

## Automatic Whitebox Tool for SONY

**Client:** Sony Global Solutions (SGS)

**Industry:** Manufacturing

### Team Structure

#### Onshore

Engagement Manager (BSE) 2  
QA/Tester 2

#### Offshore

Section Manager (SM) 1  
Project Manager (PM) 2  
Project Leader (PL) 5  
Sr. Developers (Sr. PG) 5  
Jr. Developers (Jr. PG) 15  
QA/Tester 4

---

**Total FTE** 36

### Delivery Environment

#### OS:

GNU/Linux

#### Development Tools:

GNU Flex, GNU Bison, GNU Gcc Compiler (G++), Bash, Perl

#### Testing Framework:

Original Developed Perl Scripts Suits

#### Team Environment:

Subversion, Bugzilla, MS Project

*Notes: All the platform/tools are open source projects, expect for "MS Project"*

### The Client

SGS supports Sony global business by constructing IT infrastructure and platform (Network Security Data Center, Contact Center, etc), and providing service. SGS reached about 60 billions sales (in JPN) in 2007.

### Project Summary

Automatic Whitebox Tool (AWT) is a source code static analyzer tool for C/C++ source.

AWT analyzes source code and perform code-inspection, produces report to developers for detecting defect patterns rapidly.

### Business Needs

SGS provides source static analyzing service and software development for other Sony companies.

SGS uses some commercial source analyzers, and needs a tool to:

- Cover certain patterns which are considered as 'bad coding' under Sony coding rule, and can not be detected by commercial analyzers
- Drop some miss-warning reported by commercial analyzers
- Extend new patterns easily to meet requirements from other Sony companies

### The Challenge

Actually, a 'source analyzer' is a compiler with code-generation step replaced by 'defect detection' step, and the grammar of C/C++ is one of most complex grammars.

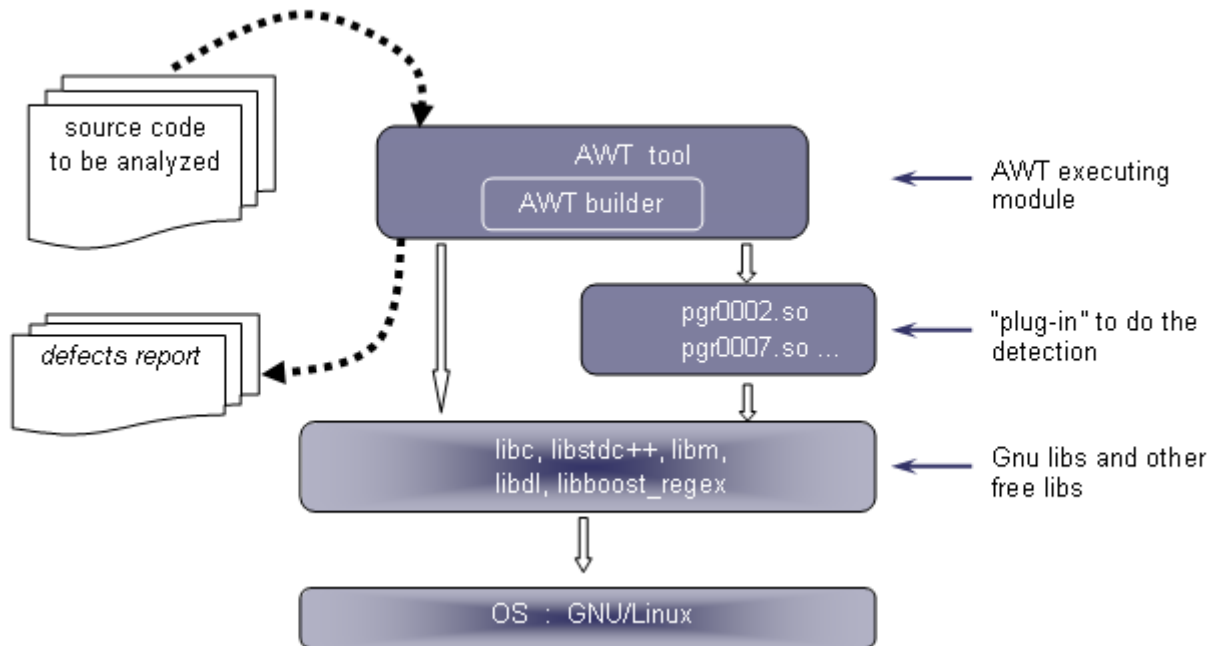
We need to parse C/C++ grammar, build AST (abstract syntax tree), make symbol tables, and finally construct a defect detection mechanism which can be extended easily in future.

## The Solution

Onsite, engagement manager contacted development leaders from other Sony companies to confirm their needs -- which code patterns they wanted to pickup, and researched the commercial analyzer used by SGS, verified what they can do / can not do. After the manager made detailed requirement of AWT, onsite QA/Tester made test suites based on the requirement.

The offshore team researched parser-generators, chose the best and most flexible 'flex/bison' combo, got standard ISO c/c++ grammar from internet, transformed it to yacc rule, and adjusted it to be more practical. Offshore team developed symbol table generating module from scratch. They used plug-in modules to detect the faults, which insure that more plug-ins can be added easily in the future. The figure below shows the technical architecture.

**Figure 1: Technical Architecture**



## The Benefits

With the AWT, SGS provides more rapidly and exactly static code analyzing service.

In detail, AWT helps SGS by covering 6 bad-coding patterns which can not be detected by existing commercial analyzers, and automatically dropping about 10% miss-warning from the result of the commercial analyzers.

Copyright © 2009 Seioglobal Co., Ltd.

ALL RIGHTS RESERVED

Copyright in whole and part of this document belongs to Seioglobal Co., Ltd. This work may not be used, sold, transferred, adapted, abridged, copied or reproduced in whole or in part in any manner or form or in any media without consent of Seioglobal Co., Ltd.