Simplifying Complex Software Assembly
The Component Retrieval Language and Implementation
Eric L. Seidel¹, Gabrielle Allen²,³, Steven Brandt², Frank Löffler², and Erik Schnetter²,⁴
¹Department of Computer Science, City College of New York
²Center for Computation & Technology, Louisiana State University
³Department of Computer Science, Louisiana State University
⁴Department of Physics & Astronomy, Louisiana State University

Abstract
Assembling simulation software along with the associated tools and utilities is a challenging endeavor, particularly when the components are distributed across multiple source code versioning systems. It is problematic for researchers compiling and running the software across many different supercomputers, as well as for novices in a field who are often presented with a bewildering list of software to collect and install.

In this work, we introduce a language (CRL) designed to address these issues, as well as our implementation called GetComponents, which is now used by the Cactus Framework.

Introduction and Motivation

Distributed Software Frameworks are becoming increasingly difficult to assemble and maintain:

• They can be spread across multiple repositories and various version control systems.
• They require a large number of individual commands to retrieve completely.
• Many users want to run their applications on a variety of compute resources.
• Users may also want to include other tools to assist in building and running the application.

The Cactus Framework [1] used a tool called GetCactus to facilitate this process. GetCactus was created in 2000 and designed only for use with a few specific Version Control Systems (VCS), and after 10 years was no longer sufficient. We designed and developed a new application, GetComponents, to address these issues and to create a general-purpose tool.

Grammar

```
# NAME is an alphanumeric or ':' character
DOCUMENT : DIRECTIVES ;
DIRECTIVE : DEFINE NAME = 'PATH' EDL |
| CHECKOUT = '' EDL COMPONENTLIST EDL |
| NAMESPACE = '' EDL COMPONENTLIST EDL |
| HL_CTX_LOC = '' LOC EDL |
| HL_CTX_LOC = '' LOC EDL |
| NAMESPACE = '' EDL |
| COMPONENTLIST = '' |
| PATH_DIRECTIVE = '' |
# @REPO_PATH, @CHECKOUT, @TARGET, |
# @AUTH_PASS, @NAME |
| NAME_DIRECTIVE = '' NAME EDL |
| COMPONENTLIST = EDL PATH = '' |

LOC : PREV_REPO_PATH # CVS repository |
| NAME = ''/ ''/ ''/ '' | Git/SV repository |
| NAME = '' '' '' '' | Git repository |
| PATH = NAME |

COMPONENTLIST : PATH |
| COMPONENTLIST EDL PATH = '' |
```

Design Issues

The Component Retrieval Language (CRL) [3] was designed to alleviate issues that had arisen with GetCactus. CRL and GetComponents focus on the following design issues:

• Easy distribution of component lists.
• Support for both anonymous and authenticated retrieval of components.
• Support for different repository and distribution types.
• Support for updating components.
• Support for multiple component lists specified together.

Authentication

```
Authentication

Are components available anonymously?

Yes

no

Use anonymous checkout

Are usernames to URL known?

Yes

no

Prompt for username

Verify access

Checkout components

Use known component

Checkout
```

Acknowledgements

This work was supported by NSF #0904015, NSF #0725070, NSF #0721915, NSF #1005165, and the Blue Waters Undergraduate Petascale Internship. We used TeraGrid resources under allocation TG-MCA02N014.

References