PRAQMA Blueprints

Memory map parser core library

Help us architect and develop the future core library for memory map parsing, moving from simple regular expressions in a Jenkins plugin to a library doing lexical analysis on memory map file so our effort of implementing support for new linkers is minimized in the future.

Embedded developers always keep a sharp eye on the size of their application if it is something that needs to be flashed to an embedded device's memory. It has to fit the size of the memory.
They use the memory map file, an output from the linking process that describes the memory layout and usage. Those files are not easy to read for humans and Praqma has earlier developed a parser for such files supporting a few linkers. It has gained quite high usage as a plugin for the massively popular Jenkins build server. However it is hard to continuously implement support for more linkers with the current approach and architecture so it needs to change.

User Scenario

As a customer doing embedded development, requesting support for new linkers in the current Jenkins Memory Map plugin, the implementation time and effort that needs funding isn't attractive to drive the development of continuously supporting new linkers and compiler versions.

We imagine a solution where we move away from using regular expressions to parse the memory map files. They have shown to be very error prone, and very easily affected by just simple updates in the linker and compiler versions and thus hard to maintain.
We want to replace the regexp parsing with real lexical analysis, building a grammar to parse memory map files. We believe this simplifies the maintenance of the project and minimizes the effort of supporting new linkers and compilers that only imply minor differences to the older versions.

With such a large change to the core of the current project, we will also move towards separating the core functionality from the use of it as a Jenkins plugin to improve re-usability in other projects.

Planned changes

- Core library for parsing memory map files
- Parsing done with lexical analysis
- Grammars implemented for popular compiler/linker choices like GCC, IAR,

TI.
- Auto generated lexers based on grammars

Design Proposal

There is no design proposal - you will have to come up with a new architecture and design as one of the first tasks.
We do expect you will start from our visions above, looking into lexical analysis for parsing the memory map files and how grammars can be constructed.
You are free to to choose the technology behind the core library, as long as it fulfils our requirements of being reusable, platform independent and testable.
The design is also bound to be modular, and we imagine a core, some plugins to support the different linker/compilers, and some different modules to support output of some kind.

Estimate and skills

The project can be done over 3 months, by a 2-3 person group. It is important to be more than one person to be able to internally discuss and interact on getting good design and architecture in place.
We will expect that only ¼ of the time is actual implementation, while the rest is analysis, design, architectural work and research.
You will need to have a computer science background with some acquired knowledge with how lexers, parsers and compilers work as the lexical analysis can be compared to implementing a new programming language.

References

The current Memory Map Parser as a Jenkins Plugin:
https://wiki.jenkins-ci.org/display/JENKINS/Memory+Map+Plugin

The following links are some base information on lexers and parsers, and should somehow be familiar reading on conceptual level:
- http://www.pling.org.uk/cs/lsa.html
- https://en.wikipedia.org/wiki/Lexical_analysis
- http://dinosaur.compilertools.net/lex/
- http://parsingintro.sourceforge.net/

Contact

Bue Petersen, bue@praqma.net, +45 29 72 64 12