トップエスイー修了制作

Java PathFinderを利用したスレッドアンセーフなコードの検出

富士通株式会社　松田　友隆　t.matsuda@jp.fujitsu.com

開発における問題点

ソフト製品を出荷すると、サポート業務が発生。サポート業務には、技術情報整備やノウハウ蓄積によりサポート工数削減が見込める類と、そうでない類がある。後者は、専門の技術者により、それなりの時間をかけた対応が必要となる。API製品の場合、APIを利用するアプリが動かない、などのインシデントがこの類に該当する。インシデントが滞留すると、本業の開発計画にも影響するため、少しでも機械的に対処可能な範囲拡大が期待されている。

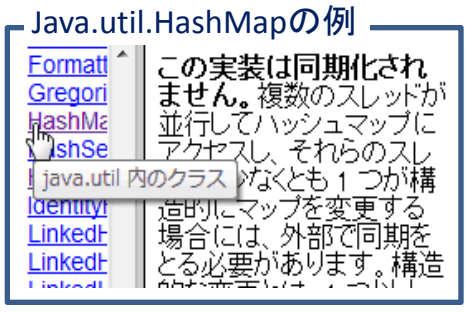
手法・ツールの提案による解決

単純なアプリの場合、静的解析やEclipse TPTPなどのツールが利用可能だが、スレッド並行動作が絡むと支援が弱い。例えば、静的解析ではインスタンスの考慮が無い。TPTPなども並行動作に特化した可視化支援は無い。今回、スレッドのタイミングにより発生する問題を機械的に検出する方式を検討。網羅的確認の基盤としてJava PathFinderを利用し、業務アプリをそのまま動作させ、問題の有無を検査するプロトタイプを開発し、実用可能性を評価した。

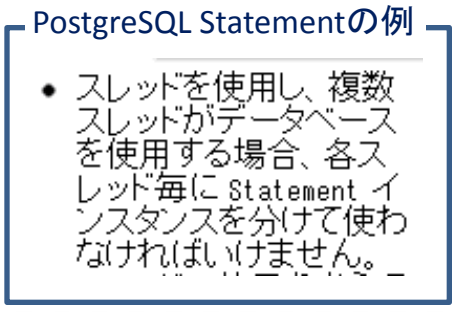
課題／目標

スレッドの実行タイミングにより発生する問題は、レース状態、性能問題、排他考慮漏れ、スレッドアンセーフAPIの利用考慮漏れなどがある。今回は、スレッドアンセーフAPIの利用考慮漏れに着目した。
通常、スレッドアンセーフAPIの仕様はドキュメントで記述され、間違った使い方に対する開発環境の支援はない。

Java.util.HashMapの例



PostgreSQL Statementの例



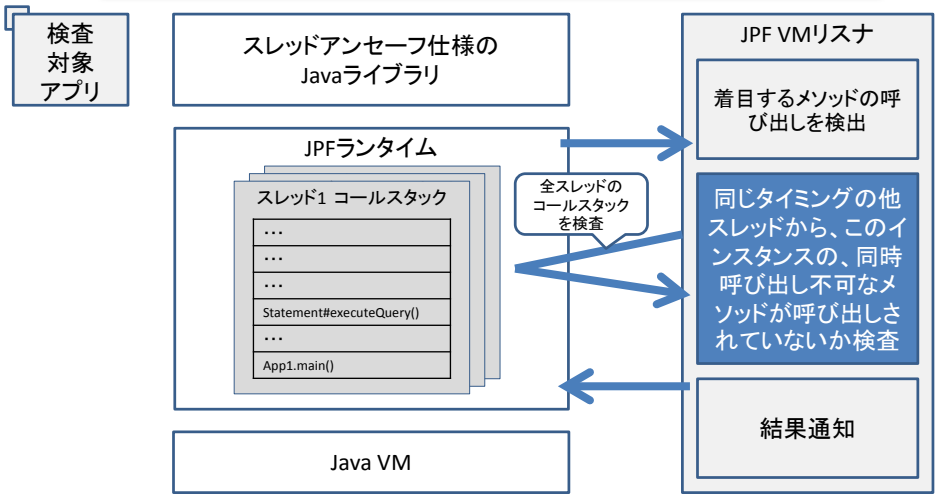
APIに不慣れな開発者は常に存在するため、スレッドアンセーフAPIの利用考慮漏れインシデントは常に発生する。インシデント対応側として、確認すべき観点は明確である。しかし、確認対象はインシデント毎に異なる業務アプリであり、都度、スキルのある技術者によるアプリ解析が必要とされている。

■ 目標

ユーザから送られてきた業務アプリケーションプログラムが、スレッドアンセーフ仕様なAPIを、複数スレッドから同時に呼び出すようなプログラムになってないか、機械的に検査する。

- 問題の有無を出力する。問題がある場合、問題を起こすタイミングを解釈できる情報も出力する。
- Javaアプリケーションの全スレッドの実行タイミングの組み合せを作り出す基盤としてJava PathFinder(JPF)を利用する。

検出方式



JPF VMリスナを実装。JPF VMリスナは、JPFランタイムから、確認すべきスレッドの状態毎にイベントを受ける。イベントを受けたVMリスナでは、その状態におけるVM内の全てのスレッドコールスタックを検査し、同一オブジェクトに対する、同時スレッドアンセーフメソッドの呼び出しが無いか検査を行う。

評価／考察

■ 有効性
- PostgreSQL JDBCドライバを素材にプロトタイプツールを実装し実験を実施。実現可能であることを確認した。

■ 実装コスト
- JPF VMリスナ（工数小）
  - JPF標準APIの範囲で実装可能であることを確認した。
  - 今回、他ライブラリにも応用可能な方式を確認した。
  - 他ライブラリに適用した場合も工数小と考えられる
- Nativeメソッド対応
  - ネットワーククライアントなどNativeを多く含むライブラリの場合工数が大きくなる（中～大）。
  - HashMapやXMLパーサなどPureJavaのライブラリは対応不要（もしくは工数小）である。

■ 実業務への適用可能性
- 検査対象の業務アプリが、小規模、もしくは要点のみにサンプル化されているなどの場合は適用可能と考える。
- JPFが持つ状態削減の仕掛けと組み合わせることにより、より大きなアプリへ適用範囲を拡大できる可能性がある。