UPPSALA UNIVERSITET

Project in Computational Science FALL 2020

Project members:
Shuyi Qin
Sotirios Chatzigeorgiou

Supervisor:
Jørn Letnes (jletnes@slb.com)
Schlumberger

BIG DATA IN GUI LISTS

Background

Populating big data in GUI lists is a challenge, in this project we simulate different populate scenarios in the real world. What happens when some elements:

- slow the population?
- new undefined characters appear?

Introduction

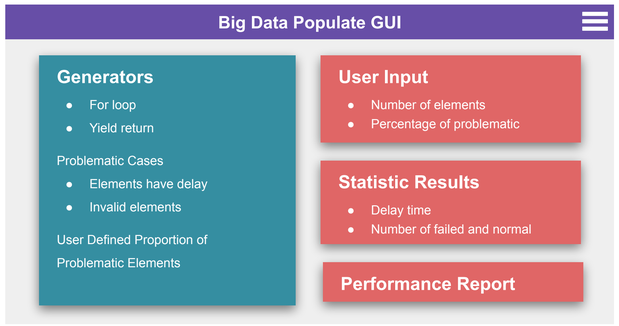
The GUI developed in WPF which uses as frontend XAML and as backend C#. In total 4 projects created to face the problem:

1. WPF application.
2. Data source generators.
3. Unit testing.
4. Performance measurements.

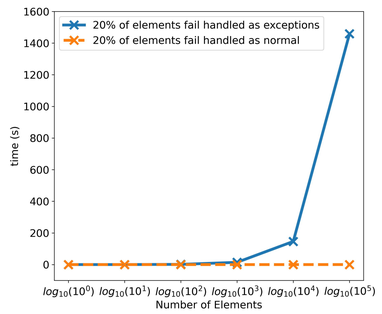
Generators

As elements in the populating list random strings were used. Generators created to compare the return with yield statement feature of C#. In addition, generators that include some percentage of elements with static extra overhead and elements that fail.

The generators where includes failures results in extremely slow population when treated as exceptions to recover instead of direct recovery.



The left part of the GUI consists of the different generators that are populating. The right part indicates how the user can control the generators and gain some metrics.



Time measurement comparison between try-catch exception method and without using exceptions method.

nunit Testing

The population of big data in GUI is yielding particular challenges for developing test cases.

NUnit framework was used to test the functionality and measure the time of populating a list in GUI. NUnit tests that the defined percentage of:

- normal elements are correct.
- slow elements are correct.
- fail elements are correct.

GitHub Actions

Every push to the GitHub repository demands to be compilable and functional. GitHub actions give the possibility to the contributor to ensure the correctness of an updated version of the project.

The workflow on GitHub actions is constructed to compile and test the project right in GitHub repository.

Conclusion

The project was based on the methodology of developing software that Schlumberger follows with unit testing and collaboration through GitHub.

Aside from methodology, the use of try-catch exceptions led to a great time overhead compared with the method without using exceptions which provides the same functionality when a fail element appears.