Create your Embedded Life!

# 정적 분석을 통한 효과적인 결함 검출 사례 소개

## CodeSonar

MDS테크놀로지
우경일 팀장
kyungil@mdstec.com
010 – 5123 – 6562

mds technology

# 정적 분석 이란?

mds technology

- **정적 분석의 의미**
  - 소스코드를 실제로 실행 시키지 않고 분석하는 방식
  - 실행 시 발견 될 수 있는 오류를 감지
  - Coding Rule / Guide에 맞게 작성 되었는지 검사

- **CodeSonar를 통한 정적 분석의 특성**
  - 소스코드 컴파일 시 바로 분석 수행
  - 소스코드의 전체가 아닌 부분에도 분석 수행
  - 완성되지 않은 소스코드 분석
    **→ 프로그램 완성 전 까지 중간 중간 적용**
    **→ 계속해서 소스코드의 품질에 대한 Feed-Back 반영**
  - 실행 가능 한 모든 경로에 대하여 분석
    **→ 예상치 못한 Case 나 흔하지 않은 Case에 대한 분석 수행**

# 메모리 누수 검출 예제 (1/2)

mds technology

referenced by sbcset

```
3567
3568  sbcset = (re_bitset_ptr_t)calloc(sizeof(bitset_t),1);
3569  mbcset = (re_charset_t *)calloc(sizeof(re_charset_t),1);
3570
3571  if(BE (sbcset == NULL || mbcset == NULL, 0))
3572  {
3573      *err = REG_ERROR;
3574      return NULL;
      }
```

Generated on Thu Sep 6 14:27:27 2007 by GrammaTech CodeSonar 2.1p1 which was built on May 26 2007 05:10:53

→ 위와 같은 Code 실행 시 **메모리 누수 현상**이 Run-Time 때 발생할 수 있다.
→ 하나의 calloc()에서만 메모리 할당 실패 동작이 발생해야만 한다.

# 메모리 누수 검출 예제 (2/2)

- **예제 분석**

- 앞의 예제에서 보듯이 **쉽게 일어나지 않는 상황까지 모두 분석**하는 정적 분석 방식은 **메모리 누수 현상을 조기에 감지**해낼 수 있다.

- **동적 분석의 경우 예제와 같은 경로로 실행하는 특정 Test Case를 작성 해야만** 예제 상의 **메모리 누수 현상을 검출**해낼 수가 있다.

  → 이러한 **Test Case를 작성**하기 위해서는 **Tester의 경험이 풍부해야 하며 시간과 비용이 많이 들 수** 있다.

  → **정적 분석의 경우, 모든 실행경로에 대해 분석**하므로 **Test Case의 제약을 받지 않는다.**
  **(Test Case 작성에 소요되는 시간과 비용이 줄어든다.)**

# 실제 사례 I (Commercial)

mds technology

## Leap-year glitch freezes Zune MP3 players

STORY HIGHLIGHTS
- NEW: Microsoft says
- Thousands of older
- Message boards are
- User: "It seems that

Next Article in Technolo

READ | VIDEO

By Brandon Griggs
CNN

TEXT SIZE

**(CNN)** -- A leap-year related glitch caused thousands of Zune MP3 players to simultaneously stop working late Tuesday and early Wednesday, Microsoft said on the product's Web site.


GETTY IMAGES/FILE

Microsoft issued the first Zune portable music player in 2006 to compete with the iPod.

The problem should resolve itself after 7 a.m. ET Thursday, Matt Akers of the Zune Product Team wrote on Zune.net. "A bug in the internal clock driver related to the way the device handles a leap year" is to blame, he said.

The issue was limited to older Zune 30GB models, the Web site said.

The Zune support page says users should allow the internal battery to fully drain. Then, after noon GMT on January 1, 2009 (7 a.m. ET), users should recharge by connecting the Zune to a computer or AC power.

Internet message boards were flooded with complaints about Zunes freezing, prompting Y2K-like speculation about end-of-year hardware or software problems.

"It seems that every Zune on the planet has just frozen up and will not work," posted a Mountain Home, Idaho, user on CNN's iReport.com. "I have 3 and they all in the same night stopped working."

Another iReporter said he was working the night shift at a Toys R Us store in Puerto Rico when his Zune player and the Zunes of four co-workers all failed about 1:30 a.m. ET Wednesday. **iReport.com: They all froze up**

```
view plain  copy to clipboard  print  ?

01. year = ORIGINYEAR; /* = 1980 */
02.
03. while (days > 365)
04. {
05.     if (IsLeapYear(year))
06.     {
07.         if (days > 366)
08.         {
09.             days -= 366;
10.             year += 1;
11.         }
12.     }
13.     else
14.     {
15.         days -= 365;
16.         year += 1;
17.     }
18. }
```

- **Zune Bug**

  - MS사의 MP3 플레이어 Zune이2008년 12월 31일에 일제히 동작을 멈추는 사건 발생

  - Leap-year(閏年) 버그가 그 이유였으며, 무한 루프에서 비롯됨

  → **<u>무한 루프 이슈는 정적 분석 도구로 쉽게 검출 할 수 있는 결함</u>임**

  → **이 사건으로 Zune에 대한 소비자들의 신뢰도를 떨어뜨렸으며, 판매하락으로 이어졌으며, MS사의 명성에도 큰 누가 되었음**

# 실제 사례 II (Medical)

mds technology

- **FDA**
  - A사의 의료기기 제품인 혈액 공급기 제품은 검수 시 항상 철저한 동적 테스트 과정을 거치도록 되어 있음
  - 실제 사용 시 문제가 발생하여, 회수 후 **정적 분석 도구로 분석한 결과, 제조사도 몰랐던 45개의 추가적인 결함들이 검출** 됨
  - 결국 이러한 결함을 수정하기 위해 엄청난 비용을 소모함

## GrammaTech Aids FDA Investigators in Determining Why Products Fail

When a widely-deployed, commercial medical device failed in the field, investigators at the FDA used GrammaTech CodeSonar® to help determine root cause. CodeSonar analyzed the device's 200,000 lines of code and identified the following problems.

| Warning Class | Actual Problems |
|---|---|
| Cast Alters Value | 29 |
| Missing Return Statement | 1 |
| Null Pointer Dereference | 28 |
| Redundant Condition | 4 |
| Uninitialized Variable | 36 |
| Unreachable Code | 20 |
| Useless Assignment | 9 |
| Total Found By CodeSonar | 127 |

The results of the analysis were compared with the 118 bugs known to the manufacturer from complaints. Of these, 82 were successfully detected by CodeSonar, yielding a rate of approximately 70%. More importantly, CodeSonar detected an additional 45 errors that had not yet been reported. Conventional testing can only check code being executed. For real-world systems, the number of execution paths far exceeds the number that can be exercised by testing. Static analysis, on the other hand can examine all program paths. That is why CodeSonar can find many hidden bugs, even in software that has undergone the most extensive testing.

*"We have reason to believe that static analysis—whether used in pre-deployment analysis by the manufacturer or during post-market surveillance by an investigator—has the potential to greatly reduce software anomalies and lead to safer, more dependable medical devices."*

– **Raoul Jetley, Researcher at the FDA**

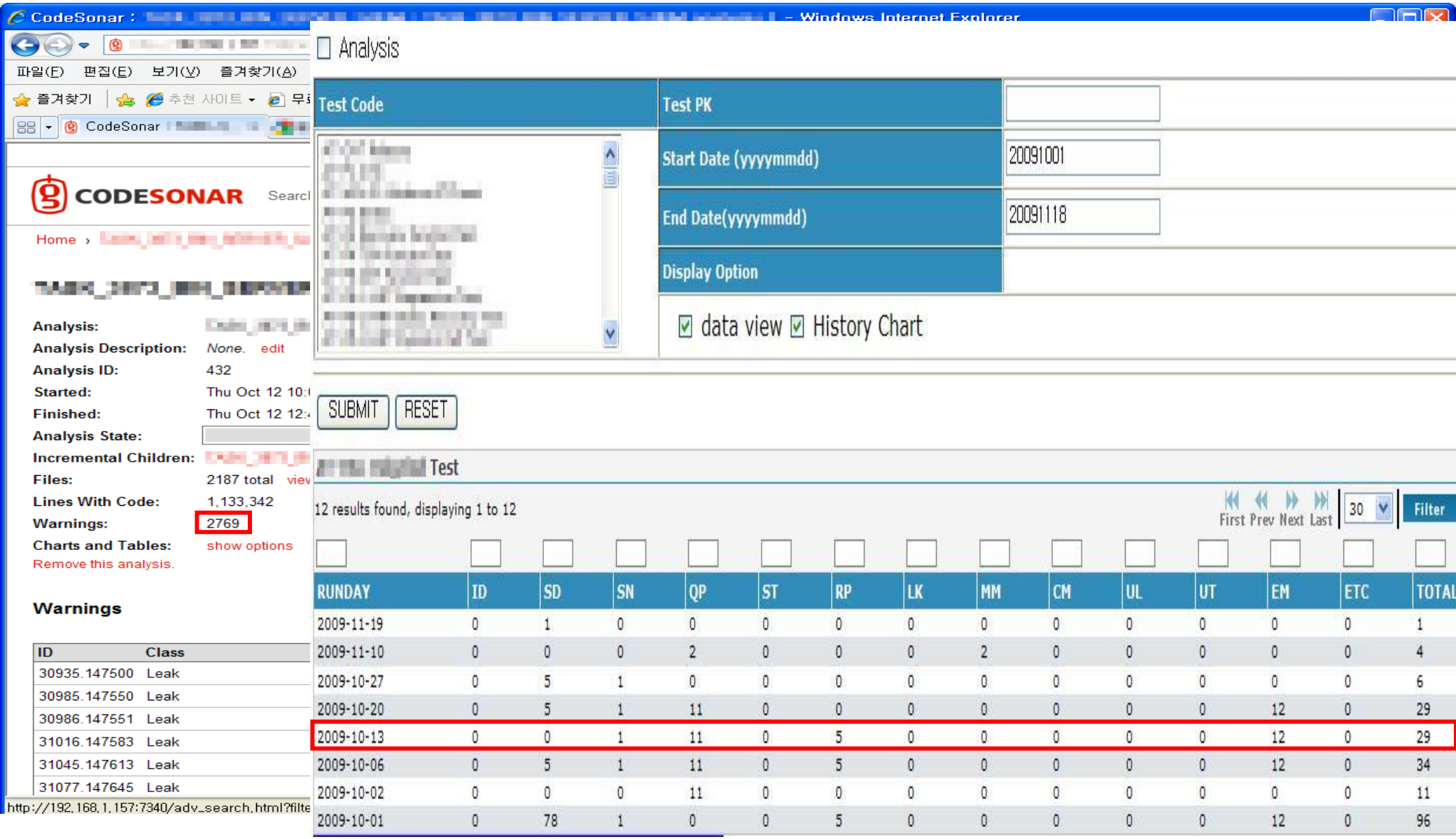
# 실제 사례 III (Domestic)

- 실험기업 : A사
- 실험프로젝트 종류: 데이터베이스 서버 모듈
- 프로젝트 라인 수 : 약 113만 라인
- 프로젝트 컴파일러 : Linux – ARMCC
- 실험 방법 : CodeSonar를 이용해 검출된 결함의 수와 실제 동적 테스트 시 발생 되는 System 오동작 및 Reset 발생 빈도와의 상관관계 측정

- 실험 기간 : 2009. 10. 12 ~ 2009. 11. 18 ( 약 한달 )
- 실험 목적 : CodeSonar를 이용한 정적 테스팅이 실제 동작에 미치는 효과

# 실제 사례 III (Domestic)

## 1차 적용 프로세스 (도입 초기)



# 실제 사례 III (Domestic)

mds technology

☐ Analysis

| Test Code | Test PK | |
|---|---|---|
| [redacted] | Start Date (yyyymmdd) | 20091001 |
| | End Date(yyyymmdd) | 20091118 |
| | Display Option | |
| | ☑ data view ☑ History Chart | |

SUBMIT RESET

[redacted] Test

12 results found, displaying 1 to 12 First Prev Next Last 30 Filter

| RUNDAY | ID | SD | SN | QP | ST | RP | LK | MM | CM | UL | UT | EM | ETC | TOTAL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2009-11-19 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 2009-11-10 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 4 |
| 2009-10-27 | 0 | 5 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 |
| 2009-10-20 | 0 | 5 | 1 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 12 | 0 | 29 |
| 2009-10-13 | 0 | 0 | 1 | 11 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 12 | 0 | 29 |
| 2009-10-06 | 0 | 5 | 1 | 11 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 12 | 0 | 34 |
| 2009-10-02 | 0 | 0 | 0 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 11 |
| 2009-10-01 | 0 | 78 | 1 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 12 | 0 | 96 |

**CodeSonar에서 검출된 문제 수정 후**

**동적 테스트 시 결함 발생 비율이 현저히 감소**

# 실제 사례 IV (현대자동차 개발 프로세스)

개발자

Coding

Defect Message

Code Inspection

Repository

QAC (MISRA)

CODE SONAR

## OEM

- 기능 테스트
- 시스템 테스트
- 보안 테스트
- 성능 테스트

## Software Release

- QAC Compliance Report
- CodeSonar Defect Report

# 실제 사례 IV (현대자동차 개발 프로세스)

mds technology

## Coding Rule 선정

- 멀티미디어 특성을 고려하여 심각도 높은 Coding Rule 선정

## Bug Class 선정

- 시스템에 치명적인 영향을 주는 Bug List 위주로 항목 선정

## 신뢰성 높은 MISRA Coding Rule 기반

- 141개중 108개의 Rule 선정

## 위험도 높은 Bug class

- Defect 종류 선정

# 실제 사례 IV (현대자동차 개발 프로세스)

mds technology

## 효 과

- 테스팅의 자동화로 인해 전체 **개발 비용 감소**
- 개발프로세스의 구현단계부터 테스팅을 통한 **품질 향상**
- 개발자에 대한 학습 효과에 따른 **역량 향상 및 반복적인 에러 감소**
- 초기단계에서부터 소프트웨어의 **품질을 관리** 가능

## 한 계 점

- **실제 타겟 기반의 동적 분석과 이외의 테스팅 단계도 매우 중요**
- 정적 테스트는 언어적 결함 위주이며 기능상의 오류와는 별개임

# 실제 사례 V (노틸러스 효성)

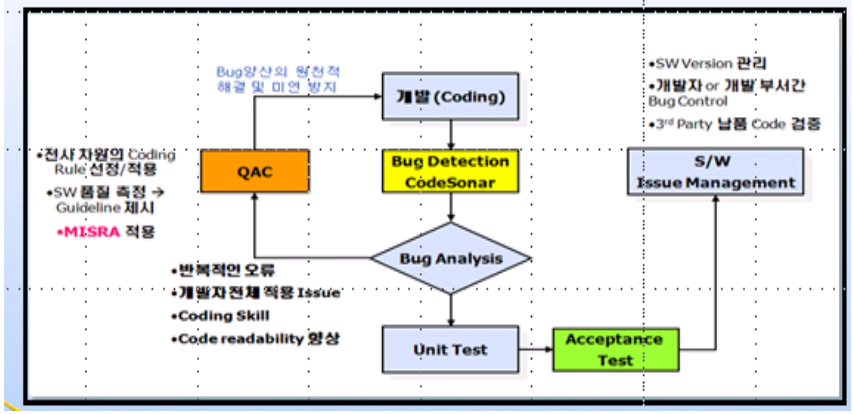
| 구분 | 내용 |
|---|---|
| **목적** | • **자동화 기기의 SW 품질 향상**<br>• 개발자의 Coding Skill 강화<br>• Source Code의 이해 및 **유지보수성 향상** |
| **활용 내역** | • QAC/C++의 Coding Rule 선정 및 적용<br>• Build 시마다 **CodeSonar**를 통한 Bug 검출<br>• **QAC/C++ 및 CodeSonar의 결과 Report**가 있어야 품질팀에서 **SW Release 진행** |
| **효과** | • 대형 외국 은행의 품질 기준 만족을 통한 ATM 기기 성공적인 수출.<br>• 적용 후 적은 Bug 및 빠른 Debug로 인한 개발 시간 단축<br>• 사용의 편리성으로 인한 개발자들의 추가 라이센스 요청 |
| **MDS의 역할** | SW 품질향상 컨설팅<br>- 주요 제품의 Bug 및 SW 품질 평가<br>- 각 모듈별 코딩 표준 및 평가 지표 선정<br>- 임베디드 코딩 기법에 대한 개발자 교육<br>- SW 품질 평가 프로세스 제시 |

◆ 품질 개선 지표

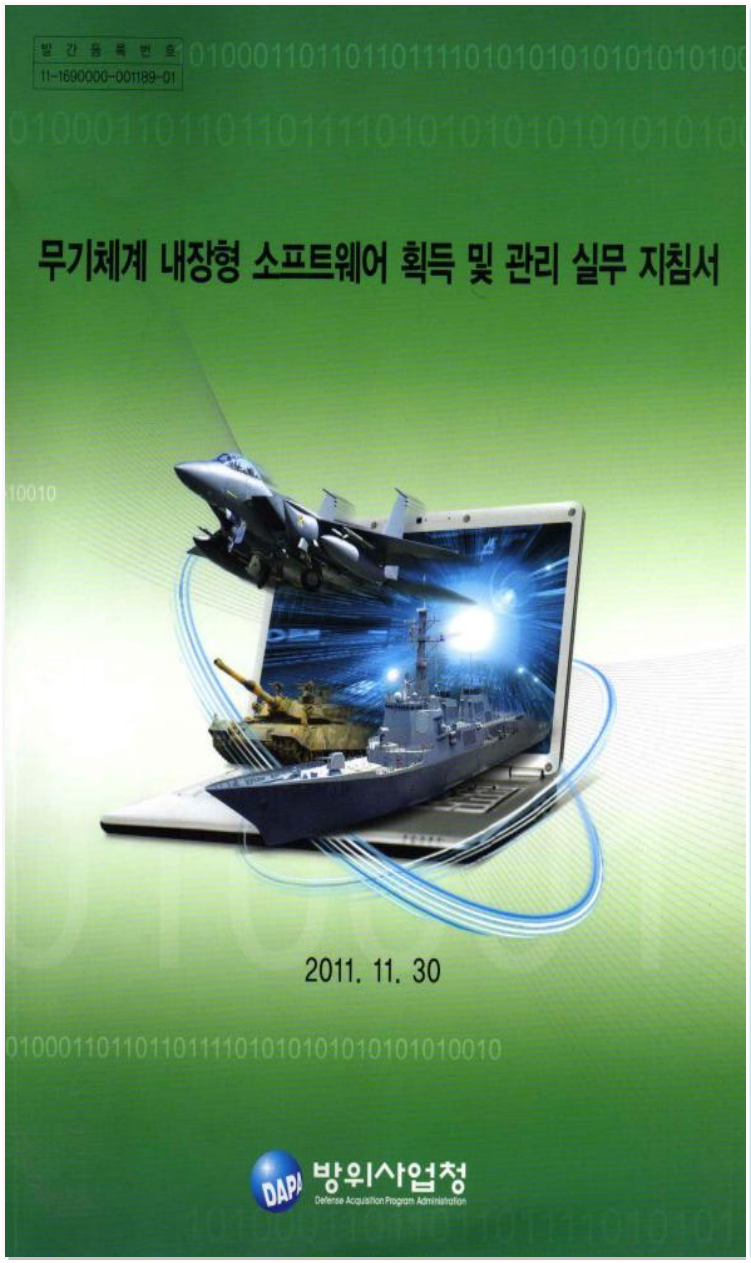
| group | 항목 | 권고수치 | 적용결과 초과 | 적용결과 미달 |
|---|---|---|---|---|
| 신뢰성 | Cyclomatic Complexity (STCYC) | | | |
| | Number of executable lines (STXLIN) | | | |
| | Static Path Count (STPTH) | | | |
| 유지 보수성 | Nesting of Control Structure (STMIF) | | | |
| | Number of Maintainable Code Line (STLIN) | | | |
| 재사용성 | Number of Gotos (STGTO) | | | |
| | Coupling to other classes (STCBO) | | | |
| | Response for class (STRFC) | | | |



◆ 품질 관리를 위한 프로세스

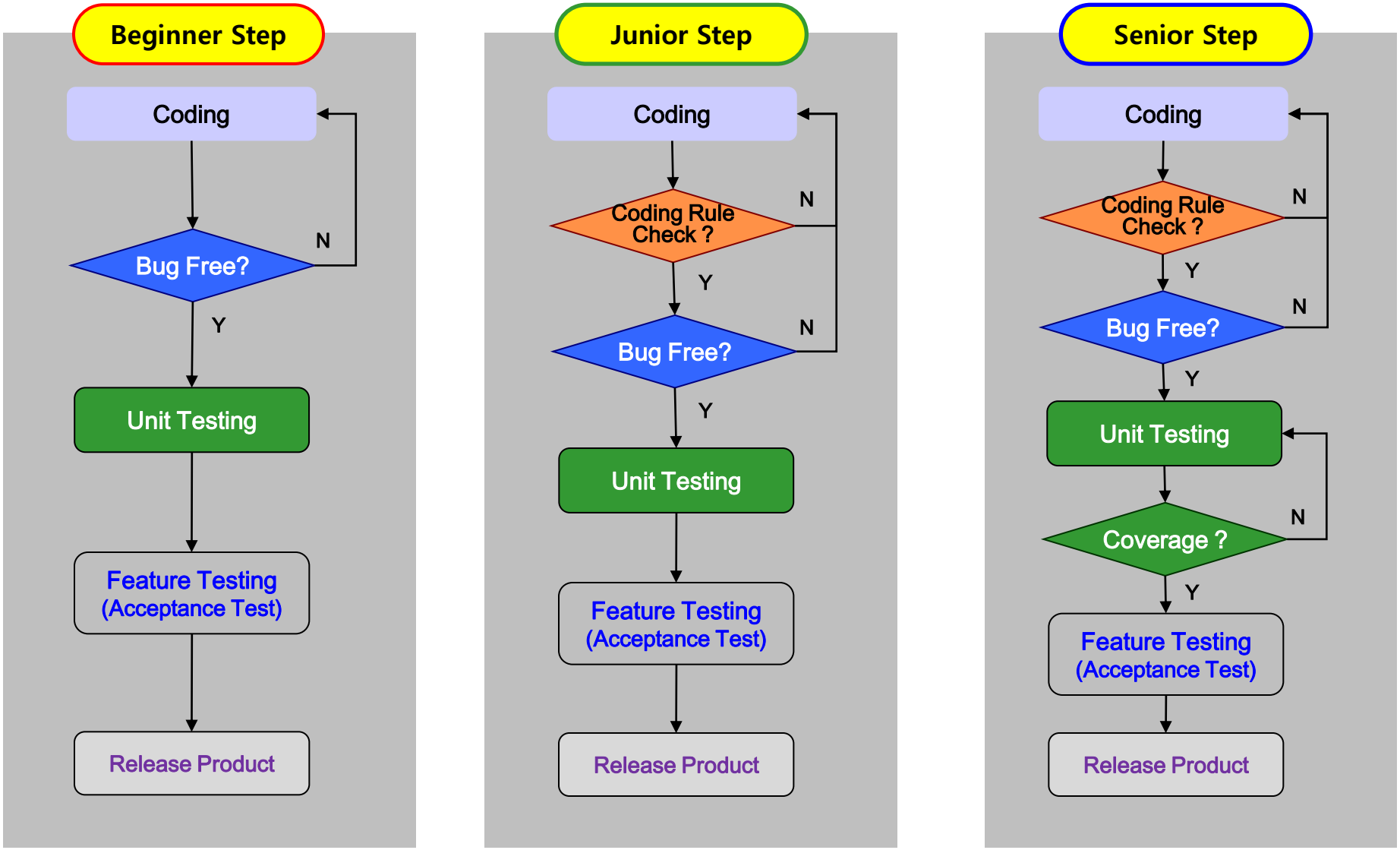


# 실제 사례 VI (방위사업청 무기개발 지침서)




SW신뢰성 시험 평가 추가

# S/W 품질 향상을 위한 개발 Process (요약)



# 결함 검출 도구 - CodeSonar

mds technology

CodeSonar™ 는 Inter Procedural 방식으로 결함을 검출하는 Static Analysis Tool입니다.

CodeSonar™ 는 수백만 라인 이상의 소스 코드의 검증이 가능합니다.

CodeSonar™ 는 프로그램 실행 시 발생 가능한 매우 다양한 결함을 빠른 시간에 검출합니다.
( 평균 Full Compile 시간의 1.5~ 2.5배 )

CodeSonar™ 는 Target 없이도 SW의 결함을 찾을 수 있습니다. (Only Source Code)

CodeSonar™ 는 버그의 발생지와 근본 원인을 쉽게 추적할 수 있으며, HTML 형식으로 Reporting하므로 팀원간 공유가 편리합니다.

# Supported Environment

mds technology

## ▪ Supported Languages

- C
- C++

## ▪ Supported Platforms

- Linux
- Windows
- Solaris

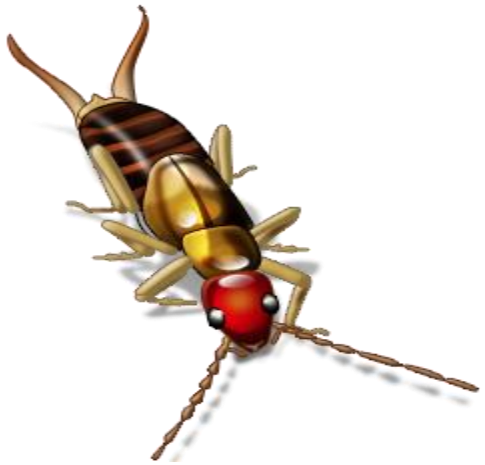
## ▪ Supported Compilers

- GCC and G++
- MS Visual Studio
- ARM cc
- Wind River compilers
- Green Hills compilers
- Other cross compilers
- Many More...

# CodeSonar의 결함 검출 기능

시스템에 치명적인 영향을 주는 결함 을 비롯하여 115가지 Bug 항목을 검출합니다. (Version 3.7 기준)

- Buffer Overrun
- Buffer Underrun
- Division by Zero
- Null-Pointer Dereference
- Uninitialized Variable
- DeadLock
- Free Non-Heap Variable
- Null Test after Dereference
- Format String Problem



- Type Overrun
- Type Underrun
- Memory Leak
- JPL POW10 Rule (MISRA Partial)
- Excessive Stack Depth
- Potential Unbounded Loop
- Use after Free
- BSI Security Check

# CodeSonar Result Page (Version 3.7)

mds technology

**Null Pointer Dereference** at tetris.cpp:787 *No properties have been set.* | edit properties
Jump to warning location ↓ warning details...

Show Events | Options

**add_src_nodes** *(c:\a 문서\codesonar 문서\codesonar_test_source\tetris_m\tetris\tetris.cpp)*

```
774     int add_src_nodes(int nodes, int b)
775     {
776
777             int acquire_err = REG_NOERROR;
778             char* value = (char*)malloc(sizeof(char));
779 [–]         char* state = acquire_state(&acquire_err, nodes);
```

**acquire_state** *(c:\a 문서\codesonar 문서\codesonar_test_source\tetris_m\tetris\tetris.cpp)*

```
753     char* acquire_state(int *err, int nodes)
754     {
755             char* node = NULL;
756             if(nodes == 0)
757             {
758                     *err = REG_NOERROR;
759                     return NULL;
```

⚠ Event 2: acquire_state() returns 0, which evaluates to NULL.
- Dereferenced later, causing the null pointer dereference.

▲ ▼ hide

```
780
781             if(acquire_err != REG_NOERROR)
782             {
783                     printf("REG_ERROR_OCCURED");
784                     return acquire_err;
785             }
786
787             if(*state !=NULL)
```

**Null Pointer Dereference**
state is dereferenced here, but it is NULL.

The issue can occur if the highlighted code executes.

See related event 3.
Show: All events | Only primary events

info | def | **acquire_state**

**acquire_state is a function**

[–] Definition (all)
Tetris.cpp: 753, char* acquire_state(int *e

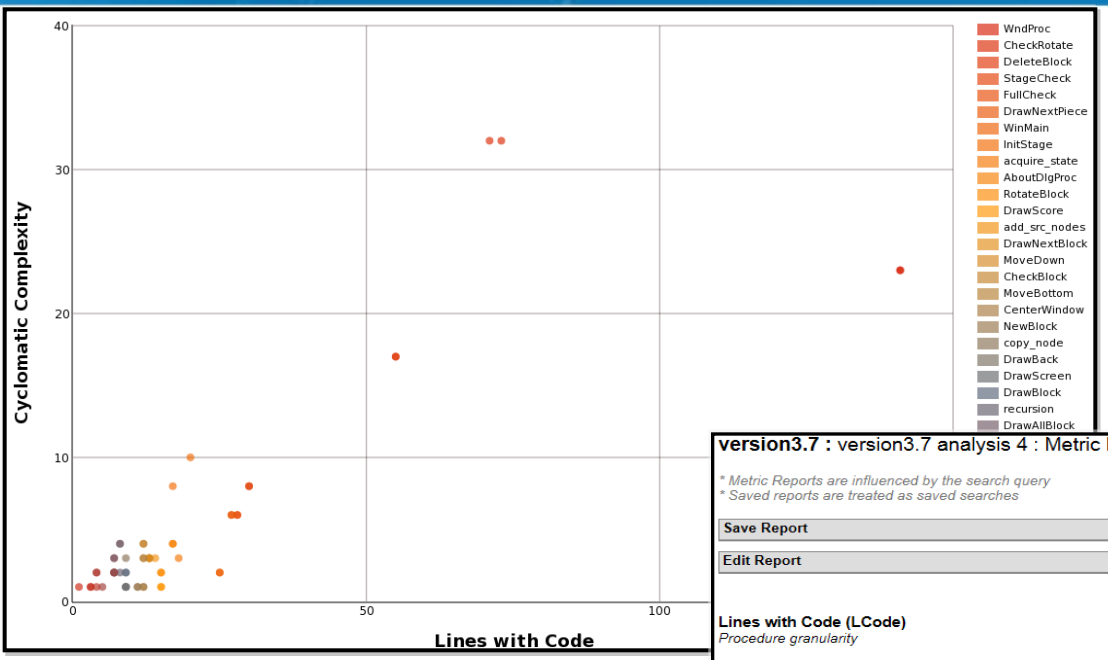
[–] Call (all)
Tetris.cpp: 779, char* state = acquire_stat
Tetris.cpp: 800, place = (int*)acquire_stat

[–] Metrics (all)

| | |
|---|---|
| Modified Cyclomatic Complexity (mvG): | 3 |
| Blank Lines (LB): | 1 |
| Mixed Lines (LMCC): | 0 |
| Lines with Code (LCode): | 18 |
| Code Lines (LCodeOnly): | 18 |
| Total Lines (TL): | 19 |
| Comment Lines (LComOnly): | 0 |
| Cyclomatic Complexity (vG): | 3 |
| Lines with Comments (LCom): | 0 |

문제 발생 시 Event에 대한 내용 Log를 상세하게 보여주며
문제 발생시 실행 라인을 녹색 음영으로 표시하여 문제의 근본원인 추적이 용이

# 함수별 Cyclomatic Complexity (Version 3.7)




**version3.7 :** version3.7 analysis 4 : Metric Report

** Metric Reports are influenced by the search query*
** Saved reports are treated as saved searches*

Save Report

Edit Report

Analysis Metrics: show

**Lines with Code (LCode)**
*Procedure granularity*

| File | Procedure | Value |
|---|---|---|
| Tetris.cpp | WndProc | 141 |
| Tetris.cpp | WndProc | 141 |
| Tetris.cpp | CheckRotate | 73 |
| Tetris.cpp | CheckRotate | 71 |
| Tetris.cpp | DeleteBlock | 55 |

add metrics | view entire table | compare with other metrics

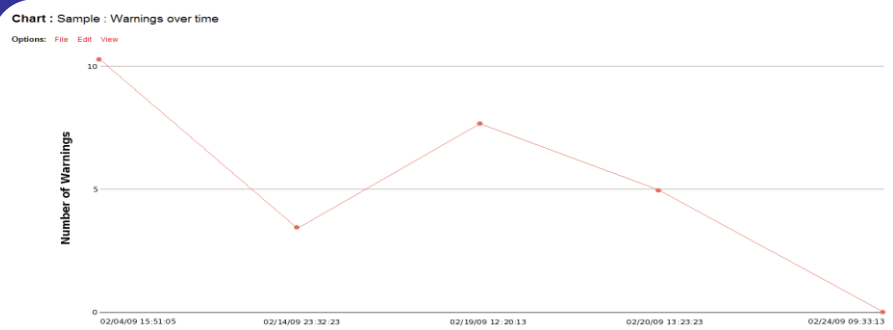
**Cyclomatic Complexity (vG)**
*Procedure granularity*

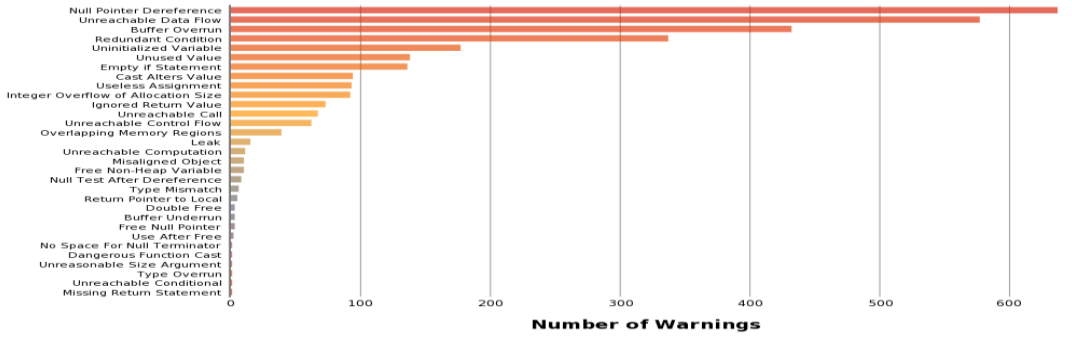
| File | Procedure | Value |
|---|---|---|
| Tetris.cpp | CheckRotate | 32 |
| Tetris.cpp | CheckRotate | 32 |
| Tetris.cpp | WndProc | 23 |
| Tetris.cpp | WndProc | 23 |
| Tetris.cpp | DeleteBlock | 17 |

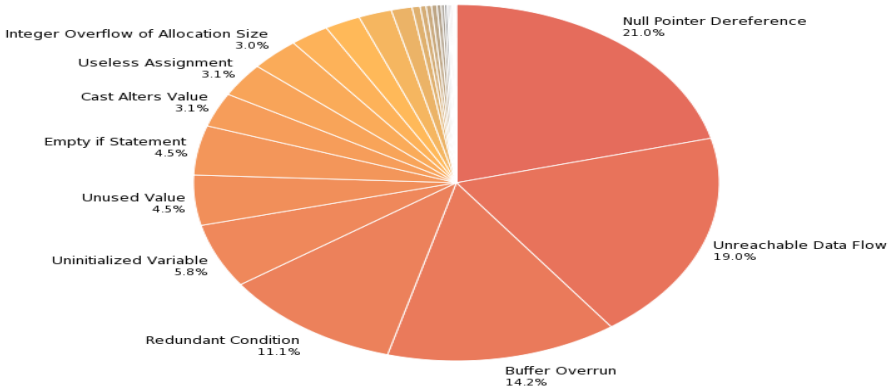
add metrics | view entire table | compare with other metrics

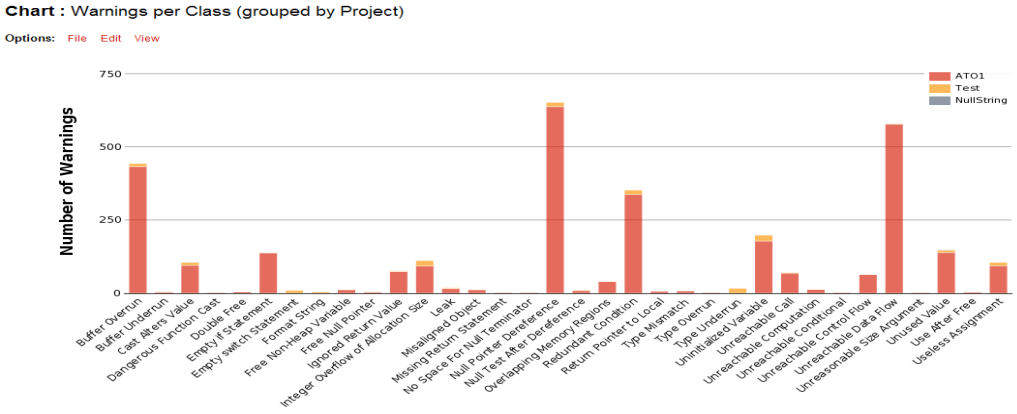
**함수별 Cyclomatic Complexity 값을 자동으로 제공하여 소스코드의 품질향상이 용이함**
**(복잡도 설정 시 특정 복잡도를 넘어가는 함수를 자동으로 검출)**

# CodeSonar Convenience Feature




각종 그래프로 프로젝트의 진행 상황을 편리하게 확인할 수 있습니다.
또한 프로젝트의 파일 수, Line수, 결함 변화도 차트로 확인 할 수 있습니다.

# CodeSonar - Primary Reference

mds technology

# Thank you!!


- TA 사업팀 우경일 팀장
  - Mail : kyungil@mdstec.com
  - TEL : 031-627-3140
  - H.P : 010-5123-6562



MDS테크놀로지㈜

463-400 경기도 성남시 분당구 대왕판교로 644번길 49 한컴타워 3층

Tel 031-627-3140 / Fax 031-627-3100 / www.mdstec.com