제22회 한국정보처리학회 추계학술발표대회 논문집 제11권 제2호 (2004. 11)

내장형 소프트웨어를 위한 성능 분석 레포팅 뷰어의 설계 및 구현

조용윤°, 박호병, 신경희, 유재우
숭실대학교 컴퓨터학과
{yycho, hbpark, khshin}@ss.ssu.ac.kr, cwyoo@computing.ssu.ac.kr.

A Design and Implementation of Performance Reporting Viewer for Embedded Software

Yong-yoon Cho, Ho-byoung Park, Kyoung-hee Shin, Chae-woo Yoo
Dept. of Computing, Soongsil University

요 약

내장형 소프트웨어 성능 평가 및 분석 도구는 제한된 자원에 효율적인 내장형 소프트웨어의 개발 속도와 신뢰성 향상을 위한 유용한 도구이다. 본 논문은 내장형 시스템 개발자가 쉽고 편리하게 성능 평가 결과를 분석하고 효율적인 소프트웨어 개발 기회를 보장받을 수 있는 GUI 형태의 성능 분석 레포팅 뷰어(reporting viewer)를 설계하고 구현한다. 제안하는 성능 분석 레포팅 뷰어는 내장형 소프트웨어에 대해 생성된 저수준의 성능 평가 로그를 API 수준의 자료구조로 변경하는 정보 변환기 모듈과 API 형태의 자료구조를 이용해 사용자가 원하는 결과를 GUI 형태로 출력하는 레포팅 생성기로 구성된다. 제안하는 성능 분석 레포팅 뷰어는 개발자나 사용자에게 그래픽 형태의 편리한 성능 분석 레포팅을 제공하며, 나아가 API를 통해 개발자 자신의 취향에 맞는 레포팅 화면의 구성과 개발 가능성을 제공할 수 있을 것으로 개대된다.

1. 서론

일반적인 소프트웨어와는 달리 내장형 소프트웨어는 개발자가 적은 자원을 가지고 있는 타겟에서 CPU나 메모리와 같은 자원을 얼마나 소비하고 어느 정도의 성능을 발휘하는지 파악하는 것이 매우 중요하다. 따라서, 내장형 소프트웨어에 대한 성능 평가 및 결과 분석은 제한된 내장형 시스템 자원에 효율적인 내장형 소프트웨어 개발이 효율성 향상에 기여할 수 있다. 보통 내장형 소프트웨어의 개발은 호스트/타겟 방식에 의한 교차 개발(cross development) 환경을 통해 이루어진다. 따라서, 성능 평가를 위한 실제 프로세스 테스팅은 타겟에서 실행되며, 그 결과는 호스트에서 적절한 분석 방법에 의해 사용자에게 레포팅 되어야 한다.

그림 1은 내장형 소프트웨어의 성능 분석을 위한 일반적인 호스트/타게 방식의 교차 개발 환경을 나타낸다.

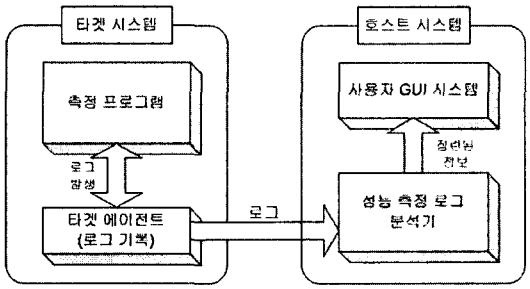

그림 1. 호스트/타겟 방식의 내장형 소프트웨어를 위한 성능 분석 시스템의 전체 구조도

이때, 타겟에서 성능 테스팅을 위해 실시되는 실험은 개발 소프트웨어에 있는 각 함수의 CPU와 메모리(스택, 힙, 텍스트) 사용량 과 타겟 전체의 메모리 사용량 등과 같은 자원 사용 정보이다. 성능 측정 정보는 호스트에서 테스팅 결과 정보의 판독성 향상과 이용 효율성을 위해 다양한 GUI 형태로 제공될 수 있다. 다양한 성능 측정 정보 보고기(reporter) 또는 뷰어(viewer)는 저 수준 텍스트 기반의 성능 측정 정보를 GUI 형태로 제공하기 위해 성능 측정 도구에 포함될 수 있다. 또한, 이러한 GUI 형태의 성능 테스트 결과 보고기 또는 뷰어는 개발자나 사용자의 성향이나 소프트웨어의 특성에 따라 다양하게 재 작성될 수 있는 API 형태로 제공되는 것이 바람직하다. 지금까지 소프트웨어의 실행 상태를 로그로 저장하여 저장된 로그를 개발자가 직접 분석하는 방법으로 GUI 형태의 보고기 또는 뷰어를 생성하였다. 하지만, 이 방법은 개발자가 가공되지 않은 저 수준의 정보를 직접 분석하여 성능을 측정해야하는 어려움이 있고, 개발비용의 증가를 가져올 수 있다. 그러므로 사용자의 기호에 따른 다양한 형태의 성능 측정 정보 보고기 또는 뷰어의 생성을 위해 원래의 저 수준 테스트 결과 자료는 보다 유연하고 편리하게 가공될 수 있는 형태로 변환되어 제공되어야 한다. 개발자나 사용자의 시각에서 그래픽 형태의 직관적이고 종합적인 성능 측정 정보는 성능 평가에 대한 신뢰성 판단에 빠른 결정을 유도할 수 있어 내장형 소프트웨어 개발비용을 감소시킬 수 있다. 본 논문은 가공되지 않은 저수준의 내장형 소프트웨어에 대한 성능 측정 결과를 다양한 그래픽 형태의 결과로 출력하고 사용자의 기호에 맞는 결과 출력 뷰어의 생성을 지원하기 위해 API 수준의 인터페이스를 제공하는 성능 측정 레포팅 뷰어를 제안한다. 제안하는 성능 측정 레포팅 뷰어는 저수준의 성능 측정 결과 로그를 분석하는 프로파일 로그 분석기와 프로파일 로그 분석기로부터 가공된 결과 정보를 분류하고 GUI형태로 쉽게 변경할 수 있도록 API 형태로 제공하는 프로파일 변환기로 구성된다.

본 논문은 2장 본론에서 제안하는 뷰어의 구성에 대해 알아보고 3장에서 제안한 뷰어를 통한 테스팅 결과에 대한 레포팅 실험에 대해 알아본고 4장 결론 및 향후 연구방향으로 맺는다.

2. 본론

일반적으로 내장형 응용프로그램의 개발은 호스트/타겟 방식에 의한 교차 개발(cross development) 방법을 사용한다. 그러므로 제안하는 성능 측정 레포트 뷰도 이와 같은 구조에 포함되어 설계되어야 한다. 그림 2는 일반적인 성능 측정 로그 분석기를 포함하는 내장형 소프트웨어 성능 분석 시스템의 전체 구조를 보여준다.

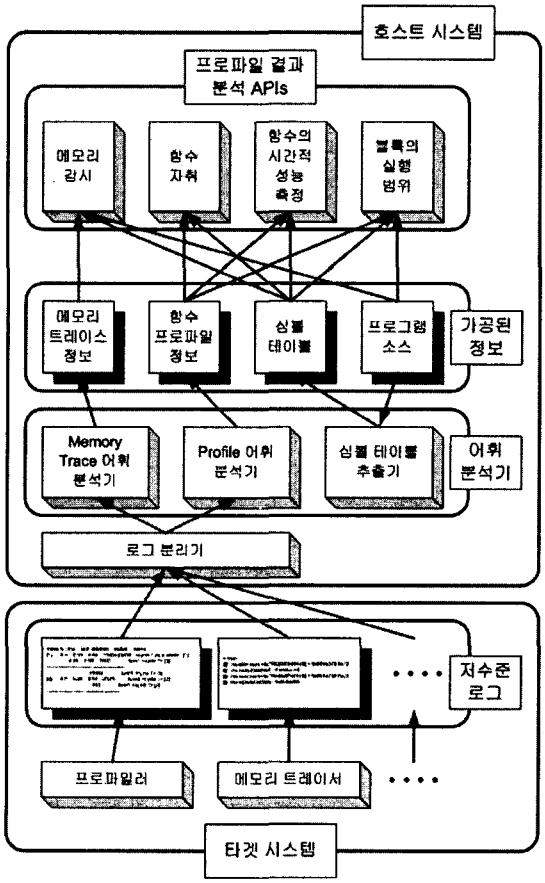

그림 2. 타겟/호스트 시스템 성능 측정 로그 분석기 전체 구조도

타겟 시스템은 측정 프로그램이 실행되고 타겟 에이전트를 통해 가공되지 않은 저수준의 로그가 기록된다. 타겟 에이전트에서 생성된 로그는 성능 측정 로그 분석기를 통해 정련되어 사용자 GUI 시스템에게 전달된다.

성능 측정 로그 분석기는 프로파일 분석기와 프로파일 결과 분석 API로 이루어져 있고, 타겟/호스트 구조에서 호스트에 위치한다. 타겟 시스템은 내장형 시스템으로써 자원이 풍부하지 않아, 성능 측정 로

그 분석기는 측정 프로그램의 성능 측정에 영향을 주지 않도록 자원이 풍부한 범용 컴퓨터인 호스트 시스템에 위치해야 효율적이다. 프로파일 결과 분석 API는 사용자에게 메모리에 관한 정보와 함수의 자취에 관한 정보, 블록의 실행 범위에 관한 정보, 함수의 시간적 성능에 관한 정보를 제공한다. 메모리에 관한 정보는 메모리의 할당과 해제에 관한 로그를 분석하여 메모리 누수와 메모리 사용량에 관한 정보를 분석한다. 메모리의 누수는 소프트웨어의 메모리 자원을 불필요하게 소모하는 것으로 메모리 자원이 적은 내장형 시스템에서 치명적이 오류가 된다. 함수의 자취에 관한 정보는 재귀적으로 함수가 호출되는 응용프로그램에서 외부 입력에 따른 함수 호출 경로를 파악할 수 있는 정보가 된다. 그리고 블록의 실행 범위에 관한 정보는 실행되지 않은 블록이 소스 코드에 삽입되어 실행되지 않는 기계어 코드를 제거하는 정보가 된다. 마지막으로 함수의 시간적 성능에 관한 정보는 전체 소프트웨어의 시간적 성능을 모듈별로 측정할 수 있는 정보가 된다.

프로파일 분석기는 세 부분으로 나누어져 있다. 첫째는 로그 분리기이고, 둘째는 어휘 분석기 이며, 셋째는 로그 변환기이다. 본 논문은 리눅스 기반의 내장형 시스템에서 GNU 툴과 메모리 할당/해제를 감시하는 메모리 트레이서를 통해 생성된 저수준의 프로파일 정보를 가공하고 분석하는 방법을 소개한다.

프로파일 분석기의 첫 번째 단계인 로그 분리기는 혼합되어 있는 로그를 종류에 따라 분리한다. 예를 들어 프로파일러로부터 생성된 저수준의 로그에는 함수의 호출 횟수와 발생한 재귀적으로 발생한 사이클, 실행된 시간등과 같은 정보를 포함하고 있다. API 수준의 인터페이스를 제공하기 위해서는 이 정보를 따로 분리하여 분석하여야 한다. 타겟은 호스트에서 서비스를 요청하면 시리얼이나 이더넷을 통해 호스트로 성능 평가 정보를 보내준다. 그러나 발생된 모든 성능 테스트 결과는 저수준의 텍스트 기반 정보이기 때문에 개발자가 직관적으로 성능 테스트 결과 정보를 파악하기는 어렵다. 즉, 실시간 에이전트를 통해 발생한 테스트 결과에 대해 개발자가 직접 대화형 쉘과 자원 모니터를 통해 분석해야 한다. 개발자가 직접 분석하는 방법은 응용프로그램 개발 이외의 수고를 들게 하여 응용프로그램의 개발을 비효율적으로 만드는 약점이 있다. 다음 그림 3은 제안하는 성능 평가 레포팅 뷰어의 구성요소를 나타낸다.

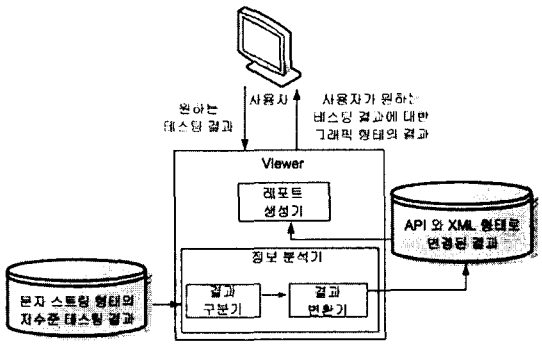

그림 3. 성능 평가 레포팅 뷰어의 구성

레포팅 뷰어는 정보 분석기와 레포트 생성기로 구성되어진다. 정보 분석기는 문자 스트링 형태의 저수준 테스팅 결과를 입력으로 받아 생성된 테스팅 결과 종류에 따라 분류하는 결과 구분기와 구분된 토큰 스트림(token stream) 형태의 결과 정보를 레포트 생성기에서 사용자의 요구에 따른 레포트 화면 출력을 위해 사용하기 위한 API형태로 변경하는 결과 변환기로 구성된다. 또한, 레포트 생성기는 정보 분석기에 의해 생성된 API 수준의 구분된 테스팅 결과 정보를 이용해 사용자의 요구에 맞는 그래픽 형태의 결과 레포트를 출력한다.

3. 구현 및 실험

제안하는 성능 평가 레포팅 뷰는 Java를 이용해 구현되었으며, 생성되는 API는 Java 클래스이다. 레포팅 뷰의 정보 분석기에서 사용하는 문자 스트링 형태의 저수준 테스팅 결과는 다음 표 1과 같은 4가지 성능 평가 항목에 대해 실시한 결과이다.

표 1. 실험에 사용되는 테스팅 범위의 종류

테스팅 범위	기 능
Tracing 테스팅	타겟 기반의 Runtime Tracing 결과를 UML의 Sequence Diagram형태로 나타낸다
Memory 테스팅	Heap 공간에 설정된 메모리 누수를 찾아낸다.
Performance 테스팅	Function 및 Method의 수행을 나타내어 어플리케이션의 병목지점을 찾아낸다.
Code Coverage 테스팅	Code 수행에 대한 적용 범위를 완벽하게 분석한다.

그림 4는 예제 C 코드에 대한 레포팅 뷰어의 초

기 빌드 구성 화면이다.

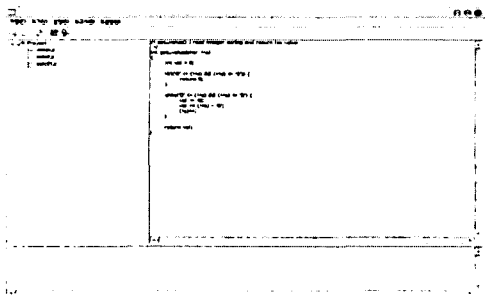

그림 4. 임베디드 소프트웨어 성능 평가 뷰어

예제 C 코드는 일반적인 이동 단말기에서 많이 쓰이는 계산기 프로그램의 간단한 예이다. 3개의 모듈로 이루어져 있으며, 이들을 실행 파일로 컴파일하여 타겟에서 실행한다..

표 2는 그림 4에서 컴파일을 통해 실행 파일로 생성된 예제 C 코드에 대해 타겟에서 실시된 성능 테스팅에 대한 문자 스트링 형태의 저수준 결과의 예제 일부분이다.

표 2. 가공되지 않은 저수준의 로그

```
index % time    self  children    called     name
                                    95000           func1 <cycle 1> [3]
[2]     0.0    0.00    0.00     95000         func2 <cycle 1> [2]
                                      900           func1 <cycle 1> [3]
-----------------------------------------------
                                      900           func2 <cycle 1> [2]
                0.00    0.00    1000/1000         main [13]
[3]     0.0    0.00    0.00     1900          func1 <cycle 1> [3]
                                    95000           func2 <cycle 1> [2]
-----------------------------------------------
Index by function name
   [3] func1                   [2] func2                   [1] <cycle
1>

@ <Location>:(Called function + Called Location)[Instruction Location]
+/- Address Size
= Start
@ ./ex-n:(mtrace+0x169)[0x80484e9] + 0x804a378 0x12
@ ./ex-n:[0x8048596] - 0x804a378
```

표 2는 프로파일러와 메모리 트레이서가 타겟 시스템에서 생성한 저수준 정보이다. 생성된 정보는 네트워크를 통해 호스트 시스템으로 이동한다. 이동한 정보는 정보 분석기의 정보 변환기를 통해 API로 변환되어 사용자에게 그래픽 형태의 레포트를 제공한다. 다음 그림 5는 예제 C 코드에 대해 사용자의 메모리 관련 레포트 요구에 대한 뷰어가 제공하는 그래픽 형태의 출력 예이다.

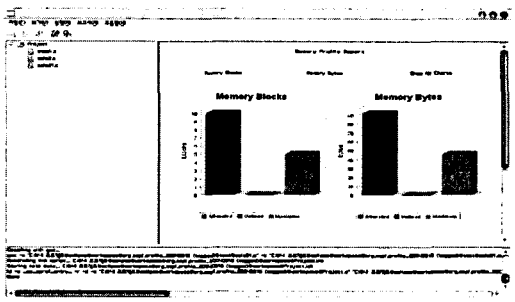

그림 5. 메모리 테스팅 결과

4. 결론 및 향후 연구 방향

본 논문은 제한된 내장형 시스템 자원에 대해 효율적인 내장형 소프트웨어 개발을 위해 실시되는 성능 평가 테스트 결과를 편리한 그래픽 형태의 레포트로 제공하는 테스팅 결과 레포트 뷰를 제안했다. 제안한 뷰를 이용해 개발자나 사용자는 GUI형태의 결과 레포트를 통해 빠르고, 직관적인 결과 분석이 가능하며, 임베디드 소프트웨어의 개발 시간과 노력을 줄여주어 임베디드 관련 소프트웨어 개발의 효율성을 증가 시킬 것으로 기대된다.

향후 본 논문에서 제안한 레포팅 뷰어는 웹 기반의 뷰어로 개발되어 인터넷을 통해 환경에 독립적인 테스팅 결과 분석이 가능하도록 연구되어야 한다.

참고문헌

[1] 공기석, 손승우, 임채덕, 김홍남, "내장형 실시간 소프트웨어의 원격 디버깅을 위한 디버그에이전트의 설계 및 구현", *한국정보과학회 가을 학술발표논문집 Vol. 26. No 2*, pp.125~127, 1999.

[2] Dr. Neal Stollon, Rick Leatherman, Bruce Ableidinger, "Multi-Core Embedded Debug for Structured ASIC Systems", proceedings of DesignCon 2004, Feb, 2004.

[3] K. Weiß, T. Steckstor, C. Nitsch, W. Rosenstiel, "Performance Analysis of Real-Time-Operation Systems by Emulation of an Embedded System". 10th IEEE International Workshop on Rapid System Prototyping (RSP) Florida, USA, 1999.