Modeling the CoCoME with KobrA

Monika Juhasz
August 1st, 2007

Chair of Software Engineering
University of Mannheim

UNIVERSITY OF MANNHEIM

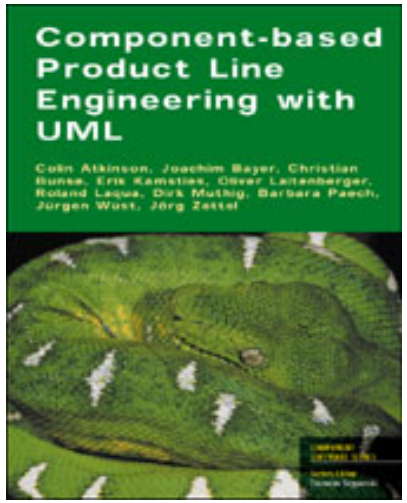
Chair of Software Engineering

- Leader: Prof. Dr. Colin Atkinson
- Research focus
  - Component Development
  - Multi-Level Modeling
  - Component Discovery
  - Mobile Business
- Current projects
  - Aristaflow
    - component-oriented development of adaptive process-oriented enterprise software
  - Mobile Business
    - development of a generic software platform for mobile devices
  - ECOMODIS
    - efficient component-based development of dependable systems

Contents

KobrA

- UML-based modeling method for component-based systems
- developed at Fraunhofer Institute for Experimental Software Engineering
- KobrA = „**Ko**mponenten **b**asierte **A**nwendungsentwicklung“ (German for component based application development)
- here: updated version of KobrA



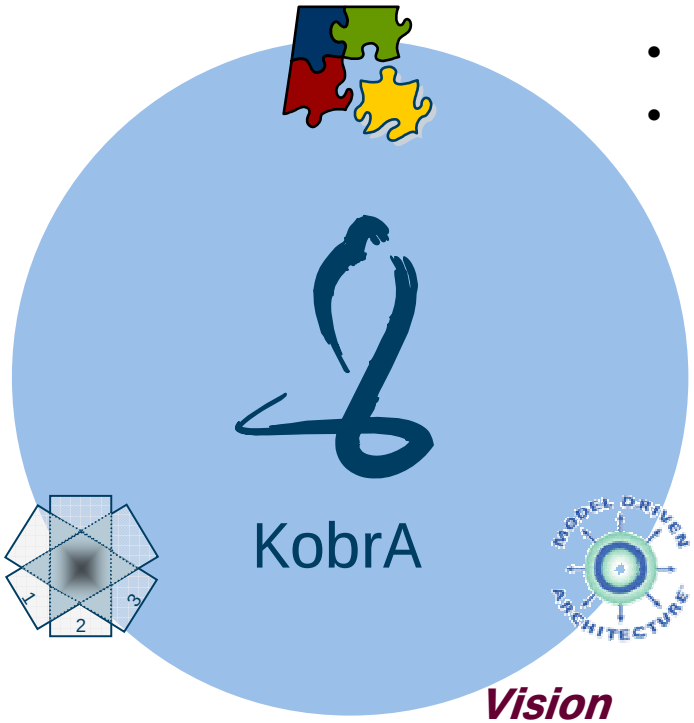
2002

Reuse technologies

Component-based Development (CBD)

Vision

- Assemble applications from prefabricated parts
- COTS component market
- Web Services



Vision

- Development activities oriented around product families
- Manage commonalities and variabilities

Product-Line Engineering (PLE)

Model-Driven Architecture (MDA)

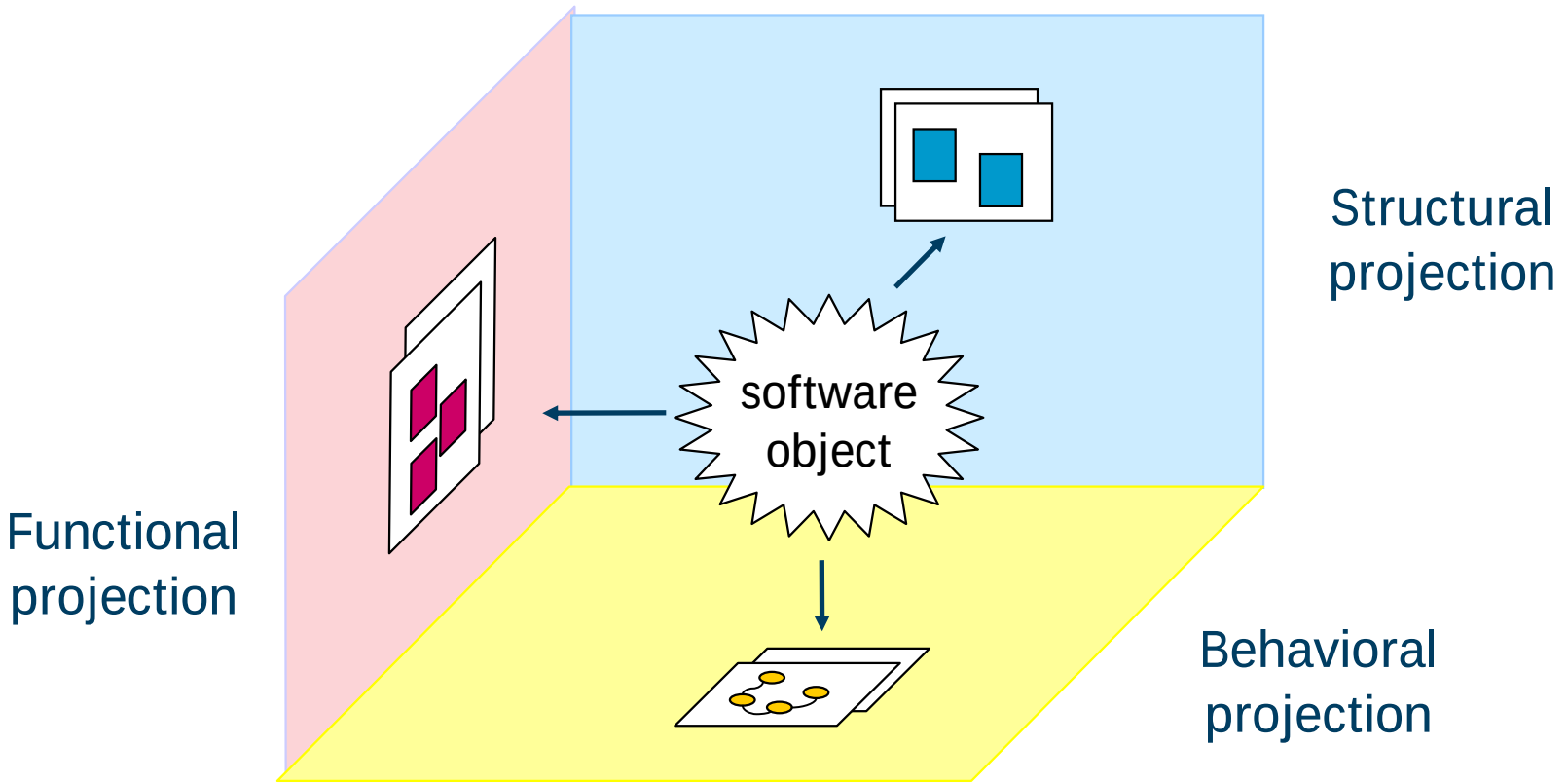
Vision

- Capture core software assets as platform-independent models (PIMs)
- Automatically map PIMS to platform-specific models (PSMs)

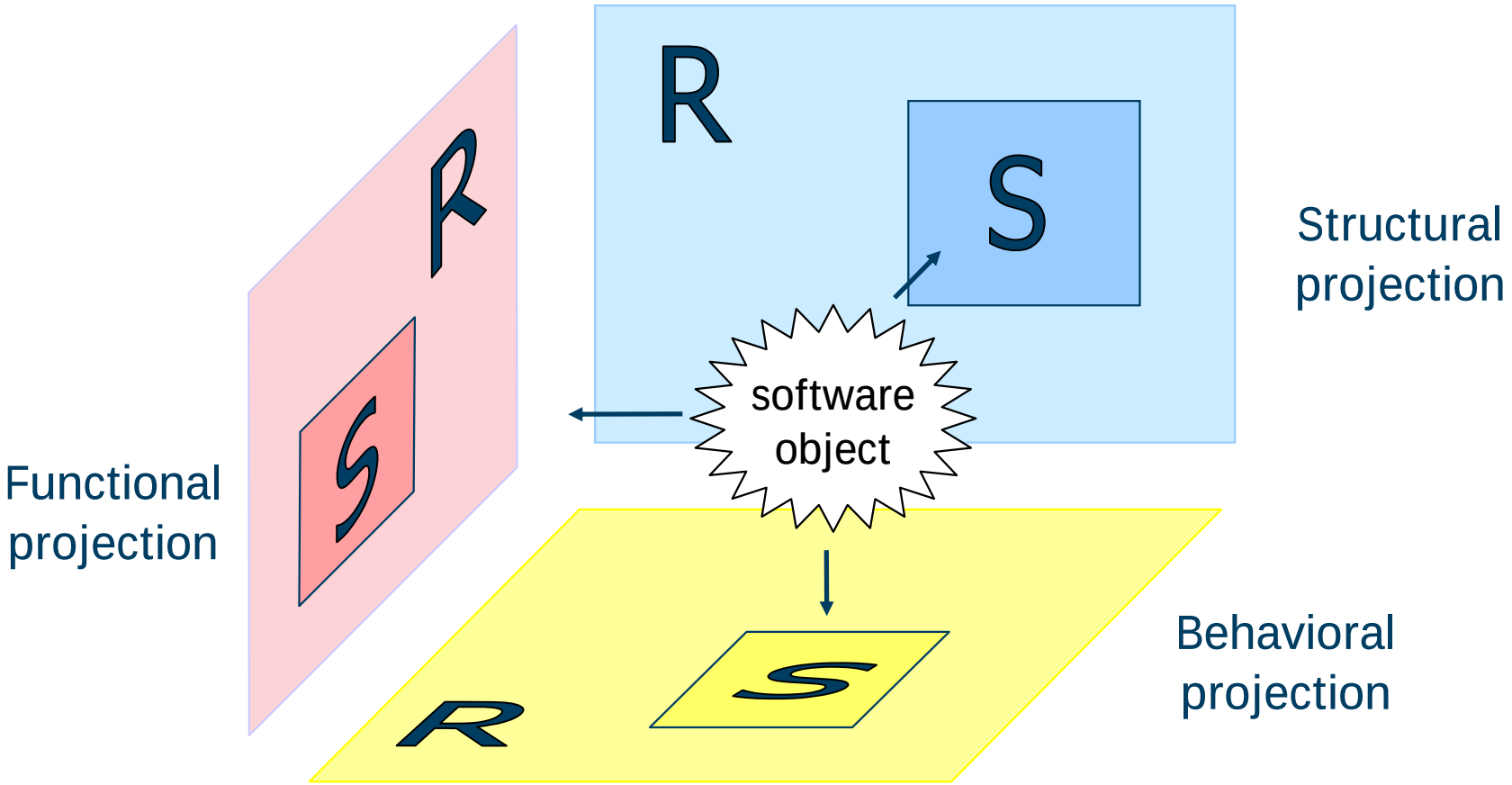
Contents

- KobrA – an overview
  - Reuse technologies
  - Core concepts
    - Separation of Projections
    - Separation of Specification and Realization
    - Separation of Process and Products
    - Separation of Development Dimensions
    - Modeling Principles
- Modeling the CoCoME
- Conclusion

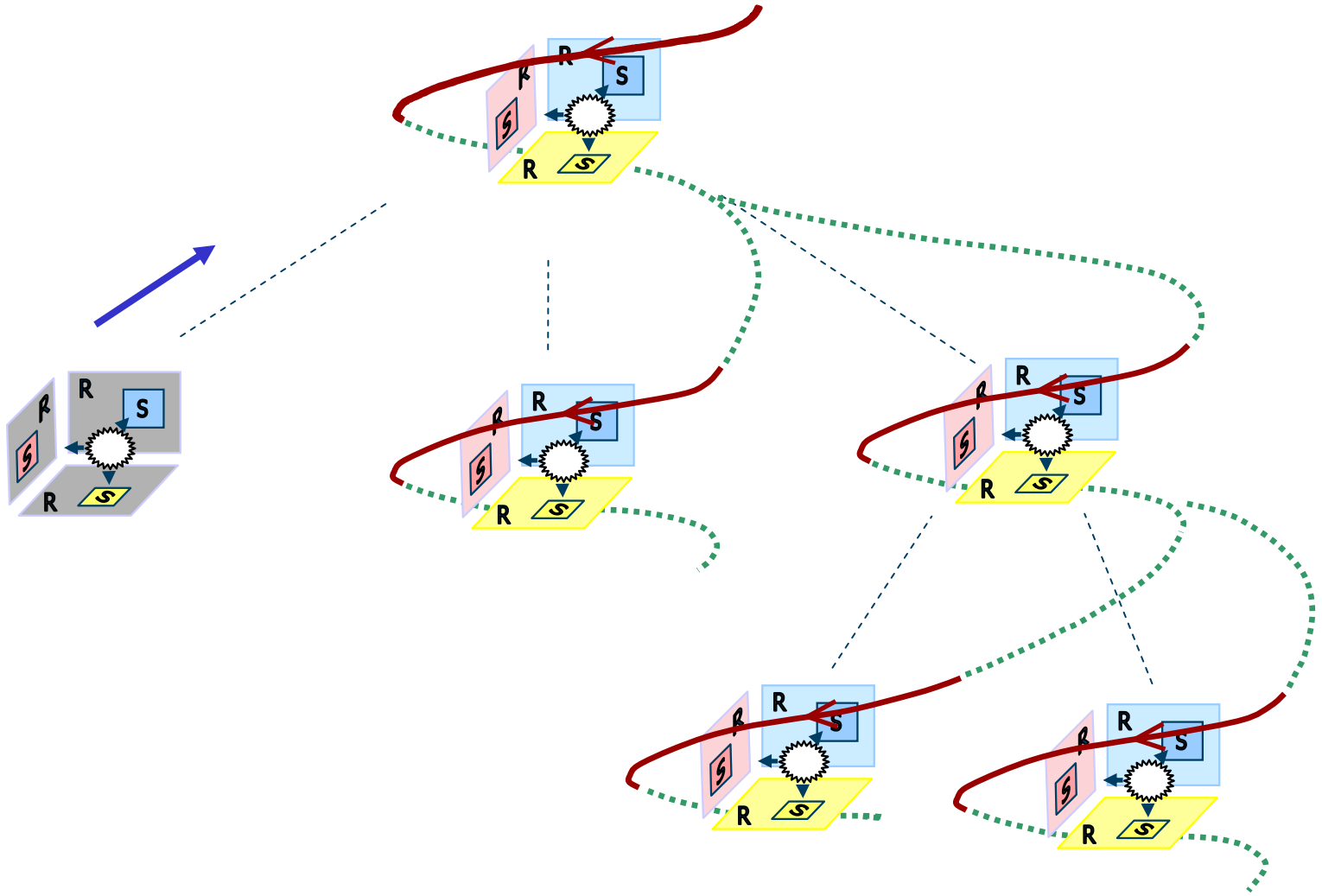
Separation of Projections



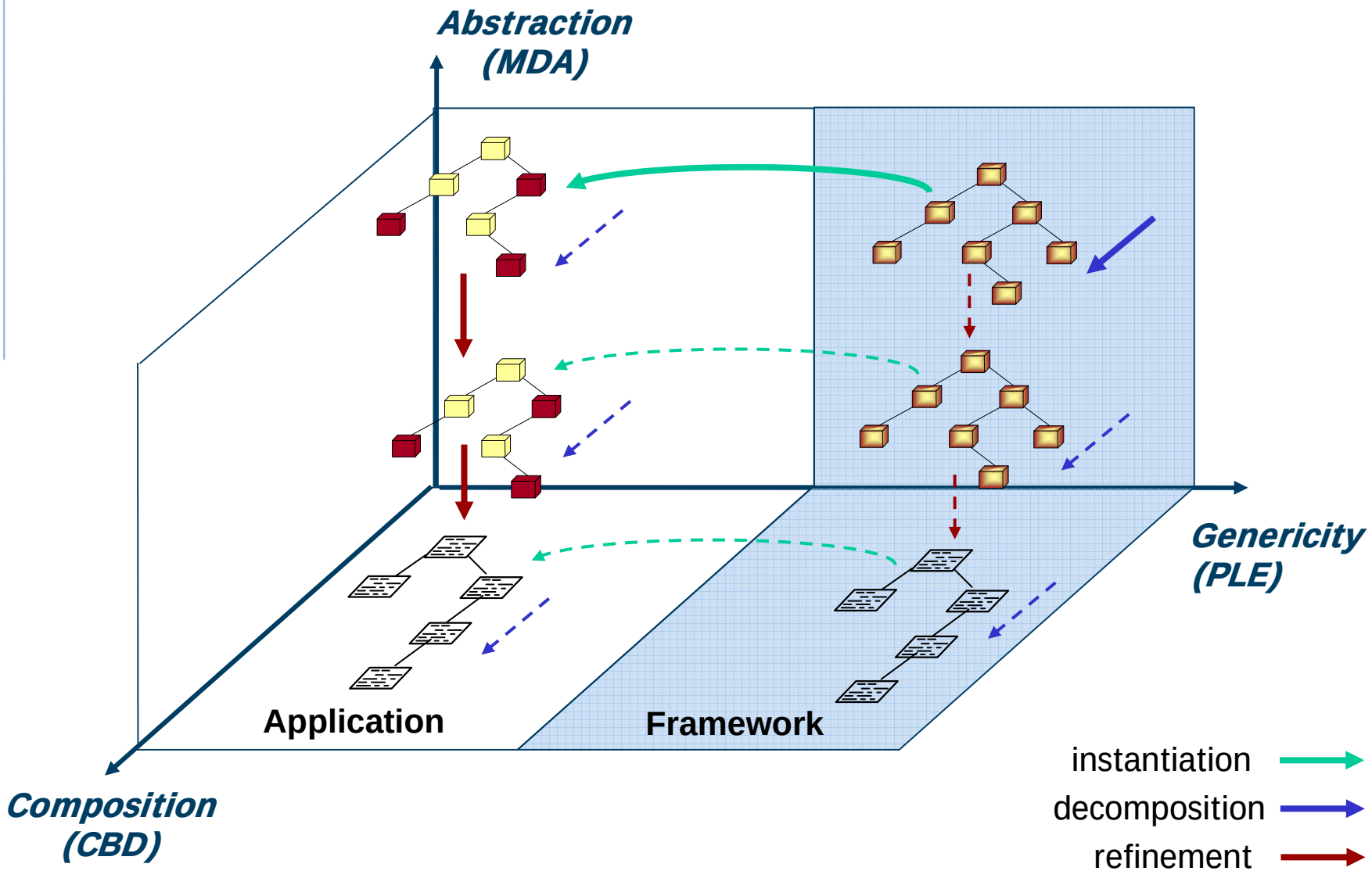
Separation of Specification and Realization



Separation of Product and Process



Separation of Development Dimensions



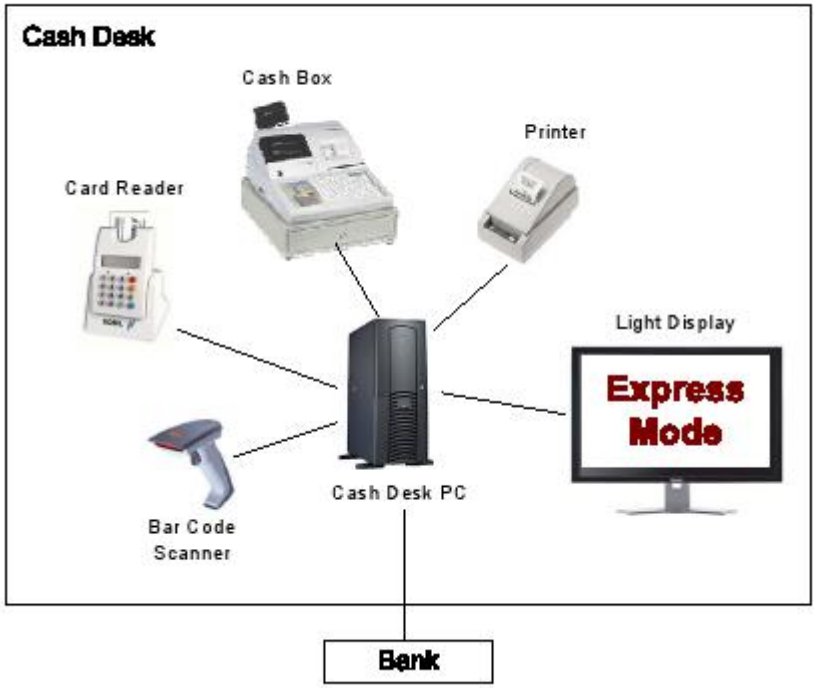
Modeling Principles

- Uniformity
  - **all** behavior rich elements should be viewed as components, including (sub)systems
- Locality
  - all models are views
  - all views should be local to a component
- Parsimony
  - minimal set of concepts (no redundancy)

Contents

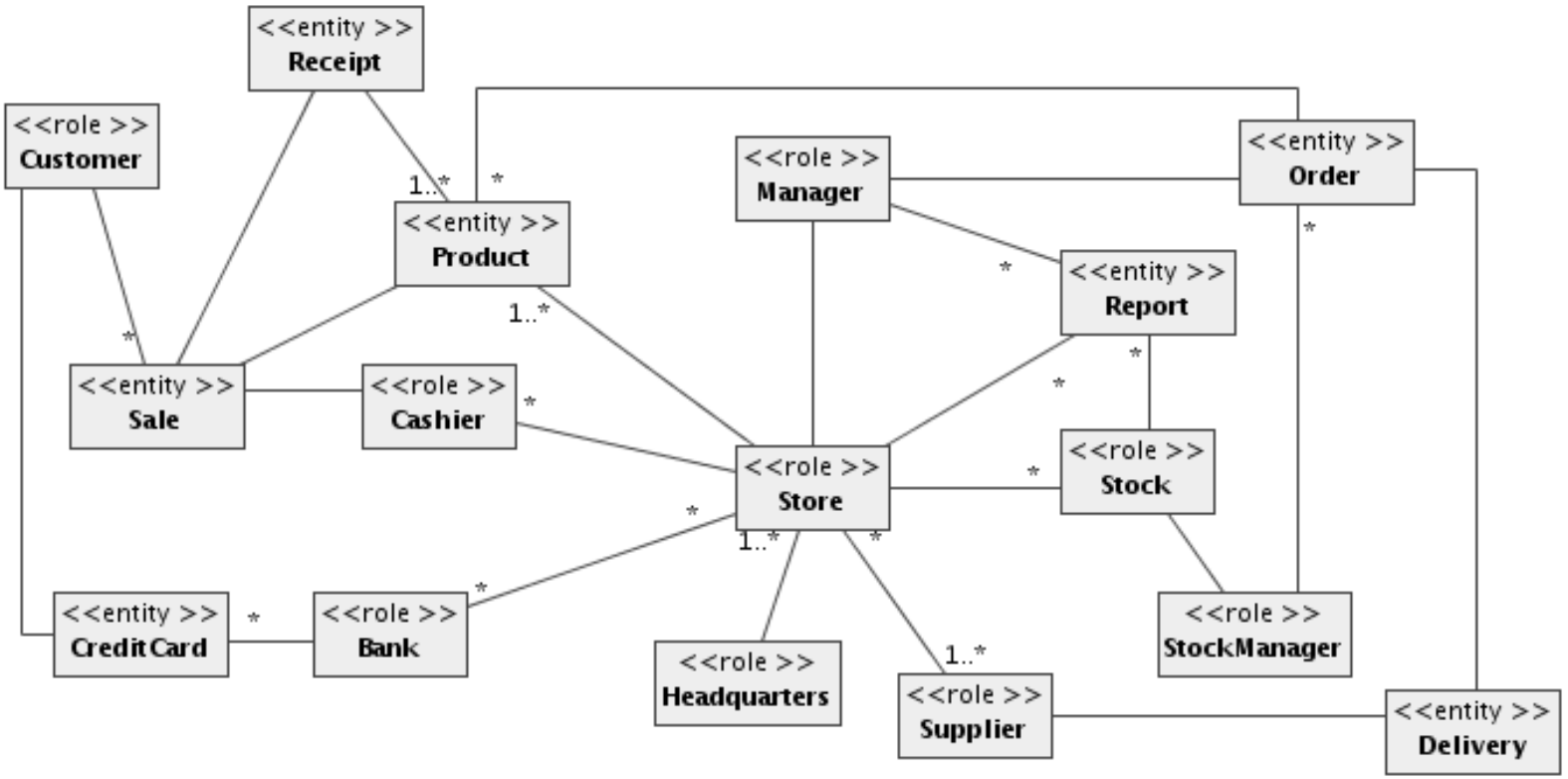
- KobrA – an overview
- Modeling the CoCoME
  - Computation Independent Model
  - Platform Independent Model
  - Platform Specific Model
- Conclusion

Modeling the CoCoME



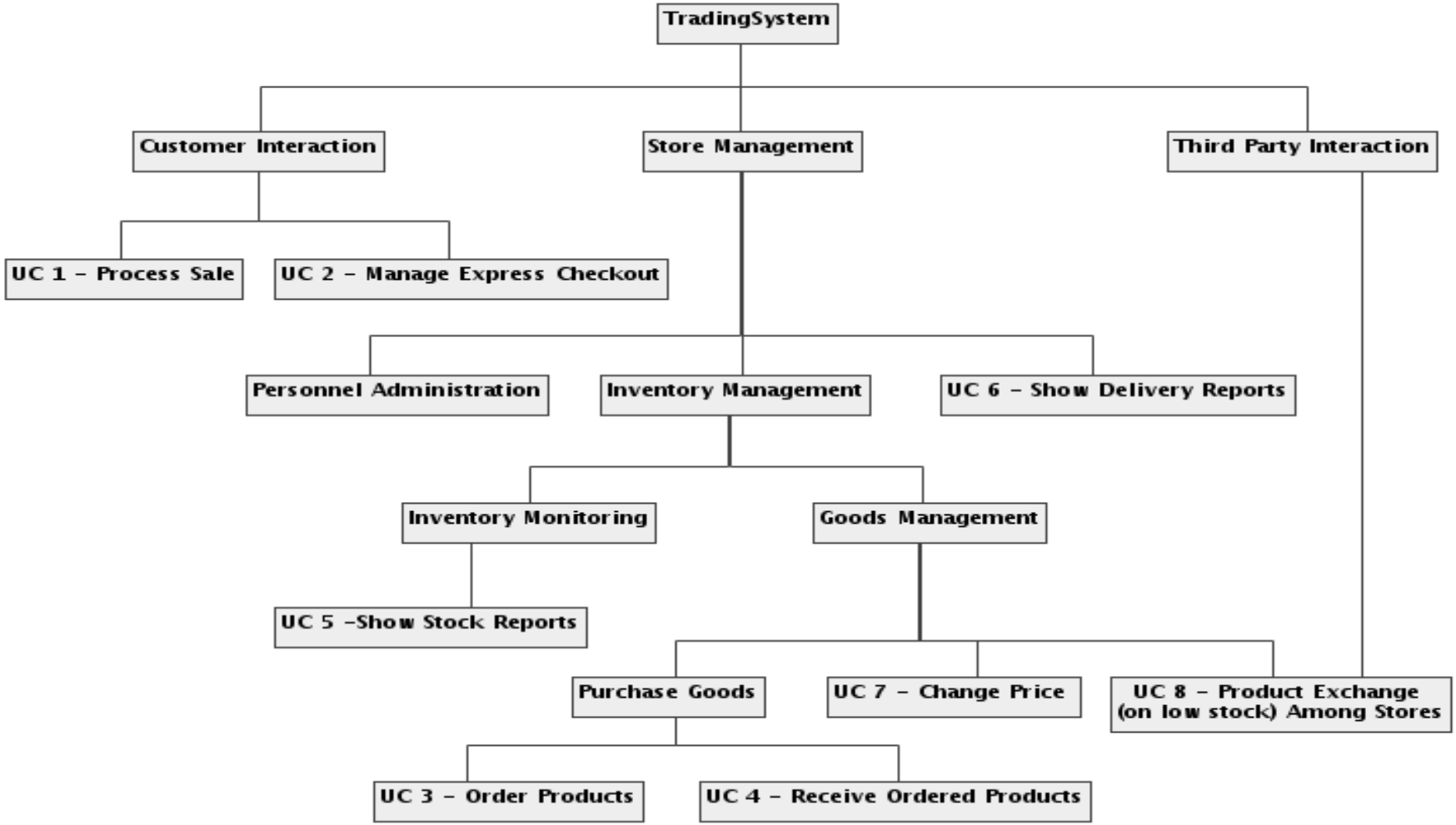
Computation Independent Model (CIM)

- Enterprise Concept Diagram (Structural view)



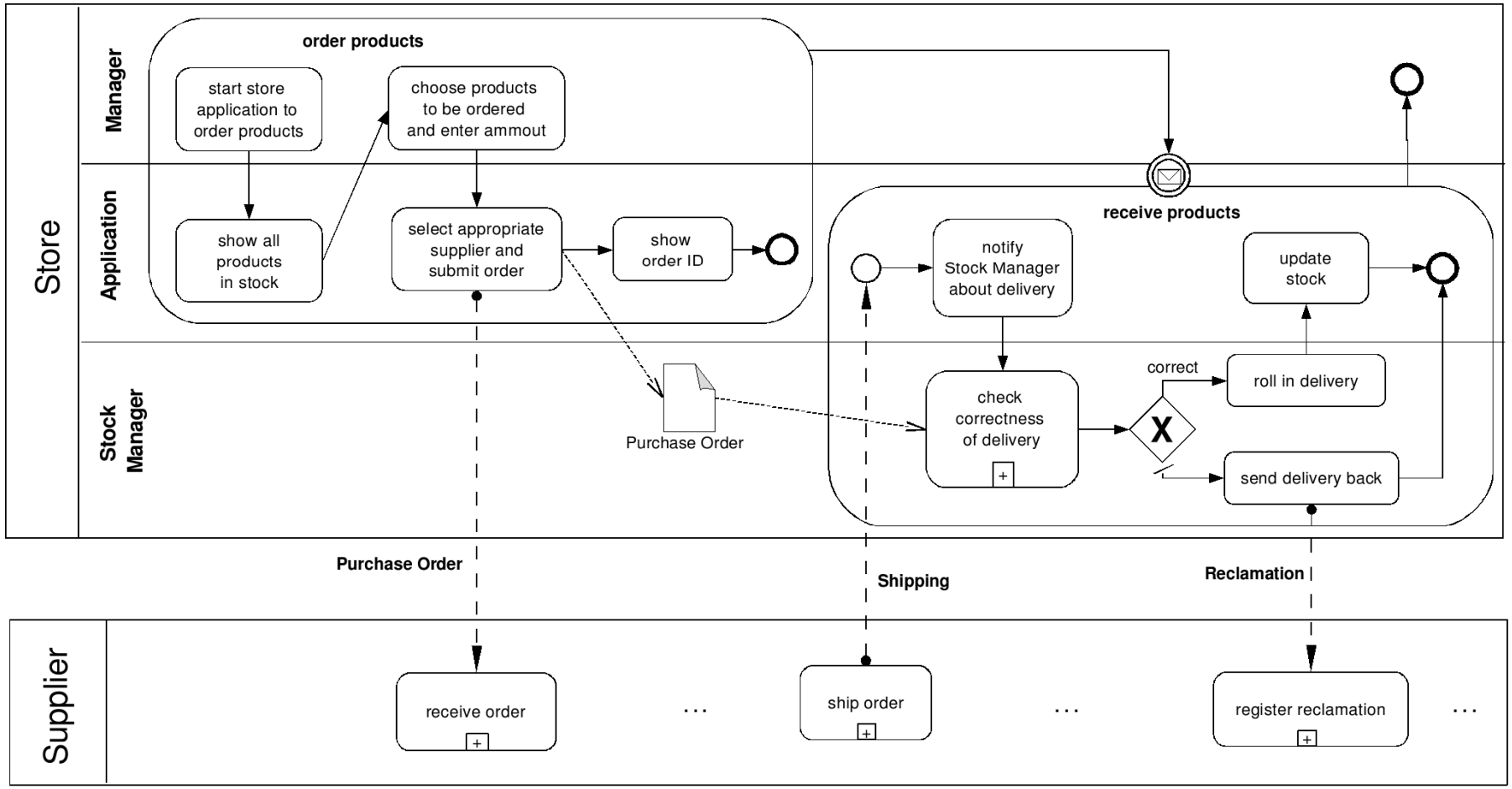
Computation Independent Model (CIM)

- Enterprise Process Diagram (Functional view)



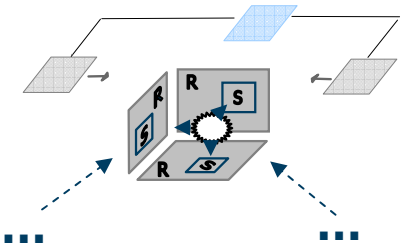
Computation Independent Model (CIM)

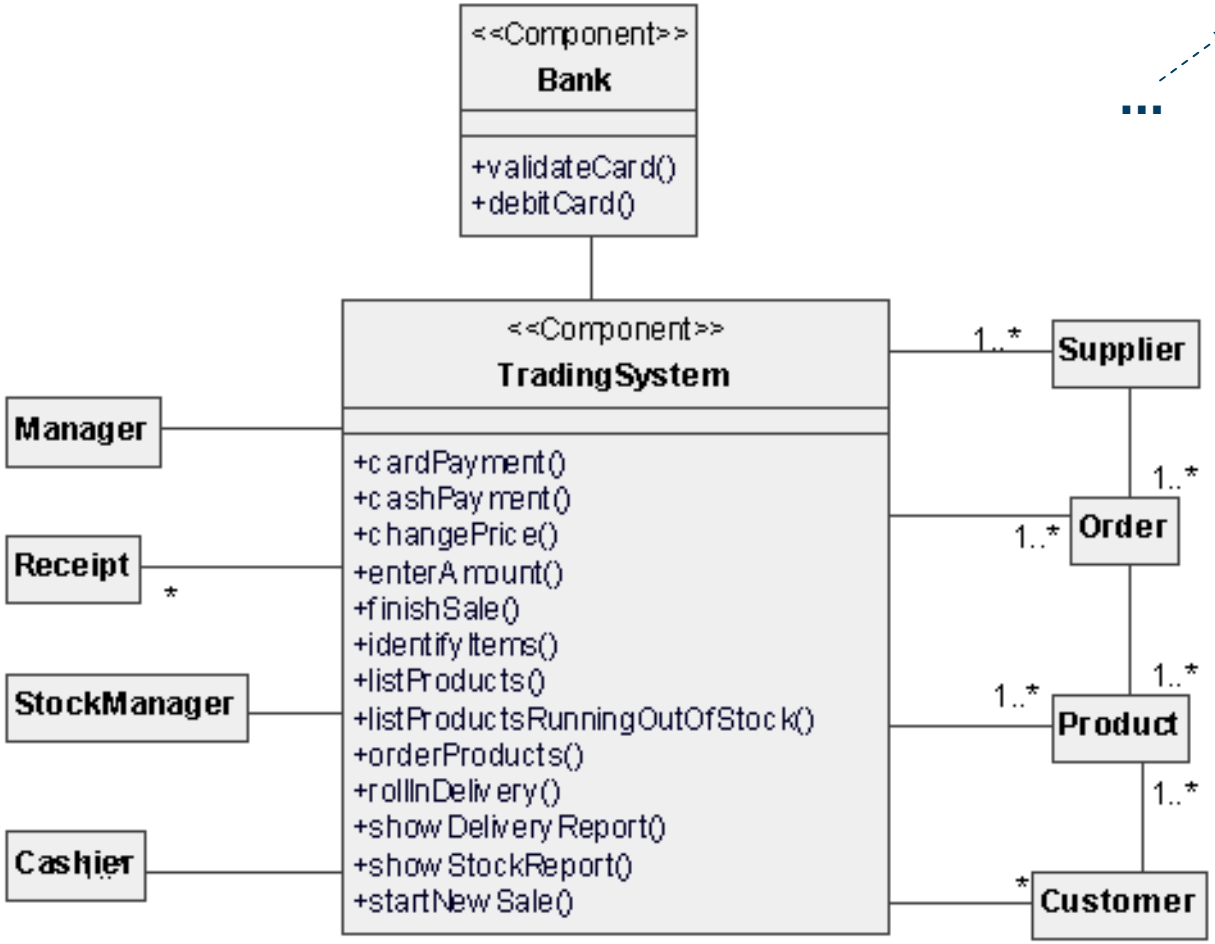
- Enterprise Workflow Diagram (Behavioral view)



Platform Independent Model (PIM)

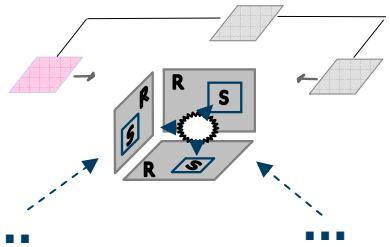
- Context Realization - Structural View

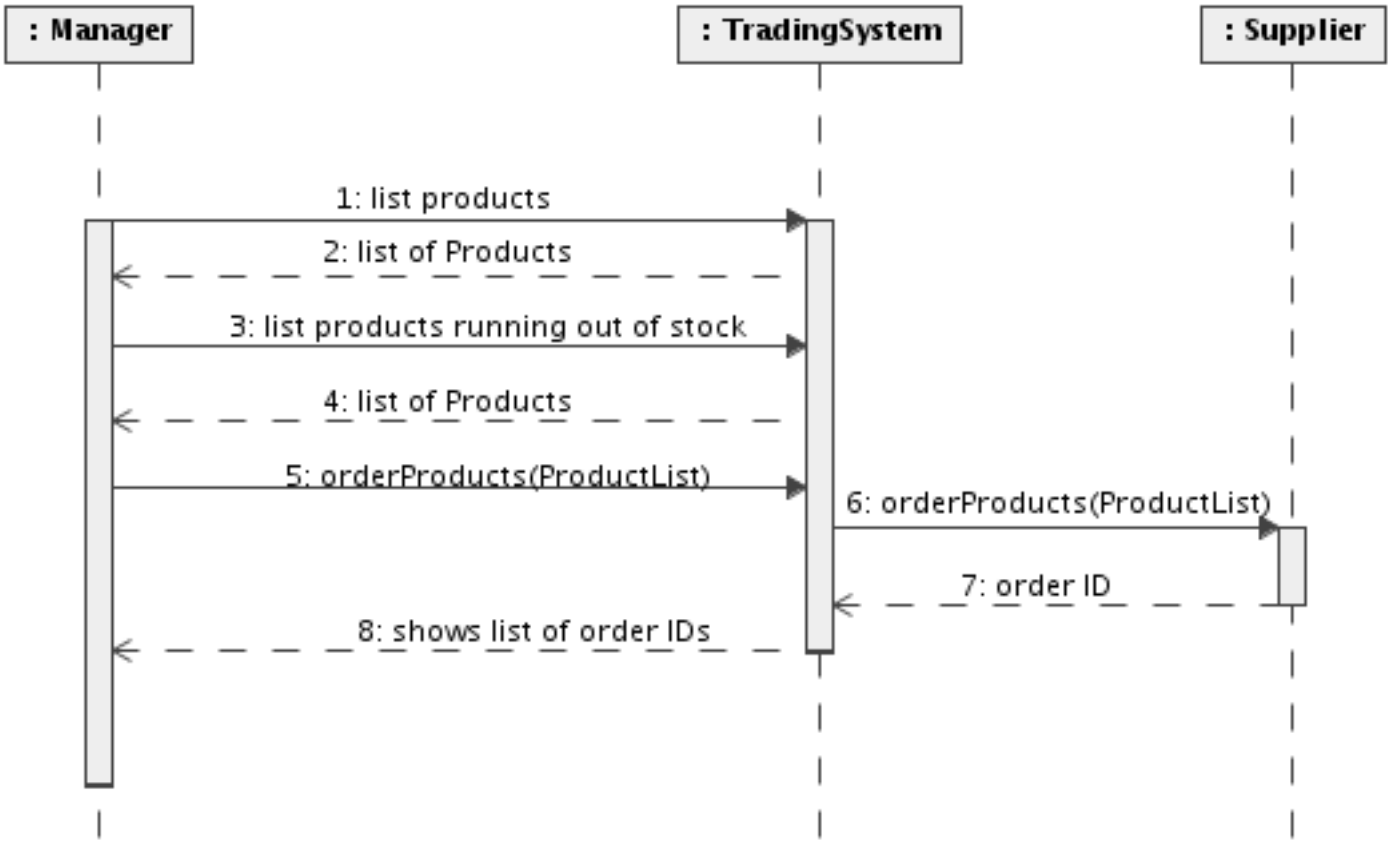




Platform Independent Model (PIM)

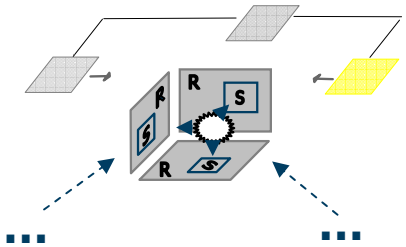
- Context Realization - Functional View
  - OrderProducts task

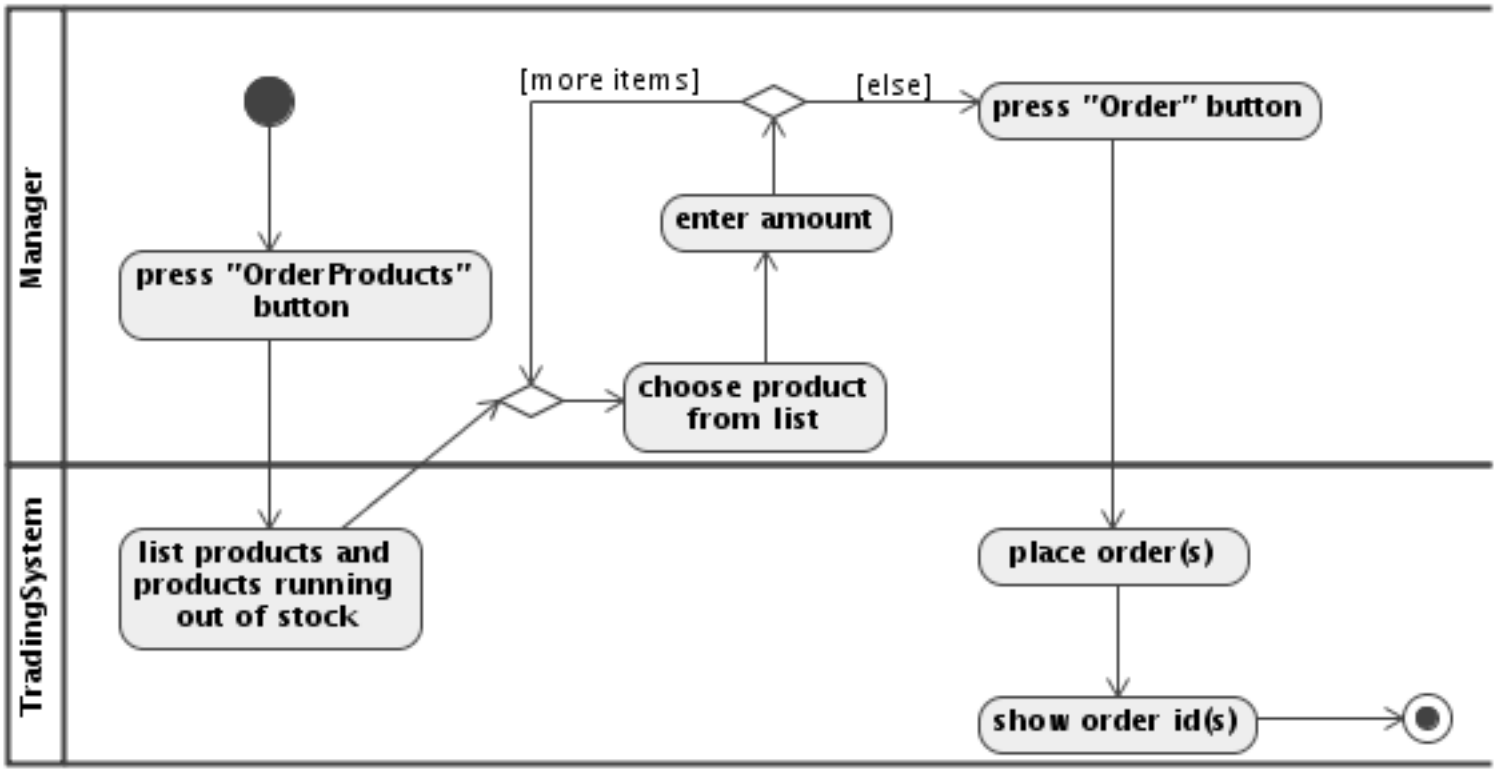




Platform Independent Model (PIM)

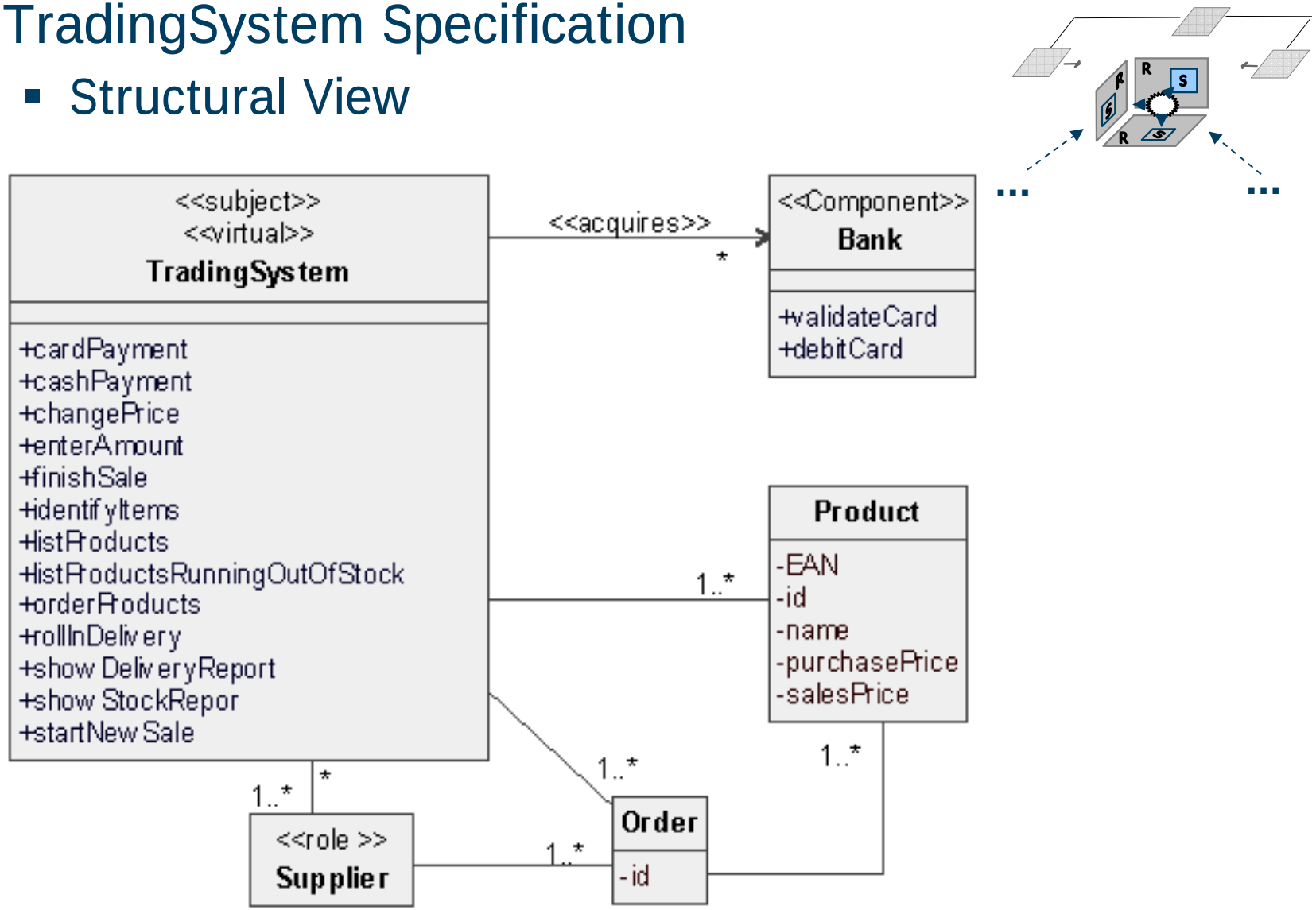
- Context Realization - Behavioral View
  - OrderProducts task





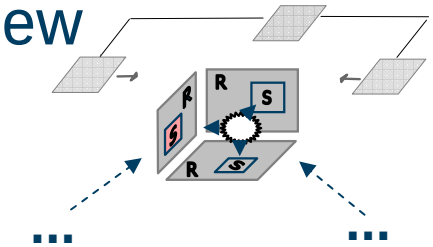
Platform Independent Model (PIM)

- TradingSystem Specification
  - Structural View



Platform Independent Model (PIM)

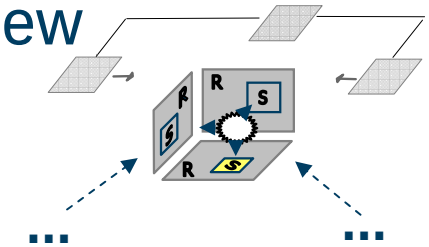
- TradingSystem Specification - Functional View
  - changePrice responsibility



Name	**changePrice**
Description	This method changes the salesPrice of a product.
Receives	*productID* : String – the identifier of the product *price* : Real – the new salesPrice of the product
Returns	*Success* : Boolean – true, if the salesPrice is changed – false, otherwise
Changes	The salesPrice of the product.
Assumes	The *productID* must be valid and the *price* parameter has to be greater than zero.
Result	The salesPrice of the product with the *productID* is set to *price*.

Platform Independent Model (PIM)

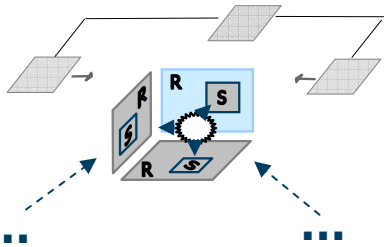
- TradingSystem Specification - Behavioral View

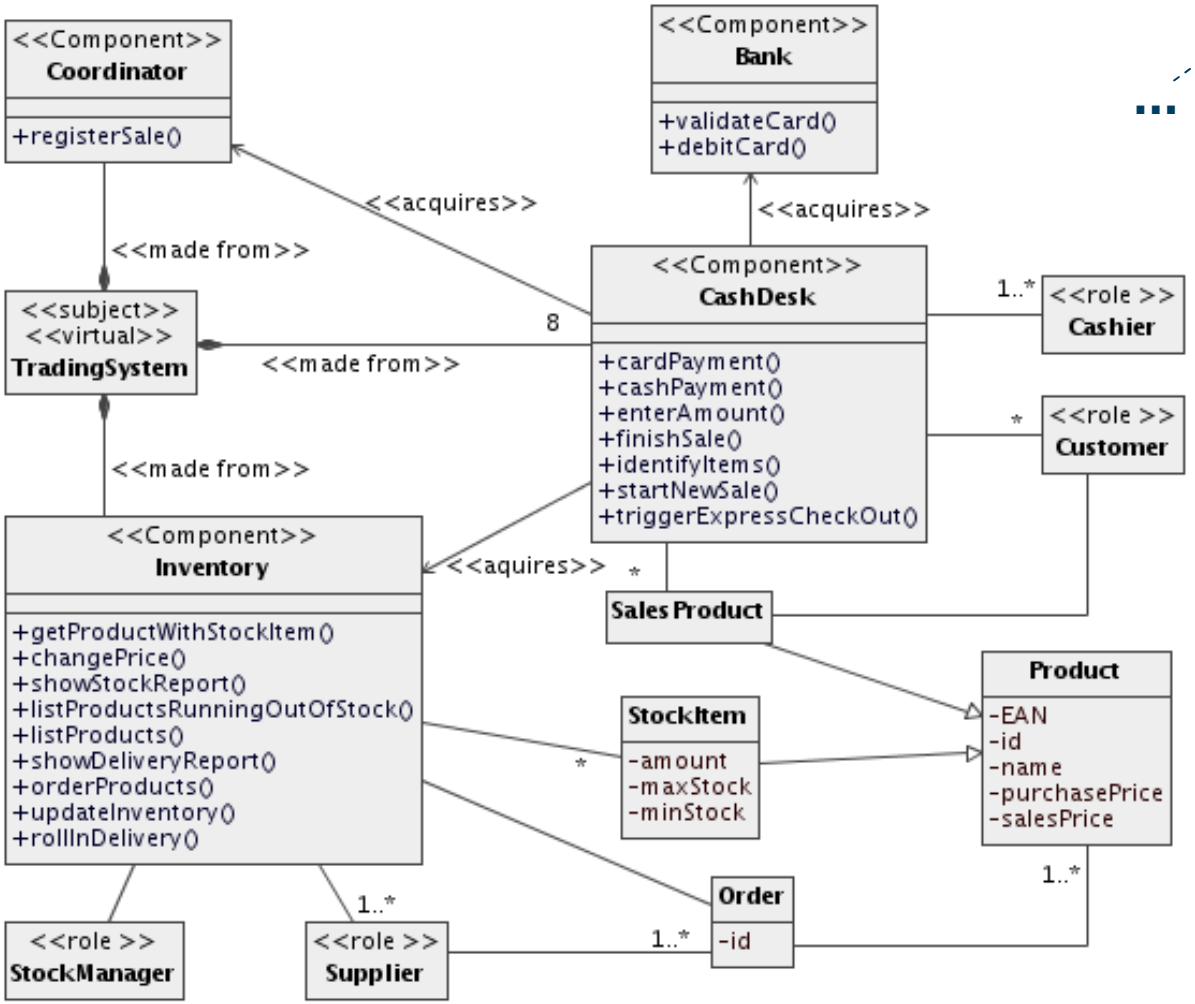


**TradingSystem has no externally visible states,
as it serves multiple users simultaneously**

Platform Independent Model (PIM)

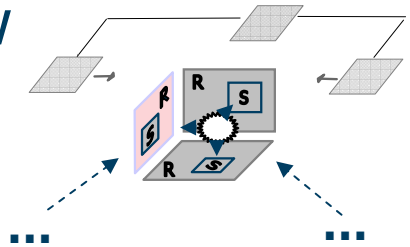
- TradingSystem Realization - Structural View

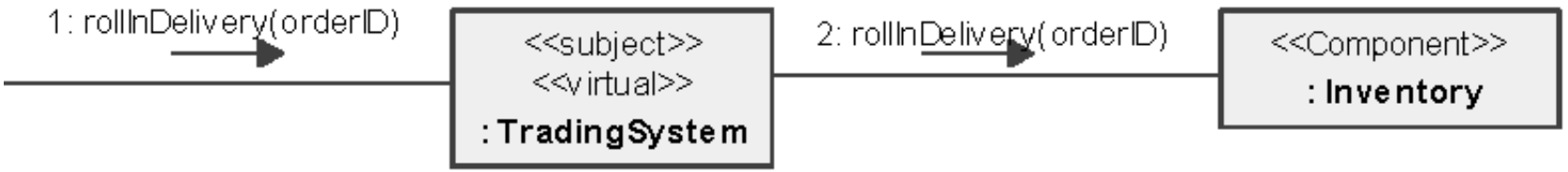




Platform Independent Model (PIM)

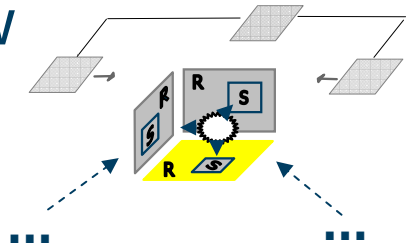
- TradingSystem Realization - Functional View
  - rollInDelivery responsibility

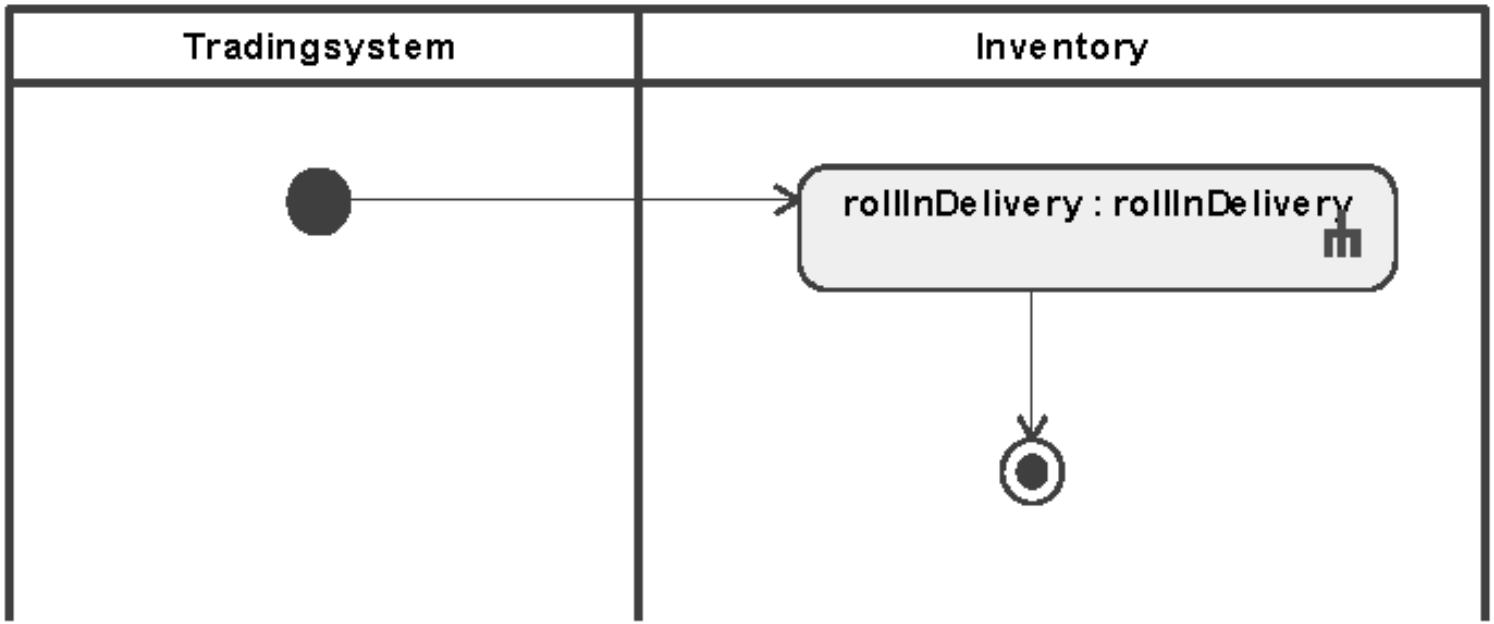




Platform Independent Model (PIM)

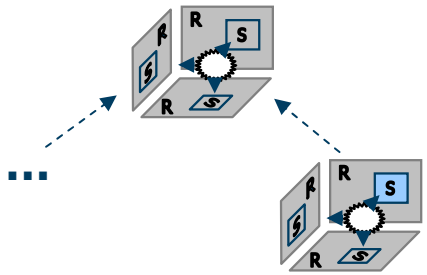
- TradingSystem Realization - Behavioral View
  - rollInDelivery responsibility

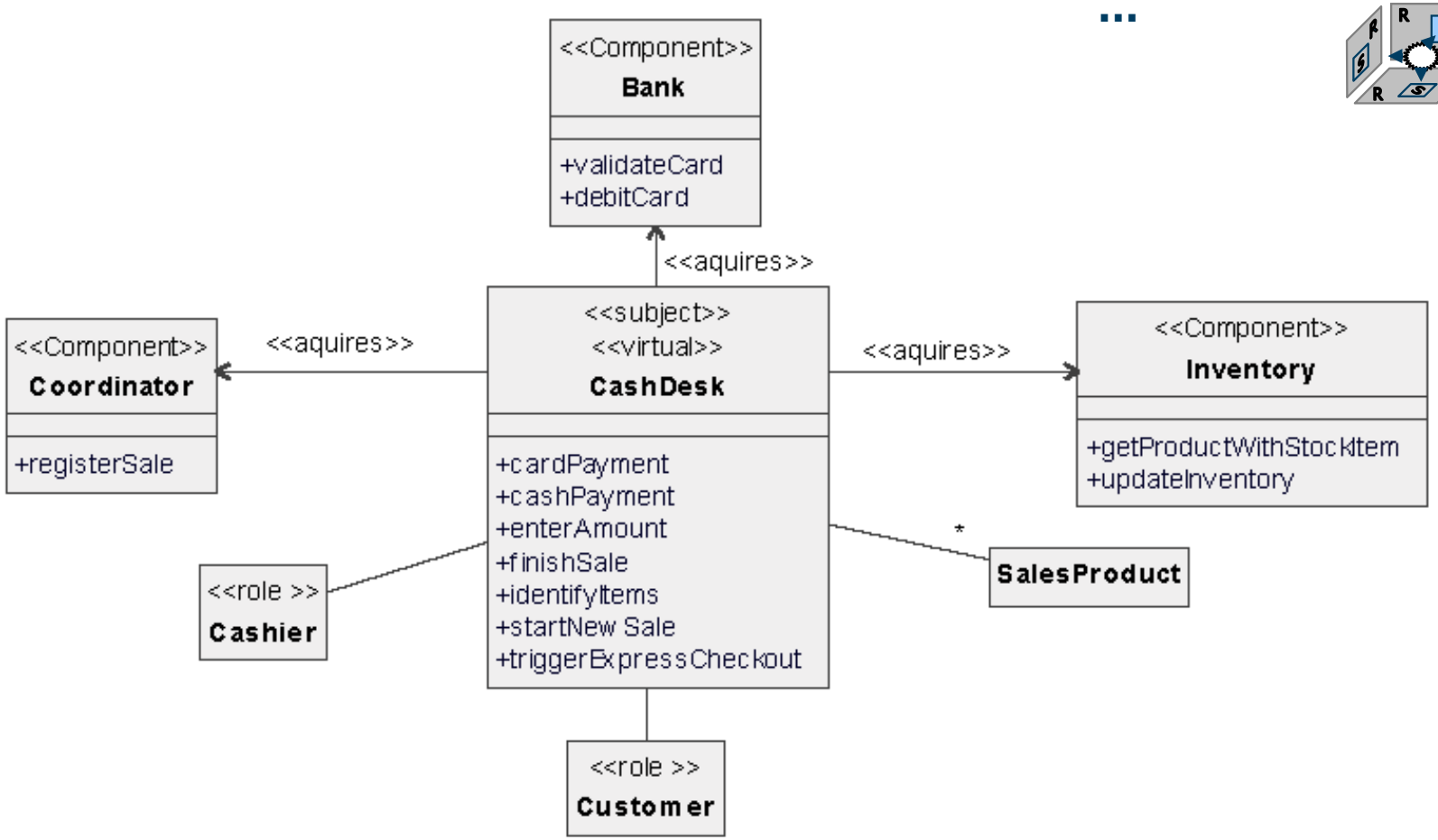




Platform Independent Model (PIM)

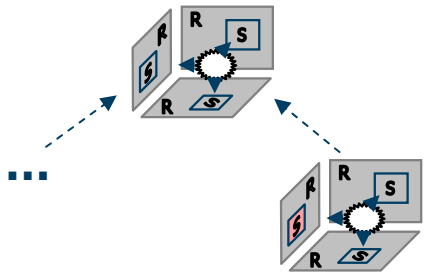
- CashDesk Specification - Structural View





Platform Independent Model (PIM)

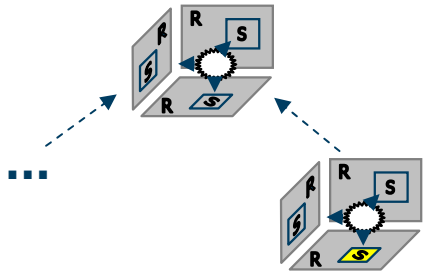
- CashDesk Specification - Functional View
  - CardPayment responsibility

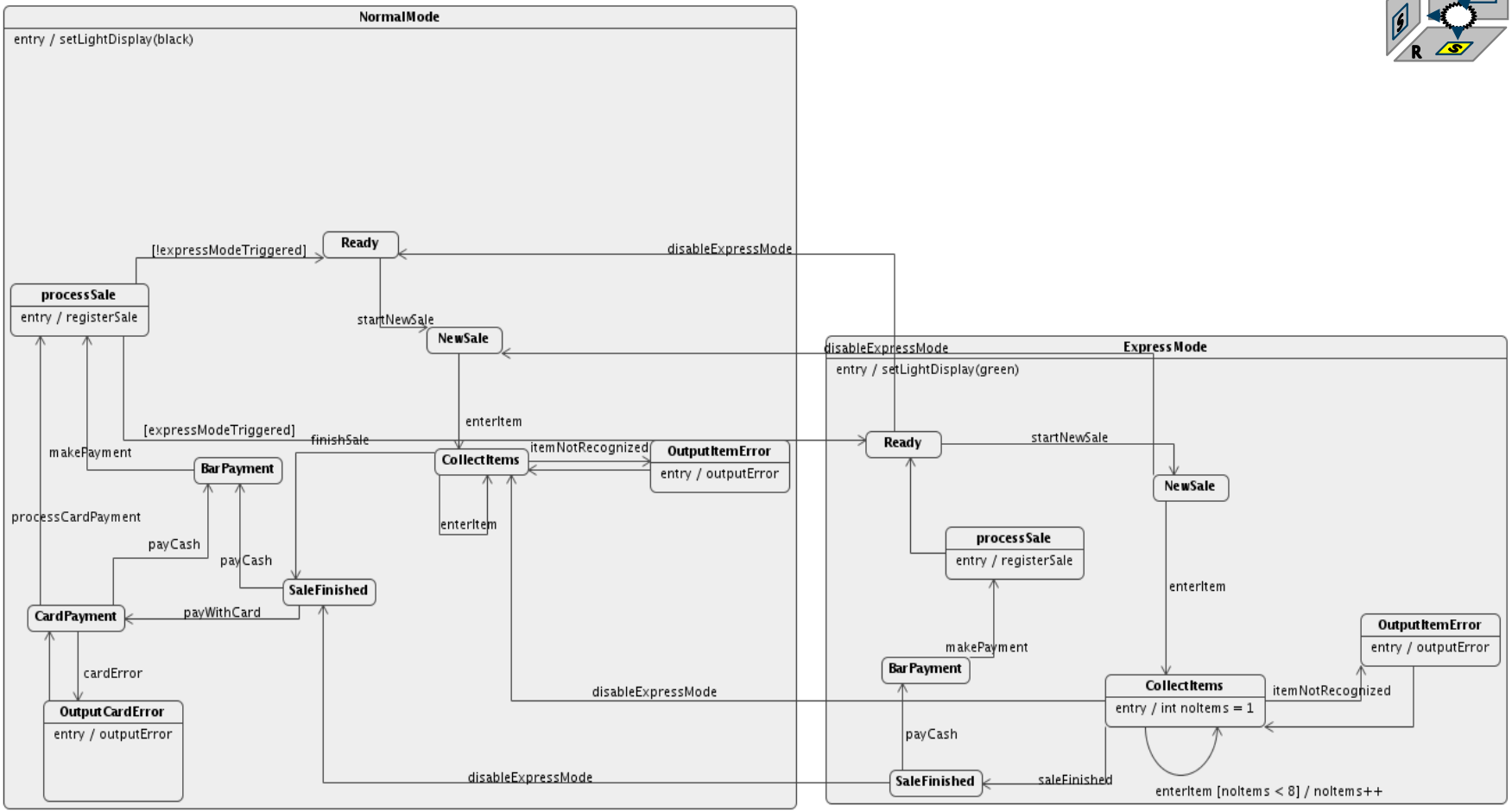


Name	**cardPayment**
Description	This method realizes the payment via card.
Constraints	For card payment the connection to the bank has to be established.
Receives	*sum* : Real – the sum to pay
Returns	*success* : Boolean – true, if payment successfull; false, otherwise
Sends	Bank::validateCard(*cardInformation*,*pin*) Bank::debitCard(*transactionID*)
Changes	The card of the customer is debited.
Rules	Bank::debitCard(*transactionID*) can only be called, if the card is valid. To validate the card the customer has to enter his PIN. The *cardInformation* is read from the credit card of the customer.
Assumes	The sum of the prices of the goods must be greater than zero.
Result	The card of the customer is debited with the *sum*.

Platform Independent Model (PIM)

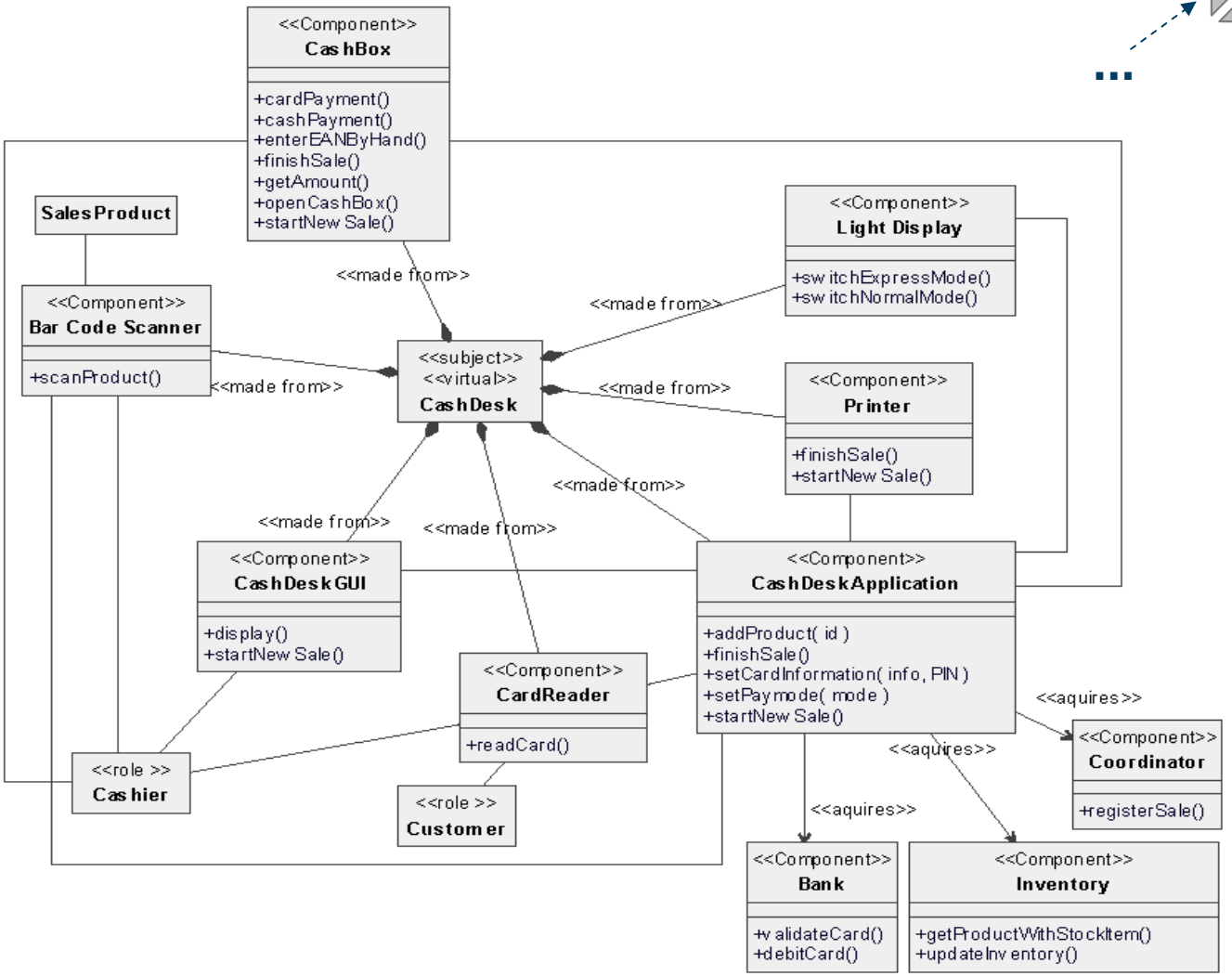
- CashDesk Specification - Behavioral View





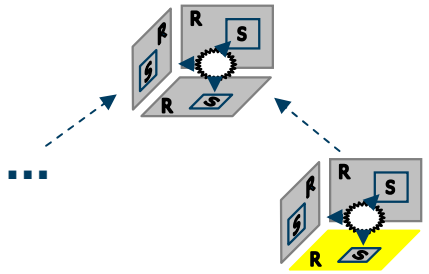
Platform Independent Model (PIM)

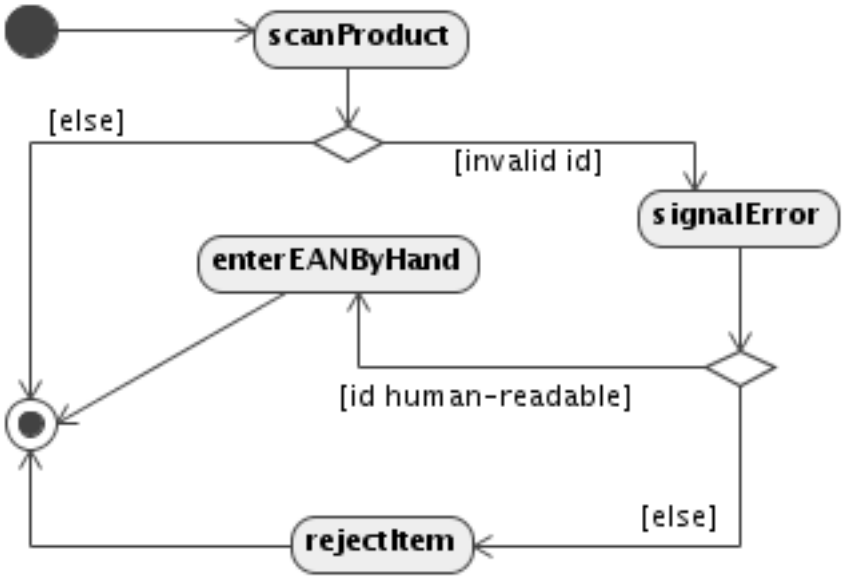
- CashDesk Realization - Structural View



Platform Independent Model (PIM)

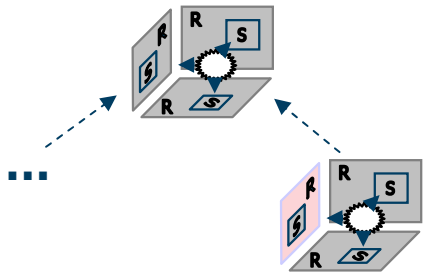
- CashDesk Realization - Behavioral View
  - IdentifyItems “operation”

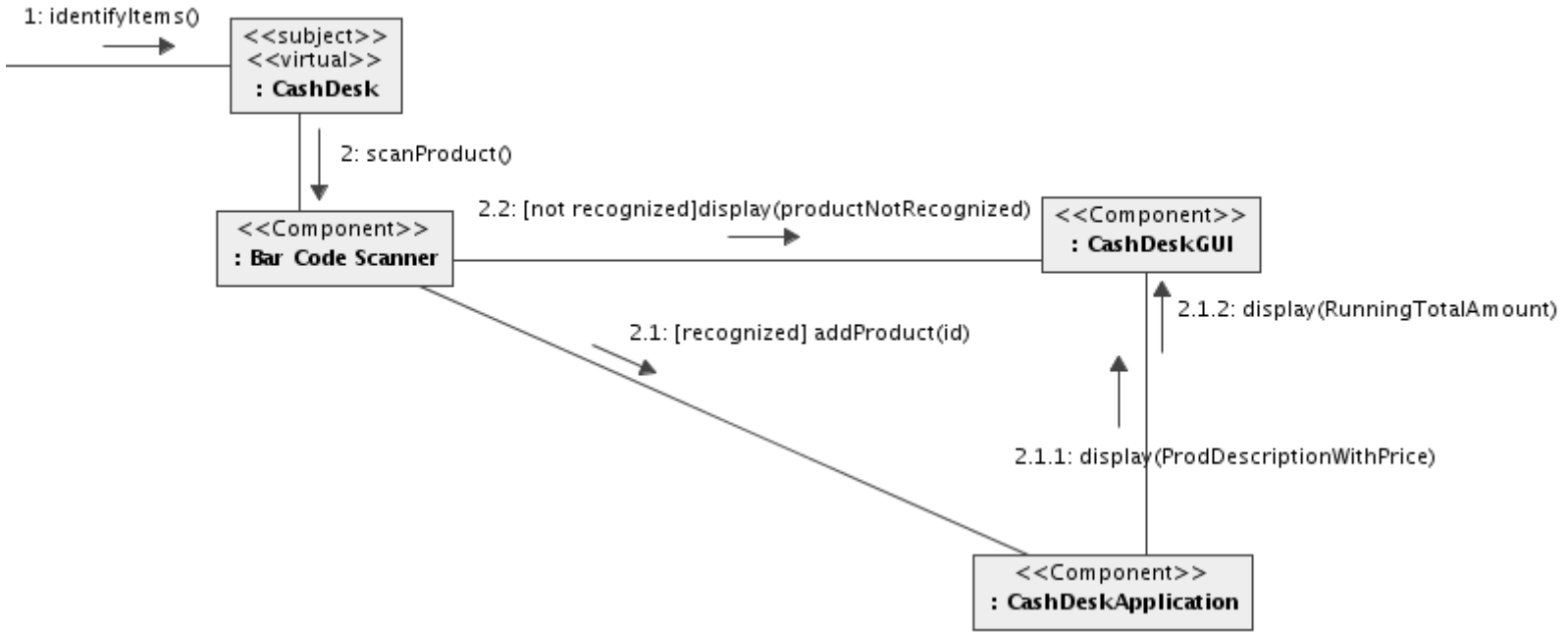




Platform Independent Model (PIM)

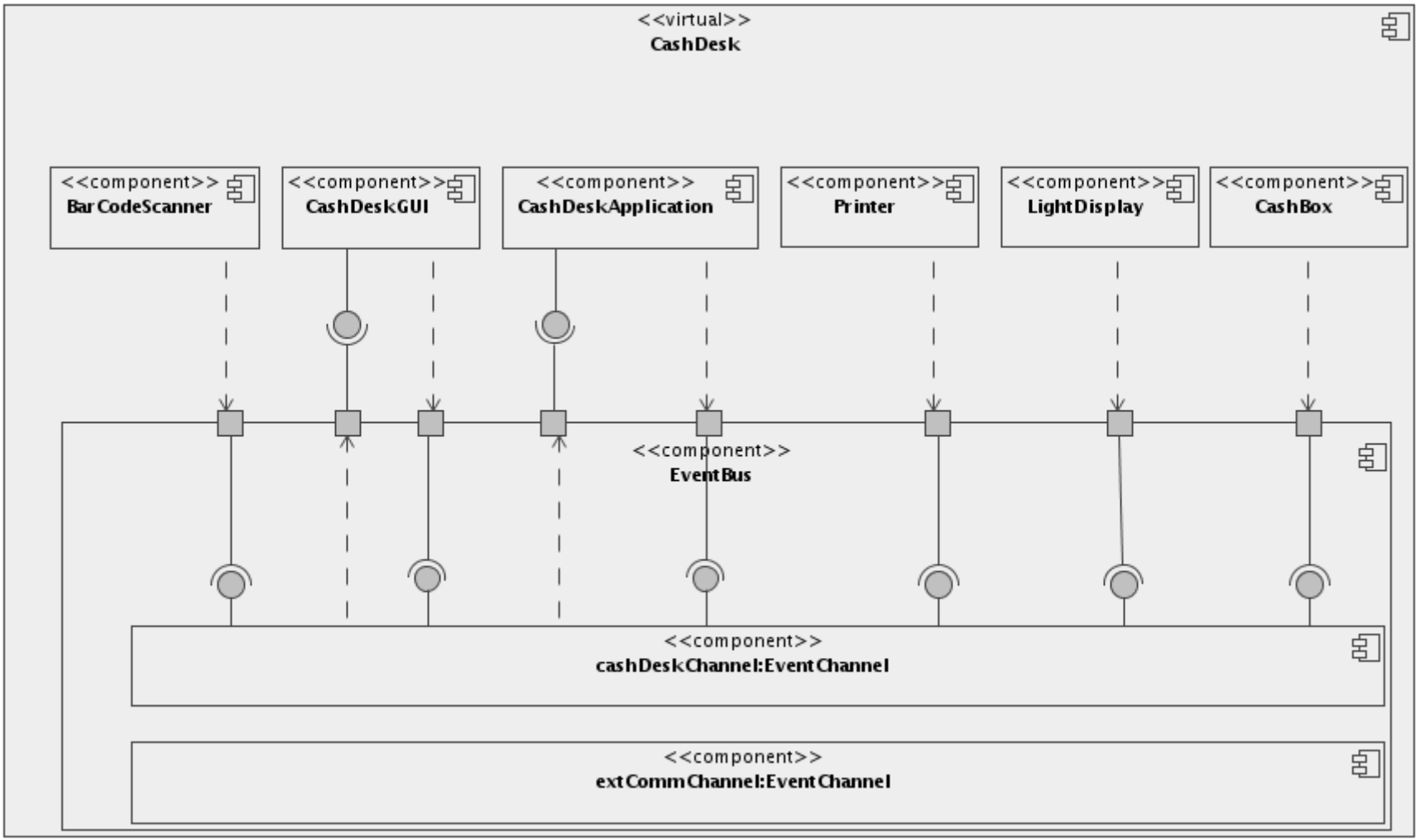
- CashDesk Realization - Functional View
  - IdentifyItems “operation”





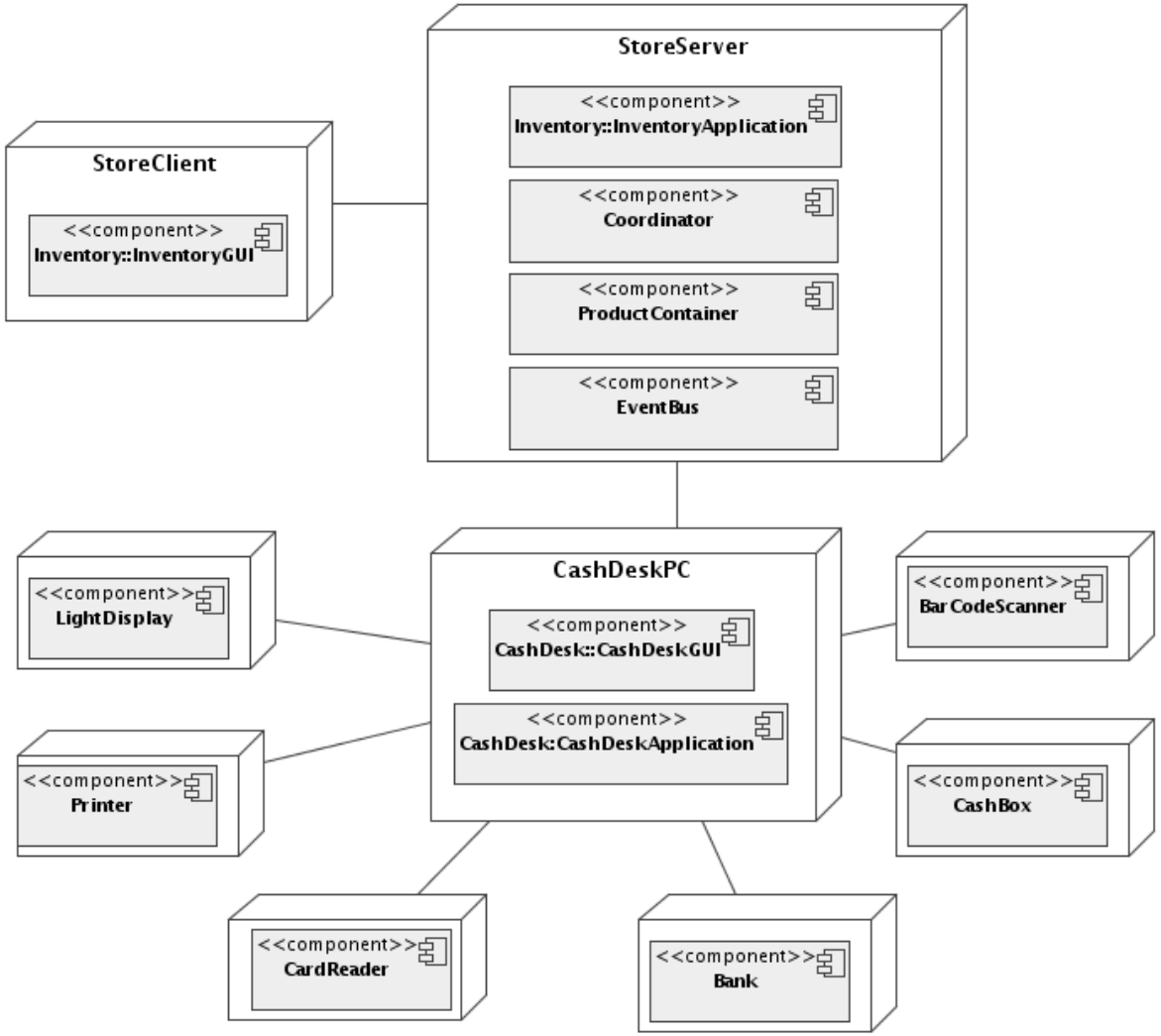
Platform Specific Model (PSM)

- CashDesk



Platform Specific Model (PSM)

- Deployment Diagram



Contents

- KobrA – an overview
- Modeling the CoCoME
- Conclusion
  - Summary
  - Pros and Cons
  - Tool support

Summary

- Uniform representation of components and composite systems in a hierarchical form
- Separation of concerns
  - Orthogonal Development Dimensions
  - Orthogonal Projections/Views
  - Separation of Specification and Realization
  - Separation of Product and Process

Pros and Cons of KobrA

- Simple and Systematic
  - strict separation of concerns
- Incremental introduction of components and product lines
- Uniform treatment of systems and components
  - component assembly = component creation
  - fractal-like product, recursive process

BUT

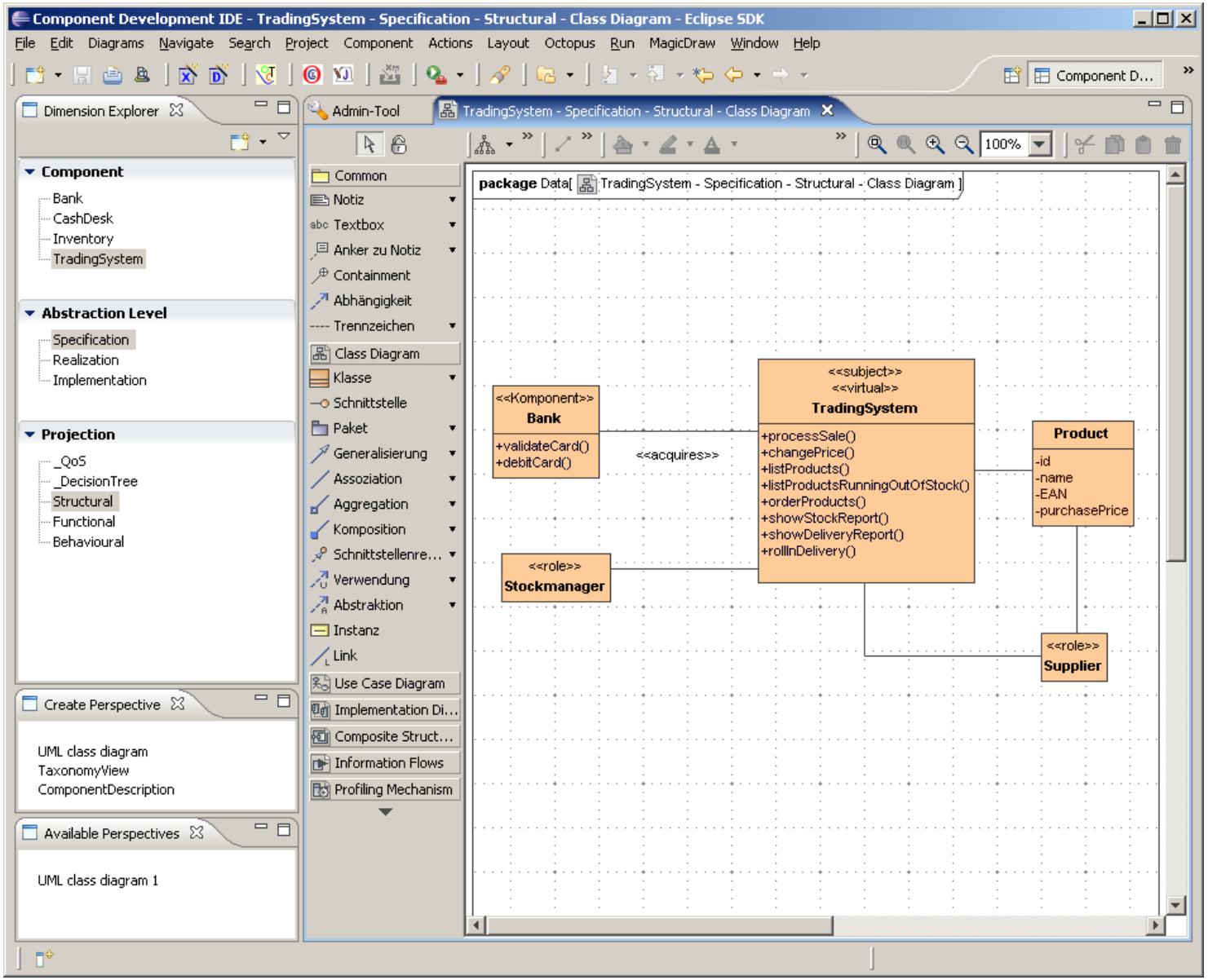
- Fairly complex and difficult to apply

Tool support needed

Tool Support

- Main objectives:
  - User-friendly framework for defining and navigating around the different views
  - Infrastructure for consistency checking and view-generating tools
  - Unifying meta-model allowing all views to be generated automatically from a single underlying representation of a component

Tool Support



Thank you.