Visualization of program execution in gforth

Proposal

Bachelorarbeit

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Software & Information Engineering

eingereicht von

Mario Gastegger

Matrikelnummer 0726289

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Martin Ertl

Visualization of program execution in gforth

Proposal

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Software & Information Engineering

by

Mario Gastegger

Registration Number 0726289

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Martin Ertl

Abstract

Problem definition

In software engineering, the most important part is to verify the correctness of a program. The difficulty of this task grows with the size and the complexity of a program. Thus the task of finding faulty code takes a considerable amount of time and the efficiency of finding faulty code a major concern. There a several methods to improve the quality of code.

correctness faulty code methods to improve quality debugging watch and break in forth/gforth println method and visualization of program execution transparent and automated

Expected results

improvement of awareness of whats happening during execution and efficiency of finding faulty code

Methodology and approach

1. research on program flow visualization

2. research on forth

3. research on architecture of gforth

4. research on "debugging" in forth/gforth

5. research of similar approaches

6. extracting several technical approaches to accomplish the task(hooks, word-wrapping, level of implementation)

7. evaluation of the approaches(automation?, performance, feasibility)

8. prototyping the approaches in order of quality(?)

State of the art

?

Relation to Software engineering

- software testing(analysis, debugging)

- development process(rapid prototyping)

- stack-based language forth

References

- correctness

- software quality

- debugging watch and break points

- println method

- visualization of data manipulation and

- program flow

Contents

Introduction

1.1 General Information

This document is intended as a template and guideline and should support the author in the course of doing the master's thesis. Assessment criteria comprise the quality of the theoretical and/or practical work as well as structure, content and wording of the written master's thesis. Careful attention should be given to the basics of scientific work (e.g., correct citation).

1.2 Organizational Issues

A master's thesis at the Faculty of Informatics has to be finished within six months. During this period regular meetings between the advisor(s) and the author have to take place. In addition, the following milestones have to be fulfilled:

1. Within one month after having fixed the topic of the thesis the master's thesis proposal has to be prepared and must be accepted by the advisor(s). The master's thesis proposal must follow the respective template of the dean of academic affairs. Thereafter the proposal has to be applied for at the deanery. The necessary forms may be found on the web site of the Faculty of Informatics. `http://www.informatik.tuwien.ac.at/dekanat/formulare.html`

2. Accompanied with the master's thesis proposal, the structure of the thesis in terms of a table of contents has to be provided.

3. Then, the first talk has to be given at the so-called "Seminar for Master Students". The slides have to be discussed with the advisor(s) one week in advance. Attendance of the "Seminar for Master Students" is compulsory and offers the opportunity to discuss arising problems among other master students.

4. At the latest five months after the beginning, a provisional final version of the thesis has to be handed over to the advisor(s).

5. As soon as the provisional final version exists, a first poster draft has to be made. The making of a poster is a compulsory part of the "Seminar for Master Students" for all master studies at the Faculty of Informatics. Drafts and design guidelines can be found at `http://www.informatik.tuwien.ac.at/studium/richtlinien`.

6. After having consulted the advisor(s) the second talk has to be held at the "Seminar for Master Students".

7. At the latest six months after the beginning, the corrected version of the master's thesis and the poster have to be handed over to the advisor(s).

8. After completion the master's thesis has to be presented at the "epilog". For detailed information on the epilog see: `http://www.informatik.tuwien.ac.at/studium/epilog`

1.3 Structure of the Master's Thesis

If the curriculum regulates the language of the master's thesis to be English (like for "Business Informatics"), the thesis has to be written in English. Otherwise, the master's thesis may be written in English or in German. The structure of the thesis is predetermined. The table of contents is followed by the introduction and the main part, which can vary according to the content. The master's thesis ends with the bibliography (compulsory) and the appendix (optional).

- Cover page

- Acknowledgements

- Abstract of the thesis in English and German

- Table of contents

- Introduction

    - motivation

    - problem statement (which problem should be solved?)

    - aim of the work

    - methodological approach

    - structure of the work

- State of the art / analysis of existing approaches

    - literature studies

- – analysis
- – comparison and summary of existing approaches

- Methodology

  - – used concepts
  - – methods and/or models
  - – languages
  - – design methods
  - – data models
  - – analysis methods
  - – formalisms

- Suggested solution/implementation

- Critical reflection

  - – comparison with related work
  - – discussion of open issues

- Summary and future work

- Appendix: source code, data models, . . .

- Bibliography

Typographic Design

For working with LaTeX you can take advantage of a variety of books and free introductions and tutorials on the internet. A competent contact point for LaTeX beginners is the LaTeX Wikibook, which is available under `http://en.wikibooks.org/wiki/LaTeX`.

The following sections give examples of the most important LaTeX environments and commands.

2.1 Tables

Tables have to be realized with the help of the *table* environment. Tables shall be sequentially numbered for each chapter and described in terms of a short caption (cf. Table 2.1).

Name	Date	Title
Mustermann Adam	18.5	T1
Musterfrau Eva	22.6	T2

Table 2.1: Seminar for Master Students

2.2 Figures

Like tables, figures shall be sequentially numbered for each chapter and described in terms of a short caption). You could either produce your drawings directly inside Latex using PSTricks[1], Tikz[2], or any set of macros dedicated to your requirements (cf. Figure 2.1). Alternatively, you may include figures prepared in external tools (cf. Figure 2.2). Note, to ensure high quality printing, all figures must have at least 300 dpi.
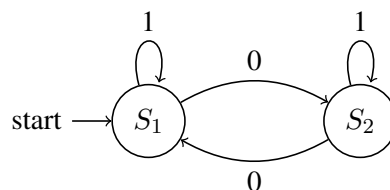
---

[1] `http://tug.org/PSTricks`
[2] `http://sourceforge.net/projects/pgf`
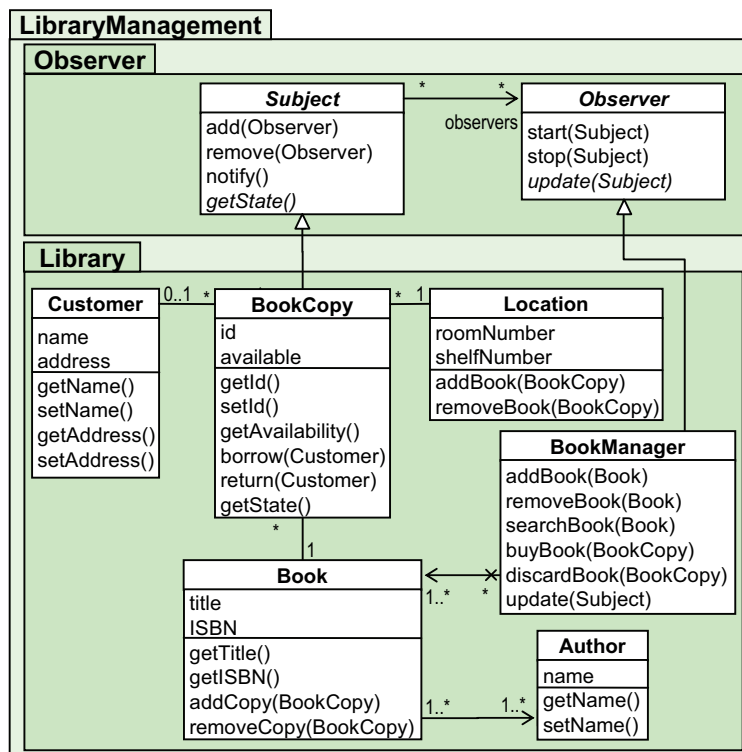
Figure 2.1: Sample figure

Figure 2.2: Sample figure

2.3 Fonts

When introducing important terms for the first time use *emphasize*. For a consistent look and feel of proper names like Class Diagram and `Observer` pattern you may define macros in the main document `thesis.tex`.

2.4 Code

For short code fragments use the *verbatim* environment.

input : A bitmap Im of size $w \times l$
output: A partition of the bitmap

1 *special treatment of the first line*;
2 **for** $i \leftarrow 2$ **to** l **do**
3   *special treatment of the first element of line* i;
4   **for** $j \leftarrow 2$ **to** w **do**
5   left $\leftarrow$ `FindCompress`$(Im[i, j-1])$;
6   up $\leftarrow$ `FindCompress`$(Im[i-1,])$;
7   this $\leftarrow$ `FindCompress`$(Im[i,j])$;
8   **if** left *compatible with* this **then** ;   `// O(left, this) ==1`
9
10   **if** left $<$ this **then** `Union(left,this)`;
11   ;
12   **else** `Union(this,left)`;
13   ;
14   **end**
15   **if** up *compatible with* this **then** ;   `// O(up, this) ==1`
16
17   **if** up $<$ this **then** `Union(up,this)`;
18   ;
        `// this is put under up to keep tree as flat as possible`
19   **else** `Union(this,up)`;
        ;   `// this linked to up`
20   **end**
21   **end**
22   **foreach** *element* e *of the line* i **do** `FindCompress`(p);
23 **end**

Algorithm 2.1: Sample algorithm

```
//Start Program
System.out.println("Hello World!");
//End Program
```

A much better alternative is the *algorithm* environment (cf. Algorithm 2.1). This environment offers special formatting features for loops, operations and comments.

7

Bibliographic Issues

3.1 Literature Search

Information on online libraries and literature search, e.g., interesting magazines, journals, conferences, and organizations may be found at `http://www.big.tuwien.ac.at/teaching/info.html`.

3.2 BibTeX

BibTeX should be used for referencing.

The LaTeX source document of this pdf document provides you with different samples for references to journals [**?**], conference papers [**?**], books [**?**], book chapters [**?**], electronic standards [**?**], dissertations [**?**], masters' theses [**?**], and web sites [**?**]. The respective BibTeX entries may be found in the file `references.bib`. For administration of the BibTeX references we recommend `http://www.citeulike.org` or JabRef for offline administration, respectively.