Code Generation Approaches for an Automatic Transformation of the Unified Modeling Language to the BOINC Framework

Christian Benjamin Ries
and Vic Grout

E-Mail: mail@christianbenjaminries.de
Website: www.visualgrid.org

SCSE'13
San Francisco (USA), 1\textsuperscript{st} March 2013
Overview

- Briefly Introduction of **BOINC**
- Briefly Introduction of **Code Generation (CG)**
- A selection of **CG approaches**
Berkeley Open Infrastructure for Network Computing

Image source:
http://www.seti.cl/aprendiendo-mas-sobre-boinc-y-la-computacion-voluntaria/
BOINC is a Public Resource Computing Middleware Framework

seq1: Work request

seq2: Workunit (WU)

seq3: result

Compute!

Participants!

Company-/Institute network

Earth
BOINC's Workunits (WUs)

Scientific Application
(Independent Software Vendor, Legacy-applications, personal creative implementations...)

[1..*] various sets of runtime-parameter

[0..*] additional files
Excerpt of BOINC's WU Lifetime

WU creation

- Is done by an scientific application.

WU performance

- Is done by specific scripts or other applications.

WU validation

- Is done by an scientific application.

WU assimilation

- Is done by BOINC's default tools or own implementations.

Any part which access WUs need specific interfaces. Furthermore, the validation and assimilation process can vary for any WU.

This transition is on the next slide.
ANother Tool for Language Recognition

http://www.antlr.org

Current version is 4!
ANTLR’s Abstract-syntax Tree

1 + 21 => res

input

lexer

tokens

parser

output

_ _ print "res is 22" _ _ ->

AST (optionally)

tree walker

INT ws ADD ws INT ws ASSIGN ws ID

ws := Whitespaces

summation : leaf* -> leaf* ;
leaf : INT (’+’ INT)* ’=>’ ID -> ^ (ASSIGN ^ (ADD INT*) ID);

Listing 2. Definition of the AST of the summation.
Code Generation Approaches

Work Validation

Transformation to an AST; followed by a code generation process (included in source).

Figure 16. Part of BOINC’s high-level validation framework in UML.

```c
int compare_results (RESULT & r1, void *data1,
                    RESULT & r2, void *data2,
                    bool & match) {
    match = (r1.cpu_time >= 10 && r2.cpu_time >= 10);
    return 0;
}
```

Listing 13. Part of BOINC’s high-level validation framework.
Code Generation Approaches

- Easier recognition of functionality
- UML stereotypes can be replaced by specific icons (i.e. similar to Matlab or Simulink).
Code Generation Approaches

Client-side

```c
std::vector<std::string> messageFields;
/* ...fill messageFields with data! */
if (messageFields.size() >= 3) {
    char msgUp[1024] = {'\0'};
    snprintf(msgUp, 1024,
            "<to>%s </to><n<from>%s </from><n<msg>%s </msg><n",
            messageFields[0],
            messageFields[1],
            messageFields[2])
} 
int ret = boinc_send_trickle_up(
    (char*)variety, msgUp);
if (ret) { /* Error-handling, ... */ }
```


Figure 12. Code-mapping for UML4BOINC’s «TrickleUp».
Code Generation Approaches

- Easier recognition of functionality
- UML stereotypes can be replaced by specific icons (i.e. similar to Matlab or Simulink).

![Diagram](image1)

- Figure 10. «Action» for file handling with two parameters.

- Figure 11. «Action» for file handling with three parameters.

<table>
<thead>
<tr>
<th>Mode</th>
<th>BOINC’s Function Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>Touch</td>
<td>int res = boinc_touch_file (path);</td>
</tr>
<tr>
<td>Delete</td>
<td>int res = boinc_delete_file (path);</td>
</tr>
<tr>
<td>Mkdir</td>
<td>int res = boinc_mkdir(path);</td>
</tr>
<tr>
<td>Rmdir</td>
<td>int res = boinc_rmdir(path);</td>
</tr>
<tr>
<td>Cleanr</td>
<td>int res = clean_out_dir (path);</td>
</tr>
</tbody>
</table>

**Table II**

Mapping of UML to BOINC’s API.

- [Copy](#) int res = boinc_copy(path, newpath);
- [Chown](#) int res = boinc_chown(path, owner);
- [Rename](#) int res = boinc_rename(path, newpath);
- [Open](#) FILE *res = boinc_fopen(path, filemode);
Code Generation Approaches

- Easier recognition of functionality
- UML stereotypes can be replaced by specific icons (i.e. similar to Matlab or Simulink).
Thank you,

Christian Benjamin Ries

e-mail: mail@christianbenjaminries.de
website: www.visualgrid.org