



Unit 21: Model Driven Architecture
Advanced Software Engineering
(3C05/D22 2003/04)

Presented by Deniz Özsen
Email: d.ozsen@cs.ucl.ac.uk
Web: <http://www.cs.ucl.ac.uk/students/D.Ozsen>



Overview


- Problems that have lead to the development of MDA
- How these problems were addressed in conventional Software processes before MDA
- Model Driven Architecture: a new approach
- Conclusions
- Acknowledgements and references



Problems in Software Engineering...

...which arise due to the nature of the subject:

- Platform/Technology dependence
 - OS: Windows, Dos, Unix etc.
 - Language: C, C++, Java, C#
- Rapidly changing technology
 - Software (e.g. Windows Versions)
 - Hardware (processors, graphics cards etc.)
- Attention is split between functionality of system and technical implementation issues



Platform Independence...

...can be achieved using:

- Java (platform independent bytecode)
- Compiler directives in C++ (code on the right)
- Platform independent libraries


The latter two involve a lot of work! And even Java has some drawbacks.

```

#define WINDOWS
#ifdef WINDOWS
    // Windows specific code
#endif
#ifdef LINUX
    // Linux specific code
#endif
// ...

```


Only need to modify #define directive to compile on a different platform



Keeping up with technology...


...is quite difficult. But we can isolate the technology dependent parts of the system and have them in just one place, thus minimizing the number of places where changes will be necessary:

- Create abstraction layers (e.g. Interfaces in Java)
 - BUT: It's likely that we'll still have to rewrite some code. And our system, as is, might just not work with a new piece of technology.
- Alternatively: Don't use technology
 - Not an attractive alternative



How to focus on the problem domain?

- Component-based Software Engineering
 - BUT: Components still have underlying platform-, language- and possibly technology-dependent source code!
- Any thoughts / suggestions?



Introducing Model Driven Architecture I

- Model Driven Architecture (MDA) developed by the Object Management Group (<http://www.omg.org>) a few years ago
- Some descriptions from the OMG's website:
 - "Application architecture for modeling first, mapping later"
 - "Software architecture to connect what already has been built with what will be built in the future"
- Idea is to focus first on the functionality and behavior of an application or system without worrying about the technologies that will be used to implement it
- At the time of release the system specification is mapped to the supported platform automatically using tools.

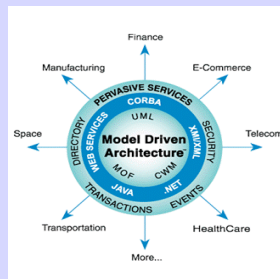


UCL

ComputerScience

Introducing Model Driven Architecture II

- MDA is Built on CORBA (Common Object Request Broker Architecture) and consists of three core technologies:
 - UML: Unified Modeling Language
 - MOF: Meta Object Facility
 - CWM: Common Warehouse Metamodel



<http://www.omg.org/mda>



UCL

ComputerScience

Models

What is a model?

- An object that represents another object, leaving out unnecessary detail
 - Concrete: e.g. a model aero plane
 - Abstract: e.g. aero plane design
- In the Software Engineering context: System design, ie the information needed to build the actual system



UCL

ComputerScience

MDA Concepts: PIM and PSM

© 2002 Clear View Training Limited, see <http://www.cs.ucl.ac.uk/staff/c.mascolo/www/arlou.pdf>
<http://www.clearviewtraining.com>

- Can use an MDA enabled tool to apply a standard mapping in order to generate a PSM from the PIM
- The MDA tool will also generate most or even all of the implementation code for specific deployment technologies such as J2EE, .Net, SOAP ("Simple Object Access Protocol")

UCL
ComputerScience

PIM: Platform Independent Model

- Represents business functionality and behavior, undistorted by technology issues
 - Independent of technology platforms e.g. J2EE, Web Services or .Net
- A detailed model:
 - Usually in UML
 - Should capture pre and post conditions in Object Constraint Language (OCL)
 - OCL is part of UML
 - Should specify semantics in Action Language
 - at the moment this is not possible as Action Language has not been fully specified
 - UML 1.4 specifies action semantics, but not syntax

Note: This is quoted from <http://www.cs.ucl.ac.uk/staff/c.mascolo/www/arlou.pdf>

UCL
ComputerScience

PSM: Platform Specific Model

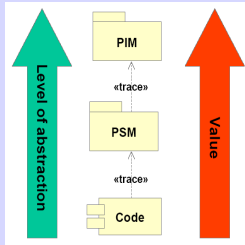
- The PSM is created by mapping the PIM to a specific technology platform
 - The CASE tool is supplied with information about a specific implementation technology and how to map the PIM to this technology
- The PSM is dependent on a specific technology platform e.g. J2EE
- The PSM is also a detailed model:
 - *Must* be in UML
 - Semantics are specified in Code
 - partly generated
 - partly hand written

Note: This is quoted from <http://www.cs.ucl.ac.uk/staff/c.mascolo/www/arlou.pdf>

UCL
ComputerScience

Value increases with level of abstraction

- Since both the PSM and the code are generated from the PIM, the PIM has the highest business value
- However, need an MDA tool which is capable of doing the mappings
- Such tools have started emerging and are still being developed further



© 2002 Clear View Training Limited
<http://www.clearviewtraining.com>



UCL

ComputerScience

How do we get a PIM??

We've seen the main concepts of MDA

- Software development using a platform independent model (PIM)
- Mapping of PIM to a platform specific model and subsequently to platform specific code

But how do we actually create a PIM?

- Object Oriented Analysis etc.
- An interesting alternative: Archetype patterns
 - See <http://www.cs.ucl.ac.uk/staff/c.mascolo/www/arlou.pdf> and booklist on last slide



UCL

ComputerScience

The future and MDA...

- Some estimates of the OMG:
 - MDA market will reach about \$500,000,000 by 2005
 - MDA will influence 50% of the tools market by 2007
- MDA is the OMG's strategic direction for the future of software development
- MDA is very likely to be the future of Software Engineering




UCL

ComputerScience


Conclusions

- Model Driven Architecture used to:
 - Achieve true platform independence
 - Construct software models that stay consistent even when technology changes (using PIMs)
 - Increase productivity and efficiency by being able to focus on the problem domain rather than the implementation domain
 - Ensure easy information exchange between different technologies and applications by using open, platform neutral information model standards (such as XML)
- However:
 - People with different skill sets might be required for this way of engineering software
 - Very sophisticated tools are necessary

 **UCL ComputerScience**

Acknowledgements

- Remember Dr. Jim Arlow's seminar on MDA and Archetype Patterns last year? I would like to thank Jim Arlow for his presentation and very helpful handouts, which you can find on the web:
<http://www.cs.ucl.ac.uk/staff/c.mascolo/www/arlow.pdf>

 **UCL ComputerScience**


References

Books for further reading:

- R. Hubert, D.A. Taylor: "Convergent Architecture: Building Model Driven J2EE Systems with UML". Wiley 2002.
- Jim Arlow, Ila Neustadt: "Enterprise Patterns and MDA: Building Better Software with Archetype Patterns and UML". Booch Jacobson Rumbaugh

Web:

- <http://www.omg.org/mda>
- Google Search: "Model Driven Architecture"

 **UCL ComputerScience**
