BUSINESS PROCESS TRENDS NEWSLETTER

Volume 2, No. 5 May 2004

The OMG's Model Driven Architecture and BPM

Executive Summary:

The Object Management Group, an organization usually associated with software standards has recently begun to promote a new software development methodology, the Model Driven Architecture. In conjunction with MDA, the OMG has started to move aggressively to create standards for business process modeling and business rules. These new standards will likely impact most business process automation throughout the remainder of this decade. This issue of BPTrends Newsletter provides an overview of what the OMG is doing and the motivation behind it.

Contents:

Paul Harmon
Executive Editor
pharmon@bptrends.com

www.bptrends.com

The OMG

The Object Management Group was established in 1989 by a consortium of companies that anticipated growth in the use of object technology and wanted to establish open, object standards to facilitate the growth of the market.

Throughout the Nineties, the member companies that make up the OMG labored to create a set of open standards, collectively known as the Object Management Architecture (OMA). The centerpiece of the OMA was CORBA (Common Object Request Broker Architecture) a middleware standard which defined how messages from diverse languages could be defined via a common intermediary language, the OMG's Interface Definition Language (IDL), moved from client to target platforms via platform-independent protocols (e.g. IIOP), and then retranslated into the language of the target object or component.

In another effort, the OMG formalized the Unified Modeling Language (UML), a modeling system that could be used to represent software designs. Initially, UML was to be a merger of several object-oriented notations. As it evolved, however, it became a more generic software development notation and incorporated diagrams that were not, strictly speaking, object-oriented.

For the past several years, the OMG has been moving beyond its roots in object standards. Its members, some 700 companies from throughout the world, are concerned with integrating and organizing all their software assets. Once they accepted that multiple middleware standards had established themselves in the market and that CORBA would always need to be combined with other middleware standards to create a complete solution at any large company, they began to think about more generic solutions to the integration problems they all faced.

MDA

The Model Driven Architecture (MDA) represents a major effort to create the standards necessary to facilitate a comprehensive new approach to the creation, integration, and maintenance of software assets. The goal of MDA is to create an enterprise architecture modeling capability that analysts and developers can use to describe a company's business and software assets. By creating the architecture with software tools, companies are in a position to generate specific applications to implement the architecture and to modify those applications as the organization's needs change. In other words, MDA represents a major step in the direction of a real-time enterprise in which managers can make changes in architectures that are subsequently represented in code.

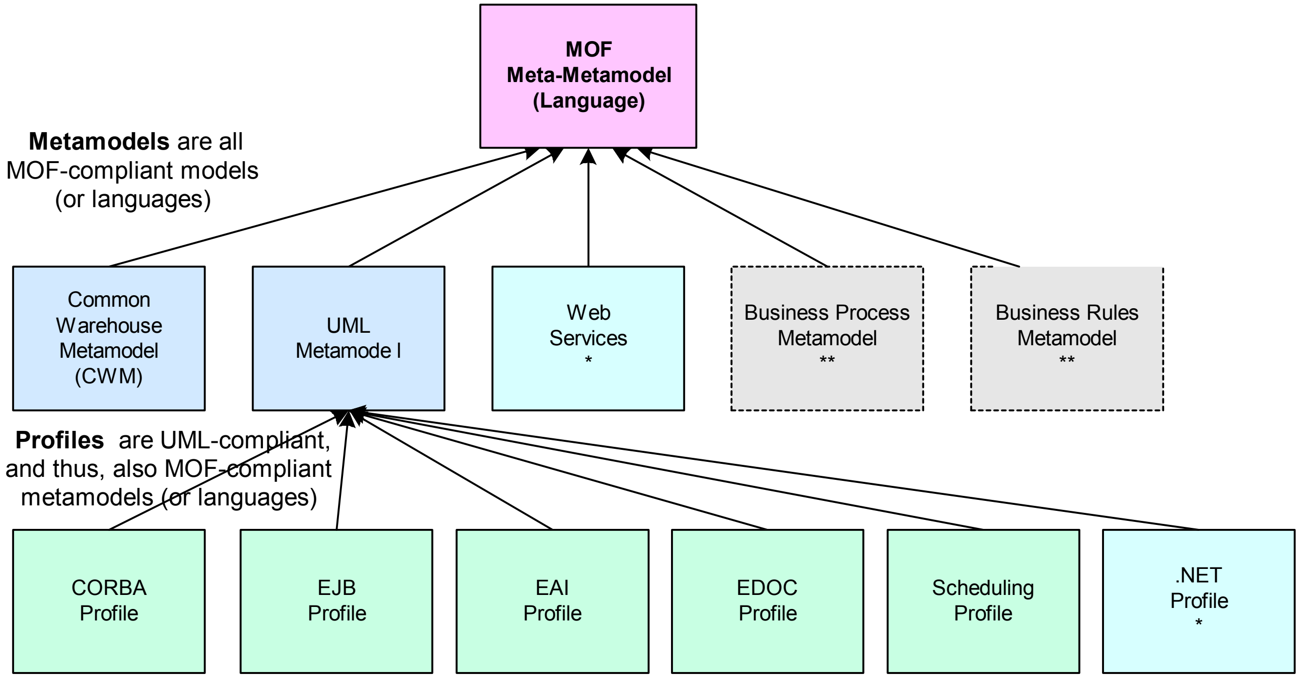

Figure 1. An overview of some of the elements in the Model Driven Architecture.

MDA is concerned with models and talks about them in two different ways:

- **Model Standardization.** First MDA is concerned with techniques that assure that all models used in software development can be aligned with each other. This focus emphasizes the use of MOF and metamodels.

- **Models for Software Development.** Second, MDA is concerned with organizing models used in the software development process so that developers can move from abstract models to more concrete models. This focus emphasizes the use of Computation Independent Models (CIM), Platform Independent Models (PIMs), Platform Specific Models (PSM) and mappings that allow a developer to transform one model into another.

We'll consider, first, how MOF provides common modeling standards, and then how models can be organized to facilitate efficient and flexible software development.

Model Standardization

The Model Driven Architecture is supported by a number of models and standards. All MDA models are related because they are all based on a very abstract metamodel – the Meta Object Facility or MOF. (MOF is sometimes called a metametamodel.) Every other model used in MDA is defined in terms of MOF constructs. (See Figure 1.)
In other words, every MDA model is mapped to MOF. That guarantees that all models used in the MDA system can communicate with every other MOF-compliant model. All of the different data representations in the OMG's Common Warehouse Model (CWM) are MOF-compliant. Similarly, all of the diagrams supported by the Unified Modeling Language (UML) are MOF-compliant. Thus, if I want to move from a Common Warehouse Model to a UML model, I move from the CWM model, to a MOF representation. Then I move from the MOF representation to the UML model.

UML supports extensions which are termed *profiles*. Profiles are MOF-compliant as a result of being extensions of UML. Profiles, in UML 2.0, are used to describe various functional uses of UML. Profiles are extensions to UML and are, themselves, MOF metamodels.

Some MDA metamodels have been formally defined by the OMG. Some are still being developed by OMG task forces. In some cases outside groups or vendors have developed MOF-compliant metamodels. These non-OMG metamodels are MOF-compliant and can be used in MDA development, but they are not, as yet, official OMG standards.

The MDA models and profiles shown in Figure 1 are defined as follows, starting at the left of the second row:

The Common Warehouse Metamodel (CWM)

The OMG's formal model of metadata is used to manage data warehouses. Using CWM, developers can generate a number of more specific data models or formats, including relational tables, records or structures, OLAP, XML, multidimensional database designs, and so forth. This includes aspects which also have value outside the data warehousing environment such as data models, transformations, software deployment, and business nomenclature.

The UML Metamodel

Early versions of UML were not completely MOF compliant, but the latest release of UML, version 2.0 is MOF-compliant. UML defines a set of core modeling concepts which can be combined into various diagrams, including, for example: Class Diagrams, Sequence Diagrams, State Diagrams, Activity Diagrams, Component Diagrams, and Package Diagrams. In addition, the UML specification includes a facility, C , that allows developers to establish constraints on various UML elements.

Web Services

Web Services is an example of a non-OMG metamodel developed to facilitate the development of MOF-compliant Web Service models.

The Business Process Definition Metamodel

This is an example of a metamodel that is still in the development phase. The OMG has called for proposals for a MOF-compliant metamodel for business processes. Such a metamodel would be independent of specific process definition languages and would allow MOF models to interface with languages like WSBPEL and notations like BPMN.

Business Semantics for Business Rules

Another example of a metamodel in development is an RFP for a MOF Metamodel for capturing business rules in business terms, and the definition and semantics of those terms in business vocabularies. In fact there will be two specifications, a more generic standard for business rules, and a more specific one for production rules that are actually used by rule engines.

CORBA Profile

This metamodel defines how to use UML to create CORBA-specific models. The CORBA specification includes the definition of a CORBA component model that can be modeled in UML and used in application development.

EJB Profile

This metamodel defines how to use UML to create J2EE or EJB-specific models. Developed by the Java Community Process.

EAI Profile

(The UML Profile and Interchange Model for Enterprise Application Integration.) This metamodel defines how to use UML to model event-driven EAI solutions.

EDOC Profile

(The UML Profile for Enterprise Distributed Object Computing.) This metamodel defines how to use UML to model distributed enterprise systems and the aspects of the business that they support (business processes, entities, events, etc.). The EDOC standard includes a Java metamodel that defines how to create Java-specific models.

Scheduling Profile

(The UML Profile for Scheduling, Performance and Time.) This metamodel defines how to use UML to model temporal aspects of (primarily real-time) computer systems.

.NET Profile

Another example of a profile created by developers independent of the OMG. A .NET profile defines how to use UML to create .NET-specific models.

The latest version of UML includes the ability to specify the behavior of models so that they can be converted directly to code. At the moment there is no standard way to transform any MOF model to any other, but an OMG committee is working on standardizing transformations.

There are other MOF compliant metamodels and UML profiles in existence and in development. MDA is a work in progress, and the OMG will continue to develop new MOF metamodels as new technologies, languages or modeling elements need to be defined so that they can be integrated with MDA. For example, IBM has recently developed a profile for BPEL. Once BPEL is completely defined by the OASIS committee, the OMG will probably create an official, open BPEL metamodel.

XMI

The OMG's XMI (XML Metadata Interchange) standard assures that any MOF-compliant model can be represented

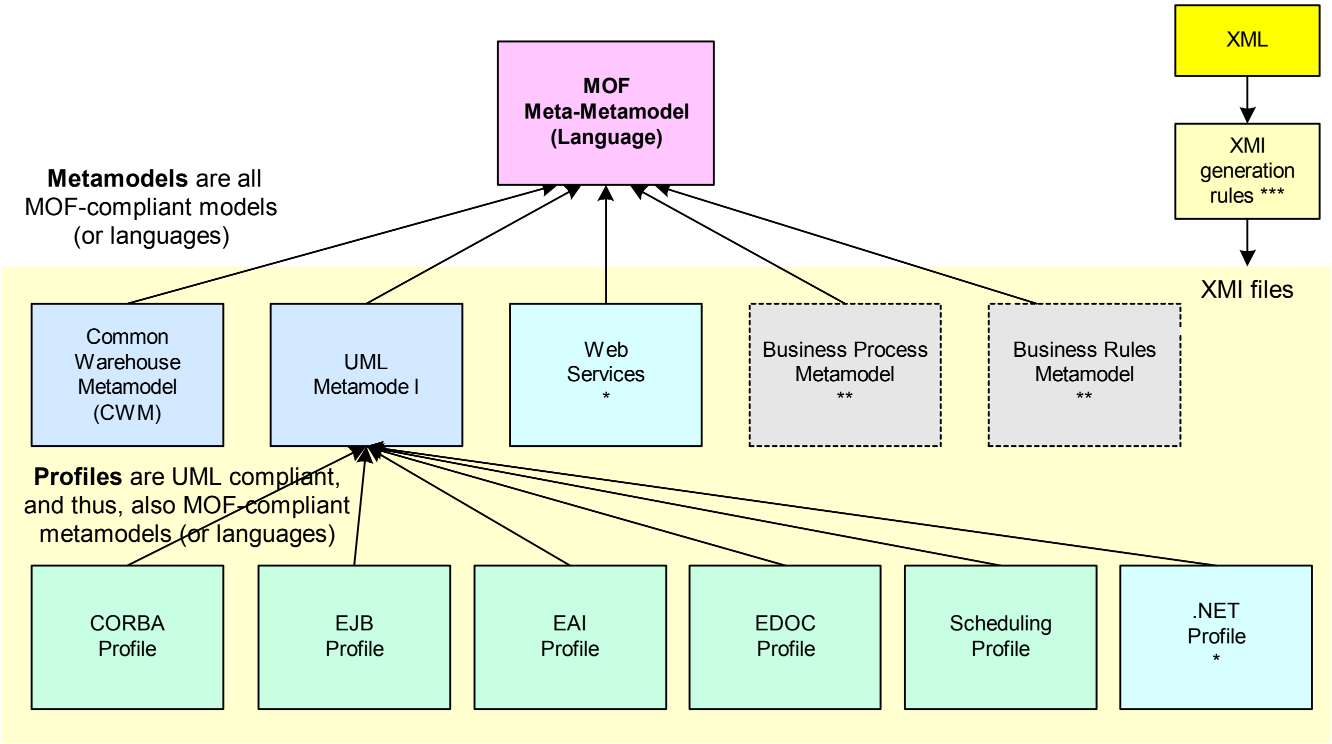

Figure 2. Any MDA metamodel can be transformed into an XML document for transport over the Internet.

as an XML document and stored in a MOF-compliant database. Thus, in effect, an XMI document is a MOF XML document. The XMI facility is built into MOF, so once one creates a new MOF metamodel one automatically gains the ability to represent the instances of the new metamodel is XMI documents.

Figure 2 uses a field of yellow to suggest that we can represent any MDA metamodel as an XMI document.

The OMG's Business Process Metamodels

MDA would be of interest to those involved in business process work, if only because it is emerging as a major approach to software development. Increasingly, processes are automated by software components and data is stored in associated databases. As processes change, software components and databases must change. Thus, any system that facilitates rapid and traceable changes in software supports business process efforts. In fact, MDA promises much more.

The highest level development model in the MDA system is the Computation Independent Model (CIM). A software developer might well imagine that a Use Case diagram was a CIM model, and it would function in that manner in some cases. But Use Case diagrams have never been popular with business managers. Instead, business managers have tended to prefer organization diagrams and process diagrams, which they usually refer to as *workflow diagrams*. In UML terms, an activity diagram is a workflow diagram. Similarly, the process diagrams that once can create in tools like Popkin's System Architect, Proforma's ProVision or Casewise's Casewise product are all examples of CIM diagrams. In other words, MDA provides a conceptual structure that stretches from the diagrams used by business managers and analysts to the various diagrams used by software developers and offers to organize them so that requirements specified in one diagram can be traced through other, more detailed diagrams derived from them. Thus, MDA is a major tool available for everyone that is interested in aligning business processes with IT systems.

In fact, however, the OMG is going much further. The OMG's Business Enterprise Integration Domain Task Force (BEIDTF) has begun to work on a number of metamodels that will facilitate the integration of the whole range of business process models.

The Business Process Definition Metamodel

The centerpiece of the BEIDTF's current work is the Business Process Definition Metamodel. The Business Process Definition Metamodel is a semantic description of the logical relationships among the various elements of any possible business process description. It is not a notation. It simply describes logical relationships.

As with all OMG standards efforts, the task force initially worked together to describe the requirements for the model. They then asked for proposals. Several companies submitted proposed what should be included in the Business Process Definition metamodel. Currently, the various submitters are working together to refine a common proposal that the committee could adopt and recommend to the OMG's Architecture Board, and, ultimately, to the OMG membership for approval. This process will probably take the reminder of 2004.

Once the OMG's Business Process Metamodel is complete, any vendor will be able to map its specific metamodel to the OMG's metamodel, and then use XMI to transport information about models developed in that tool to any other tool that is also mapped to the OMG metamodel. In a similar way, external standards, like the BPMI's Business Process Management Notation and UML 2.0 activity diagrams will be mapped to the OMG's Business Process metamodel. Via MDA, companies will be in a position to move models created in specific business process tools to the Business Process Definition Metamodel and then to software languages like J2EE, BPEL, or to other BPM tools. (See Figure 3.)

The Business Rules Metamodels

At the same time that the BEIDTF is working on a Business Process Definition Metamodel, it is also working on several other metamodels. For example, a subcommittee is working on a Business Rules Metamodel, and on a more detailed Production Rule Metamodel (for systems that use an inference engine to process the rules). These efforts are in earlier stages of development and probably won't be completed till 2005. When they are in place, however, Business Rule tools will be able to exchange information by mapping to the Business Rules Metamodel, and Business Rule tools will be able to integrate with Business Process tools by moving from one metamodel to another, since their will automatically be on OMG mapping between the Business Rules Metamodel and the Business Process Definition Metamodel.

The Business Organization Metamodel

The BEIDTF has recently begun to consider the creation of some kind of more general metamodel to describe business organization diagrams. At this point they are still collecting ideas. Assuming they go forward then they will try to create a metamodel that describes the logical concepts and relationships that organizations maintain, including, how processes relate to other organizational entities.

The Business Ontology Metamodel

A separate OMG task force has been working on a Ontology Metamodel, a model that describes vocabulary relationships. Many companies find that just getting a process team to agree on the terms they use to describe different subprocesses and activities, or entities and outputs, can consume an inordinate amount of time. An ontology metamodel is an effort to formalize some of the work that has been done in this area. As the Ontology Metamodel goes forward, it will undoubtedly need to be systematically

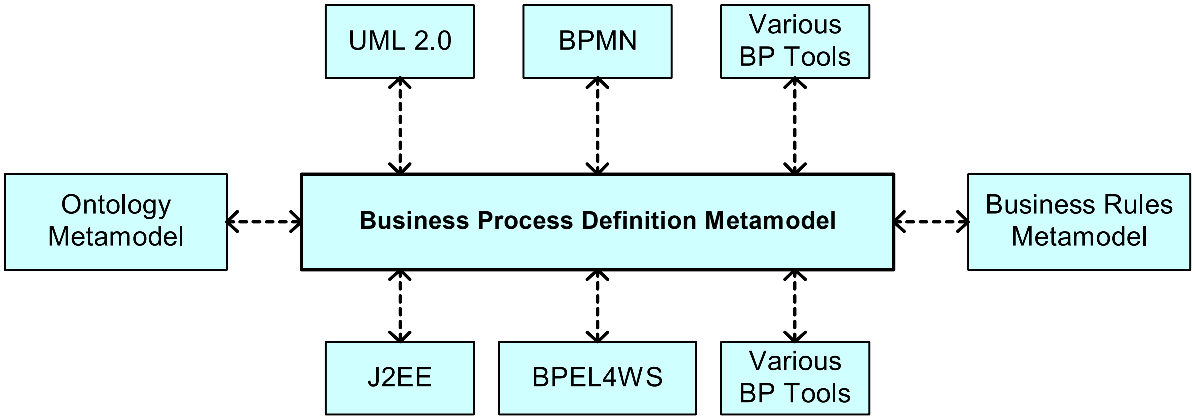

Figure 3. The Use of the OMG's Business Process Definition Metamodel.

linked with the Business Process Definition Metamodel and with any Organization Metamodel that BEIDTF develops. The OMG BEIDTF team is also beginning to work with other OMG task forces to assure that domain or industry specific languages will map to the metamodels they are developing. Thus, for example, the BEIDTF has begun to meet with the Telecom Task Force to see what can be done to assure that the eTOM/NGOSS business process framework developed by the TeleManagement Forum is aligned with the Business Process Definition Metamodel.

Consider one powerful example of what MDA will do for software developers. At a recent show we had a talk with Mark Riggle, the president of CAUSal Aspects, a IT consultancy that helps data warehouse vendors. Riggle mentioned that several data warehouse vendors were interested in BPM. Increasingly the clients of the Data Warehouse vendors will want the data from the warehouse tools displayed in terms of business processes. Of course the data warehouse tools were never designed to do this. Riggle explained that he was following the work of the OMG's BEIDTF very closely. Several Data Warehouse vendors already map to the OMG's CWM (Common Warehouse Metamodel) so that they can exchange data between warehouse products and store it in standard data repositories. “As soon as the OMG publishes the Business Process Definition Metamodel and BPM vendors map to it, we will be able to link to their products via XMI,” Riggle explained. Obviously any specific Data Warehouse vendor could develop a specific mapping between their tool and a specific BPM tool. But a specific mapping is expensive. Which BPM tool would you chose? What Riggle wants is a generic solution. A Data Warehouse vendors maps once, to CWM. Each BPM vendor maps once, to the OMG's Business Process Definition Metamodel. Then, suddenly, the Data Warehouse tool can pass data to any of the BPM tools that support the Business Process metamodel. This is a nice example of what MDA is so exciting and why its so important to those involved in business process change.

An Aside on UML and UML 2.0

UML 1.1 was standardized and released by the OMG in 1997. It was originally intended to be a synthesis and standardization of the best features of several competing object-oriented notation – a universal object-oriented notation. As it progressed through committees, developers fro industries with specialized interests suggested non-object extensions. There was debate, and, at the last minute, some additional notations were added, including an Activity Diagram which was integrated with a State Diagram. This made sense to some, but has never proved very useful to those involved in business process mapping. Often, at business process meetings, we hear UML dismissed as just an object oriented notation, or as the notation that supports a totally dysfunctional Activity Diagram.

In June of 2003, the OMG began a major revision of UML, referred to as UML 2.0 which will be complete this summer. UML 2.0 is a major advance on UML 1.n. Everyone interested in business process notation owes it to themselves to carefully examine UML 2.0 before they make pronouncements on UML in the future.

First, UML was revised so that it is now based on the UML metamodel which is, naturally, tied to MOF. Second, the metamodel defines core concepts much more rigorously than in the past. Third, and of special interest to those involved in business process modeling, the new activity diagram is entirely independent of state diagrams, and capable to supports most of the tasks that business managers seek to analyze.

BPMN was developed by a team of major business modeling vendors working together under the auspices of the BPMI.org. Stephen White has written an excellent paper on BPMN and UML in which he compares how one would model any of the standard workflow sequences using either BPMN or UML.[1] In most cases the differences are trivial. There are a few areas in which BPMN is superior in the sense that it can support process tasks that will occur if one moves to a Web Services architecture (SOA) better than UML. The BPMN team is now moving toward defining BPMN as a profile of UML. Thus, in effect, BPMN will be UML, plus some additional constructs that will be useful in specialized BPM applications. At the same time, members of the BPMI team are working closely with the OMG's BEIDTF to assure that the OMG's Business Process Definition Metamodel supports everything in BPMN, and IBM has shown how a BPMN-UML profile could serve as a notation for the eventual Business Process Definition Metamodel.

MDA and Software Development

MOF assures that all compliant metamodels share a common set of core assumptions and definitions. MDA, however, is primarily focused on organizing the development and maintenance of software resources. Thus, MDA also describes how models are used in the software development process.

Figure 4 suggests how an IT group can derive models from either business process descriptions or software descriptions and use them, in turn, to convert the abstract models into executable implementations. Note that the models used in this process would refer to a specific organization's data and processes. These models would be derived from metamodels like UML, but would refer to specific processes within the organization. Thus, in effect, a specific model of a company's business classes could be classified two ways. It would be a model that complied with the UML metamodel, and if it was platform independent, it would be a PIM model. The former tells what modeling conventions are being used (UML) and the latter tells how the model functions in the development process.

Figure 3 highlights the three different types of MDA models one uses in developing a system, and who is involved in using each. In this case we see that business analysts develop Computation Independent Models (CIM) that describes the business. Architects and designers subsequently create Platform Independent Models (PIM) to illustrate the organization's architecture, without reference to any specific implementation. Later, developers and testers will use tools to generate specific software designs from the PIM architecture and then use their Platform-Specific Models (or designs) to generate code.

CIM Models. CIM models refer to any models used to describe a business. Thus, models that are mapped to the Business Process Definition Metamodel, the Business Rules Metamodel, or even to a Use Case UML Diagram could all be examples of CIM models.

PIM Models. PIM models describe a software architecture in a way that does not assume any specific implementation. Thus UML Class or Sequence Diagrams could be PIM models.

PSM Models. PSM models assure a target platform. Thus, a UML Class or Sequence Diagram that has been extended to incorporate assumptions about Unix, J2EE or .NET would qualify as PSM models.

Language Mappings. Mappings between specific PSM models and code depend on the available to language profiles that specify exactly how a given model is to be transformed.

The various mappings shown in Figure 3, in fact, represent the use of XMI, MOF and metamodel mappings. Thus, a CIM model is mapped to a PIM model by moving via mappings to an appropriate metamodel, using XMI, to MOF to another metamodel and then to the target PIM model.

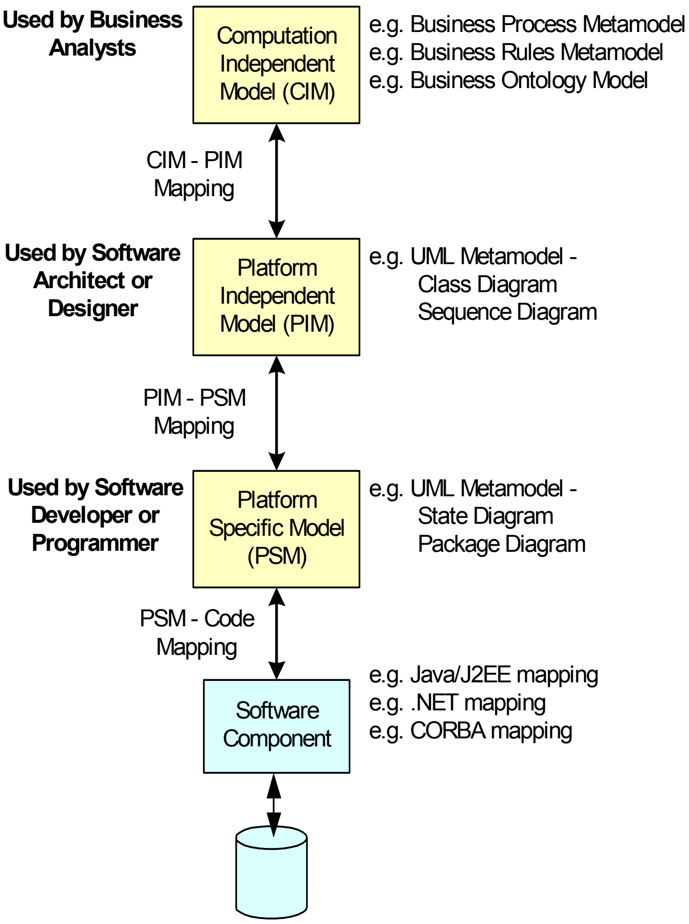
Used by Business Analysts
Computation Independent Model (CIM)
e.g. Business Process Metamodel
e.g. Business Rules Metamodel
e.g. Business Ontology Model
CIM - PIM Mapping
Used by Software Architect or Designer
Platform Independent Model (PIM)
e.g. UML Metamodel - Class Diagram Sequence Diagram
PIM - PSM Mapping
Used by Software Developer or Programmer
Platform Specific Model (PSM)
e.g. UML Metamodel - State Diagram Package Diagram
PSM - Code Mapping
Software Component
e.g. Java/J2EE mapping
e.g. .NET mapping
e.g. CORBA mapping

Figure 3. The MDA Development Sequence

The OMG has issued a number of standards. Task forces are still working on others. Meantime, vendors have been working to implement existing standards. In some cases the vendor have added proprietary extensions to facilitate mappings or code generation that is not yet defined by official OMG specifications.

MDA Tools

A number of companies are already offering software tools to support MDA. In some cases the vendors are UML modeling vendors who are extending their tools from modeling to code generation. In other cases the vendors sell real-time software development tools that already used UML and proprietary code generation capabilities, and in these cases they are retrofitting the tools to conform with MDA's newly released standards. Predictably, the MDA tool market is

Company	MDA Product	Contact
Adaptive Bournemouth, UK www.adaptive.com	Adaptive Framework	Pete Rivett, CTO Pete.rivett@adaptive.com
Codagen Technology Montreal, Quebec, Canada www.codagen.com	Gen-it Architect	Michel Brassard, CTO Mbrassard@codagen.com
EBuilt, Inc./Codigo Solutions Irvine, California, USA www.codigoXpress.com	CodigoXpress	Phillip Lindsay, VP Plindsay@ebuilt.com
IBM Rational Software www.ibm.com	WebSphere / Rational Rose	Stephen A. Brodsky, WebSphere MDA Architect. sbrodsky@us.ibm.com
Interactive Objects Software GmbH Freiburg, Germany www.io-software.com	ArcStyler	Richard Hubert, CEO Richard.Hubert@io-software.com
Kabira Technologies Inc. San Rafael, California, USA www.kabira.com	ObjectSwitch, Kabira Business Accelerator	Grover Righter, VP Tech Strategy Grover.righter@kabira.com
Kennedy Carter Ltd Guildford, Surrey, UK www.kc.com	iUML and iCCG	Ian Wilkie, Tech. Director Ian@kc.com
Project Technology Tucson, Arizona, USA www.projtech.com	BridgePoint and DesignPoint	Stephen J. Mellor, CTO and Chairman Steve@projtech.com
Secant Technologies Cleveland, Ohio, USA www.secant.com	ModelMethods	Keith Rupnik, Product Management sales@secant.com
Softeam Paris, France www.objecteering.com	Objecteering/UML	Joan Le Bris, Marketing and Communication Manager joan.lebris@softeam.fr

Table 1. Some leading MDA Tool Vendors.

going to grow rapidly in the next few years as software development organizations embrace MDA.

Table 1 lists some of the better known MDA tool vendors.

BPM and MDA

Moving beyond what the OMG is doing, we can speculate on how MDA will increasingly merge with Business Process Management. BPM is hot, and a wide variety of vendors are repositioning their tools to function as BPM tools. In the last issue of the BPTrends Newsletter, we described a BPM tool as a product that combined a business process modeling capability with a BPM engine that manages manual activities and invokes software components, as specified by the process model. Figure 5 provides an overview of a generic BPM product.

We emphasized that the major value that BPM products offer is the ability to allow business analysts to modify how a process is executed without requiring reprogramming. Thus, if a business analyst wants to change the process described in Figure 5, they use a process modeling interface and a mouse to rearrange activities. Subsequently, the BPM engine will invoke the manual and software components in a new sequence.

Some of those who have written about BPM have argued that BPM tools will allow business managers, or at least business analysts to actually create code. Consider Figure 5. Activity A is currently done by employees. Assuming we wanted to automate the activity, how would we go about it? According to the more enthusiastic BPM advocates, we would open Activity box A and proceed to diagram the workflow of the employees in such detail that, ultimately, our BPM engine could execute the diagram. In fact, this can be done in some cases. We have seen this illustrated in Intalio's Intalio n|3 product. The developer wasn't a business analyst, however, but a skilled designer who was very familiar with Intalio n|3, and the problem was a simple task that could easily be diagrammed as a sequence. We have illustrated the use of a BPM product for code generation in Figure 6.

There is a sense in which BPML, BPEL and the notational system, BPMN, were developed to support this vision of code generation from a process specification. If you look at a BPMN diagram that describes simple decomposition of what the employee in Activity A is doing, it might consist of only 6-7 boxes. If you proceed to define the detail required to actually generate code from that diagram, however, you move to something like 30 boxes and lots of complex notation to show where decisions are made and to trace the flow of alternative paths. In addition, in many cases,

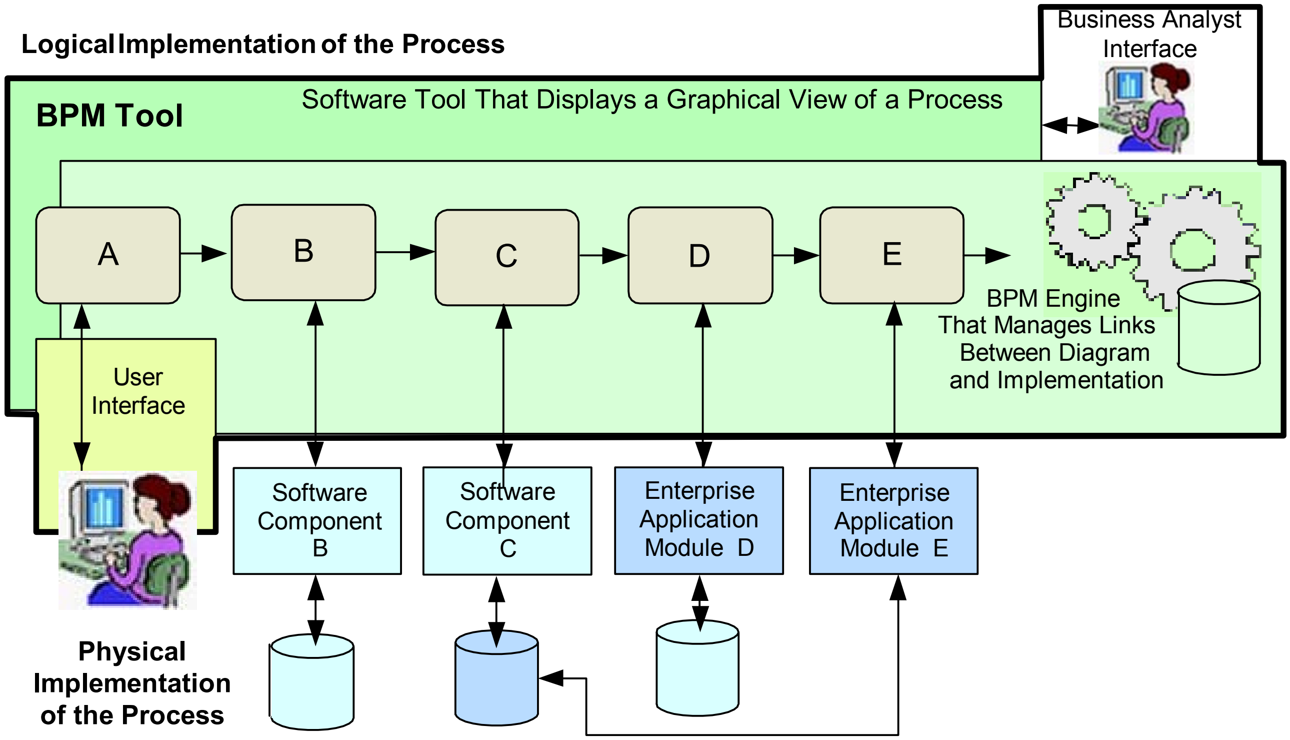
Logical Implementation of the Process
Business Analyst Interface
BPM Tool
Software Tool That Displays a Graphical View of a Process
A
B
C
D
E
BPM Engine That Manages Links Between Diagram and Implementation
User Interface
Software Component B
Software Component C
Enterprise Application Module D
Enterprise Application Module E
Physical Implementation of the Process

Figure 5. A generic BPM product.

someone would have to proceed to define a number of business rules to assist with decisions that must be made during the actual execution of the process. In fact, at this point BPMN is unable to generate BPEL code. Any tool that can do it can only do it because they have added quite a bit of proprietary code into the mix.

In fact, we suggest that is much more likely that most BPM tools will provide a mapping between their BPM models to the OMG's Business Process Definition Metamodel, and then rely on MDA and MDA tools, like IBM's Rational Rose, to actually generate any software code needed. We have illustrated this in Figure 7.

As we noted in the last issue of this newsletter, that is exactly the strategy that Chordiant has followed with their Chordiant 5 BPM suite. Chordiant provides a BPM tool and a set of components to support customer-facing processes. They also support a variety of manual functions. If one needs to automate one of the manual functions, Chordiant generates a description of the manual process which it passes to IBM's Rational Rose for software development. Clearly IBM intends to follow a similar strategy with its Business Process Development Suite and with Rational Rose both included in its WebSphere package.

Conclusion

In the Sixties, when the first business software systems were created, programmers wrote in long sequences of 0s and 1s, the code that computers actually read and execute. In the Seventies, software developers learned to create high-level languages like COBOL and C. In essence, *words* in COBOL or C were translated, by an interpreter, into the long sequences of 0s and 1s that the machine read. Using those languages programmers became much more productive because they could describe the software application using abstractions and the interpreter would handle the tedious process of converting each word into a numerical sequence.

Since the early Eighties, business and IT analysts have dreamed of being able to create diagrams to describe applications and then using those diagrams to automatically generate the needed code. We have succeeded in using this approach to create user interface screens. Today, a business analyst can create a report screen by using a mouse to drag and drop various graphic components into place and then simply clicking on the database items to be displayed in various areas. The actual generation of complete applications from diagrams has proved harder. CASE tools were developed in the late Eighties that were

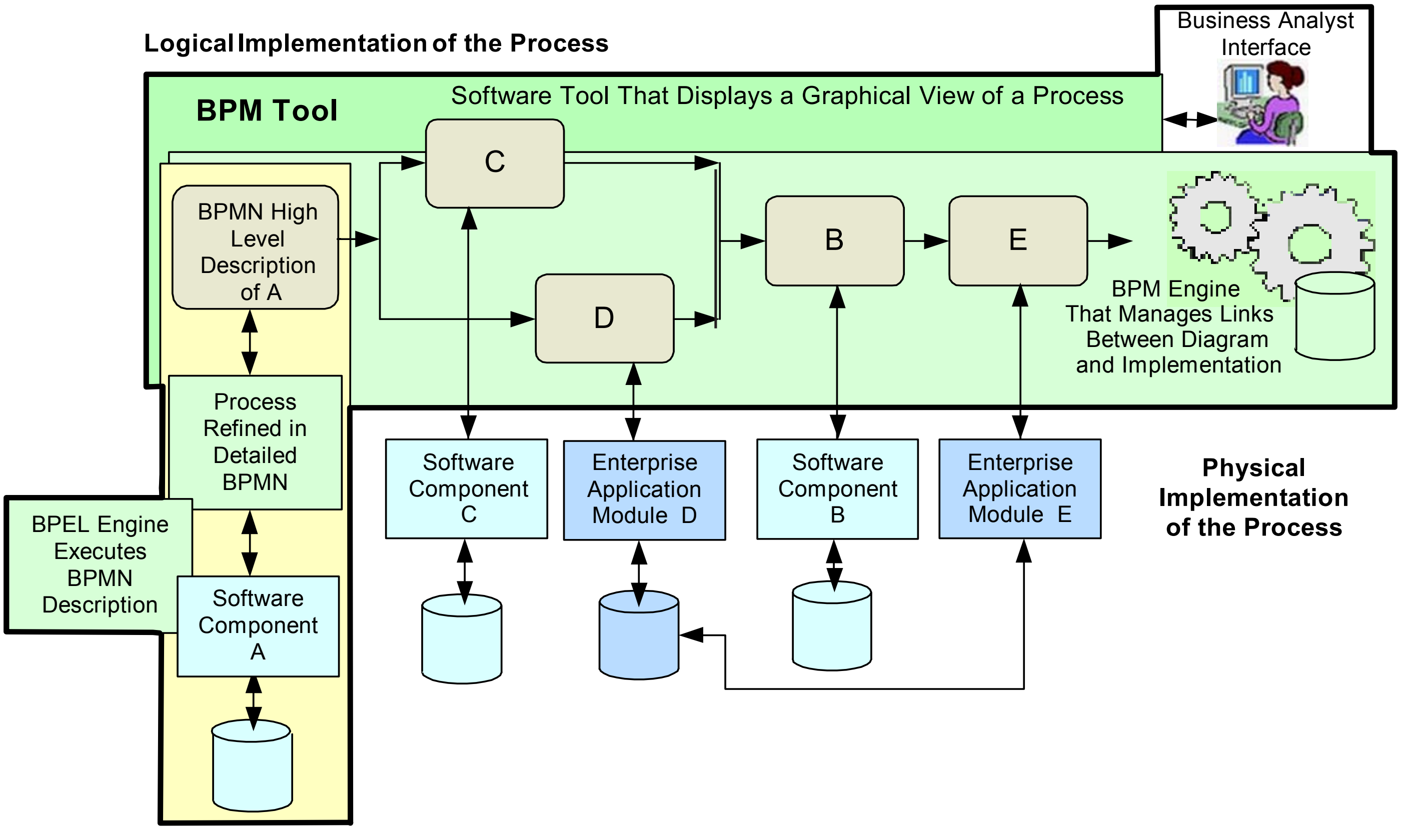
Logical Implementation of the Process
Business Analyst Interface
BPM Tool
Software Tool That Displays a Graphical View of a Process
C
BPMN High Level Description of A
D
B
E
BPM Engine That Manages Links Between Diagram and Implementation
Process Refined in Detailed BPMN
BPEL Engine Executes BPMN Description
Software Component A
Software Component C
Enterprise Application Module D
Software Component B
Enterprise Application Module E
Physical Implementation of the Process

Figure 6. A BPM product that uses BPEL and supplemental proprietary code to generate software code.

reasonably good at generating most of the code for COBOL applications running on mainframes, but these products were unable to scale up to the complex environments of the Nineties, with object-oriented languages and a variety of platforms with different operating systems.

MDA is a major effort to build on what the OMG has learned about enterprise integration and the widespread acceptance of UML. One key is the development of MOF that, in turn, allows architects to create a whole variety of metamodels that can all be mapped to each other via MOF. The other is major advance in techniques for generating code from diagrams that has been advanced by those involved in creating process control and real time systems in manufacturing.

MDA represents a major effort to organize what has been learned about abstraction, metamodels and software development into an open, well-integrated methodology. We fully expect the MDA approach to dominate enterprise software development practice throughout the remainder of this decade.

Business Process Management (BPM) is also concerned with using models. In some sense, BPM is more abstract and attempts to do things MDA would not try. For example, many BPM products generate task lists that can structure the tasks of employees working on manual tasks. Similarly, BPM tools seek to be so user friendly, that business analysts and even managers can examine a business process and modify the sequence in which its components are executed.

We suggest, however, that we have not yet reached the point where business analysts can routinely create software by specifying an activity sequence. There is simply too much involved in working out the logic of complex tasks. It isn't something that it makes sense for a business manager to attempt. In the course of this decade, we will see BPM tools created that empower business analysts and they will be combined with MDA products that empower software developers. Both groups will increasingly rely on diagrams and models and interpreters to facilitate their work, but the two groups will remain independent. Business managers will increasingly have control over the flow of existing processes, but IT developers will still be needed to create

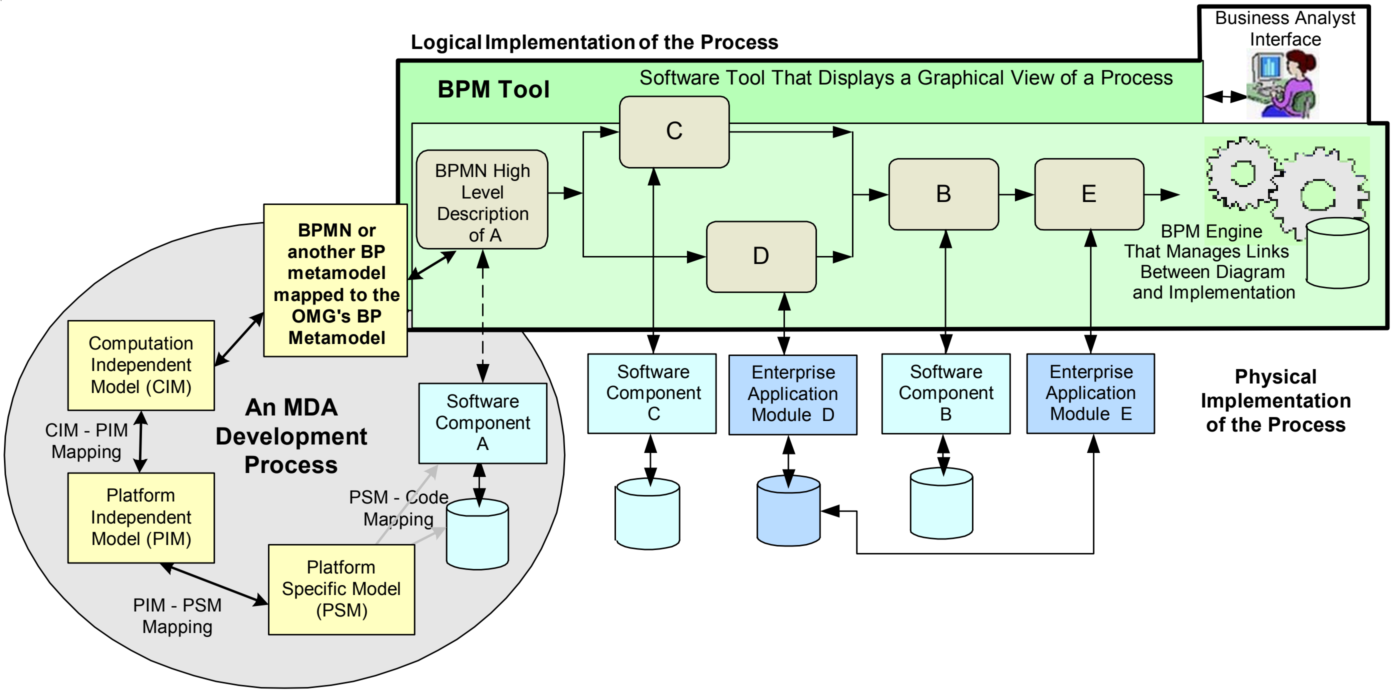
Logical Implementation of the Process
Business Analyst Interface
BPM Tool
Software Tool That Displays a Graphical View of a Process
BPMN High Level Description of A
C
D
B
E
BPM Engine That Manages Links Between Diagram and Implementation
BPMN or another BP metamodel mapped to the OMG's BP Metamodel
Computation Independent Model (CIM)
CIM - PIM Mapping
An MDA Development Process
Platform Independent Model (PIM)
PIM - PSM Mapping
Platform Specific Model (PSM)
PSM - Code Mapping
Software Component A
Software Component C
Enterprise Application Module D
Software Component B
Enterprise Application Module E
Physical Implementation of the Process

Figure 7 A BPM product that relies on MDA for software development.

new software components. Luckily, MDA will made it possible for IT developers to rely on models that are better designed and integrated than in the past, and increasingly BPM and MDA will be linked together to speed change and increase everyone's productivity.

Notes

[1] For a comparison of UML and BPMN, see the white paper by Stephen White, *Process Modeling Notations and Workflow Patterns,* on www.bptrends.com. Use search.

For the latest developments on MDA, check **David Frankel's *MDA Journal*** on www.bptrends.com. Dave has recently published important papers by Microsoft and IBM, for example, defining their visions for MDA.

In addition, starting this month BPTrends has added a new columnist, **Stan Hendryx**, who is going on report on developments in Business Rules. Stan is a co-chair of the OMG's BEIDTF task force and reports, this month, on the latest OMG activities on the **Business Rules** Metamodels.

For detailed information on any MDA specifications, go to the OMG website: www.omg.org.

Several good books have recently appeared on MDA. Among the best are:

David Frankel. *Model Driven Architecture: Applying MDA to Enterprise Computing*. Wiley, 2003.

Anneke Kleppe, Jos Warmer and Wim Bast. *MDA Explained: The Model Driven Architecture: Practice and Promise.* Addison-Wesley, 2003.

Jim Arlow and Ila Neustadt. *Enterprise Patterns and MDA: Building Better Software with Archetype Patterns and UML.* Addison-Wesley, 2004.

The following books are more focused on the special problems involved in generating code from MDA models.

Stephen J. Mellor and Marc J. Balcer. *Executable UML: A Foundation for Model-Driven Architecture.* Addison-Wesley, 2002.

Richard Hubert. *Convergent Architecture: Building Model-Driven J2EE Systems with UML.* Wiley, 2002.

For examples of MDA applications, go to the OMG site, www.omg.org, and click on the MDA logo which takes you to the MDA site where there are many case studies.