
MDA Journal
November 2003

**David S. Frankel**
David Frankel
Consulting

## Introduction

As I go around the world talking with people about MDA, I always stress that you can't simply dump an MDA tool on an IT organization and expect a successful changeover to this new way of developing and managing software, no matter how good the tool is. Despite the fact that MDA stands on the shoulders of a number of previous technical advances, using it entails significant changes in the way IT puts its architecture and development teams together. It also alters IT's relationship to the business side of the house, and requires fairly deep changes in IT culture. Thus, embarking upon MDA requires an organized transition plan.

Mike Guttman, who played a key role in writing the CORBA 1.0 standard and in defining the GIOP/IIOP interoperability protocol for CORBA 2.0, spent the rest of his time in the 1990s co-founding and building a consulting company that was dedicated to helping IT organizations make the transition to distributed object computing and enterprise architecture. Now the OMG has recruited Mike to head up its "MDA FastStart" program. FastStart is a package of consulting services geared toward helping IT organizations to make the transition to MDA. The OMG has created a pool of service providers that can provide the program to end-users.

In this month's MDA Journal article, Mike lays out the basic structure of an organized transition to MDA, and describes the elements of the FastStart program. From speaking with Mike I know that he sees MDA as a natural evolution of distributed enterprise architecture, and his inclusion of an architecture review as part of FastStart attests to his strong background in this area.

Mike has seen both successful and unsuccessful attempts to undertake major new technologies in his 30 years in the industry. He has begun to chart a cautious, deliberate path to success with MDA. We would be wise to consider his perspective carefully.

Until next month,

David Frankel


MDA Journal
November 2003

**Michael Guttman**
Director
MDA FastStart Program
Object Management Group

Contents







# BRINGING IN MDA

As MDA gains market awareness, many organizations want to find out how to bring MDA into their own software development process. Interestingly, transitioning to MDA can itself be treated as a process that can formally modeled using an MDA-based approach. A new initiative from the OMG called MDA FastStart is doing just that.

## What's Important About MDA?

Model Driven Architecture® (MDA) is a comprehensive approach to information systems engineering that systematically addresses the complete life cycle of automating business processes through software. MDA focuses on formalizing and standardizing the artifacts associated with designing, deploying, integrating, and evolving supporting software applications.

In short, MDA is a way to bring order to the increasingly complex and often fragmented world of enterprise computing. The introduction of MDA into a typical corporate IT environment rapidly pays for itself in reduced software development and maintenance costs, as well as improved time-to-market and quality for mission-critical applications.

More specifically, MDA provides a way to enhance the development of robust and flexible enterprise applications and computing infrastructures. To do so, MDA leverages a number of powerful enterprise technology standards related to the OMG's popular Unified Modeling Language (UML). In particular, this includes OMG's Meta-Object Facility (MOF) and the closely related XML Metadata Interchange (XMI), which are used to manage the metadata generated during MDA-based modeling activities. Other powerful standards included under the MDA umbrella include the Common Warehouse MetaModel (CWM), used for data modeling and warehousing, and the Software Process Engineering Metamodel (SPEM), used to model and document software engineering processes.

CWM and SPEM are examples of MDA's recognition that UML will exist in an environment that includes other modeling languages, such as data modeling languages and business process modeling languages. MOF is the basis for harmonious use of multiple modeling languages.

Functionally speaking, MDA provides a set of conceptual tools that support the modeling of software in a way that is completely independent of any specific technology ultimately used to implement and deploy that software. In doing so, MDA produces platform-independent models (PIMs) that can be mapped to multiple present or future computing platforms. The resultant platform-specific models (PSMs) can then be used to generate code, messaging formats, communications protocols, data structures, configuration information, etc., as necessary for the chosen implementation and deployment technologies.

In this way, MDA has already been applied to a wide array of common computing technologies, including popular middleware and messaging technologies such as W3C's XML, Sun's J2EE™ and EJB™, Microsoft's .NET, and, of course, OMG's own CORBA®. In conjunction with such technologies, MDA provides a way to gracefully incorporate legacy systems and legacy languages into MDA-based enterprise architecture.

Although the Object Management Group (OMG) formally introduced MDA quite recently (in 2002), and it will take some time before the MDA vision is fully realized, interest in MDA has already grown rapidly. In part because MDA systematically builds upon earlier widely accepted computing technology standards such as UML and leverages accepted approaches such as design patterns, component-based development, and n-tier architecture, MDA has already received strong endorsements from major systems and software vendors such as IBM/Rational, Microsoft, and Sun. At the same time, a significant number of smaller vendors have already "bet the farm" on MDA as the backbone for their new product sets.

In particular, many popular existing UML-based modeling tools and Java-based interactive development environments (IDEs) have already been adapted or extended to support MDA, giving MDA a kind of instant market share. Major examples include IBM's Eclipse and Rational Rose and Compuware's Optimal J. There are also a number of powerful MDA-centric tool suites from smaller vendors, such as Interactive Object's ArcStyler, Kennedy-Carter's iUML, Codagen's Architect, Adaptive's Framework, Metanology's MDE, and Osellus' IRIS, to name just a few (for a more complete list, see http://www.omg.org/mda/committed-products.htm).

As a result of this commercialization of MDA, a substantial number of end-users have already started incorporating such MDA-ized tools into their development process, providing the industry with some valuable experience about how to best introduce MDA. As industry awareness of MDA continues to build, many other software vendors and end-users are beginning to ask, "What exactly is MDA and how can my organization best start benefiting from it?" While there is clearly no "one size fits all" answer to this question, there are already some clear best practices on how to introduce MDA into a "typical" IT environment.

## What's Different About MDA?

However, before we go any further, we first need to make clear that MDA is a little different in composition than most earlier IT standards, even those coming from the OMG. For example, the OMG's CORBA standard describes the architecture of an "object request broker," a specific piece of software, while OMG's UML standard defines a specific language for expressing models. Similarly, related non-OMG standards such as Java and XML are narrowly focused – Java is a specific implementation language, and XML is a specific messaging format. In most cases, the main audience for such standards is largely technical.

By comparison, MDA is much broader, potentially incorporating a whole range of industry standards and best practices into a comprehensive framework. Ultimately, when MDA reaches its full maturity, it will be the basis by which an organization can manage all software artifacts in a consistent manner.

In this respect, MDA is really as much of a way of thinking and a set of guiding principles as it is a collection of specific standards. Each individual part of MDA has value in itself, but MDA as a whole is truly very much greater than the sum of its parts. Therefore, for an organization to get a full understanding of MDA—and the maximum benefit from applying it—a broad internal audience ultimately needs to be involved in that organization's transition to MDA.

In other words, MDA may start out in the chief architect's office, or in the business modeling group, or in data warehousing, or the Web services group, but ultimately it will find its way around the whole IT organization, including both business and IT management. This may initially start to happen by some form of informal osmosis, but ultimately the organization's best "bang for the buck" will be realized by having IT management more proactively plan and manage the MDA transition process.

On the other hand, this certainly doesn't mean that everyone in an end-user IT organization has to understand all aspects of MDA all at once. This kind of firehouse approach to introducing new technology seldom works very well, and it definitely won't work for something as comprehensive as MDA. Instead, it means that management—both IT and business—needs to develop a systematic, organized approach to rolling out MDA over a period of time to a number of different constituencies.

It should come as no surprise that the kind of systematic, formal approach to a technology transition that we are advocating for MDA is quite different than most IT organizations have experienced with the introduction of earlier technology standards. In the past, most organizations have adopted new technologies only after they were incorporated into commercial products and thoroughly commoditized. In fact, within many organizations, the technology itself only had meaning in terms of product set that was used to introduce it.

For example, Java and later J2EE became household acronyms in the corporate IT world only after most vendors adopted them as the basis for Web-enabling software development environments. Before that, only a small and narrow audience was very interested in the object-oriented languages that preceded Java or distributed component deployment platforms that preceded J2EE, even though they had many—if not most—of the same basic features. The awareness of the underlying technologies behind Java and J2EE only made headway in mainstream IT organizations once they were tied to specific products whose use was imperative for the rapid development and deployment of Web applications.

Certainly some elements of MDA will also follow this traditional awareness trajectory. For example, some of the software development environments mentioned earlier (e.g. IBM's Eclipse) are incorporating MDA-based features that

significantly improve the software development process, such as synchronizing models with code. Some tools provided such features before MDA, but MDA will ultimately allow all tools to provide them in a way that is based on standards.

Initially, many such features have been introduced as "embedded" MDA—that is, the IT end-user (i.e. a programmer or modeler) initially doesn't necessarily need to know that this or that particular feature is driven by MDA. Over time, use of such tools will gradually raise the awareness of how the various features fit together to constitute a greater whole. At the same time, vendors will find it increasingly beneficial to advertise their features as MDA-based, and MDA compliance will probably become at least a "check-list" item for certain kinds of tools.

This kind of "transition" to MDA has its uses, particularly in terms of slowly elevating the awareness of MDA in the mass market and creating commercial traction for certain MDA-based products. But there are many IT organizations that can to benefit from a more rapid and systematic transition to MDA now—without waiting for MDA to completely cross the mass-market chasm and become a household acronym. These organizations need an efficient, cost-effective way to incorporate MDA into their software development activities as soon as possible.

Furthermore, it runs against the grain of MDA itself to view the transition to MDA only in terms of incorporating a specific MDA-based product set. The whole point of MDA is that it allows users to separate out the platform-independent elements of a business process from the platform-specific components used to automate it. Therefore it behooves organizations intending to introduce new MDA-based technologies to take the same approach.

In other words, we are advocating nothing more or less than an MDA-based approach to implementing the business process of introducing MDA itself! In practice this means developing a formal MDA transition process, independent of any particular toolset for implementing that transition process. Once the MDA transition process is formalized in a platform-independent way, we will be better able to customize the process to the needs of individual end-user organizations. We'll also be better able to support the implementation of the transition process using whatever best-in-breed MDA-based tools emerge over the next few years.

## Towards a Formal Process for Transitioning to MDA

The continued development and popularization of MDA is ultimately the responsibility of OMG, a non-profit computer industry consortium. Not surprisingly, the OMG receives numerous requests from end-user organizations–including some of OMG's own members–looking for help to accelerate their own transition to MDA.

Recently, the OMG decided to inaugurate the MDA FastStart Program Office, with the specific aim of helping such organizations initiate a formal transition to MDA. As you may have already guessed, the need for a formal model of the

MDA transition process has not escaped the notice of the new FastStart Program Office.

Details about the overall activities of the MDA FastStart Program Office can be found at the OMG website (www.omg.org/fast-start). However, for the purposes of this article, we will concentrate on a single activity–the development of a formal MDA FastStart transition process, which we call MDA-FS. As its name implies, MDA-FS is a formal process to guide a rapid transition to MDA within a prototypical IT organization.

Perhaps the most interesting and unique characteristic of this activity is that the MDA FastStart Program Office has decided to use MDA itself to document the process elements of MDA-FS. When complete, the entire MDA-FS transition process will be available as an integrated set of MDA-based models. In this context, the completed MDA-FS can be seen as a business process formally modeled as a PIM. For once, the cobbler's children are getting some shoes!

There are several rationales behind the effort to formalize MDA-FS using MDA itself. Obviously, it doesn't hurt to be able to show that the OMG itself believes enough in MDA to use it to model its own processes–particularly those related to training people to use MDA. And, of course, the formal model of MDA-FS can be showcased as a specific example of how MDA can be used to define, document and automate a significant business process.

But the advantages of using MDA to define MDA-FS don't end there. Because MDA-FS is defined using formal elements of MDA, the various elements of the MDA-FS PIM can be loaded into a variety of MDA-compliant tools. For example, the PIM models can be loaded into an MDA modeling tool, which can then be used to customize the MDA-FS PIM for a particular organization. The resulting models can eventually be fed into an MDA-based process tool, which can be used to associate the various MDA-FS roles with specific actors and track the transition process, while using an MDA–based repository to store, version and manage the related artifacts.

If that sounds interesting to you, it's probably time to give some concrete examples of MDA-FS and our on-going effort to formalize it. First we'll start with an English-language narrative description of MDA-FS, and then provide examples of its formal definition using MDA's Software Process Engineering Metamodel (SPEM). Given the limitations of this article format, both the narrative and model examples will be significantly abbreviated. Many thanks to Mike Rosen and Viktor Ohnjec of M2VP (www.m2vp.com) for assisting in the English-language description of MDS-FS, and to Vivienne Suen of Osellus (www.osellus.com) for helping reduce this to a set of SPEM-based specifications.

## MDA-FS Distilled -- in English

As its name implies, MDA-FS is a process specifically designed to rapidly familiarize an end-user's IT organization with MDA concepts and to start integrating the MDA approach into mission-critical software development activities.

### Goals

During a MDA-FS engagement, highly qualified MDA consultants and trainers provide an integrated set of assessment, planning, executive seminar and technical practicum activities targeted to both top executives and technical staff. MDA-FS deliverables are designed to help key decision makers:

- Analyze and plan how MDA can best be introduced and applied to most benefit their organization and its key business drivers.
- Demonstrate how MDA can provide clear-cut business value sufficient to justify further investment in MDA-related activities
- Attain sufficient knowledge of MDA to confidently initiate further MDA-related activities that will ultimately transition the entire organization

### Activities

MDA-FS includes the following major activities:

- ***MDA-FS Readiness Assessment*** - used to determine the best overall approach to introducing MDA into the organization. It includes the following steps:

  1. High-level review of the business drivers relevant to an MDA transition, with identification of target MDA-related technical activities best suited to those drivers
  2. High-level review of the various technological drivers and tools in the organization that would influence the MDA transition, including current software development processes and related computing infrastructure
  3. High-level review by FastStart consultants of the organizational scope, size, and structure of the target MDA users, as well as the human resources available to support MDA activities

- ***MDA-FS Architecture Review*** – used to determine in more detail the technology specifics of the organization's current architectural foundation with the aim of understanding:

  1. The level of maturity associated with current enterprise/application architecture efforts
  2. What parts of the enterprise architecture and computing infrastructure offer the most beneficial, cost-effective initial focal points for future MDA-related efforts
  3. How MDA can best be adapted to the architectural paradigms currently in use

- ***MDA-FS Transition Plan*** - a set of recommendations and a high level plan for introducing and incorporating MDA into the customer's organization:

1. Based on MDA best practices and the results of the Assessment and Architecture Review

2. Includes the selection of a set of specific technical activities for the MDA Practicum activity

3. Includes an evaluation of the various MDA-related tools that are appropriate for the customer's organization and technical requirements.

- ***MDA-FS Executive Seminars*** – a set of short training models customized as appropriate for each particular organization, typically including:

  1. An Executive Seminar aimed at executive and IT management describing the benefits, requirements and timeframe for implementing an MDA transition program in their organization. This seminar also includes the presentation of the Assessment and Architecture Review reports and recommendations, as well as the proposed Transition Plan.

  2. A Technical Seminar aimed at architects, designers, modelers and engineering managers to discuss the principles, techniques, tools, and so forth involved in implementing MDA.

- ***MDA-FS Practicum*** - a highly interactive activity aimed at senior technical staff, such as enterprise and system architects, designers and project managers. The Practicum includes formal, detailed presentations on MDA technology topics and a set of workshop periods that allow the participants to apply what they learn in a pragmatic fashion.

## MDA-FS Distilled -- in MDA

In order to reduce MDA-FS to a formal specification, we graphically modeled the above description using the interactive modeling tool in IRIS, an MDA-based modeling tool from Osellus, Inc. Figure 1 shows an example of how MDA-FS can be rendered as a SPEM model. This figure only shows MDA-FS at the very top level. As mentioned earlier, the OMG FastStart Office is working on a more detailed model.

As its name implies, SPEM (Software Process Engineering Metamodel) is a MOF-based metamodel defined specifically to support the specification of software engineering processes. IRIS, which is based on SPEM, provides support for modeling these processes. Furthermore, IRIS also supports automating any SPEM model you define (either in IRIS or another SPEM modeling tool) through a run-time process management engine.

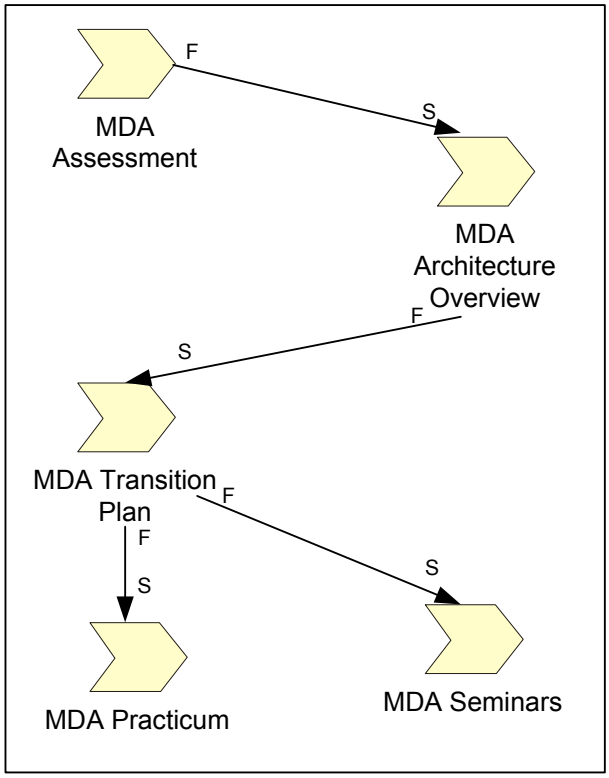


**Figure 1 - Overview of MDA-FS Modeled in MDA SPEM**

Figure 2, on the next page, shows a fragment of the SPEM specifications generated automatically by IRIS from a particular view of the graphical model of MDA-FS we developed. Loosely speaking, it represents a portion of the graphical example shown in Figure 1. In this case, we chose to render the model in XML/XMI, which is consistent with an MDA approach. Output like this will ultimately be used to feed the model into other MDA-compliant tools.

## Some Final Observations

We have just described an important process, MDA-FS, in three different ways: English narrative, a formal model diagram, and a text-based expression of that model that could ultimately used as input to other MDA-complaint software development tools used to automate MDA-FS.

Of course, before MDA, it was certainly possible to create such models by hand, or even to develop or purchase a tool to support the modeling process and automate the production of diagrams and structured text. However, any resulting models would be based on a proprietary metamodel that, in most cases, would not even be formally specified itself. Therefore, anyone using a pre-MDA tool for this purpose would, at best, be completely locked into that tool (and its vendor) forever.

```
        <UML:Activity name="MDA Architecture Overview">
            <XMI.extension xmi.extender="">
                <UML:ExternalDescription.content>The purpose of the MDA Architecture Overview is to determine in detail the
technology specifics of the current architectural foundation.</UML:ExternalDescription.content>
            </XMI.extension>
            <UML:Activity.steps>
                <UML:Step name="Step 1">
                    <XMI.extension xmi.extender="">
                        <UML:ExternalDescription.content>Assess the level of maturity associated with current enterprise/application
architecture efforts.</UML:ExternalDescription.content>
                    </XMI.extension>
                </UML:Step>
                <UML:Step name="Step 2">
                    <XMI.extension xmi.extender="">
                        <UML:ExternalDescription.content>Identify the parts of the enterpise architecture and computing
infrastructure that offer the most beneficial cost-effective initial focus points for future MDA-related efforts.</
UML:ExternalDescription.content>
                    </XMI.extension>
                </UML:Step>
                <UML:Step name="Step 3">
                    <XMI.extension xmi.extender="">
                        <UML:ExternalDescription.content>Decide how MDA can best be adapted to the architectural paradigms
already in use.</UML:ExternalDescription.content>
                    </XMI.extension>
                </UML:Step>
                <UML:Step name="Step 4">
                    <XMI.extension xmi.extender="">
                        <UML:ExternalDescription.content>Document all findings and recommendations in the Architecture Report,
and present to technical management.</UML:ExternalDescription.content>
                    </XMI.extension>
                </UML:Step>
            </UML:Activity.steps>
            <UML:Feature.owner>
                <!--refernce to MDA Architect-->
            </UML:Feature.owner>
            <UML:BehavioralFeature.parameter>
                <UML:ActivityParameter kind="in">
                    <XMI.extension xmi.extender="">
                        <UML:ExternalDescription.content>The infomation in the Assessment Report identifies the context and main
business and technical drivers for the Architecture Overview.</UML:ExternalDescription.content>
                        <!--Reference to WorkProduct-->
                    </XMI.extension>
                </UML:ActivityParameter>
                <UML:ActivityParameter kind="out">
                    <XMI.extension xmi.extender="">
                        <UML:ExternalDescription.content>The major devliverable of the MDA Architecture Over is the Architecture
Report</UML:ExternalDescription.content>
                        <!--Reference to WorkProduct-->
                    </XMI.extension>
                </UML:ActivityParameter>
            </UML:BehavioralFeature.parameter>
        </UML:Activity>
```

**Figure 2 - MDA-FS Specified in SPEM - A Fragment Expressed in XML/XMI**

As an industry consortium devoted to open standards, that kind of vendor lock-in would be a particular anathema to the OMG. Therefore, although our model is being initially developed in a commercial tool, that tool is based on SPEM, which is itself based on MOF and UML, which are in turn the basis for a whole integrated suite of open MDA standards. Therefore, the model of MDA-FS we produced with IRIS is now potentially intelligible to a wide range of other MDA-based tools (current and future), including UML modeling tools, code and message generating tools, workflow and process management engines, metadata repositories, and so on.

What does that mean to the end-users, in this case the OMG's own MDA FastStart Program Office and its clients? When our generic MDA-FS model is completed, the FastStart Program Office intends to make it available to the general public. Using that model, anyone can purchase an off-the-shelf SPEM modeling tool (or "roll" their own) and specialize the generic MDA-FS model to his/her own specific MDA transition requirements.

Furthermore, anyone will be able to feed their specialized version of OMG's MDA-FS model into any run-time SPEM-compliant process engine and use it to support and manage their own individual transition projects. This could be a real boon to large users and/or consultants who will be running multiple MDA transition projects. It may even spur more vendors to create commercial SPEM modeling and run-time products, perhaps including a commercial MDA transition modeling and management product or plug-in.

Finally, those FastStart clients who are introduced to the MDA-FS model will not only get to apply that model to their own MDA transition, but also to view the model itself as a living, breathing embodiment of how MDA can be applied to support the documentation and automation of any sophisticated process in a tool-independent fashion. In other words, the OMG's own well-shod shoemaker's children will be a living advertisement to other end-users for the efficacy of making their own transition to MDA-compliant shoes.