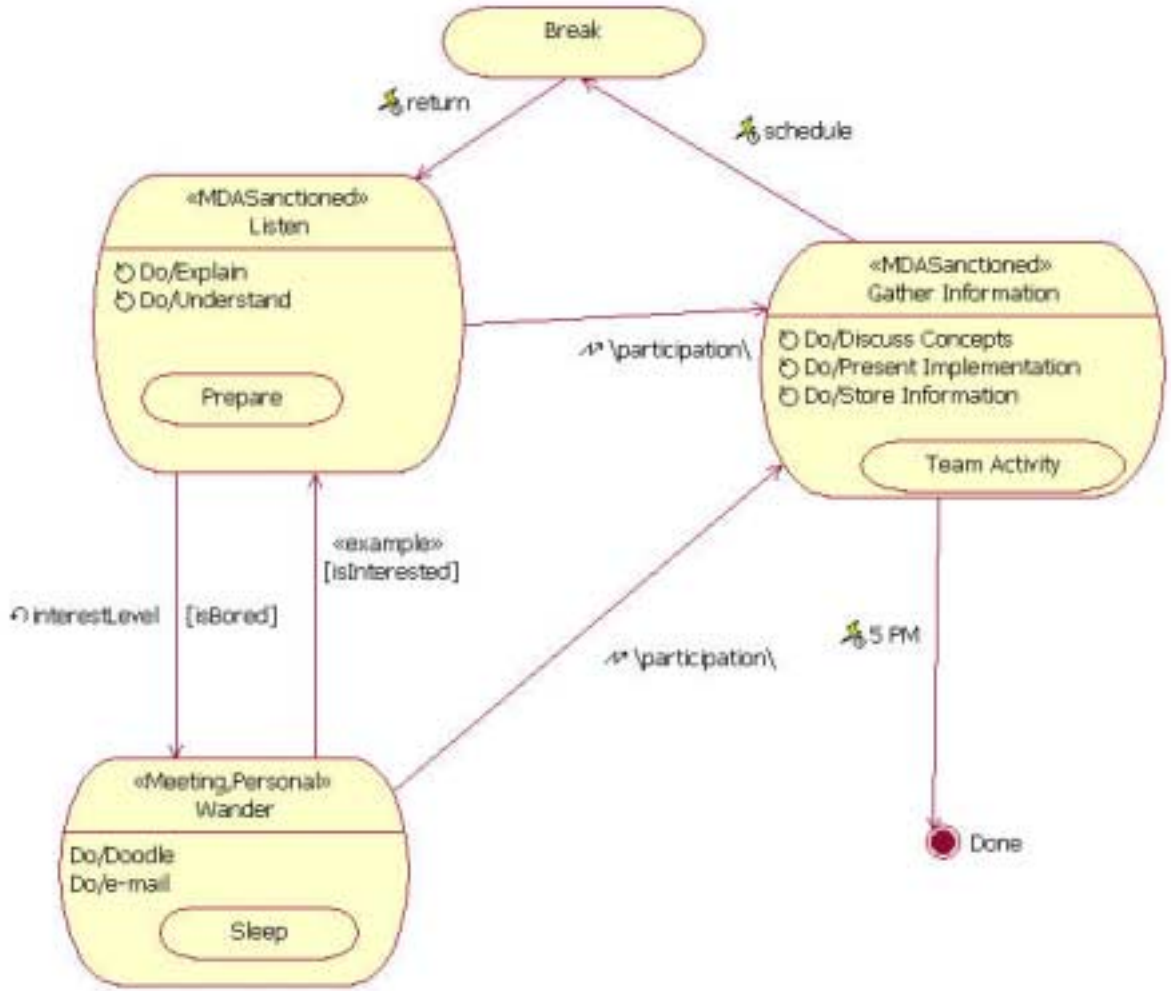
A Review of the MDA Toolkit

David H. Fado, Ph.D.

Advanced Systems & Concepts (ASC)

Note: This presentation does not represent SAIC or the Advanced Systems and Concepts Office (ASC) and is the approach of one member of the company.

Introduction: Computation Independent Model of the Tutorial



Overview

- **Section One: Review of MDA giving us:**
  - Defintion of basic MDA concepts such as PIM, PSM, CIM.
  - An invitation to discuss the MDA Logo
  - An invitation to speculate on what is MDA or not MDA based on a mature understanding of MDA's definition.
- **Section Two: Reviewing Diagrams over the Lifecycle give us:**
  - Worksheet of examples of models that worked and some examples of models that did not work.
  - Invitation to discuss work with others in the group and assess what conference participants want from the conference.
- **Section Three: Assessing Contribution give us**
  - Discussion of how MDA can help and also our concerns about MDA.
  - Time permitting, an example of how to use MDA in a decision support framework.

Questions to keep in mind for the Session

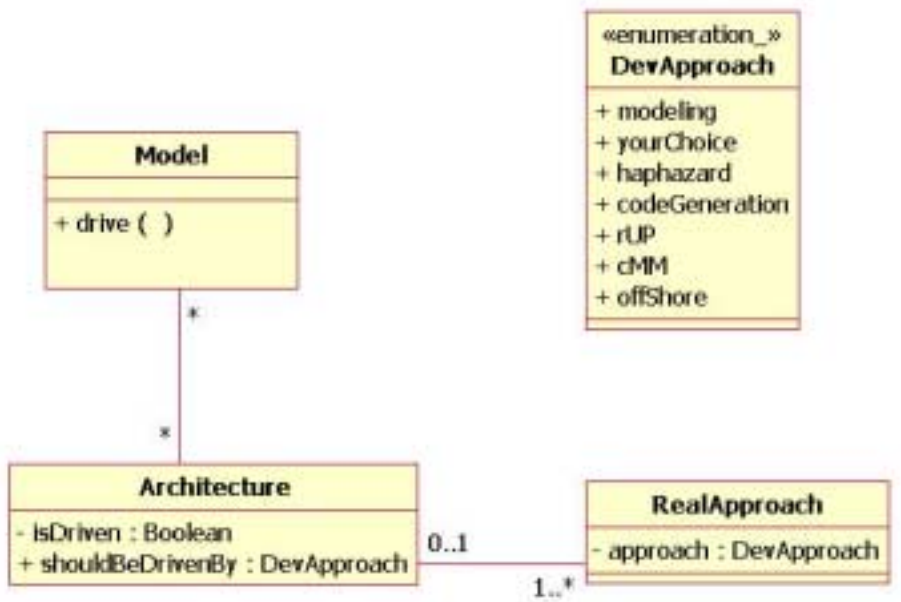
Gathering Information

- **What can one do with Model Driven Architecture?**
- **What is a Model? What is Architecture?**
- **What sort of Model can drive an Architecture?**
- **What are the key features of MDA?**
- **Why is the OMG supporting MDA?**
- **What attracts investment and attention to MDA?**
- **What will software development look like if MDA succeeds?**
- **What do you hope to gather during this conference?**

Section One MDA Basics: An Abstract Model

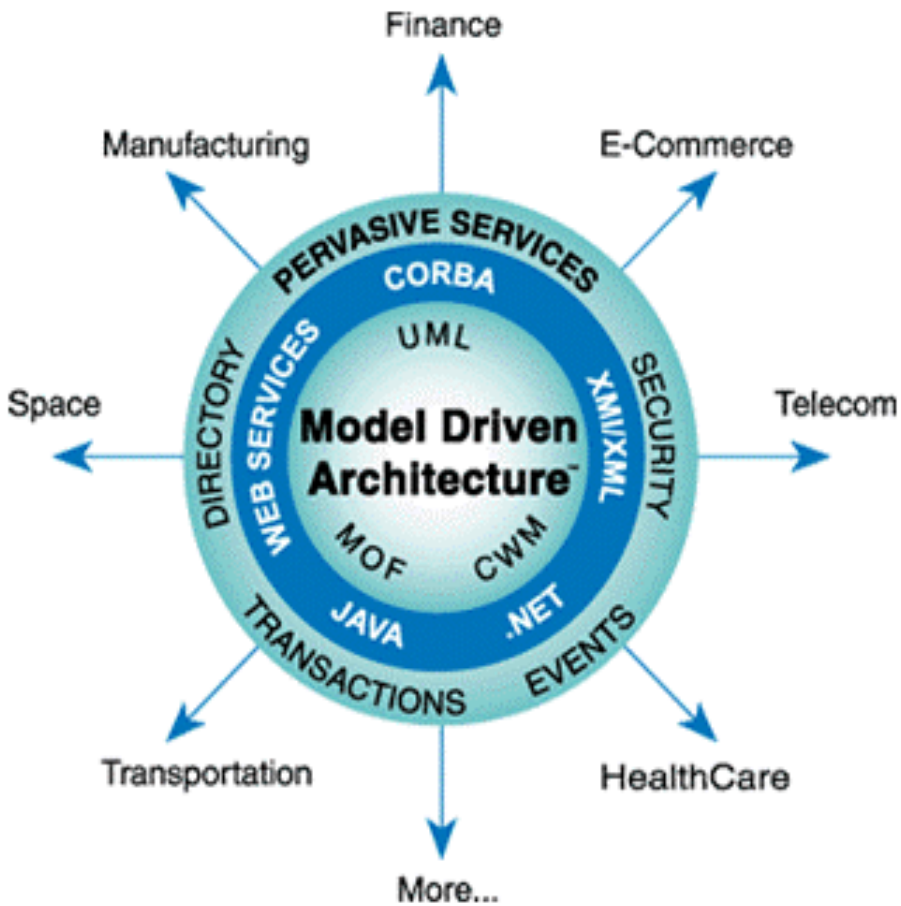
- A model is “driven” by an architecture. The return type is something valuable you produce, like a software system.
- An architecture “isDriven” or not, and is in public driven by a development approach.
- There is also a “real approach” for producing the software from the architecture which often diverges from theory. The disconnect represents a challenge.
- In the abstract, all agree that it is good to have a relevant model. The opposite, whimsically driven architecture, won’t work.

- Too often, MDA debates get into the devApproach, not MDA standards.



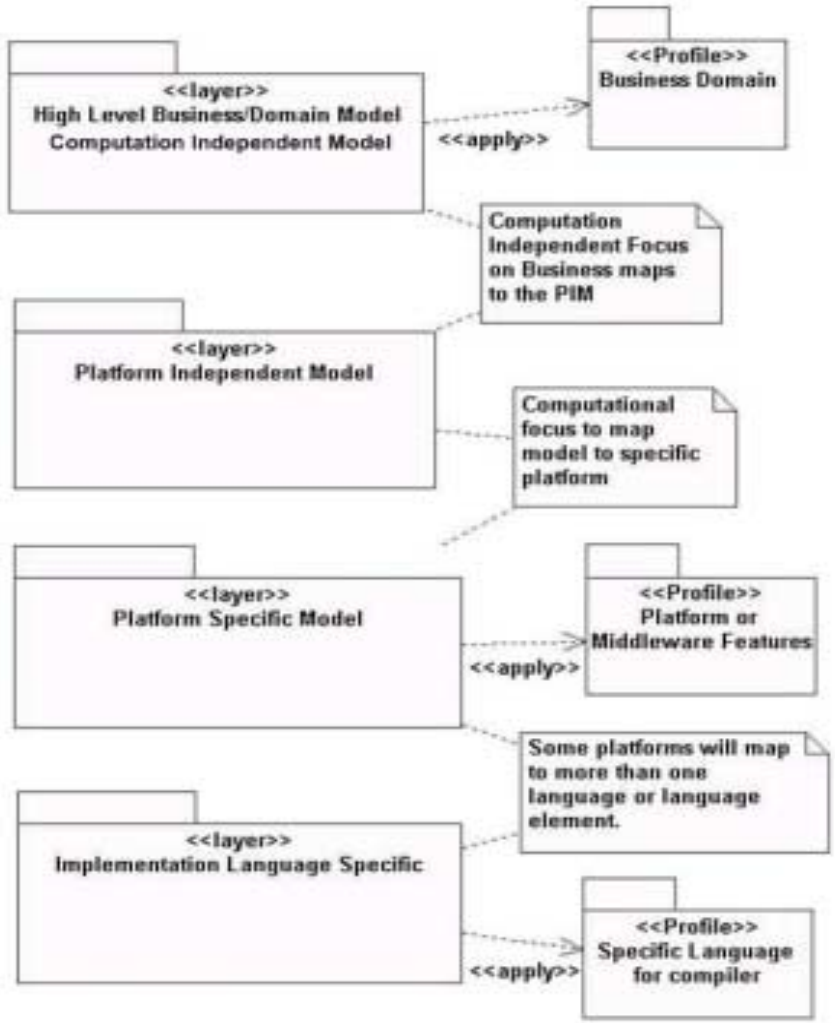
MDA: The OMG Logo/Model

- It goes in every direction.
- It shows layers in domains, languages, platforms, and modeling structures.
- Is it a navigation device, or a close-up picture of a weed spore that you can't get out of your clothes after hiking?
- The diagram shows an evolution building on successful tools in use.
- Here, we will focus on the use of the different tools in this toolkit.



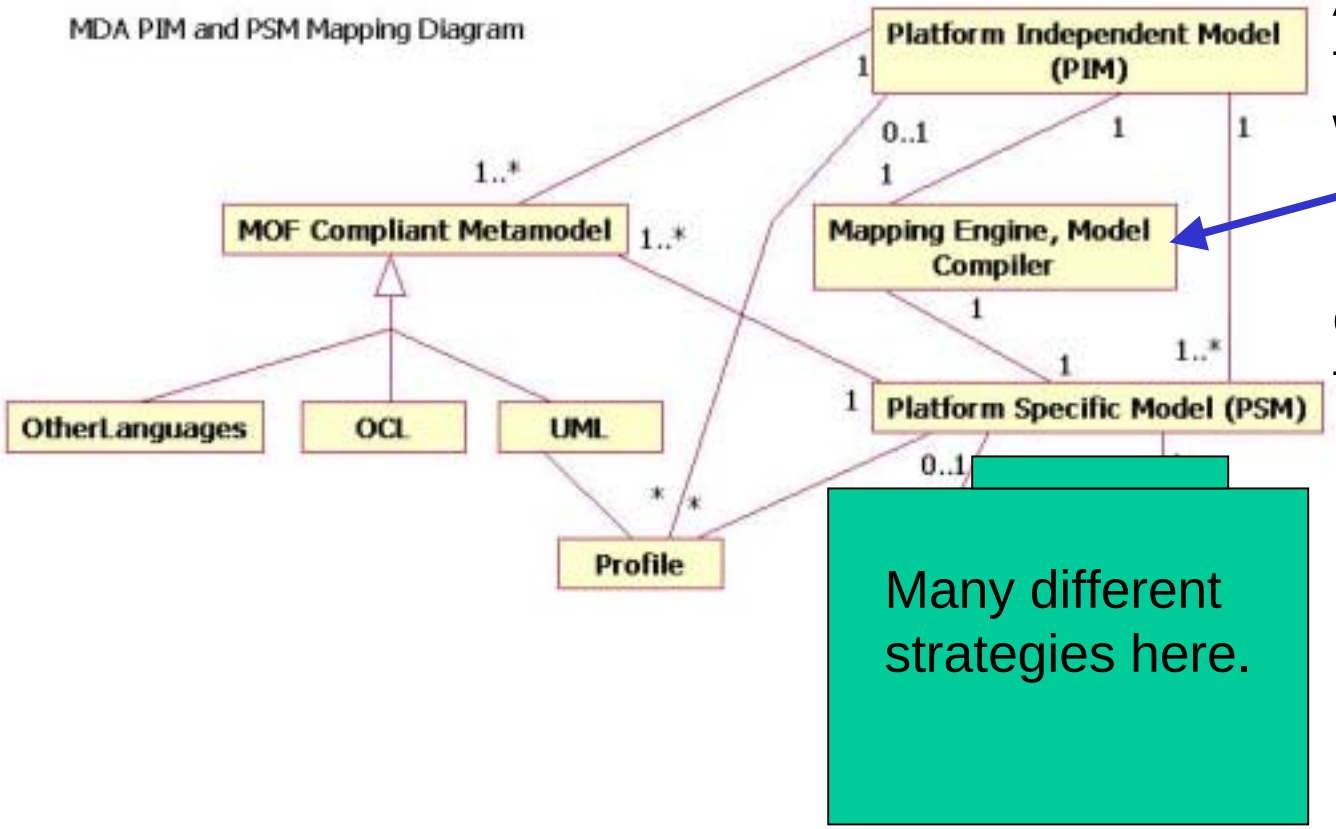
MDA Basics: Layers

- **Here are the basic layers for an organization using MDA.**
- **A key is the application and management of profiles, which was also an issue from UML 1.x.**
- **Between the layers, MDA provides specification to allow transformation of model information.**
- **What is wrong with this diagram?**



From Eriksson, Penker, Lyons, Fado, *UML 2 Toolkit,* OMG Press 2004.

MDA Basics: MOF as the Master



All transformations with the Mapping Engine will also produce a record of transformation.

Revised from Eriksson, Penker, Lyons, Fado, *UML 2 Toolkit,* OMG Press 2004.

MDA “Marking and Transformation” Strategies

From the MDA Guide, version 1.01

The guide outlines suggestive strategies and techniques for this process.

3.10.1 Marking

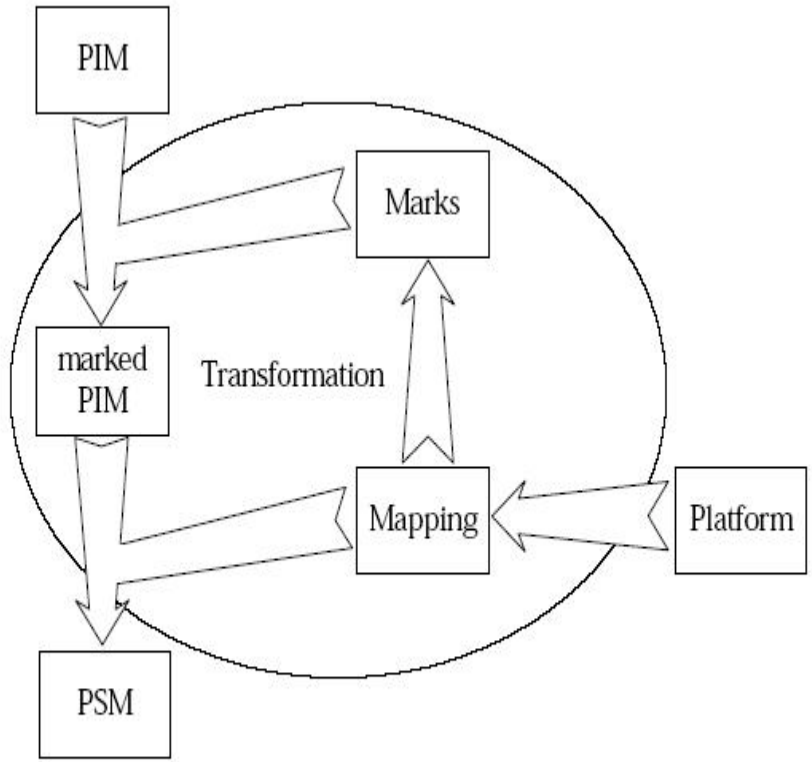


Figure 3-1 Marking a Model

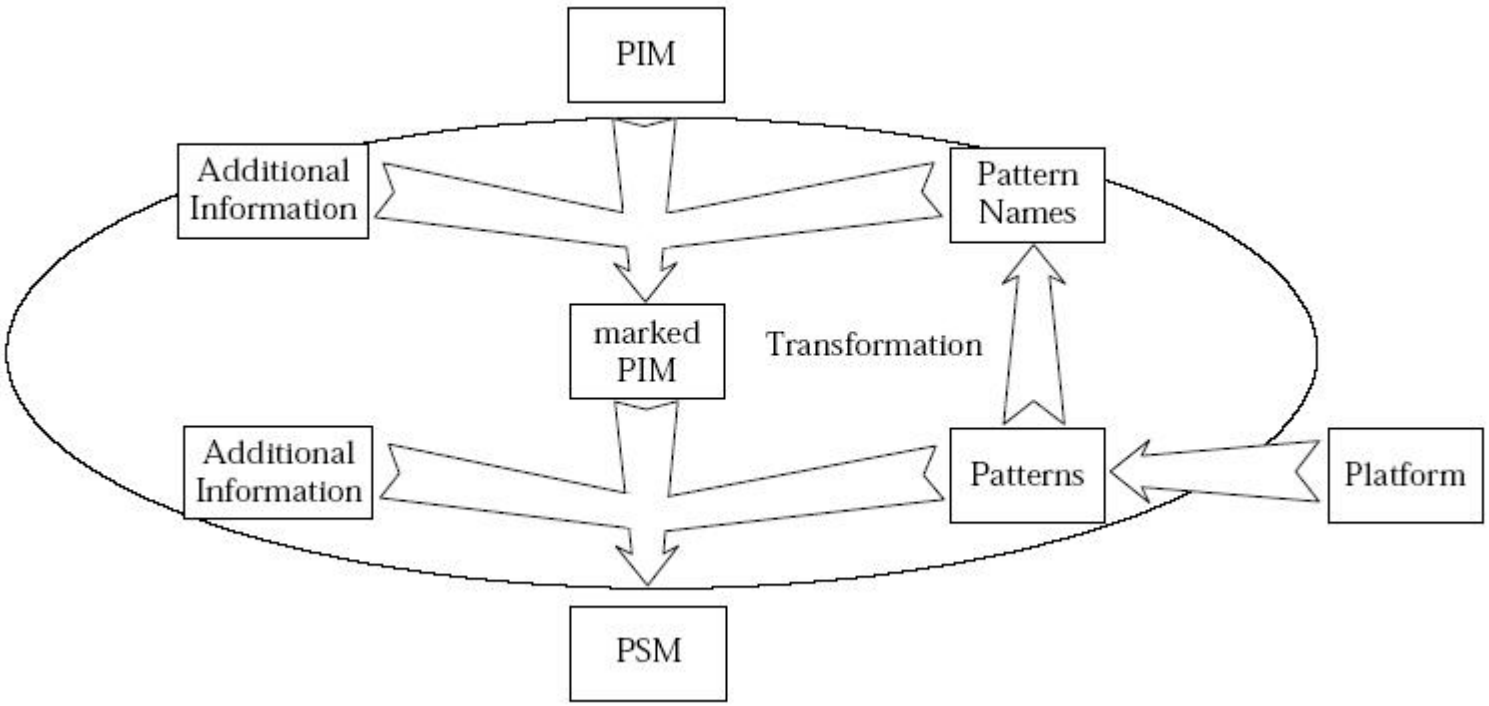
Some Suggestive Quotations from the MDA Guide

- The marks can be thought of as being applied to a transparent layer placed over the model.
- In order for marks to be properly used, they may need to be structured, constrained or modelled.
- A set of marks, instead of being supplied by a mapping, may be specified by a mark model, which is independent of any particular mapping.
- The next step is to take the marked PIM and transform it into a PSM. This can be done manually, with computer assistance, or automatically.
- *Platform independence* is a quality, which a model may exhibit. This is the quality that the model is independent of the features of a platform of any particular type. Like most qualities, platform independence is a matter of degree.

Transformation Approaches in MDA Guide

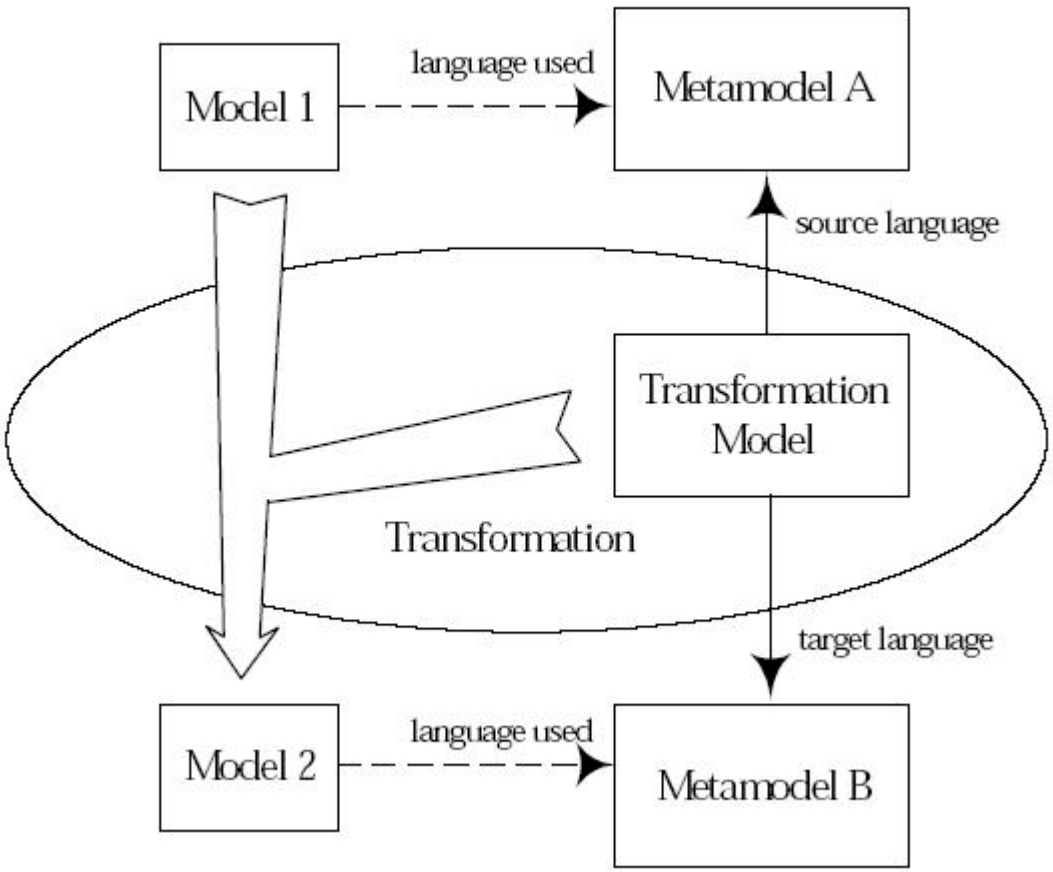
- **Marking**
- **Metamodel transformation with defined rules in a metamodel**
- **Model transformation with a transformation specification beteween types**
- **Pattern application**
- **Model merging**
- **Also, additional information can be added in for all of these approaches.**

Transformation with Additional Information



From MDA Guide version 1.0.1

Any Model Transformation is MDA: MDA Becomes General

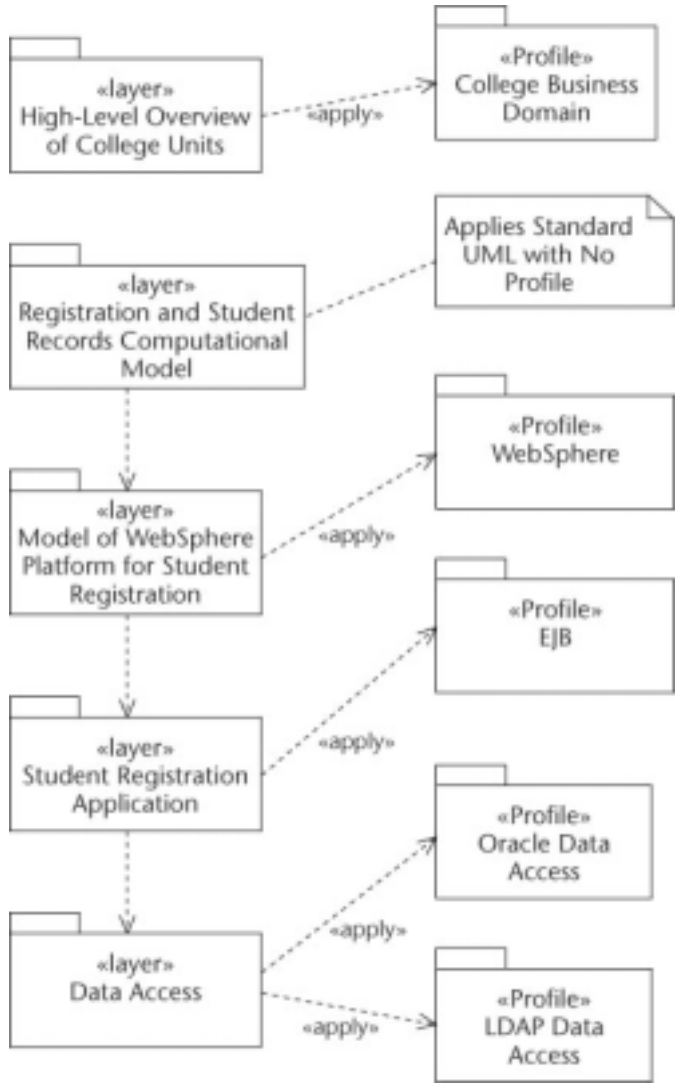


From here we will go into how to apply the powerful set of patterns in the MDA guide.

From MDA Guide version 1.0.1

Implementation: Linked Models Looking for Synchronization

- **Applying specific profiles.**
- **The transformation between the layers are a matter of much debate and focus.**
- **MDA provides a set of standards that allows the production of better tools to improve the functioning of these relationships.**
- **MDA here viewed as a toolkit: where in these relationships can it help?**



From Eriksson, Penker, Lyons, Fado, *UML 2 Toolkit,* OMG Press 2004.

To MDA from Issues Working with Software

- **The process activities MDA supports go on in projects and organizations anyway.**
- **MDA can be used to address issues with middleware complexity, application overlap, and inconsistent implementation of rules.**
- **MDA makes models useful for handling technological change.**
- **Executable Models can make communication more precise and automate many relationships.**
- **MDA provides answers to problems applying UML**
  - Proliferation of Profiles
  - Domain models that cross projects
  - Communication between different UML tools
  - Tendency for many artifacts to become Powerpoint slides.

MDA Implications

- **When a Model has been expressed precisely and integrated with the implementation, the total of all the model parts is much more valuable than the sum of the models.**
  - An enterprise view emerges.
  - A common form for communication and collaboration is in place.
  - Models and model elements become reusable components.
- **MDA embodies standards and best practices that have made software the success story of the late twentieth century.**
  - Distributed collaboration
  - Responsiveness to results
  - Minimizing dependencies on a specific technology

Two Aspects of the Revolution: MDA and UML 2

- **UML 2 Provides Precision**
  - Validation of tool compliance and clear mechanism for sharing.
  - Precision and revisions to clarify specification delays tools.
  - Foundation for professionalizing modeling: certification and training.
- **MDA Provides Vision, Meaning, and Directs Innovation**
  - General framework to support many specifications, not a single specification
  - Compliance not meant to be difficult
  - Allows for cooperation to make models more executable
- **UML 2 and MDA Work Together**
  - Latest UML tools (UML 2 Friendly) make more sense with MDA
  - MDA helps to explain why elements of OOAD appear less important in UML 2: greater flexibility enhances models, not replacing UML 1 approaches to modeling.

Audience Poll: Is it MDA?

- **Your team produces a data model to store information about recipes for use at a cooking school.**
- **Your team uses a domain model from the health care industry to implement a patient record system that complies with federal regulations.**
- **Your team produces an application to manage information about early childhood education and reverse engineers the code into a model.**
- **You use a tool the shows traces of software activity in a sequence diagram.**
- **You have captured the main business rules in activity diagrams and provided information to automatically transfer the information into Java.**
- **You reverse engineer an existing weapon control system into a precise UML model then use that UML model to rebuild the system on a new platform.**

End of SECTION ONE

- **Questions**
- **Hand-out of worksheet**
- **BREAK**

MDA Worksheet

Review of the MDA Toolkit: A Worksheet to Place MDA Implementations Presented During the Week

Note: The possible diagrams are suggested only and NOT limited to UML. They are mapped loosely to the MDA model levels. The goal is to review how you use models and then think of ways model transformation and management can improve projects.

Project Life Cycle Element	Possible Diagrams and Models	MDA Examples
Initial Project Justification	**CIM: Domain Model, Business Model**	
Requirements	**CIM: Use Case Diagrams, Activity Diagrams, Domain Model.**	
Elaboration and Design	**PIM: State Machines, Sequence Diagrams, Activity Diagrams, Class Diagrams, E-R Diagrams.**	
Technical Decision Making`	**PIM/PSM: Business Model, Domain Model, Deployment Diagram.**	
Construction	**PSM: Class Diagrams, Sequence Diagrams, IDEs.**	
Testing	**PIM/PSM: Sequence Diagrams, Activity Diagrams**	
Deployment	**PSM: Deployment Diagrams**	
Project Assessment and Project Flexibility	**CIM/PIM/PSM: Component Diagrams, Use Case Diagrams**	
Maintenance	**PSM: Class Diagrams, Deployment Diagrams**	

Section Two: Using the MDA Toolkit for Projects

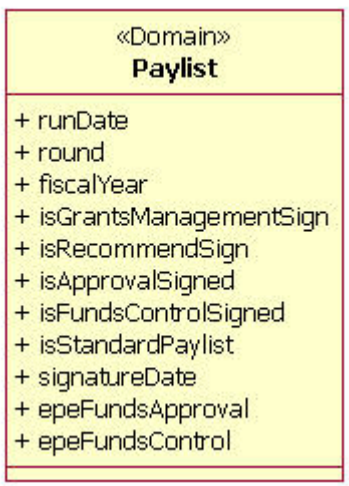
- **MDA provides a rich set of features that apply across a software projects.**
- **When faced with implementation, many can use some of these features but not all, depending on circumstances.**
- **This section will review the use of modeling for activities related to typical software projects.**
- **Activities also include more general support for software development.**
- **We will look for examples of model success as well as examples of model impotence.**
- **As many of the examples here are modified from real projects, we have instances of model impotence.**

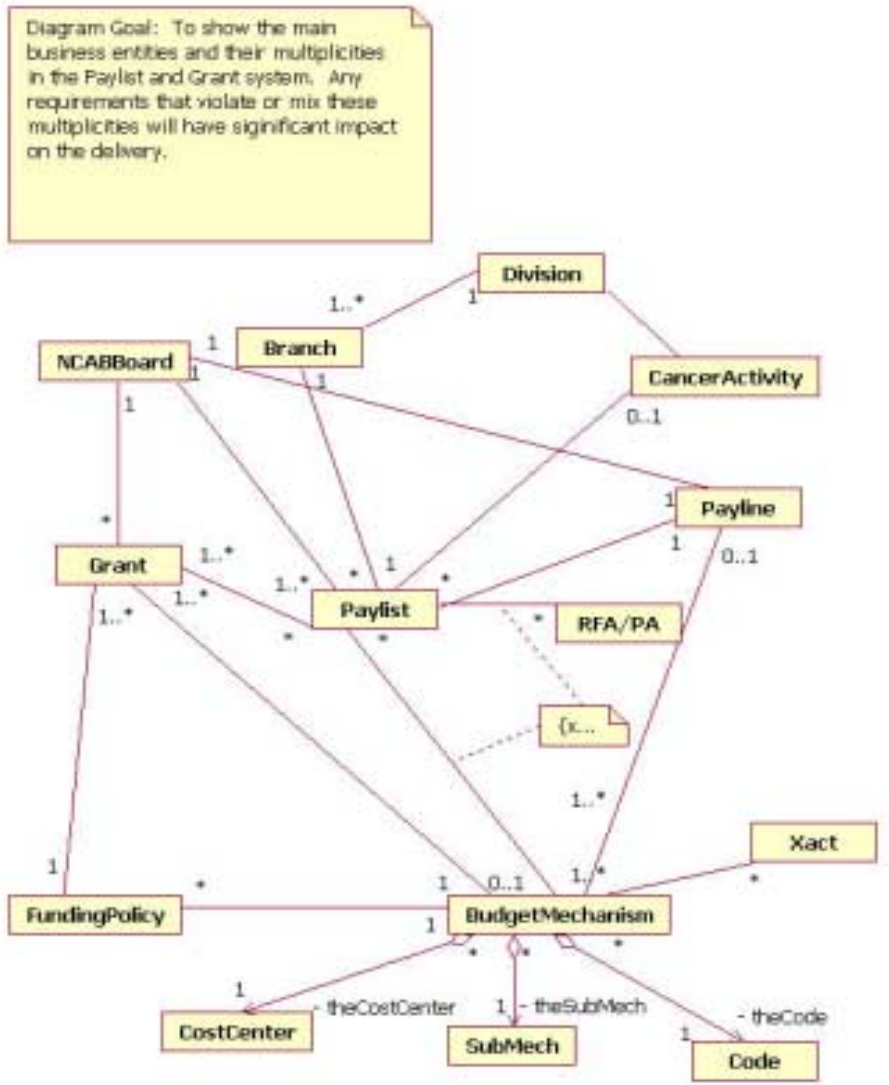
Initial Project Justification

- **Software or any technology should address a need. This point informs many enterprise architecture frameworks, such as DODAF.**
- **Project justification should not jump directly into technical platform discussions.**
- **A business model can provide high-level assessment of needs, in whatever modeling form works.**
- **A business model that can adjust to changing market conditions is more likely to support healthy growth.**
- **Use cases, Domain model, activities can all pin-point areas for technological change.**

Initial Project Justification: Domain Example

The attributes of these classes could be reused across all projects and the multiplicities will constrain design. So, all projects will mean the same thing by a “paylist” and can be notified of a change.





Initial Project Justification: Use Case Example

- **Is this MDA?**
- **What would make it MDA?**

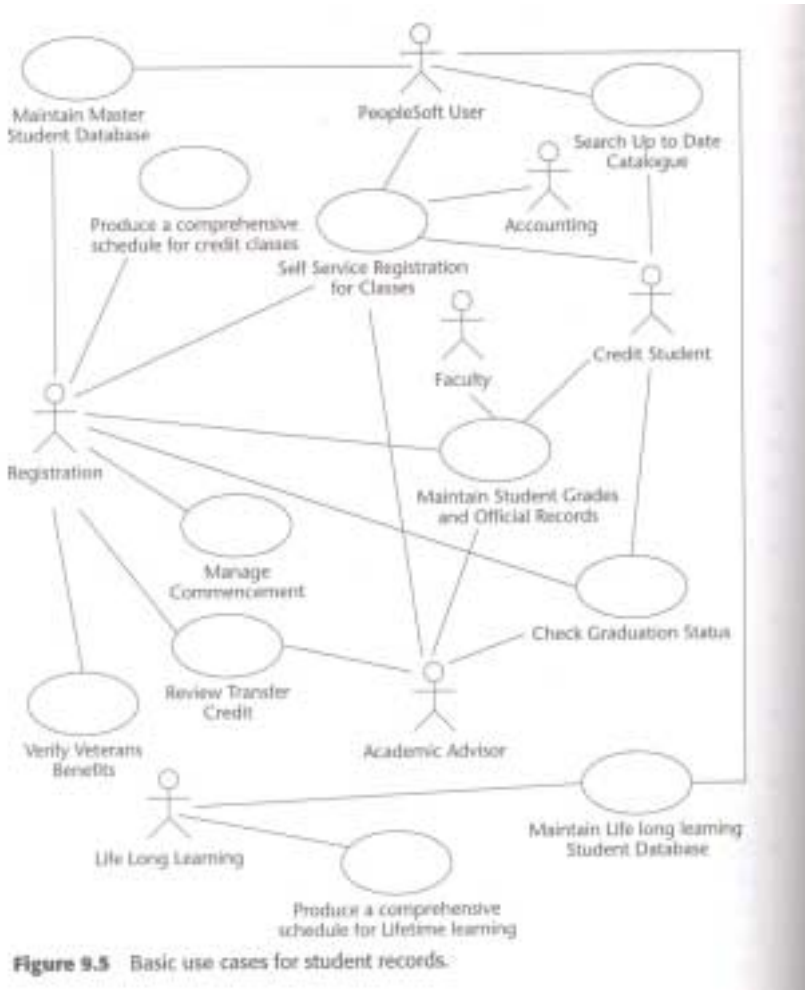


Figure 9.5 Basic use cases for student records.

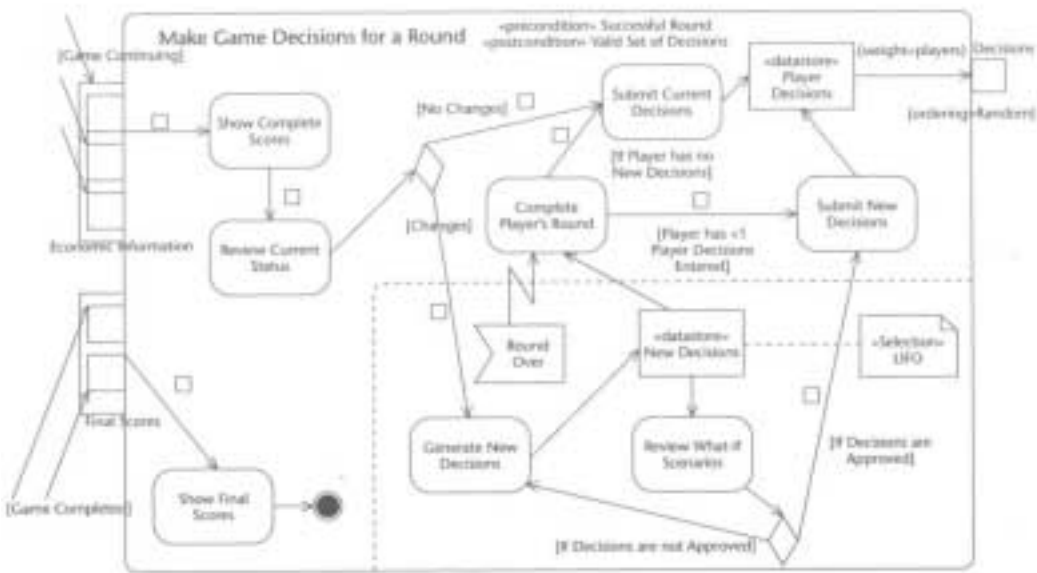
From Eriksson, Penker, Lyons, Fado, *UML 2 Toolkit,* OMG Press 2004.

Requirements

- **A project needs a form for capturing how to measure success.**
- **Often, unwritten requirements drive success as teams encouraged by the environment to “optimize” along a specific variable.**
- **Full model-driven requirements is not always possible, but there should be clear ways of moving between layers.**
- **More detailed use cases along with related activity diagrams can show the requirements.**
- **Usability, both from a machine and from users, represents a difficult thing to automate. Automate the ability to change the front end and work with other tools, and this will contribute to these requirements that are difficult to specify precisely.**

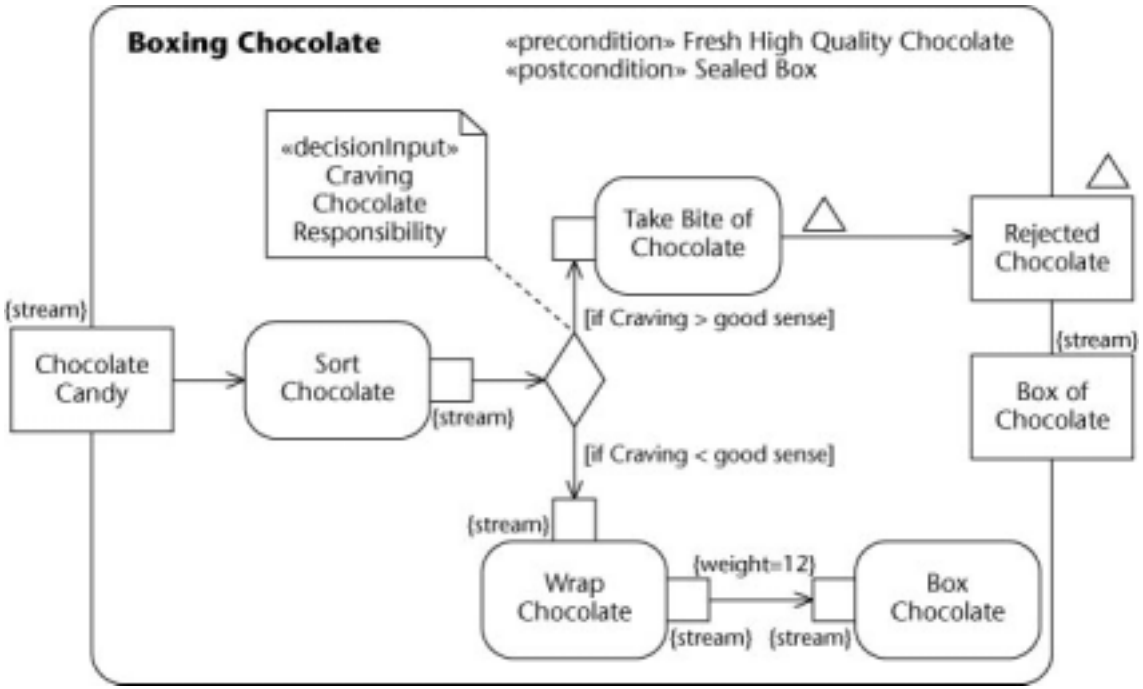
Requirements Elaboration: Activity Diagram

- **Activity Diagram from www.probanker.com**



From Eriksson, Penker, Lyons, Fado, *UML 2 Toolkit,* OMG Press 2004.

- **Token flow model borrows from Petri nets to show flow of a system.**
- **Useful for clarifying the CIM or PIM level of modeling.**



From Eriksson, Penker, Lyons, Fado, *UML 2 Toolkit,* OMG Press 2004.

Technical Decision Making

- **Often technology products will require decisions regarding platforms.**
- **Understanding the needs and long term goals of a system will provide a needed input into deciding on a platform.**
- **Projects that do not have a technical platform decided require very different management and communication strategies.**
- **Those that have an assumed technology platform run into a higher risk of not keeping a PIM separate from a PSM.**
- **Section three includes some suggestions on decision support.**

Elaboration and Design

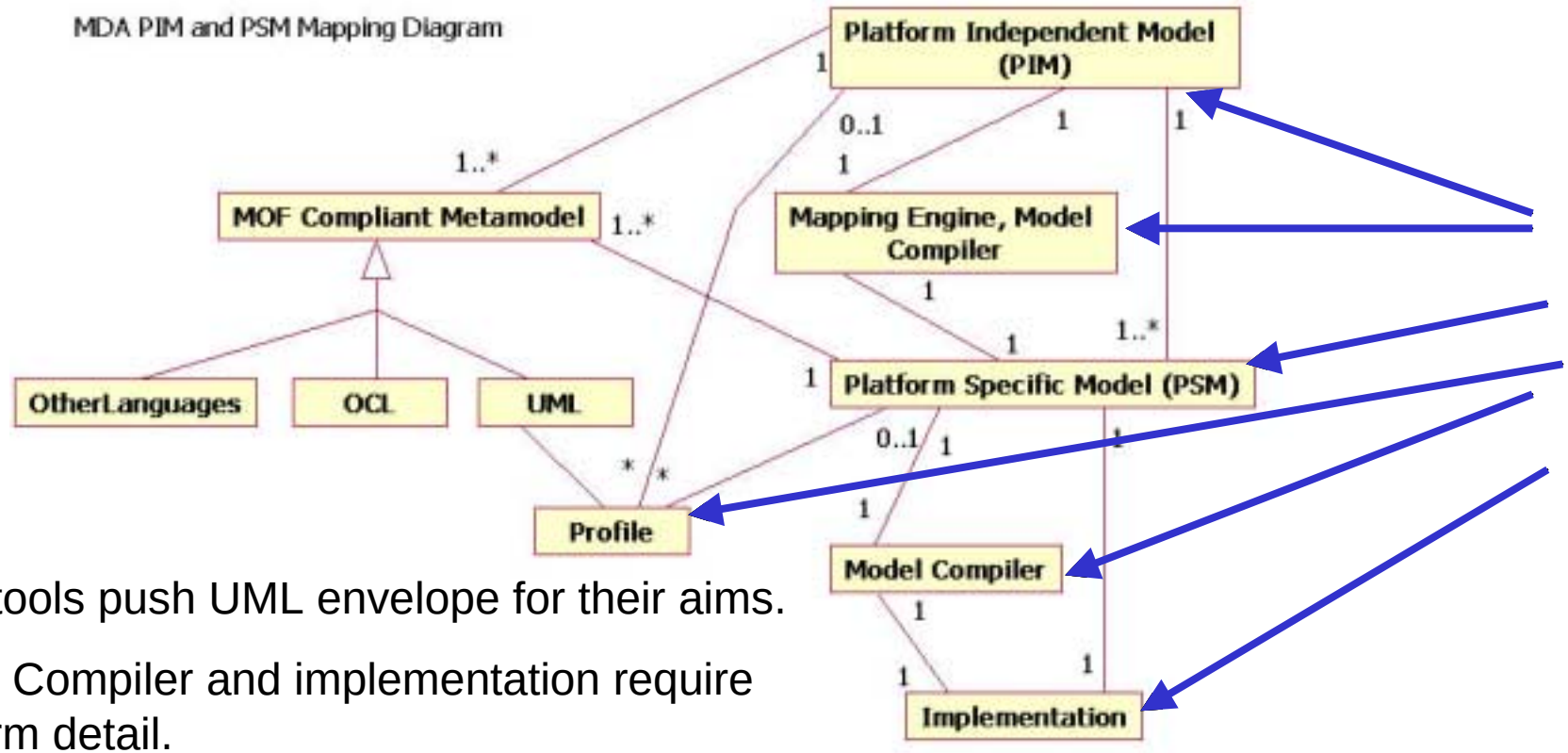
- **The “and” here hides a fundamental tension: at what point do you stop elaborating, and start working on an actual system on a real platform.**
- **A useful way of separating these two, although this may be a “drifting ontology”**
  - Elaboration involves computational definition without regard to platform
  - Design often involves a knowledge of the deployment platform, or at least an understanding of how to map to that platform.
- **MDA makes this balance more difficult to manage, because the modeling and design tool will also be the tool used for working with code, making the separation between design and construction harder to maintain.**
- **My preference, and I think the preference of MDA, is to keep the PIM and PSM elements of this activity distinct but both usable and synchronized. However, the PIM and PSM easily mix, leading perhaps to “model pollution” or an opportunity to use automated tools more effectively.**

MDA Elaboration and Design as Compression

- **MDA literature emphasizes value of abstraction as driving force in information technology efficiencies.**
- **It is worth emphasizing that models seek to compress information, rather than abstract away detail.**
- **The goal is to move from a model element down to the software language implementation to the location of that element on a deployed system.**
- **So, if in design I can go straight into a framework that provides the needed element of compression, I will do it: for a "struts" application I will go straight to implementation elements early in the design, with clarity smoothing a later port if that is needed.**
- **What is emerging is a marketplace of models where you can use enterprise assets as well as software tools to make models clearer.**

MDA Elaboration and Design: Return to Core Concepts

Most MDA tools provide the six elements in an easy to use package. Many diagrams go into these models and there will be variations between projects. Note the MDA tool is not focused on UML or MOF.



MDA tools push UML envelope for their aims.

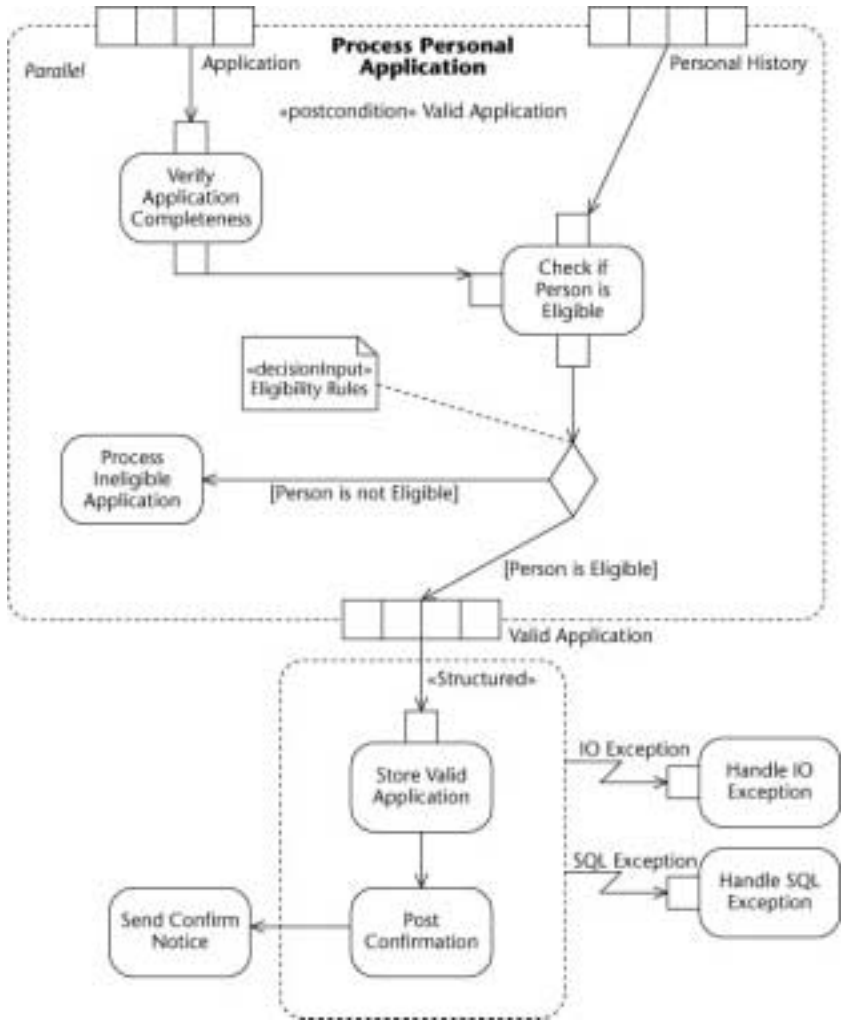
Model Compiler and implementation require platform detail.

Most “MDA Tools” focus on a technology.

From Eriksson, Penker, Lyons, Fado, *UML 2 Toolkit,* OMG Press 2004.

Elaboration: Activity Diagrams

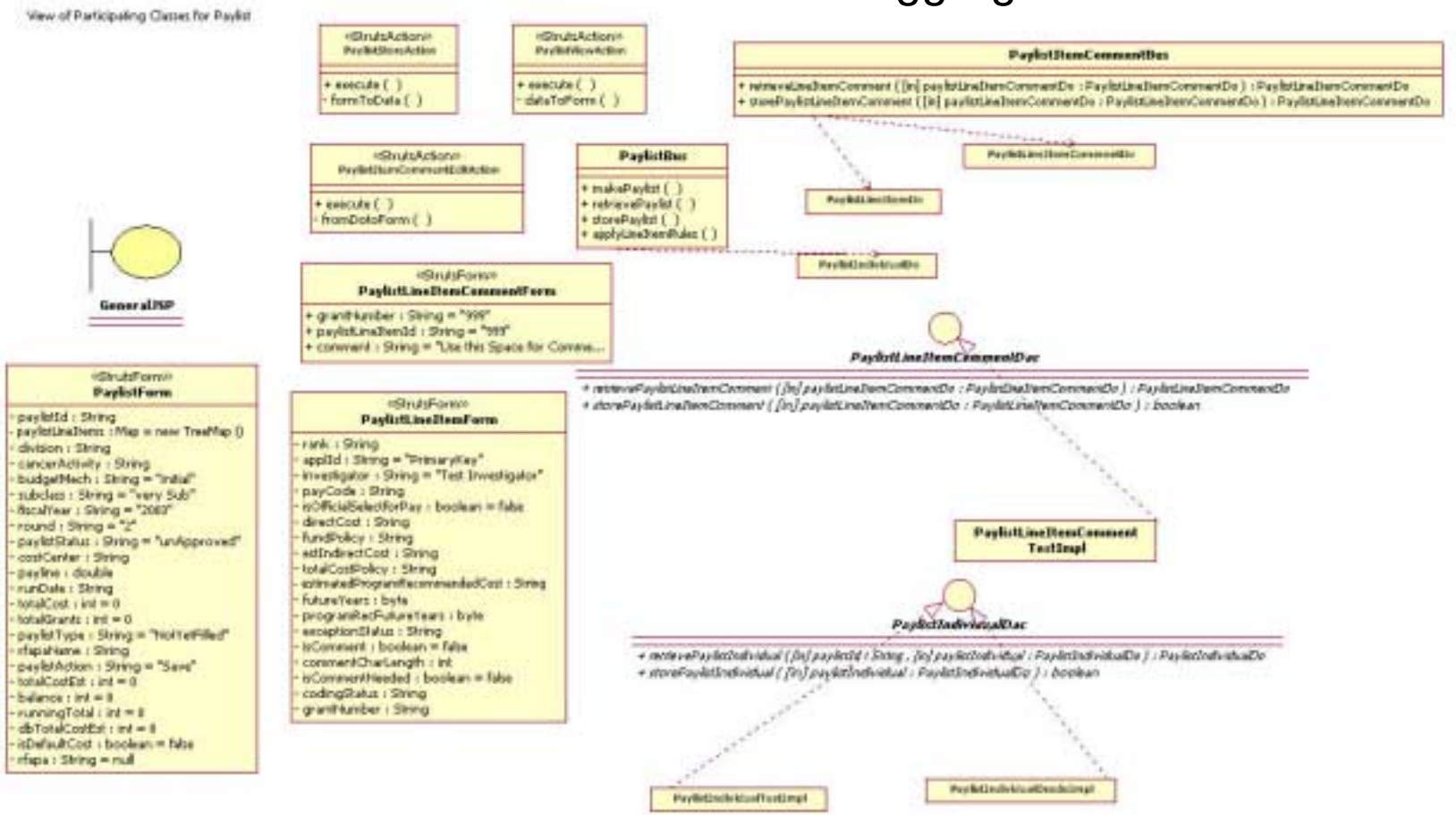
- **Activity Diagram shows collections, exceptions, areas with error handling, etc.**
- **Note the detail on the bottom includes platform specific information.**



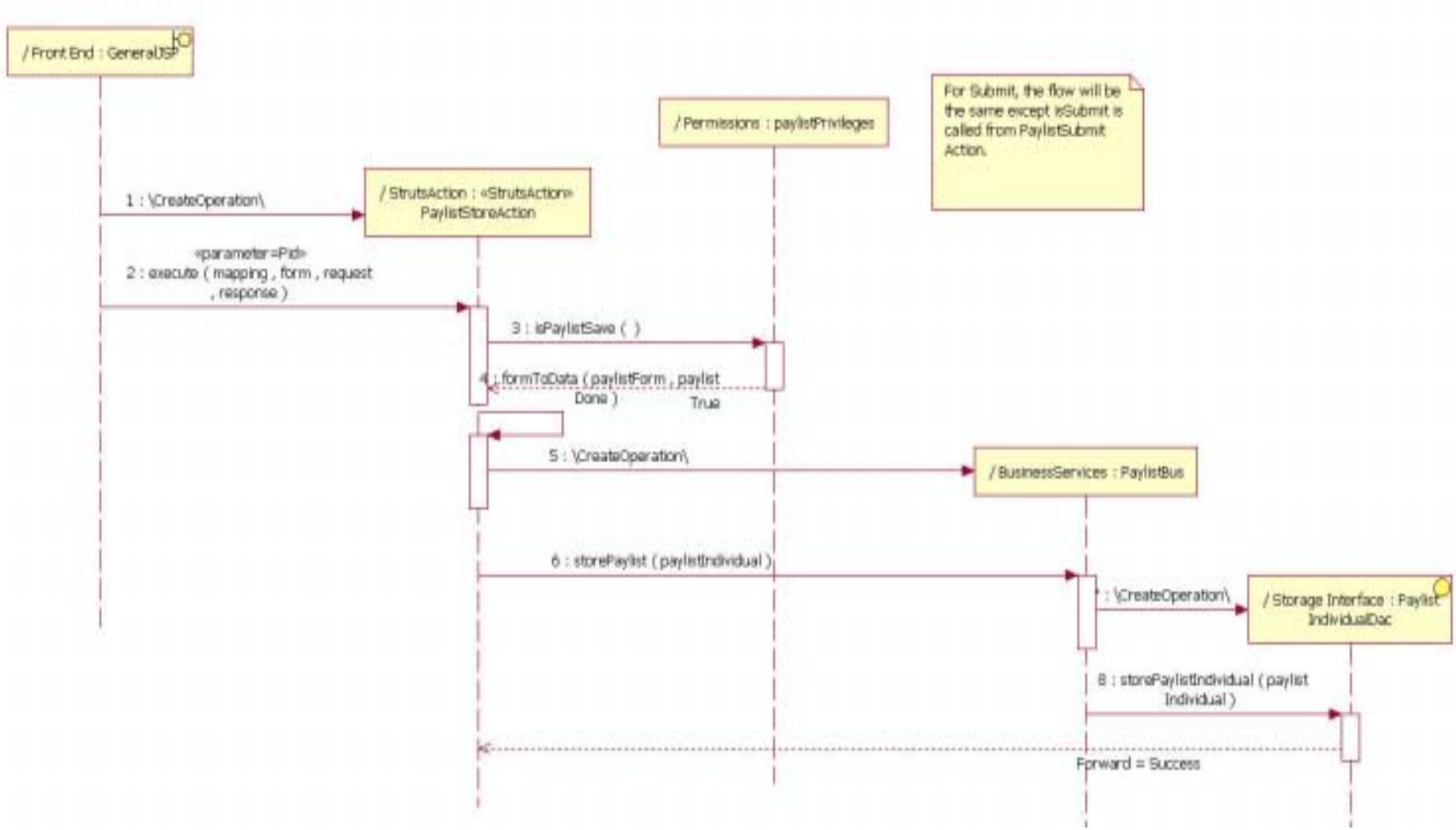
From Eriksson, Penker, Lyons, Fado, *UML 2 Toolkit,* OMG Press 2004.

VOPC: Design and Elaboration and Code View Example

Some features of this diagram are different because it is synchronized with the code: interfaces are Java interfaces and aggregations are not shown.



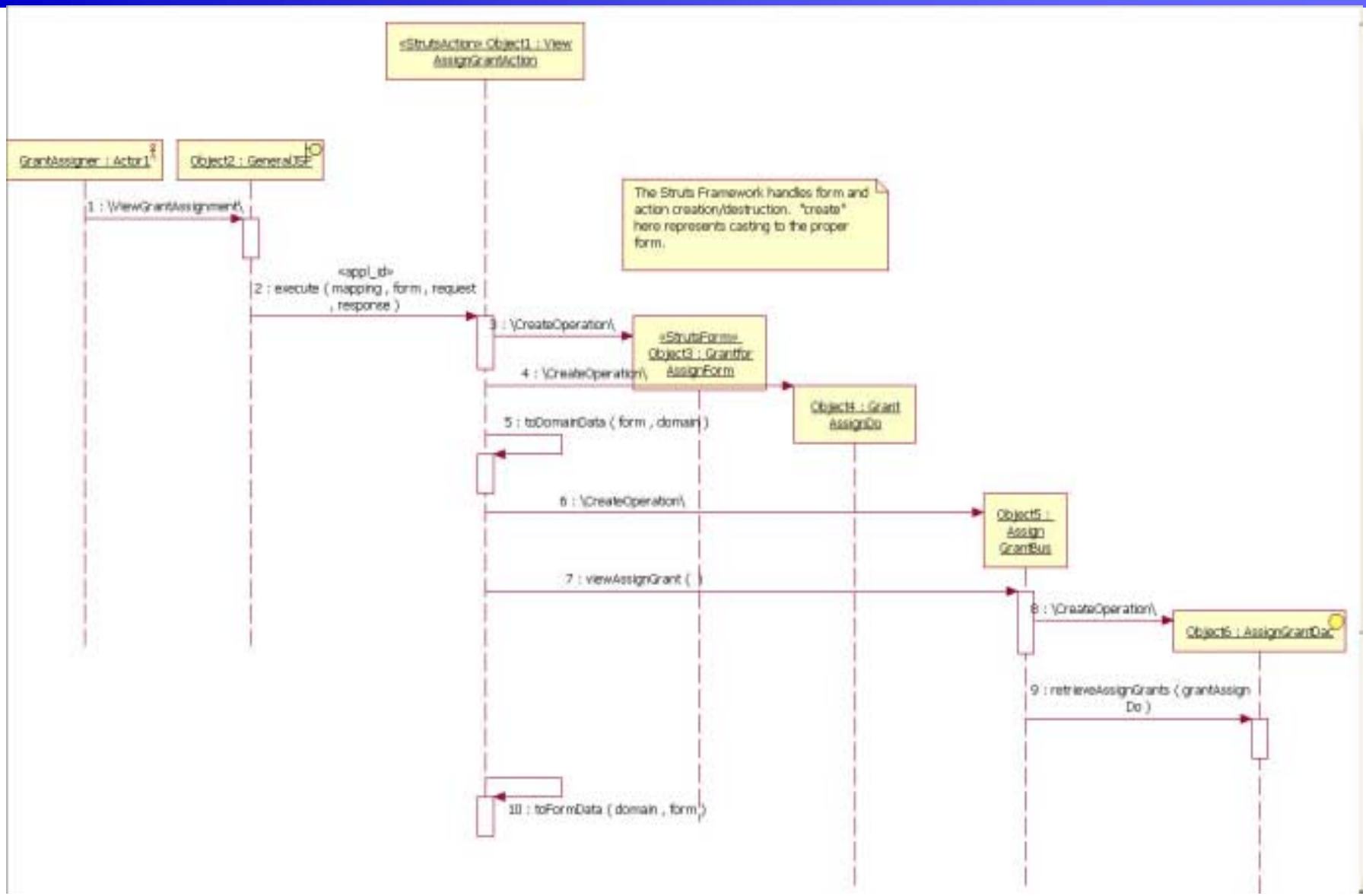
General Pattern: Design and Elaboration Example



Construction

- MDA offers a compression of the construction effort.
- From a project task point of view, this is perhaps misleading, as construction effort simply transformed to other tools. I am seeing this emergent behavior: construction activities will use the same tools and some of the modeling constructs to produce software.
- User specific items, such as front ends, will still require work.
- MDA offers a synchronization of the construction code with the model, important in keeping the model as the living representation of the system.
- MDA tools on my small projects were more useful in maintaining the accuracy of the model after coding exercises. Other projects definitely tell a different story.
- For the toolkit, MDA standards are still useful as a way of enforcing best standards, doing code reviews, etc.

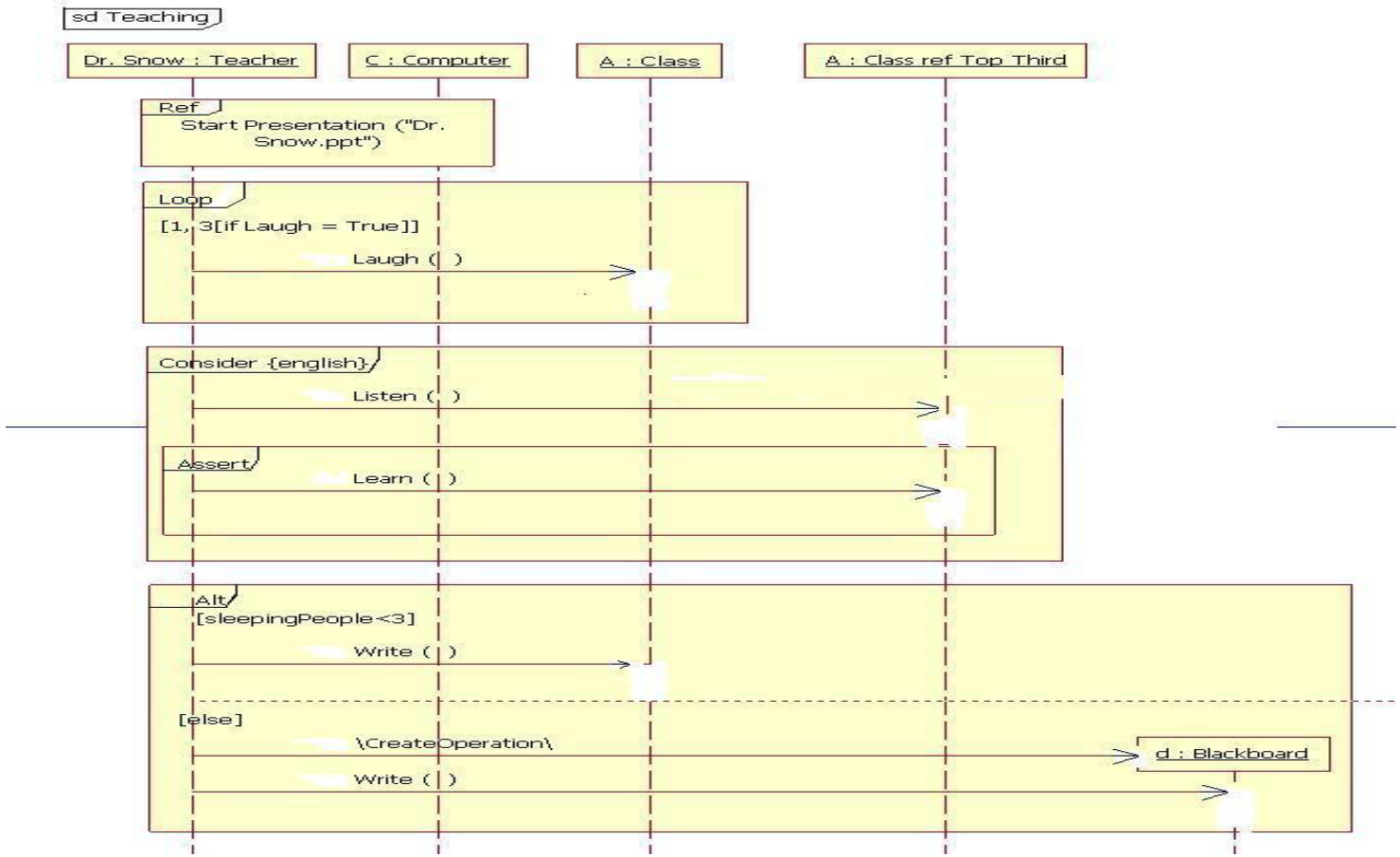
Specific Example: Sequence Diagram



- **Model driven testing comes with automated test suites.**
- **Standard patterns in test construction should make this an area of great valued added.**
- **Using models for simulation also acts as a test and can catch issues very early.**
- **UML 2 sequence diagrams include more features for these interactions to function as a test.**

UML 2 Sequence Diagram

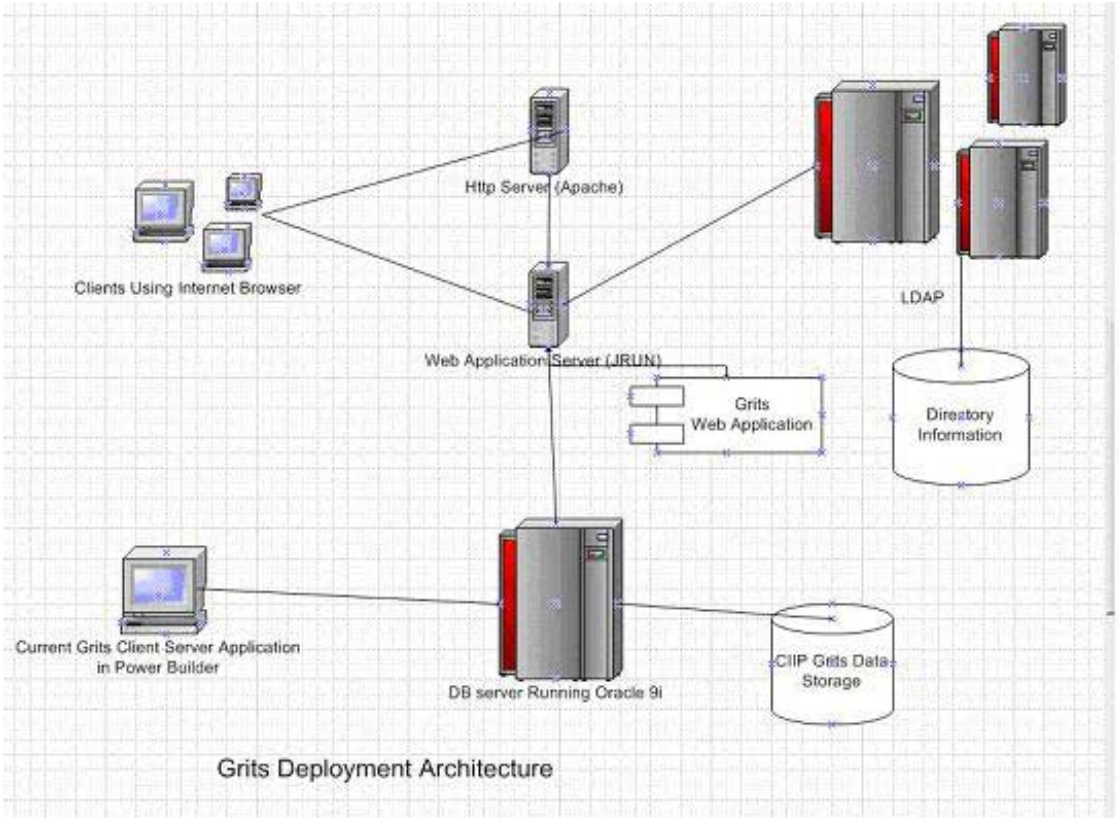
- **Assertions**
- **Loops**
- **Alternate**
- **Fragments**



Deployment

- **Placement of items in their environment helpful.**
- **As the MDA guide states, many platforms now present their own standardized view, helping make deployment much easier.**
- **War files, for example, now are much easier to deploy to different containers. Modeling will make this easier in the future.**
- **Example: Web Sphere studio and XDE.**
- **Many tool suites offer full deployment support tailored to a specific platform. Under what circumstances is that MDA? Probably when they use a common, non-proprietary standard.**

A Non-MDA Deployment Diagram

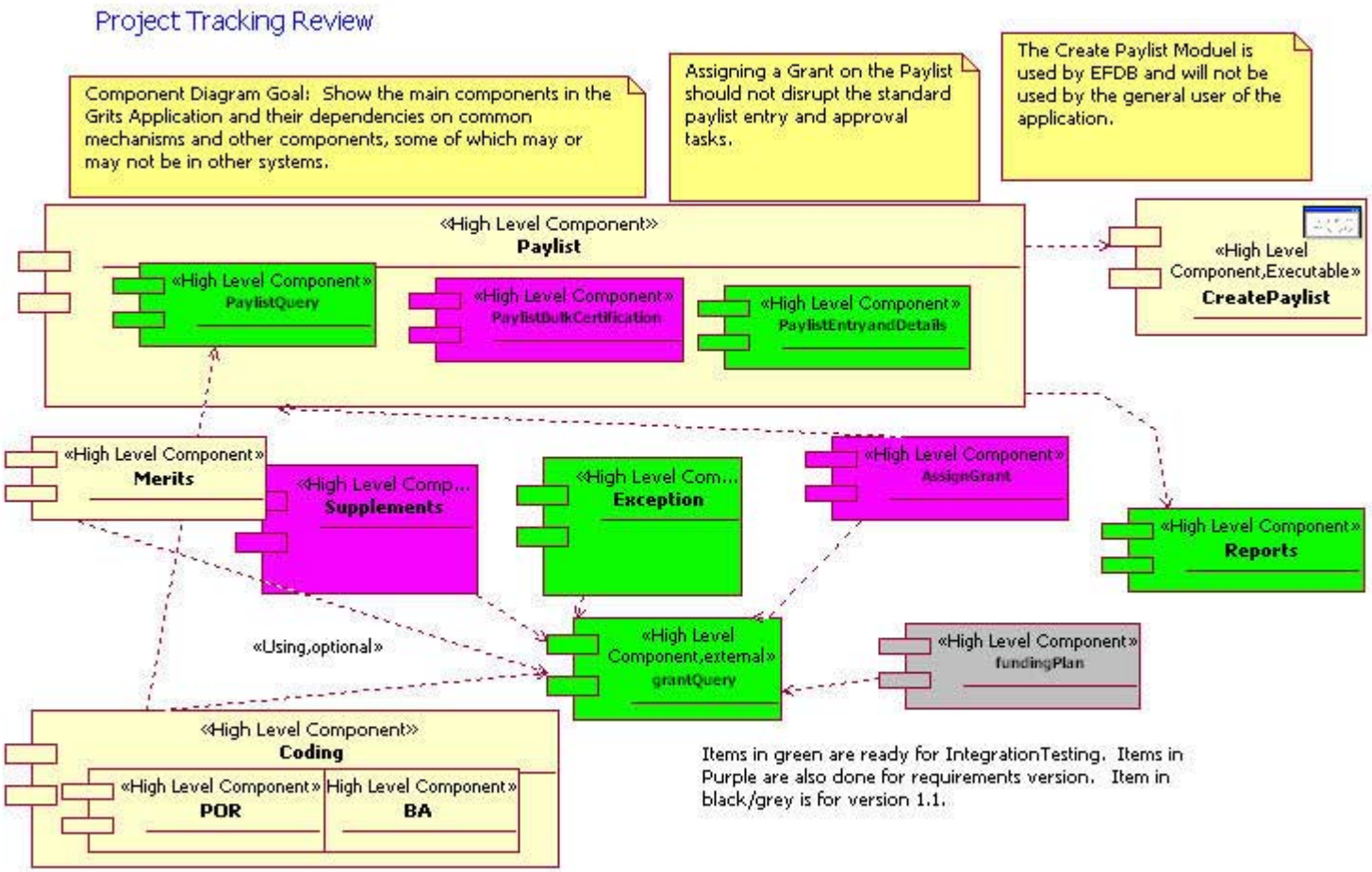


This classic “PowerPoint” level impotent model communicated but there is no way to manage it inside a model structure.

Project Assessment and Project Flexibility

- As circumstances change, projects should come under review.
- As estimates become more reliable, the cost may drift to an unreasonable level.
- Technical issues may arise during the project.
- MDA offers the possibility the high level models will remain current.
- Achieving this flexibility will require the overhead of enterprise configuration management and model management standards.
- Look at model diagrams as user interfaces and use features that communicate: translate a real diagram into PowerPoint Mode to express status, as shown on the next slide.
- Be willing to depart from automated support if the communication goals of the project require it. You can generally automate features later.
- Conditional color based on UML or modeling features would be a great thing in tools: maybe it is already done somewhere?

Component Tracking Diagram with Non-Standard Usage



These components are in spirit UML 2 variants, but no tool support yet for this concept nor for the helpful port and required/provided interfaces.

- **To the extent MDA focuses on better, faster, cheaper on a single project, the long term benefits for maintaining assets is obscured.**
- **Using standard platform profiles and standard enterprise modeling elements, assessing the impact of migration is made much easier.**
- **The ability to move from a PIM to a PSM with markings can also ease an effort to move from one platform to another.**
- **Maintenance will use all the models already discussed, but requires a standard versioning system and a standard approach to achieve reusability.**

End of Section Two

- **Questions?**
- **Break: find teams of about five to get into for the next section.**

- **Distinguish MDA's value from the value of modeling.**
  - Key of standards for reusability
  - Key of accepted basis for sharing model
  - Key of accepted form for tool production
- **Distinguish MDA's value to the enterprise from the value to a specific project.**
  - Such a distinction is difficult on a pilot project.
  - Value must be recognized by senior management.

Audience Team Exercise: How will MDA help You?

- **What are the two main benefits you hope to gain or have gained from MDA based on your implementation experience?**
- **What are the two biggest pitfalls you watch out for with MDA based on your implementation experience?**

- **Enabling the Enterprise View.**
  - As we have reviewed the MDA toolkit, many of these tools can help out individual projects or maintenance of one system.
  - MDA emphasis on precision results in an enterprise view, which requires a basic set of standards for managing.
  - Models that capture information about important enterprise assets WILL be of interest to important decision-makers.
- **Let Modeling Respond to Circumstances**
  - Models should be flexible enough to allow for different front ends and different communication styles while maintaining automated connections.
  - Underlying standards and machine communication can keep these differences from turning into confusion or a replay of the method wars.

My Issues of Concern

- **Standards can't keep up with the tools**
  - MDA becomes more focused on a specific instance and technology and is not applied across an enterprise.
  - MDA tools become another layer of middleware and legacy confusion.
- **UML 2 and other standards not organized by MDA**
  - MDA fails to organize the many OMG standards and these standards become yet another layer of confusion.
  - UML 2 fails to provide efficient enough validation to enable full model sharing.

Audience Poll: MDA Constructive Implementation

- **What are the important Issues you hope to discuss this week?**
- **What can we focus on to make MDA an answer to the problem of middleware complexity and model impotence?**
- **Any other questions or issues of concern?**

Time Check

- **If time permits, we will review the slides on MDA as an enterprise support tool.**

THANK YOU

For questions or comments:

fadod@saic.com **or** dphaedo@yahoo.com**.**