Message from Vendors

Web開発はStrutsと自動生成ツールBlastJで

第1回：EclipseベースツールBlastJの大いなるメリット

テキスト= CROSSFIRE JAPAN INC. 飯塚 友裕 IIZUKA Tomohiro http://crossfire.jp

はじめに

今回の連載では，Web開発をStrutsとEclipseおよびEclipseベースのツールであるBlastJを使って開発する方法をご紹介しようと思います．

BlastJは，Strutsフレームワークを利用してWebシステムを作る際のStrutsに関連したソースコードおよび設定ファイルを自動生成するツールです．BlastJ自体は実装段階をサポートする下流工程向けの開発ツールなのですが，背後に上流から下流までのシステムの開発全体を見渡した設計思想を持ったツールであり，設計が実装にうまく生かされるよう考えられて作られたツールです．連載の中では，ツールだけでなく設計思想のほうもご紹介できればと思います．

まずは，設計思想から入って徐々に具体的な実装や作業の話へと展開していこうと思います．

Web系Javaの効率化に対するアプローチ

●開発を効率的に行うためには

ここ最近，筆者の周りではJavaの需要が非常に増えているように感じます．それに対してJavaのエンジニアの数が足りているかというと非常に少ないのではないかと感じられ，これからの時代，ますます開発効率の向上はJava開発においては必要になっていくものと思われます．

筆者は，IT業界はサービス業というよりは製造業の一種と考えています．プログラムなどの知的財産は目に見えにくい分，特殊なものと考えられているような風潮があると思います．しかし，実際には自動車や電気製品と同じように製品の設計があり，製造現場でその設計を満たすよう製造が行われるという点では，まさにものづくりそのものだと感じています．

最近のIT業界では効率化のためにフレームワーク活用による部品の再利用が活発に議論されるようになりました．また，設計方法に対する議論も非常に多くなされています．ところが，製造現場の作業に対する議論がまだまだ十分になされていないのではないかと筆者は考えています．

そこに着目して，筆者はStrutsソースコードと設定ファイルの自動生成ツールであるBlastJを開発しました．

●フレームワークと自動生成ツールでお互いをカバー

前節で開発の効率化のためにフレームワークによる部品再利用が活用されていると書きました．では，どのような部品が再利用できるのでしょうか？ それは，一言で言ってしまえばこれから作ろうとしている業務に依存しない部分です．フレームワークを利用することによりこの部分の工数をカットすることができます．しかし，実際の開発では業務に依存する部分の比重が非常に大きい場合がほとんどで，この部分を効率化できればさらなる効率アップが可能です．

現場のエンジニアの方ならば相当実感があると思うのですが，実際の業務部分の実装作業においては単純作業が非常に多いのが現状です．特にWeb系シ

ステムの開発においてこの傾向はかなり強いと感じます．

では，この単純作業をなくすにはどうしたらよいのでしょうか？ それは，プログラムや設定ファイルの自動生成ツールを使ってツールにこの作業を行わせてしまえばよいのです．同じようなプログラムを書いたり，同じような内容をXMLに丹念に書いたりする必要がある場合には，プログラム自動生成ツールはものすごい威力を発揮します．特に，Strutsのように，機能部品をフレームワークの仕様に合わせて実装し，その機能部品をXMLに登録していくようなスタイルをフレームワーク利用者に求めるフレームワークを用いる場合に，自動生成ツールは非常に適しています（図1）．

●Java初心者も安心，実装作業を簡単に

自動生成ツールは単純作業をなくす上で非常に効率アップをさせることができますが，実は効率アップ以外にもさまざまなメリットがあります．自動生成ツールを使うことで単純作業をなくすことができるということは，開発する際の作業が簡素化されるということです．たとえば，今まで5ステップの作業で1つの部品の実装が完了したとすると，それが2ステップになるというような感じです．

1つのメリットとして，開発作業が簡単になるというメリットがあります．一般的にJavaの開発では，Javaの言語の仕様を覚えただけではなかなかスムーズにプログラマとしての作業ができません．言語の仕様に加えて使用するフレームワークに関する知識，すなわちそのフレームワークを使って実装作業を行うための作業ステップを理解しなくてはなりません．

自動生成ツールを使うことにより作業ステップ数が少なくなり，さらに，ツールが入力すべき項目をGUIでナビゲートすることにより，Javaに関して経験の浅いエンジニアでもスムーズに作業をすることができます．

また，もう1つのメリットとしてステップ数が減るほかに，作業の冗長性も排除され，同じ内容のことをソースコードやXMLファイルに何度も入力しなくてよくなります．したがって入力ミスも減ります．

さらには，今までは人間が作成するファイルやソースコードとして管理しなくてはいけなかったものが，コンピュータによって自動的に生成されたものになります．そうすることで人間が直接管理しなければいけない対象からはずすことができるので，テストや保守も楽になります．

●設計を実装の味方につけよう

BlastJのようなプログラム自動生成ツールを使って生成されたプログラムには，テキストエディタを使ってプログラムを作成するのに比べてプログラムの表現の自由度に制約がかかるという「特徴」があります．

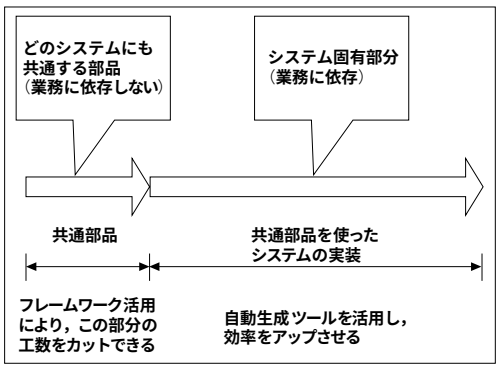
この「特徴」は，デメリットになってしまうケースが多々ありますが，メリットになるケースもあります．デメリットになってしまうケースは，ただ単に自由に書きたいのに書けないという場合なので簡単に思いつきますが，メリットになるケースとはどのようなケースでしょうか？

それは，実装の現場で設計規約やコーディング規約を設けている場面を思い浮かべていただくと想像しやすいと思います．では，設計規約やコーディング規約は何を指標として定めているのでしょうか？

それは，設計する部分の「役割」です．どんな場合でもシステムは複数のパーツで構成され，良いプログラムは各パーツがどのような機能を提供しているか明確です．

設計規約やコーディング規約は，設計や実装する

●図1　自動生成ツールの活用



パーツの役割分担をきっちりさせるために，各パーツが自分の役割以外のことをしないようにするための制限と言えます．

よって，この設計規約やコーディング規約と自動生成ツールを使う際の制約がうまくマッチングする場合，最初に述べた「特徴」がメリットになるケースと言えます．

役割分担設計でわかりやすく設計

●そもそもMVCと役割分担

Strutsは，MVCモデルでの開発をサポートするフレームワークです．MVCモデルは，単純な「画面系」と「データ処理系」だけの設計に加えて，それらの仲立ちをする「コントローラ」が加わったものです．

このコントローラの役割は，「画面系」と「データ処理系」の直接の依存関係をなくすことです．直接の依存関係をなくすことによって，「画面系」と「データ処理系」をそれぞれ別けて開発することができます．このような設計にしておくと，要件に変更が出て画面を変えなくてはいけない場合でも，コントローラ部分が大半の変更を吸収してくれるので，保守しやすいシステムになります．

●保守性と開発効率の関係

一見，開発効率と保守性はまったく別物のように感じられる方が多いのではないでしょうか．しかし，実際の開発ではスパイラル式の開発が多く，いかに出戻り作業を少なく，そしてスムーズに行うかが非常に開発効率を上げる上での大きなファクターになっているのではないかと思います．

ですので，開発効率を追求する場合には，ただ単に生成するソースコードや設定ファイルの量だけに注目するのではなく，実作業がどれだけ減るかにフォーカスを当てて評価を行わないと，本当の開発効率向上を追及することはできないと筆者は考えております．

●サービス指向のSOA

最近，SOA（Service Oriented Architecture：サービス指向アーキテクチャ）という言葉をよく聞きます．SOAは，データに対する処理をサービスが一括して管理してくれるという概念で，非常にプログラム設計をやる上で都合の良い概念だと筆者は考えています．この考え方を用いることにより，さらに役割分担的な設計がしやすくなります．

たとえ話をするならば，駅の売店で「みかんジュースをください」と言うと売店の人がお金と引き換えにみかんジュースをわたすというサービスをしてくれます．これをシステム屋風に解析すると次のようになります．

- 売店には，ジュースや雑誌食べ物などが棚などに整理されているわけで，在庫オブジェクトがあります．
- 販売サービスは，販売員がこれらの在庫オブジェクトを取り出して（操作して），客のほしい商品を取り出す動作です．
- 客は，販売員に対して「販売」というサービスを通して（インタフェースとして）売店の機能を使います．

このモデルの特徴は，「動作モデル」の存在が強調されているところです．オブジェクト指向で定義されたオブジェクトを操作するものとして「動作モデル」（＝サービス）が利用され，このサービスを組み合わせてシステムを組んでいくことになります．

●BlastJが対象とするSOA的4層設計

BlastJは，開発する際に取られるアーキテクチャとして4層設計を取っています．

Strutsはまさに，コントローラ部分を主にサポートするフレームワークであり，BlastJはStrutsを利用したコントローラ部分のソースコードおよび設定ファイルを，GUIから自動生成するツールです．

このとき，コントローラの役割は，サービスであるビジネスロジックを組み合わせて，ビュー（JSP）にわたすための値を取得することと，ビジネスロジックから得た値を元にどのビューへ遷移するかを決めることです．

BlastJでは，この役割のために必要な部品が用意

されており，それらをGUI上で組み合わせて設定していくことによってコントローラ部分のプログラムを完全に自動生成することができるようになっています（図2）．

●要件変更に対する対応

理想を言えば，システムの開発は要件定義の段階で一度定義したものを作って終わりになることが望ましいです．しかし，現実の開発では途中まで作った段階で「ここの画面はやっぱりこうしよう」などという要求が必ず出るものです．サービス指向で実装をきちっと行うことによって，このような場合もプログラムの変更を少なくすることができます．要件に変更が加えられる場合，ビュー（JSP）の仕様が変化しますので，ある画面の仕様に変更が加えられる場合を考えてみましょう．ビューが必要とするデータが変わりますので，それに対する対応がコントローラに求められます．仕様が削除される場合には，不必要になるサービスの利用をコントローラから削除することで対応できます．この場合は，サービス部分の実装に手を加える必要はありません．

では，仕様が追加される場合にはどうなるのでしょうか？ この場合，既存のサービスを使って追加仕様を満たせられる場合にはコントローラの変更だけで対応することができます．もしも，既存のサービスには存在しない機能が必要な場合には，新規でサービスを追加してからコントローラを変更します．

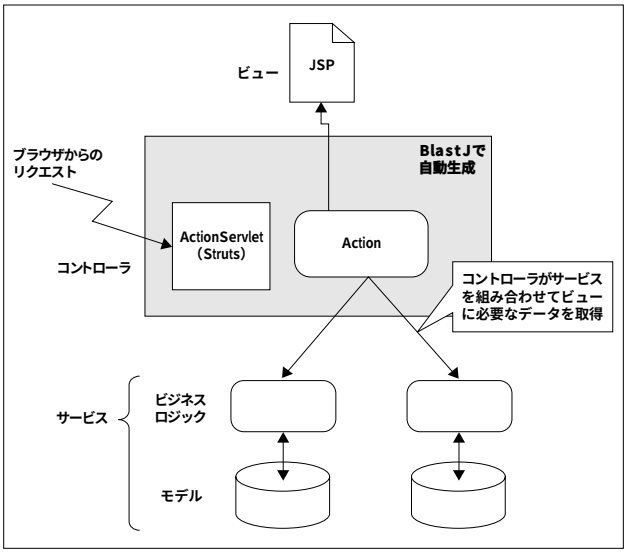
以上から言えることは，コントローラが変更を吸収することで，サービスの実装部分に関して「新規作成」はあっても「変更」がないため，サービスの実装部分の作業で，テストの必要性による出戻り作業を減らすことができます．

さらに，BlastJを使って実装する場合は，実装に変更が加えられることが多いコントローラ部分を完全に設定で行ってしまうため，非常に開発効率が良くなります（表1）．

●表1　仕様変更に対するコントローラ，サービスの処置

	新規作成	変更	削除
コントローラ		○	
サービス	○		○

●図2　BlastJによる自動生成



まとめ

今回は，設計と実装の関係とBlastJが取っているアーキテクチャについて書きました．最初の設計を実装まで見通してうまく行えれば，実装段階の作業が楽になります．

しかし，実装フェーズには実装フェーズのテクニックがあります．実装フェーズでは複数の人間が作業するので，その作業のやり方で開発効率がかなり違ってきます．

次回は実装フェーズの作業のやり方の工夫について，よりリアルな内容を書こうと思います．

また，BlastJは試用版がWebページ

http://crossfire.jp/blastj/

から無料でダウンロードできるようになっていますので，ぜひ試しに使っていただけますと幸いです．

問い合わせ先
CROSSFIRE JAPAN INC.
〒160-0023
東京都新宿区西新宿7-23-9
西新宿小林ビル2F（アルディートシステム内）
TEL：03-5386-9921
FAX：03-5386-3911
URL：http://crossfire.jp