OJL によるトレースログ可視化ツールの開発

350702101　**後藤 隼弐**

要旨

要旨

Development of Visualization Tool for Trace Log by OJL

350702101 Junji Goto

Abstract

Abstract

修 士 論 文

OJLによるトレースログ可視化ツールの開発

350702101　後藤 隼弐

名古屋大学 大学院情報科学研究科

情報システム学専攻

2009年1月

目 次

第1章　はじめに

1.1　開発背景

近年，PC，サーバ，組み込みシステム等，用途を問わずマルチプロセッシングシステムの利用が進んでいる．その背景には，シングルプロセッサの高クロック化による性能向上効果の停滞や，それに伴う消費電力・発熱の増大がある．マルチプロセッシングシステムでは処理の並列性を高めることにより性能向上を実現するため，消費電力の増加を抑えることが出来る．組み込みシステムにおいては，機械制御と GUI など要件の異なるサブシステム毎にプロセッサを使用する例があるなど，従来から複数のプロセッサを用いるマルチプロセッサシステムが存在していたが，部品点数の増加によるコスト増を招くため避ける方向にあった．しかしながら，近年は，1 つのプロセッサ上に複数の実行コアを搭載したマルチコアプロセッサの登場により低コストで利用することが可能になり，低消費電力要件の強い組み込みシステムでの利用が増加している．

マルチコアプロセッサ環境でソフトウェアを開発する際に問題になる点として，デバッグの困難さが挙げられる．これは，マルチコアプロセッサ環境では，従来のブレークポイントやステップ実行を用いたデバッグの挙動が，シングルプロセッサ環境における場合と異なるからである．たとえば，複数のコアで実行される可能性のあるコードにブレークポイントを置きデバッグを行う場合，ブレーク後にステップ実行を行い処理を追う最中に，他のコアのブレークにより中断される可能性があるため，ブレーク後にブレークポイントを一時的に削除する必要があり，頻繁にブレークポイントの設置，削除を行わなければならなくなる．また，マルチコアプロセッサ環境では，特殊な状況下でのみ起こるバグが発生する可能性がある．その場合，原因を特定するのは通常困難である．なぜならば，マルチコアプロセッサ環境では，特別な制御を行わない限り各コアが非同期的に並列動作するため，バグの発生を再現することが難しいからである．また，再現が可能であったとしても，原因を特定するためには，バグが発生するまでの過程をすべてのコアの実行状況を監視しながら行う必要があり，シングルプロセッサ環境の場合に比べ非常に煩雑になる．

一方，マルチコアプロセッサ環境で有効なデバッグ手法としてトレースログの解析によるデバッグがある．これは，プログラムの実行中に，デバッグの判断材料となる情報をログとして出力することによりプログラムの動作を確認する手法である．ログの出力元としては，OS や ICE，シミュレータなどがあり，情報の粒度もアプリケーションレベル，タスクレベル，カーネルレベル，ハードウェアレベルなど様々である．

しかしながら，トレースログを開発者が直接扱うのは困難である場合が多い．これは，ログの情報の粒度が細かくなるほど単位時間あたりのログの量が増える傾向にあり，膨大なログから所望の情報を探し出すのが困難であることや，各コアのログが時系列に分散して記録されるため，逐次的にログを解析することが困難であるからである．そのため，トレースログの解析を支援するツールが要求されており，ログを解析し統計情報として出力したり，可視化表示することで開発者の負担を下げる試みが行われている．

既存のトレースログを可視化表示するツールとしては，ICE 付属のデバッガソフトウェアや，各 OS のシステムモニタリングツール，波形出力用ツールの流用などがある．しかしながら，これらは，有料であったり，ターゲット OS，ターゲット CPU が限定されていたり，可視化表示項目，形式が固定であるなど，汎用性，拡張性の面で制限が多い．

こういった背景を鑑み，我々は，ログの形式非依存化，可視化表示項目のプラグイン化というアプローチで先例の制限を解決したトレースログ可視化ツール TraceLogVisualiser を開発した．

1.2 開発目的

1.3 論文の構成

第2章　既存のトレースログ可視化ツール

2.1　ICE付属デバッガソフトウェア

2.2　統合開発環境

2.3　波形出力ツールの流用

2.4　既存のトレースログ可視化ツールの問題点

第3章　トレースログ可視化ツール TraceLogVisualizer の設計

3.1　開発方針

3.2　トレースログの抽象化

3.2.1　標準形式トレースログ

3.3　可視化の対象と単位

3.4　可視化方法

第4章　トレースログ可視化ツール TraceLogVisualizer の実装

4.1　TraceLogVisualizer のプロセス

4.1.1　標準形式への変換

4.1.2　図形データの生成

4.2　TraceLogVisualizer のインターフェイス

4.2.1　Json 形式

4.2.2　トレースログファイル

4.2.3　リソースファイル

4.2.4　リソースヘッダファイル

4.2.5　変換ルールファイル

4.2.6　可視化ルールファイル

4.2.7　TLV ファイル

第5章　トレースログ可視化ツール TraceLogVisualizer の利用

5.1　組み込みRTOSのトレースログの可視化

5.2　マルチコア対応への拡張

第6章　開発スタイル

6.1　OJL

6.1.1　フェーズ分割

6.2　ユースケース駆動アジャイル開発

6.2.1　プロジェクト管理

6.2.2　設計

6.2.3　テスト

6.3　開発成果物

第7章　おわりに

7.1　まとめ

7.2　今後の展望と課題

OJLによるトレースログ可視化ツールの開発

350702101　後藤 隼弐