PDT

Program Database Toolkit

Overview

Many tasks in an integrated programming environment require access to program information for their implementation. Program Database Toolkit (PDT) is a framework for analyzing source code written in several programming languages and for making rich program knowledge accessible to developers of static and dynamic analysis tools. PDT implements a standard program representation, the program database (PDB), that can be accessed in a uniform way through a class library supporting common PDB operations. Software tools can use this library to accomplish tasks such as

- documentation of program components;
- creation of graphic program browsers that show class hierarchies, function call graphs, and template instantiations;
- insertion of instrumentation for performance profiling and tracing; and
- generation of interface details for calling library routines or building interlanguage bindings.

Figure 1 (on reverse side) shows the PDT framework and the tools that have been developed to use it. These are discussed more fully below.

PDT Intermediate Language Analysis

PDT utilizes state-of-the-art front-end parsers from the Edison Design Group and Mutek. Currently, PDT supports the C and C++ languages. Support for FORTRAN 77 and FORTRAN 90 is in development; support for Java will be added during the coming year.

Each language front end produces its results in an "intermediate language" (IL) tree form. Although the IL program trees are similar, they are distinguished by the language constructs. Our IL Analyzers input the IL, walk the IL tree, and filter and reorganize the information about the parsed program into a more structured, standard format, the PDB.

The level of conversion provided by PDT is determined by the amount and detail of program information required by tools. Currently, the PDB contains all information on functions and classes, including template instantiations, and also contains information on templates and macros. The routine section lists source identification, parent class and access, signature, characteristics, and functions called in each routine. The class section specifies source information, characteristics, direct parent classes, member function IDs, and information on other members. The template and macro sections report source information, type, and text of each entity. As the needs of the tools increase, so will the sophistication of the IL conversion and the PDB.

PDB Query and Management

A tool called DUCTAPE (C++ program Database Utilities and Conversion Tools APplication Environment) implements a C++ library that enables applications to access PDB files. With DUCTAPE, tools can read, merge, write, and, most importantly, query the PDB for specific program information. The structured form of the PDB is reflected in the DUCTAPE application programming interface, allowing easy access to all high-level source data, such as functions, classes, templates, source files, namespaces, and macros.

Our Intermediate Language (IL) Analyzers input the IL, walk the IL tree, and filter and reorganize the information about the parsed program into a more structured, standard format, the program database (PDB).

PDT Application

As shown in the figure, PDT has been applied in a variety of contexts, mainly targeting the requirements of the Advanced Computational Testing and Simulation (ACTS) toolkit and Accelerated Strategic Computing Initiative (ASCI) application developers. PDT is able to handle large, "industrial strength" source code including the POOMA (Parallel Object-Oriented Methods and Applications) framework.

Four tools for static analysis and documentation generation have been developed using PDT:

- PDBconv converts PDB files to a human readable ASCII form;
- PDBmerge merges PDB files from separate compilations;
- PDBtree prints file inclusion, class hierarchy, and call graph trees in ASCII form; and
- PDBhtml "htmlizes" the program information.

Other tools are being implemented to provide graphical user interfaces for interactive PDB query and for graphical displays of program structure.

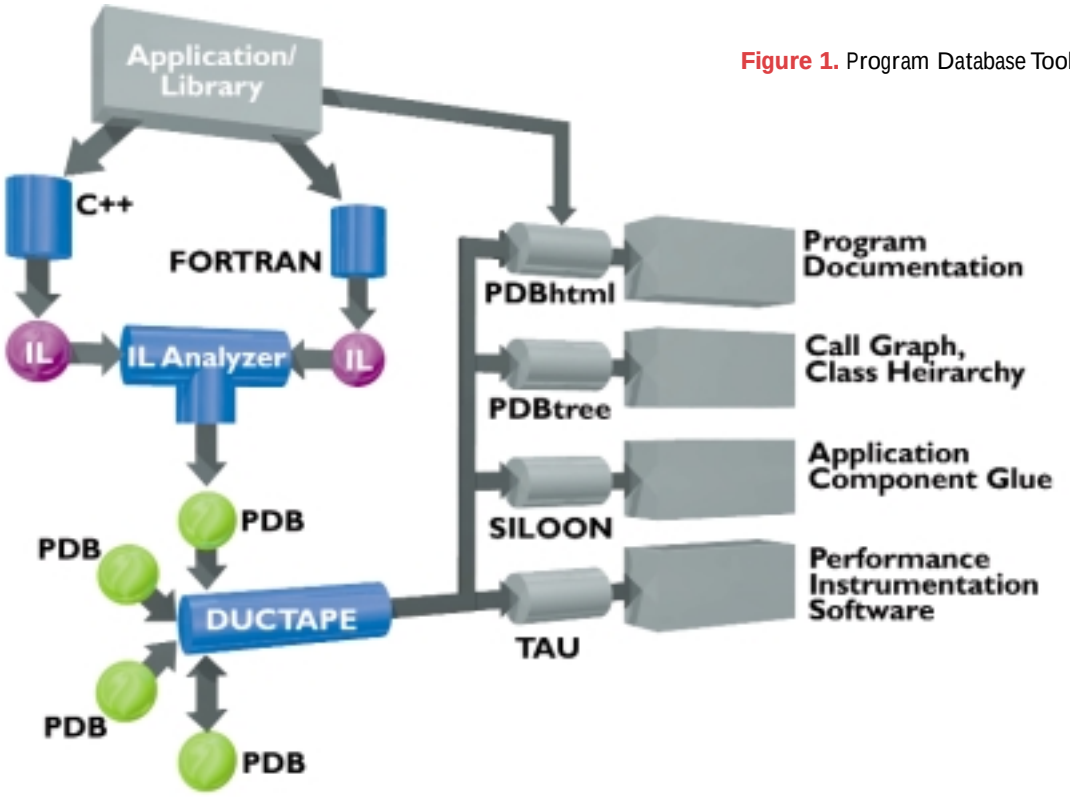

Figure 1. Program Database Toolkit architecture (PDT).

PDT can also be applied for source-to-source program translation. For example, TAU (Tuning and Analysis Utilities) uses PDT to instrument C++ programs for profiling and tracing. 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. The TAU instrumentor has been used to instrument the entire POOMA C++ framework. Because PDT supports common program abstractions, the TAU instrumentor can be easily modified to work with other languages.

The power of PDT to represent and access program information can be seen in its use by the SILOON (Scripting Interface Languages for Object-Oriented Numerics) project to automate the generation of "glue" code that enables C++ library routines to be called remotely from routines written in scripting languages. The PDT tool ensures that interfaces are used correctly and that interlanguage data representation differences are properly accounted for.

PDT Availability

The first version of PDT for C and C++ has been released and is included on Los Alamos National Laboratory's 1999 Advanced Computing Laboratory Software CD. The release includes the IL Analyzers as well as DUCTAPE's library and source code analysis tools. The TAU instrumentor is also available on the CD.

FORTRAN 77 and 90 support is in development, and support for Java is expected within the next year.

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

Acknowledgements

The PDT project is a joint collaboration between the University of Oregon, the Advanced Computing Laboratory at Los Alamos National Laboratory, and Research Centre Juelich in Germany. The IL Analyzers were written by Kathleen Lindlan at the University of Oregon. DUCTAPE was developed by Bernd Mohr at the Research Centre Juelich.

More information about PDT...
web: www.acl.lanl.gov/pdtoolkit/

Get PDT and other Advanced Computing Laboratory Software...
web: www.acl.lanl.gov/software/
cd: 1999 Advanced Computing Laboratory Software

More information about IL Analyzers...
contact: Kathleen Lindlan, University of Oregon
e-mail: klindlan@cs.uoregon.edu

More information about DUCTAPE...
contact: Bernd Mohr, Research Centre Juelich
e-mail: B.Mohr@fz-juelich.de

This work supported by the US Department of Energy.

ACL advanced computing laboratory

os Alamos NATIONAL LABORATORY

LALP 99-123 **November 1999**