
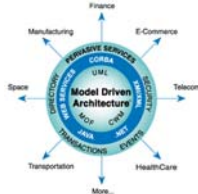



## KnowFuture 2003: UML 2.0 & MDA

Zürich, 29 January 2003

KnowGravity Inc, Badenerstrasse 808, 8048 Zürich  
 Tel. 01/434'20'00, Fax. 01/434'20'09  
<http://www.knowgravity.com>


KnowFuture 2003: UML 2.0 & MDA
1
© Copyright 2003, KnowGravity Inc.




## Agenda

17:00 Welcome	Patrick Grässle
17:10 Overview UML 2.0	Patrick Grässle
17:40 CASSANDRA/xUML-Simulator & Demo	Markus Schacher
18:10 Short Break	
18:30 Overview MDA	Markus Schacher
19:00 CASSANDRA/xUML-Generator & Demo	Reto Stahel
19:30 Summary	Reto Stahel
19:40 Apéro & Discussion	

KnowFuture 2003: UML 2.0 & MDA
2
© Copyright 2003, KnowGravity Inc.




# UML 2.0



## An Overview

KnowFuture 2003: UML 2.0 & MDA 3 © Copyright 2003, KnowGravity Inc.




## UML 2.0 Goals

- Restructure and refine UML to make it *easier* to apply, implement and customize
- Infrastructure goals
  - define a reusable *metalanguage kernel* to define the UML
  - provide more powerful mechanisms to *customize* UML
- Superstructure goals
  - Improve support for *component-based development*
  - Refine *architectural specification* capabilities
  - Increase the *scalability, precision* and integration of behavioral diagrams
  - Review all UML 1.x constructs and diagrams

KnowFuture 2003: UML 2.0 & MDA 4 © Copyright 2003, KnowGravity Inc.

## UML History




- 1995: UM 0.8 (initial collaboration)
- 1996: UML 0.9 (redefinition of objectives)
- January 1997: 1.0 (initial OMG standard)
- September 1997: UML 1.1 (first revision)
- 1998: UML 1.2 (internal release only)
- 1999: UML 1.3 (published update)
- September 2001: UML 1.4 (current version)
- October 2002: UML 1.5 (with Action Semantics)
- 1Q 2003?: UML 2.0 (reorganization and alignment with MDA)

UML has come a long way!

Benefits for users:  
• proven in practice

KnowFuture 2003: UML 2.0 & MDA 5 © Copyright 2003, KnowGravity Inc.

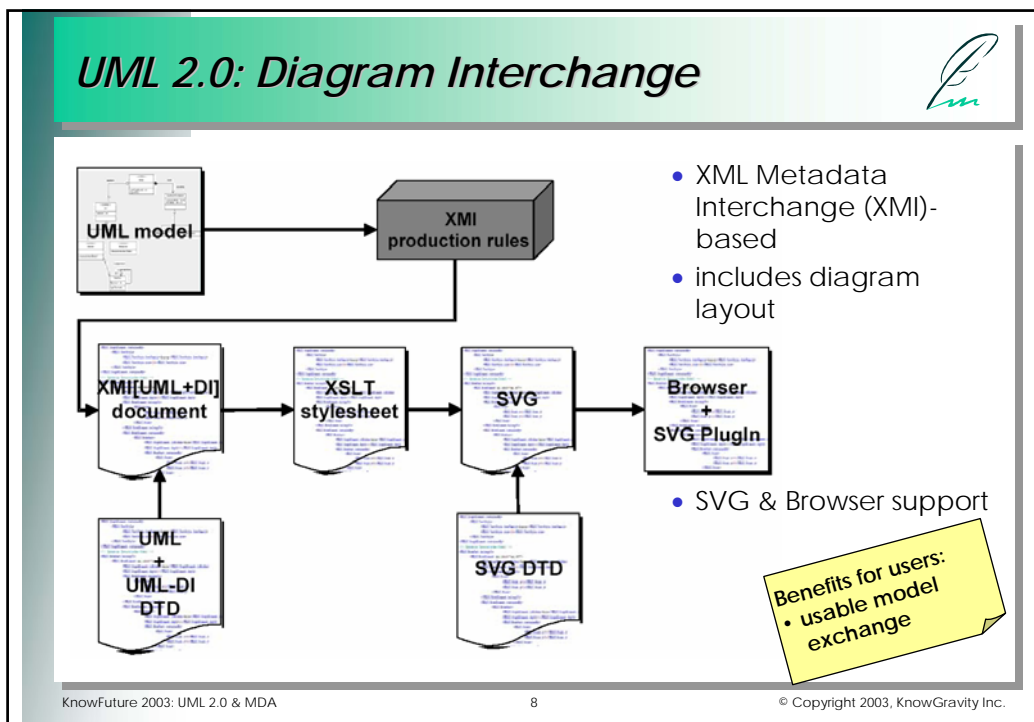
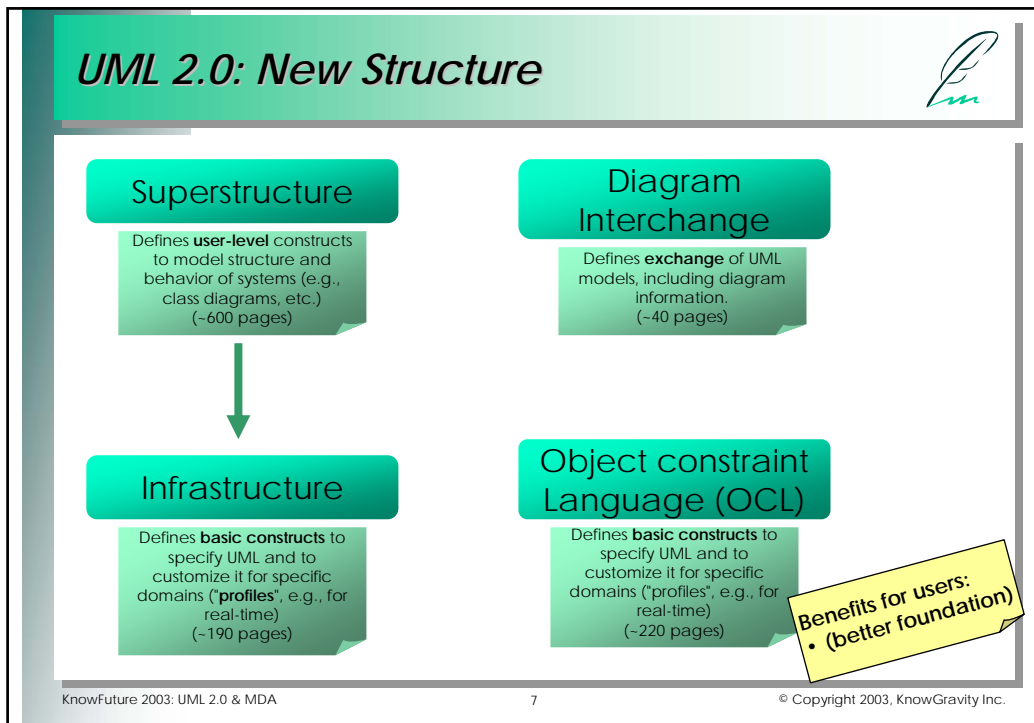
## UML 2.0 Highlights



- New structure
- Diagram Interchange
- Compliance points and levels
- Ports, Connectors & Parts
- Updated State Diagrams
- Action Semantics
- Updated Sequence Diagrams
- Updated Activity Diagrams

Benefits for users:  
• better encapsulation  
• executable / early testing

KnowFuture 2003: UML 2.0 & MDA 6 © Copyright 2003, KnowGravity Inc.



## UML 2.0: Compliance Points & Levels



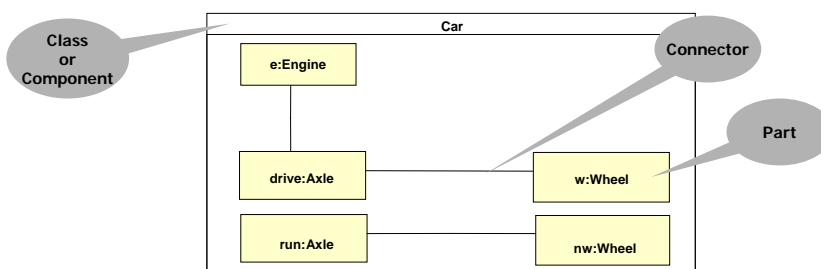
- Compliance points
  - are based on packages from the meta model
  - define the level of support for the UML meta model
  - can be used to specify how good a tool supports UML 2.0
- Compliance levels are sets of compliance points.
- UML Superstructure compliance levels:
  - Basic (level 1)
  - Intermediate (level 2)
  - Complete (level 3)

✘<sup>1.3</sup> Our tool is supports UML

✔<sup>2.0</sup> Our tool is compliant with compliance level 2

Benefits for users:  
• better comparison of tools

## UML 2.0: Parts & Connectors



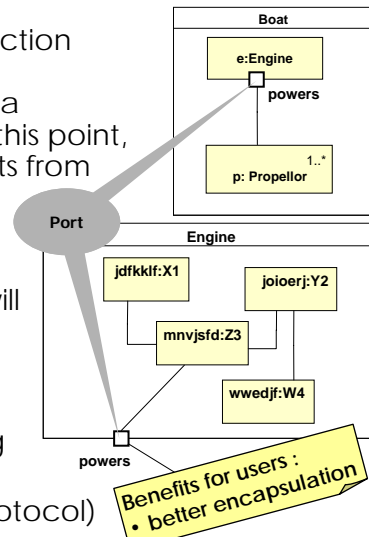
- **Part:** Represents a set of instances that are strongly aggregated within a containing classifier instance. Parts may be joined by attached connectors and specify configurations of linked instances.
- **Connector:** Specifies a link (an instance of an association) that enables communication between parts in a structure and with the environment.

Benefits for users :  
• show internal structure

## UML 2.0: Ports



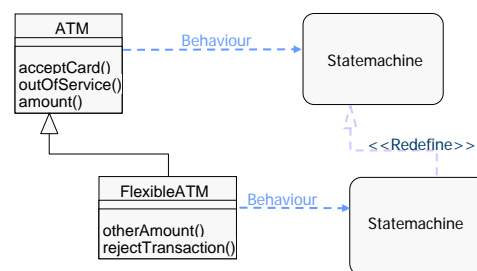
- **Port:** Specifies an addressable interaction point between a classifier and its environment. Describes the services a classifier offers to its environment at this point, and the services the classifier expects from the environment.
- Enables specification of a classifier **without knowing anything** about the environment in which the classifier will be used
  - send to port, receive messages from port
  - ports relay communication along connectors
  - may have complex behavior (protocol)



## UML 2.0: Updated State Diagram



- Notational enhancements
- Backward Compatibility
  - In general all UML 1.4 state machine concepts are preserved in the 2.0 proposal
- New core constructs added
  - (Action Language)
  - State machine specialization/redefinition
  - Transitions pre/post conditions

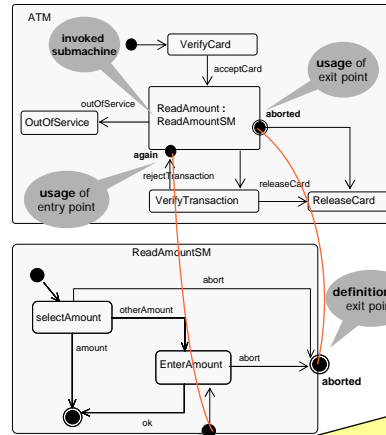


Benefits for users :  
• reuse and refinement

## UML 2.0: Updated State Diagram (2)



- New core constructs added (cont.)
  - Connection points replace nested State Diagrams



**Benefits for users:**

- reuse via encapsulation of state machine control interfaces
- scalability

## UML 2.0: Action Semantics



- not completely new in UML 2.0
- action in UML: fundamental unit of behavior specification
- used to specify the behavior of transitions, activities, operations, etc. in a procedural way
- might be replaced in later versions by OCL which allows declarative description

**Benefits for users:**

- formal way to describe behavior
- makes behavior executable

## UML 2.0: Updated Sequence Diagrams

- Interaction context
- Control structures (sequence, iteration, selection, parallel)
- Nested sequence diagrams
- Interaction overview diagram
- Timing diagram
- New notation

**Benefits for users:**

- gerealization of scenarios
- better alignment with state diagrams
- improved decomposition and structuring

KnowFuture 2003: UML 2.0 & MDA 15 © Copyright 2003, KnowGravity Inc.


## UML 2.0: Updated Activity Diagrams

- separated from state diagram
- Better alignment with Petri nets
  - Parallelism
  - Token queuing
- Better support for workflow modeling
  - streaming parameters
  - hierarchical partitions

**Benefits for users:**

- better suited for a variety of processes
- more intuitive for process modelers

KnowFuture 2003: UML 2.0 & MDA 16 © Copyright 2003, KnowGravity Inc.

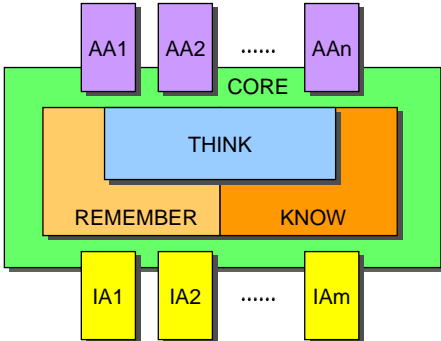


## CASSANDRA/xUML-Simulator

Executable UML Models

KnowFuture 2003: UML 2.0 & MDA
17
© Copyright 2003, KnowGravity Inc.

## An Operating System fo SE Applications




- **CORE:** basic infrastructure (GUI, XML, persistency, licensing, etc.)
- **REMEMBER:** a UML-based declarative database
- **KNOW:** common sense in form of a „class model of the world“
- **THINK:** an inference engine to process know how
- **IAx:** A set of Interface Agents for various CASE tools
- **AAx:** A set of (hopefully useful) Application Agents

KnowFuture 2003: UML 2.0 & MDA
18
© Copyright 2003, KnowGravity Inc.

## CASSANDRA Application Agents

- Use case model reviewer
- Domain object model reviewer
- Data warehouse builder
- Effort estimator
- **xUML simulator**
- **MDA-based code generator (soon)**
- General information source
- Domain object suggester
- more ...



} CASSANDRA/CS only

KnowFuture 2003: UML 2.0 & MDA 19 © Copyright 2003, KnowGravity Inc.

## The xUML Simulator (based on UML-VM)

Simulates directly

- Actors, use cases and sequence diagrams
- Classes and associations (including association classes)
- Multiple instances of communicating state machines
- (Real-)time

Action language for state transitions

- Instance construction and destruction
- Link construction and destruction
- Association navigation
- Boolean, arithmetic, set and term expressions
- All and exist quantifiers
- Events for synchronous and asynchronous communication
- Broadcasts over associations

KnowFuture 2003: UML 2.0 & MDA 20 © Copyright 2003, KnowGravity Inc.

A presentation slide with a green header bar containing the KnowGravity logo. The main content area is white with the word "Demonstration..." in a large, bold, italicized black font. The footer contains the text "KnowFuture 2003: UML 2.0 & MDA", the number "21", and "© Copyright 2003, KnowGravity Inc.".

***Demonstration...***

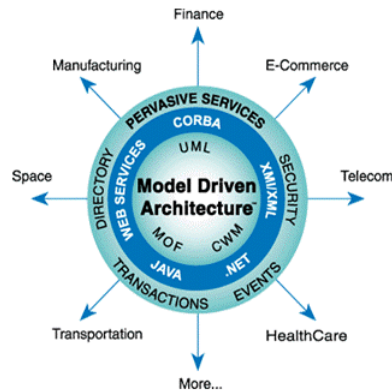
KnowFuture 2003: UML 2.0 & MDA 21 © Copyright 2003, KnowGravity Inc.

A presentation slide with a green header bar containing the KnowGravity logo. The main content area is white with the words "Short Break" in a large, bold, italicized black font. The footer contains the text "KnowFuture 2003: UML 2.0 & MDA", the number "22", and "© Copyright 2003, KnowGravity Inc.".

***Short Break***

KnowFuture 2003: UML 2.0 & MDA 22 © Copyright 2003, KnowGravity Inc.

## OMG's Model Driven Architecture (MDA)

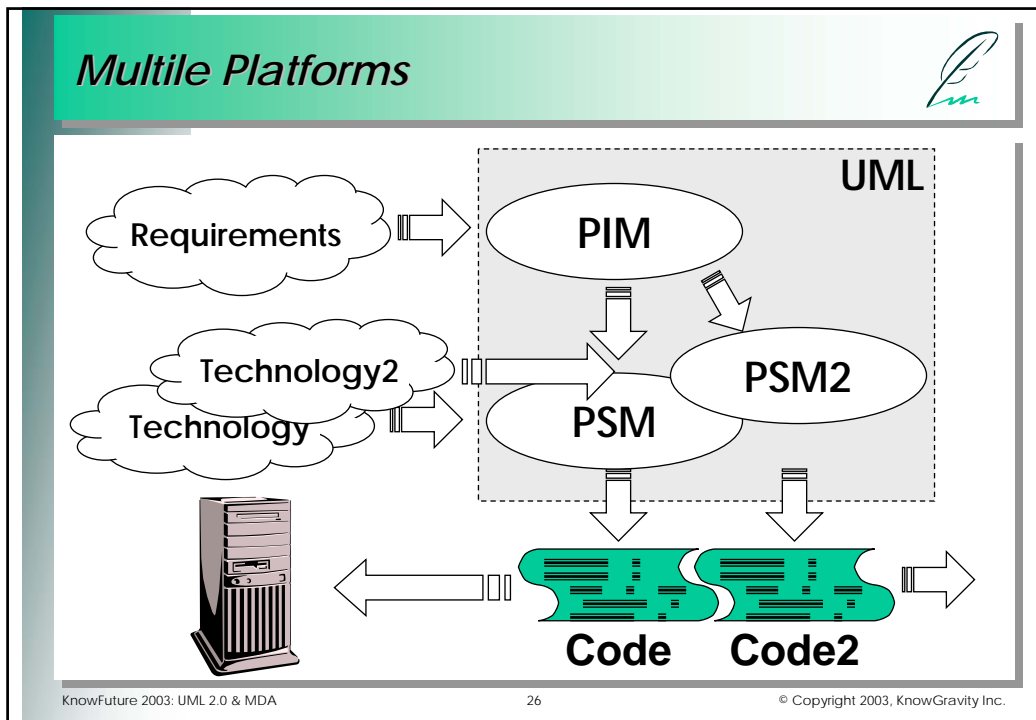
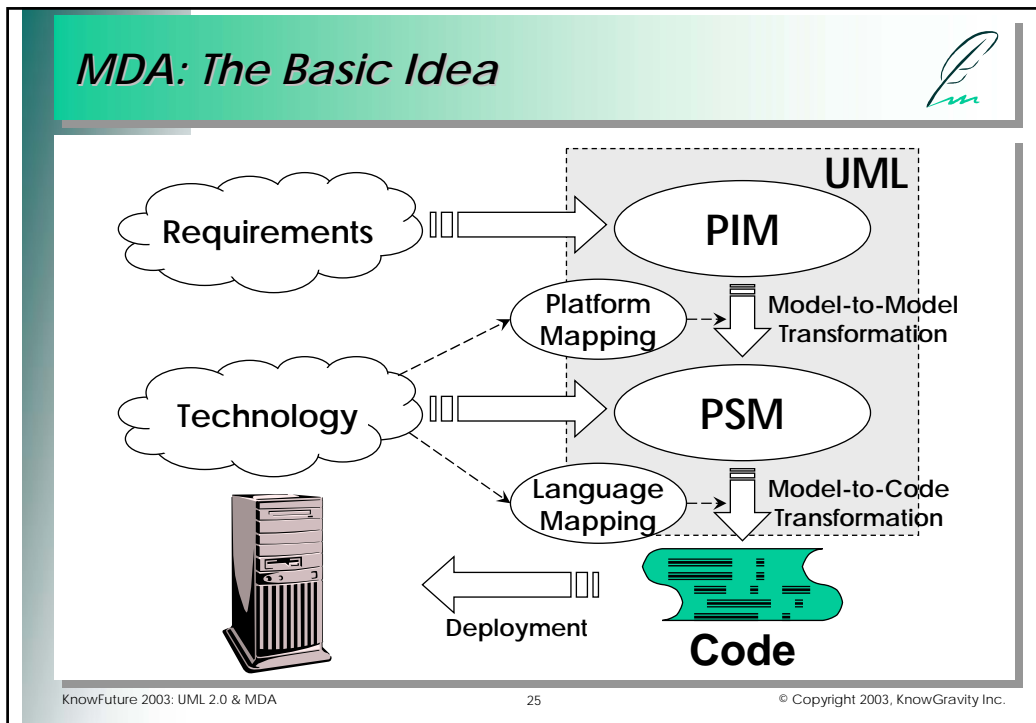


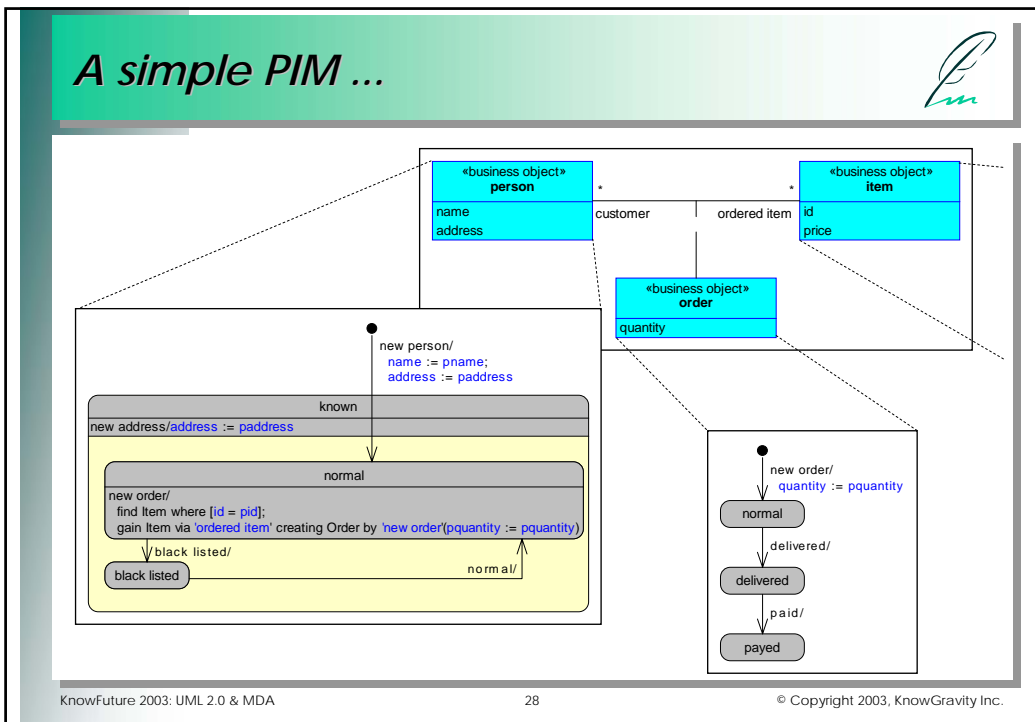
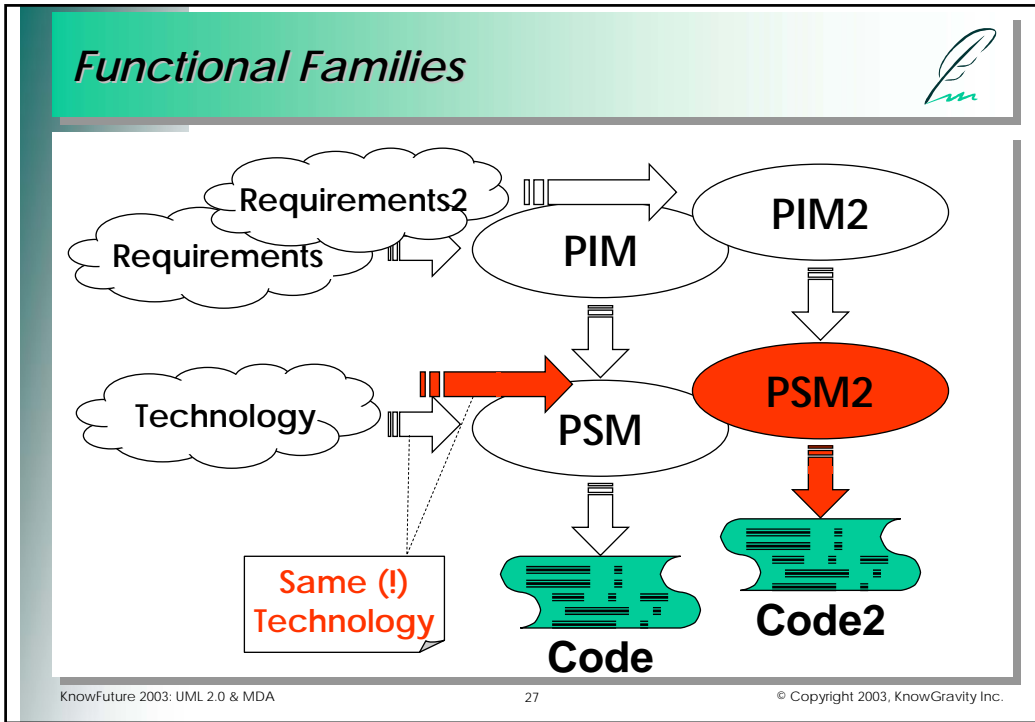
## MDA: OMG's Definition

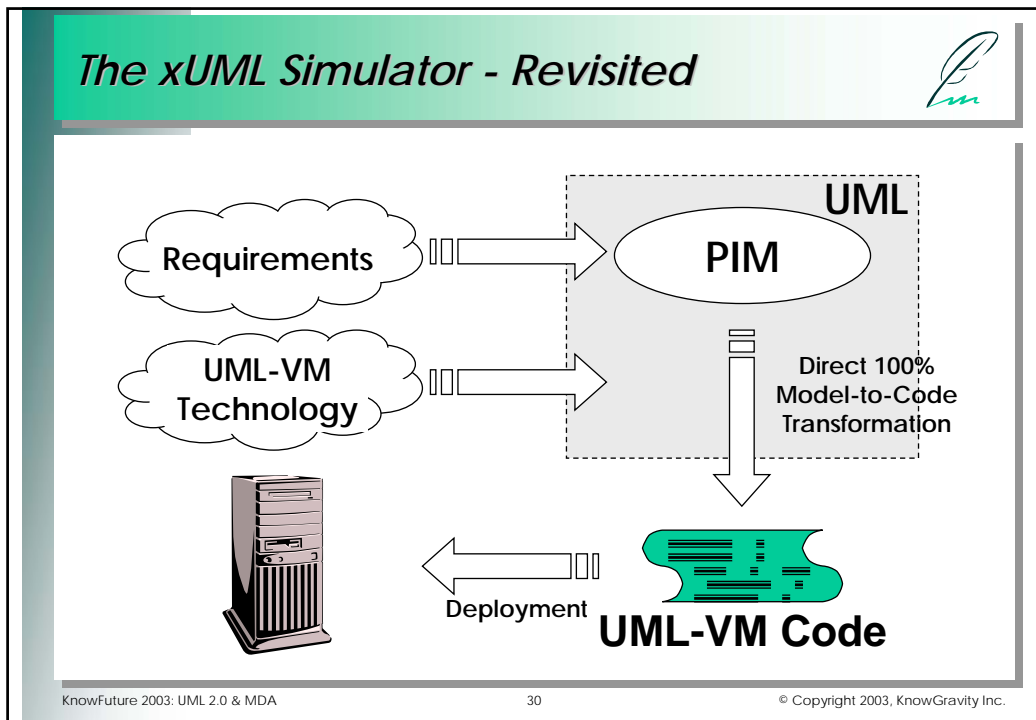
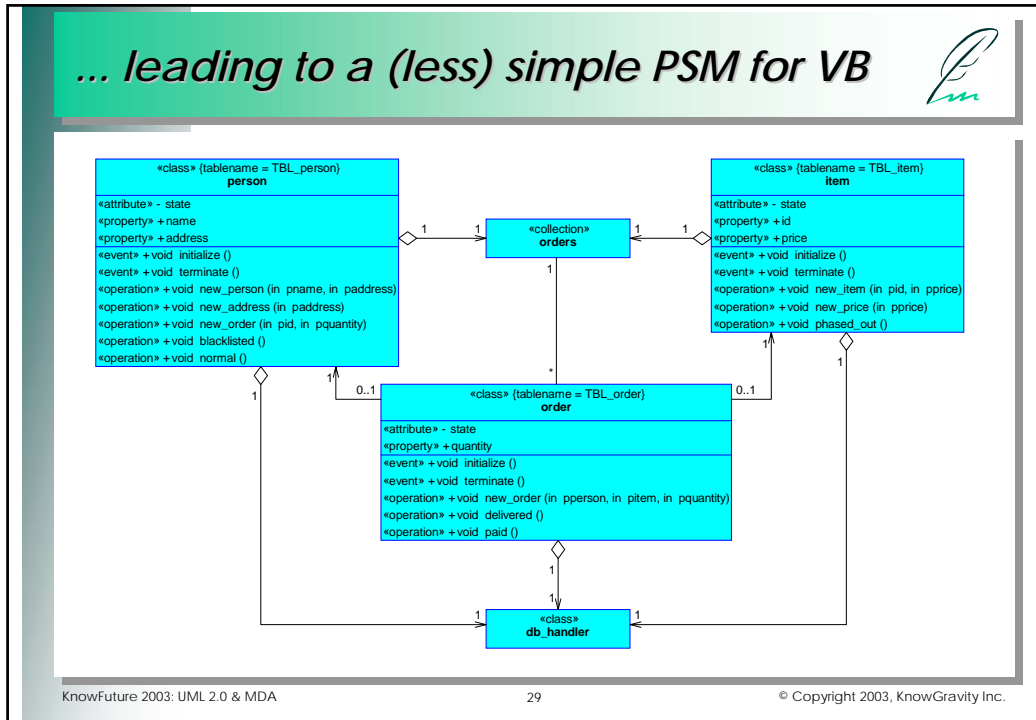


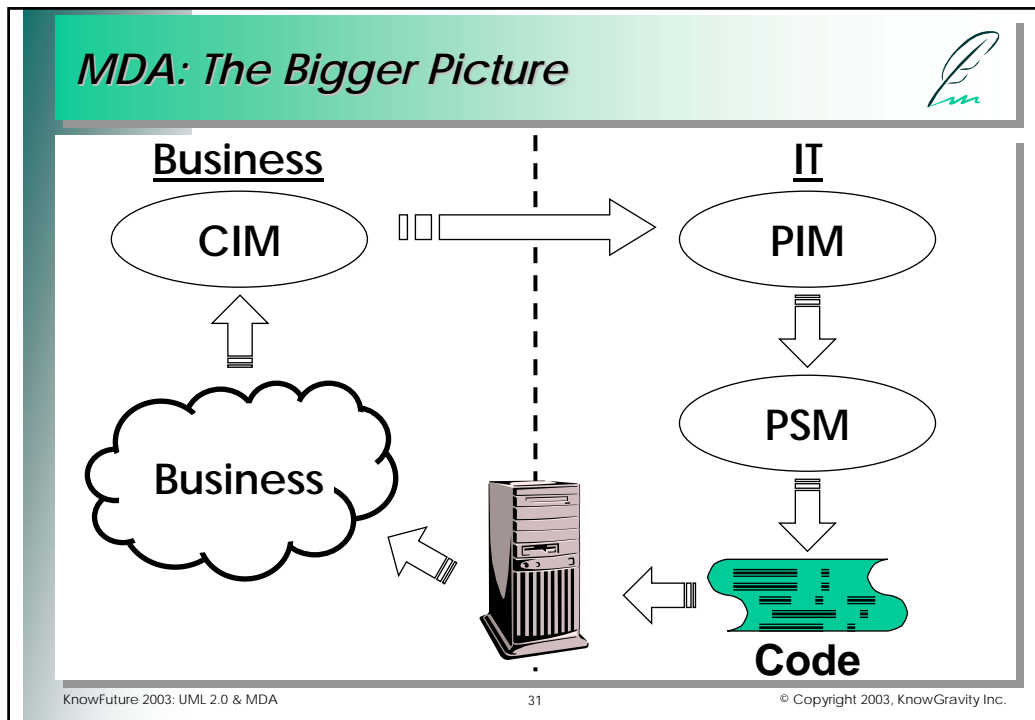
„The MDA defines an approach to IT system specification that separates the specification of system functionality from the specification of the implementation of that functionality on a specific technology platform. To this end, the MDA defines an architecture for models that provides a set of guidelines for structuring specifications expressed as models.“

OMG, „Model Driven Architecture (MDA)“, July 2001









### MDA Models (1)

CIM: Computation Independent Model

- Describes the business
- Source for business rules (requirements)
- Represented in UML (?)

PIM: Platform Independent Model

- Specifies business rules (requirements) to be automated
- Represents reusable business knowledge
- Represented in UML

KnowFuture 2003: UML 2.0 & MDA 32 © Copyright 2003, KnowGravity Inc.

## MDA Models (2)



### PSM: Platform Specific Model

- Respects technology concepts and constraints
- Derived by applying design patterns to PIM
- Represented in UML


### Code

- Represents the executable IT system
- Derived by applying (programming) language idioms to PSM
- Represented in any (programming) language

## Related OMG Standards




- **MOF: Meta Object Facility**  
A meta-metamodel, the „mother“ of all OMG metamodels
- **XMI: XML Metadata Interchange**  
An XML-based interchange format for UML (MOF) models
- **HUTN: Human Usable Text Notation**  
A (really) human readable interchange format for UML (MOF) models
- **CWM: Common Warehouse Model**  
A metamodel supporting data exchange between data warehouse tools
- **SPEM: Software Process Engineering Model**  
A metamodel supporting standardized representations of software engineering processes



## CASSANDRA/xUML-Generator

Towards 100% Code Generation

KnowFuture 2003: UML 2.0 & MDA
35
© Copyright 2003, KnowGravity Inc.




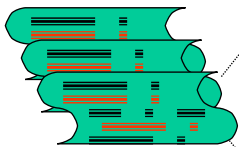
### Usual Code Generation Approach: Generate X% of 100%

#### PSM

```

class «class» {tablename = TBL_order
    order
    «attribute»- state
    «property»+quantity
    «event»-void initialize ()
    «event»-void terminate ()
    «operation»void new_order (in pperson, in pitem, in pquantity)
    «operation»-void delivered ()
    «operation»-void paid ()
        
```





#### Generated Code

#### Disadvantages:

- Generated & manual code mixed
- Gen. Code may be filled up with sync tags
- Generates only static structures
  - ⇒ Must add complex dynamics by "hand"
  - ⇒ No speed up for maintenance

```

Class order{
/** CASETOOLX::MEMBERVAR::52::BEGIN::DO_NOT_REMOVE
    private int m_quantity;
/** CASETOOLX::MEMBERVAR::52::END::DO_NOT_REMOVE

/** CASETOOLX::OPERATION::12563::BEGIN::DO_NOT_REMOVE
    public order(int personid, int itemid, int pquantity){
        // TODO: Enter your code here...
    }
/** CASETOOLX::OPERATION::12563::END::DO_NOT_REMOVE
/** CASETOOLX::OPERATION::16234::BEGIN::DO_NOT_REMOVE
    public void delivered(){
        // TODO: Enter your code here...
    }
/** CASETOOLX::OPERATION::16234::END::DO_NOT_REMOVE
}
        
```

KnowFuture 2003: UML 2.0 & MDA
36
© Copyright 2003, KnowGravity Inc.

## KnowGravity Approach: Generate 100% of X%

### PSM

```

class «class» {tablename = TBL_order
  order
  «attribute» state
  «property» quantity
  «event» void initialize ()
  «event» void terminate ()
  «operation» void new_order (in ppersonid)
  «operation» void delivered ()
  «operation» void paid ()
    
```

### Advantages:

- You don't have to touch generated code
- Generates clean & understandable code
- Generates full dynamic system  
⇒ Huge timesaver for testing, maintenance & further development!
- Technology becomes a don't care!

```

class order{
  private int m_quantity;
  private String m_state;

  // Constructor
  public order(int personid, int itemid, int pquantity){
    m_quantity = pquantity;
    enter_state("order:normal");
  }

  // Operations
  public void delivered(){
    if(m_state.equals("order:normal")){
      enter_state("order:delivered");
    }
  }
}
    
```

KnowFuture 2003: UML 2.0 & MDA
37
© Copyright 2003, KnowGravity Inc.

## Bridge to the Outer World

### Why is there still Manual Code?

- Stepwise migration to code generation
- Existing Code
- Very specific interface (to GUI, HW, etc.)
- Other code generator

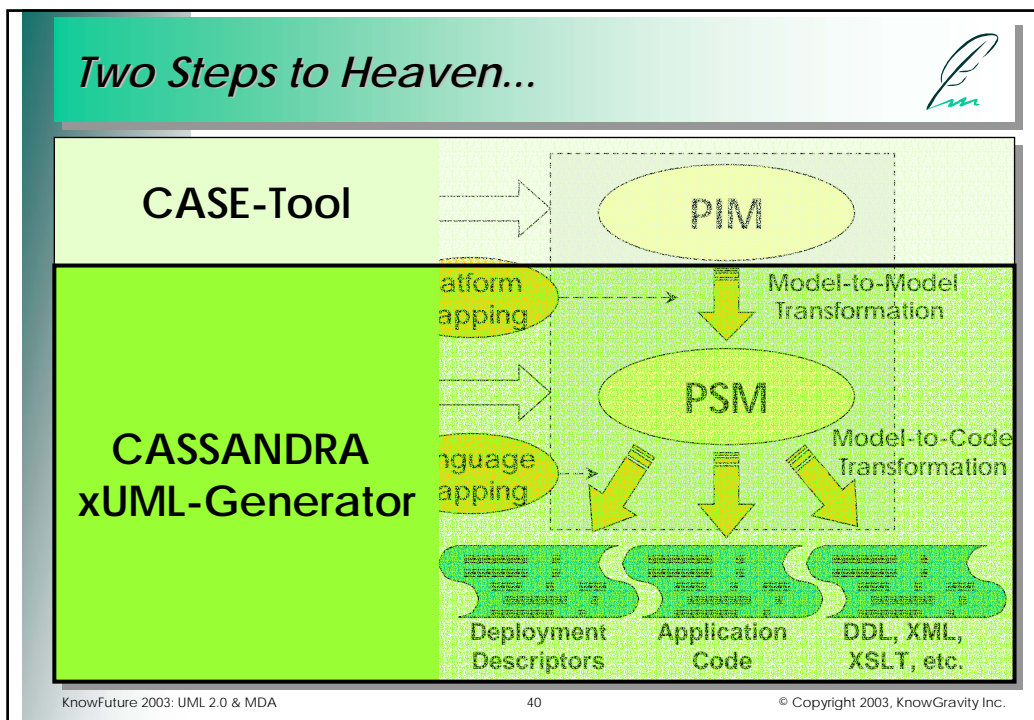
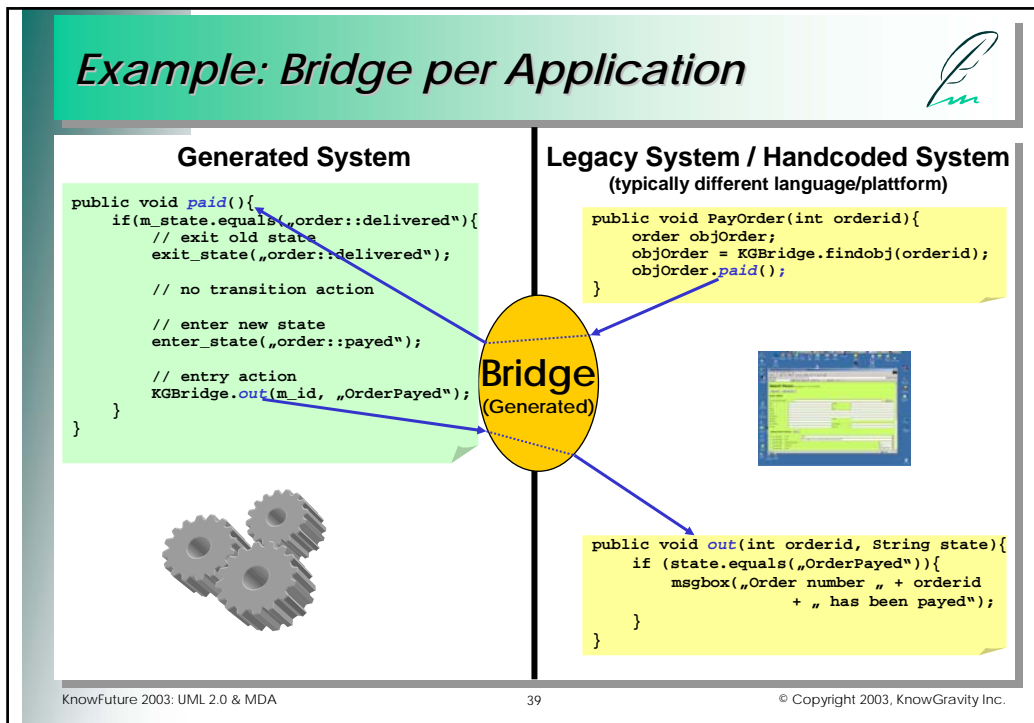
⇒ Generated & Manual Code need to communicate (logo...)

⇒ We need a Bridge over this gap, with similar technology like a middleware (i.e. CORBA/COM)

Several Bridge-Pattern possible:

- Bridge per Application/Package** (typeless)
- Bridge per Class** (typesafe)
- Bridge per Object** (typesafe)

KnowFuture 2003: UML 2.0 & MDA
38
© Copyright 2003, KnowGravity Inc.



KNOW GRAVITY

*Demonstration...*

KnowFuture 2003: UML 2.0 & MDA 41 © Copyright 2003, KnowGravity Inc.

This slide features a green header with the KnowGravity logo. The main content area is white with the word "Demonstration..." in a large, italicized, black serif font. The footer contains the text "KnowFuture 2003: UML 2.0 & MDA", the slide number "41", and the copyright notice "© Copyright 2003, KnowGravity Inc."

KNOW GRAVITY

*Summary*

MDA is here today!

KnowFuture 2003: UML 2.0 & MDA 42 © Copyright 2003, KnowGravity Inc.

This slide features a green header with the KnowGravity logo. The main content area is white with the word "Summary" in a large, italicized, black serif font. Below it, the text "MDA is here today!" is centered in a smaller, black sans-serif font. The footer contains the text "KnowFuture 2003: UML 2.0 & MDA", the slide number "42", and the copyright notice "© Copyright 2003, KnowGravity Inc."

## Summary



### UML 2.0

- Is bigger than UML 1.x but has a better structure
- Provides better support for black box components
- Has precise semantics and action language

### MDA

- Separates PSM from PIM and PIM from CIM
- Is the foundation for reusable domain models and architectures
- Emphasizes precise modeling: „the model is the code“

### CASSANDRA

- Now provides immediately executable UML models
- Will provide targetable full code generation soon
- Already provides some other software engineering services

## KnowGravity's xUML Starter Kit



### Why?

- You need to specify complex functionalities
- You are interested in customizable code generation
- You don't believe in xUML or MDA (yet)

### What?

- You get an executable model in your domain
- You will learn how to model functionality precisely
- You get a (demo) license of CASSANDRA/xUML and ARTISAN RtS

### How?

- Teamwork between you and a KnowBody
- Price: CHF 9000, including 8 days consultancy
- Available now from KnowGravity

## European Business Ruls Conference 2003



- Zürich, 10-12 June 2003
- Keynotes from
  - Jim Sinur (Gartner)
  - Tony Morgan (XNP: Extreme Non-Programming)
  - Ron Ross (father of Business Rules)
  - Stan Hendryx (OMG)
- 3 Tracks
  - Experience
  - Solutions
  - Concepts
- 2 Half-day Tutorials
  - John Zachman
  - Ron Ross
- Comparative case study
- Exhibition with 8+ vendors