2J-03　CORBA機能試験における試験スイート記述方式の検討

寺島 美昭、 今井 功、 木野 茂徳、 下間 芳樹
三菱電機（株）情報技術総合研究所

1. はじめに

近年、CORBA(Common Object Request Broker Architecture)など分散オブジェクト技術の利用により、アプリケーション API(Application Programming Interface)レベルの相互接続性を確保したソフトウェア開発が増加している。我々は CORBA サーバを実装した通信機器をブラックボックスとする機能試験を効率的に行う事を目的に、CORBA 機能試験手法[1]に関する研究に取り組んできた。本稿では試験ケースを定義する試験スイート記述に焦点を当て、課題の整理と解決について報告する。

2. CORBA 機能試験手法の概要

2.1 試験の位置付け

CORBA 機能試験手法は、通信機器に搭載される CORBA サーバに対して、試験クライアントが外部環境を摸擬する事により機能評価を実現する。特に異なる開発プロジェクトの試験実行者が、相互に試験資産を流用し試験コストを低減する事を狙い、様々な開発へ適用できる汎用性を考慮している。

2.2 手法の構成構成

これまでオブジェクト間のメッセージ通信を観測する事により、分散オブジェクトシステムを構成するクライアントとサーバ間の動作を評価する方法が研究されてきた[2]。この方法を基本に我々は、CORBA 機能試験手法として、試験スイートに基づいて試験クライアントが通信機器との間で CORBA API を介して通信する機能と、その結果として通信機器が起こす動作を観測する機能を規定した。例として図 1 に試験対象とする通信機器と試験クライアント間の API 規定である CORBA IDL(Interface Definition Language)定義を示す。この例は通信機器が搭載するデバイス AmpFan に対して、状態取得用 API である getStatus()と、パワーレベルを設定する setPowerLevel()を定義している。また、図 2 では試験対象である通信機器をブラックボックスとして図 1 に示す CORBA IDL により通信を行う構成を示している。

試験クライアントは試験スイート定義に基づく動作により外部環境を模擬し、この結果として試験対象の状態が遷移する。この遷移動作を試験クライアントの評価ポイントにて観測する事により、試験対象の内部アーキテクチャには手を加えず機能評価を実現できる。この観測ポイントは、試験クライアントの全ての CORBA 動作情報を捕らえるために、GIOP(General Inter-ORB Protocol)処理部を設定している。

```
// CORBA IDL Example
interface AmpFan {
  long getStatus(in long index, out long status)
  long setPowerLevel(in long level)
}
```

図 1 CORBA IDL による API の規定例

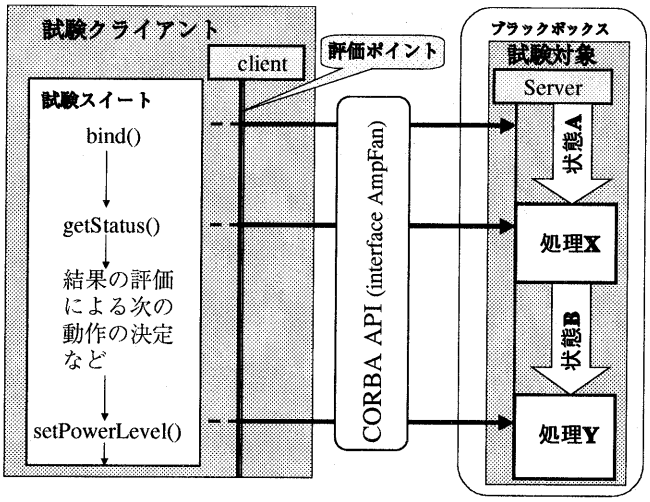

図 2 CORBA によるブラックボックス試験構成

3 試験スイート記述方法の規定

3.1 課題の設定

試験クライアントが試験に必要な様々な動作を通信機器に与えるために、試験スイートの定義には、高い記述能力が必要である。以下に試験スイートとして必要な課題を整理し、インターネット環境での利用において完成度、普及度の高いスクリプト言語である JavaScript による実現を検討した。

- 課題 1：CORBA IDL に基づくメソッド呼び出し、及びリターン情報、out データを観測し、その内容を評価する複雑なロジックを記述できる。
- 課題 2：試験動作が実運用時と同等の性能にて動作できる。

A study of testing suite description methods for function evaluations of communication equipments based on CORBA, Yoshiaki Terashima, Isao Imai, Shigenori Kino, Yoshiki Shimotuma, Information Technology R&D Center, Mitsubishi Electric Corp.

- 課題 3：試験スイートの再利用など、プロジェクトに跨る試験資産の有効利用ができる。

3.2 JavaScriptの拡張

設定した課題を解決するために、JavaScript に対して、以下の拡張を行った。

(1) CORBA IDL→JavaScript メソッドマッピング

JavaScript から CORBA API の呼び出しを行うために、CORBA IDL→JavaScript メソッドマッピング方法を定義した。ここでは CORBA IDL 呼び出しを①Interface Repository(IR)情報から動的に組み立てる方法[3]と、②静的にマッピングを決定する方法を比較した。この結果、①は処理時間が大きく課題 2 を満たせない可能性がある事、また通信機器の試験では CORBA IDL 試験を予め限定できる事から、②の方法を採用した。このため図 3 に示す機械的な変換ルールを規定している。

```
CORBA IDL Example
Interface AmpFan {
    long getStatus(in long index, out long status)
}
```

↓

```
JavaScript Example
AmpFan_getStatus(in long index, out long status)
```

図 3 CORBA IDL→JavaScript メソッドマッピング例

JavaScript は、CORBA に規定される全てのデータ型に対するデータ型を備えていない。このため課題 1 に対して、対応するデータ型が存在しない CORBA IDL の論理型と文字列型以外は、JavaScript の数値型へマッピングし、さらに Java マッピングによるデータ型へ変換する事により通信を行う。

(2) 試験共通ライブラリの規定

課題 3 に対応するために試験スイート開発に対する記述者の負担軽減、試験結果の正確な評価、及び通信機器固有のアーキテクチャ依存環境構築の局在化を目的に、以下の機能を試験共通ライブラリとして検討した。

- CORBA IDL 対応関数：既に図 3 に基づいて説明した CORBA IDL→JavaScript マッピングにより必要となる JavaScript 関数を用意する。
- CORBA ドライバ：CORBA IDL→JavaScript メソッドマッピングによる CORBA IDL 呼び出しは、CORBA IDL 対応関数から、Java マッピングを行う CORBA ドライバを介して通信する。

3.3 試験環境の構築

通信機器アーキテクチャ毎に必要な試験クライアントの生成手続きを図 4 に示す。ここでは個々の通信機器アーキテクチャ依存の機能を吸収する CORBA IDL 対応関数と CORBA ドライバを、CORBA IDL から CORBA IDL Compiler、及び規定した CORBA IDL→JavaScript メソッドマッピングによる変換ルールを実行する CORBA IDL 対応関数プログラムジェネレータからの出力を用いて自動的に生成する。この結果、予め規定された CORBA IDL 定義を用いて必要な試験スイートを JavaScript の文法を用いて自由に記述できる。

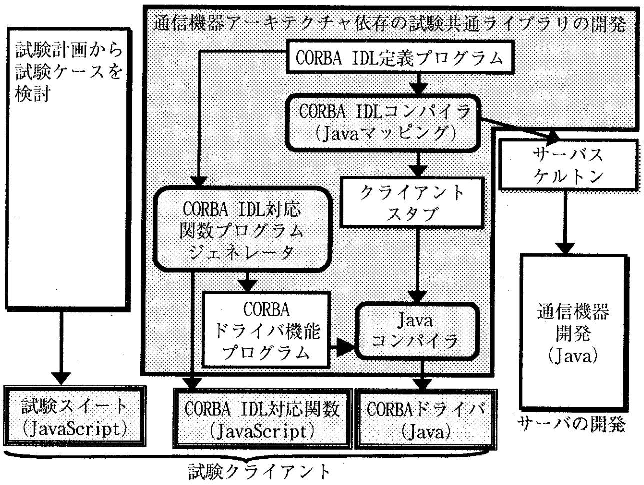

図 4 CORBA 機能試験環境の生成

4. 考察とまとめ

通信機器を対象とした CORBA 機能試験において試験スイート記述の課題を整理し、JavaScript による解決方法を検討した。提案した方法は JavaScript の高度な記述能力を継承できる事に加え、試験対象毎に異なる試験環境を機械的に構築する汎用性を実現できる。また、この試験スイートの実行を、Java を利用して開発した通常のクライアント動作と比較した。この結果、試験スイート実行は約 6%の負荷増加に留まり、ほぼ実運用と同等の性能が得られる事が分かった。現在、この方法を実際の通信機器開発へ適用している。

参考文献

[1] 寺島、他：「ソフトウェア機能試験手順の状態遷移表に基づく生成方法」、情処DiCoMo2000 (2000.6.28)
[2] 久保田：「分散オブジェクト指向システムにおけるメッセージのトレース」、情処論文誌Vol.39 No.1 (Jan1998)
[3] OMG：「CORBA Scripting Languagehttp Joint Submission」,http//www.omg.org(1999/7/17)