Using UML for Problem Frame Oriented Software Development

Christine Choppy
LIPN, CNRS UMR 7030
Université Paris 13
Villetaneuse
FRANCE

Gianna Reggio
DISI
Universita di Genova
Genova
ITALY

Motivation

- Problems are difficult to understand and specify
- Problem Frames help to understand problems

Motivation

- Problems are difficult to understand and specify
- Problem Frames help to understand problems
- We propose an associated specification development method
- (other specification methodology issues, e.g. related to a specific language use)

Motivation

- Problems are difficult to understand and specify
- Problem Frames help to understand problems
- We propose an associated specification development method
- General formally grounded specification methodology (Choppy & Reggio 03-04)

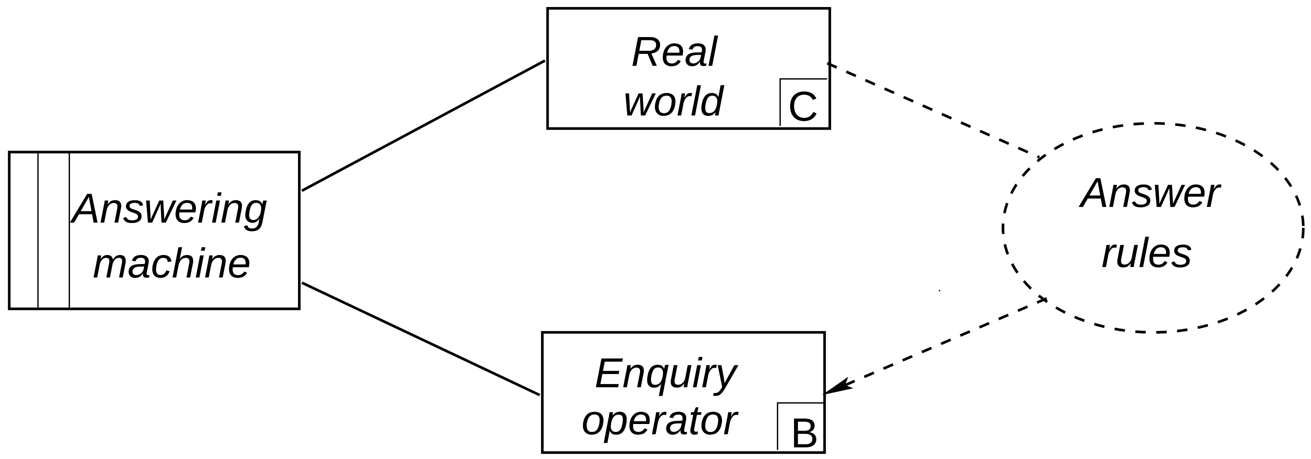
Motivation

- Problems are difficult to understand and specify
- Problem Frames help to understand problems
- We propose an associated specification development method
- General formally grounded specification methodology (Choppy & Reggio 03-04)
- Here approach for UML description: a well founded methodology for UML descriptions
- Help to choose between various constructs, provide a precise, structured and well-founded way
(e.g. Astesiano & Reggio, SEFM 2003)

Motivation

- Problems are difficult to understand and specify
- Problem Frames help to understand problems
- We propose an associated specification development method
- General formally grounded specification methodology (Choppy & Reggio 03-04)
- Here approach for UML description: a well founded methodology for UML descriptions
- Help to choose between various constructs, provide a precise, structured and well-founded way (e.g. Astesiano & Reggio, SEFM 2003)
- Here for the Commanded Information Frame

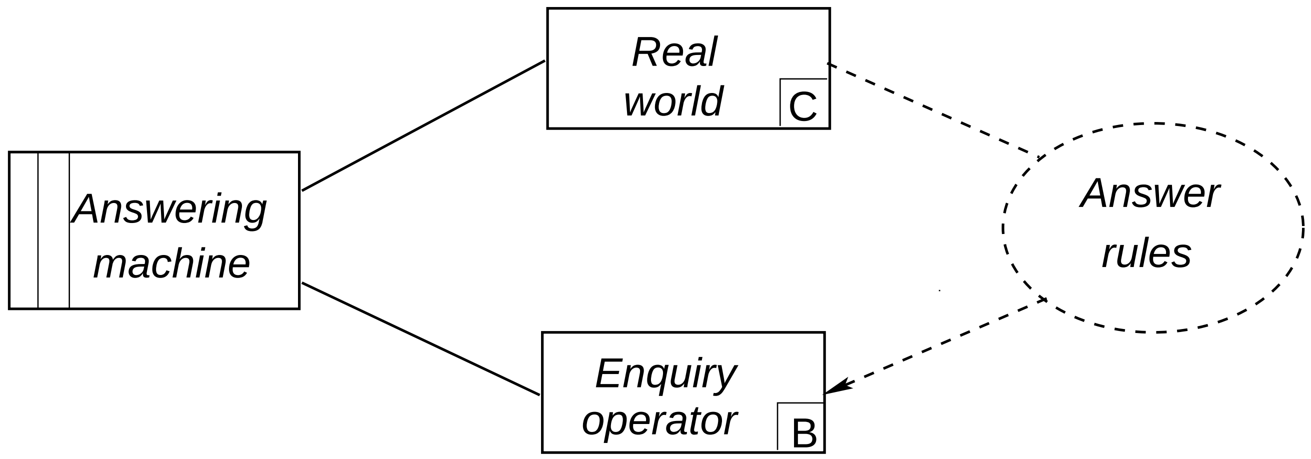
Commanded Information Frame



Design Domain Requirements

B stands for “biddable”(people), C for “causal”(dynamic)

Commanded Information Frame

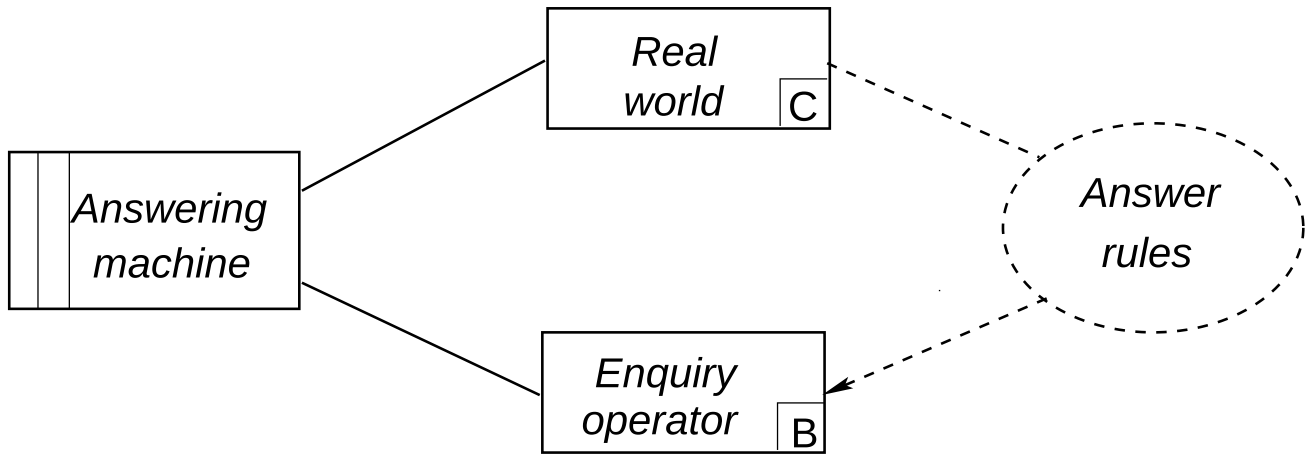


Design Domain Requirements

B stands for “biddable”(people), C for “causal”(dynamic)

“There is some part of the physical world whose states and behavior are needed upon requests from an operator.
The problem is to build a machine that will obtain this information from the world and present it in the required form.”

Commanded Information Frame



Design Domain Requirements

B stands for "biddable"(people), C for "causal"(dynamic)

"There is some part of the physical world whose states and behavior are needed upon requests from an operator.
The problem is to build a machine that will obtain this information from the world and present it in the required form."

Information directly presented to the Enquiry Operator (not by means of a Display)

Method and Outline

- Match a Problem Frame: Domain, Requirements, Design

Method and Outline

- Match a Problem Frame: Domain, Requirements, Design
- Follow the UML description guidelines
  - UML Domain Model
  - UML Requirements Model
  - UML Design Model

Method and Outline

- Match a Problem Frame: Domain, Requirements, Design
- Follow the UML description guidelines
  - UML Domain Model
  - UML Requirements Model
  - UML Design Model
- Running Example: a Company Information System

Domain Model: *Real world*

- class diagram with an active class RealWorld
- a detailed description of the behaviour of RealWorld, or just a small conceptual model containing RealWorld and few other classes.

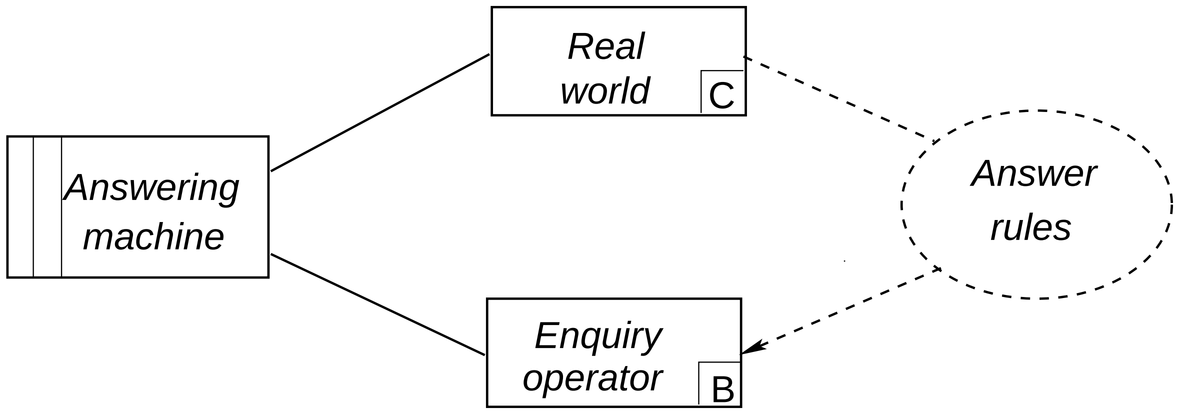
Requirement Specification (1)

- Choppy & Reggio 99: formal specification skeleton for the various parts of the Commanded Information Frame
- events (yield changes in the system state)
- the history of the events that occurred

Requirement Specification (1)

- Choppy & Reggio 99: formal specification skeleton for the various parts of the Commanded Information Frame
- events (yield changes in the system state)
- the history of the events that occurred

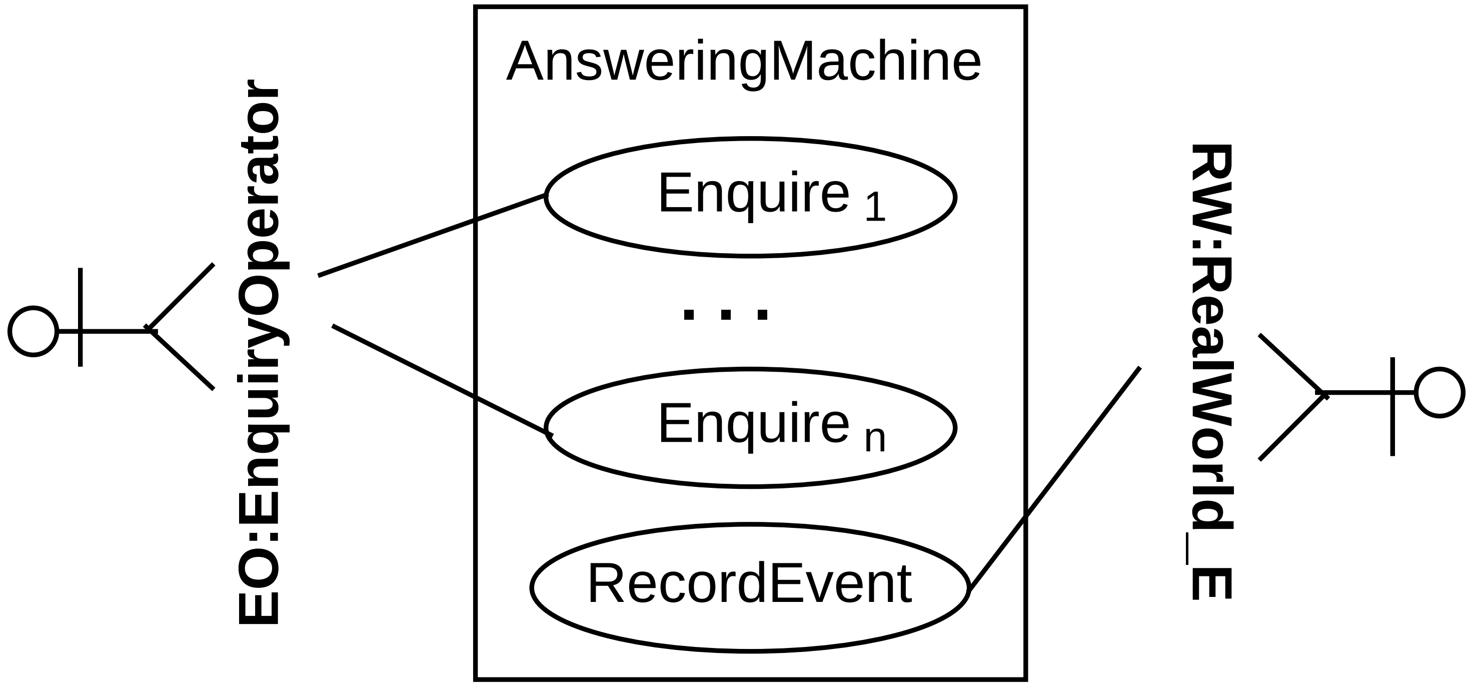
Here



- interface between *Real world* and *Answering machine* labelled by events generated by the *Real world* that convey information about it.

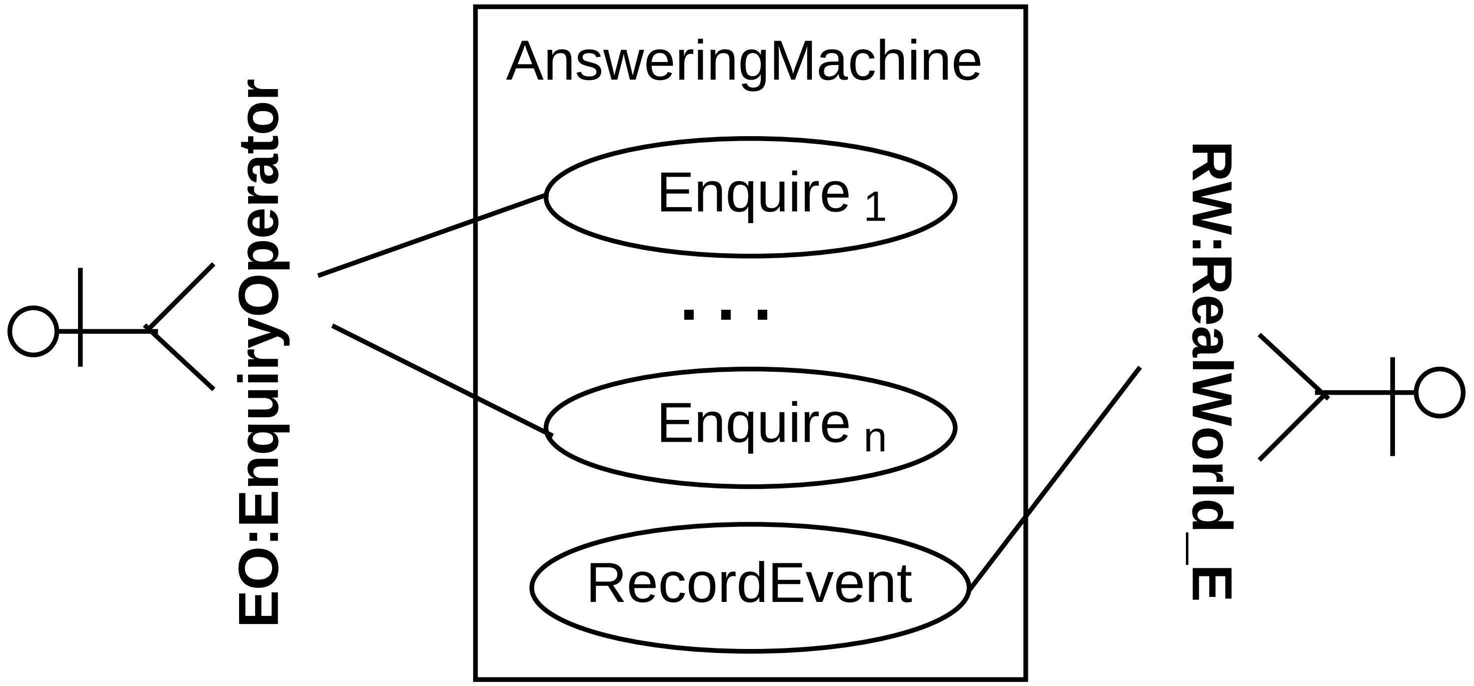
Requirement Specification (2)

- use case diagram



Requirement Specification (2)

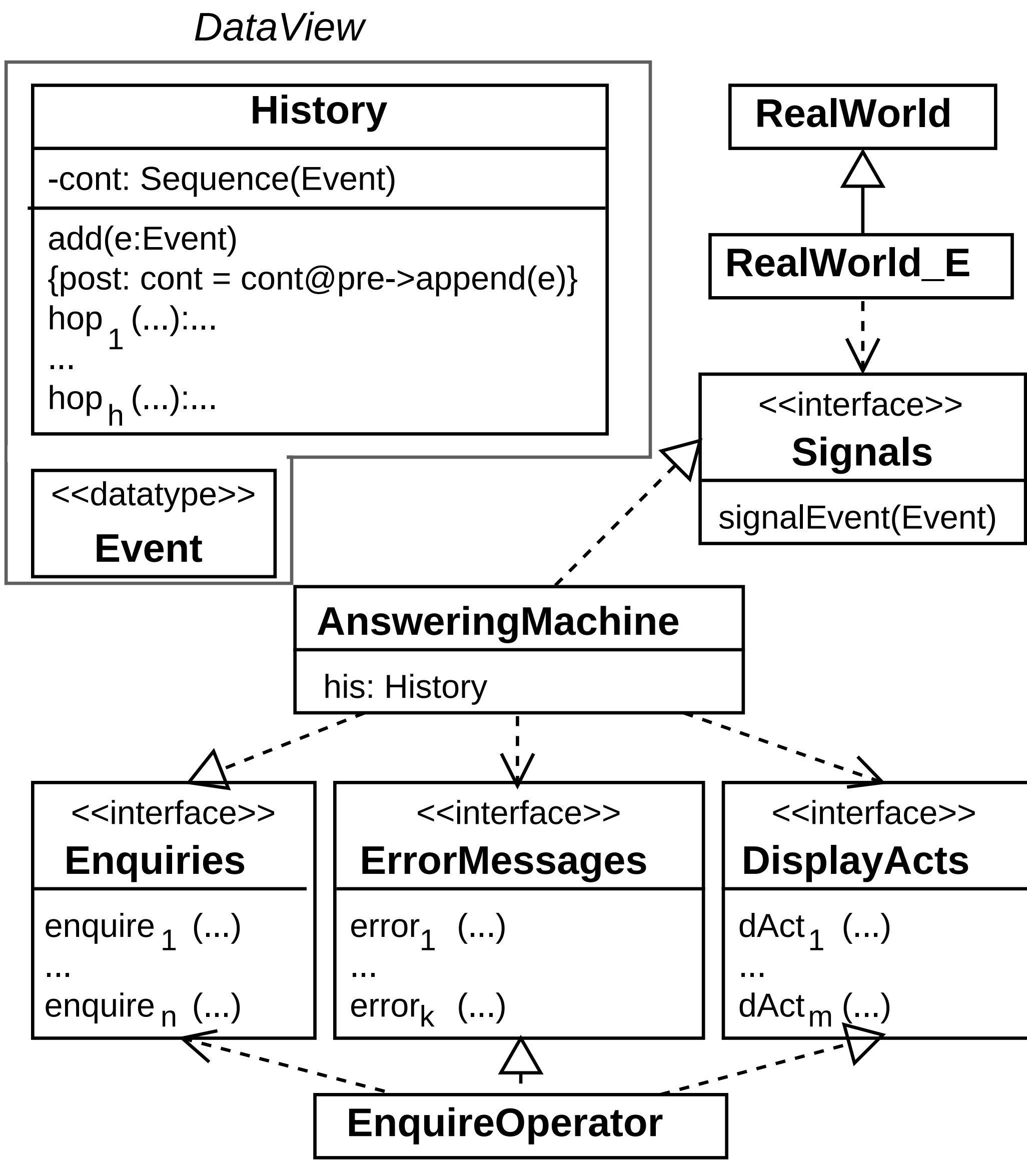
- use case diagram



- class diagram
- use case description (statechart)
- possibly other diagrams and model elements

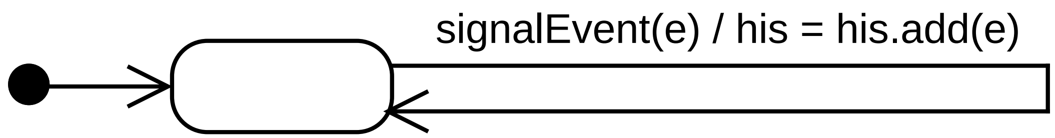
Requirement Spec: class diag

- RealWorld_E specializes RealWorld (signal events to *Answering machine* - Signals interface)
- Event
- AnsweringMachine interacts with RealWorld_E (event signals received) EnquiryOperator (Enquiries, DisplayActs, ErrorMessages)
- AnsweringMachine state: *Real world* past history
- History : hop_i



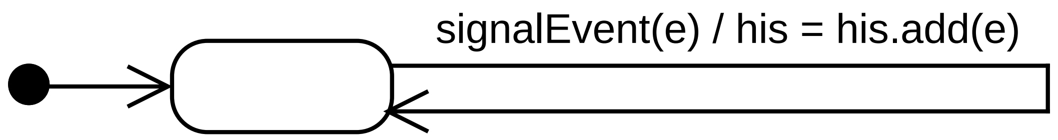
Req Spec: use case description

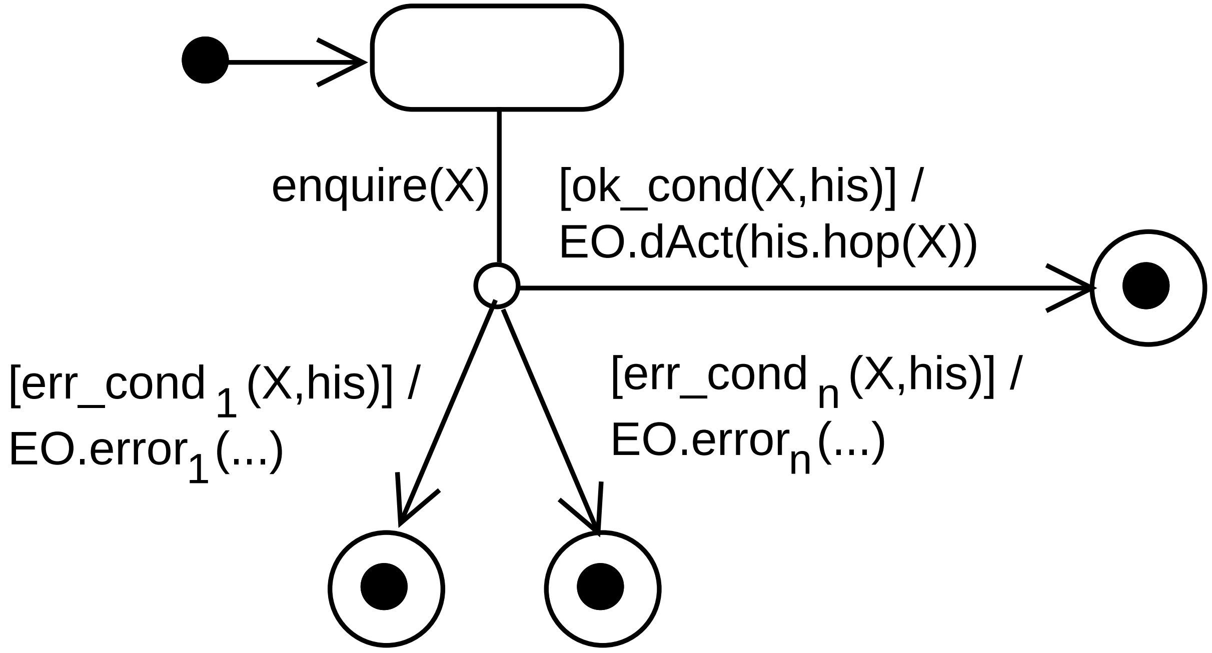
use case RecordEvent



Req Spec: use case description

use case RecordEvent

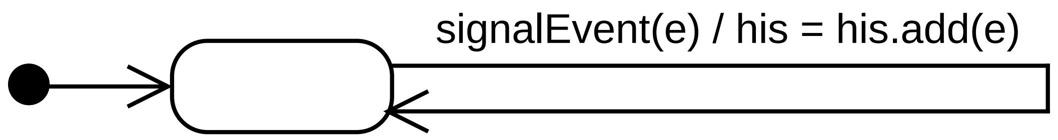


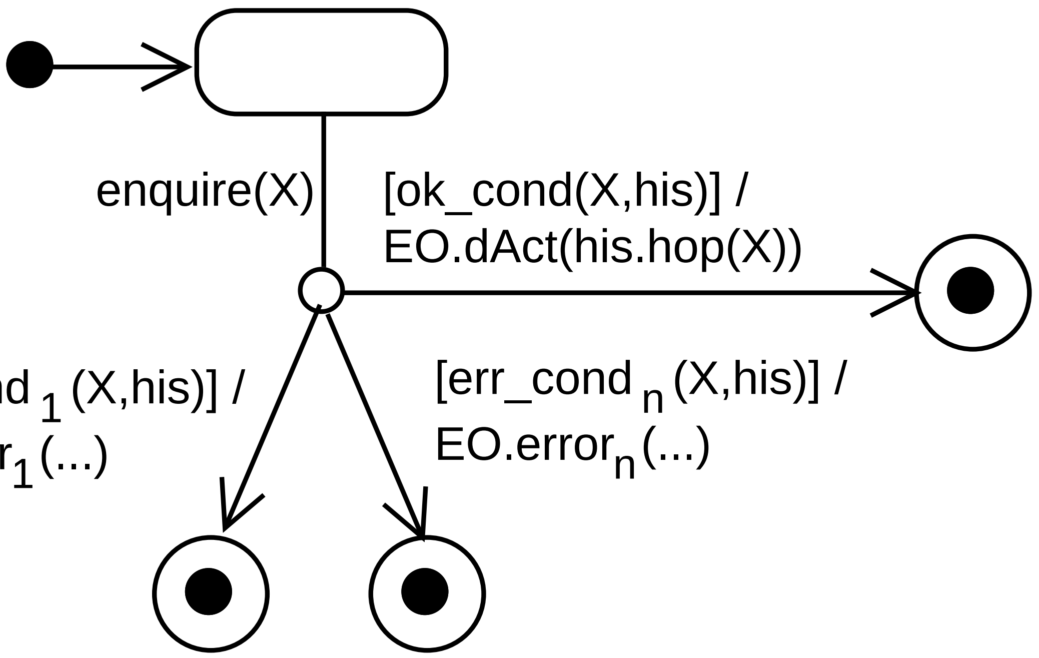


use case Enquire

Req Spec: use case description

use case RecordEvent





use case Enquire

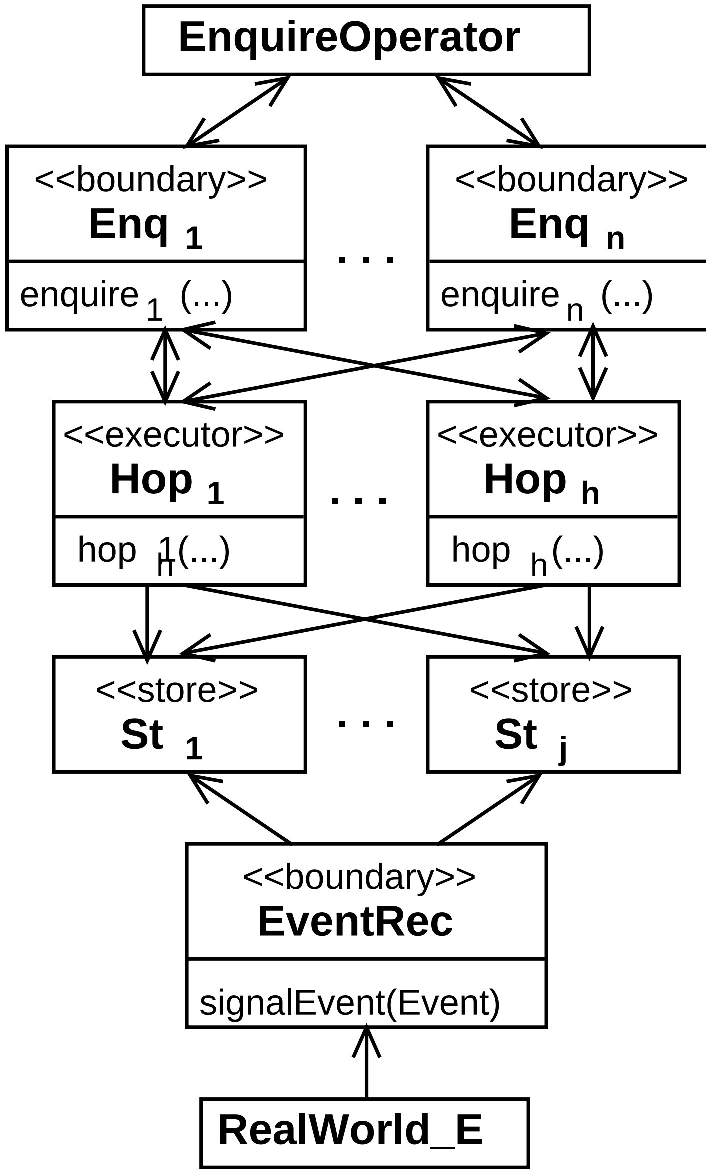
- other diagrams and model elements EnquiryOperator, RealWorld_E, and Event

Design Spec

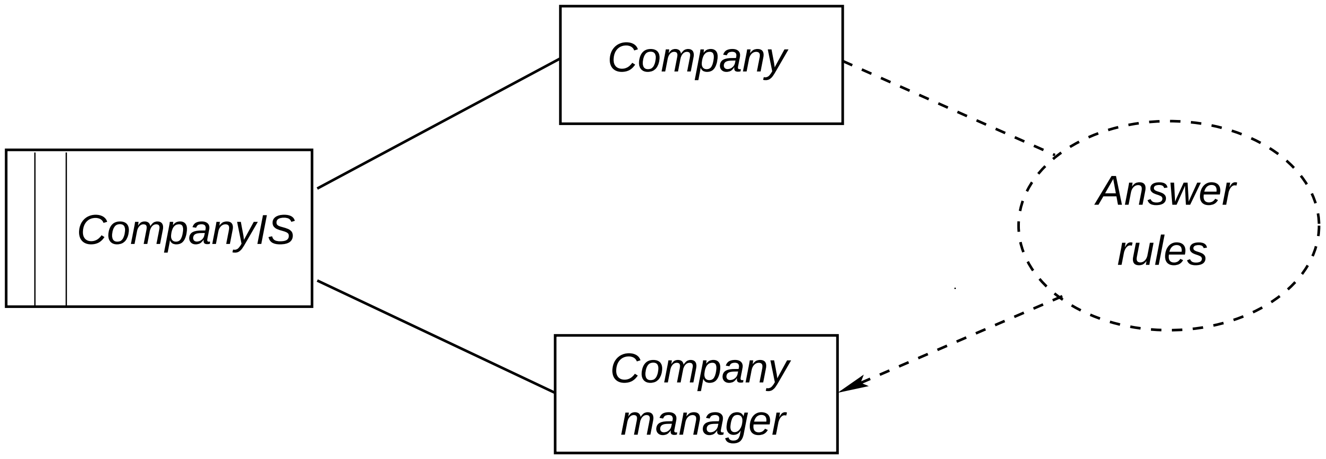
A class diagram

- EnquiryOperator & RealWorld_E external entities interacting with *Answering machine*
- three kinds of (stereotype) classes:
  ≪boundary≫ (sys interaction with external entities)
  ≪executor≫ (system activities)
  ≪store≫ (persistent data)

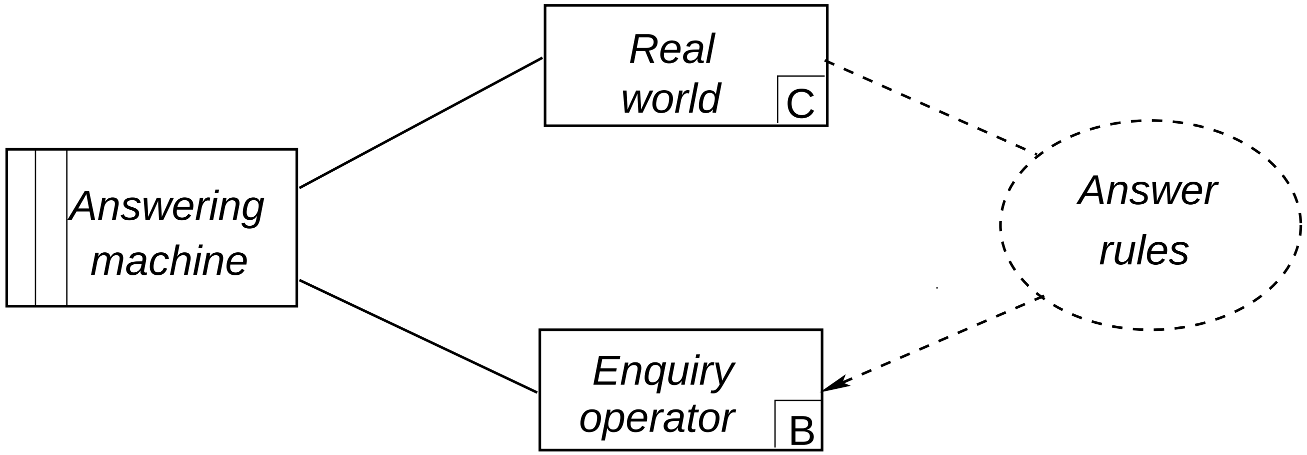
other diagrams and constraints:
behaviour of each class
- bound., exec. : statecharts
- methods, pre/post : store op



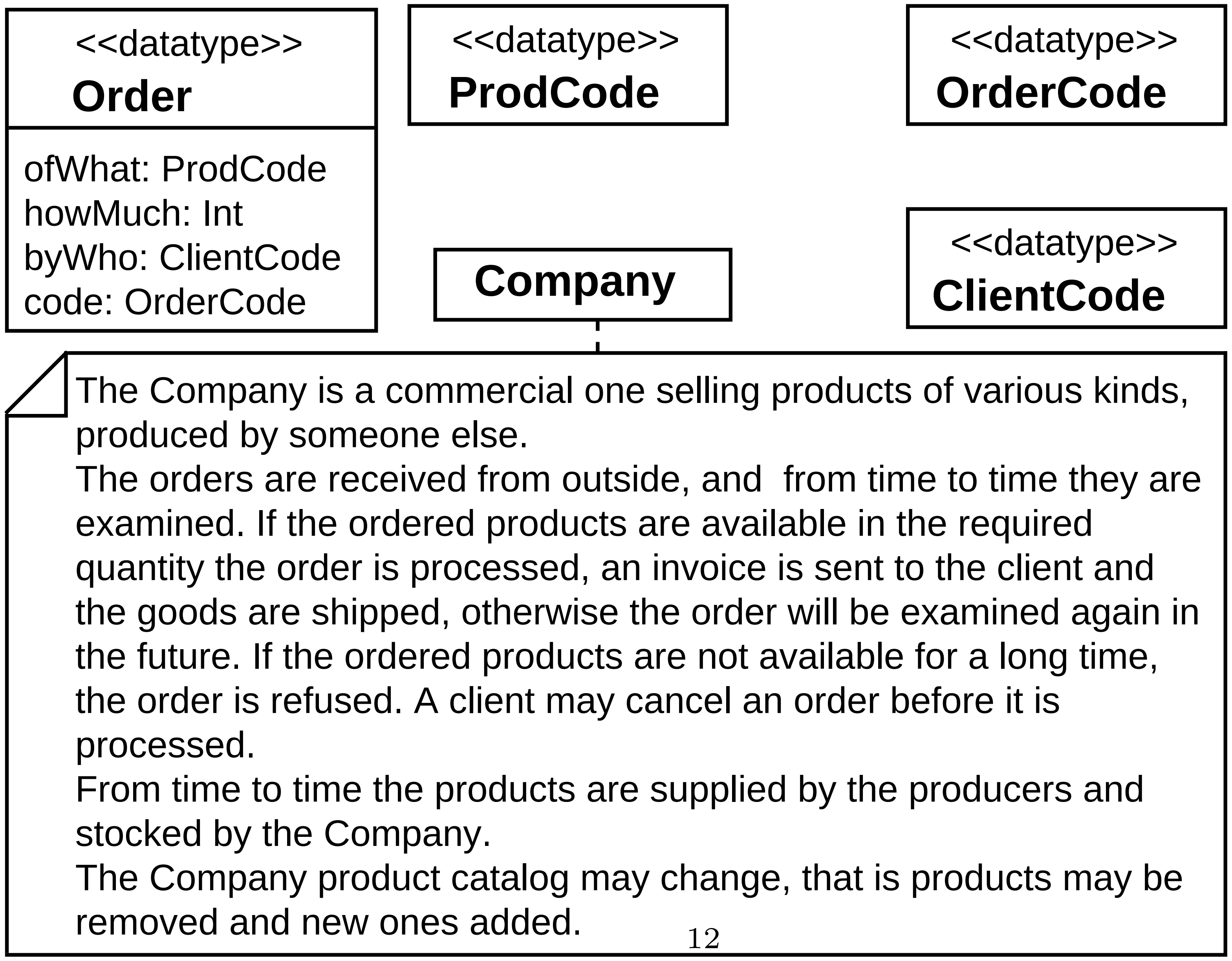
Company Information System



Matching the commanded information problem frame

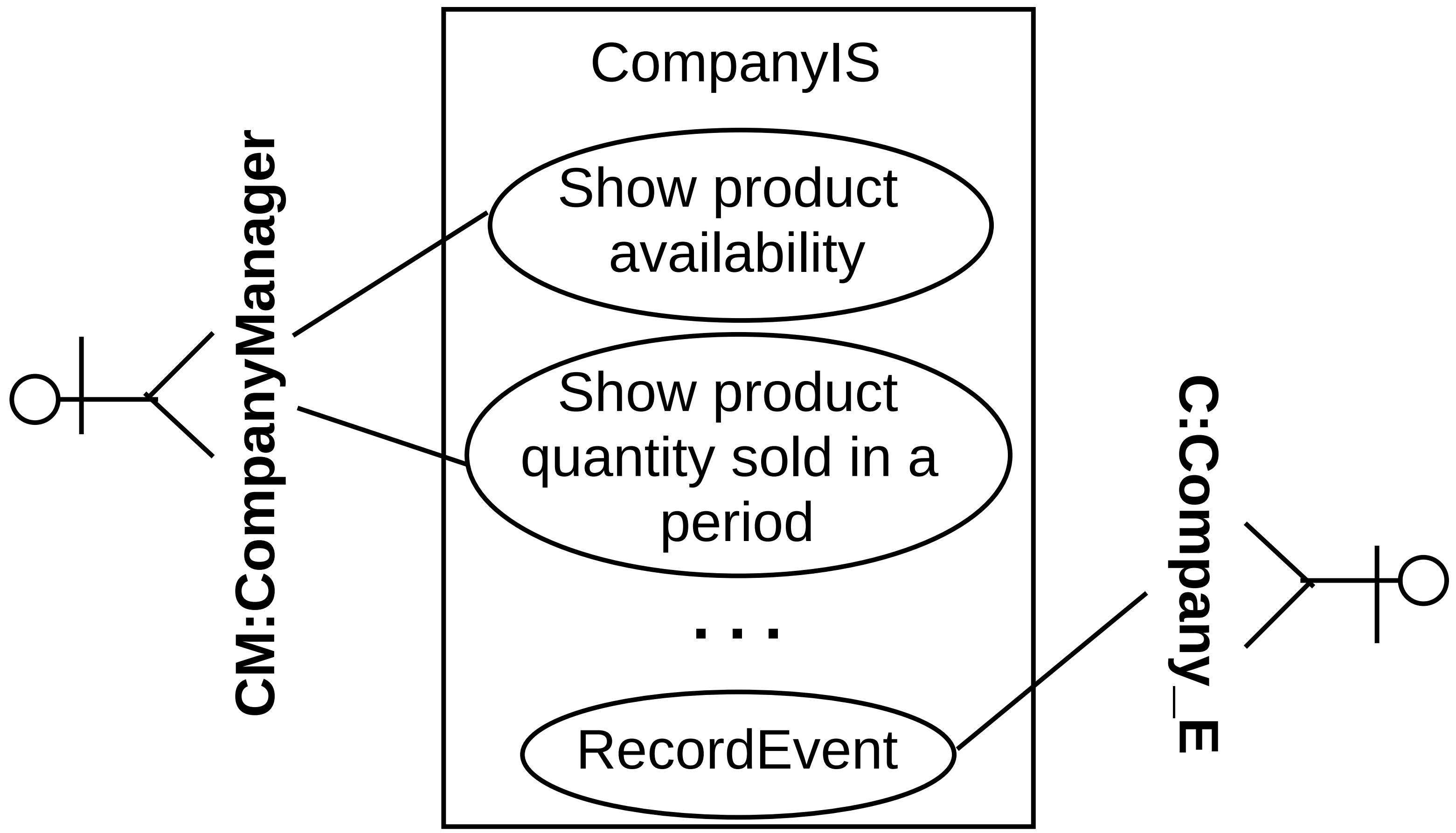


Company Domain Model

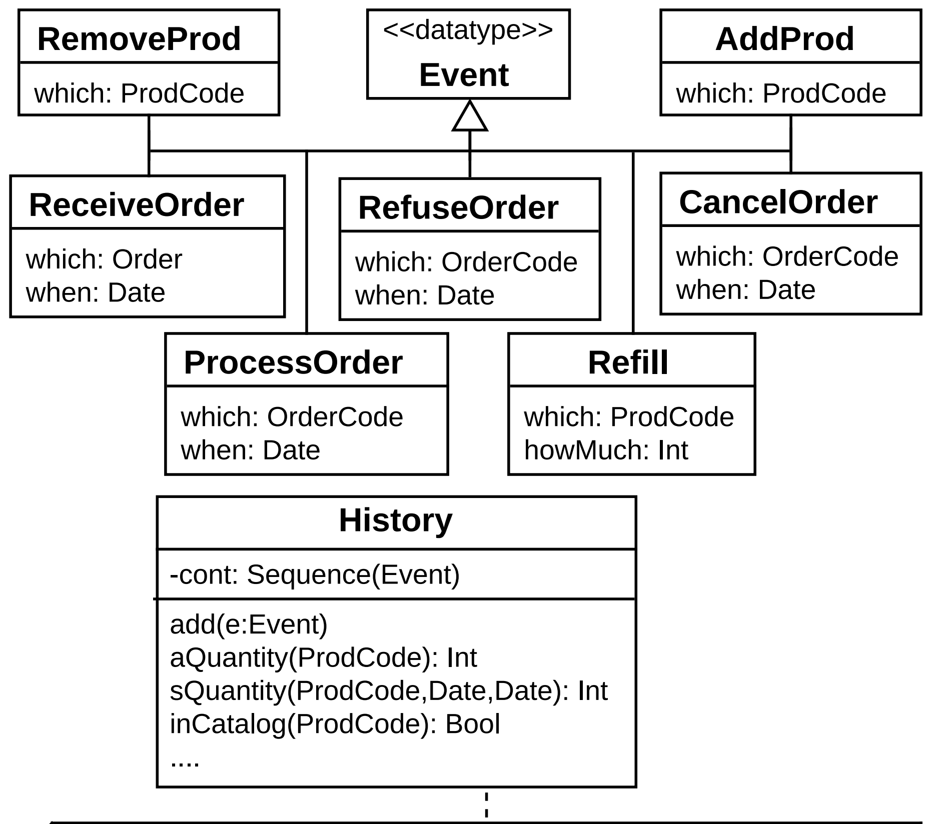


The Company is a commercial one selling products of various kinds, produced by someone else.
The orders are received from outside, and from time to time they are examined. If the ordered products are available in the required quantity the order is processed, an invoice is sent to the client and the goods are shipped, otherwise the order will be examined again in the future. If the ordered products are not available for a long time, the order is refused. A client may cancel an order before it is processed.
From time to time the products are supplied by the producers and stocked by the Company.
The Company product catalog may change, that is products may be removed and new ones added.

Company Requirement Spec 1

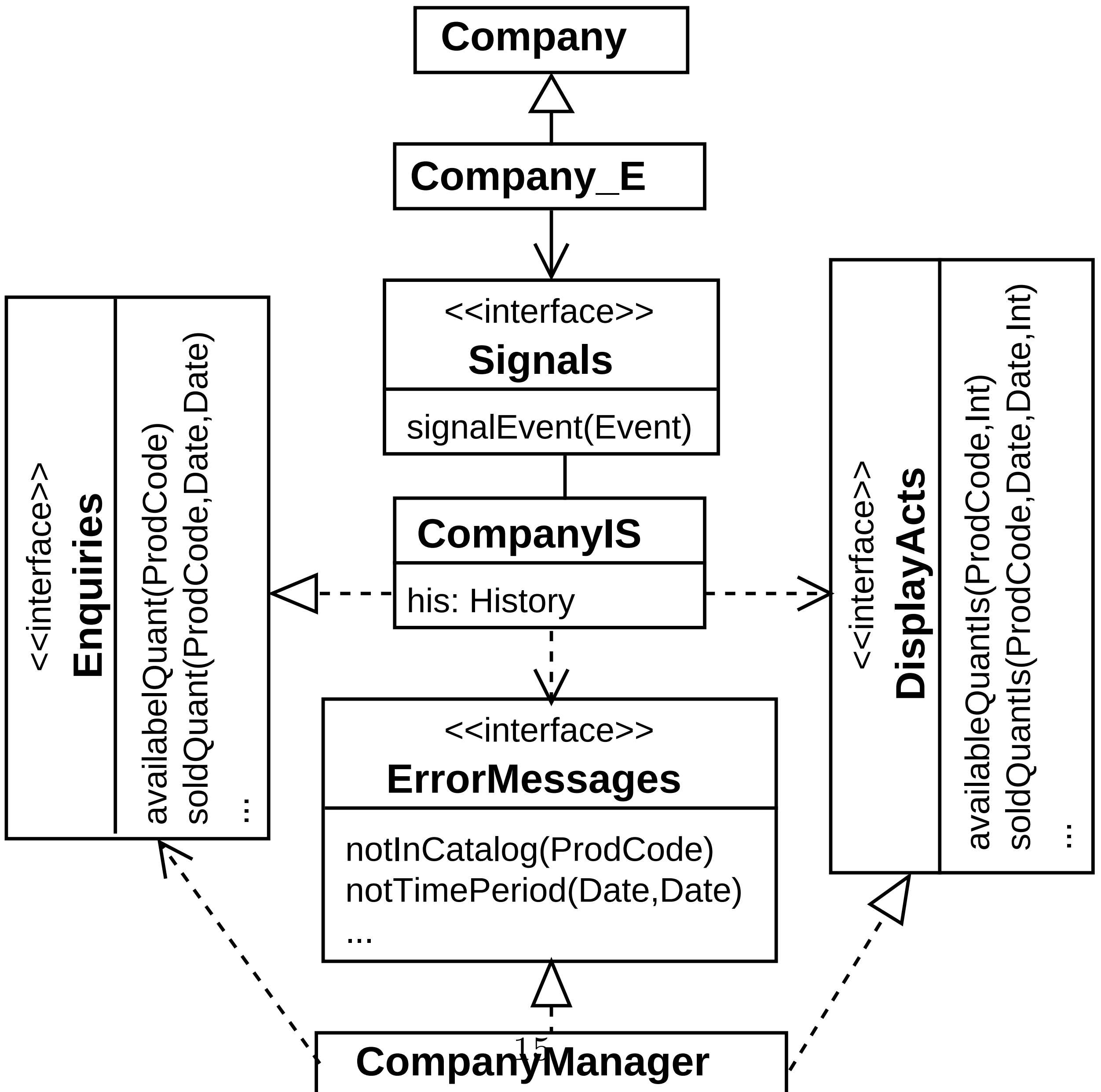

CompanyIS
Show product availability
Show product quantity sold in a period
. . .
RecordEvent
CM:CompanyManager
C:Company_E

Company Req Spec 2 (class 1)

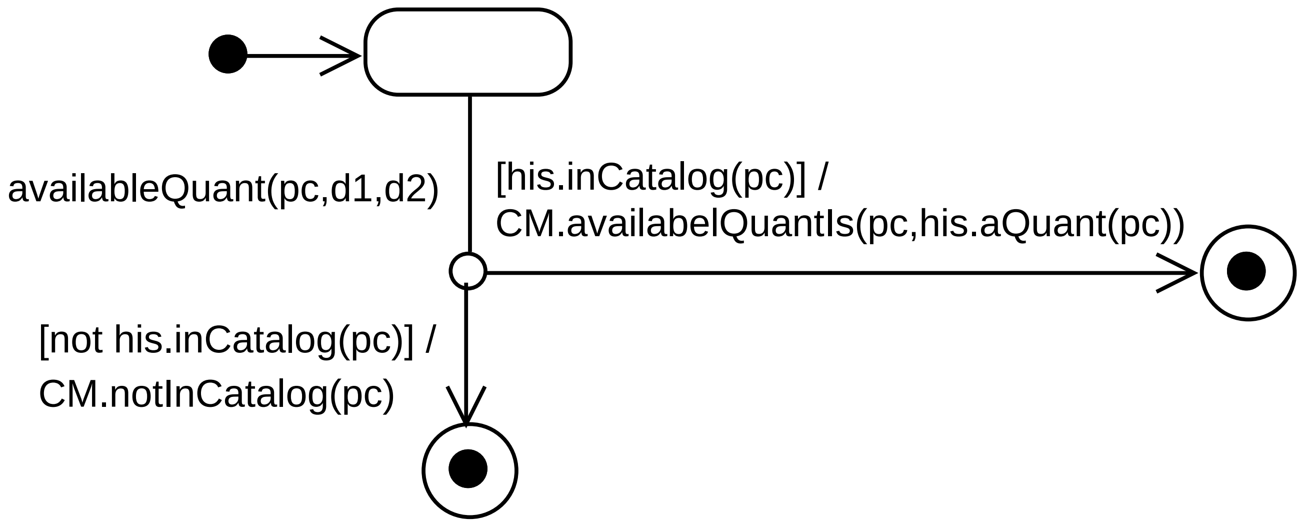


```
inCatalog(pc:ProdCode): Bool post:
result = exists i >= 1 s.t.
    self.cont[i]  is the addition of pc and
    for all j > i self.cont[j] is not the removing of pc

aQuantity(pc:ProdCode):  Int post:
result = cont -> iterate(e: Event; aQuant = 0 |                    )
    if e.hasType(Refill) and pc = e.which then
       aQuant + e.howMuch
  else  if e.hasType(ProcessOrder) and pc = e. which.ofWhat then
      aQuant-e.which.howMuch
  else aQuant )

sQuantity(pc:ProdCode, d1,d2: Date): Int post:
```

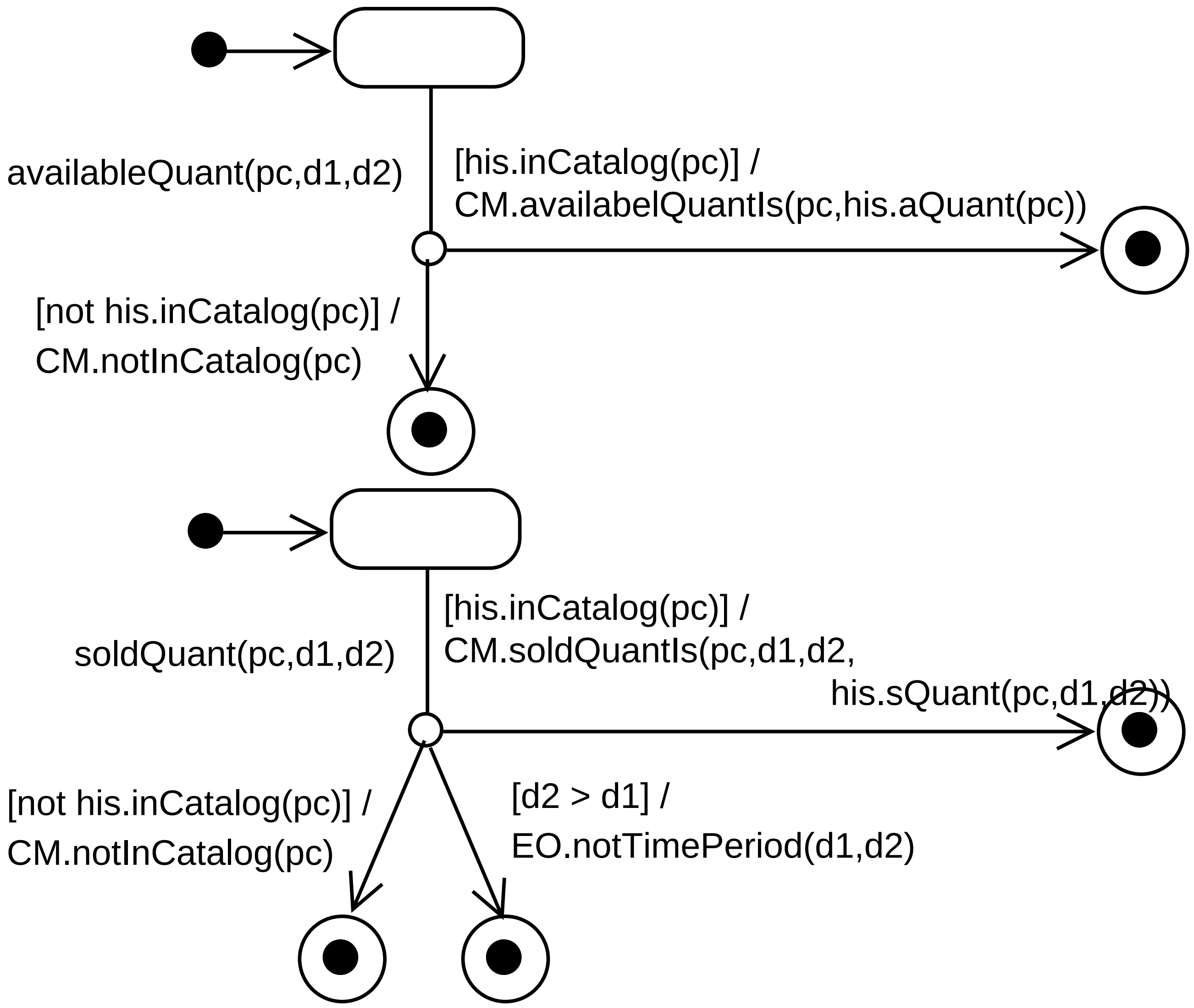
Company Req Spec 3 (class 2)



Company Requirement Spec 4



Company Requirement Spec 4



Conclusion, . . .

- A well founded methodology for UML descriptions in relationship with the basic Problem Frames (following Jackson’s idea was that, once a convenient problem frame is identified for a given problem, then the appropriate development method could be available).

Conclusion, . . .

- A well founded methodology for UML descriptions in relationship with the basic Problem Frames (following Jackson’s idea was that, once a convenient problem frame is identified for a given problem, then the appropriate development method could be available).
- Precise guided use of UML constructs (time gain & concentration on relevant development aspects), for domain, requirements and design

Conclusion, . . .

- A well founded methodology for UML descriptions in relationship with the basic Problem Frames (following Jackson’s idea was that, once a convenient problem frame is identified for a given problem, then the appropriate development method could be available).
- Precise guided use of UML constructs (time gain & concentration on relevant development aspects), for domain, requirements and design
- founded on formal specification experience

Conclusion, . . .

- A well founded methodology for UML descriptions in relationship with the basic Problem Frames (following Jackson's idea was that, once a convenient problem frame is identified for a given problem, then the appropriate development method could be available).
- Precise guided use of UML constructs (time gain & concentration on relevant development aspects), for domain, requirements and design
- founded on formal specification experience
- the developer has to explicitly consider and describe the existing parts of the real world interacting with the system to be developed.

Related work, and . . .

- Previous work in the same spirit to guide formal specifications of complex systems in relationships with Problem Frames and for different formal specification languages.

Related work, and . . .

- Previous work in the same spirit to guide formal specifications of complex systems in relationships with Problem Frames and for different formal specification languages.
- Formal specifications skeletons (CASL Common Algebraic Specification Language `http:/cofi.info` , CASL-LTL extended with Labelled Transition Logics, temporal)
  for Translation/JSP, and Information System (IS)
  (Choppy & Reggio WADT'99 LNCS 1827)

  (LOTOS), together with connections to architectural styles
  (Choppy & Heisel WADT'02 LNCS 2755)

Perspectives

- Report with similar approach for other problem frames: Transformation, Required Behaviour, Commanded Behaviour.

Perspectives

- Report with similar approach for other problem frames: Transformation, Required Behaviour, Commanded Behaviour.
- May be completed with Required Information, Workpieces

Perspectives

- Report with similar approach for other problem frames: Transformation, Required Behaviour, Commanded Behaviour.
- May be completed with Required Information, Workpieces
- to get a complete development approach with the benefit of the basic problem frames structuring concepts.

Perspectives

- Report with similar approach for other problem frames: Transformation, Required Behaviour, Commanded Behaviour.
- May be completed with Required Information, Workpieces
- to get a complete development approach with the benefit of the basic problem frames structuring concepts.
- may be used with or without (any) formal specifications (try to keep both worlds happy . . . and efficient !)

Perspectives

- Report with similar approach for other problem frames: Transformation, Required Behaviour, Commanded Behaviour.
- May be completed with Required Information, Workpieces
- to get a complete development approach with the benefit of the basic problem frames structuring concepts.
- may be used with or without (any) formal specifications (try to keep both worlds happy . . . and efficient !)

THE END ! THANKS FOR YOUR ATTENTION !

Other issues

- 6 basic problem frames ?

Other issues

- 6 basic problem frames ?
- explore new problem frames ?
- failing to match a problem frame is interesting !

Other issues

- 6 basic problem frames ?
- explore new problem frames ?
- failing to match a problem frame is interesting !
- Composition ?
  associate on problem frame with each use case
  (goal level?)
  compose developed solutions through component based architectures
  Choppy & Heisel, AFADL'04 (June)