一般論文

Web開発向けフレームワークを活用した開発手法の確立

神戸事業所　開発部　開発第2課
佐藤 智紀

1．まえがき

顧客の要求の変化に伴い、ソフトウェア開発は、複雑、短納期となってきている。

これに対応するソフトウェア開発手法として、開発量を削減し、生産効率向上を図るオープンソースソフトウェア(OSS)(注1)や各種ツールの活用が挙げられる。

今回の開発では、Webシステムを実現するソースコード量を削減するために、Javaの画面部品ライブラリ“JavaFX(注2)”とデータベースアクセスフレームワーク“Spring(注3)”を活用する開発手法を適用した。JavaFXは、Javaバージョン8から標準搭載されている画面作成ライブラリであり、Springは開発環境との親和性が高いことも採用理由にある。

JavaFXとSpringを適用した結果、新規85画面の製作量において、従来の開発手法では544kLの実装が必要な開発量を127kL(約23%)削減し、425kLで開発することができた。

さらに、ソースコード量の削減により、可読性と保守性の高いソースコードが作成できるようになった。

本稿では、今回のWeb開発において活用したOSSによる開発手法を紹介する。

2．Web開発における新技術の紹介

JavaをベースとしたWeb開発手法は、2010年のJavaバージョン8(以下、Java8)の登場により大きく変化した。

従来は、画面を作成するためのライブラリ“Swing(注4)”と、画面遷移などの機能を提供する“Struts(注5)”とを組み合わせた環境での開発が主流であった。

Java8の登場によって、JavaFXとSpringとを組み合わせた環境での開発が主流になってきた。Javaバージョンの遷移に伴う開発手法の変化を図1に示す。

図1．Javaバージョンの遷移

2.1　JavaFXの概要

JavaFXは、従来のSwingよりも多くの画面部品をサポートし、簡単に表現力が高い画面が作成できるため、現在では、広く利用されるようになってきた。

JavaFXの登場により、画面の作成方法が変化した。従来は、画面のレイアウトをJavaのソースコードに記述していたが、JavaFXでは、FXMLと呼ばれるXMLベースの言語で画面デザインを記述するようになり、Javaのソースコードを編集することなく、画面作成が可能になった。

また、従来の画面作成ライブラリと比較すると、グラフィックや動画、サウンドなどのメディア関連の機能が充実し、映像やサウンドを多用したコンテンツの作成にも対応できるようになった。このほか、カレンダー部品やグラフ部品などが標準的にサポートされ、表現力が高い画面が作成できる。

JavaFXやSwingといった画面作成ライブラリが提供する部品は、独自の機能を追加した拡張部品に変更することができる。JavaFXでは、サポートする部品数の増加によって、システム仕様に特化した機能拡張部品の作成が容易となる。

2.2　Springの概要

Springは、データベースアクセスを容易に実現するためのJava用のフレームワークであり、OSSとして提供されている。

また、従来、アプリケーション側のソースコードに記述していた複雑なデータベースアクセス処理を隠蔽し、少ないソースコード量で目的の処理を実現することができる。

従来のコーディングとSpring利用時の開発範囲の違いを図2に示す。Srpingがデータベースのオープン・クローズやSQL文生成などを行うため、開発者はテーブル定義とデータ抽出後のデータ処理を記述するだけで良い。これにより、データベースアクセスに必要なSQL文法を熟知していない開発者でも、データベースアクセス処理の実装が可能になる。

(注1) OSS：ソースコードが無償で公開され、誰でも利用可能なソフトウェア。オープンソースソフトウェア
(注2) JavaFX：Java向けの画面作成ライブラリ
(注3) Spring：オープンソースのJava向けアプリケーションフレームワーク
(注4) Swing：Javaの画面を開発するためのライブラリ
(注5) Struts：Java向けWebアプリケーションフレームワーク

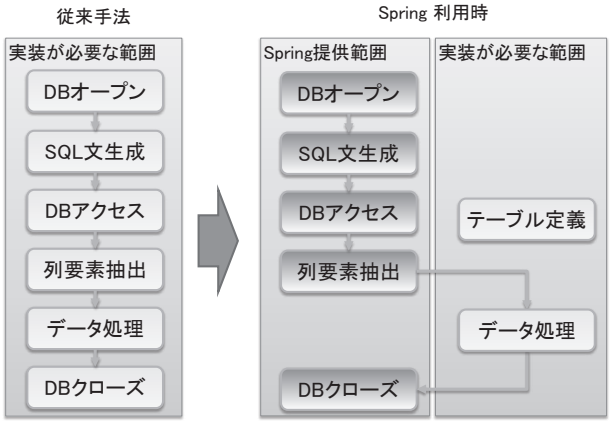

図2．データベースアクセス実現差異

3．開発手法の遷移

JavaFXとSpringの採用により、Web開発におけるソフトウェア構成と開発手法が大きく変化した。

実行環境とソフトウェア構成の違いを図3に示し、開発手法の違いを次節に記載する。

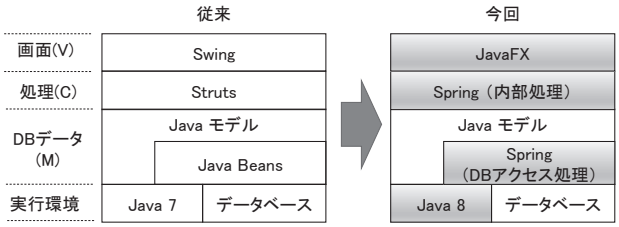

図3．Web開発のソフトウェア構成

3.1　JavaFXにおける画面の開発手法

画面開発では、JavaFXと連携が可能な画面作成ツール“Scene Builder”(注6)を導入した。このツールはOSSとして提供されているものである。

従来、Javaプログラムで実現していた画面レイアウトは、ツール画面上に表示される画面部品群をマウスで選択・配置することにより、画面が作成可能となった。これにより、画面レイアウトに使用する代表的な10種類程度の部品名を覚えることで、画面レイアウトを作成できるようになった。

また、Scene Builderでは、画面レイアウトを決定するとFXMLファイルを出力することができ、XMLベースのFXMLファイルについても、画面作成者が直接記述する必要がなく、画面作成が容易になった。

図4にScene Builder適用による画面作成のイメージを示す。

3.2　Spring活用によるデータベースアクセス処理の記述方法

データベースアクセスの実装では、従来、データベースにアクセスするためのJava標準APIであるJDBC（Java Database Connectivity）のほか、“SQL”の知識が必要であった。

JDBCの利用には、Java言語技術とデータベースに関する知識が必要となり、使い方によっては、応答性能の劣化を招くことがあった。

Springでは、Springが提供するデータ操作を行うAPIの知識を修得することで、テーブル入出力ができるようになった。

（注6）Scene Builder：JavaFX向け画面作成用のツール。FXMLファイルが出力可能な画面ビルダ。

図4．Scene Builder適用による画面作成イメージ

4．適用事例紹介

4.1　トレンドグラフ画面の適用事例

JavaFXとSpringを採用した画面の開発手法として、今回開発したWebシステムで最も複雑な画面である“3段グラフ画面”を紹介する。

3段グラフ画面の仕様は、トレンドグラフを3つ縦に並べる3段のグラフを1画面に表現し、一つのトレンドグラフには最大8種類の積上げ棒グラフと1種類の折れ線グラフを重ね合わせるものである。データ数は、30分間隔のデータを1週間分表示するもので、1画面あたり全データ点数は8000点以上になる。

本システムでは、多くのデータをグラフ表示するための実現方法のほか、応答性についても課題となった。

グラフの実現には、JavaFXで追加された、折れ線グラフ部品と棒グラフ部品を採用した。

JavaFXが提供する標準部品の採用により、従来、線や矩形の描画によって、すべてを一から実装していたグラフが簡単に実現できるようになった。3段グラフ部品の開発ソースコード量は、従来手法で約5.0kL必要であるところを、共通部品の適用によって約3.1kLで実現することができた。

応答性の面では、目標の要求応答性能である5秒以内を満足し、標準グラフ部品の表示性能に問題がないことを確認できた。

開発した3段グラフ画面の適用事例を図5に示す。

4.2　ソースコード削減効果

JavaFXとSpringを適用した開発手法の採用により、新規画面の開発において、1画面あたりの開発ソースコード量は、表1に示すとおりとなった。

表１．1画面あたりの開発ソースコード量

処理対象	1画面あたりの開発ソースコード量		
	従来	今回	削減量
画面処理	3.7kL	2.7kL	1.0kL（27%削減）
データベース処理	2.7kL	2.2kL	0.5kL（19%削減）

5．採用した新技術の評価

5.1　開発手法の確立による効果

今回の開発で、Web開発向けのフレームワークを活用する開発手法が確立できた。

画面開発では、画面作成ツールScene Builder上で画面部品を配置するだけで、簡単に画面を作成できるようになった。データベースアクセスでは、Springが提供するAPIを理解するだけで、SQLを使用せず、データベースにアクセスできるようになった。

結果、ソースコード量の削減、高度なJava言語･SQL文法に精通していない開発者でも画面レイアウト作成やデータベースアクセス処理が実現できる、という効果が得られた。（図6参照）

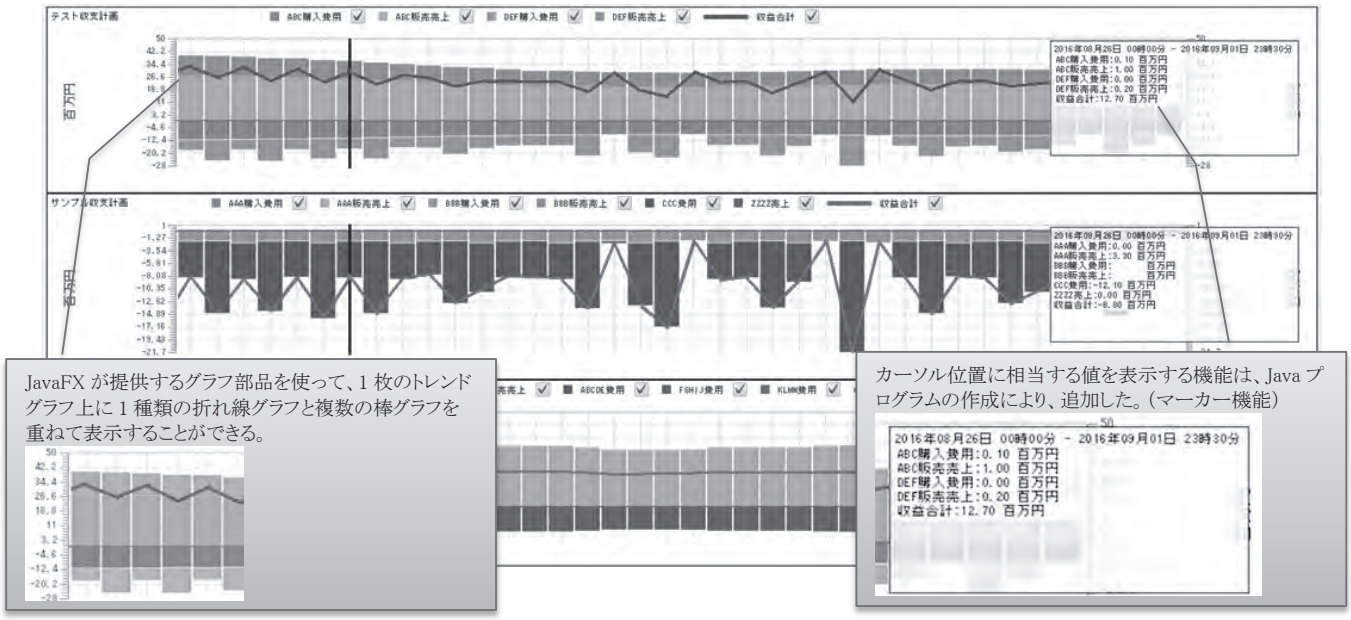

図５．開発したトレンドグラフ画面

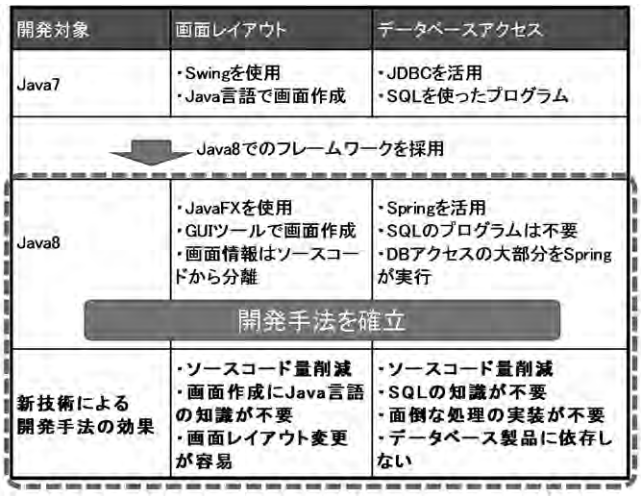

図６．新技術による開発手法の効果

5.2　新技術採用のデメリット

JavaFX、Springは、Java言語で開発する様々な分野に利用可能である反面、導入の難しさもある。

書籍やインターネットでの公開情報が少ないため、今回の開発ではシステム仕様に合わせた画面部品の選定、画面部品の拡張仕様の決定など、実現性の評価に時間を要した。

また、トレンドグラフ横軸(日時)のラベル間引き表示、テキストボックスに入力中の文字数制限等、一般的なシステム仕様をJavaFXでは実現できず、一部、作り込みが発生した。

本システムの試験段階においても、技術情報不足と不具合事例の少なさから、調査･改修に苦労することになった。

本開発で蓄積した知識やノウハウは、当社神戸事業所部内情報Wikiとして登録し、他部門でも活用できるよう、2017年度内をめどに整備を進めていく。

６．今後の取り組み

今回の開発では、ソースコードの開発量削減をテーマにOSSを選定した。開発効率向上のためには、製作フェーズのみではなく、試験フェーズでの各種ツールの活用も有効である。

今後は、コーディング規約チェックツール、静的解析ツール、自動試験ツールなどを適用し、生産性と品質の向上を目的とした開発手法の確立を図っていく。

また、今後、OSSや各種ツールを積極的に活用するためには、短期間でメリットとデメリットを評価する必要がある。そのために、新技術に対する生産性と性能を評価する観点および方法を確立し、開発対象にマッチする開発手法を決定していく。

７．むすび

今回の開発で、Javaの画面部品ライブラリ“JavaFX”とデータベースアクセスフレームワーク“Spring”を採用したソフトウェア開発手法を確立した。

この開発手法により、期待する効果が得られることが実証できた。今回採用したフレームワークは、画面数が多いシステムほど、生産性の向上の効果が大きくなる。

今後も積極的に新技術を活用した開発手法の改善を推進し、生産性向上を実現していく。

最後に、本開発にあたり、貴重なご意見、ご指導を頂いた関係者の方々に深く感謝申し上げる。

執筆者紹介

佐藤　智紀　　サトウ　トモノリ
1999年入社。主に粒子線治療装置のソフトウェア開発に従事。現在、神戸事業所開発部開発第2課。