解説 3

ソフトウェア開発の新潮流
ー ソフトウェア開発スタイルの変化 ー

ソフトウェア開発スタイルは日夜変化し続けていますが、IBM の製品開発も変化してきました。

本稿では開発トレンドの幾つかを紹介します。最初は、Groovy 言語（Java™ プログラマー向け）、PHP 言語（Web プログラマー向け）などの、軽量言語による開発スタイルの多言語化傾向です。次に Apache や Eclipse などオープンソース型開発。3 番目は、Wiki やブログなどソーシャル・ネットワーク・システム（SNS）の活用が挙げられるでしょう。4 番目に、アジャイル開発を支援するために IBM が立ち上げた Jazz™ というオープン・プロジェクトをご紹介します。

最後に、ユーザーからの意見を取り上げながら製品開発を進める、コミュニティー駆動型開発について触れてみます。本号に掲載された、チェンジビジョン様の取り組みは、Jazz がアジャイル開発環境として、チームのビジョン共有を促進している事例の代表です。

Article 3

A New Wave of Software Development
- Innovation in Software Development Style -

The style of software development has been changing day and night, and development at IBM has been changing, too.

I will introduce some development trends in this article.

First is the trend of using a development style where several light languages are employed such as Groovy (for Java programming) and PHP (for Web programming).

Another trend is open source type development, such as in the case of Apache and Eclipse.

Thirdly, social network systems (SNS) like Wikis and blogs, etc. have become available for use. Fourth, there is the introduction of the open project that IBM started up to support agile development, Jazz™. Finally, there is a trend that touches on “community-drive” type development that advances CDCD (Community Driven Commercial Development) while taking in user opinions.

In the change vision’s approach described in this article, Jazz is an example of an agile development environment where the sharing of the vision of the team is promoted.

❶ 多言語化する開発スタイル

ソフトウェア開発において、開発言語、開発環境など、そのスタイルは時代と共に変化し続けています。ここでは、そうした変化について開発言語の視点から詳しくご紹介します。

多言語化とは、今まで、ソフトウェア開発のリンガ・フランカ（共通言語）として使用され続けてきた、Java の汎用的な実行環境の価値を継承しつつ、表記法の言語としては多様性を持つということです。

Java の特徴的な価値は二つあります。一つはその言語仕様の厳格さによる互換性、信頼性です。C 言語のような方言は Java にはありません。従ってマシン環境が変わっても再コンパイルする必要はありません。一度書けば複数の環境で同じように動きます。これを “Write once run anywhere” といい、実行コードに可搬性をもたらします。これは Java の大きなメリットです。

そして、もう一つの価値は、多数のハードウェア環境で動作可能な JavaVM の普遍性です。

JavaVM は Java がコンパイルした bytecode を実行する仮想マシンです。これにより、一度 Java がコンパイルしたコードは、異なった OS 上でもそのまま動作させることができるという汎用性を持ちます。これにより CPU のアーキテクチャーの違いを意識する必要がなくなることが**図 1** で分かります。

ここまでが、今まで知られてきた Java の価値です。

それを踏まえて、今回新規に採用された Groovy と

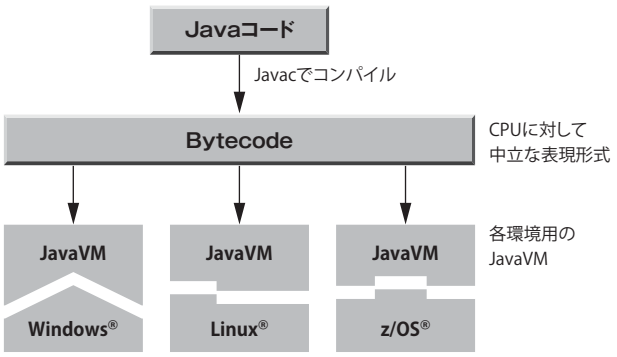

図 1. ハードウェアに中立な Java の価値

いう言語の特徴を見てみましょう。Groovy 言語自体は IBM が開発したものではありません。JSR241 で定義された、Java 言語仕様の一部です。Java 自体も多言語化が進んできており、JavaVM で動作可能な言語か各種登場してきています。

Groovy はそういった Java の派生言語の一つです。

従って実動作は JavaVM に依存します。逆にいえば、現行の Java の runtime を変更することなく Groovy の簡潔な表現力を享受することができます。通常、ビジネス環境において runtime の変更はとても大変な作業です。そのプラットフォーム上で動作するすべてのアプリケーションに影響するからです。PTF（Program temporary fix：OS リリースの間に暫定的にリリースされるフィックス）の変更を一つ当てるだけでも気を遣います。そういった runtime 変更がいらないというのは、大変なメリットです。

図 2 の Groovy [1] の動作イメージを見てみましょう。

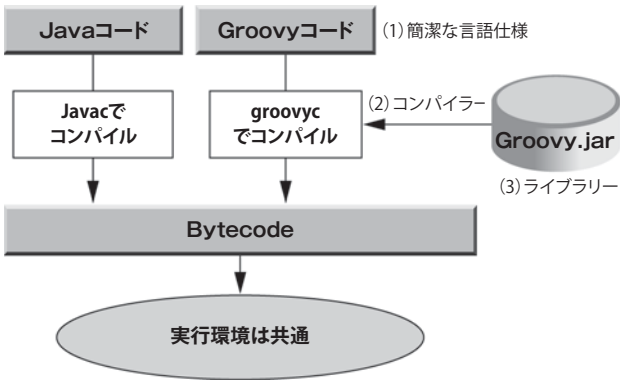

図 2. Groovy の動作イメージ

Groovy とはすなわち、Java とは異なった文法で記述されたコード（1）に対して Groovy 用のライブラリー（3）を混ぜながらコンパイル（2）することで、Java の実行環境で動作可能な bytecode を生成する仕組みです。

正確には、Groovy の動作方法は 2 種類あって、groovy のコマンド・プロセッサーで直接実行する方法と、groovyc（groovy Compiler）でコンパイルして Java 環境で動かす方法があります。しっかりした高速動作を狙う場合は、後者を使います。Groovy の特長を Java との比較で見てみましょう。まず < コード 1> は Java での HelloWorld です。

< コード 1 Java での HelloWorld>

```
1: public class HelloWorld {
2:     public static void main(String[] args) {
3:         System.out.println("Hello World");
4:     }
5: }
```

Java の場合、プログラミングのエキスパートが書いても最低 5 行になります。

これを Groovy で書き下すと < コード 2> になります。

< コード 2　Groovy での HelloWorld>

```
1: println "Hello World"
```

たった一行です。またこの書き方は、Java のコードの 3 行目の一部分であることが分かります。

つまり、本質的な部分のみユーザーが書き、それ以外の部分は Groovy のライブラリーが Java として実行可能なように補完してクラス・ファイルを完成させる方法です。これによりプログラマーは、ボイラー・プレートといわれる、繰り返し書かなくてはならないような当たり前のコード部分を記述する負担から解放されます。Groovy は Java を簡潔に書き崩したような印象があると思います。実際 Java プログラマーが Groovy を学習するコストはとても低いものです。現実的に Java を安定的に動かすためには JavaVM の runtime の性質まで意識する必要がありますが、Groovy に関しては、VM は共通なのでそこを追加して意識し、学習する必要はありません。オブジェクト生成の負荷の程度も、Garbage Collection の性能も、profiling tool も同じスキルで対応できます。プラットフォーム運用の負荷が変わらない点はメリットです。

従って Groovy 言語の活用される分野は、Java 寄りなものとなります。

Java のそばにあって、Java の苦手な部分をアシストしていくような使われ方が普及していくと思われます。

ここでの疑問は、なぜ二つの軽量言語なのでしょうか。Groovy のよさは分かったとして PHP が ProjectZero [2] に採用された理由は何でしょうか。PHP の場合は、その普及度です。世のレンタル・サーバーで PHP をサポートしていないものはほとんどありません。

Groovy は Java プログラマーのための軽量言語ですが、PHP は Web プログラマーのための軽量言語です。これによりプログラマー層の拡大を狙います。実際 ProjectZero は Ruby on Rails のような純粋な Web アプリケーション・フレームワークではなく、Lightweight

SOA（Service Oriented Architecture）ともいうべき、軽量なSOAの実行環境としての側面があります。エンタープライズ側からのアプローチはGroovyで、Web側からのアプローチはPHPで実現します。Groovyは、あくまでもJavaテクノロジーを基にしているため、Javaスキルの上への積み上げが必要ですが、PHPに関しては、すでに多数のWebプログラマーがいるので、PHPでの開発は開発者の確保の視点でも容易です。もちろんJavaVMの上で動くので、Javaの機能とは相互乗り入れ可能です。<コード3>を見てください。

<コード3 Javaオブジェクトを呼び出すPHP>

```
<?php
$date = new Java("java.util.Date");
var_dump($date->toString());
?>
```

これは通常のPHPコードですがJavaのオブジェクトを呼び出しています。ProjectZeroではP8と呼ばれるPHP runtimeをJava上に実現しており、Webとエンタープライズの架け橋としての機能を狙っていることが分かると思います。通常のWebアプリケーション・フレームワークとは異なり、SOAとの高いコネクティビティーを目指しています。

このように、ソフトウェア開発スタイルは、各言語の価値や特長を生かしながら日々変化し続けているのです。

❷ オープンソース型開発

現在多くのソフトウェアがオープンソースで開発されています。ApacheやEclipseなどで開発されるソフトウェアは商用にも使われるほどの品質の高さを誇ります。IBM社内ではこのオープンソースで行われている開発形態を製品開発にも活用しようという取り組みを行っています。いわゆる社内オープンソースという考え方です。

企業内の製品開発は、閉じた製品開発チームが製品の要求管理から、設計、コード、テスト、出荷まで行い、チーム外の人間は、社内の人間であっても製品のソースコードはおろか、製品の進捗さえも最終段階に近くなるまで知ることができないのが一般的だと考えられます。これは、製品の機密性やチーム内のコミュニケーションのしやすさなどを考えると自然なことでしょう。しかし、オープンソース開発の進化とともに、そのコンセプトは製品開発にも生かせると考えるのも自然な成り行きです。IBMではそのコンセプトの下、コミュニティー・ソースという社内サイトを立ち上げ、そこでオープンソースの開発で行われているような形態でソフトウェア製品の開発を行うという試みを2002年に始めました。コミュニティー・ソース・サイトはユーザー管理、ソースコード・バージョン管理、問題管理、プロジェクト管理などのツールを提供し、開発者は誰もがそこにプロジェクトを登録していつでもすぐに開発を始めることができます。一般的なオープンソースと違い、基本的にソースコードへのアクセス権はそのプロジェクト・オーナーが与えない限り、誰もが見えるわけではありませんが、権限さえ与えられればソースコードへのアクセスさえ可能になります。ソースコードへのアクセス権はなくても、頻繁にビルドを行い、作成した実行可能モジュールだけを公開して素早く広く使ってもらうなどということも可能です。少なくとも、どういうものを作っているかなどの情報は公開され、それを検索する手段も提供されているので、コンポーネントの再利用などの点でも活用されています。従来の開発プロセスがブラックボックス化された開発プロセスだったのに対して、ガラス張りに近い形の開発が行われているわけです。

開発者から見れば、頻繁なフィードバックや、テスターだけに頼っていたバグ・レポートなどを広いユーザーから得ることも可能であり、製品あるいは製品で利用されるコンポーネントの品質を高めるという点でも有効に活用できます。さらには、開発メンバー情報も公開されるので、開発者自身の認知度向上の場としての役割も果たされています。

すでに多くのIBM製品あるいは製品のためのコンポーネントがこのコミュニティー・ソースで開発されており、毎日数件のペースで新しいプロジェクトが登録されて次の製品やテクノロジーの元を生み出す場として活用されています。

❸ ソーシャル・ネットワーク・システム（SNS）の活用

開発におけるSNSの活用もここ数年で広まりつつある新しい流れの一つです。最も使われているSNSツールの一つがWikiやブログです。Wikiやブログは軽量の情報シェアツールとしてIBM社内においても利用が進んでいます。Wikiをはじめとする SNSツールを開発で活用する最大の利点は、情報をチームに閉じたものではなく、チーム外の人間にも提供することが容易になる点でしょう。機密性の高い情報はチーム内で閉じているべき

ものだとしても、前節のオープンソース型開発と同じように、広く利用者などのステークホルダーからの意見をくみ上げることは、製品開発やコンポーネント開発では重要になってきています。Wikiやブログなどの SNS ツールは、その公開情報の共有という観点でとても有効です。

実際に IBM 社内でも世界中の社員が利用できる全社で共通の Wiki サイトとブログ・サイトが立ち上がっており、多数の製品開発プロジェクトがその中に登録され、最新の情報を提供しています。ソースコード、問題の管理を前述のコミュニティー・ソース・サイトで行い、Wiki でそのほかの情報共有を行い、ブログで情報発信するという形態を取る開発プロジェクトも多数行われています。その製品にかかわる人間から見れば、今まで知り得なかった情報が公開され、フィードバックを与えられることは非常に有用であるということが理解できるのではないでしょうか。

❹ アジャイル開発の台頭

オープンソース型開発、SNS ツールが開発基盤での新しい流れだとすると、アジャイル開発は開発プロセスにおける新しい流れです。アジャイル開発において重要視されることとして、「常にステークホルダーの意見をくみ上げること」と、「変化に柔軟に対応すること」が挙げられます。ステークホルダーとはお客様、営業、開発など、その製品にかかわるすべての人間ですが、それらの人間から常にフィードバックを受けるべきというのがアジャイルでは基本原則［3］の一つだと定義されています。もうお分かりかもしれませんが、オープンソース型開発や SNS ツールの活用の利点もここにあります。つまり、今までの閉じたチームでの開発から、ステークホルダーからのフィードバックを常に受けながら日々の開発を行うという形態を、プロセスとして着目しているのがアジャイルであり、ツールとして着目しているのがオープンソース型開発で提供されるツールであり、SNS ツールの活用なのだと考えられるわけです。

もう一つのアジャイルでの重点項目の「変化に柔軟に対応する」ことに対して、アジャイルでは開発全体を細かな開発期間（イテレーション）に区切り、それぞれの期間で常に動くものを作るという手法を取り入れています。これによって、イテレーションの区切りごとにフィードバックを得、変化に追従することを可能にするわけです。さらに、アジャイル手法によっては、15 分程度のミーティングを毎日持ち、チーム内の状況判断をし、チーム全体での最適化を図るように推奨するものもあります［4］。毎日の現状をチームの全員が把握し、自分のなすべきことを理解することが重要になってきます。

IBM はそのような新しいチーム開発環境をサポートするために、Jazz［5］というオープン・プロジェクトを立ち上げました。Jazz プロジェクトは次の節で紹介するコミュニティー駆動型開発という形態で開発を行っているプロジェクトで、そのプロジェクトの最初の成果物である製品が 2008 年 6 月に出荷が開始された Rational® Team Concert（RTC）［6］という名の製品です。この RTC は、アジャイル開発における、チーム管理、イテレーション管理、ソースコード管理、ワーク・アイテム管理、ビルド管理などの機能を提供し、幾つかのアジャイル開発手法をプロセス・テンプレートとしても提供し、アジャイルにおける開発者およびプロジェクト管理者の日々の作業に対して自動化や助言などのサポートを行います。さらにインスタント・メッセージングのようなメンバー間で即座にコミュニケーションをとるための仕組みなどのチーム内コラボレーション機能も提供します。開発者は Eclipse ベースあるいは Web ベースのクライアントを通して、RTC で管理されたさまざまなプロジェクト情報にアクセスすることが可能です（**図 3**）。

このような機能によって、アジャイルなどのプロセスを利用者が自然に実行していくことを支援します。

アジャイル開発の考え方は今後の開発において広く使われるようになっていくと考えられます。それとともにそれをサポートする Jazz/RTC のようなツールも重要な役割を果たしていくことになるでしょう。

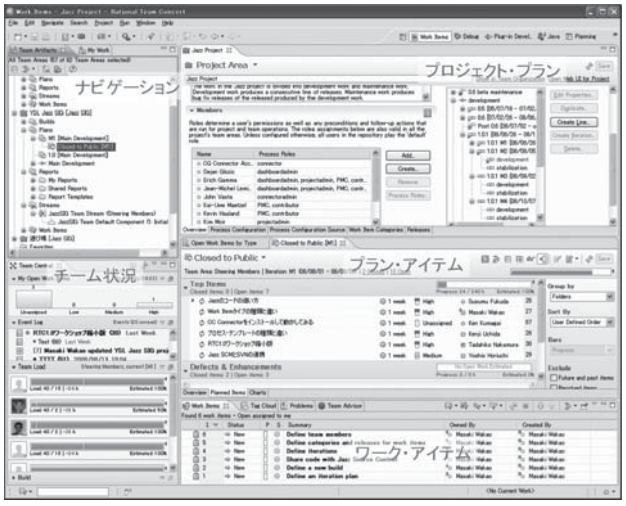

図 3. RTC Eclipse クライアント画面例

❺ コミュニティー駆動型開発

IBM では社内オープンソース開発の考えをさらに進め、社内だけではなく、真のユーザーからの意見も取り入れながら製品開発を進める新しい開発形態を CDCD（Community Driven Commercial Development）と呼び、その考え方に基づいたプロジェクトを立ち上げ始めています。前述の ProjectZero と Jazz です。これらのプロジェクトは、開発にかかわる情報のほとんどをインターネット上で公開しています。公開される情報には、例えば、リリース・スケジュール、開発メンバー、進捗状況、デザイン・ドキュメント、ソースコード、バグ・レポートなどを含みます。現状ではソースコードの変更権限や、製品化の権限が IBM にのみ属するなどの部分を除いては、いわゆるオープンソースとほとんど同じレベルの情報を公開しているわけです。それによって、開発の非常に早い段階から、利用者のフィードバックを得ようとしているのです。すでに述べてきている通り、ソフトウェア開発、特に製品開発では、いかにそのソフトウェアを使う人と作る人との距離を近くするかに大きな重点を置いてきています。特に市販製品では利用者も千差万別です。さまざまな意見を持った利用者に満足感ある製品を提供するためには、利用者からの意見を製品出荷後ではなく、製品開発の早い段階から得る必要があり、そのためにはオープンソースのような開発形態は非常に有効であるという認識をしているのです。コミュニティー駆動型開発は、そのオープンソース開発を企業の製品開発に取り組んだ新しい開発形態といえます。

❻ まとめ

ソフトウェア開発においての変化とは常に、いかに早く、効率的に、良いものを作るかという方向で進んでいると考えられます。そのために、IBM のように、商品としてソフトウェア製品を開発する企業内でも、オープンソースやアジャイルなどの手法を積極的に取り入れ、利用者の声を最大限に聞くことに取り組んでいます。また ProjectZero や Jazz のように、IBM 自体が新しい開発スタイルで製品を投入し始めています。新しいプロセスと新しいチーム開発体制でのソフトウェア開発はすでに現実のものになってきているといえます。

［参考文献］

［1］ Groovy, http://groovy.codehaus.org/Japanese+Home
［2］ ProjectZero, http://www.projectzero.org/
［3］ Agile Principles, http://agilemanifesto.org/principles.html
［4］ Scrum, http://www.scrumalliance.org/
［5］ Jazz.net, http://jazz.net/
［6］ Rational Team Concert, http://www.ibm.com/software/awdtools/rtc/

日本アイ・ビー・エム株式会社
ソフトウェア事業部　ソフトウェア技術推進
IT スペシャリスト

根本 和郎 Kazuo Nemoto

［プロフィール］
大和ソフトウェア開発研究所で音声システム関連の開発を長年担当、13 件のソフトウェア特許を持つ。2007 年から渋谷テクニカル・エクスプロレーション・センター。「渋谷テクニカルナイト」という技術者向け先進セミナーを主催。最近は、「どう書く?org」という多言語プログラミングコロシアムサイトに Groovy でエントリーしている。

日本アイ・ビー・エム株式会社
ソフトウェア開発研究所
シニア・テクニカル・スタッフ・メンバー

若尾 正樹 Masaki Wakao

［プロフィール］
1990 年、日本 IBM 入社。ホームページ・ビルダー®、Rational Application Developer などの製品開発に従事。現在は Rational チーム開発製品の開発に携わる。