

IBMリサーチによるソフトウェア・エンジニアリング 関連技術への取り組みと展望



日本アイ・ビー・エム株式会社
理事 東京基礎研究所 所長
久世 和資

Kazushi Kuse

Director

IBM Tokyo Research Laboratory
IBM Japan, Ltd.

IBMの基礎研究所は、米国に3カ所、欧州に2カ所、アジアに3カ所あります。東京基礎研究所もIBMリサーチ部門の一員として、グローバルな連携と協力体制の中でさまざまな研究開発や取り組みを行い、ソフトウェア・エンジニアリングも大きな研究テーマの一つとなっています。

ここでは、東京基礎研究所 所長 久世 和資が、ソフトウェア・エンジニアリング関連の技術的な展望を語るとともに、IBMリサーチにおけるソフトウェア・エンジニアリング戦略チームのリーダーであるダニー・イエリンがソフトウェア・ライフサイクル戦略を解説します。

なお、この戦略策定には米国のIBMリサーチだけではなく、ハイファ基礎研究所のガビ・ゾディック、インド基礎研究所のサティシュ・チャンドラ、東京基礎研究所の三ツ井 欽一、中国基礎研究所のゾン・ティアンが主要メンバーとして参画するなど、文字通りワールドワイドな取り組みとなっています。

Management Forefront 3

SPECIAL ISSUE: Software Engineering

IBM Research's R&D in Software Engineering-related Technologies and Its Business Prospects

Currently, IBM has three basic laboratories centers in the United States, two in Europe and three in Asia. As a key member of the IBM's Research section, Tokyo Research Laboratory has been conducting a wide range of research, including the software engineering- one of its major research themes.

In this section, Kazushi Kuse, director of Tokyo Research Laboratory, will talk about the prospects of developing the software engineering-related technologies, and then Danny Yellin, the leader of IBM Research's Software Engineering Strategy Team, will explain the IBM's software life cycle strategies.

IBM Research's strategies are decided as a truly worldwide project, involving such prominent researchers as Gabi Zodik of Haifa Research Laboratory, Satish Chandra of India Research Laboratory, Kinichi Mitsui of Tokyo Research Laboratory, and Zhong Tien of China Research Laboratory.

四つの階層で企業をモデル化

現在、IBMリサーチでは、MDD(Model Driven Development : モデル駆動型開発)を最重要課題の一つとして研究を行っています。この概念が登場した背景は幾つかありますが、IBMが推進するオンデマンド・ビジネスを実現する主要なIT(Information Technology : 情報技術)の一つになっています。

オンデマンド・ビジネスは、会社やビジネスの効率を分析して、非効率な部分を指摘したり、自動修復したりすることができます。さらには、ビジネスの内容・規模ややり方を変更したり、新しいビジネスを始めるときにも、ITシステムが迅速にかつ柔軟に対応できるようになります。IBMでは、オンデマンドを実現するために、四つの階層で企業をモデル化することを提唱しています。上位から順に戦略層、ビジネス・プロセス層、プラットフォーム独立のシステム層、プラットフォーム依存のシステム層となり、上位2層のビジネス層、下位2層のシステム層に大別されます。各層が密接に上 / 下双方向に関連付けられ、企業全体がモデル化され、それが企業の日常のビジネス活動を支援するシステムとなります。

つまり、この4層のすべてがモデルで表現され、実装されることとなります。上位から下位に向かうに従ってモデルは詳細化され、ITシステムで実行可能な形式になっていきます。このように、開発のすべての局面でモデルを中心に開発を進めていくのがMDDであり、ビジネス戦略のモデルから、ビジネス・プロセスのモデルへ詳細化されます。さらに、ビジネス・プロセス

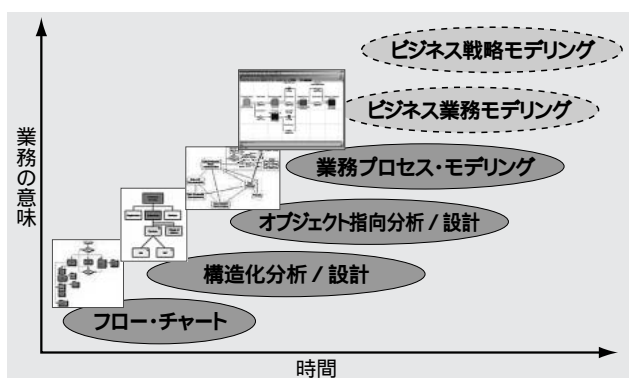


図1. 企業全体のモデル化

のモデルは、プラットフォーム独立のシステム・モデルへ変換され、最後には、プラットフォーム依存のシステム・モデルへと開発が進んでいきます。各層の中には、さらに幾つかの詳細化のレベルが存在し、そのレベルごとにモデルが対応しています(図1)。

オンデマンド・ビジネスを支える企業システム

開発は、モデルを介して上位層から下位層に向かって進められますが、ITシステムの実行時には、それぞれのビジネスや組織の内容と効率に対するフィードバックが、逆方向すなわちモデルを通して上位に向かっていきます。各トランザクションの効率を測定するITモジュールの測定値が、上位のアプリケーション・モデルに上げられ、対応するビジネス・プロセス・モデル上に反映され、最終的に、戦略モデル上で会社全体のビジネスが順調に機能しているかどうかを示されます。

仮にビジネス上の問題があったときには、戦略モデルやプロセス・モデル上で操作すると、それがモデルを介してIT層まで反映されることとなります。自動的にITシステムが変更・拡張されることが理想ですが、完全自動が実現するまでは、ツールや人による開発作業が入ります。しかし、これもモデルを介して首尾一貫して行われるため、ビジネス上の戦略・プロセスとITシステムの整合性はより強固なものとなります。その結果、迅速なシステム開発やシステム対応が可能となり、上位のビジネス・モデルと密接に関連付けられます。会社全体の効率や様子をビジネス戦略のレベルで観察することができ、ビジネス戦略を迅速かつダイナミックに変更でき、ITシステムも新しいビジネス戦略に柔軟に追従することが可能となります。

このようにオンデマンド・ビジネスを支える企業システムは、ビジネス戦略からITシステムまで、モデルを通して関連付けられます。戦略変更はモデルを通して上位層から下位層に実現され、ビジネス実績は逆に下位層から上位層へ伝播されます。その実績によって戦略を変更するわけですから、オンデマンド・システ

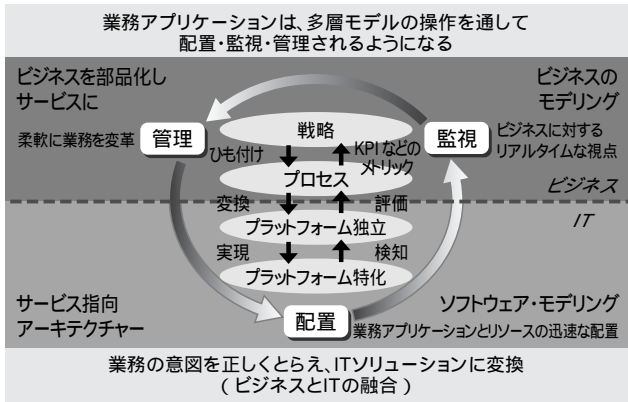


図2. ビジネスとITの融合

ムでは、4層間を通じた監視 管理 配置の繰り返しとなります(図2)。

CBMとODOE

最初に述べたように、四つの層はビジネス層とシステム層に大別され、各層にはモデル駆動型開発に次ぐ重要な概念がさらに二つ追加されています。

それは、CBM(Component Business Model)とODOE(On Demand Operating Environment : オンデマンド・オペレーティング環境)です。

CBMによって、ビジネス戦略はモデル化されます。主要ビジネス戦略は、コンポーネントとして定義されます。各コンポーネントは、戦略の重要性(短期 / 中期 / 長期)とビジネス分野(販売・会計・購買など)の2軸でマッピングされます。さらに、ビジネス上の改革の優先順位が決められます。優先順位の高いものから改善していく必要があります。CBMの各コンポーネントとコンポーネント間の関係は、ビジネス・プロセス・モデルで表現され、そのビジネス戦略を実現するための組織や業務プロセス、業務ルールが記述されます。

ODOEは、オンデマンド環境を実現するための実行プラットフォームです。ODOE上のシステムを開発するために、ITシステムのモデル化が必要になります。先ほどのビジネス・プロセスとプラットフォーム独立のITシステムのモデル化には、UML[®](Unified Modeling

Language : 統一モデリング言語)が使われることがあります。

さて、オンデマンド・システムを、会社の創設に合わせて構築するなら、戦略モデル ビジネス・モデル システム・モデル ITモデル 実装と進みますが、通常、既に会社は存在し、既存のITシステムが稼働しています。この場合に威力を発揮するのが、LT(Legacy Transformation)です。LTは、既存のシステムのコードから、ビジネス・ルールやプロセスを抽出します。古いプログラミング・モデル(例えば、CICS[®]やIMS[™])を新しいプログラミング・モデル(例えばJ2EE : Java[™] 2 Platform Enterprise Edition)に変換し、既存のITシステムを新しい動作環境に移行するという役割もあります。このレベルでは、プラットフォーム独立のITモデルの導出までです。しかし、より重要なLTの役割は、IBMの推進するオンデマンドに、いかに既存のシステムを融合させるかということです。このためには、プラットフォーム依存のシステム・モデルから、ビジネス・プロセス・モデルまでを変換することが必要になります。

オンデマンド・ビジネスとMDD

実は約8年前、コンサルティング・グループ(現・ビジネス・コンサルティング・サービス)と東京基礎研究所が協業して、あるお客様のビジネスを事例に、柔軟に適合する企業システムのモデル化を行いました。私は、オブジェクト指向グループを担当していたのですが、このテーマは、研究所にとっても大変重要なもので、約4カ月にわたって、週の半分は徹夜という勢いで、この仕事にのめり込みました。

コンサルティング・グループでは、現在のCBMに当たるビジネス・ブループリントという戦略モデルを構築しました。この戦略モデルに加えて、法規制などの環境モデルや、組織と人のモデル、設備の物理モデルを作り、各モデルごとに5年くらいを想定してその間に変化する可能性があるものを特定しました。

その後、ビジネス・モデルに当たるものを、オブジェ

クト・モデルとイベント・チャートでモデル化しました。システム側は既存システムとは連携させずに、プロトタイプ・レベルでしたが、人、物、ビジネス・ルールは、分散オブジェクト(当時は、DSOM : Distributed System Object Model)で実装し、ビジネス・プロセスは、ワークフロー・システムを使って実現しました。アプリケーションに登場する端末画面も、ビジネス・プロセスやルールを変えることで変更が起こるため、柔軟に対応できるようにSmalltalk(VisualAge® Smalltalk)を活用しました。

モデルやオブジェクトのコンポーネントを使ってモデル化したところや、ビジネス・ルール、法規制の変更に柔軟に適応するITシステムという考え方は良かっ

たのですが、問題がありました。それは、各モデル間がなかなかスムーズに連携できなかつたり、上流から下流に変換する際に誰でも同じようにできる一定ルールがなかったのです。これを解くカギが今にして思えばMDDだったのです。

このように、MDDは、ソフトウェア・エンジニアリングの世界を大きく変える概念であり、オンデマンド・ビジネスの実現には不可欠なものです。後半は、IBMリサーチのソフトウェア・エンジニアリングの戦略チームのリーダーであるダニー・イエリンに、MDDを中心に、研究動向や新しい展開・展望について説明してもらいます。

ソフトウェア・ライフサイクルの研究動向



IBMコーポレーション
Director of Programming Models & Tools
in IBM Research

ダニー・イエリン

Danny Yellin
Director of Programming Models & Tools
in IBM Research
IBM Corporation

IBMリサーチの取り組み

IBMリサーチでは、要求工学(Requirements Engineering) 分析、設計、コーディング、テスト、運用、メンテナンスなどソフトウェア開発プロセスのすべての局面において、オンデマンド・プラットフォームの実現や、ソフトウェア開発コストの削減、システムの安全性の強化、ソフトウェア資産の再利用などの研究に取り組んでいます。

この分野は、WebSphere®、Rational®、Lotus®な

どのソフトウェア製品から、AMS(Application Management Services)まで多岐にわたり、IBMのビジネスに深く関係しています。AMSは、お客様のIT戦略に即した、アプリケーションの企画・計画・開発・保守・運用をサービスとして提供するものです。MDDやLTなどのこの分野の研究が、BCS(Business Consulting Service)の各種分野のサービス・プラクティスを通して、ITとビジネスの融合を実現することも期待されています。

ソフトウェア・エンジニアリングにおける技術動向

IBMリサーチは、市場における二つの重要な動きに注目しています。

一つは、ビジネスのオンデマンド化です。

これはITの効率化にもつながり、ITの合理化や開発や保守のアウトソーシング化、既存のITインフラ

トラクチャーの再利用を意味します。オンデマンド・ビジネス・モデルは、さらに深いレベルでのインフラストラクチャーの統合を推進します。こうした統合は、通常、組織の枠を超えたものになるため、さらに強力な標準化が重要になります。私たちは、インダストリーごとの標準(例えば、各インダストリー用のXMLスキーマやWebサービス)が、組織を超えたシステム統合を可能にすると考えています。

もう一つは、Microsoft[®].NETと標準ベースのJ2EEが、開発者の選択する主要な開発環境として絞られていくであろうということです。

モデル指向開発、標準化、Webサービスを通じてこの二つを結ぼうとする動きがありますが、しばらくは二つの環境の並存が続きそうです。しかも、この二つのプラットフォームの違いは、システムの複雑さを引き起こしています。

例えば、ポータル・システムが、ワークスペース・クライアントと協調動作するには、多くの複雑なオプションを導入する必要があります。その一方で、Microsoft .NETサーバーも、サーバー機能の複雑化につながっています。こうしたプラットフォームの品質・信頼性・利便性・拡張性が十分なものになれば、業務アプリケーションもその上で開発されるようになるでしょう。ミドルウェアがコモディティー化するころには、これらのプラットフォームは、さらにシステム・スタックの上流に向かい、パッケージ化されたソリューションによる価値や、ビジネス指向の機能を供給するようになるはずです。

それでは、幾つかのキーワードを挙げて、もう少し具体的に技術動向を見ていくことにしましょう。

【ソフトウェア産業界における技術動向】

ソフトウェア産業界が最も注目している技術動向の一つがMDDです。MDDはモデルを開発時の共通の概念として使います。ソフトウェアは、抽象度の異なる複数モデルの変形によって開発されます。例えば、最上流の要求仕様はUMLで表現され、さらに再利用可能な動作環境に依存しない実行モデルに変換され、最終的に特定の計算機環境で動作するコードにまで変換されていくのです。

MDDの特徴は次の通りです。

- ・ ソフトウェアとアプリケーション開発の複雑度を緩和します(例えば、アプリケーション、プラットフォーム、言語の意味的なギャップを吸収します)。
- ・ ソフトウェア部品とアーキテクチャー・パターンの統合化と再利用を強化します
- ・ ビジネスとITのより良い関連を実現します。
- ・ 要求仕様に良く適合したソフトウェアの開発を可能にします。
- ・ ソフトウェアをビジネスや技術の変革に、より迅速に適合します。

MDDが普及すれば、ソフトウェア開発の革新的な手法となるはずで

【大学・研究機関における技術動向】

私たちと同様に、大学・研究機関でもソフトウェア・エンジニアリングの研究を進めていますが、その取り組みは、実世界における現実問題の解決から出発しているというよりも、むしろ革新的な手法や技術を目指しています。研究機関における技術動向を分野別に見ていきましょう。

・ MDD

多くの研究グループがMDDの研究開発を積極的に進めていますが、中でもOMG(Object Management Group)のモデリングに対する活動は、大学・研究機関も注目しています。また、UMLの意味論(特にBehavioral Semantics)や、モデル変換、ドメイン指向モデリング、メタモデリングなどに関心が集まっています。

・ AOSD

MDDと同様に、AOSD(Aspect Oriented Software Development: アスペクト指向ソフトウェア開発)についても、研究グループ間による広範囲かつ深い議論が進んでいます。AOSDの研究は、プログラミング言語とソフトウェア・エンジニアリングの学会でスタートしましたが、現在ではAOSD単独の学会や研究会があり、linguistic issues、software weaving、aspect interactionなどの問題が活発に議論されています。

・モデル検証と定理証明

ソフトウェア品質の研究をリードしている分野です。この分野の研究には、abstraction、compositional techniques、model formalisms、hybrid approachesなどが含まれます。これらの手法や技術は、近年技術的な進展を見せていることもあり、今まで以上に大きな期待が持たれています。モデルが議論の中心になっているMDDは、モデル検証の分野における革新的な進歩に大きく貢献するはずで

・ソフトウェア・プロセス管理

最近になってアジャイル・ソフトウェア開発(例えば、extremeプログラミング)の議論が活発になっています。アジャイル手法の適応性や有効性、オブジェクト指向プログラムのためのリファクタリングのサポート、アジャイル・ソフトウェア開発のためのツールなどの関心が高まっています。

【標準化】

この分野の標準化活動は、OMGのMDA(Model Driven Architecture:モデル駆動型アーキテクチャー)、Webサービスの標準化、JCP(Java™ Community Process)での議論などが代表的です。

MDAは、OMGで提案およびリードされている標準の枠組みであり、UML、MOF(Meta Object Facility)、XMI(XML Metadata Interchange)などから構成され、J2EE、Microsoft .NET、Webサービス、CORBA(Common Object Request Broker Architecture)などの実行環境から、ソフトウェア設計を独立させ、組織を超えて稼働するソフトウェア設計も可能にします。

Webサービスの標準化は、複数の標準や団体が存在し、UMLに比べると若干統制が取れていません。基本要素であるSOAP(Simple Object Access Protocol)、WSDL(Web Services Description Language: Webサービス記述言語)、UDDI(Universal Description, Discovery and Integration)などの標準化が完了すると、標準化の議論は、さらに複雑なパートであるセキュリティ、トランザクション管理、高信頼メッセージなどに興味を移しました。Webサービスには、OASIS(Organization for the Advancement of

Structured Information Standards)とW3C(World Wide Web Consortium)という二つの主要な標準化団体があり、加えて、各種のインダストリーに依存した標準の重要性が増しています。

また、幾つかの新しいJSRs(Java Specification Requests)は、JavaでXML、Webサービス、UMLをサポートすることになります。これはJ2EEが、Webサービスのサポートと、UMLのより強固なインテグレーションを可能にすることを意味しています。

IBMリサーチの研究方針

私たちの研究は広範にわたりますが、特に力を入れているエリアは次のようなものになります。

【ソリューション開発のライフサイクルを支援するモデル駆動型開発】

現在、ソフトウェア開発の分析と設計のほとんどにおいて、ソフトウェアのモデルが使われています。モデルをさらに上流で使うことによって、ビジネスのモデル化が可能になります。また、ビジネス・モデルから下流方向を定式化することにより、モデルからのコード生成の機会が増えます。その結果、要求仕様から設計、開発、テスト、保守、拡張まで、つまりソフトウェア開発の全フェーズでモデルは不可欠なものとなります。言い換えれば、より定式化されたモデル群の導入により、ソフトウェア・エンジニアリングのブレークスルーが可能になります。こうしてアプリケーション開発の効率、信頼性、品質の画期的な向上が可能になり、複雑なサブシステムやそれらの通信の整理ができるようになります。モデリング言語の利点は、問題領域とモデル表現のセマンティック・ギャップを解消し、より強力な関連付けができることですが、そのためにはUMLベースの領域依存言語を拡張する必要があります。

MDDオートメーションは、高レベルの要求仕様から実行可能なアプリケーションまでのモデル駆動のプロセスのことです。その実現には、プロセスとツールが密接に連動するモデル・ベース・ツーリングが必

要になってきます。オープン・スタンダードはこの分野でも重要です。

【簡易ソフトウェア開発とデリバリー・モデル】

ソフトウェア開発の大きな課題の一つは、対象とするソフトウェアの開発と運用の複雑さです。IBMの分散コンピューティング環境も例外ではありません。IBMリサーチは、この複雑さの解決に取り組み、より単純な抽象化の手法とツールの研究開発を行っています。今日のソリューション開発では、まだまだ多くの小さなコンポーネントを苦勞してつなぎ合わせる必要がありますが、この分野の研究が進むことで、開発者のソリューションに対する期待とプラットフォームの特性のギャップを小さくできるでしょう。

【ソフトウェア進化】

ソフトウェアの保守と既存のアプリケーションの活用は、企業が直面している主要なチャレンジの一つです。企業のビジネス・ロジックは、メインフレーム・システムおよび莫大なビジネス・データや処理フローの中に存在しています。それにもかかわらず、こうしたシステムを扱える技術者は減りつつあり、この問題はより深刻化しています。

そこで各企業は、新規のデザイン・フレームワーク（例えば、ユーザー・インターフェースのビジネス・ロジックやデータ・アクセスからの分離）へのシステム移行や、J2EEなどの新しい実行環境への移行のペースを速めなければなりません。さらに、オンデマンド化などによるビジネス要求の変革が、企業のシステムの移行を加速するはずで、そのためのツールとして、COBOLからCやC++、Visual Basic、Javaまでの広範囲のサポートが必要になってきます。

既存のレガシー・アプリケーションの変換も、この目的に適合していますが、それでは不十分です。扱いやすく、保守性や拡張性が高いようにアプリケーションやシステムを構築するためのサイエンスが必要であり、さらにシステム品質の向上と保持も重要です。

設計における問題点は一般に共通であることから、ソフトウェア進化においてプログラム解析技術を

統一するツール・アーキテクチャーは重要な役割を果たすはずで、効率的なプログラム形式や、拡張性の高いプログラム・リポジトリ、柔軟で効率的な問い合わせ機能、コントロール・フローおよびデータ・フロー・アルゴリズム、プログラム・スライスの技術、プログラム分析用のルール・ベース・システム、リファクタリングと変換、アスペクトからプログラム・ウィービング機構などが関係しています。

IBMリサーチでは、ソフトウェア進化技術のためのアーキテクチャーを設計しており、前述したリサーチ技術のほとんどが、このアーキテクチャー上で共通および共有化されます。

【ソフトウェア品質】

ソフトウェアがより広範囲で使われるようになったため、ソフトウェアの障害はより深刻な問題を引き起こします。この分野における研究の目標は、ソフトウェアの商品化や開発のスピードを落とすことなく、ソフトウェア品質を向上させることです。具体的には、新しい世代のツール群を研究開発しています。

これらのツールは、プログラムの意味上のエラーを判定するだけでなく、プログラミング・モデルに反するエラーやモデルのベスト・プラクティス（例えばJ2EEの最良の使用法）に合わないエラーも判定することができます。この技術は、各プロジェクトでモデルのベスト・プラクティスを推進するためにも有効です。

IBMリサーチでは、ベスト・プラクティスのパターンのパフォーマンス・チェックをする解析エンジンを利用して、ベスト・プラクティスの活用を進める研究にも取り組んでいます。この技術をより広範囲に適用できるように拡張し、新しいベスト・プラクティスのテンプレートが容易にツールに導入できることが必要です。さらに、複数のソフトウェア解析ツールがお互いに強調して動作するアーキテクチャーが必要になってきます。これらのツールは、エラーの特定だけでなく、自動的にプログラムを最適化するのにも使われます。