

PIM to PSM transformations for an event driven architecture in an educational tool

Geert Monsieur, Monique Snoeck, Raf Haesen, Wilfried Lemahieu

KULeuven

Faculty of economics and applied economics
Management Information Systems Group
(Belgium)

Geert.Monsieur@econ.kuleuven.be

Presentation overview

- Introduction
- Solution
 - Platform Independent Model (PIM)
 - Platform Specific Model (PSM)
 - Transformation Rules
- Conclusion

Presentation overview

- Introduction
- Solution
 - Platform Independent Model (PIM)
 - Platform Specific Model (PSM)
 - Transformation Rules
- Conclusion

Why do we need MDA in an educational environment?

- Course on Business Modelling
 - understand organisational impact of business model
 - Requires concrete understanding of abstract domain model (with all behavioural and interaction aspects)
- Goal of our MDA tool
 - create a prototype to improve concretisation



Presentation overview

- Introduction
- Solution
 - Platform Independent Model (PIM)
 - Platform Specific Model (PSM)
 - Transformation Rules
- Conclusion

PIM

PSM

Transformation rules

Platform Independent Model (PIM)

3 views/diagrams

Class diagram

Object Event
Table (OET)

Finite State
Machines

PIM

PSM

Transformation rules

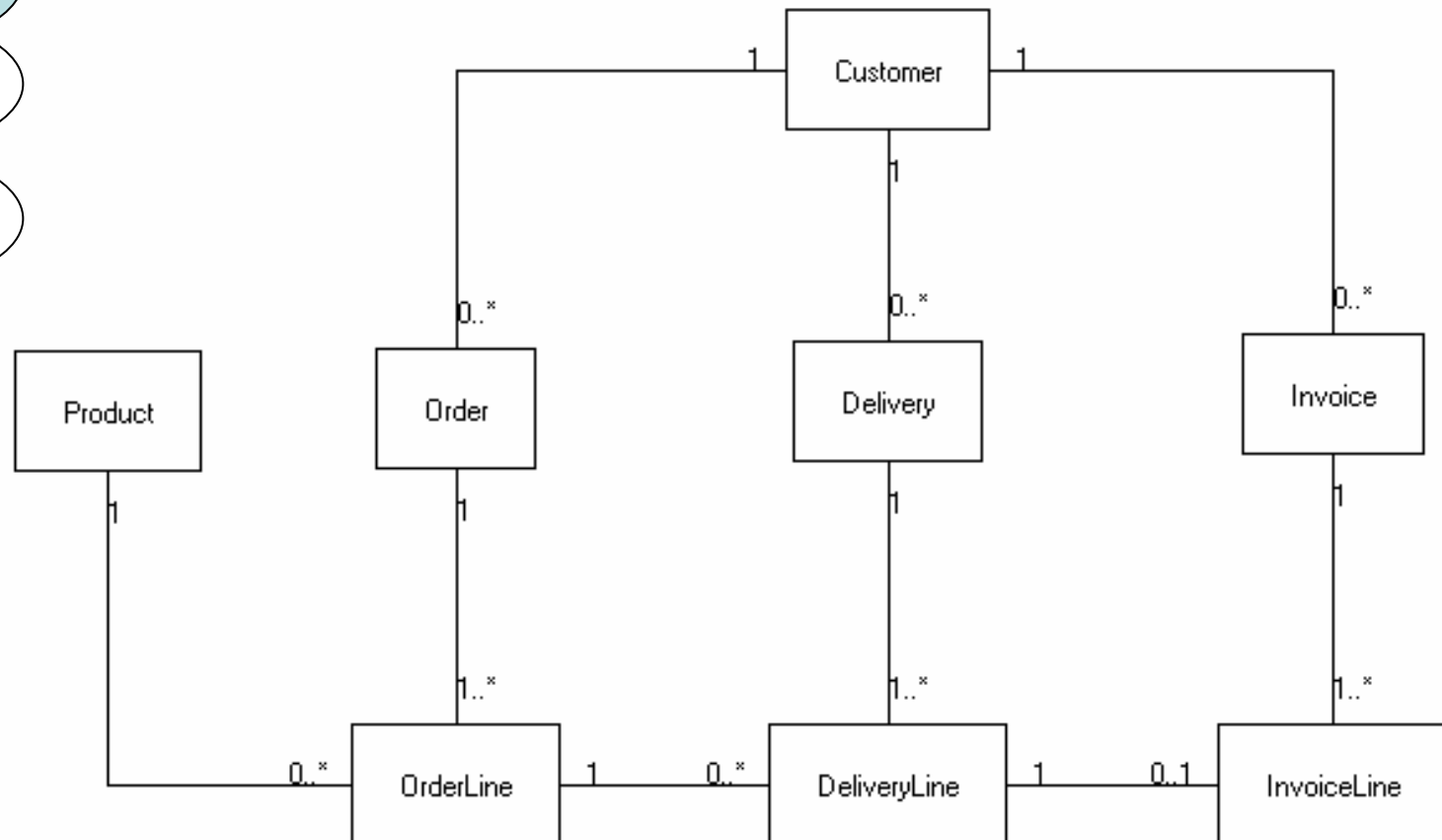
Class diagram

running example

Class diagram

Object Event
Table (OET)

Finite State
Machines



PIM

PSM

Transformation rules

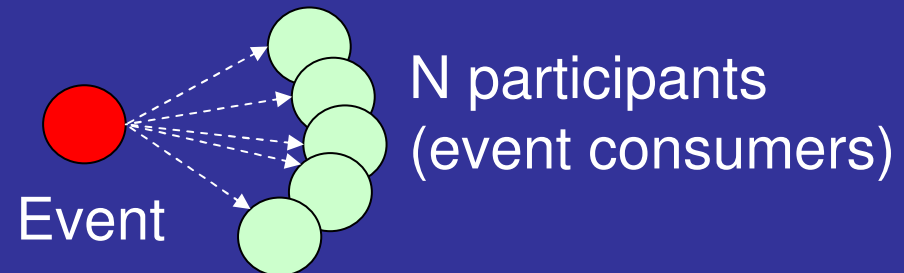
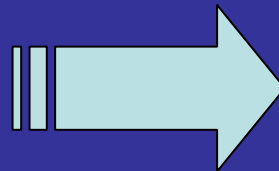
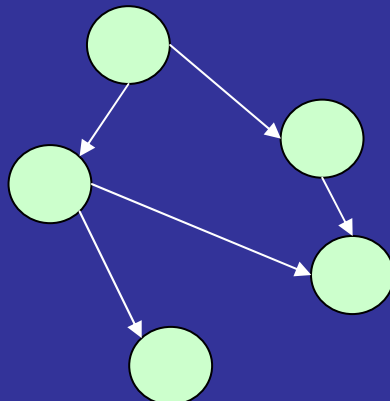
Object-Event Table

Class diagram

Object Event
Table (OET)

Finite State
Machines

- Event = atomic unit of action that can involve several domain classes
- Standardised event-based interaction pattern instead of sequence charts/ collaboration diagrams
- Motivation = raise abstraction level



PIM

PSM

Transformation rules

Object event table (OET)

Class diagram

Object Event
Table (OET)

Finite State
Machines

Different roles in processing of an event

V = validation (checking some preconditions)

M = modification of the instance

C = creation of an instance

E = ending of an instance

(for the moment we include V in all roles in our MDA tool)

	Customer	Product	Order	OrderLine	...
<i>cr_order</i>	V		C		
<i>cr_orderLine</i>		V+M	M	C	
....					

running example

PIM

PSM

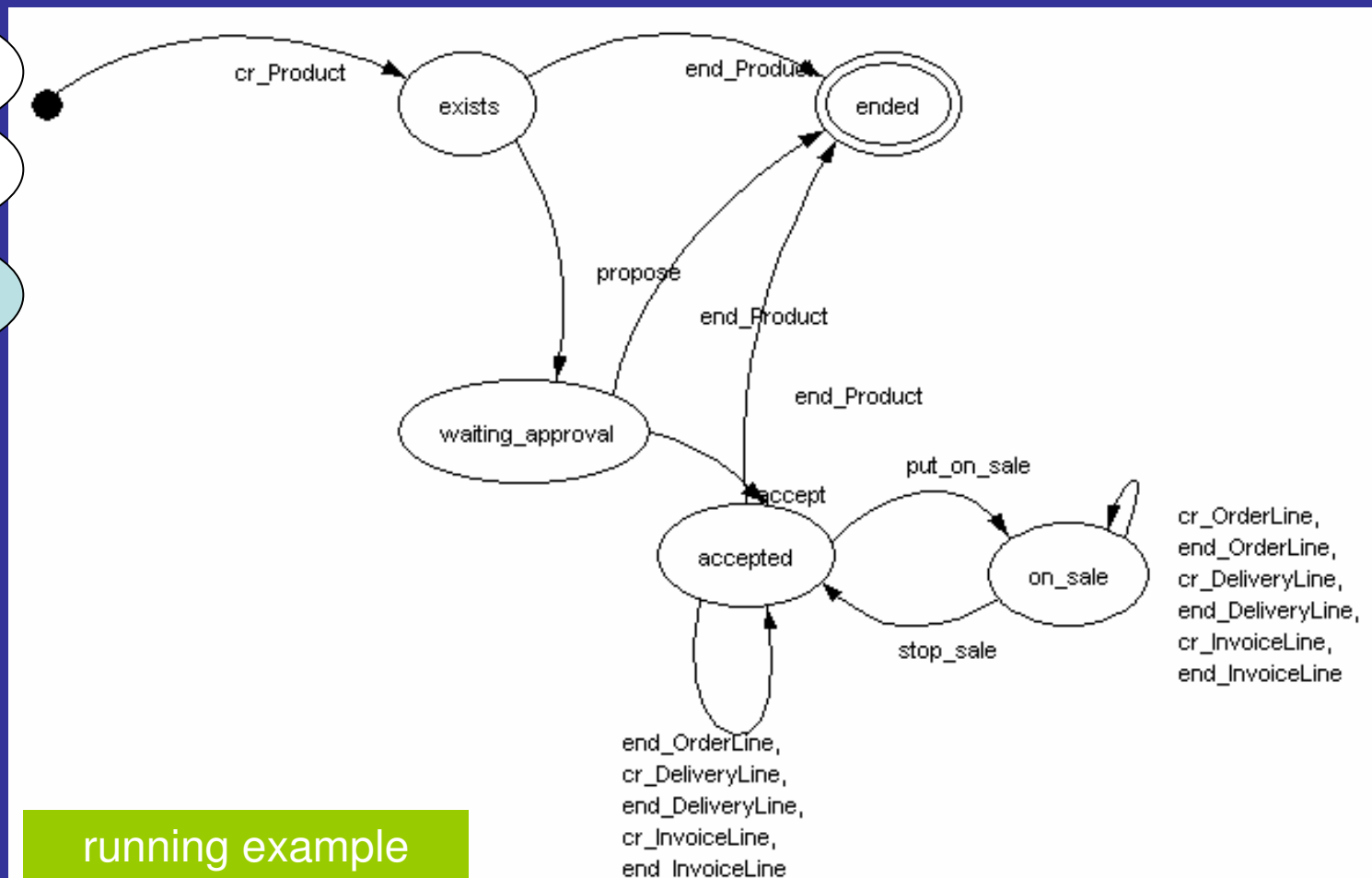
Transformation rules

Finite State Machines (FSM) (1 per class)

Class diagram

Object Event
Table (OET)

Finite State
Machines



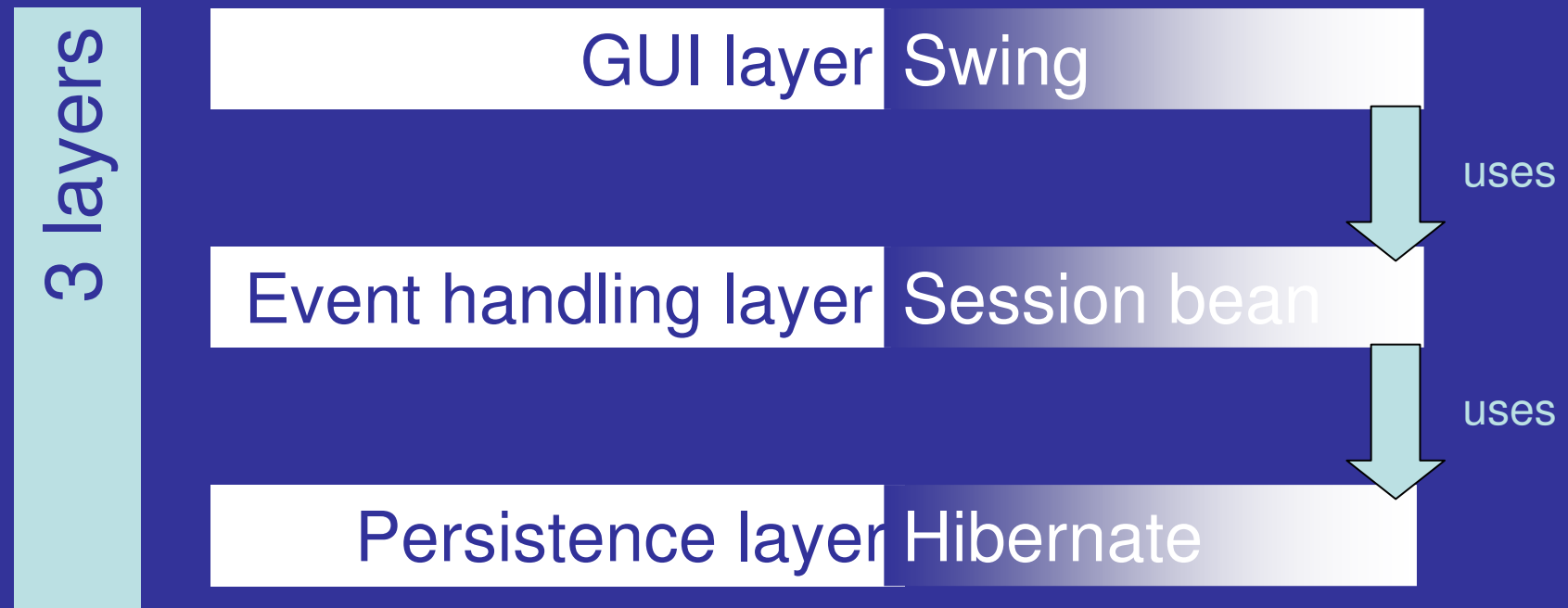
running example

PIM

PSM

Transformation rules

Platform specific model



PIM

PSM

Transformation rules

Generating the persistence layer

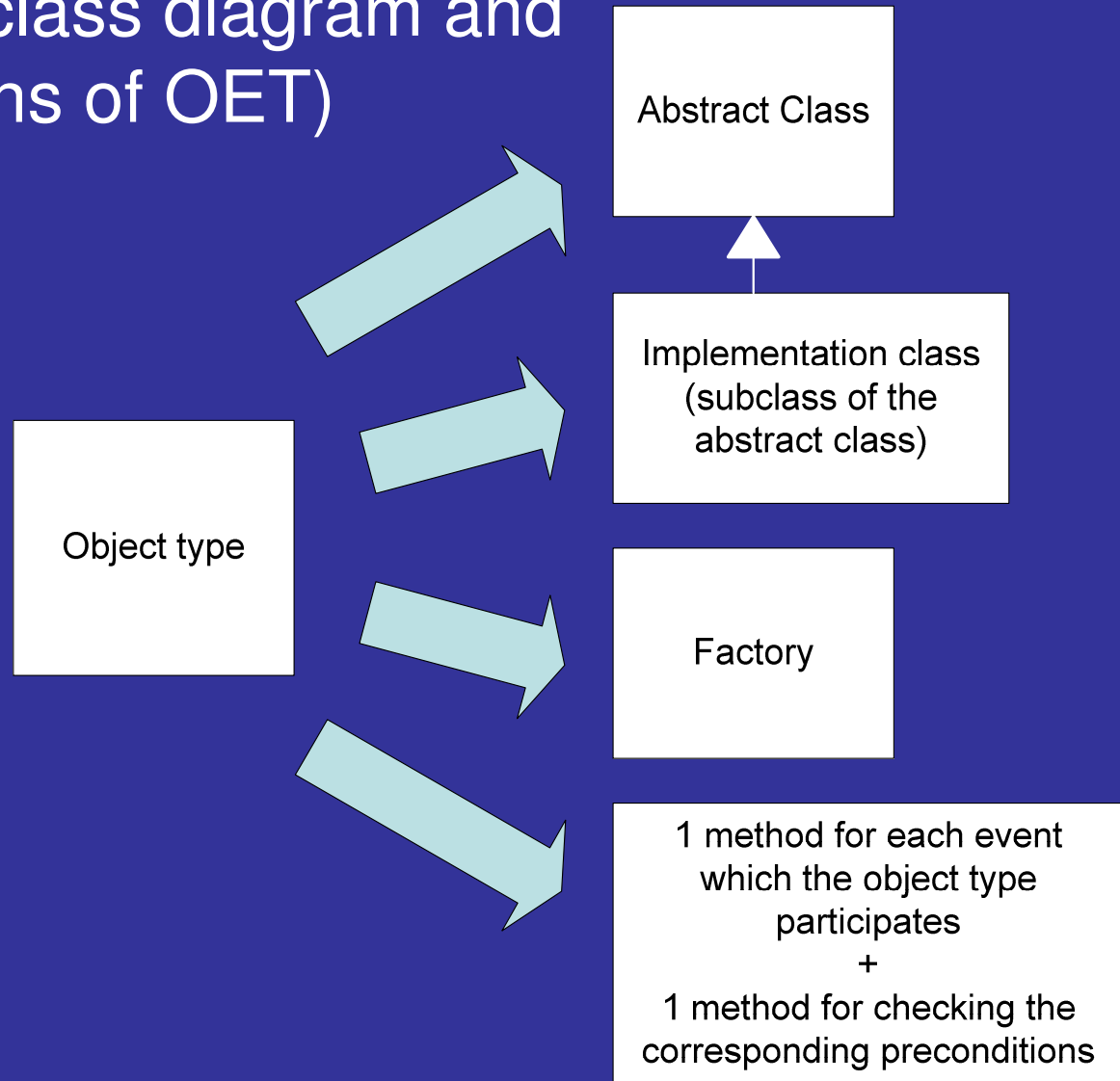
GUI layer

Event handling layer

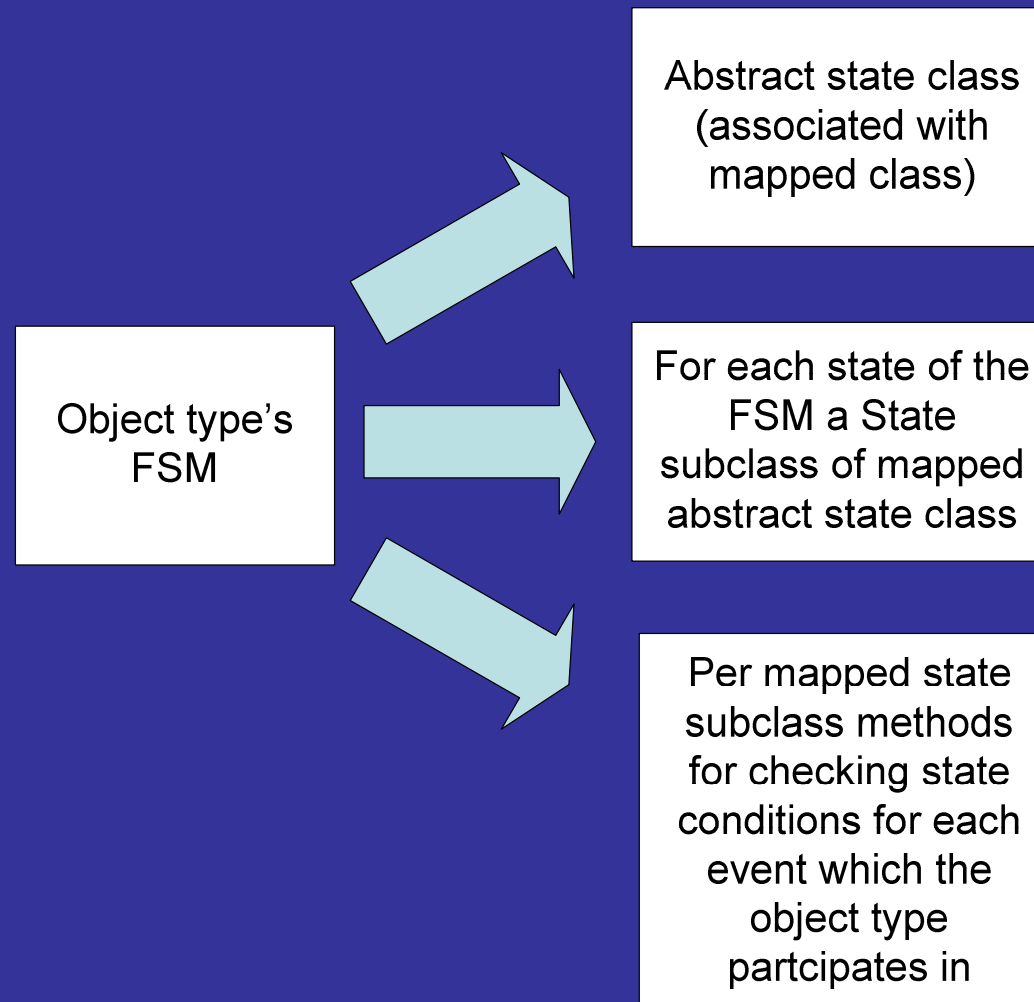
Persistence layer

- Transforming the class diagram and operations (columns of OET)
- Transforming the finite state machines

Transforming the class diagram and operations (columns of OET)



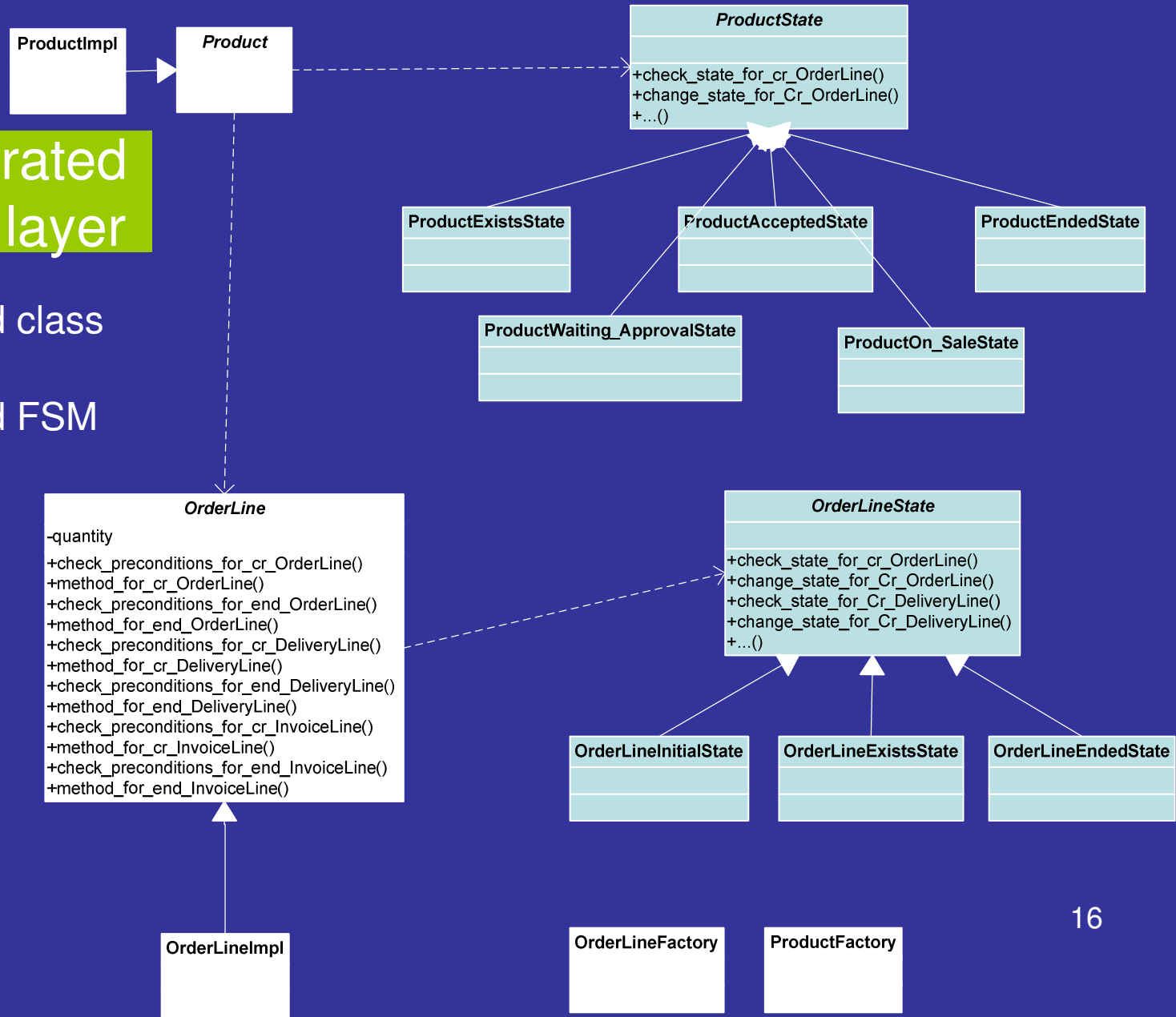
Transforming the finite state machines



Partial generated persistence layer

 = transformed class

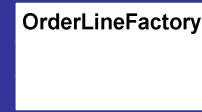
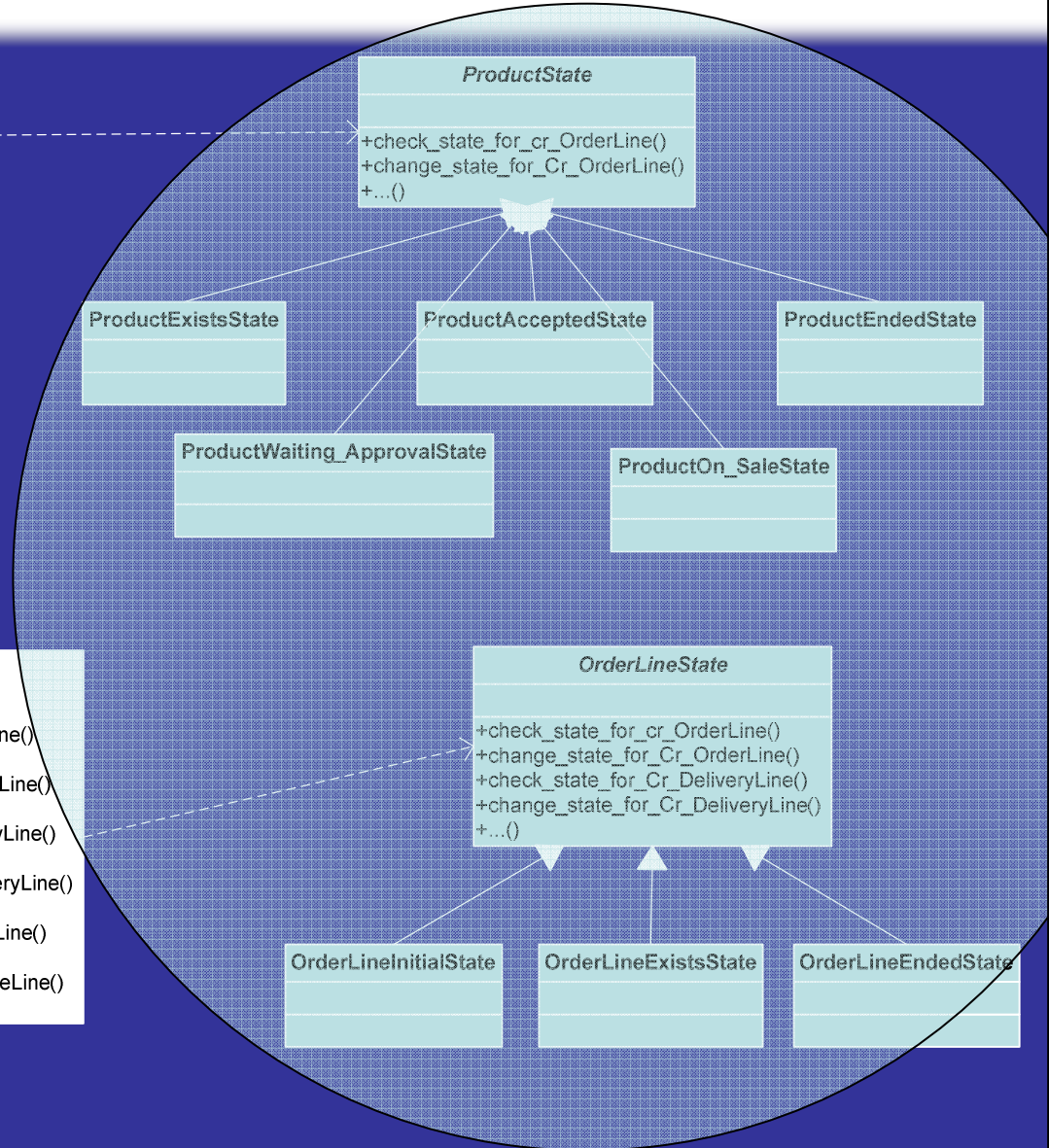
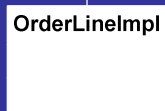
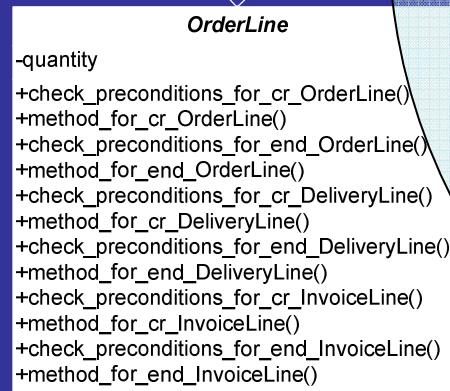
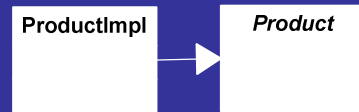
 = transformed FSM



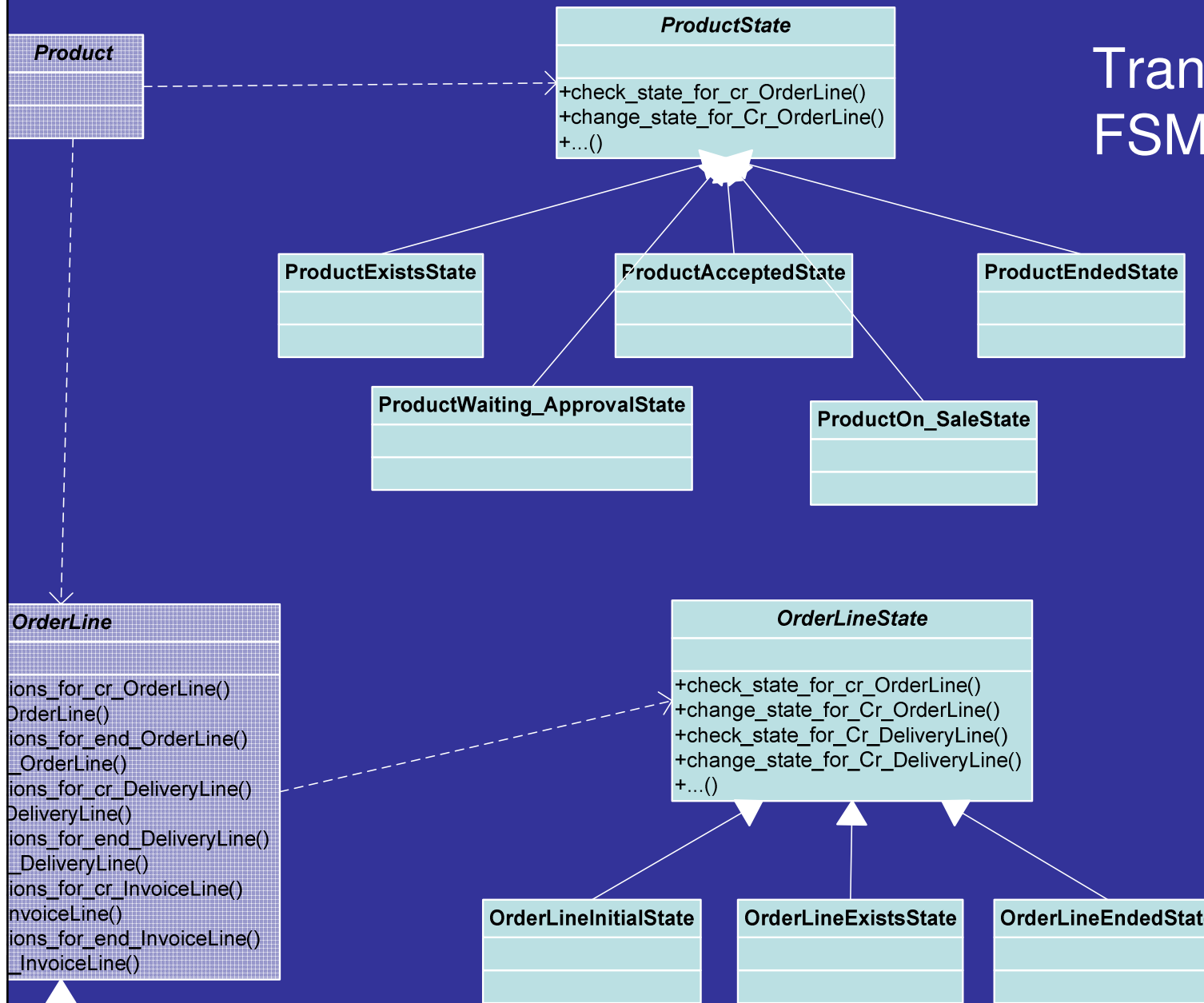
Partial generated persistence layer

 = transformed class

 = transformed FSM



Transformed FSM



PIM

PSM

Transformation rules

Generating the event handling layer

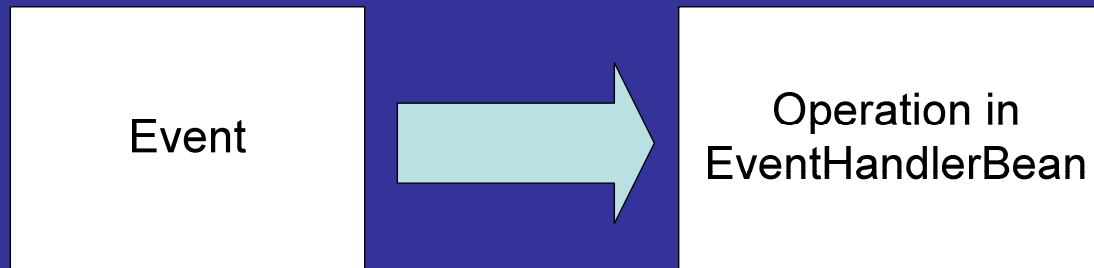
GUI layer


Event handling layer

Persistence layer

- Transforming the events (rows of OET)
- Standard collaboration pattern for generation of an event handler

Transforming the events (rows of OET) into event handlers



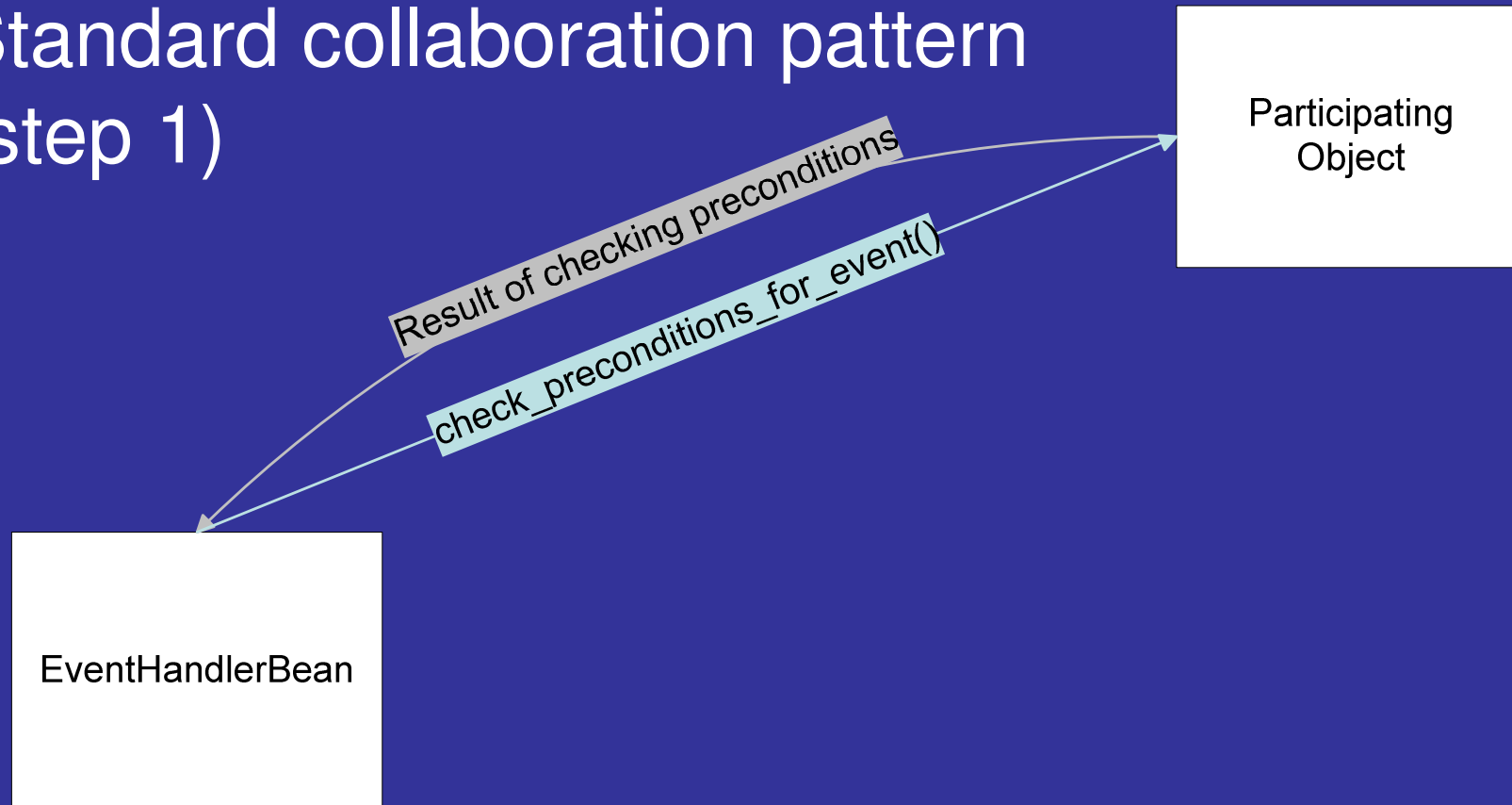
 *How do these generated event handlers work?*

Standard collaboration pattern required!
(see next slide)

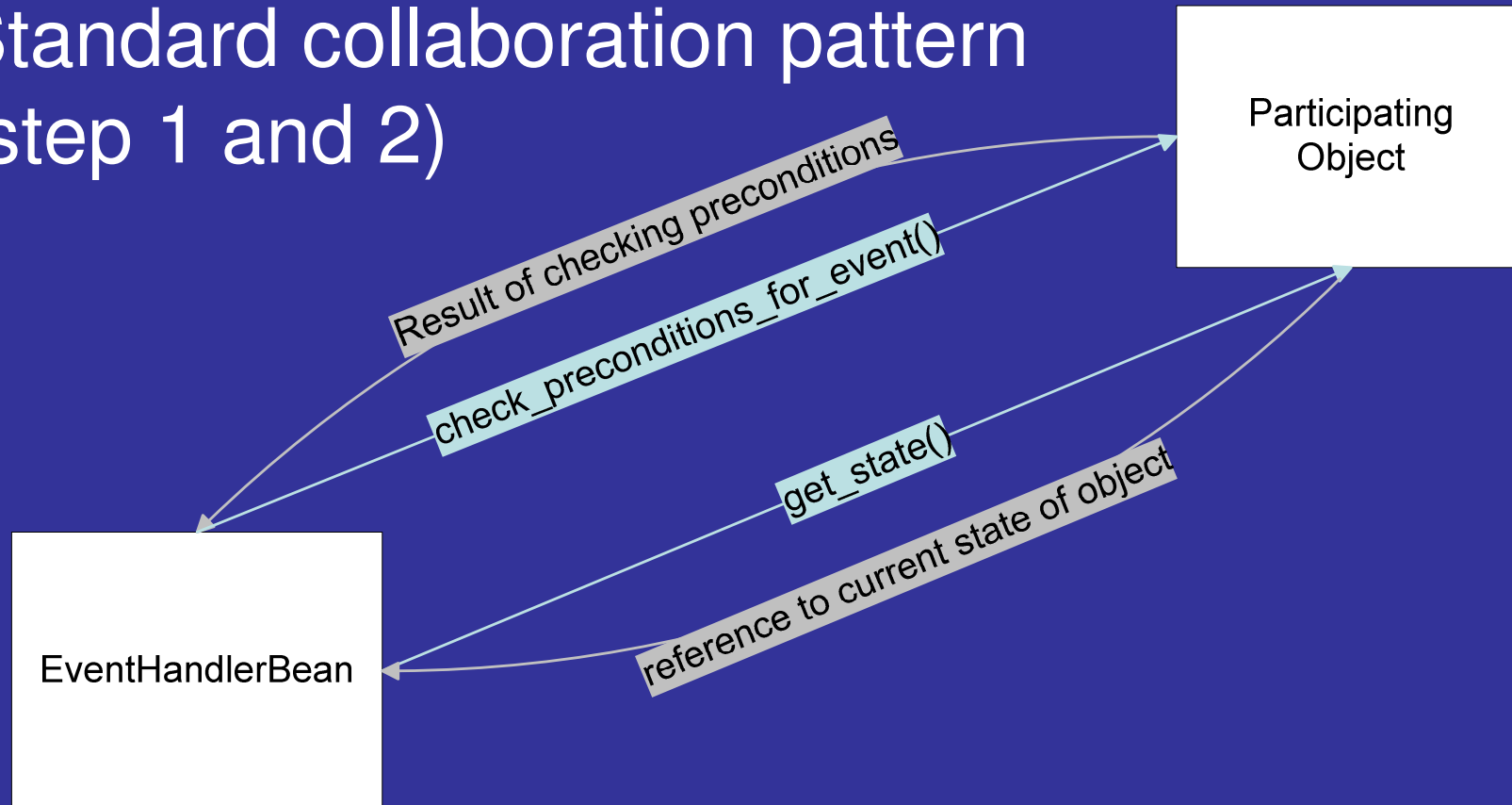
Standard collaboration pattern for generation of an event handler

- Pattern consists of 4 steps
 - Step 1
Checking event preconditions in every participating object
 - Step 2
Checking state conditions in every participating object
 - Step 3
Processing the event in every participating object
 - Step 4
Changing state in every participating object

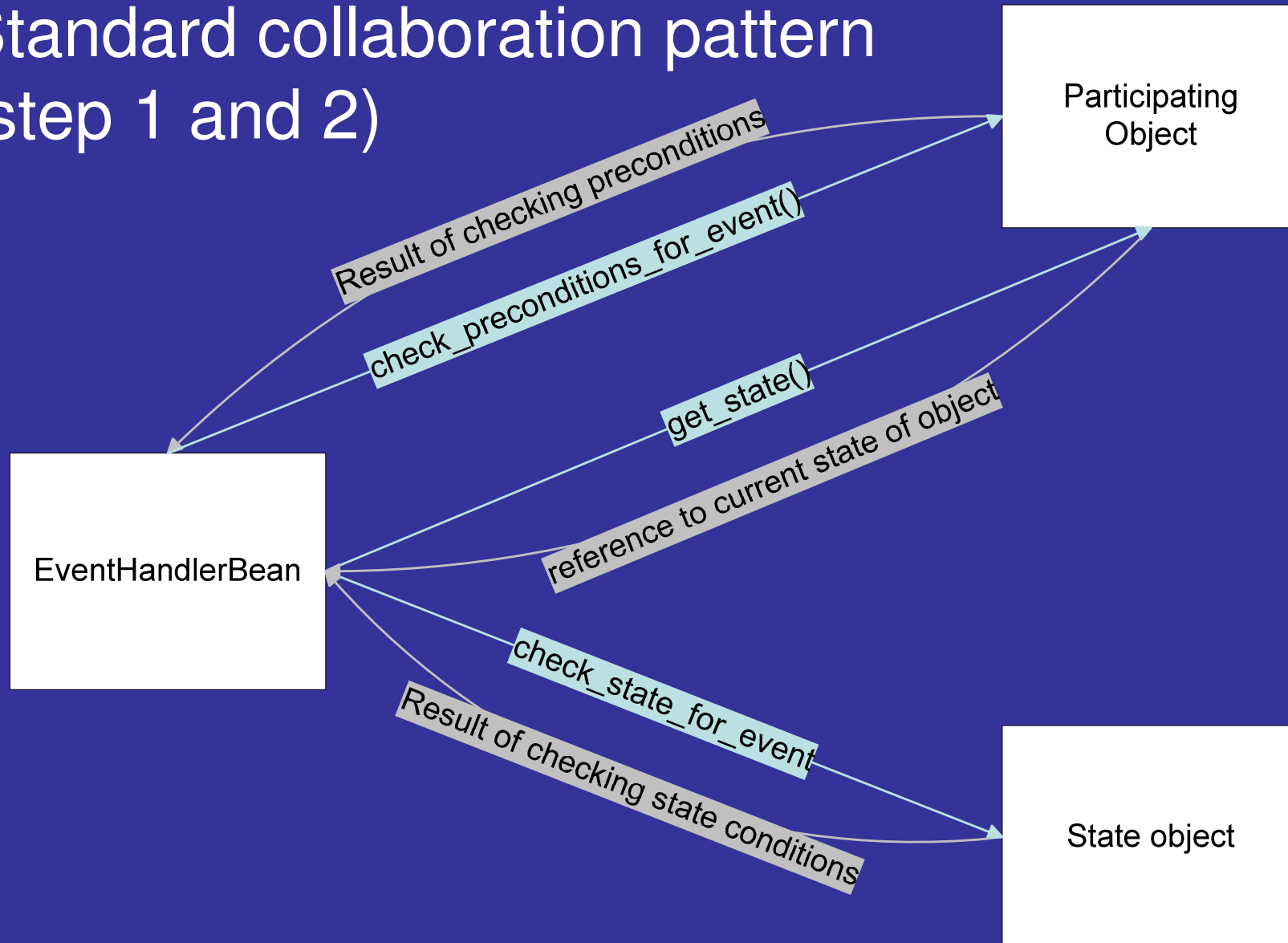
Standard collaboration pattern (step 1)



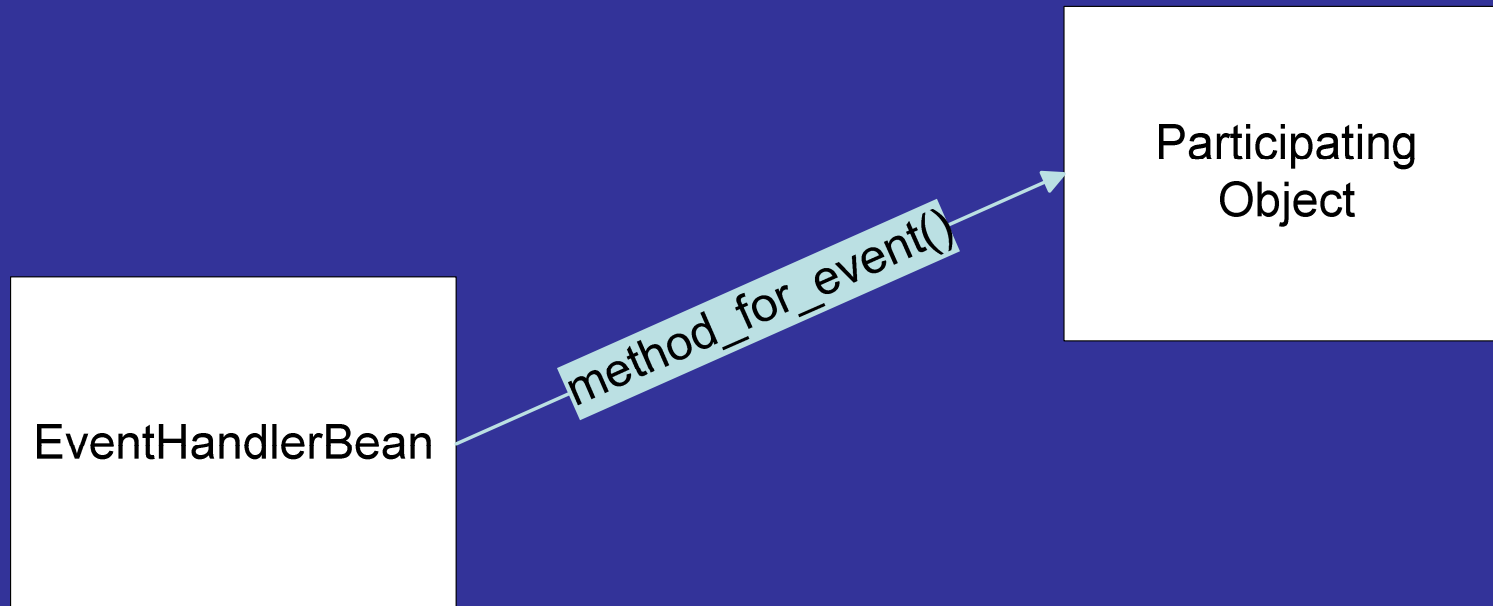
Standard collaboration pattern (step 1 and 2)



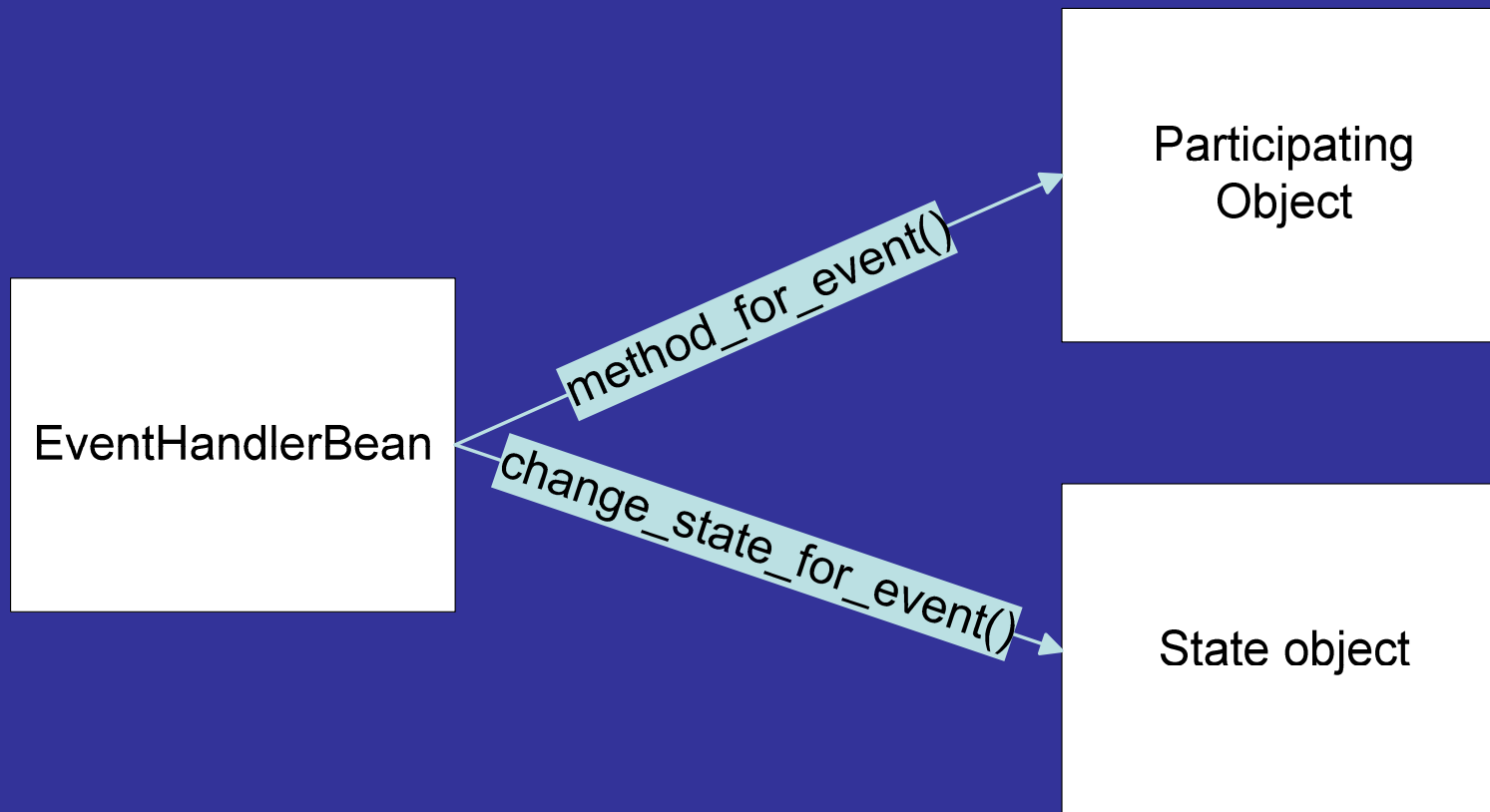
Standard collaboration pattern (step 1 and 2)



Standard collaboration pattern (step 3)



Standard collaboration pattern (step 3 and 4)



PIM

PSM

Transformation rules

Generating the GUI layer

GUI layer

Event handling layer

Persistence layer

- Also strongly based on the defined events
- Details are beyond scope of this presentation...

Main Window:
Select Domain Object



DO Window:
•Select Business Event
•List of Instances



Business Event Window:
•Event Attributes
•Select Participating Instances

Generated event handling layer

Standard collaboration pattern applied for a specific event handler

SessionBean
(all event handlers)

handle_cr_OrderLine()
Involved object types: OrderLine, Order, Product

EventHandlerSessionBean

```
+handle_cr_Customer()
+handle_end_Customer()
+handle_cr_Product()
+handle_end_Product()
+handle_propose()
+handle_accept()
+...()
+handle_cr_OrderLine()
+...()
```

// checking preconditions (step 1)

```
orderLine.check_preconditions_for_cr_OrderLine()
order.check_preconditions_for_cr_OrderLine()
product.check_preconditions_for_cr_OrderLine()
```

// checking state conditions (step 2)

```
orderLine.getState().check_state_for_cr_OrderLine()
order.getState().check_state_for_cr_OrderLine()
product.getState().check_state_for_cr_OrderLine()
```

// event processing (step 3)

```
orderLine.method_for_cr_OrderLine()
order.method_for_cr_OrderLine()
product.method_for_cr_OrderLine()
```

// state modifications (step 4)

```
orderLine.getState().change_state_for_cr_OrderLine()
order.getState().change_state_for_cr_OrderLine()
product.getState().change_state_for_cr_OrderLine()
```

Presentation overview

- Introduction
- Solution
 - Platform Independent Model (PIM)
 - Platform Specific Model (PSM)
 - Transformation Rules
- Conclusion

Conclusions

- If a generated application doesn't work as expected, students are wondering what's wrong?

Model (PIM) *or*
Transformation rules



- MDA can only be manageable with error-free transformation rules
- How can we support the student's process of making a high-quality PIM?

Conclusions (cont.)

- Realisation of error-free transformation
→ central role of 'business events' in our transformation
- Supporting students to make a high-quality PIM
→ advanced consistency techniques implemented in the modelling tool
(by construction, monitoring, analysis, etc.)

Conclusions (cont.)

- Inheritance in OET can yield complex models
→ hard to translate in PSM in an automated way
(future research)
- Still difficult to discover transformation rules that are in general robust to arbitrary combinations of PIM concepts

Questions, comments, reflections, ...

