific aspects of the source code. Most of the key steps in this strategy have been sensibly applied and are evidenced by commits to the source code.	There is a sensible description of a reengineering strategy, linked to specific aspects of the source code. All of the key steps in this strategy have been sensibly applied and are evidenced by commits to the source code. The write-up also includes evidence to show that the resulting system has been improved by the reengineering efforts.	25%

Submission

Your submission material should be placed inside a directory called “submission”, in the root directory of your Git repository [the other two directories in the root directory should be “analysisCode” and “subjectSystem”].

Your submission will be your GitHub Classroom repository (specifically, the final commit you make to it before the deadline). Your report, containing all of your written work, should be placed in the `submission' directory as either a GitHub MarkDown file (https://guides.github.com/features/mastering-markdown/), or a PDF. This should include figures, tables, and charts where appropriate.

Please place any data files or analysis outputs (e.g. CSV files, images, etc.) that you have produced into your `submission' folder. If the files are particularly large (>3MB, such as trace files), these should be compressed first, but may also be left out.

Please submit your assignment as a commit to your personal GitHub repository. There will not be a submission-point on Blackboard, and submissions via email will not be accepted.

The deadline for the submission is 5pm GMT, on the 28th of May 2020.

Support

To ensure fairness, we will only respond to queries on the Blackboard forum dedicated to this assignment. As usual, please read other queries before you post your own, to ensure that your query has not already been answered. To assist your colleagues when they are checking the forum, please give your query a descriptive title.

You must under no circumstances post your own solutions (or parts thereof) to the forum - so please be mindful when you are posing questions.