トップエスイー修了制作

Androidアプリケーションにおける Java Path Finderを用いた並列処理の品質向上検討

フェリカネットワークス株式会社　和田真　Makoto.Wada@FeliCaNetworks.co.jp

開発における問題点

並列処理に関する品質確保手段としては、開発の現場ではソースコードレビューや試験実施によるものが一般的に用いられる。
しかしながら、いずれの手法を用いる場合も、全ての実行パターンを網羅的に検査することはできず、特定のタイミングで発生する問題を検出することは困難である。

モデル検査の適用による解決

業務で実際に開発しているAndroidアプリケーションに対してモデル検査を適用し、問題を解決する。モデル検査を適用するために必要となる手順を確立する。AndroidはJavaのコードで実装されていることから、モデル検査の手法としてJava Path Finder(JPF)を採用した。

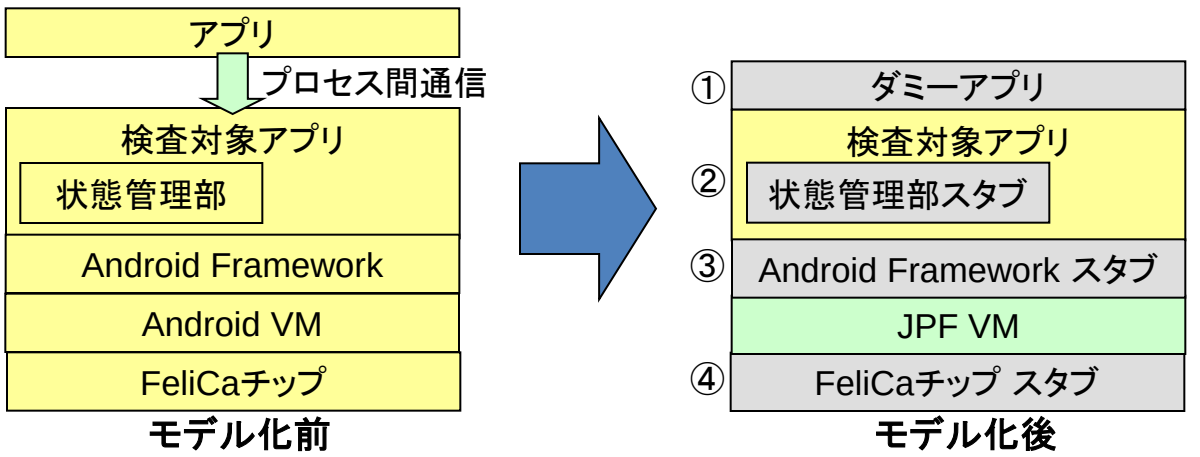
ツールの選択

モデル検査手法の選択にあたって、状態の網羅性を実現でき、かつ、既存のソースコードに対して効率的に手法の適用が可能であるJPFを選択した。
JPFでは状態管理数が増えやすい課題が一般的に存在するが、検証対象を絞り込むことでこの課題を解決した。

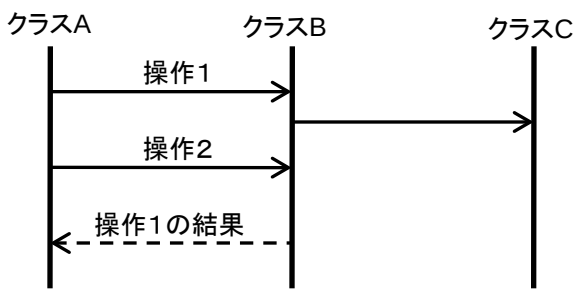
	JPF	SPIN
状態網羅	可能	可能
非決定的動作	可能	可能
モデル記述	Java	Promela
状態管理数	多い	少ない

モデル検査適用のためのモデル化

①プロセス間通信の捨象
　JPFでの検証には1プロセス化する必要があるため、プロセス間通信を捨象し、検証対象アプリと同一プロセスで動作するダミーアプリを実装
②状態管理部のスタブ化
　検証対象のロジックでは使用しない検証対象アプリ内の状態管理部をスタブ化
③Android Frameworkのスタブ化
　JPFではAndroid FrameworkのAPIは提供されていないため全てスタブ化
④FeliCaチップのスタブ化
　JPFでは実デバイスの振る舞いを検証することはできないため、FeliCaチップの振る舞いをスタブ化



検証



•操作1の実行中に操作2が実行された場合、常に期待動作となることを検証
•操作1の実行中を意味するフラグと、操作1の実行中に操作2が実行されたことを表すフラグを既存コードに実装し、各フラグの値と、操作1の結果から常に期待動作となることをJPFによって判定

評価・まとめ

•業務で開発しているAndroidアプリケーションにJPFを適用し、必要なモデル化の手順を確立することができた
•JPFを適用するためには検証対象のロジックの絞り込みが重要となるが、実際のユースケースなどを考慮し、検証箇所を絞り込むことも重要であることがわかった
•検証実施には多くのスタブ化実装が必要であったため、より効率的に検証を行うためにはスタブ実装の自動化を検討する必要がある