Developers of high-performance scientific applications and frameworks expect tools to provide complete analysis capabilities for the advanced constructs of object-oriented languages, such as templates and namespaces. We developed an analysis infrastructure that provides this support, the

Program Database Toolkit

Based on the latest EDG Front End, the Program Database Toolkit consists of the following components: the C/C++ II Analyzer and DUCTAPE.

EDG Front End

The Edison Design Group (EDG) Front End is a parser that is nearly up-to-date with the C++ standard. The Front End parses a source file, and creates an intermediate language (IL) tree.

II Analyzer

The II Analyzer processes the intermediate language tree, and creates another file. This file contains the high-level “interface” of the original source. It consists of item descriptions that characterize functions and classes, including template instantiations, as well as other types, source files, namespaces, templates, and macros. This file is in program database (PDB) format, and can be easily and efficiently read by a programming or a scripting language.

DUCTAPE

DUCTAPE (C++ program Database Utilities and Conversion Tools Application Environment) provides a C++ library that enables applications to access PDB files.

The Program Database Toolkit is used in the development of applications. PDT enables static analysis, generation of documentation and “glue” code, and source-to-source translation.

Static Analysis and Documentation Generation

Four DUCTAPE applications have been developed: pdbcnv converts PDB files to a more readable format, pdbmerge merges PDB files from separate compilations, pdbtree prints file inclusion, class hierarchy, and call graph trees, and pdbhtml “htmlizes” C++ source.

Code Generation

For SILOON (Scripting Interface Languages for Object-Oriented Languages), PDT assists in generating “glue” code. This automates access to C++ routines from programs written in scripting languages.

Source-to-Source Translation

The TAU (Tuning and Analysis Utilities) Instrumentor uses PDT to instrument C++ programs for TAU profiling and tracing instrumentation. It traverses the PDB list of functions and templates, and inserts the TAU profiling macros in the source. The programs are then recompiled and linked with the TAU library to generate profile data files during execution.

Program Database (PDB) Format

```
Program Database (PDB) Format

all ITEMS:   # source position, parent class, access
HEADER:     # PDB 1.0
FILES:      # header files included by source file
Routines:   # signature, characteristics, functions called, template from which instantiated
TYPES:      # various characteristics, depending on type
CLASSES:    # various template classes, functions, functions called, template from which instantiated
NAMESPACES: # members of namespace or alias
TEMPLATES:  # test of template
MACROS:     # test of macro
```

Kathleen Lindlian, Allen D. Malony, Jan Cuny, Sameer Shende

Bernd Mohr

Peter Beckman

http://www.accl.llnl.gov/pdtoolkit/