次世代ネットワークにおける OpenAPI のアプリケーション認定プログラム

6F−3 Application Certification Program for Application on Open API for Next Generartion Network

田中 慎司　山田 哲靖

Shinji TANAKA　Tetsuyasu YAMADA

日本電信電話株式会社　NTT ネットワークサービスシステム研究所

NTT Network Service Systems Laboratories, NTT Corporation

1　はじめに

次世代ネットワークでは，ネットワークを制御する次世代ネットワーク制御装置 (以下，“CA”) 上で，現在標準化が進められている標準 API (以下，“OpenAPI”) を利用するさまざまなサービスアプリケーションが動作する．本稿では，サービスアプリケーションを CA 上に搭載するためのアプリケーション認定プログラムと，その支援ツールであるアプリケーションポータビリティテスタについて論じる．

2　CA の位置付け

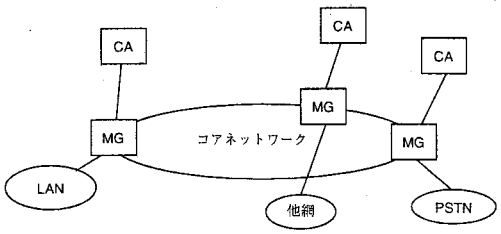


図 1: 次世代ネットワーク構成図

MSF にて規定されているモデルに基づいている次世代ネットワーク (図 1) では，CA は PSTN，VoIP 等の呼制御を中心とした，将来的な IP 系サービスを含む各種ネットワークサービスを行うため，OpenAPI を通して制御され，MG は各種ネットワーク間のプロトコル変換を行い，CA によって制御される．CA において，ネットワーク制御能力をアプリケーションに公開する OpenAPI の一種として，JAIN が存在する．JAIN は，Java の拡張 API の一種であり，Java による生産性向上やアプリケーションポータビリティの利点を考慮し，CA 上のアプリケーションは Java で記述される．

3　CA 上のアプリケーション認定プログラム

OpenAPI は，標準化され広く公開されるため，OpenAPI に対応した様々なアプリケーションが開発され，市販されることが期待できる．CA にこのようなアプリケーションを搭載するためには，アプリケーションポータビリティが達成されており，様々な CA 上で正常に動作するかなど，様々な点からの事前検証が必要となる．しかし，実機上での詳細な検証は，非常にコストが大きいため，シミュレーターなどを利用した検証を事前に行うことが有効である．

現在，アプリケーションポータビリティ検証を支援するアプリケーションポータビリティテスタを利用したアプリケーション認定プログラム (図 2) を検討中である．

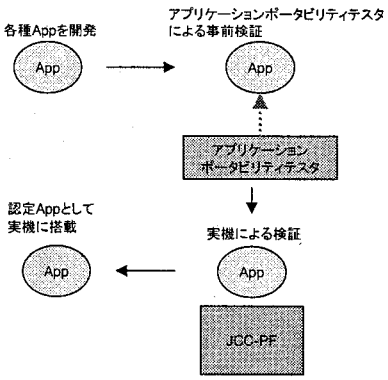


図 2: アプリケーション認定プログラム

3.1　アプリケーションポータビリティ

アプリケーションポータビリティとは，あるアプリケーションがさまざまな実装上で正しく動作することである．

CA 上の JAIN プラットフォームは，単一の実装ではなく，マルチベンダによる複数の実装がネットワーク上に存在することを想定している．それぞれのプラットフォームは，事前にコンフォーマンステストを行い，JAIN 準拠のプラットフォームであることを認定されている．

さまざまな実装上でのアプリケーションポータビリティを検証するには，実際に利用する実装上での詳細の検討を行うことが確実であるが，検証コストが非常に大きなものとなる．そのため，アプリケーションポータビリティテスタを利用し，プラットフォームシミュレータ上ですくなくとも一つの実装上で動作することを検証し，シミューレータ部を，ほかのシミュレータや，実機のプラットフォームに入れ替えることにより，アプリケーションポータビリティ

を検証する．

一般にアプリケーション検証では，さまざまなテストケースを実行し，正しく動作することの検証や，過負荷試験，性能評価試験を検証する．テストケースによる検証では，実行されるテストケースが，可能なすべての入力パターン，すべてのシーケンスパスを含むことが理想となる．これらの観点で作成したテストケースを実行し，アプリケーションの挙動，出力が，想定通りであれば，正しく動作している，とすることができる．過負荷試験，性能評価試験では，実機上で動作させ，要求している負荷で正しく動作するか，要求している性能を満たしているか，を検証する．

過負荷試験や性能評価試験に関しては，実機上で試験を行う必要があり，シミュレータ上でのアプリケーションポータビリティ検証の観点では，必要ない項目と考えられる．

そのため，アプリケーションポータビリティ検証では，テストケースを対象とするアプリケーションが想定するサービスシーケンスとし，ある実装上で，そのアプリケーションがそれらのサービスシーケンスがすべて正しく再現でき，JAIN プラットフォームの出力が想定通りかどうか，で判定する．

また，JAIN の仕様上では，それぞれのメソッドにおいて，precondition が定義されている．そのため，アプリケーションが JAIN のメソッドを呼び出す際に，

- メソッド呼び出し時の precondition が満たされており，
- メソッドの引数が正しい範囲に収まっている

必要がある．

しかし，前者は，precondition が満たされていない場合，プラットフォームは，なんらかの Exception を返却するように定義されており，アプリケーションがその Exception を handling することにより，正常動作するような構造になっている可能性が考えられる．そのため，この条件が満たされていないからといって，一概に JAIN 準拠ではない，ということはできない．また，後者は仕様において，具体的な引数のない内容が実装依存となっているため，正しい範囲は実装ごとに異なることが考えられる．そのため，JAIN 準拠かどうかの判断では対象とならず，特定の環境下で動作するかどうかの条件となる．

3.2 アプリケーションポータビリティテスタ

前章で述べたように，アプリケーションポータビリティテストでは，そのアプリケーションが想定する各種サービスシーケンスを用意し，それが端末や外部装置 (Web サーバや DB など)) と正しく通信し，前章であげた二つの条件を満たすことを確認すれば，十分であると言える．

アプリケーションポータビリティテスタの機能構成図を図 3に示す．アプリケーションポータビリティテスタは，シナリオ制御部，疑似端末部，外部装置制御部，ログ収集部から，構成される．

シナリオ制御部 投入されたサービスシーケンスを元に，疑似端末部，外部装置制御部に命令を発し，全体のシーケンスを進める．

疑似端末部 シナリオ制御部からの命令に従い，CA に対する，H.323，SIP の端末として動作する．

外部装置制御部 各種外部装置 (Web サーバ，DB など) の端末として，各外部装置にリクエストを要求する．現在のところ，プロトコルは HTTP，RMI を想定しているが，検証するアプリケーションが必要とする外部装置を制御可能な必要がある．

ログ収集部 動作している JAIN プラットフォームのログを収集する．アプリケーションが呼び出す JAIN プラットフォームのメソッドと，その前後の JAIN プラットフォームの内部状態を確認する．

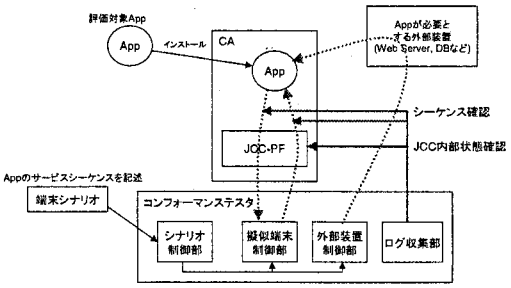


図 3: アプリケーションポータビリティテスタの機能構成図

実際の試験では，シナリオ制御部に，各アプリケーションの検証用シナリオを投入し，疑似端末制御装置，および外部装置制御装置から JCC プラットフォームおよび，対象アプリケーションを動作させる．そのときのアプリケーションと JCC プラットフォームの間のシーケンス，および JCC プラットフォームの内部状態をログ収集部により収集する．このログを想定されたシーケンスと比較し，正しいシーケンスとなっているかどうかによりアプリケーションポータビリティ認定を行う．

4 今後の進め方

今後は，実際にアプリケーションポータビリティテスタを開発していくことにより，その方式のより詳細な検討を行う．

参考文献

[1] JAIN specification, http://java.sun.com/products/jain/