Accelerating Embedded Software Development with a Model Driven Architecture®

Carolyn K. Duby

Pathfinder Solutions

September, 2003

www.pathfindersol.com

Introduction

Embedded system software developers are challenged to meet delivery dates in the face of changing requirements, complex and sometimes-fragile system architectures and ever-evolving technological platforms.

To address these challenges, organizations have invested in application modeling using Unified Modeling Language™ (UML™) or earlier-generation approaches such as Shlaer-Mellor. While many expected UML modeling to substantially improve requirements capture, system design and increase component reuse, in reality its benefits have often been limited to facilitating design documentation and discussion.

Enter the Model Driven Architecture (MDA®) framework from the Object Management Group® (www.omg.org). Created by a consortium of software development professionals, MDA raises the return on your modeling investment to a new level.

This white paper will help you understand what MDA is, how best to adopt it and the benefits it provides when implemented with a model automation and transformation environment such as PathMATE™ from Pathfinder Solutions. Benefits include:

- Faster, more predictable software delivery cycles
- Minimizing the impact of requirements changes on development schedules
- Greater component reuse & implementation consistency
- Architectural flexibility and platform independence
- And others…

MDA Defined

MDA is a standard framework for modeling software systems. MDA-conformant models not only capture and delineate the objectives and design of an application, but when processed by MDA automation and transformation tools, they also drive:

- The automation and execution of your application model for testing and validation purposes– before you write any code
- The automatic transformation of models into tested, deployable applications

MDA accomplishes this by separating "what" the system must do from "how" it is implemented on a specific technology platform. MDA systems comprise two parts:

- The **Platform Independent Model** (PIM) specifies what the system does
- The **Platform Specific Model** (PSM), specifies how the system is implemented

The PIM captures the essential features, or "business logic" of the system. The PSM determines how the PIM executes in the target deployment environment. The PSM may be represented in a variety of forms including executable code such as C, C++, or Java. MDA tools transform PIMs into PSMs as illustrated in the next section.

Applying MDA for Embedded & Real-time Systems

MDA is well-suited for embedded software development because it separates functional logic from implementation details and with the right MDA technology, automates the generation and testing of any embedded application architecture. MDA provides embedded software developers with a fundamentally different and higher-level way to accommodate changing requirements, increase reuse and extend system longevity.

Transformation of PIMs to executable PSMs is automated via off-the-shelf, yet customizable template-based transformation technology such as PathMATE. Figure 1 illustrates the construction, transformation, and verification of models with PathMATE.

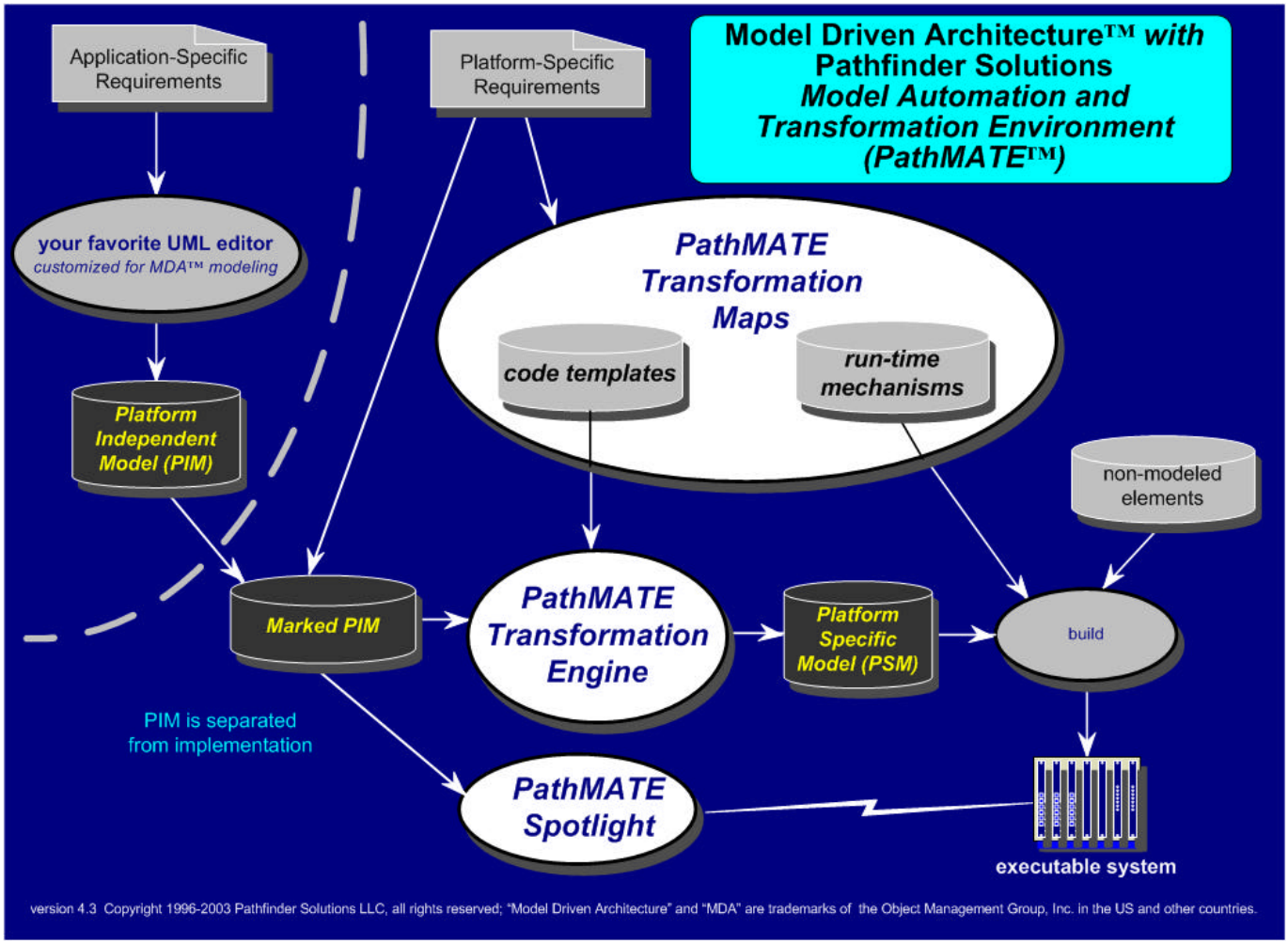

Figure 1 – MDA work flow and key technology elements

1. The PIM is constructed in one of several supported UML editors such as Rational Rose from IBM, which have menu items customized for integration with PathMATE.
2. Optionally, a set of platform-specific markings may be specified on the elements of the PIM. The markings consist of a set of properties and stereotypes that guide the choice of transformation rules and optimizations (such as whether to

generate a single- or multi-threaded application, how much debug code to generate into a resource-constrained deployment target, etc.). Because markings are stored separately from the PIM, different markings can be applied to the same PIM to yield different PSMs.

3. After completing an increment of the PIM, the developer invokes the PathMATE Transformation Engine ("the Engine"). The Engine reads the PIM and applies to it an off-the-shelf or custom PathMATE Transformation Map (e.g., the C Map, the C++ Map, the Java Map, etc.) for the target implementation platform.
4. The Map consists of a set of templates and run-time mechanisms. The templates specify the rules for transforming the PIM into executable code. The run-time mechanisms are a set of implementation utilities required by the generated code. The templates read platform specific markings to determine when to apply an optimization or transformation rule that can't be derived directly from the PIM.
5. The Engine generates the source code representing the PSM from the marked PIM and templates.
6. The PSM is integrated with the run-time mechanisms and any non-modeled code including off-the-shelf components or hand-written code to form the executable system.
7. Then PathMATE Spotlight ("Spotlight") is used to debug and test the PSM. Spotlight can execute the PSM running on the development environment for early, iterative testing or on the target hardware to quickly isolate environmental causes behind behavioral and performance-related defects.

Why MDA?

Through the application of MDA standards, customizable model automation & transformation technology and conscientious PIM definition, a software organization eliminates substantial downstream coding and QA from the development process– without sacrificing implementation flexibility or control. As we all know– less manual coding and earlier bug detection can dramatically increase the probability of delivering a high-quality system on time and within budget.

MDA also enables you to:

React quickly to changing functional & technological platform requirements
The separation of the PIM from the PSM allows you to quickly react to changes in execution requirements without having to change the PIM. For example, if you need to deploy on a new platform there is no need to change the PIM – you simply apply a different Map. If you need to change the processor topology, you just adjust model markings. If you need to apply a new optimization, Map templates and mechanisms are readily customized. If functional requirements change, you can integrate the new feature at the PIM level.

Substantially extend the longevity of the system
As platform-dependent systems are maintained, the original architecture may no longer be able to satisfy new requirements. Rather than take the time to re-architect,

most organizations only have time to apply patches and spot corrections, which can cause the architecture to become brittle. With MDA, function and architecture are defined separately and architectural changes are implemented automatically via transformation. Radical changes to architecture and function can occur– independent of the other, which extends the life cycle of most MDA systems.

Improve developer productivity

Best-practice modeling techniques separate a system into highly cohesive components. This separation is fundamental to MDA and simplifies each system component, which yields consistently implemented components that are easier to create, develop, reuse and maintain. The Engine automatically produces high-quality and complete implementation code from models freeing developers to concentrate on defining additional customer-driven functional specifications in the PIM or on creating or extending transformation maps and optimizations.

Enable large-scale reuse of PIMs

Different Maps and settings can be applied to the same PIM to produce multiple component implementations in different application contexts. Thus a PIM can be reused in more than one system.

Lower maintenance costs

Since the code is generated from the models, you know that your models and code are always in sync. Developers new to the system are able to get up to speed quickly because they have a reliable high-level graphical view of the system.

Ease documentation burdens

Keeping design documents up to date with code by hand is tedious and time-consuming. However, with MDA the models, code, and documentation are always in sync. The PathMATE Documentation Map generates custom documentation containing the models and their associated descriptions.

Reduce quality assurance costs

The later a software error is discovered during the development process, the more expensive it is to fix and the more jeopardized a delivery date becomes. MDA model automation and testing tools like PathMATE Spotlight help developers test their applications– at the model level, before coding begins. As a result, design flaws and application logic errors are uncovered much further upstream in the development process. In addition, Spotlight can automate and test models on the target hardware to help uncover platform-specific problems earlier in the process.

Improve quality

The fundamental simplicity of PIMs brings substantially improved system quality. Modeling helps improve communication between team members and facilitates early elimination of defects. The Engine automatically applies coding patterns to the models eliminating defects introduced by hand coding.

Getting Started with MDA

Often the biggest barriers to adopting effective software engineering techniques and technology are not technical or even financial. Even with standards such as MDA based on proven technology, significant barriers to progress can stem from both management and cultural aspects. To facilitate the adoption of MDA, Pathfinder Solutions suggests the 'wedge' approach outlined below to mitigate the risks of new technology adoption:

Diagnose Your Unique Challenges:

- Outline the key software development challenges that you face
- Develop a detailed diagnosis of these issues with an expert practitioner
- Identify where effective MDA techniques and tooling can meet these challenges

Build a Solution Strategy:

If your key organizational goals are aligned with MDA benefits, and MDA techniques and tools address your top challenges effectively:

?? Acquire MDA tools, which are appropriate for your environment:
  - Integrate with your existing UML tools & infrastructure
  - Support your development and deployment languages and platforms
  - Possess fast, configurable and easy-to-extend transformation technology
  - Enable the testing of models on target deployment platforms– even if they are resource constrained
  - Work seamlessly across modeled and non-modeled system components

?? Secure the assistance of proven experts and methods to help you identify the MDA solution elements to meet your unique needs. Pathfinder Solutions has highly experienced consultants with many successful deployments, and can help your team build a successful deployment strategy.

?? Consider technological and cultural hurdles, and design a deployment plan that manages complexity and risk with a step-wise introduction

Deploy

Once you have identified a step-wise strategy for deploying MDA, execute it:

?? Train the team with Pathfinder's Waypoints Training for MDA / UML

?? Manage the initial scope of adoption through a pilot effort, or by focusing on an application fragment for MDA deployment

?? Mitigate technology risk with Pathfinder's expert practitioners to mentor your staff, establish sound modeling techniques, provide critical feedback and help your team avoid common and costly pitfalls

Refine and Expand:

?? Based on the experience of your initial effort, refine your solution strategy

?? Widen the deployment to reap the benefits on a larger scope

Summary

For embedded systems and other high-performance software, the separation aspects of MDA bring fundamental improvements in system component simplicity. This simplicity in turn yields substantial benefits in a wide range of areas such as better responsiveness to changing requirements and faster, more reliable software delivery via application generation and the ability to perform integration testing, at the model level, much earlier in the development cycle.

By focusing on key issues, you can build a feasible and low-risk plan to move your organization to MDA by applying basic technology management fundamentals including the application of proven technology like the Pathfinder Solutions PathMATE toolset.

About Pathfinder Solutions

Headquartered outside of Boston, Massachusetts, Pathfinder Solutions provides embedded software developers with the tools, methodologies and services required to advance their development processes and gain time to market and product quality advantages over the competition. Pathfinder Solutions is an active member of the OMG and is helping to shape the future of MDA.

Pathfinder Solutions PathMATE is the industry's most open and flexible MDA model automation and transformation environment. It is the only MDA solution that integrates with existing UML infrastructure and offers embedded software developers the control and performance they need to automate and test the production of applications in real-time or constrained environments.

If you would like to learn more about MDA or PathMATE or would like to discuss MDA adoption, please visit www.pathfindersol.com or contact them at:

Pathfinder Solutions
90 Oak Point
Wrentham, MA 02093
Phone: 508-384-1392
Email: info@pathfindersol.com