異種ベンダツール連携による統合開発環境基盤

Integrated Development Environment Enabled Integration of Industry Leading Tools

松　倉　　司

要　約　Windows システム環境でのアプリケーション開発ツールは，最近ますますその種類が増えるのみならず，その機能強化も急速である．世の中に存在する数多くの開発ツールの中から苦労してみずからのアプリケーション開発に使用するツールを選択したとしても，これらのツールの機能は，開発工程の一部分をカバーするだけである事が多い．例えば，モデリングツールは，オブジェクトモデルを生成する作業に有用であるが，コードを記述・編集する機能については，言語系製品を凌駕する事はない．開発工程としてはモデリングからコード作成・編集及びデバック・テストと進みたいにも拘わらず，それぞれのツール間でのギャップの存在を意識せざるを得ない．また，一度使用を決めたツールを，より機能に優れた他のツールに置き換える事は，それまで作成してきたソフトウェア資産を放棄する結果をもたらしかねず，止む無く旧来ツールの使用を継続しなければならなくなる．

IntegratePlus は，これらの課題を解決し，eNT でのアプリケーション開発環境を整備するためのインフラストラクチャの役割を果たすものである．その具体的な機能としては，種々の異なるベンダーから提供される開発ツール間での情報の受け渡しをオブジェクト・リポジトリを介して可能とし，シームレスなツール連携を実現するものである．

本稿では，IntegratePlus が異種ベンダーツールをどのように連携させ，統合開発環境を実現しているかの概要を報告する．

Abstract　Recently, many kinds of application development tools have been rushing in the open systems software marketplace, and their functionalities have been enhanced dramatically. It is quite defficult for customer to select the tools for their own development work among many kinds of tools. Even if they could select an optimal tool on the basic of their own requirements from market, these tools may not cover all of works throughout development process, only a part of the entire process.

The interoperability among these tools is insufficient. A modeling tool functions as only a modeling tool, and a language/editing tool functions similatly as only the language/editing tool. Even when we could combine information from one tool to anather tool, the usability gaps exists quite obviously. Moreover, the transition from once selected tools to a new more excellent tool is defficult, becauce this transition may bring the result that abandons the current software assets. Customers have to stay current tools in use continuously.

IntegratePlus will resolve these problems and play the role of the infrastructure for improving the application development environment in eNT. As the concrete function, IntegratePlus will realize an integrated development environment that seamlessly works with tools from several different venders.

This paper will describes how IntegratePlus has realized new development environment by integrating tools for a variety of vendors.

1. はじめに

パーソナルコンピュータ（Personal Computer：PC）の普及により，ソフトウェア開発は作業の大部分が PC を用いて行われる形態になってきている．また，ソフトウェア開発ツールは急速な改良が行われ，特にビジュアル開発ツールの普及が著しい．このようなことからアプリケーション開発は，当然の結果 PC の上でビジュアル開発ツールを用いて行うことになった．しかし，ここにある種の問題がある．

市場には多くのビジュアル開発ツールが存在するため，どのツールを開発工程のどの部分で使用するかを決定する難しさがある．また，使用ツールを決定したとしても，そのツールは一部の工程をカバーするのみなので，意図するアプリケーション開発に際しては異なるベンダー提供のツールを連携させて工程間を進行させる必要がある．更に，開発ツールの進展は非常に著しいのでより適切なツールが出てきた場合にはスムースに入れ替えができるような構造であることが必要となる．

このような要求を満たすものとして異種ベンダツール連携による統合開発環境基盤を案出し，そのコンセプトを IntegratePlus としてまとめた．

本稿では，IntegratePlus がどのような機能を実現するものであるかを紹介する．

2. IntegratePlus の概要

IntegratePlus は，環境の変化に迅速に対応できる企業情報システムの開発を可能とするために以下のような機能を提供する．

① ビジネス中心の開発アプローチ
② コンポーネントベース開発方法
③ 開発ライフサイクル管理
④ 異種ツール連携機能

2.1 ビジネス中心の開発アプローチ

従来の企業情報システムの開発の形態は，図 1 左側に示すような手順をとっていた．

・当該企業業務の一部分を分析・設計して複数個のサブシステムを開発する．
・それら複数のサブシステムを組み合わせ，一つのアプリケーションとする．
・これらを更に組み合わせて企業全体のビジネス・プロセスに対応させたアプリケーションシステムとして構築する．

この開発アプローチは，ボトムアップ的な企業情報システム開発方法といえるものであるが，次のような問題点を有している．

・それぞれのアプリケーションを組み合わせた結果が，当該企業が求めるシステム要件に完全に一致しているか否かを見極めることが容易ではないこと
・企業のビジネス・プロセスに変更が発生した場合に，どの部分を見直すべきかの切り分けおよび改造が容易ではないこと

また，この方法で開発された企業情報システムを使用する場合には，アプリケーションの動きに業務プロセスを合わせることになりがちという問題もある．環境変化に迅速に対応した企業情報システムを開発する，または新しいビジネス・モデルを規定し競合相手に先駆けて情報システムを開発し市場を獲得するためには，従来のボトムアップ型アプローチではなく，トップダウン型にビジネス・モデルから個々のアプリ

ケーションを導き出し，更にサブシステムを生成する開発形態を実現することが要求される．

この要求実現のためには，企業のビジネス・モデルを作成し，企業における人々の役割・業務プロセス・個々の業務内容・相互作用の対象となるデータベースや他システムとの関連性を定義する必要がある．すなわち，ビジブルで理解しやすいビジネス・モデリングと，関係するソフトウェア開発ツール間を連携させることで，この要求を実現するのである．

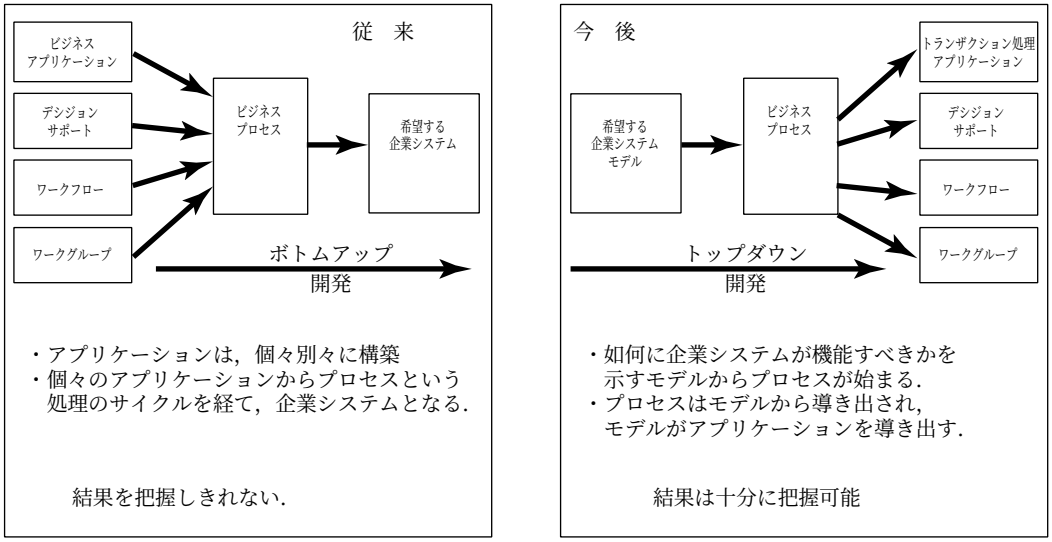

図 1　アプリケーション開発のアプローチの変化

2.2　コンポーネントベース開発方法

アプリケーション開発の生産性向上をもたらすものとして，部品再利用による開発工数の削減が久しく言われてきたが，なかなか普及までに至っていない．この理由として，ある特定環境内でのみコンポーネントの共有が可能になるだけで，再利用の範囲が限定的であることがあげられる．しかし，この障害は次のような形で改善されつつある．

1)　コンポーネントモデルとして 3 種のデファクトスタンダード（CORBA，COM および JavaBeans）が普及してきたことにより，これまでとは異なり，再利用コンポーネントを開発するための技術基盤が明確になったので，再利用についての大きな課題の一つが解決された．

2)　コンポーネントの仕様を明示して，再利用を図る技術基盤としては，オブジェクト指向の実質的な標準規格である UML[*1] を用いることが有効になってきている．この作業を行うモデリングツールが提供されているため，開発者の作業負荷は大幅に改善されている．このようなツールを使用することでハイレベルな抽象度での情報が保持・参照できることになるので再利用を容易にすることが可能となっている．

このような技術基盤・開発ツールが整備されることに加えて，再利用の効果をあげるにはコンポーネントの共有を組織的に行えることが必須である．共有を組織的に行うにはリポジトリにすべての必要情報を格納して開発者全員が同じように参照できる仕組みのもとで作業を進めることである．このリポジトリの中にはコンポーネントに関わるすべての情報（コンポーネントの仕様，インタフェース情報，コードなど）を格納し，容易に参照できるようになっていなければならない．

リポジトリ内部に格納されているコンポーネントに関する属性の一覧表示機能や関連性表示機能および検索機能などをコンポーネント管理機能として提供することにより，必要なコンポーネントを取り出し，再利用する利便性を実現している．

2.3 開発ライフサイクル管理

IntegratePlus は開発作業の全工程をサポートするインフラストラクチャであり，この全工程を統一的に管理する機能こそが IntegratePlus の主機能である．この開発工程については，6 種の工程を規定している．これらの工程間の関連は，図 2 に示しているとおりである．

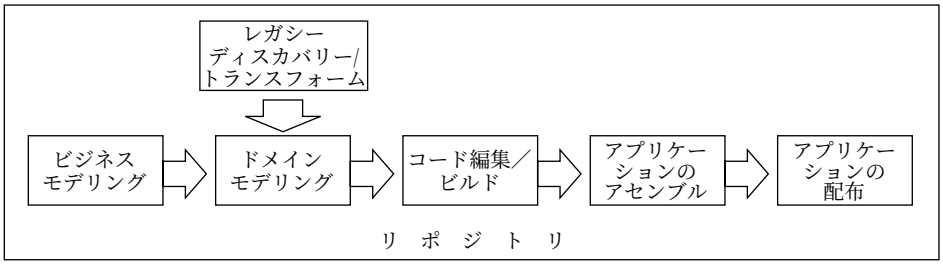

図 2　IntegratePlus の開発ライフサイクル管理

1）　レガシーディスカバリー/トランスフォーム：

企業には既に多くの情報資産（アプリケーション・プログラムやデータベースなど）が存在している．これらの情報資産を保護し有効活用する観点から，新たに開発されるアプリケーションシステムからこれらを再利用可能とすることが必要とされる．この既存資産活用を実現するための作業を行う工程であり，レガシーラッピング（Legacy Wrapping）による既存資産活用を実現しようとする工程である．

2）　ビジネスのモデリング：

企業の活動形態内容として，例えば企業としてどのような活動を行うのか，そのために誰が何を行うのか，その際どのような資源を使うのかなどをモデルとして作成する工程である．ビジネス中心のアプリケーション開発方法のスタート点となる．

3）　ドメインのモデリング：

ビジネス・モデルは情報システム構築を担当する開発担当者が解決すべき課題や懸案の全範囲を示したものであり，アプリケーションソフトウェアとして実現するビジネス・モデルの特定部分を決定することが対象ドメインの決定となる．

この工程では，対象とした部分の詳細なモデルを作成することを行う．

4) コード編集/ビルド：

従来，アプリケーションソフトウェアの開発実作業部分とみなされていた工程であり，プログラム・ソースコードの作成/編集/管理，データベーススキーマおよび関連文書の管理を行う工程である．

5) アプリケーションのアセンブル：

実行可能なアプリケーション形態とするために，前の工程までに開発されたものの中から必要な情報を集め展開可能な構造にする工程である．

6) アプリケーションの配布：

この工程は IntegratePlus の環境下で管理されているアプリケーション構成要素を，実行環境に適合した形に編成して配布するための作業を行う工程となる．

2.4 異種ツール連携機能

アプリケーション開発ツールは最近著しく進展しており，特にビジュアル開発ツールの分野での進展が目立っている．これらのツールは，アプリケーション開発作業の大いなる手助けとなっているばかりでなく，ユーザが抱えている課題そのものを解決することを容易にする効果をもたらしている．しかし，ユニシスがこれらのツールの使われ方を調査した結果，次のような状況が明らかになった．「アプリケーション開発作業は先に記したように複数の工程からなる．これらの開発ツールは，ある特定の工程内の作業をするものとして使用されており，工程間をまたがって使用されているケースは少ない．のみならず工程間のつなぎはシームレスではなく，何らかの継ぎ目があり，その間を自動的に繋いでいるものはない．」

IntegratePlus は，この点を解決する機能の提供を 4 番目の狙いとしている．

まず，開発者は数多く存在するツールの中から，開発の各工程で使用する開発ツールを選定しなければならない．ある一つの工程で使用するツールとして，あるツールを選択する．他の工程にも同様にあるツールを選定する．こうして選んだツールが同一ベンダー製品で統一されていることは，ほとんど無いであろう．これは，ツールベンダーは自社が最も得意とする分野で一番となりうるツールを市場に提供する戦略のもとで製品提供をしてきているため，工程ごとに最良のツールを選択すれば当然このような結果になるのである．

異なるベンダ製品を用いて開発作業を進めるためには，この工程ごとに選択されたツールの情報を，前工程より後工程へまたその逆の流れで，連携させなければならない．このために開発の全工程をつなぐリポジトリの上にツールをのせ，ツールの情報をリポジトリに格納する形態を採用することが有効となる．IntegratePlus はユニシス社の誇るリポジトリ製品（製品名：UREP）をその基本構成要素として採用し，ツール間を結ぶ種々の情報を管理するようにしている．

3. IntegratePlus の構造

IntegratePlus は，WindowsNT システムでビジネスアプリケーションを開発するためのシステムであることを目標としており，加えて開発したアプリケーションの再利用・改修を行えることをもう一つの目標としている．

IntegratePlus は，開発プロセスを，抽象度の高いレベルから徐々に抽象度を下げ，最終的には特定の実行環境に準拠する形式に変換する過程と捉えている．更に，開発したアプリケーションの再利用・改修を行うにはこの開発プロセスの中で，一般形式より特定形式にしたものを，元の一般形式に戻す仕組みを持つことが必須であるとの考えに立っている．この「新しい機能を提供するために既存のアプリケーションを進化させること」という再利用実現の考えは，次のような考えから導き出される．

現在，実行可能なアプリケーションとして存在するものは，ある環境下での特定形式にされたものであり，これをより一般的な形式にすることが必要である．この特定形式から一般形式にすることを汎化と呼び，いわゆるリバースエンジニアリングの処理をすることである．これに対し，一般形式から特定形式への変換を特化と呼び，このプロセスをフォワードエンジニアリングとする．この関連は図 3 に示すとおりである．

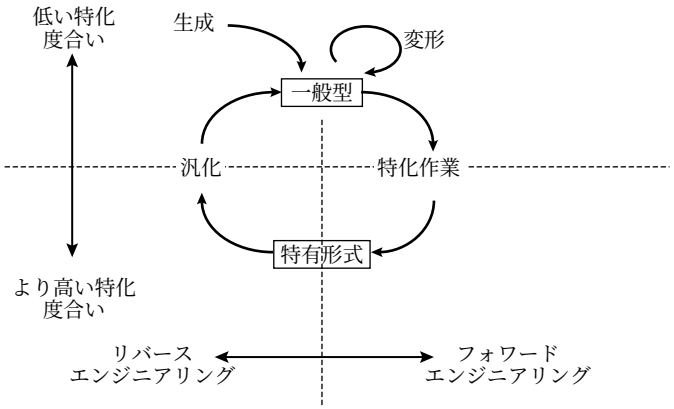

図 3　進展プロセス

IntegratePlus はこの構造を持つものとしてデザインされている．先に記したように IntegratePlus は 6 種の開発工程を持っているが，既存資産連携部分を除いた五つの工程には，この汎化⇔特化の相互変換の構造を組み込んでいる．この関連は図 4 に示すとおりである．

既存のアプリケーションを新たなビジネスの変化に対応させる為に，アプリケーションの改造が行われることになるが，その改造のためには図 4 の階層を必要なところから戻ってゆくことを行わなければならない．例えば，現在のアプリケーションの実行環境を変更し別の実行環境でそのアプリケーションを稼働させようと意図する場合は，ビジネス資産段階よりアプリケーションソース段階に 1 段階戻ってから新たな実行環境への展開操作を行うことになる．現在稼働しているアプリケーションのビジネスモデルを変更することを意図するのであれば，ビジネス資産段階よりビジネスモデル段階まで戻って再展開操作をすることになる．図 5 はこの改造の段階の概要を表したものである．

3.1 基 本 構 造

IntegratePlus は，ここまで述べた再利用を実現するプロセスおよび前章で述べた

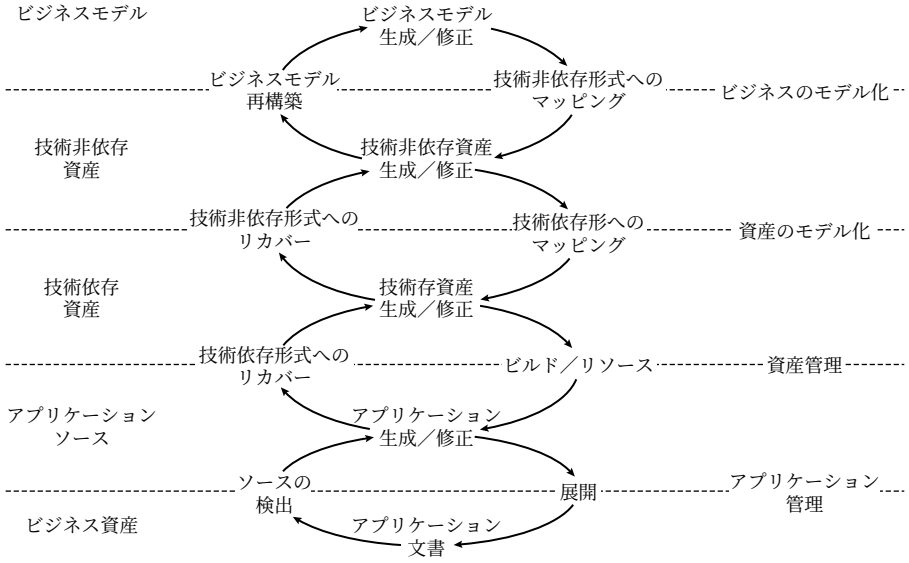

図 4　開発プロセス段階

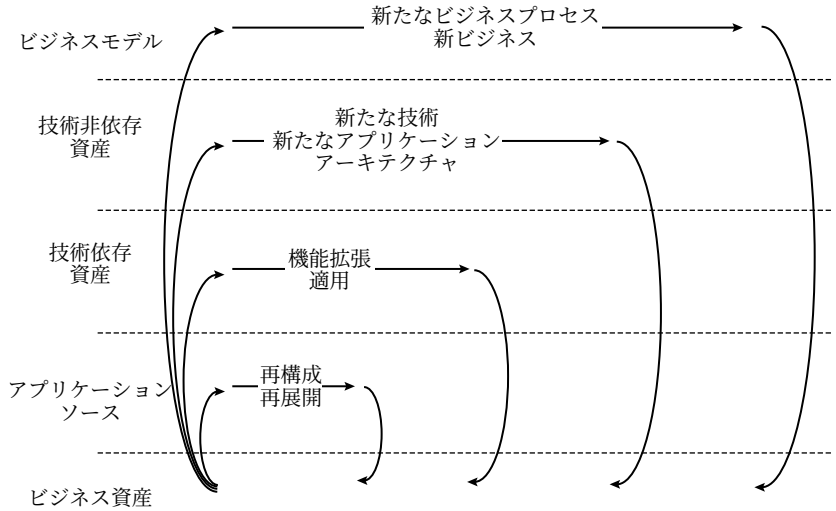

図 5　改造の段階度合い

4 種の機能を実際に実現するために図 6 に示すような構造となっている.

① IntegratePlus 利用者が統一的なインタフェースで種々の機能を呼び出せ, 開発工程を把握できるようにする核となる部分（ワークルーム部分）

② 各種開発ツールを連動させる部分（アダプタ部分）

③ 生成されたコード情報を管理する部分（バージョン管理システム部分）

④ IntegratePlus を使用する際に必要となるユーティリティ制御部分（ユーティリティ部分）

⑤ すべての情報を管理するためのリポジトリ部分

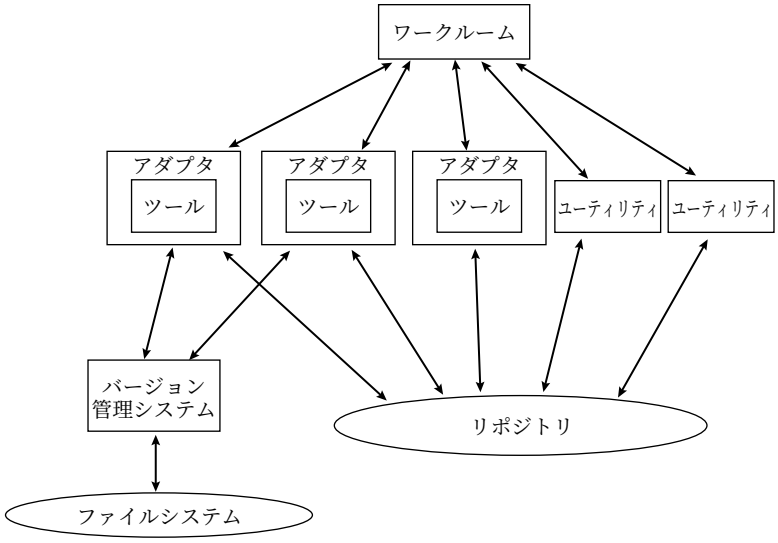

図 6　IntegratePlus の基本構造

図 6 に示されている各種のツールを囲むアダプタについて簡単に説明する．アダプタは，ツールの呼びだしオペレーションと出力結果保存方法の統一化を実現するためのものとなる．生成されたコード情報の管理はバージョン管理システムとファイルシステムを用いて実現され，ソースコード自体はリポジトリの外部に保持されるが，リポジトリ内部の情報とのリンクがつけられている（3.3.2 項参照）．

IntegratePlus 構造の主要部分について，次の項目を紹介する．

① 異種ツール連携の実現形態

② リポジトリ使用による情報連携機能（ここでは，汎化⇔特化についても簡単にふれる）

③ 大規模システムをチームで開発する作業の支援機能

3.2 IntegratePlus における異種ツール連動

IntegratePlus の最も特徴的な機能の一つは，異なるベンダから提供されている開発ツールを IntegratePlus 本体と連携し，かつツール間情報を共有できるようにしていることである．

3.2.1 ツールアダプタによるツール連携

ツールアダプタによるツール連動は図 7 のような構造をとっている．

どのような開発ツールでも IntegratePlus 経由で使用することを容易にするため，IntegratePlus 上で使用するツール自体には何の改造も加えないことが重要である．したがって，ツール自体が処理する入出力ファイルはそのままにし，IntegratePlus が成果物を管理するリポジトリとは二重構造の形態を採用している．

IntegratePlus の使用者がある開発ツールの使用を指示をすると，次のような手順で処理が進められることになる．

① ポジトリより関連する情報をそのツールが使用する入力ファイルに取り出す（前処理）

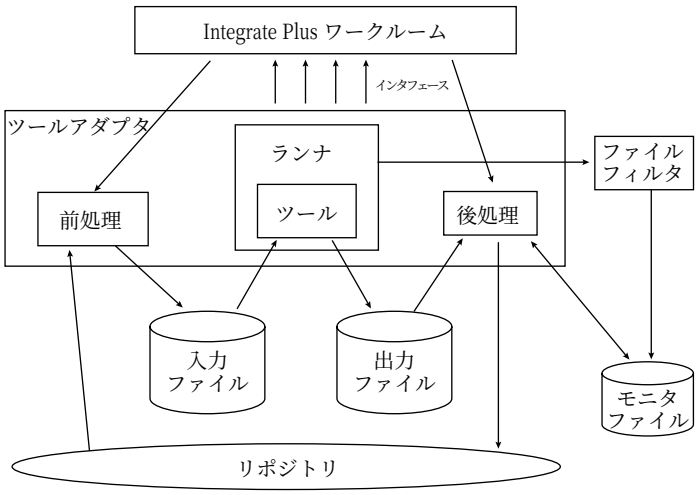

図 7　ツールアダプタ構造

② そのツールを起動する

③ ツール自体での処理を行う．ツール自身が処理する情報は入力ファイルより取り出され，出力ファイルに格納される

④ ツールの処理が完了すると，ツールが出力したファイルより関連情報を取り出しリポジトリ内の情報を更新する（後処理）

この後処理の際に，使用するのがファイルフィルタである．ファイルフィルタはツールが稼働中に行った I/O 操作に関するデータとファイル情報をモニタファイルに書き出しておく機能を実現している．後処理において，このモニタファイルより情報を読み出し，ツールが生成・更新したすべてのファイルについて，リポジトリ内の対応ファイルを更新し，新規情報の追加を行う．この結果，ツールが行った結果は IntegratePlus が管理するリポジトリ中にすべて反映されることになる．

この二重構造は，ファイル処理における効率性に問題を残すかも知れないが，ツール自体の処理を変えずにすむメリットが非常に大きい．ツールがどのような情報をどのような形態で保持しているのかが把握できれば，どのようなツールでも IntegratePlus で使用するツールとして組み込み可能となり，開発環境としての柔軟性が高まるからである．単体ツールの組み込みはこのようにして行う．

しかし，ツール間の情報連携という観点からみると，この方法には大きな難点が存在している．つまり，前処理においてリポジトリより入力ファイルに取り出す情報は，通常は前工程で生成した異種開発ツールの出力物である．ところが，前工程で生成した情報の形態のままでは，後工程のツールが使用できるような情報形式にはなっていない．この為ツールアダプタを開発する際に，前工程でのツールの出力内容を確認し，後工程のツールが受け取れる形式に変換する処理を開発して対応せざるを得ない．この情報交換処理部分を開発するのは，ツール毎に対応することが必要なため IntegratePlus に新たなツールを組み込む際の負担になっている．この負担を解決する方法となるのが，次の XML/XMI 利用によるツール連携機能の実装である．現行の In-

tegratePlus では，Rational　Rose と ERwin の組み込みに XML[*2]/XMI[*3] 連係機能を適用している．

3.2.2　XMI 利用による連携

XML は，データ交換時に共通に使用する表現形式として，その利用が急速に広がってきているものである．それぞれのツールが，XML で情報を出力するようになれば，ツール間での情報交換は極めて容易になると考えられる．また，コンポーネントを表現するには UML を用いることが有効である．この UML モデルをツール間で受け渡しができるようになれば，開発ツール間での連携が可能となる．このために IntegratePlus が行ったのが XMI の実装である．

XMI はモデル情報の交換を行うために OMG[*4] が制定した規格であり，MOF[*5] メタモデルを XML の DTD[*6] にマッピングすること，MOF メタデータを XML 文書にマッピングすることを行うものである．ツール間での情報交換のために，互いの情報を XML データとして引き渡すこととしても，その XML データ中のタグの意味がどのような内容のものであるかが判別できなければ，処理を行うことが出来ない．このために，メタデータを加えて交換することが必要となる．つまり，UML データの交換を XML 形式を用いて行うことである．

二つのツール間でのモデルデータの交換は図 8 のような形態で行われ，XML 形式で UML データを送受信するためのインタフェース部分が追加開発される事になる．また図 9 は，データ受け取り側ツールのインタフェース部分の機能図であり，XML データをどのように UML に展開し処理するのかを示している． IntegratePlus では，モデル情報を交換可能とする為に，図 9 で示すデータ受け取りツールの位置にリポジトリをおいて，データを格納/取り出す機能を実装している．

3.3　IntegratePlus のリポジトリ連携機能

図 2 のライフサイクル管理で示しているように，IntegratePlus は各開発工程のデータをリポジトリを用いて一元管理している．このリポジトリ使用によるデータ管理により，次の 2 点の特長を実現している．

① 開発者が管理保管場所を意識せずにデータの格納と取り出しが可能（チェックイン，チェックアウト機能）

② 成果物の関連を管理する情報の作成とその世代管理が可能（トレーサビリティ機能）

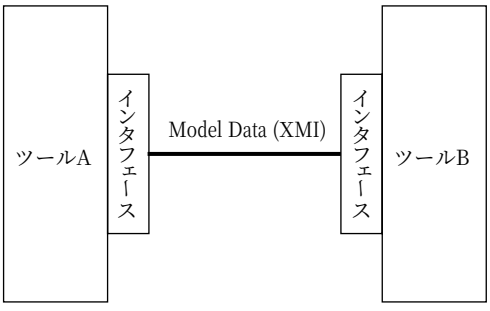

図 8　ツール間データ交換の概要

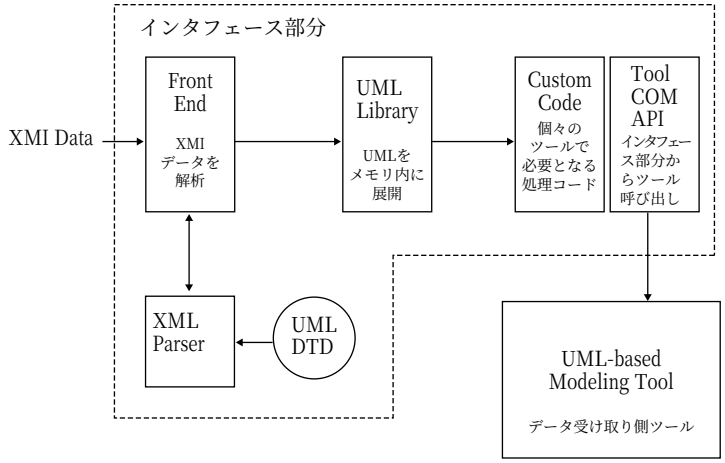

図 9　XMI データ取り込み処理

3.3.1　データのチェックイン，チェックアウト機能

アプリケーション（またはコンポーネント）の開発に際して，開発者はまずモデル（コンポーネント，サブシステム，パッケージ）として開発成果物を管理する構成を定める．関連情報は，このモデルに連携付けられて自動的に保管管理されることになるため，開発者は，自らが開発作業を開始する際にどのモデルを対象に作業するのかを IntegratePlus に対して指示し，関連情報を取り出す指示（これをチェックアウトと呼ぶ）を行い，開発作業の中断・終了時に取り出した情報の格納指示（これをチェックインと呼ぶ）を IntegratePlus に対して行うだけでよい．

更に，ある開発者がモデルをチェックアウトしている場合には，他の開発者が同じモデルの更新作業を行うことはできないよう，読み出しモードでしかチェックアウトを可能としていない．ある開発者が当該モデルを他の開発者が使用中か否かを判断するのは表示画面上にて，できるようになっている．

開発者が作業結果の格納場所を自身で管理しながら作業をすることは，時として誤りを引き起こすもととなり，大きな作業負担となっていた．IntegratePlus の使用により，この誤りが防止可能となった上，そのための作業負担がほとんど無いことから実際の開発作業時には大きな効果をもたらすものとなる．

3.3.2　トレーサビリティ機能

開発作業を進めてゆくに際して，モデル情報を格納したファイル，ソースファイル，テストドキュメント，要求仕様書など異なる種類のファイルが作成・生成されることになる．これらの異なる種類のファイルを統合的に管理・関連付け，しかもコンポーネントの世代毎にこれらの関係性を保持することで，IntegratePlus 独自のバージョン管理を実現する．バージョン毎の，必要な関連情報をすべて取り出せる機能を持つことで，前バージョンの状態を復元できることができる．これを IntegratePlus ではトレーサビリティと呼んでいる．この機能は図 10 に例示するようなファイル参照モデルをもとに，図 11 に示すようなバージョン管理を実現している．

図 11 を例に説明を加える．例えば，開発の最初の段階で，モデル情報と生成され

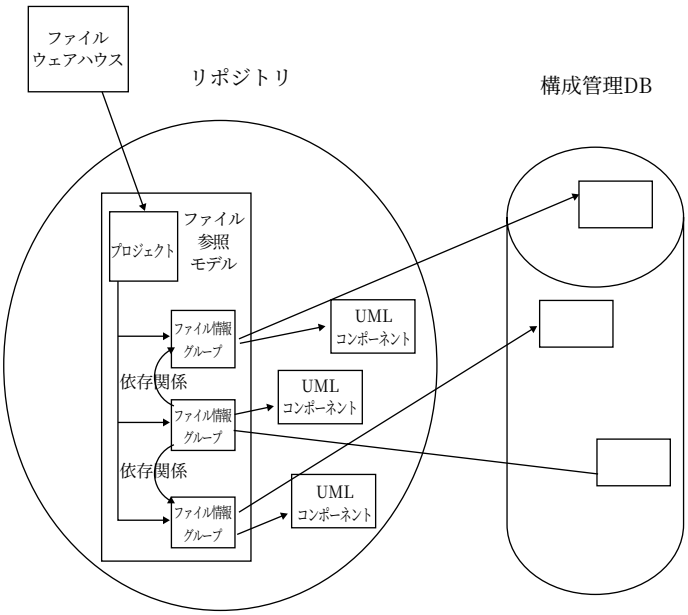

図 10　ファイル参照モデル

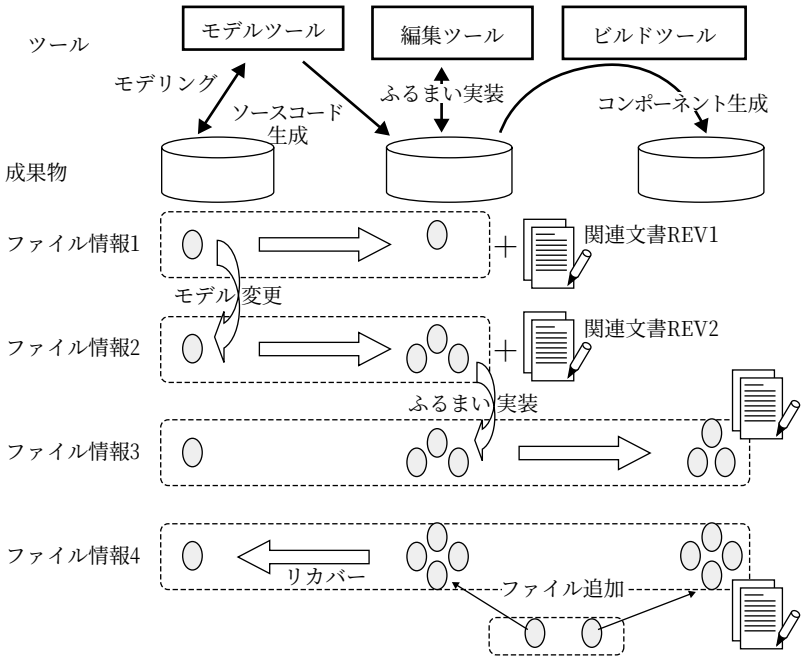

図 11　IntegratePlus のバージョン管理

たスケルトンおよび関連文書（仕様書など）を一組として，ファイル情報 1 という名前で管理する．次の段階ではモデル情報を更新し，更新したモデル情報と再生成したスケルトンおよび更新済文書を一組として，ファイル情報 2 として管理する．ファイ

ル情報 1 とファイル情報 2 の内容は差分情報ではなく，その時点での全情報がそれぞれ格納される．このような関連情報の管理の仕方で，特定時点の開発状況を把握・復元できるようにすることを，IntegratePlus のバージョン管理を利用したトレーサビリティの実現と規定している．

この IntegratePlus が実現するバージョン管理は，従来までのソースコード重視のものではなく，ある時点でのコンポーネントに関するモデル情報，コード情報，仕様書文書などを全て統一的に管理することを目指している．特定レベルごとのモデル情報やデザイン情報がそのまま存在するので，それに対するコードを再生することは容易かつ正確にできることになるのである．仕様書とコードの不一致という問題に悩まされることも無くなる．

また，図 11 のファイル情報 4 のケースで表現していることは，スケルトン（またはコード）部分に加えられた変更が，元のモデル情報にもたらす影響を反映させて管理する機能を提供していることを示している．このように，開発作業の各時点における開発工程ごとの情報を正確に対応づけて管理しているため，前工程の情報に戻ってやり直しをする汎化⇔特化の作業も容易に行えるのである．

3.4 クライアントサーバ構造

IntegratePlus は企業の基幹システム構築を行える開発環境基盤となることをも設計目標の一つとしている．このため複数開発者による同時開発が行えるように，クライアントサーバ構造を採用している（図 12）．サーバ側にリポジトリとサーバ側プログラムを配置し, 開発者ごとのクライアント PC を制御する. この際, Microsoft Transaction　Server（MTS）を使用することでスケーラビリティを確保すると同時に各人の役割を設定する．この役割設定により，その担当者が作業できる工程や開発情報にアクセスできる範囲が規定されることになる．システム管理者は，IntegratePlus が管理する全ての情報にアクセルができる. ビジネスモデリングを担当するモデラーは，当然モデリング情報の作成・更新をすることが許される．しかし，コード開発担当者は，モデル情報の参照はできるが更新は許されない．

また，開発チームの全ての情報は，サーバ側のリポジトリで一元管理される仕組み

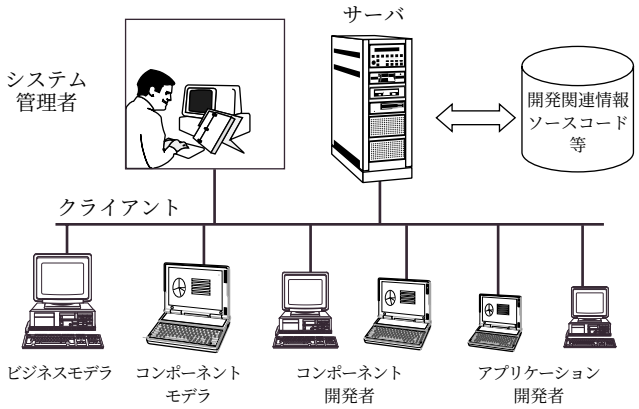

図 12　クライアントサーバ構造

になっているので，情報の散逸や不整合という問題を未然に防止できるようになっている（3.3.1 項参照）．

4. おわりに

統合開発環境基盤構築ツール IntegratePlus の概要について紹介をしてきた．

IntegratePlus は，ビジネスモデルから必要とするアプリケーションを開発するアプローチ，コンポーネントベースの開発方法，開発ライフサイクル管理，異種ツール連携機能および開発資産の再利用という機能の実現をするために開発・提供される製品である．

開発ツールは，各ツールベンダーがそれぞれの特徴を生かして提供されてくるので，必要な局面でそれらのツールを使用してゆけば良いことになる．このような場合に問題となるのがツールが作り出した成果物を，きちんと管理し，次のツールへのインプットとしてゆく基盤である．IntegratePlus はツール間連携・成果物管理を実現するものであり，それぞれの開発者にとって最適な開発環境をもたらすものである．

開発作業を行うプラットフォームと作成したアプリケーションの実行プラットフォームとが分離する方向にあるが，もう一つの利点として，IntegratePlus では PC 環境で開発を行い，アプリケーション実行を希望するプラットフォームへ展開（Deploy）することができるようにしているので，これからの開発環境に IntegratePlus を適用することが最適と言うことができる．

現行版の IntegratePlus は，Windows システム環境でのプログラムとして開発されているが，更に広範な開発環境に対応できるように Java での書き直しも計画中である．Java で開発された IntegratePlus が提供される際には，使用者とのインタフェースとなるワークルーム部分は，現行版とは異なることになるが，適用できるプラットフォーム範囲が広がり，より利用しやすくなるものと確信している．

＊1　UML（Unified Modeling Language）OMG 標準と規定されているモデリング言語
＊2　XML（eXtensible Markup Language）W3C（The World Wide Web Consortium：WWW に関する規格制定団体）が規定したインターネット時代のデータ記述言語の規定
＊3　XMI（XML Metadata Interchange）OMG 標準として制定されたもので，モデリングツールとリポジトリ間でメタデータを交換するために，XML を用いた標準的な形式を規定したもの．
＊4　OMG（Object Management Group）分散環境でのソフトウェアの相互運用性確保を目的に標準仕様を検討している非営利のコンソーシアム．CORBA などの規定をしている．
＊5　MOF（Meta Object Facility）OMG が情報交換に際して，どのようなメタデータの階層を持つべきかを規定したモデル．
＊6　DTD（Document Type Definition）XML 文書の中で使用されるタグ名やタグの階層構造を規定するもの．

執筆者紹介　松　倉　　司（Tsukasa Matsukura）
1969年日本ユニシス(株)入社．証券会社の経営情報システム（MIS）構築サービス言語トランスレータ開発，日本語処理システムのデザインおよび開発を担当，HMP/IX言語プロダクト関連のサポートを経て，現在，ESビジネス推進部プログラム開発室所属．