
Sims Associates

OPEN IT

MDA

-

The Real Value

Oliver Sims

Sims Associates

oliver.sims@simsassociates.co.uk

Background

- Consultant
  - Enterprise systems, component-based development, distributed system software architectures,
  - Transitioning to affective CBD
- Chief Architect
  - Component (app server) middleware, 1990s
  - OMG Architecture Board member
- Author
  - *Business Component Factory* (2000, with Peter Herzum)
  - *Building Business Objects* (1998, with Peter Eeles)
  - *Business Objects* (1994)
- Systems Engineer (IBM 1969-1993)
  - UK large complex systems
  - Specialist, distributed systems, programming technologies, communications, system design, manufacturing industry, etc.

Agenda

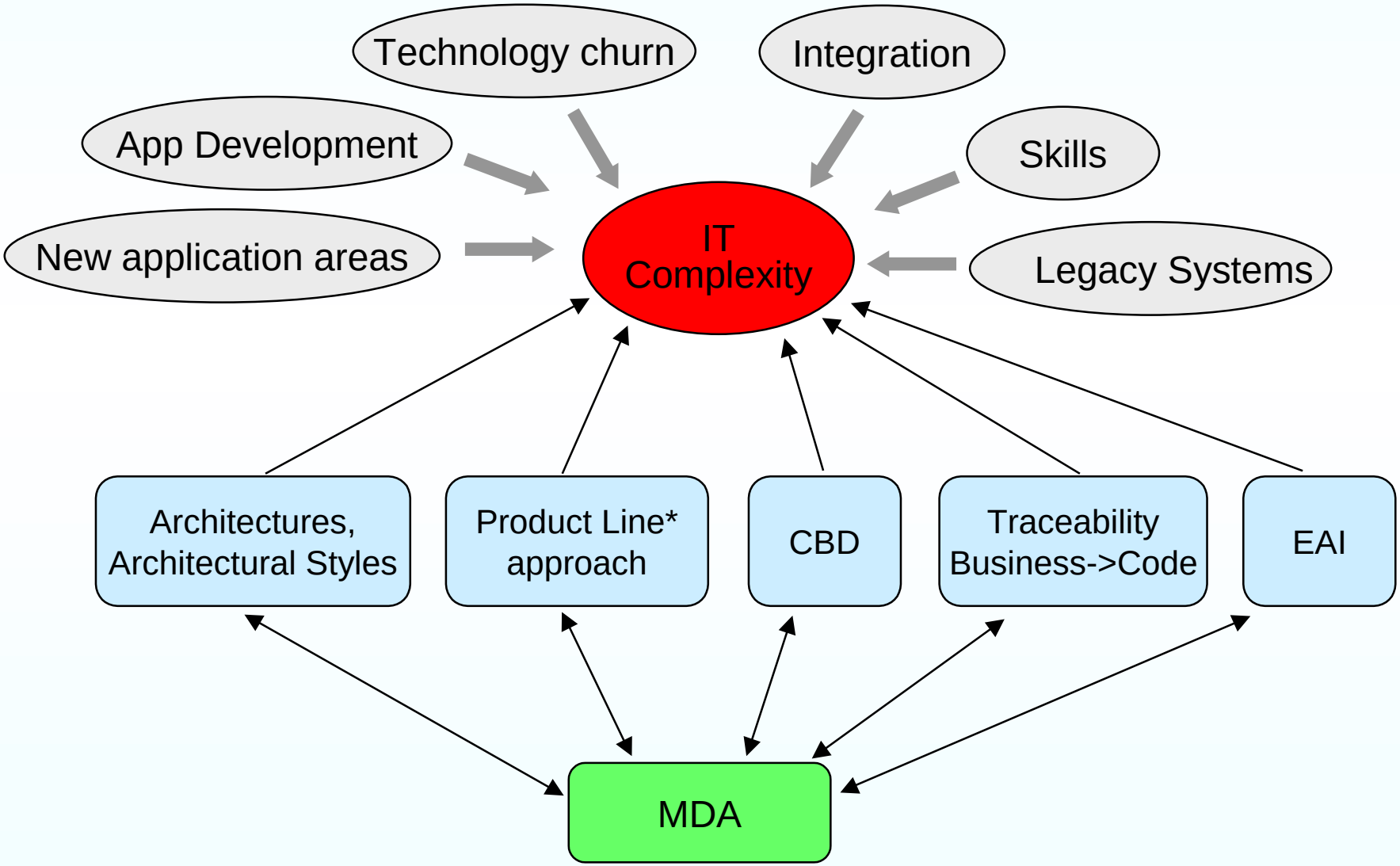
- MDA positioning
- MDA essentials
- MDA – the envelope
  - Product Line
  - Architecture
- MDA – the real value
- Next Steps

MDA Positioning (1)

- Application development is lengthy and complex
  - Software technology considerations
  - Business requirements become lost
  - Structural design re-invented
  - Architectural concerns often not separated
  - Integration demands often hugely complicated
- Many people have “done MDA” in the past
- MDA is the focus for resolution of the development crisis
  - A major strategy … from a major standards organization
  - Addresses a key architectural separation of concerns
  - Provides single conceptual framework and vocabulary
  - The focus for synergy between key resolution enablers…

* Systems Engineering Institute
http://www.sei.cmu.edu/plp/plp_init.html

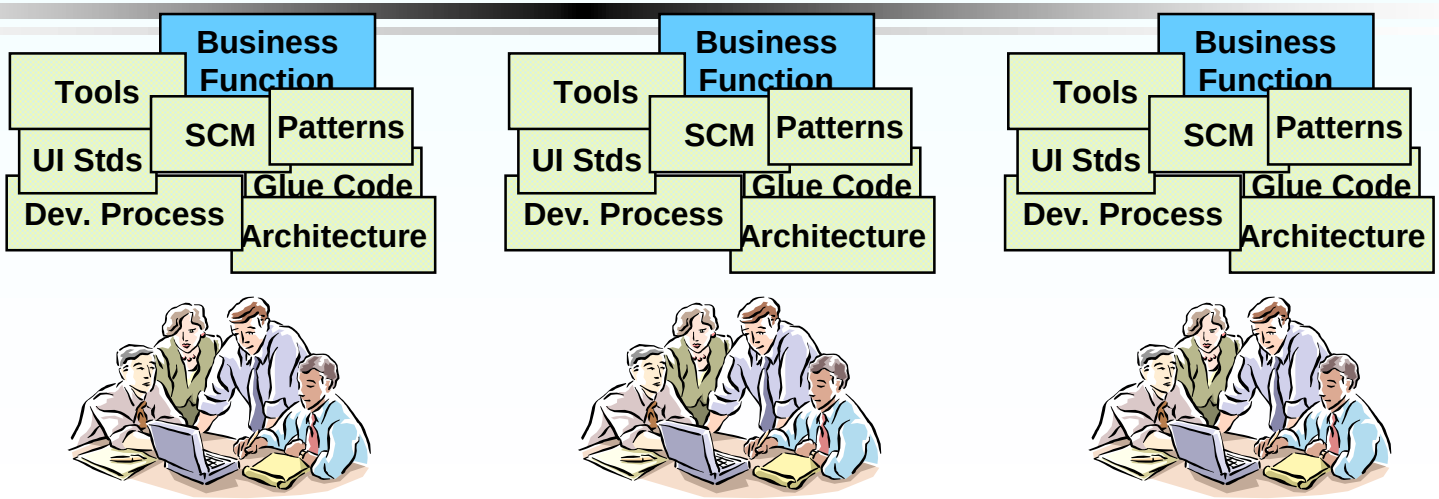
MDA Positioning (2)



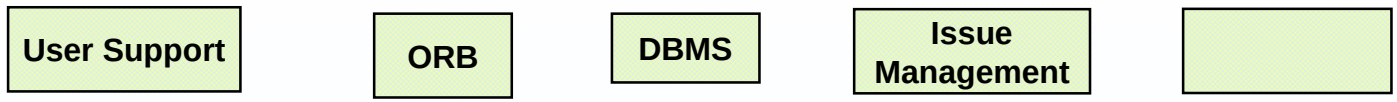
* Systems Engineering Institute
http://www.sei.cmu.edu/plp/plp_init.html

Traditional Development

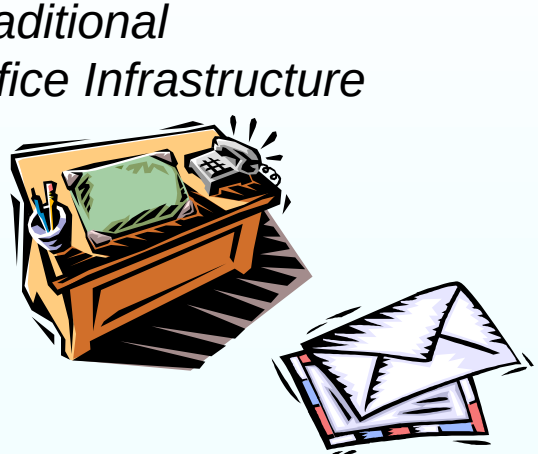
Traditional Development Projects

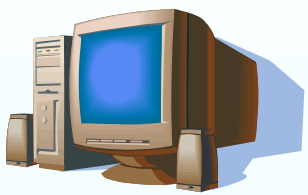


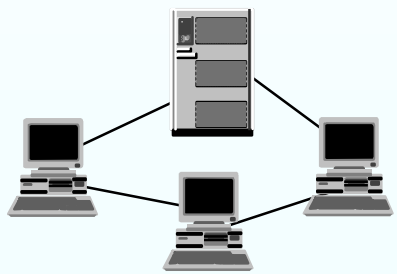
Traditional IT Infrastructure



Traditional Office Infrastructure

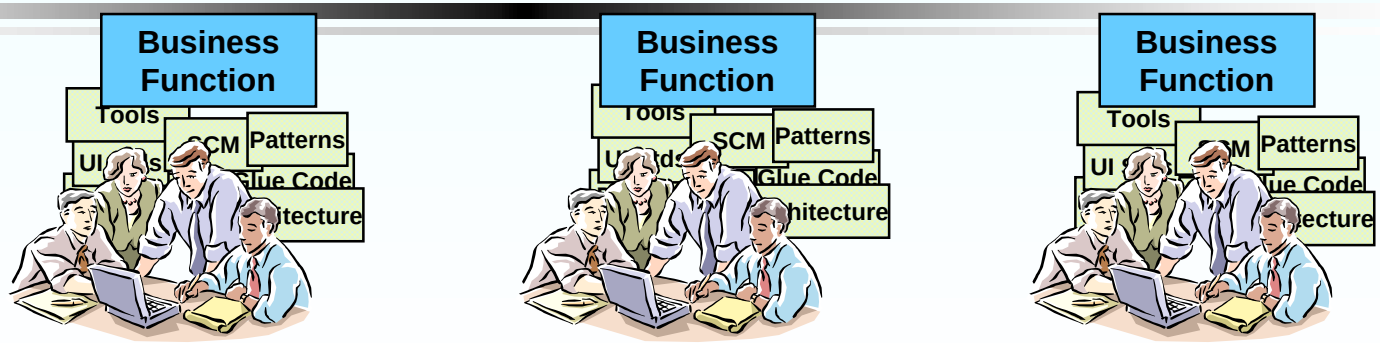






“Product Line” Development

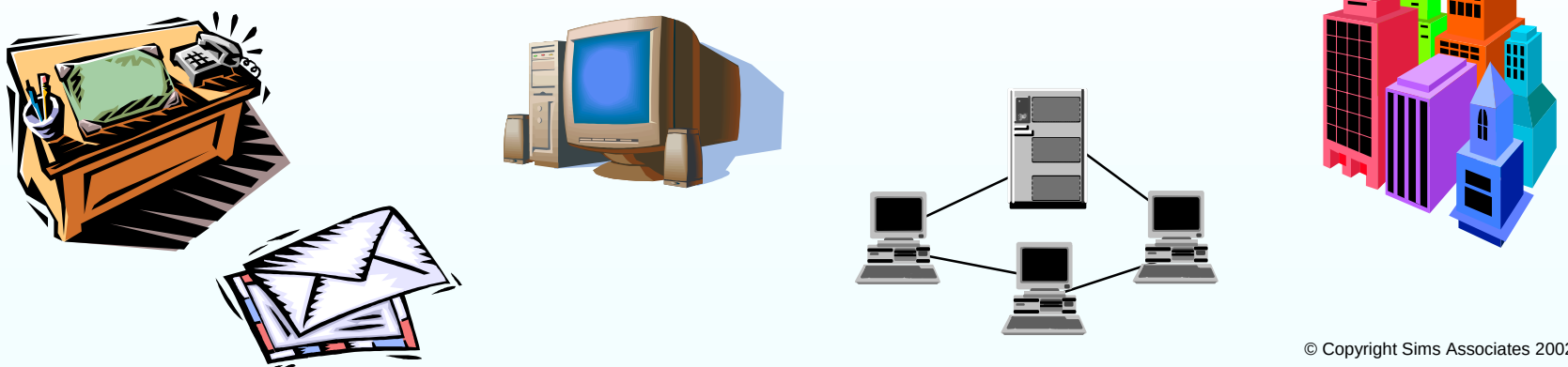
Business Function Projects



Product Line Infrastructure

Patterns
Architecture
UI Stds
Glue Code
Dev. Process
SCM
Tools
User Support
ORB
DBMS
Issue Management

Traditional Office Infrastructure



Agenda

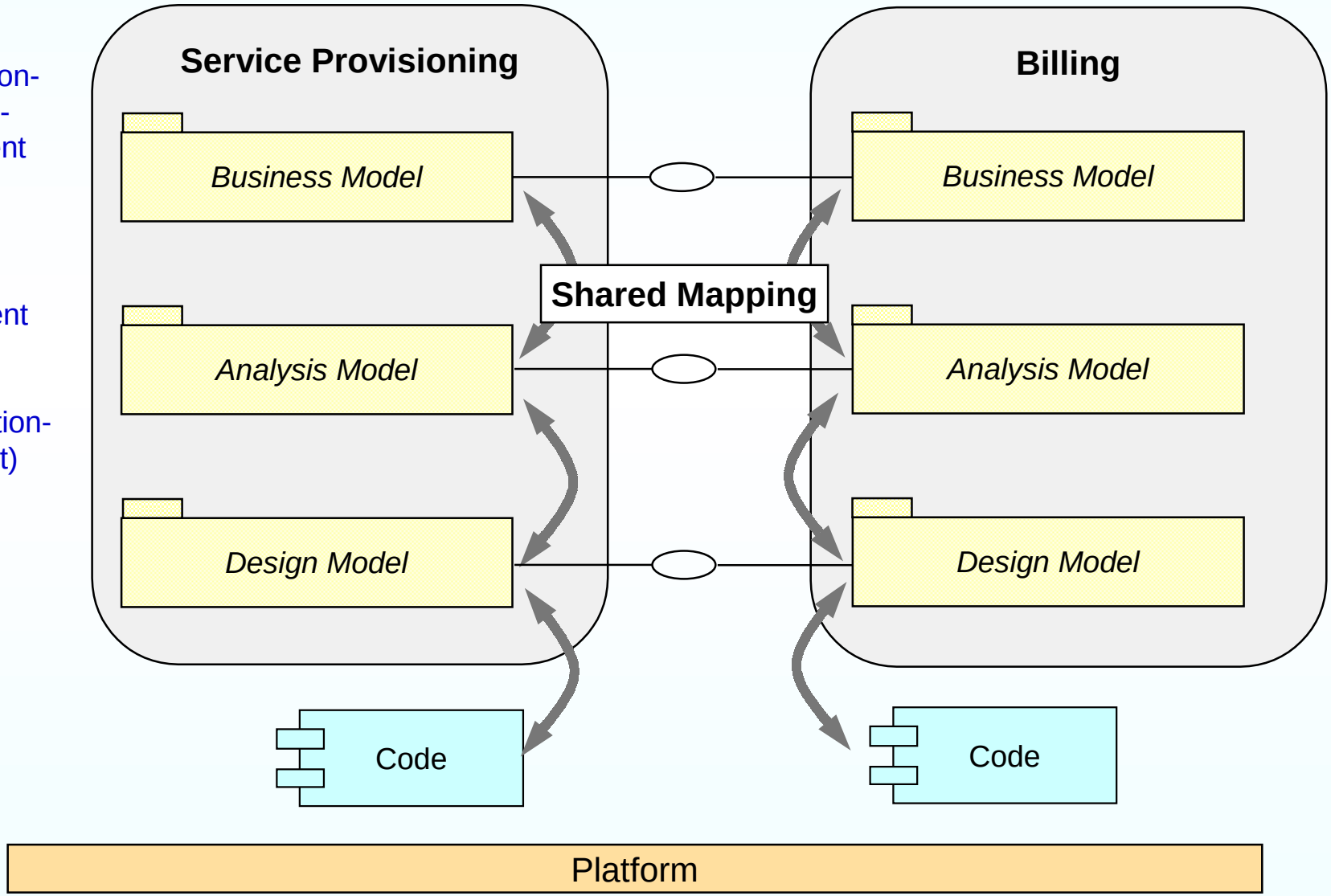
- MDA positioning
- MDA essentials
- MDA – the envelope
  - Product Line
  - Architecture
- MDA – the real value
- Next Steps

MDA Essentials

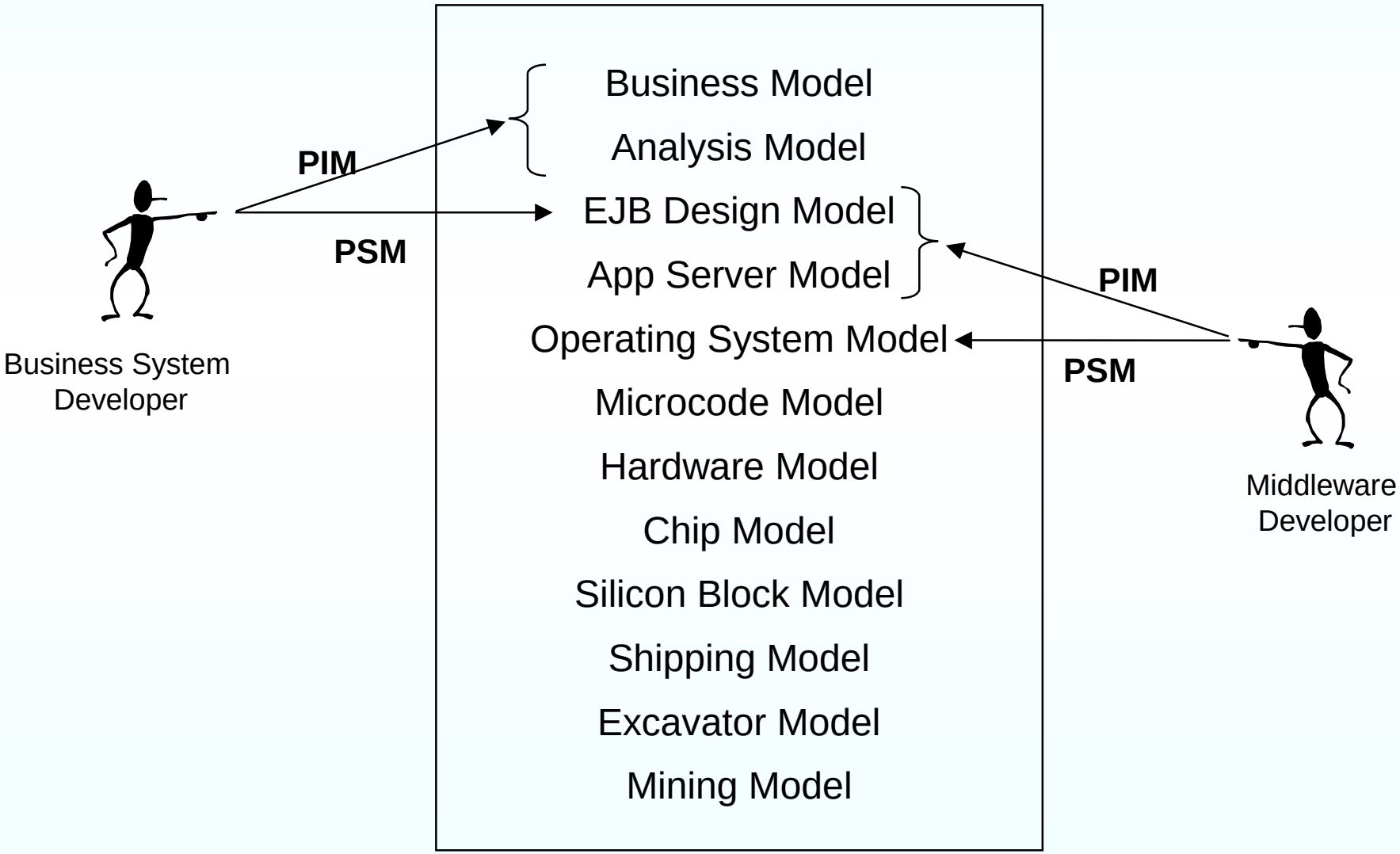
Computation- & Platform-Independent Model (**CIM/PIM**)

Platform-Independent Model (**PIM**) (Computation-Dependent)

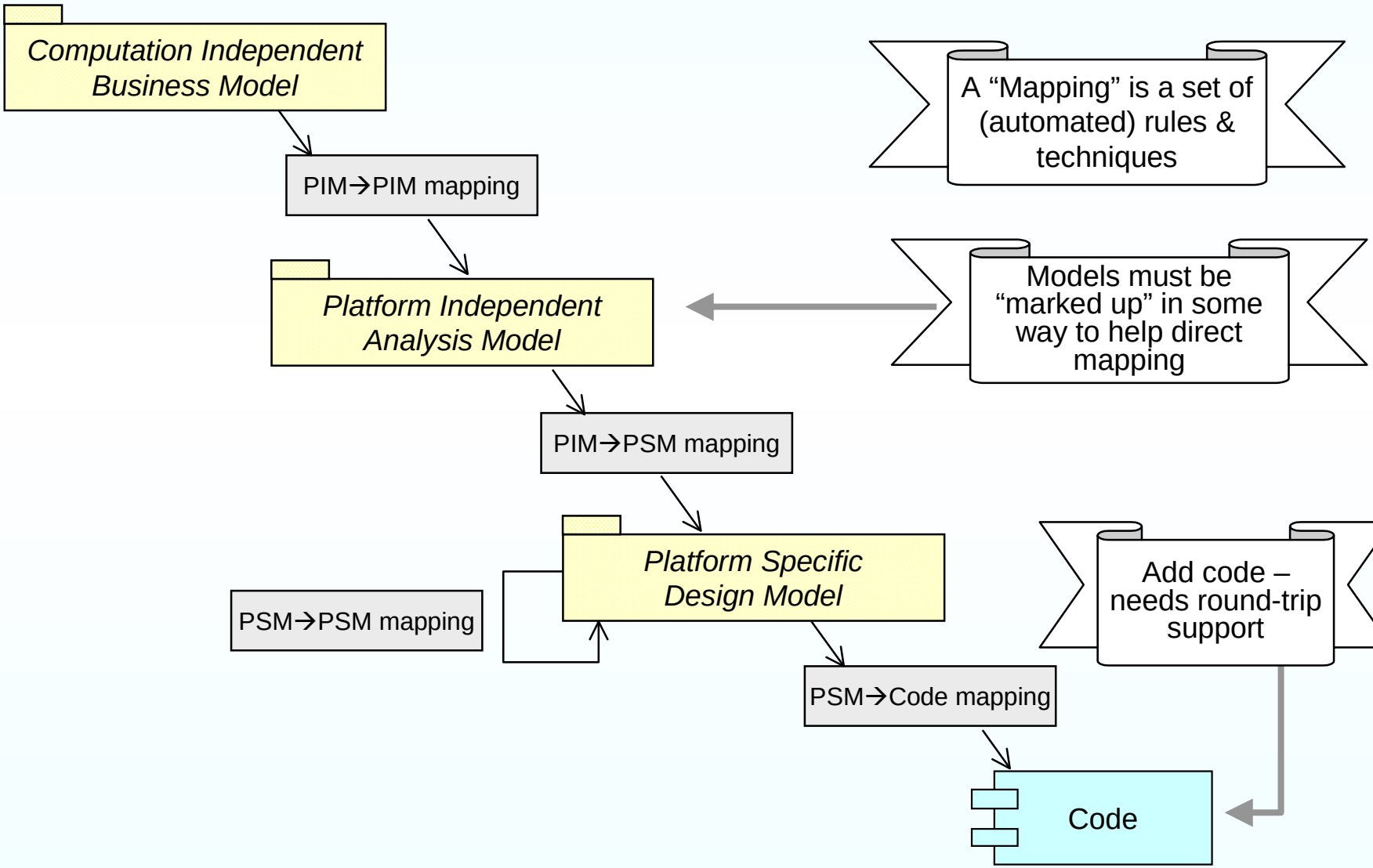
Platform-Specific Model (**PSM**)



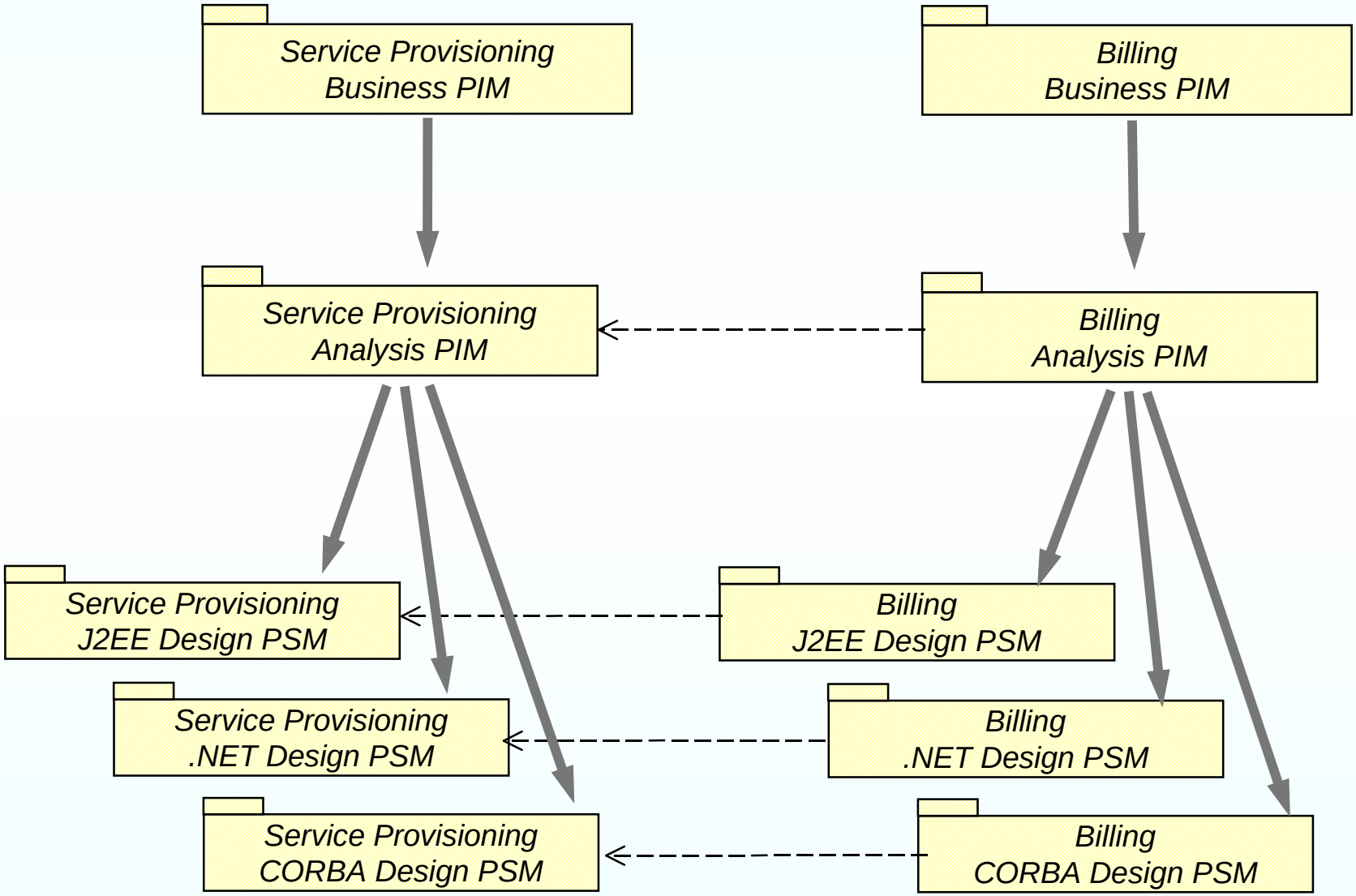
The Importance of Context



Mapping

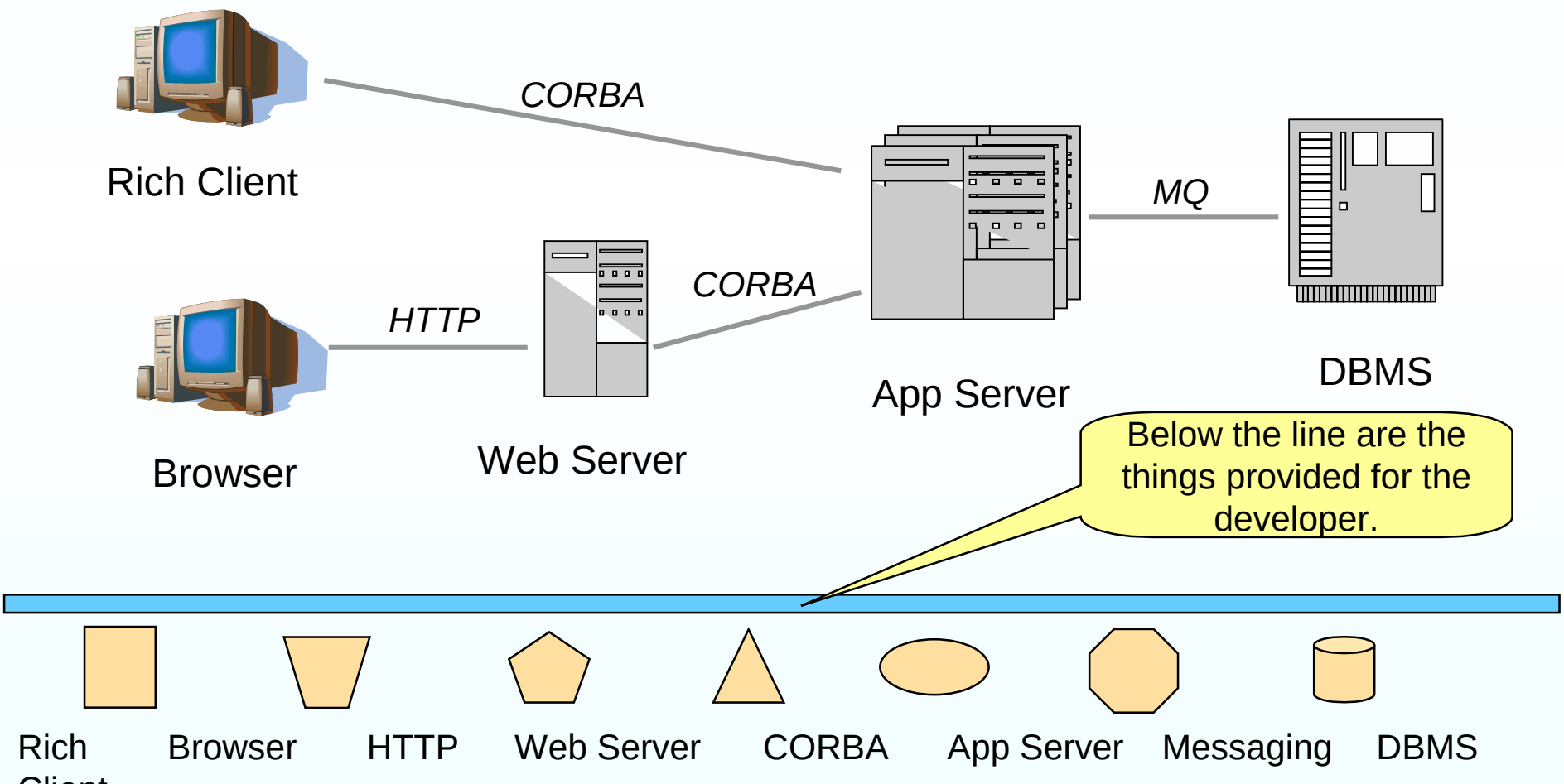


MDA Example (1)

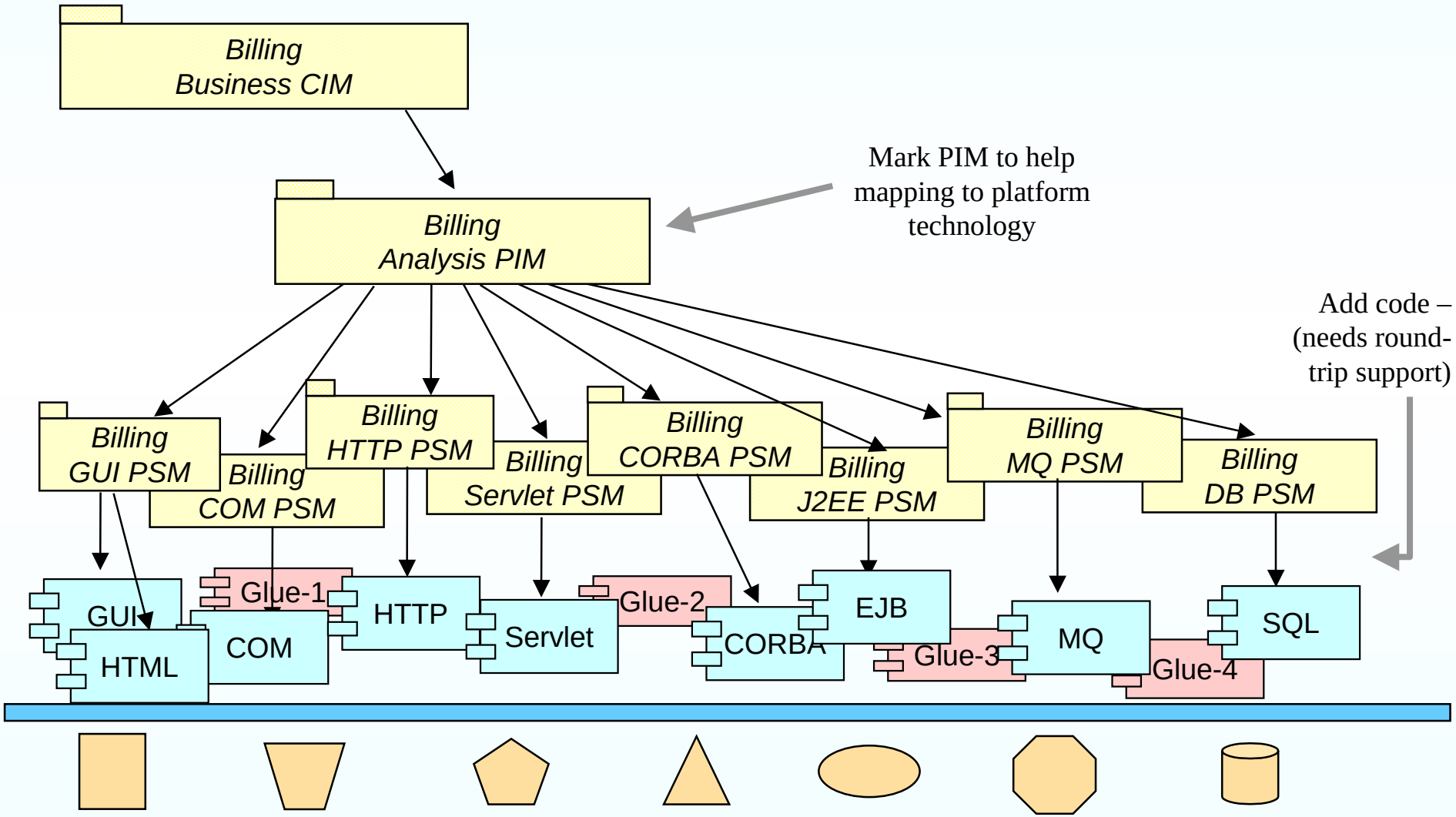


MDA Example (2)

- Target run-time environment:



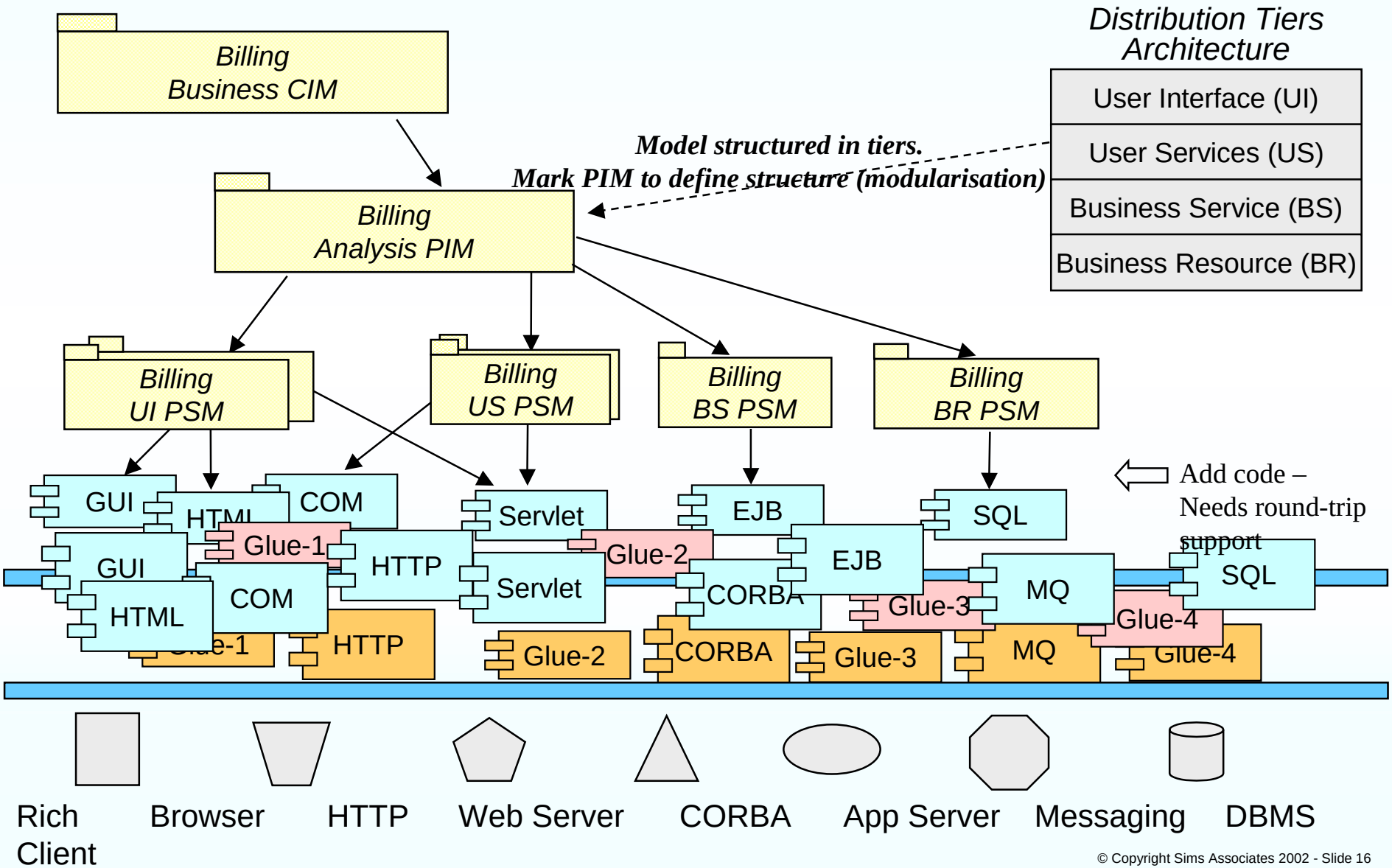
MDA Example (3)



Agenda

- MDA positioning
- MDA essentials
- MDA – the envelope
  - Product Line
  - Architecture
- MDA – the real value
- Next Steps

MDA + Product Line



Process Improvement with Product Line

"When an organization with a single project/product focus moves from CMM level 3 to level 4, the productivity gain is minimal because most of the improvements that can affect a single product or project will already have happened at level 3. …

"However, when the process improvement from level 3 to level 4 includes a shift to product line focus, the productivity increase is very significant. Vu's data indicate as much as a 70% productivity improvement, as well as highly satisfied employees."

Quote by John D. Vu, Technical Fellow and Chief Engineer at Boeing, from *Software Product Lines*, Paul Clemens and Linda Northrop, Addison-Wesley 2002.

Computational Completeness

- A model that can be executed - via code generation or interpretation - is said to be “computationally complete”
- Requires:
  - Action Language for algorithmic logic
  - Computational structure
- The aim is to build computationally complete PIMs
  - All development at the model level
  - Execute the model to test
  - Generate code where necessary
- Component architecture provides an excellent structure

What is a “Component”?

- A managed executable – runs in a middleware “container”
- A pluggable artifact
  - Through the development lifecycle
    - A managed module/package
  - Built with non-component artifacts
- Has programmatic “interface”
  - Local/Remote transparency
- Designed to represent a single “business” concept
  - Throughout the development life cycle
- Composable with other components
  - “Autonomous” not isolated – composed by reference
  - Defined granularities
  - Defined ownership
- Conforms with OO concepts
  - Encapsulation, instantiation, state + behavior
  - Inheritance
    - (where technically possible)
- Can address distributed systems end-to-end

Product Line + CBD Architecture

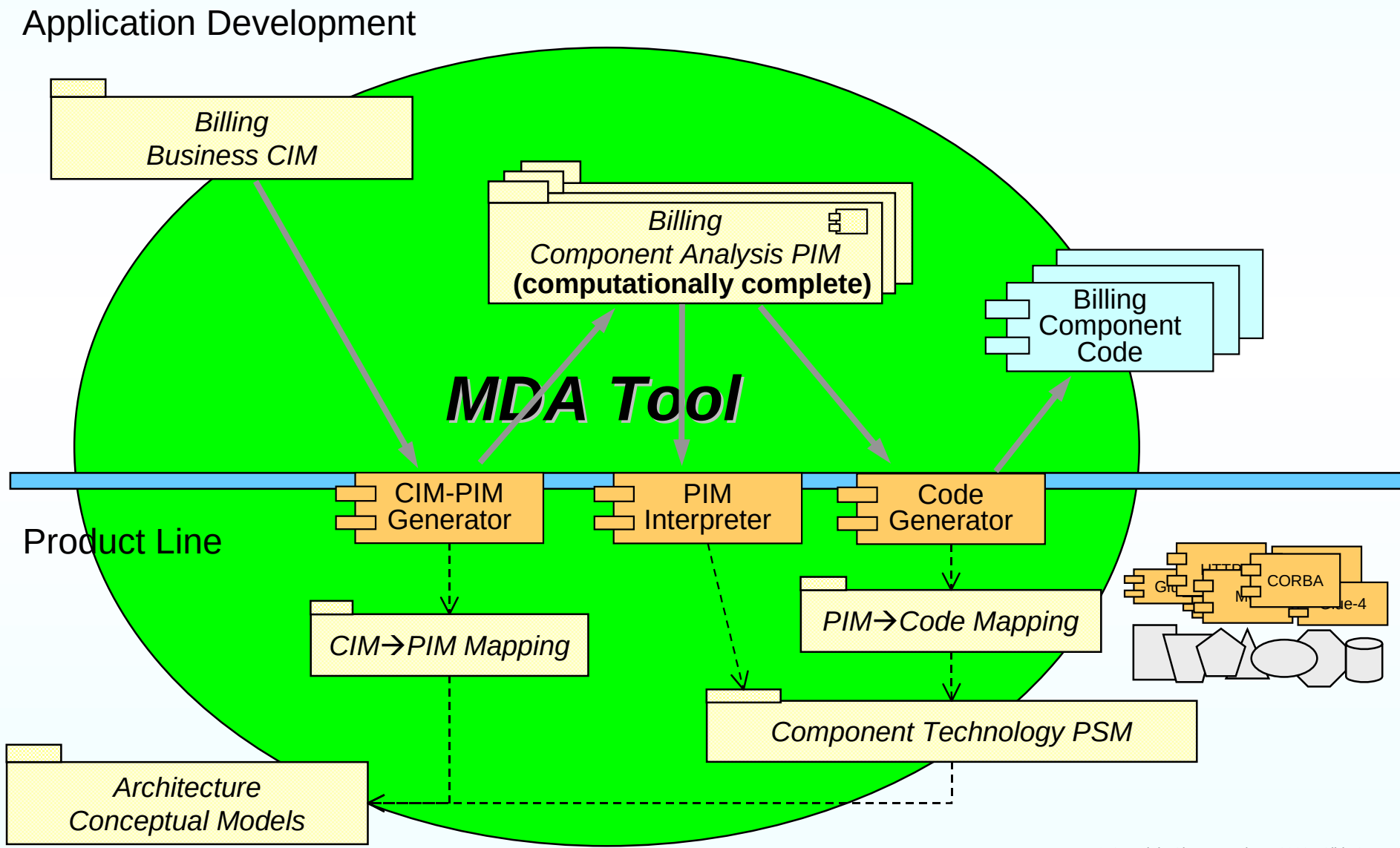
- Architecture (for a given architectural style):
  - Conceptual models that define:
    - Component-oriented application structure concepts
      - Including component granularities and re-usable artifact types
      - Superior modularisation strategy
    - scalability patterns
    - Separation of distributed system (and other) concerns
    - The product line for this architectural style, including design of technology/business separation
  - Development process
  - Skeleton structure for models
  - Mappings (and traceability) between models
    - Plus UML profile for models
  - Design for glue code
  - etc.

  (there are a number of published examples)

The "program" for the MDA tool

- Result:
  - Design for component-oriented application structure
  - Design for a product line
- Evolve the new product line
  - A new level of productivity
- ***Re-use the product line*** for all applications of the same architectural style
- Repeat for each architectural style
  - Re-using common architectural elements

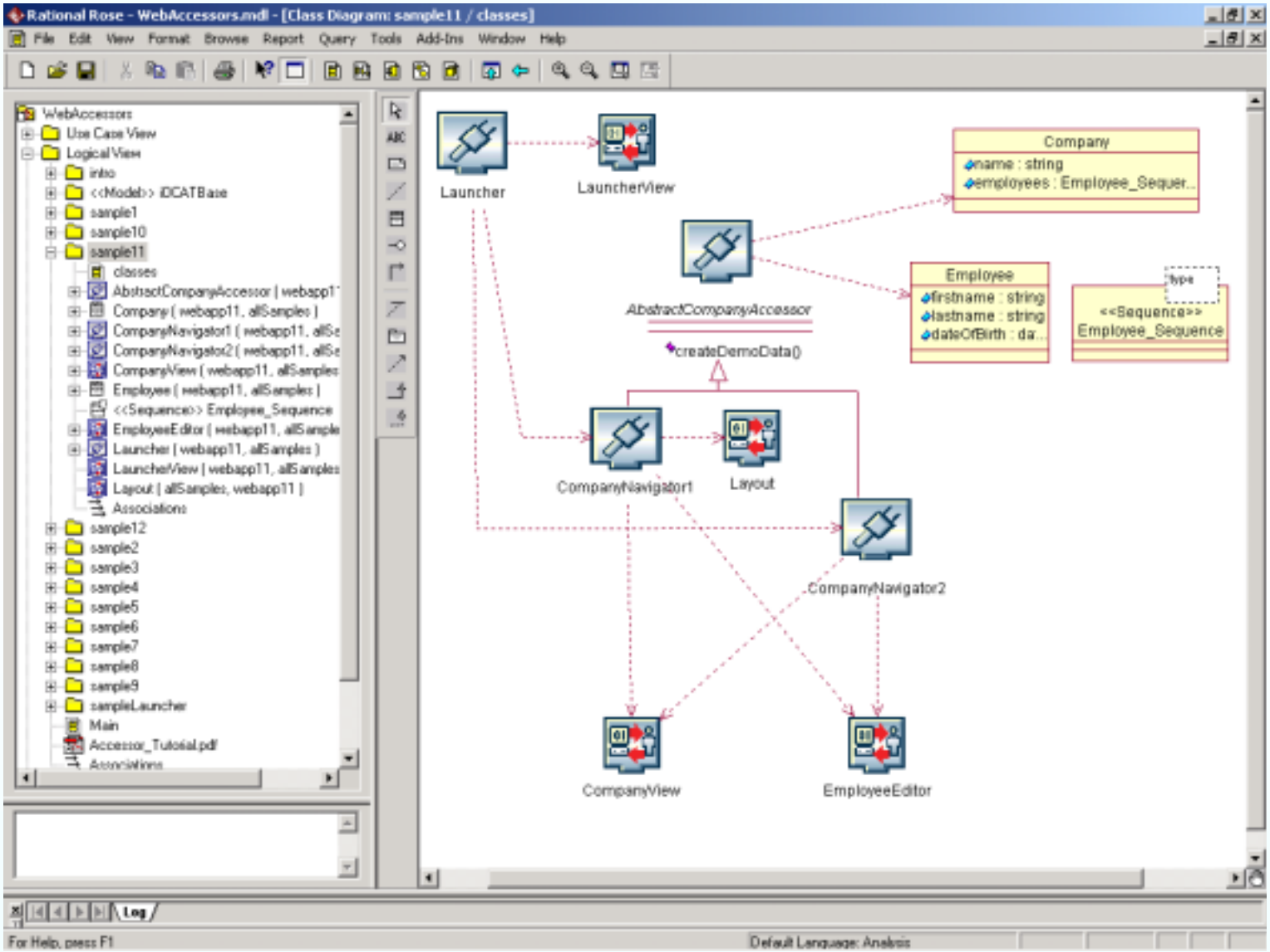
MDA + Product Line + Architecture



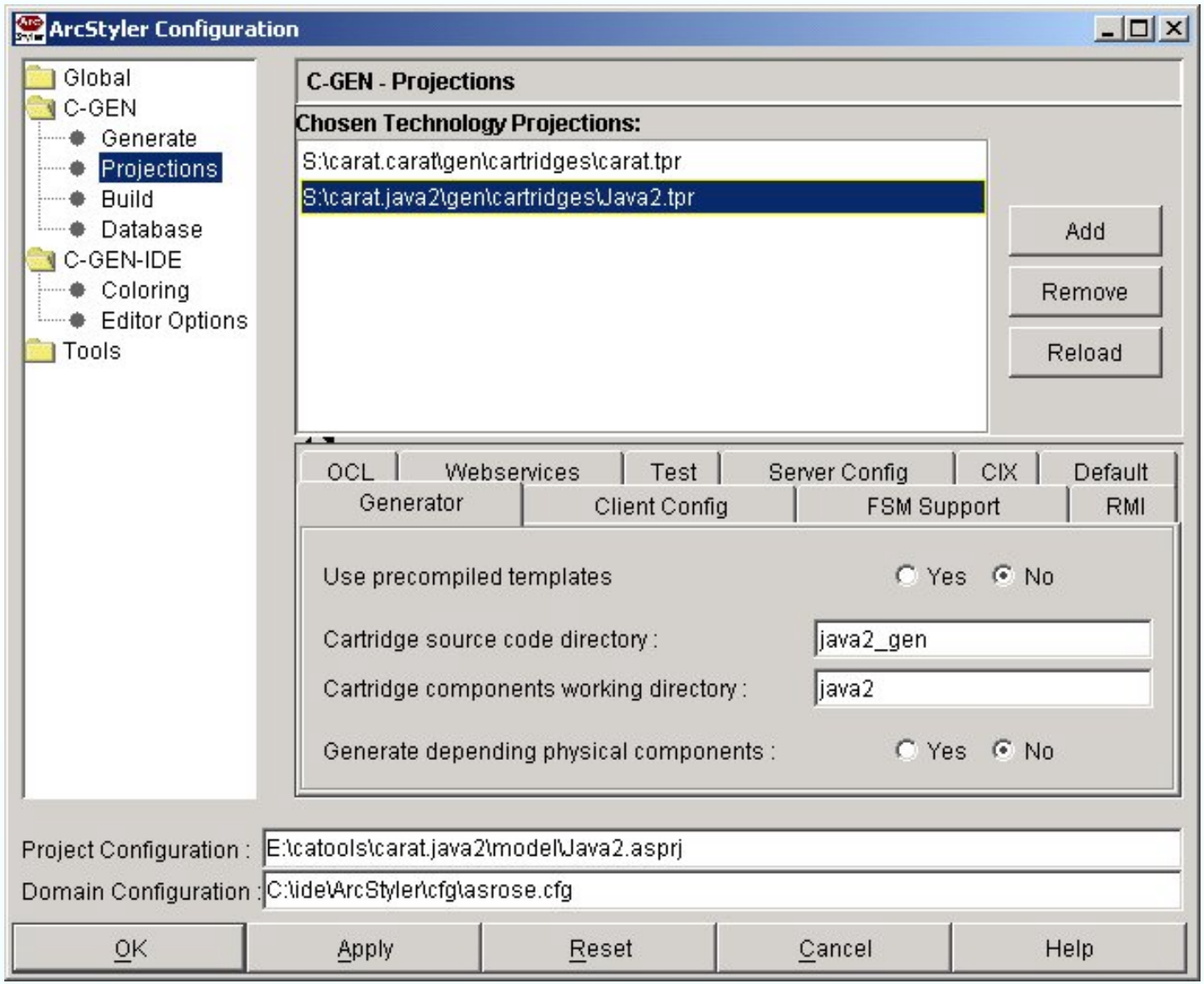
The Perfect MDA Tool (2002)

- Current “standard” capabilities
  - UML modeling
  - Models (or parts of models) can be web-published
  - Code generators for major (and some other) platforms
  - XMI model interchange
- Action Language and/or OCL
- Integrated IDE
- Integration of modules at the PIM level
  - Re-use previously-built PIM modules – e.g. in CBD
- Integration with EIA tools
- CIM/PIM/PSM differentiation available
- Reverse engineering
  - Round-trip engineering
- Executable models
  - in run-time as well as in development
- Architecture support:
  - Metamodel approach, so new UML and other metamodels can be developed and “plugged in”
    - Using MDA of course!
  - Pluggable PIM architectectures
    - E.g. CBD architecture
  - Support for GUI and Data specification consistent with architecture
  - Pluggable generators and mappings if necessary
  - Pluggable “glue” code consistent with architectures supported
- Tool designed using MDA for fast evolution
- Scalable to large development teams
  - Repository support, versioning, sharing, revision-marking, etc.
- Excellent user interaction design
  - E.g. following Alan Cooper’s guidelines (see “The Inmates are Running the Asylum”)
- ???

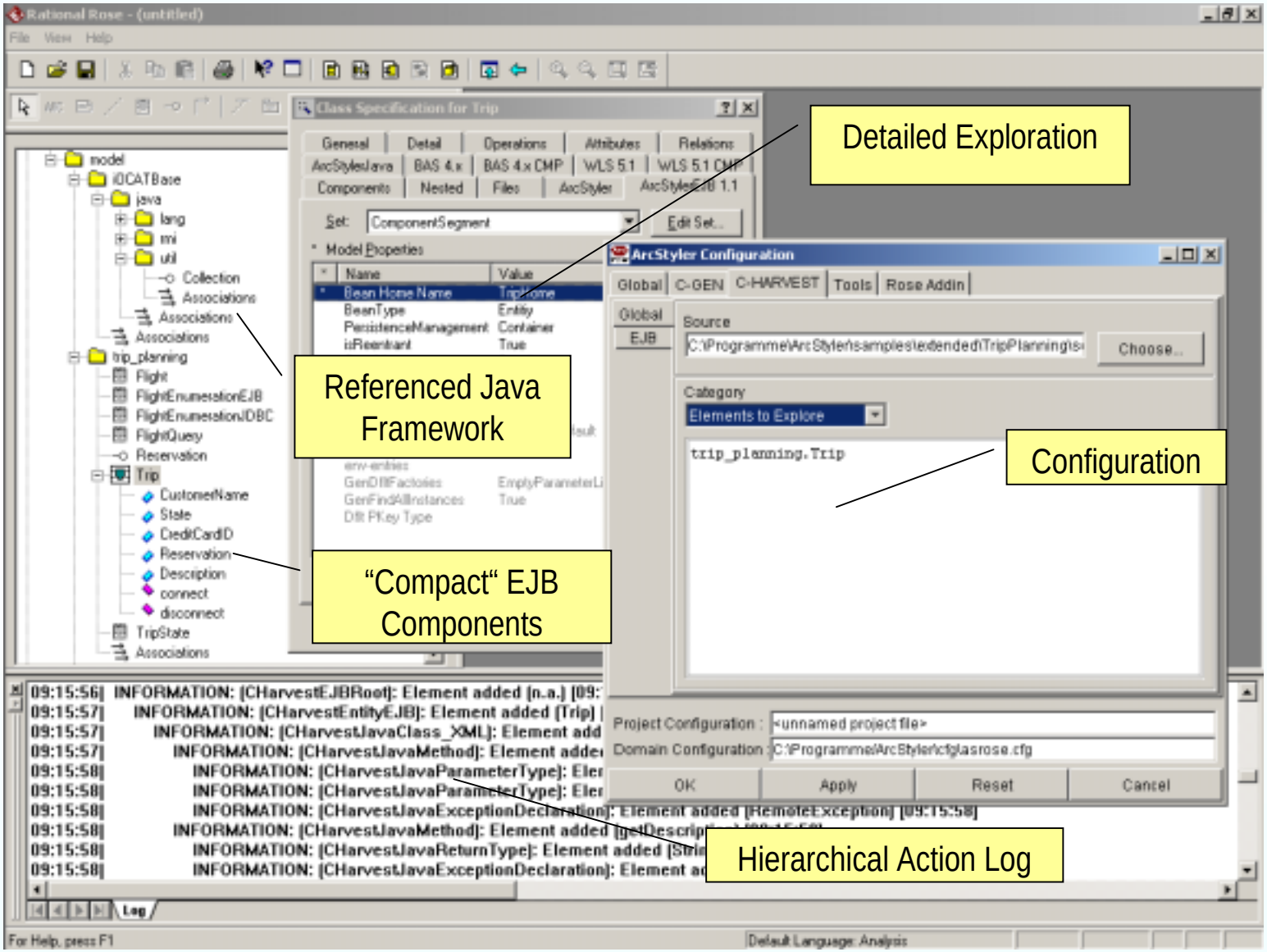
CBD Architecture support



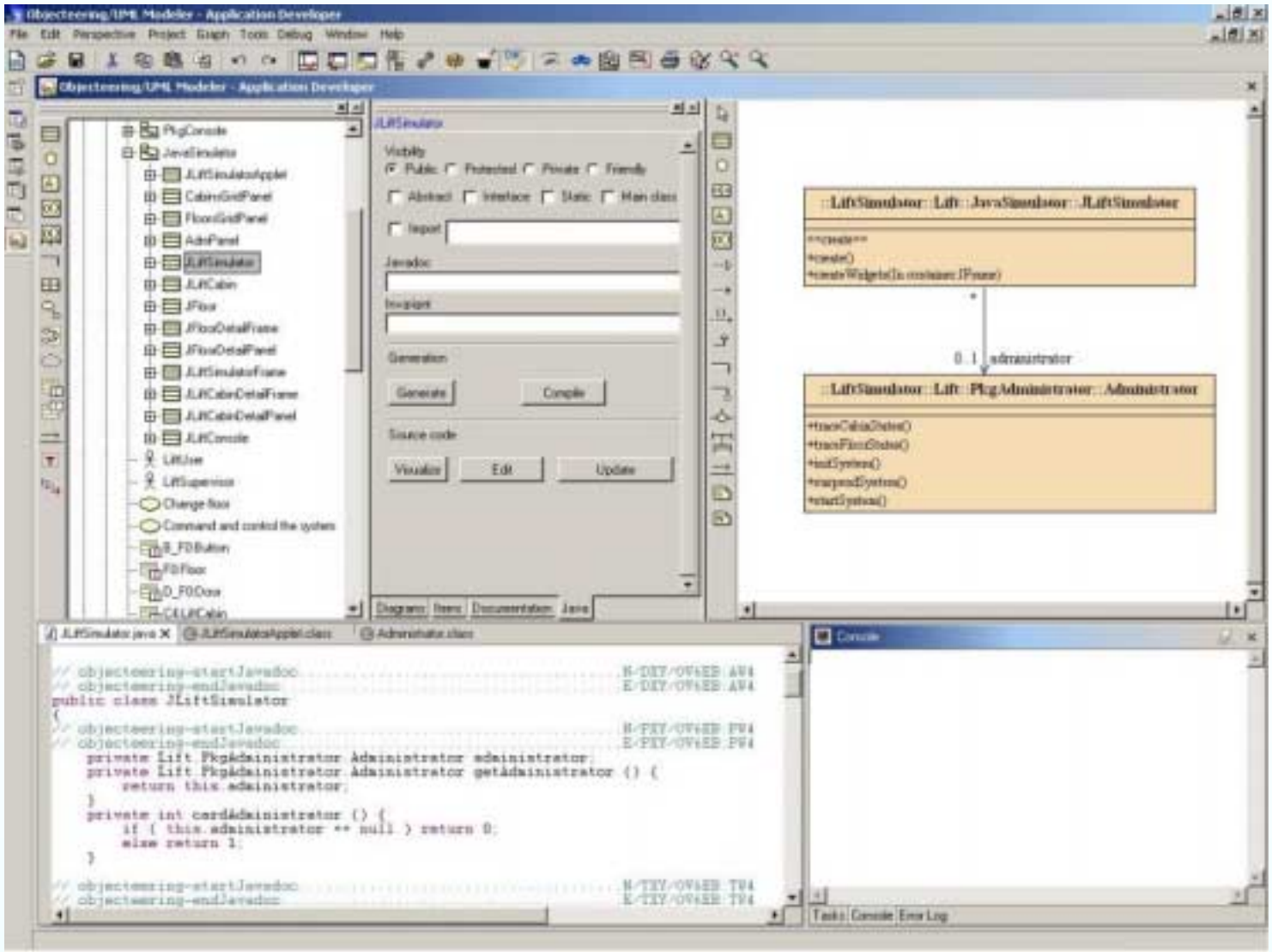
Configuration and Code Generation



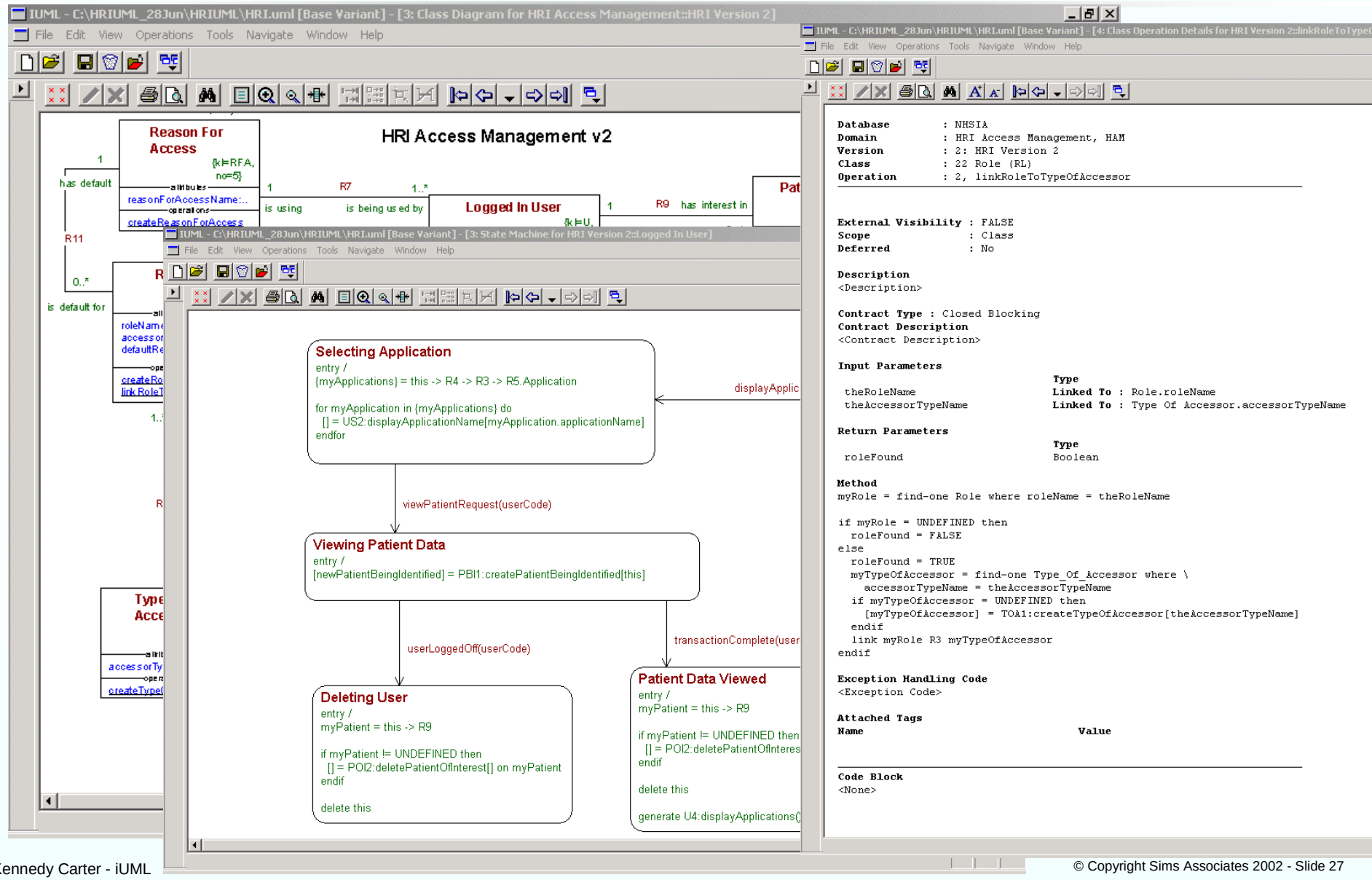
Reverse Engineering



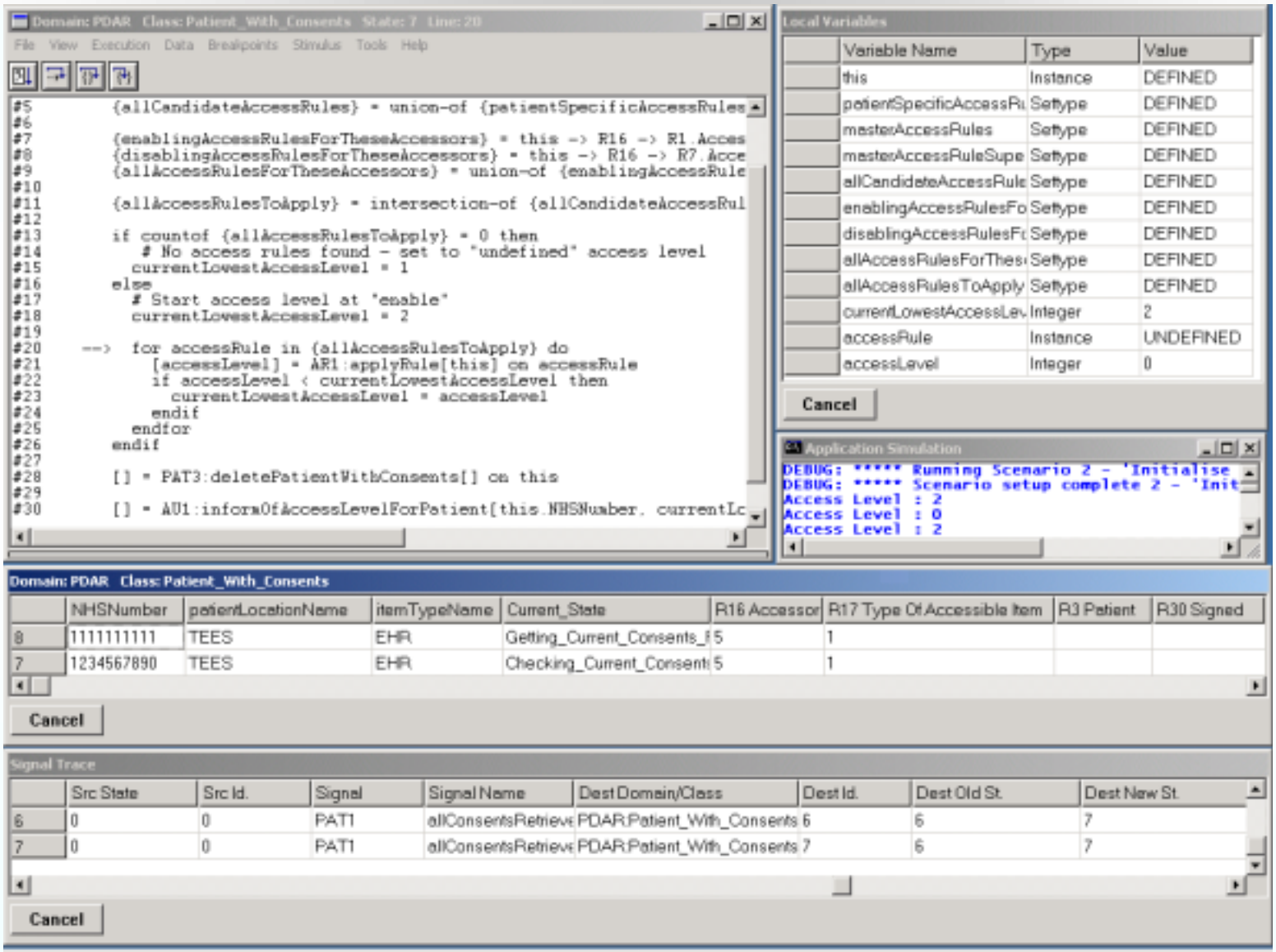
IDE Integration



Action Language



Model execution



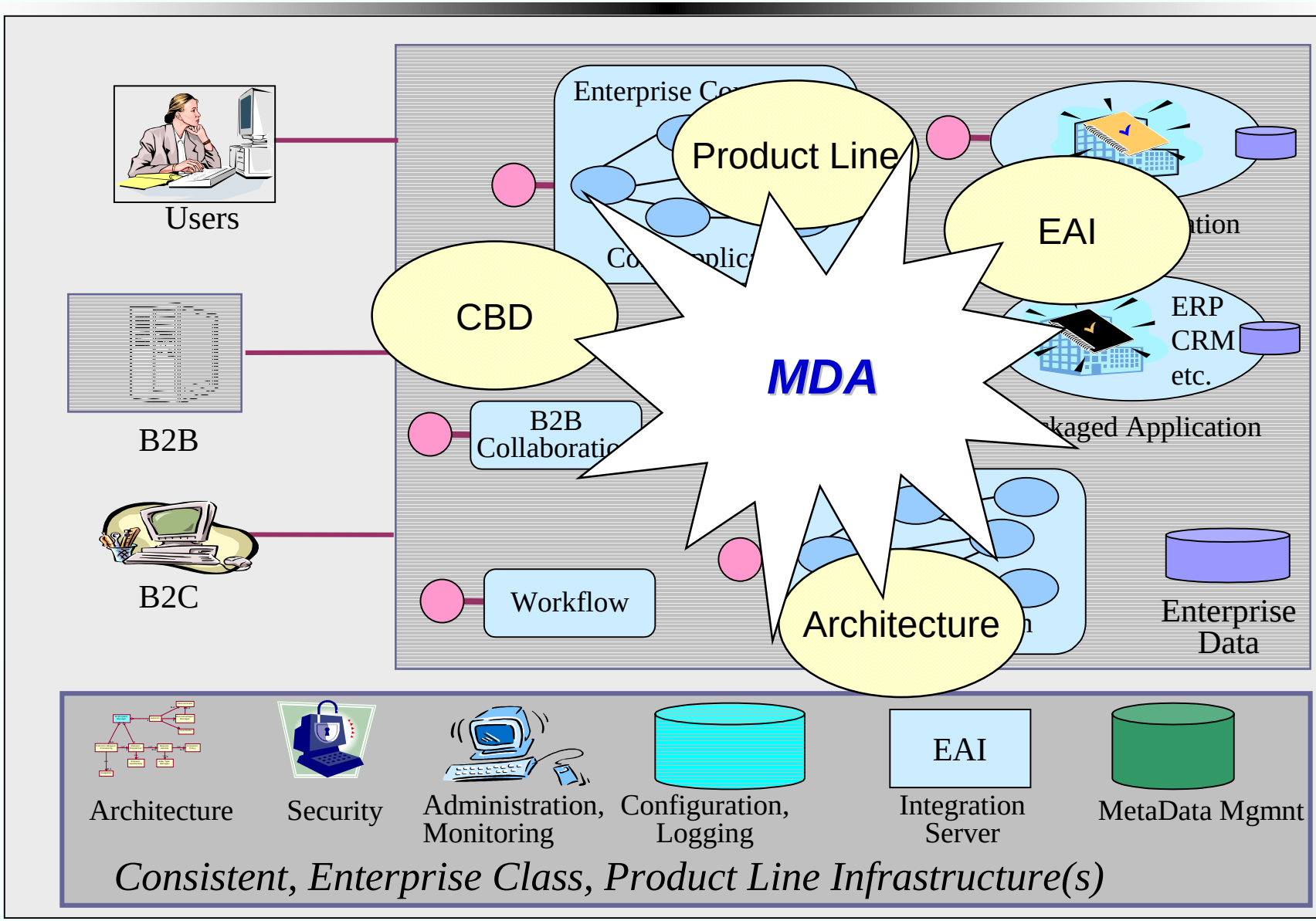
Agenda

- MDA positioning
- MDA essentials
- MDA – the envelope
  - Product Line
  - Architecture
- MDA – the real value
- Next Steps

MDA – The Real Value

- MDA possibilities will force middleware upwards (in simplicity and capability)
  - IT organisations will demand higher-level middleware
    - Because IP can be moved up from code to PIMs
    - Computationally-complete PIMs
    - Easier integration at the PIM level
    - Better management of technical churn
    - Recover PSMs and PIMs from legacy
  - Suppliers will start delivering standard “glue” code
    - E.g. concurrency, transactions, client proxy mechanism, client-side frameworks, etc.
    - The “platform” will become more productive, more developer-friendly
- MDA possibilities will drive tools upwards
  - IT organisations will demand higher-function tools
    - Along lines of “perfect MDA tools”
  - Suppliers will provide standard architectures (probably component-based)
    - CBD architecture a priority
    - Architectures will grow in scope and applicability
- Result …

Managing Complexity

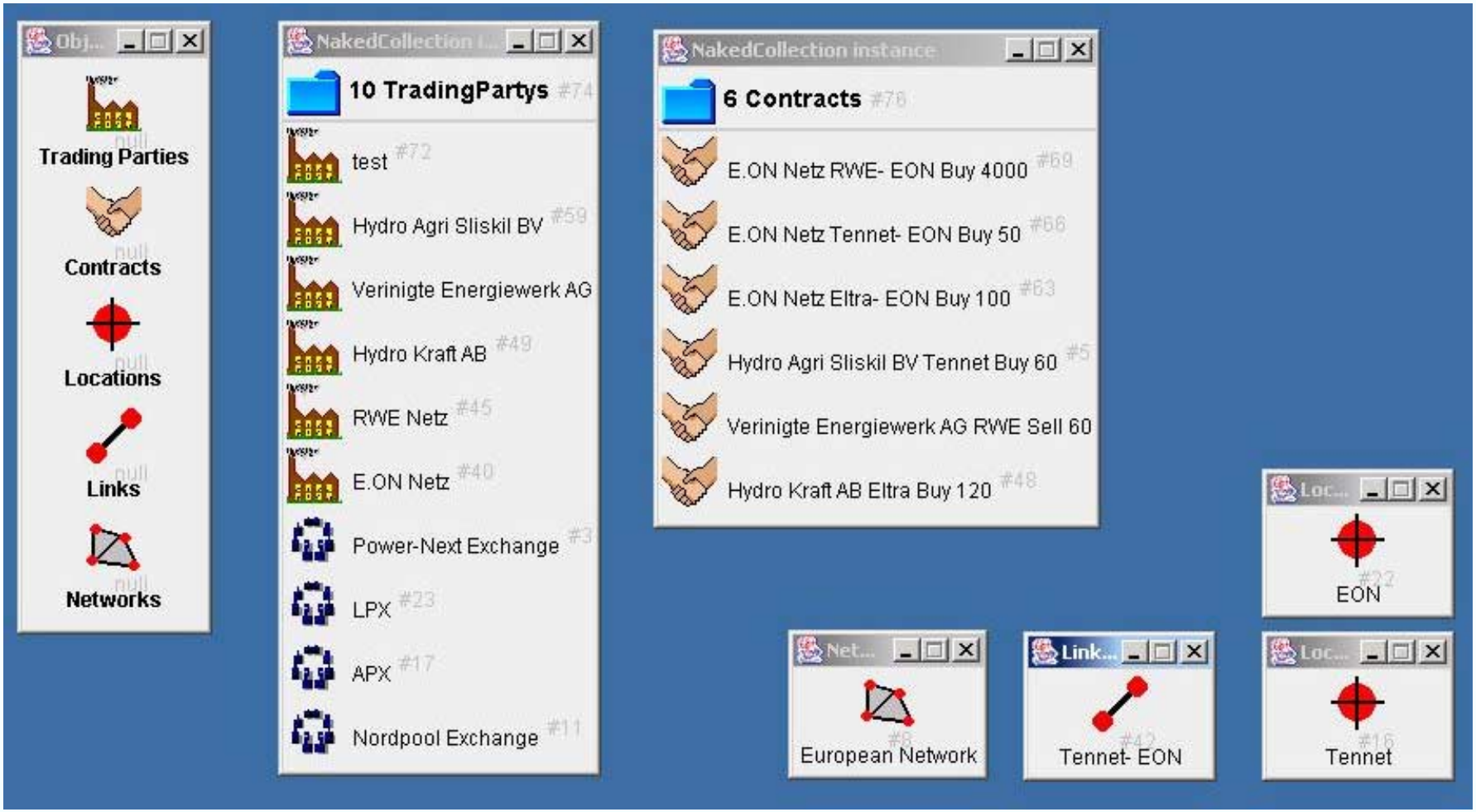


“Naked Objects”

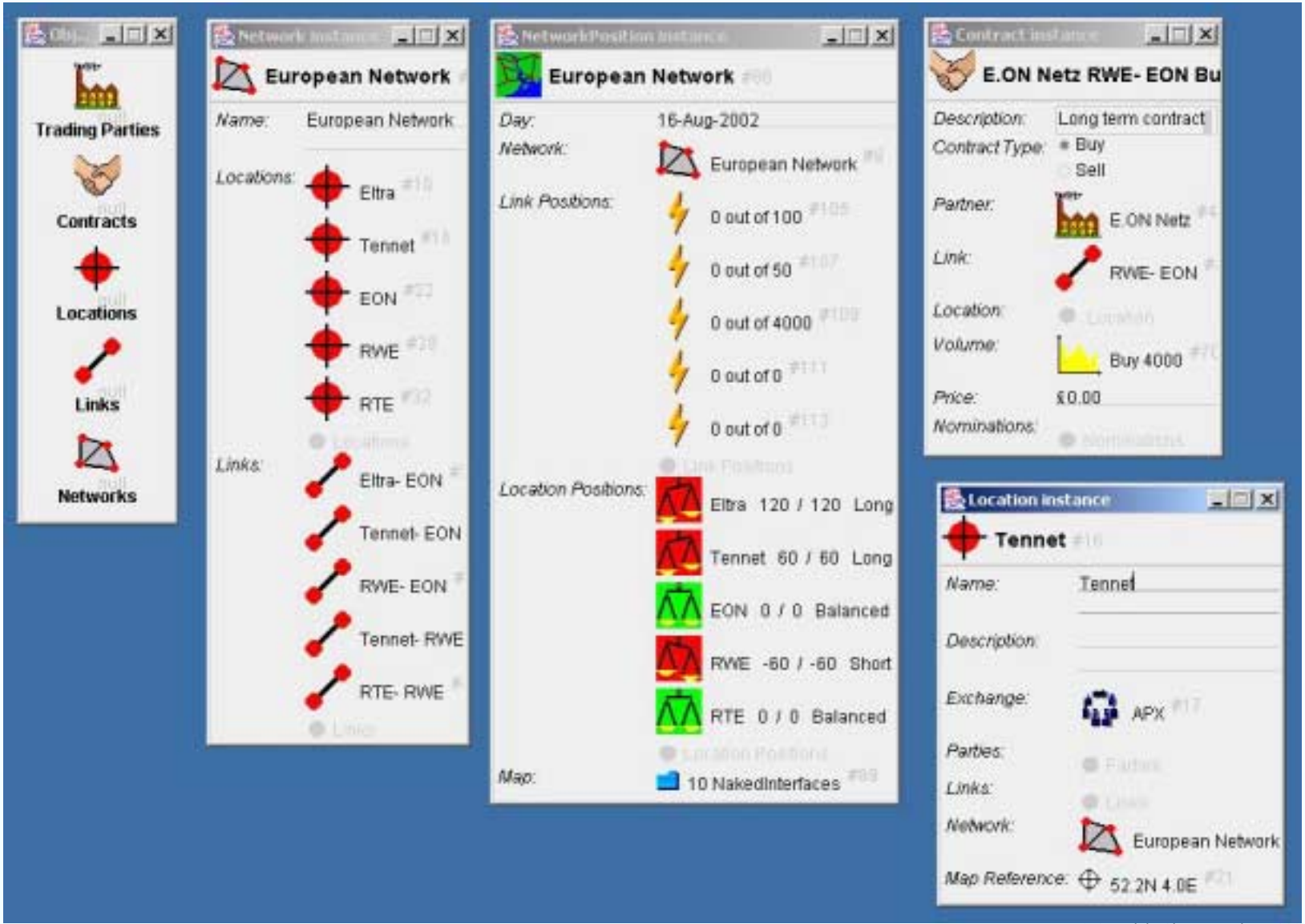
- “‘Naked objects’ are core business objects, such as Customer, Product, and Order, that show directly through to the user, rather than being hidden behind the menus, forms, process-scripts and dialogue boxes that make up most user interfaces.” (www.nakedobjects.org)
- Used to build working prototypes while business modelling…
- … and then used to implement the system!
- GUI is automatically-generated

Richard Pawson and Robert Matthews

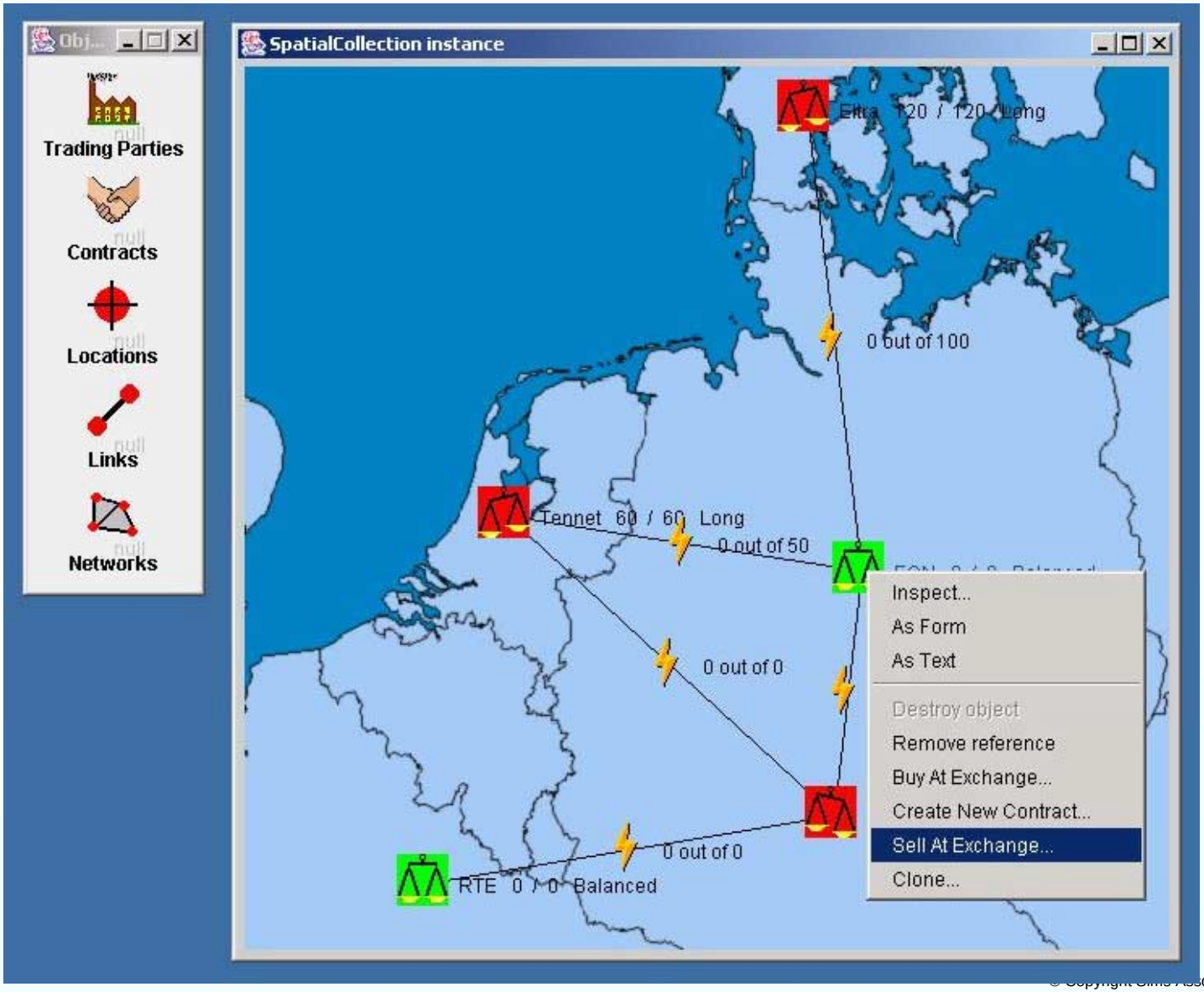
Naked Objects Example (1)



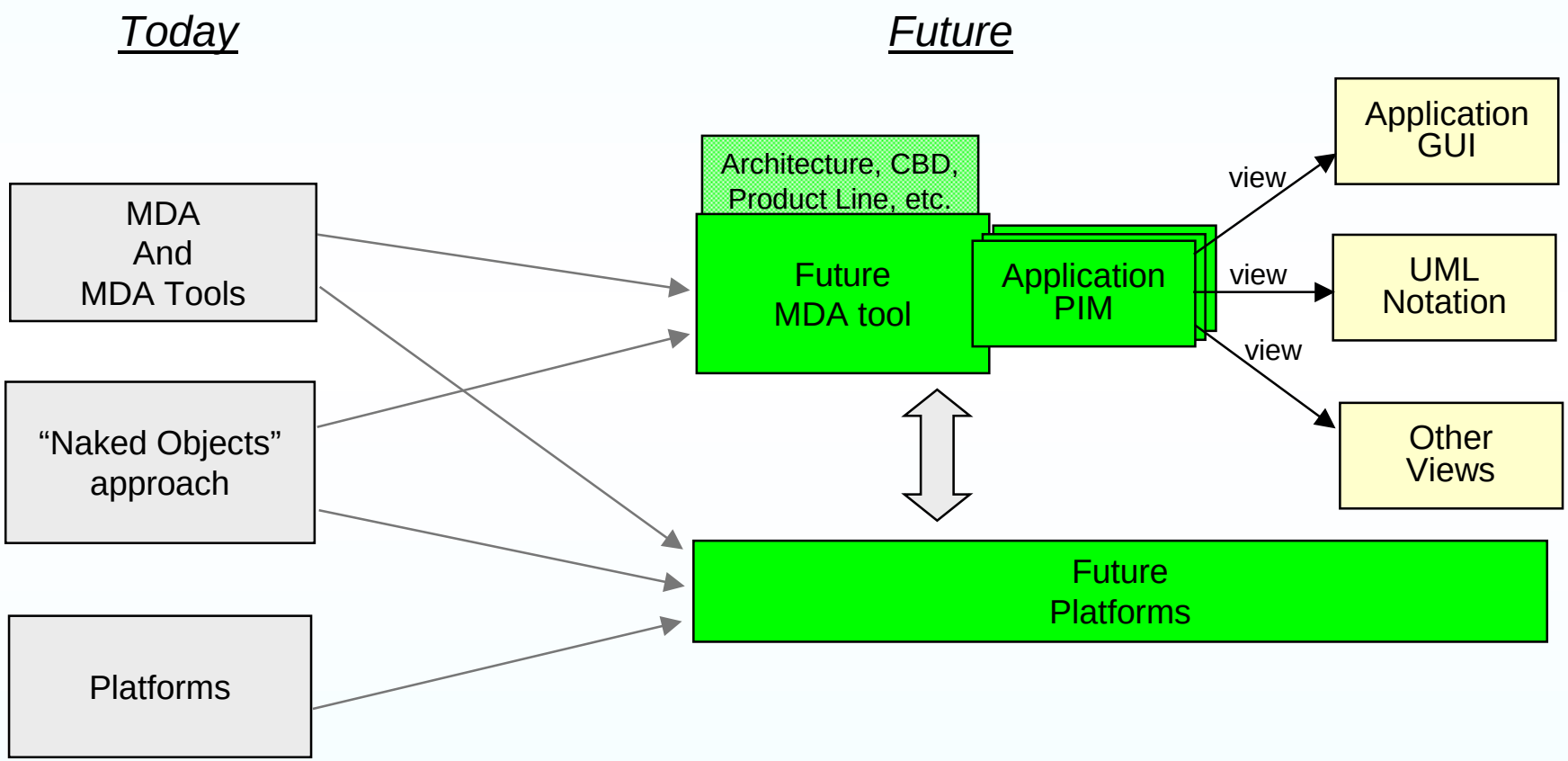
Naked Objects Example (2)



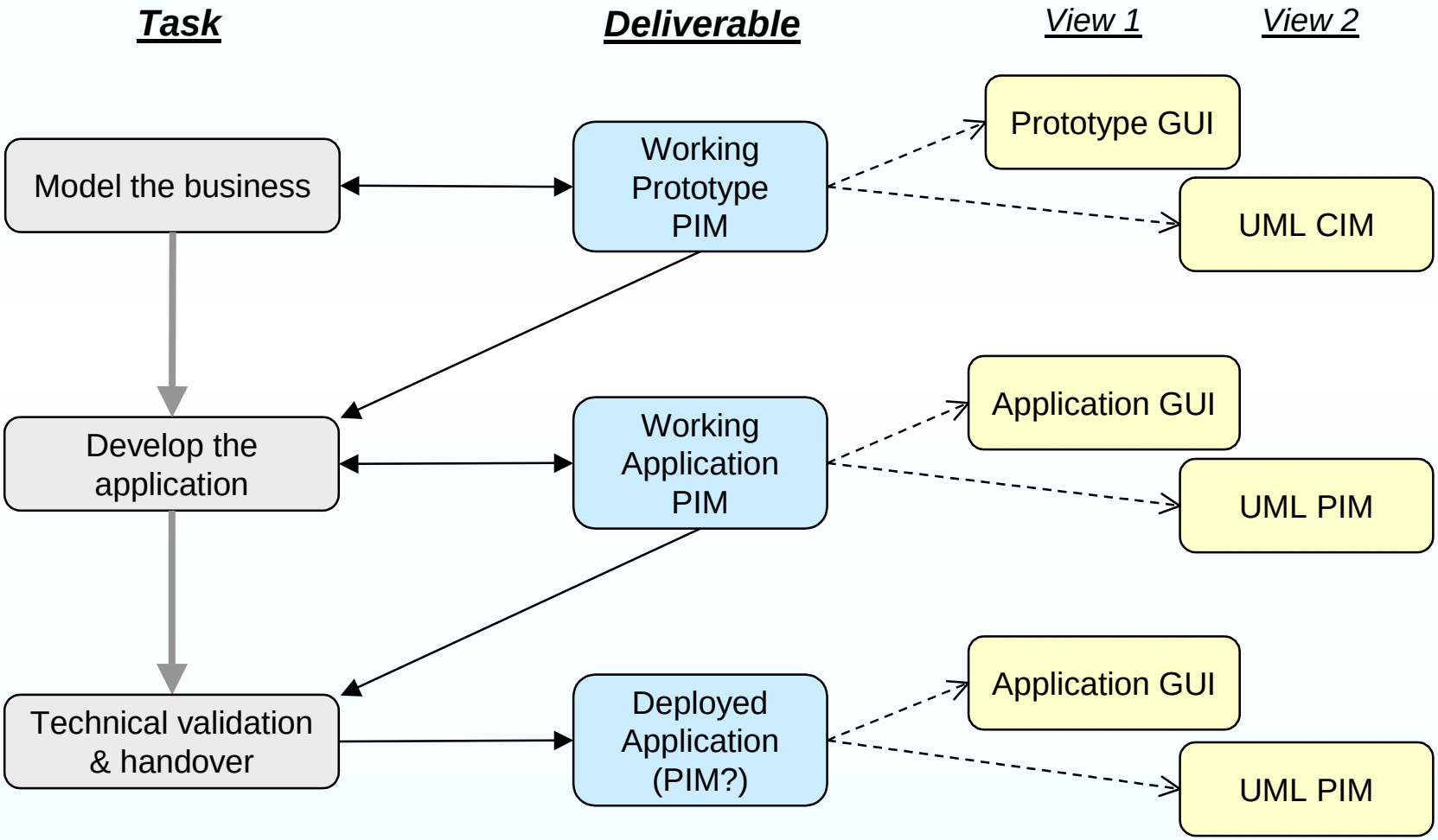
Naked Objects Example (3)



Possible MDA Evolution



Future MDA Development Process?



Agenda

- MDA positioning
- MDA essentials
- MDA – the envelope
  - Product Line
  - Architecture
- MDA – the real value
- Next Steps

Next Steps

- Tomorrow:
  - Schedule an MDA planning meeting
- Next week:
  - Review the current development situation
  - Define the goal
  - Plan the first step
  - Define process for evolution to the goal *while continuing with development schedule*
- Next month – Start!
  - Launch the first MDA project, using an MDA tool
    - Real application
    - Architecture capture
    - Product Line initiation
  - Manage the evolution to the goal
- ☺ … call us if you need help!


Sims Associates

OPEN IT

MDA

-

The Real Value

Oliver Sims

Sims Associates

oliver.sims@simsassociates.co.uk