B-026

FXML を用いた JSF ページとプログラムの自動生成

Generation of JSF pages and interface programs from FXML

原口久美子† 塚本享治†
Kumiko Haraguchi Michiharau Tsukamoto

1. まえがき

ソフトウェア開発における短納期化,高品質化,低価格化への要求を受け,近年,開発自動化への動きが高まっている.現在筆者らは,代表的なシステム構成である 3 層モデルに着目し,各層のソースコード自動生成手法を確立することで,システム全体の開発効率と品質の向上を目指している.

今回,3 層の中で UI を受け持つプレゼンテーション層の開発を自動化するツールを作成したので報告する.本ツールでは JavaFX2.2 の GUI エディタである SceneBuilder を用いて画面を作成し,出力される FXML ファイルと画面遷移図の XMI ファイルから,JSF ページと設定ファイル,データや処理の管理を行うマネージド Bean を自動生成する.

2．FXML と JSF の相違点と解決策

2.1 FXML と JSF

FXML とは,JavaFX の GUI 構成を記述する XML ベースの言語のことであり[1],SceneBuilder というエディタでグラフィカルに編集することができる.JSF とは Web アプリケーションのフレームワークのことであり[2],JSF のタグで UI を簡単に構築することができる.画面設計と UI 構築の簡易さから,本ツールではこれら二つの技術を組み合わせることにした.

2.2 相違点と解決策

FXML を JSF に変換するにあたり問題となる両者の相違点と解決策を示す.

（1）コンテナの階層構造

FXML では GUI 部品をまとめるコンテナというクラスがあり,コンテナの階層構造が許される.一方,JSF では入出力部品をまとめる form の階層構造は許されない.変換先である JSF の形式に対応するため,FXML でも階層構造は避け,コンテナ Pane を使って部品をフラット構造でまとめる.

（2）GUI 部品の相対座標

FXML ではコンテナが階層構造になっている場合,その内部部品の位置は所属するコンテナとの相対座標で扱う.一方,JSF では部品の位置を絶対座標で扱う.そこで,FXML の座標情報を Java の外部関数 stack を使って記憶させ,絶対座標に計算し直したものを JSF では使う.

（3）入出力とイベント処理

FXML では入出力部品やコマンド部品は,コントローラに変数としてバインドされる.入出力やイベント処理は,バインドされた各部品に対してプログラムを用意し,実行する.一方,JSF の場合 form の入出力項目やコマンドは,マネージド Bean という JavaBeans がプロパティや setter/getter,アクションメソッドとして管理する. 入出力やイベント処理は,JSF エンジンがマネージド Bean の setter/getter やアクションメソッドを直接呼び出し実行する.

† 東京工科大学 メディア学部メディア学科

JSF の形式に対応するため,FXML の fx:id と JSF のマネージド Bean の情報を対応付ける.具体的には,コンテナ Pane の fx:id と JSF のマネージド Bean 名（先頭は小文字）,入出力部品の fx:id とマネージド Bean のプロパティ名,コマンド部品の fx:id とマネージド Bean のアクションメソッド名を一致させる.この対応付けにより,XSLT 変換で fx:id を JSF のタグの該当箇所に埋め込むことができる.

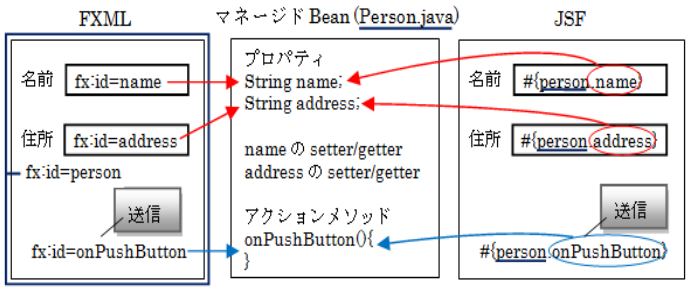


図 1. FXML とマネージド Bean と JSF の対応イメージ

3. FXML から JSF への XSLT 変換

（1）情報の付加

FXML を JSF に変換する際に,不足している情報は FXML にコメントとして付加する.例えば,JSF のマネージド Bean のスコープを付加する場合,マネージド Bean 名を fx:id で指定し,スコープを以下のように記述する.

```
<!-- fx:id="person" scope="session" -->
```

（2）タグの対応

FXML と JSF のタグについては,同機能を持つものを対応させ,変換する.表 1 はその一例である.

表 1. FXML タグと JSF タグの対応表例

FXML	JSF
Pane	h:form
TextField	h:inputText
Button	h:commandButton
ListView	h:outputText
ImageView	h:graphicImage
<!-- -->	付加情報

（3）設定ファイル

マネージド Bean の登録用ファイル(managed-bean.xml)は,FXML ファイルからマネージド Bean 名やスコープなど必要な情報を抽出して生成する.また,画面遷移のナビゲーションルールを定義するファイル(navigation-rule.xml)は,まず UML ツールで画面遷移図を描き,モデル情報を表す規格の XMI(XML Metadata Interchange)で出力する.次に XMI ファイルから遷移元・遷移先の JSP ページやアクションメソッドなどの情報を抽出してファイルを生成する.

（4）マネージド Bean

FXML ファイルと navigation-rule.xml に記述されているアクションメソッドの返り値情報から, JavaBeans の要素（プロパティ,setter/getter,アクションメソッド）を持つマネージド Bean を自動生成する.本研究ではこれを ManagedBeanBase 呼ぶ.そして, ManagedBeanBase を継承し, 開発者が独自に拡張できるようにしたものを ManagedBean とする.ManagedBean も同時に自動生成する.

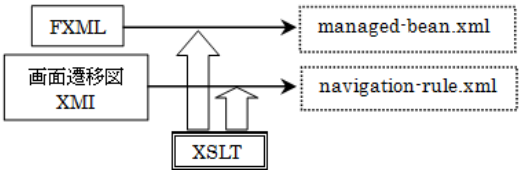
（5）変換手順

Ant のビルドスクリプトを用いて FXML から JSF ページとその他プログラムの XSLT 変換を行う.変換手順を以下に示す.

① FXML から設定ファイルを生成

1) 階層構造 FXML から,managed-bean.xml を生成

2) 画面遷移図 XMI から navigation-rule.xml を生成



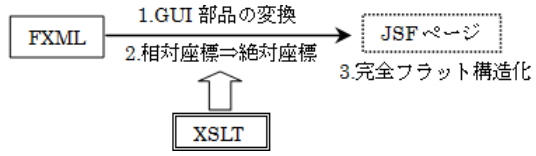
② FXML から JSF を生成

1) 階層構造 FXML を JSF に変換

・FXML の GUI 部品を JSF の部品に変換

・FXML の相対座標を JSF の絶対座標に変換

2) JSF を完全フラット構造に変換

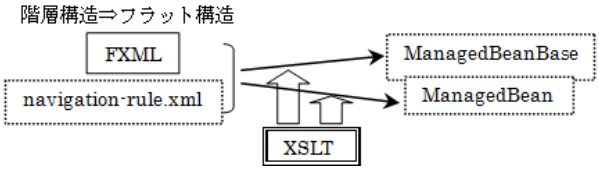


③FXML と画面遷移のナビゲーションファイルから ManagedBeanBase と ManagedBean を生成

1) 階層構造 FXML からフラット構造 FXML に変換

2) フラット構造 FXML から ManagedBeanBase を生成

3) フラット構造 FXML から ManagedBeanBase を継承する ManagedBean を生成



4. 開発プロセス

本ツールを用いた開発プロセスは以下となる.

（1）SceneBuilder で GUI 画面(FXML)を作成,

（2）テキストエディタで FXML に付加情報を追加

（3）UML ツールで画面遷移図(XMI)を作成

（4）(2)(3)から設定ファイル, JSF ページ, MangedBeanBase（＋継承した ManagedBean）を自動生成

（5）Web コンテナへ Web アプリケーションを自動配備

5. 自動生成実験：カード型データベース

本ツールを用いて,簡単なカード型データベースの Web アプリケーションを生成する実験を行った.SceneBuilder と UML ツール astah* professional[3]を使って GUI 画面と画面遷移図を作成し,本ツールで UI を自動生成した.ビジネスロジック層とデータ層に関しては,必要なソースコードやデータベースを別途用意した.

5.1 結果

本ツールで自動生成したソースコードの行数と筆者ら自身がハンドコーディングしたソースコードの行数を比較したところ,UI 部分の自動生成率は 89.2%であった.

表 2. カード型 DB プレゼンテーション層の自動生成結果

	ファイル種類	ファイル数	コード行数	コード(%)
自動生成	JSF ページ	6	390	**89.2**
	設定ファイル	2	160	
	ManagedBean/Base	12	798	
	コンテキスト xml	1	4	
	計	21	1352	
コーディング	ManagedBean (加筆)	6	163	10.8
	計	6	163	
	合計	27	1515	100

※ツールによって自動生成される FXML ファイルや XMI ファイル, ビジネスロジック層やデータ層のソースコードは対象外

5.2 考察

以上から,開発者が独自に拡張する部分(ManagedBean)以外は全て,本ツールで自動生成できるという結果が得られた.これは,変換先のソースコードに必要な情報を,予め変換元の FXML と XMI に反映させ,XML 操作の自由度が高い XSLT で変換したためである.さらに,本実験の他にもスタンドアローンアプリを含めいくつかのサンプルを試したが,同様の結果を得ることができた.

6. あとがき

本稿では,3 層モデルの中のプレゼンテーション層の開発を自動化するツールについて報告した.本ツールを使用すると,GUI 画面と画面遷移の図を描くだけで,Web ページやプログラムを自動生成することができる.つまり,迅速で誤りのない UI 開発が可能となる.

今後の課題としては,まず FXML で未対応の機能（テーブルへの GUI 部品の配置等）に対応するため,独自にコメントを付加し機能を拡張することが挙げられる.また,本研究の次の段階として,ビジネスロジック層とデータ層の自動化ツールを開発し,本ツールとの連携を図りたい.

参考文献

[1]Oracle:Introduction to FXML. http://docs.oracle.com/javafx/2/api/javafx/fxml/doc-files/introduction_to_fxml.html

[2]JCP:JavaServer Pages 2.2 Public Review. http://download.oracle.com/otndocs/jcp/jsf-2_2-pr-spec/index.html

[3]ChangeVision:http://astah.change-vision.com/ja/product/astah-professional.html

[4]斉藤賢哉:マスタリング JavaEE5, Part5 JSF, 翔泳社,2007