マルチターゲットのためのプログラミング・テスト支援の方式

7R-6

溝渕 順子 、 仙田 和彦 、 安田昭彦
(株) 東芝 府中工場

1.はじめに

東芝府中工場で生産されるソフトウェアは、産業用コンピュータシステムのものがほとんどを占める。
これらシステムのターゲットとなるマシンは、プロセスコンピュータ、マイコン、さらに最近ではエンジニアリングワークステーションなど、多種多様である。
当工場ではソフトウェア開発支援のための環境New-SWBを構築、試行中である[1]-[5]。本発表では、複数種類のターゲットマシンに対する開発環境統一と、少数のターゲットマシン共有化の二つを目的とする、プログラミング・テストの方式について検討する。

2. 従来の問題点

産業用コンピュータシステムのソフトウェアは、その対象プラントの多様さと規模の大きさから、次のような特徴をもつ。

- 開発機種が多い
- 開発期間が短い
- 開発要員が多い

このような状況で生じる問題点として、次のようなことがあげられる。

- ターゲットマシンによって開発環境が変わる
- 開発用ターゲットマシンが不足する

以上のような問題点に対する1つの対策として、複数種類のターゲットマシンに対するソフトウェア開発において、パソコンとエンジニアリングワークステーションを開発マシンとし、これらをネットワークで接続して統一化する方式を考える。

3.マルチターゲットプログラムテスト支援方式

3.1 ネットワークによる有機的結合

開発用マシン、ターゲットマシン上にそれぞれイーサネット系の標準的な接続プロトコルを用意し(一部自製)、お互いを有機的につなぐことで、資源の共有化をはかる。
またこれによって、次項で述べるようなUNIX以外のマシンでUNIXを使うなど、ツール開発量の削減にも効果をあげた。
ハードウェア構成を図1に示す。
開発用マシンは、当社のエンジニアリングワークステーションAS3000と、同じく当社のラップトップパソコンJ3100である。
今回ターゲットマシンとして考えたのは、プロセスコンピュータTOSBAC-G8000と、マイコンTOSMAP-S70、AS3000である。
これらのコンピュータを結ぶネットワークとして、イーサネットおよびパソコン間のstarLANを用いる。
また資源の管理は、ソースコードを開発用マシンで、オブジェクトコード、ロードモジュールはターゲットマシンで行う。

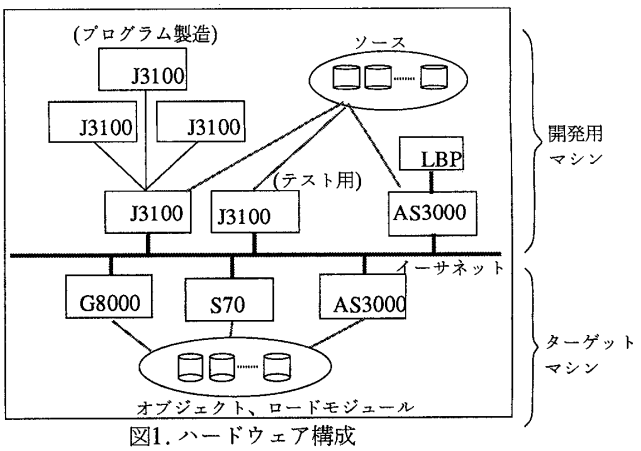

図1. ハードウェア構成

3.2 UNIX資源の有効活用

UNIXの持つ豊富な資源を充分に活用する。
AS3000が開発用マシンである場合はもちろん、J3100(MS-DOS)の場合も、リモート処理の方式でAS3000に対象ファイルを転送してUNIXコマンドを動作させ、結果を戻すことでUNIXコマンドを使えるようにする。
対象となるコマンドは、ソースコード解析のためのlint,cxref,wcなどである。
実際、静的解析ツールといっている内容の多くは、UNIXコマンドを動作させることで実現している。

3.3 共通インタフェースとテスト実行手順

開発用マシン上に、他のソフトウェア開発支援ツールとも共有する共通インタフェースをおく。
これによって、複数のツールで構成されるプログラミング・テスト支援ツール群を、作業手順に従って操作していくことができる。

---

an unified programming & test environment for various kinds of target machines
Junko Mizobuchi , Kazuhiko Senda , Akihiko Yasuda
Toshiba Corporation

テストの手順は次のようになる。
(1) エディタによるソースコード生成
(2) ソースコード解析によるコード監視
(3) メークファイルの自動生成
(4) ソースコード解析結果に基づいたスタブ生成
(5) リモートコンパイラによるターゲットマシンでのコンパイル、リンク
(6) リモートデバッガによるターゲットマシンでの単体・組み合せテスト
(7) カバレッジ・プロファイルデータの収集
(8) カバレッジ・プロファイル結果の確認

以上の処理は、すべて開発用マシンであるAS3000もしくはJ3100から行われるものである。

4.ソフトウェア構成とリモート処理

プログラミング・テスト支援を行うツール群を、図2に示す。

項目	内容
ユーザインタフェース	ツールメニューと結果表示
エディタ	フルスクリーンエディタ
静的解析	複雑度、リレーション
UNIXツール	UNIXコマンド実行
リモートコンパイル	ターゲットでのコンパイル
メークファイル生成	メークファイル自動生成
リモートリンク	ターゲットでのリンク
スタブ生成	スタブの自動生成
ロードモジュール自動生成	makeコマンド実行
リモートデバッガ	ターゲットでのデバッガ
カバレッジ	パスカバー度収集
プロファイル	実行頻度、時間収集
ファイル転送	ホスト・ターゲット間
プリンタ出力	リモートLBP出力
ターゲット接続	仮想端末機能

図2. ソフトウェア構成

4.1 静的テスト支援～ホスト処理方式

ターゲットマシンに関係なく作業を進められる、ソースコード生成から机上デバッグまでの段階を静的テスト(実際にプログラムを動作させることなくテストする)と位置づけ、この工程を支援するツールは、開発マシンセルフでの処理を行う。

これらはソースコードの情報を基に行われるもので、支援ツールとしては、ソースコードを解析してモジュールおよび変数の関係やモジュール複雑度を評価する静的解析、メークファイル自動生成、ソースコードを解析した結果を基にしたスタブの自動生成などである。

4.2 動的テスト支援～リモート処理方式

リモートコンパイラ・リンカ、リモートデバッガ、あるいはJ3100をホストにした時のUNIXツールなどは、インタフェースは開発用マシン、機能実現はターゲットマシンで行う。
これらの実現方式を、リモートデバッガを例に図3に示す。

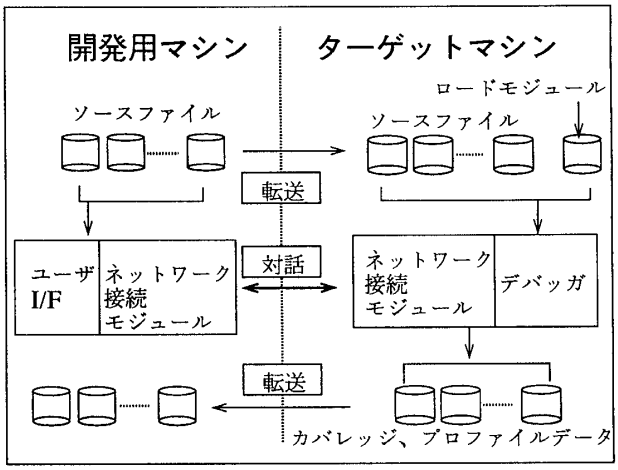

図3. リモートデバッガ実現例

いずれの場合も、ターゲットマシンでの処理に必要な資源を開発用マシンから転送し、開発用マシンでのインタフェース部とターゲットマシンでの実行部がネットワークを介して対話しながら処理をすすめる。

各種ターゲットマシン上には、それぞれネットワーク接続モジュールを用意すればよい。

5.おわりに

ソフトウェアテストの工程は、作業負荷が大きいことが知られていながら、ターゲットマシン志向が強く、統合化された環境としての整備は遅れている。

最終的なテストは、やはりターゲットマシン上で行う必要があるものの、その部分をなるべく少なくする工夫はされるべきである。

今回、エンジニアリングワークステーション、ラップトップパソコン、ネットワークを活用したプログラミング・テスト支援の方式を提案し、これに基づくツールの開発を行った。

今後、プログラミング・テストといった人手のかかる作業はラップトップパソコンで、ソフトウェア設計支援、構成管理とドキュメント管理などをエンジニアリングワークステーションでというように、New-SWBのソフトウェア開発支援ツールとの統合化を考慮しつつ、エンジニアリングワークステーション、ラップトップパソコンでの役割り分担を明確化していく。これにより、さらに生産性の向上が期待される。

参考文献

[1] 小野他:New-SWB大規模リアルタイムソフトウェア開発環境, 情報処理学会第37回(昭和63年後期)全国大会 3M-4。
[2] 建部他:New-SWBプログラミング支援ツール/EDTtools, 情報処理学会第37回(昭和63年後期)全国大会 3M-8。
[3] 山本他:Work Desk:New-SWB利用者インタフェース, 情報処理学会第37回(昭和63年後期)全国大会 3M-9。
[4] 小林他:New-SWBネットワーク構成と運用管理ツール, 情報処理学会第37回(昭和63年後期)全国大会 5F-5。
[5] 横山 他:New-SWBソフトウェア分散開発環境の構築事例,情報処理学会第38回(昭和64年前期)全国大会 。