Pathfinder Solutions

Accelerate Embedded Software Development with Model Driven Architecture

By Carolyn Duby

Version 1.3

January 5, 2005

PathMATE™ Series

Pathfinder Solutions LLC
33 Commercial Street, Suite 2
Foxboro, MA 02035 USA

www.PathfinderMDA.com
888-543-7222

Table Of Contents

Introduction 1

MDA Defined 1

MDA for Embedded Systems..2

Why MDA? 4

React Quickly to Changing Requirements 4
Substantially Extend the Longevity of the System 4
Improve Developer Productivity 4
Enable Large-Scale Reuse of PIMs...4
Lower Maintenance Costs 4
Ease Documentation Burdens 5
Reduce Quality Assurance Costs..5
Improve Quality..5

Getting Started with MDA...5

Diagnose Your Unique Challenges 5
Build a Solution Strategy..6
Deploy...6
Refine and Expand 6

Summary 7

Next Steps 7

About Pathfinder Solutions 7

Introduction

Embedded systems developers must meet tight delivery dates in the face of changing requirements, complex and sometimes fragile software architectures, and ever evolving platforms. To address these challenges, organizations have invested in application modeling using the Unified Modeling Language (UML™), or earlier generation approaches such as Shlaer-Mellor. While many expected UML modeling to improve requirements capture, streamline system design, and increase component reuse, in fact its benefits have often been limited to facilitating design documentation and discussion.

Enter Model Driven Architecture (MDA) from the Object Management Group® (www.omg.org). Created by a consortium of software development professionals, MDA raises the return on your investment in modeling tools. This white paper helps 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. Benefits include:

- Faster, more predictable software delivery cycles
- Requirements changes have less impact on development schedules
- Greater component reuse and implementation consistency
- Architectural flexibility and platform independence
- And others…

MDA Defined

MDA is a standard framework for modeling software systems. When you design a model that conforms to these standards, you capture and delineate the objectives, attributes, and operations of the embedded system. When you process the models with MDA automation and transformation tools, you achieve:

- 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 these goals 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.

MDA for Embedded 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.

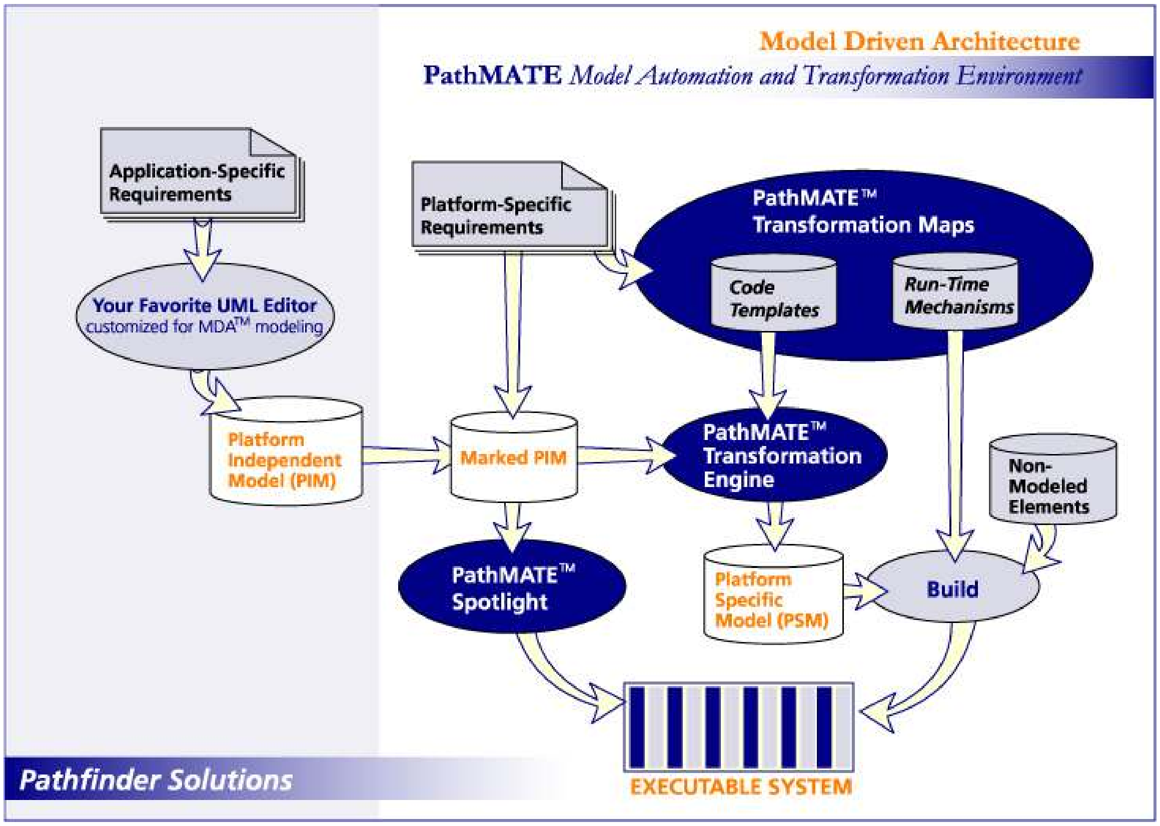
Model Driven Architecture
PathMATE Model Automation and Transformation Environment
Application-Specific Requirements
Your Favorite UML Editor
customized for MDA™ modeling
Platform Independent Model (PIM)
Platform-Specific Requirements
Marked PIM
PathMATE™ Transformation Maps
Code Templates
Run-Time Mechanisms
PathMATE™ Transformation Engine
Non-Modeled Elements
PathMATE™ Spotlight
Platform Specific Model (PSM)
Build
EXECUTABLE SYSTEM
Pathfinder Solutions

Figure 1. MDA Work Flow and Key Technology Elements

The steps below explain how this process works:

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, and how much debug code to generate into a resource-constrained deployment target.
    - Because markings are stored separately from the PIM, you can apply different markings 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 reads the PIM and applies to it an off-the-shelf or custom PathMATE Transformation Map that guides code generation for the target platform. PathMATE comes with Transformation Maps for these target languages:
    - C
    - C++
    - Java
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 use PathMATE Spotlight to debug and test the PSM. Spotlight executes the PSM in 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?

When you apply MDA standards, carefully drawn PIMs, and automated transformation technology to your embedded systems development, you eliminate substantial downstream coding and testing from the development process. 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. You also maintain close control over the development process and the quality of the embedded system your engineers develop.

React Quickly to Changing Requirements

The separation of the PIM from the PSM allows you to react quickly 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. Developers can concentrate on defining additional customer-driven functional specifications in the PIM, or on creating and 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 as software changes is tedious and time consuming. With MDA the models, the code, and the 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 factors. 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 appropriate for your environment:
  - ✓ Integrate with your existing UML tools and 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 scale.

Summary

Embedded MDA with PathMATE separates the logic of a system from its deployment on a specific platform. That simplifies the design of system components. Simple design yields substantial benefits all across the development cycle:

- Improve productivity in your initial development effort.
- React quickly to changing software and hardware requirements.
- Test integration of all system components at the model level, much earlier in the development cycle.
- Substantially reduce the time required for debugging.
- Improve reliability and performance.
- Consistently meet your deadlines.

Pathfinder's tools help you transform your models into executable code, predictably and accurately. Deploy faster, highly reliable embedded systems much more quickly than you thought possible.

Next Steps

If you would like to learn more about MDA and PathMATE, or learn if Pathfinder Solutions has helped an organization like yours benefit from this technology, please call or write. The white papers at www.PathfinderMDA.com contain much additional information. Above all, try the PathMATE toolset with a real system that you develop. The *PathMATE Quick Start Guide* and our engineers can help you get started.

About Pathfinder Solutions

Headquartered outside of Boston, Massachusetts, Pathfinder Solutions provides embedded software engineers with the tools, methods and services needed to reduce development costs and improve quality. Pathfinder Solutions is an active member of the Object Management Group, 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. It 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, please contact us at:

Pathfinder Solutions
33 Commercial Street, Suite 2
Foxboro, MA 02035
Phone: 508-543-7222
Email: info@PathfinderMDA.com