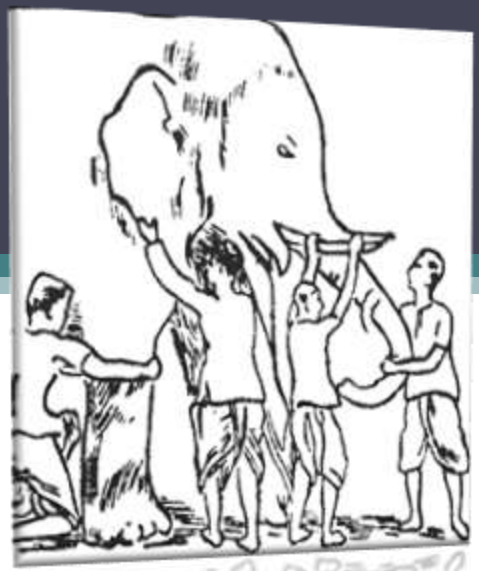
Architecture and Design of Multi-Agent Systems for Business-Process Control and Traceability Applications



HIGHER SCHOOL OF ECONOMICS
NATIONAL RESEARCH UNIVERSITY
NIZHNY NOVGOROD

Prof. Eduard BABKIN
eababkin@hse.ru

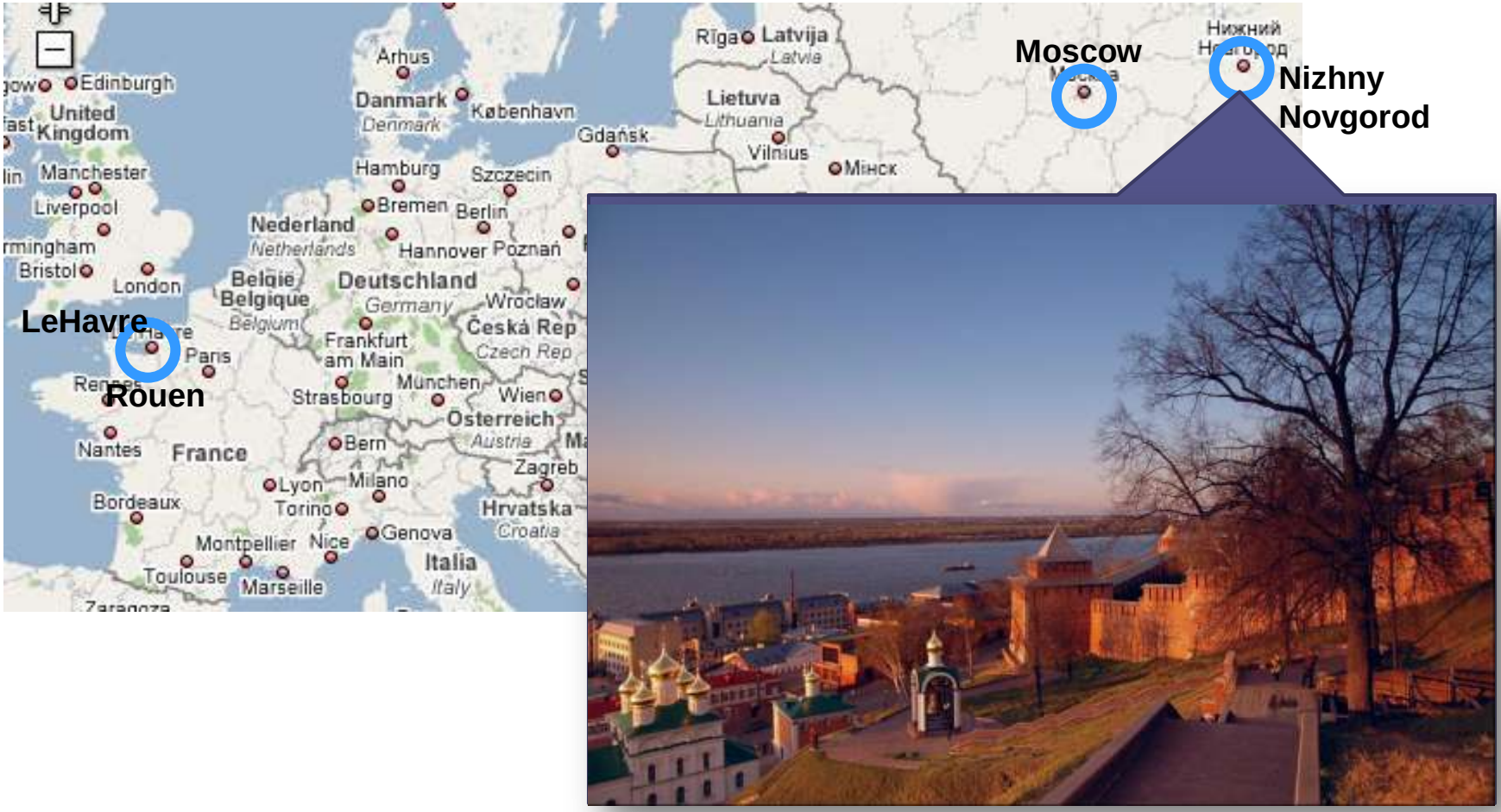
HIGHER SCHOOL OF ECONOMICS
NATIONAL RESEARCH UNIVERSITY

•The HSE has grown into a multiregional structure housing its education and research facilities in four campuses:

- HSE – Moscow
- HSE – Saint-Petersburg
- HSE – Perm
- HSE – Nizhny Novgorod (the oldest and largest among HSE regional campuses)

Nizhny Novgorod

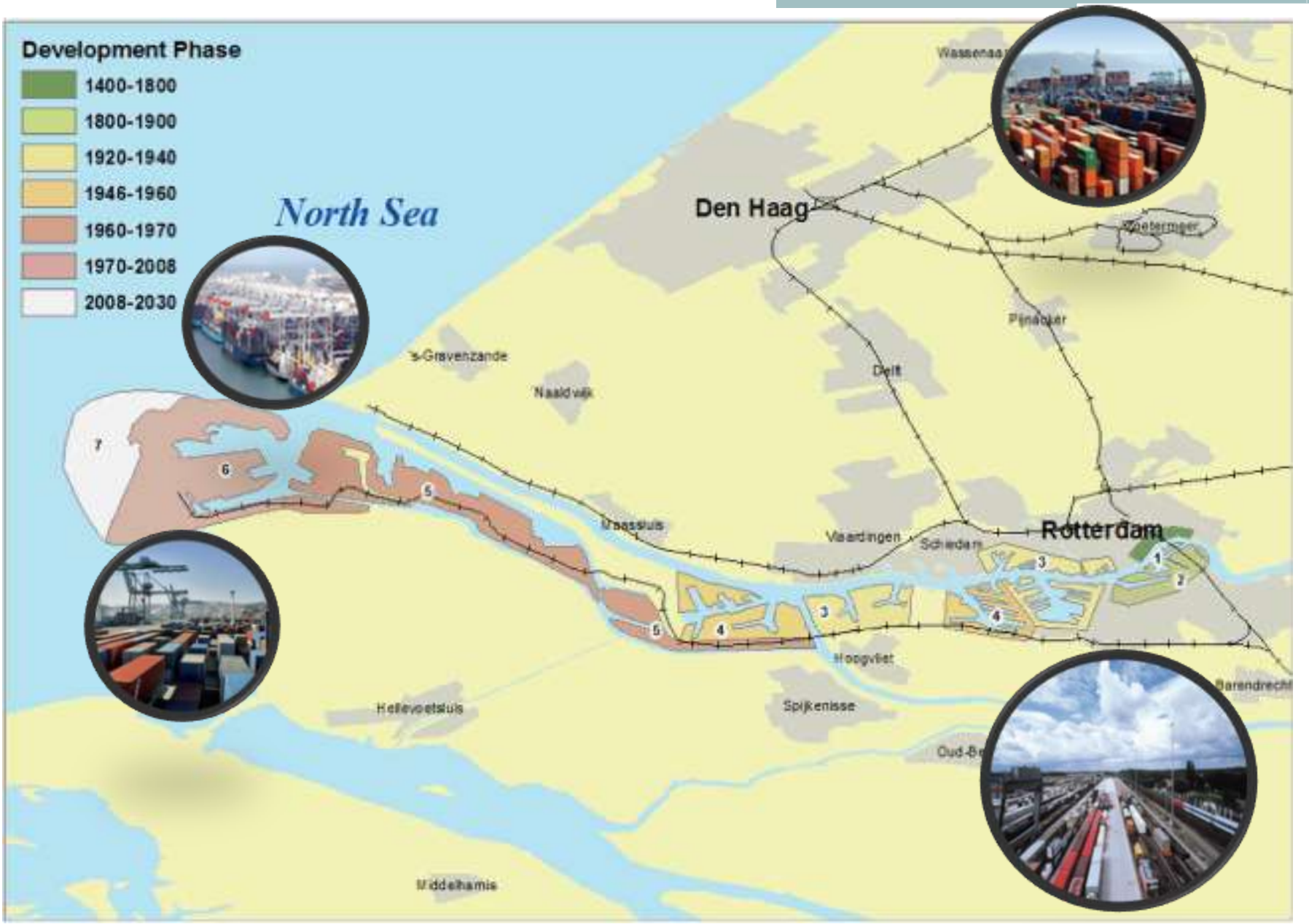
what does it means in Russia – “not so far from Moscow” ?



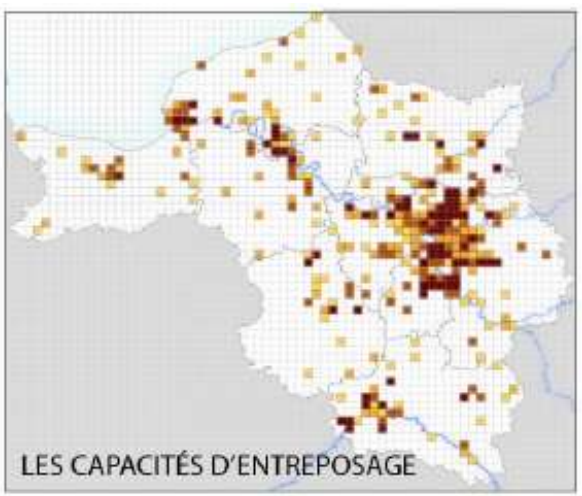
Agenda

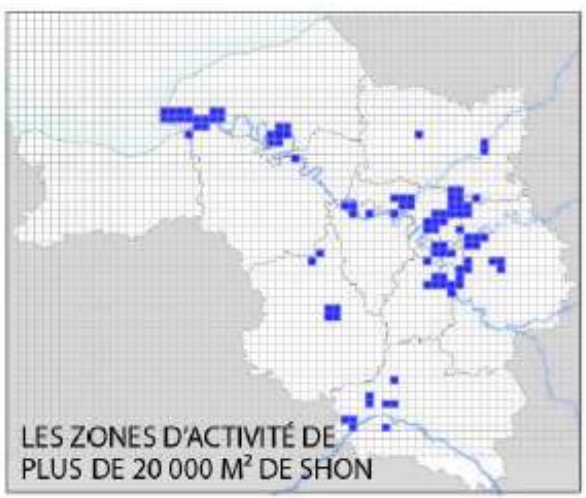
1. Traceability Domain Overview
2. Key Problems in the Traceability Domain
3. Architecture Principles of Multi-agent Systems
4. Proposed Solution (CACS)
5. Outcomes
6. Prospective Directions

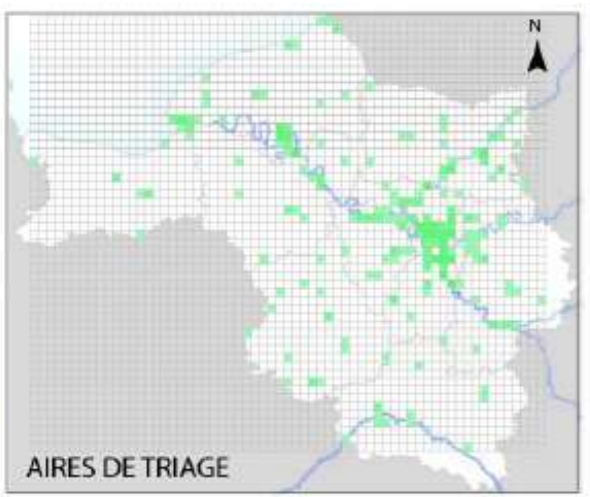
Traceability applications as integral parts of enterprise information systems are aimed to trace the state of objects, discover information regarding its past states, correctly predict future states and estimate different kinds of risks.

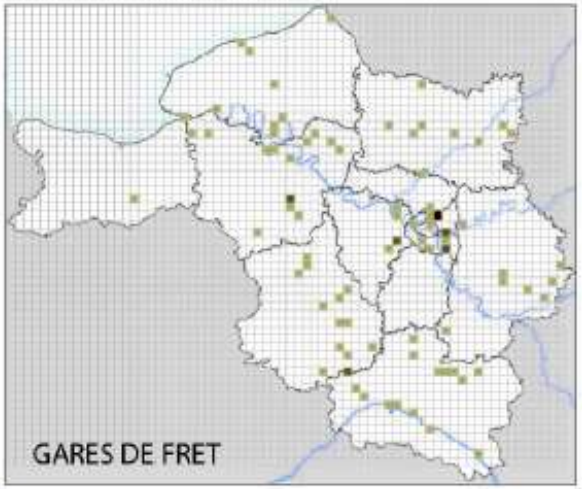

Development Phase
1400-1800
1800-1900
1920-1940
1946-1960
1960-1970
1970-2008
2008-2030
North Sea
Den Haag
Rotterdam
Delft
Vlaardingen
Schiedam
Maassluis
Naaldwijk
's-Gravenzande
Hellevoetsluis
Spijkenisse
Hoogvliet
Middelharnis
Pijnacker
1
2
3
4
5
6
7

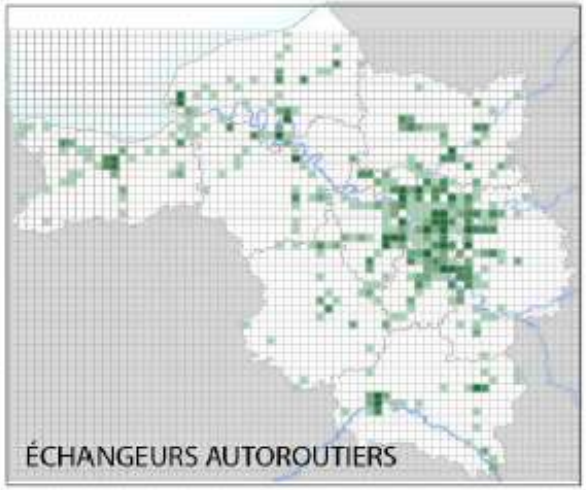
Larger Scale…

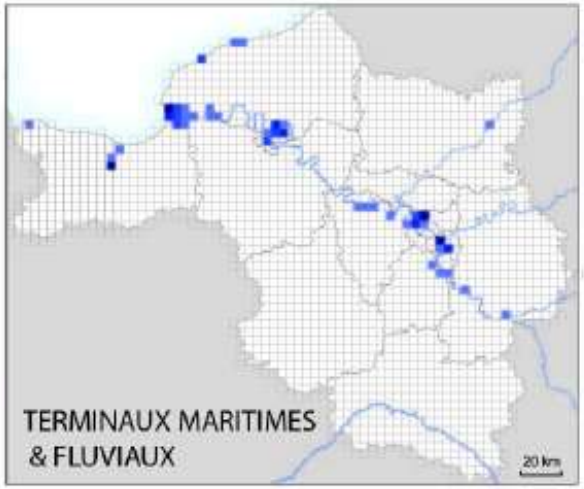




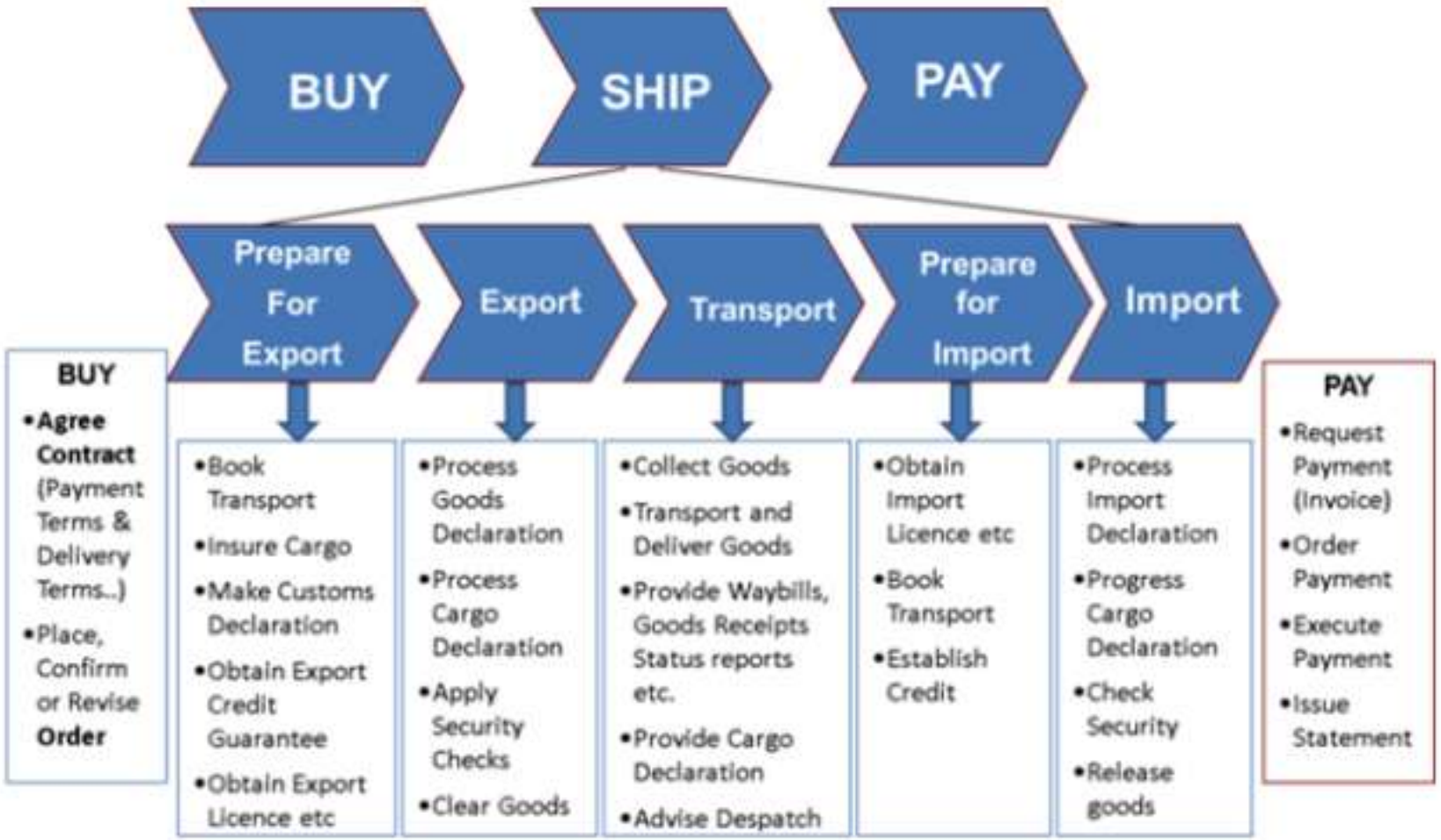




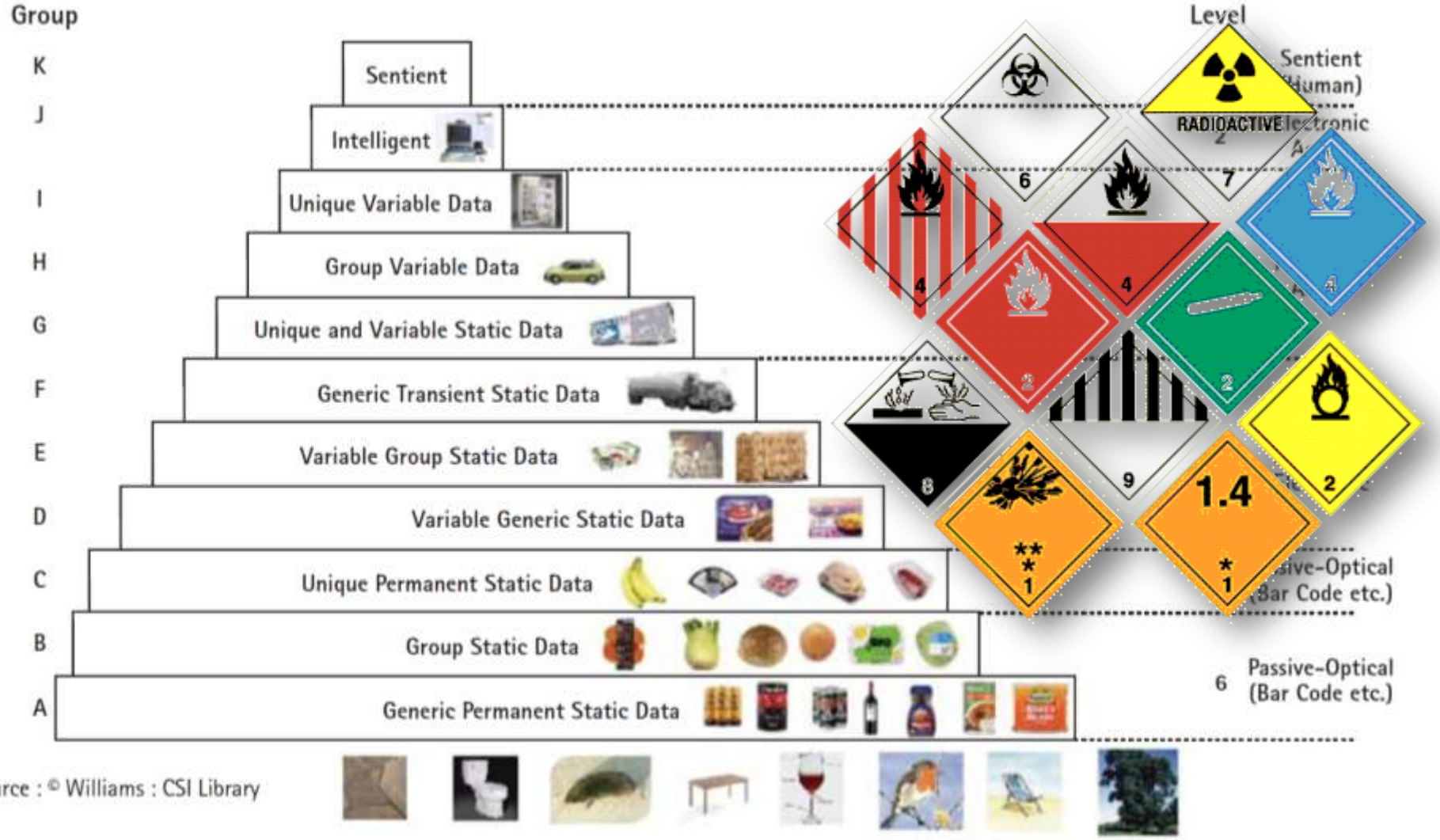




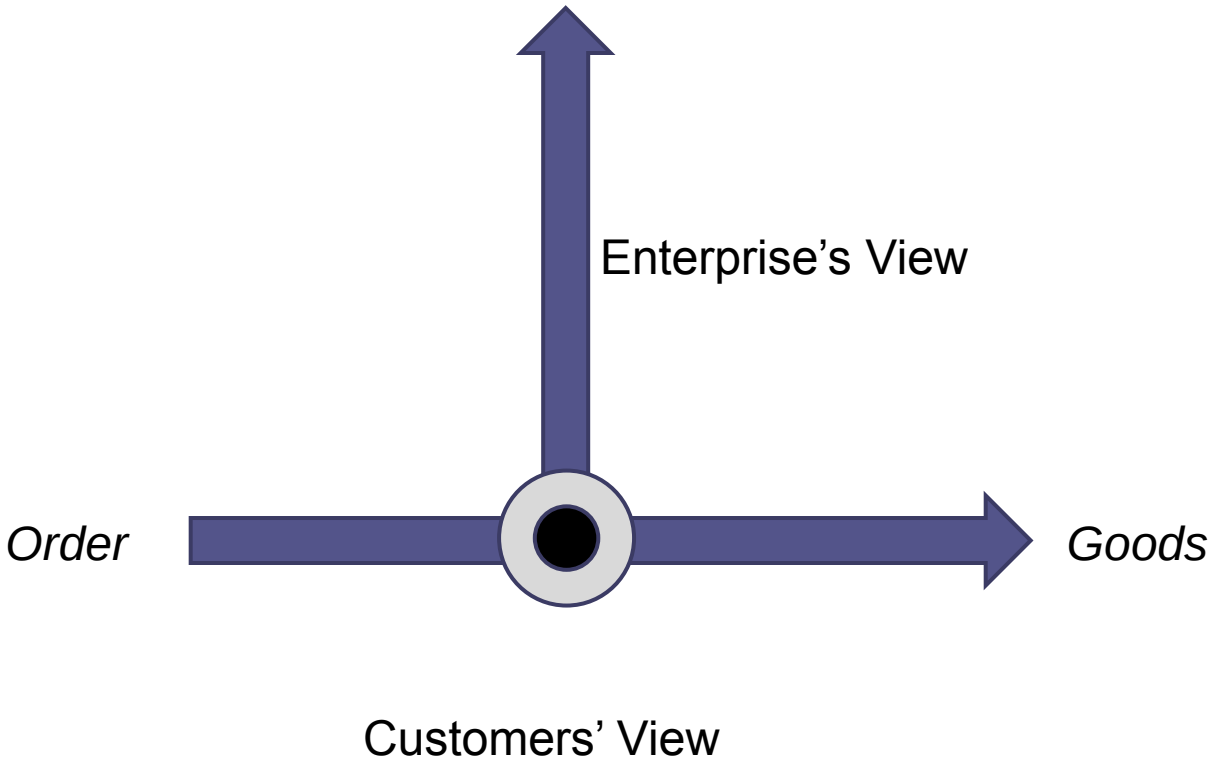
More and more complex processes…



Heterogeneous Goods



2D Aspect of Traceability Problem

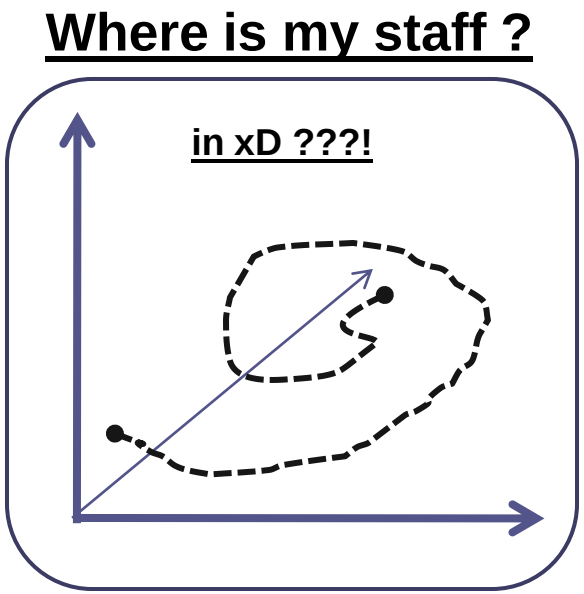


Enterprise Axis

How well my operations perform ?

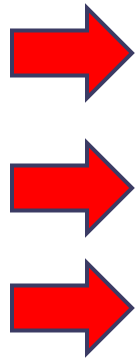
Harvard school (M. Porter et all.): generalization concepts of strategic management

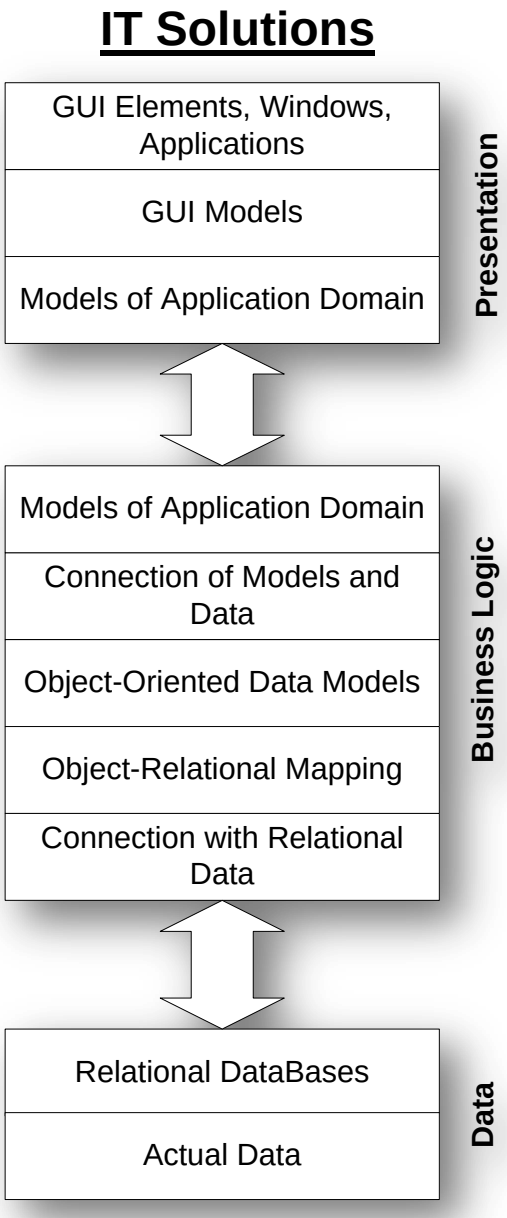
Stakeholders AxEs



Critical Problems

- Distributed essence
- Heterogeneity
- Dynamics
- Multiple stakeholders





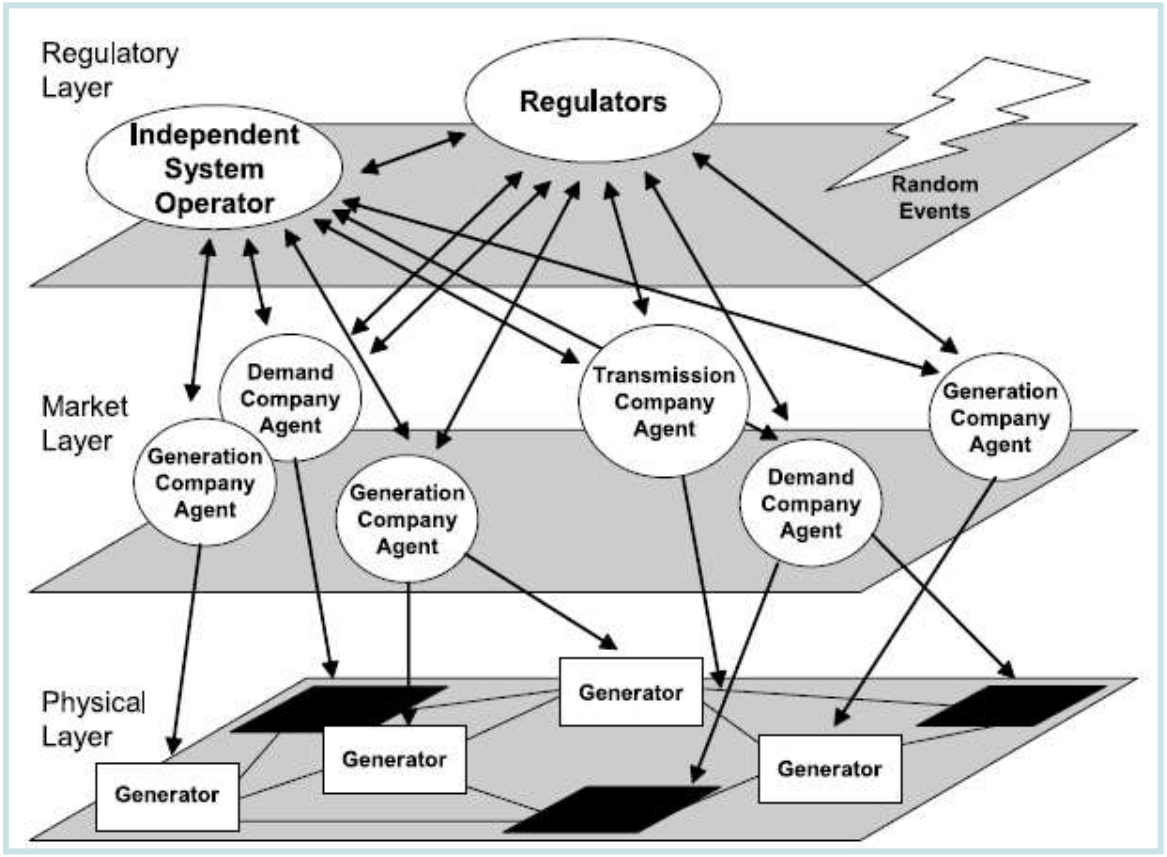
A paradigm of multi-agent systems(MAS)

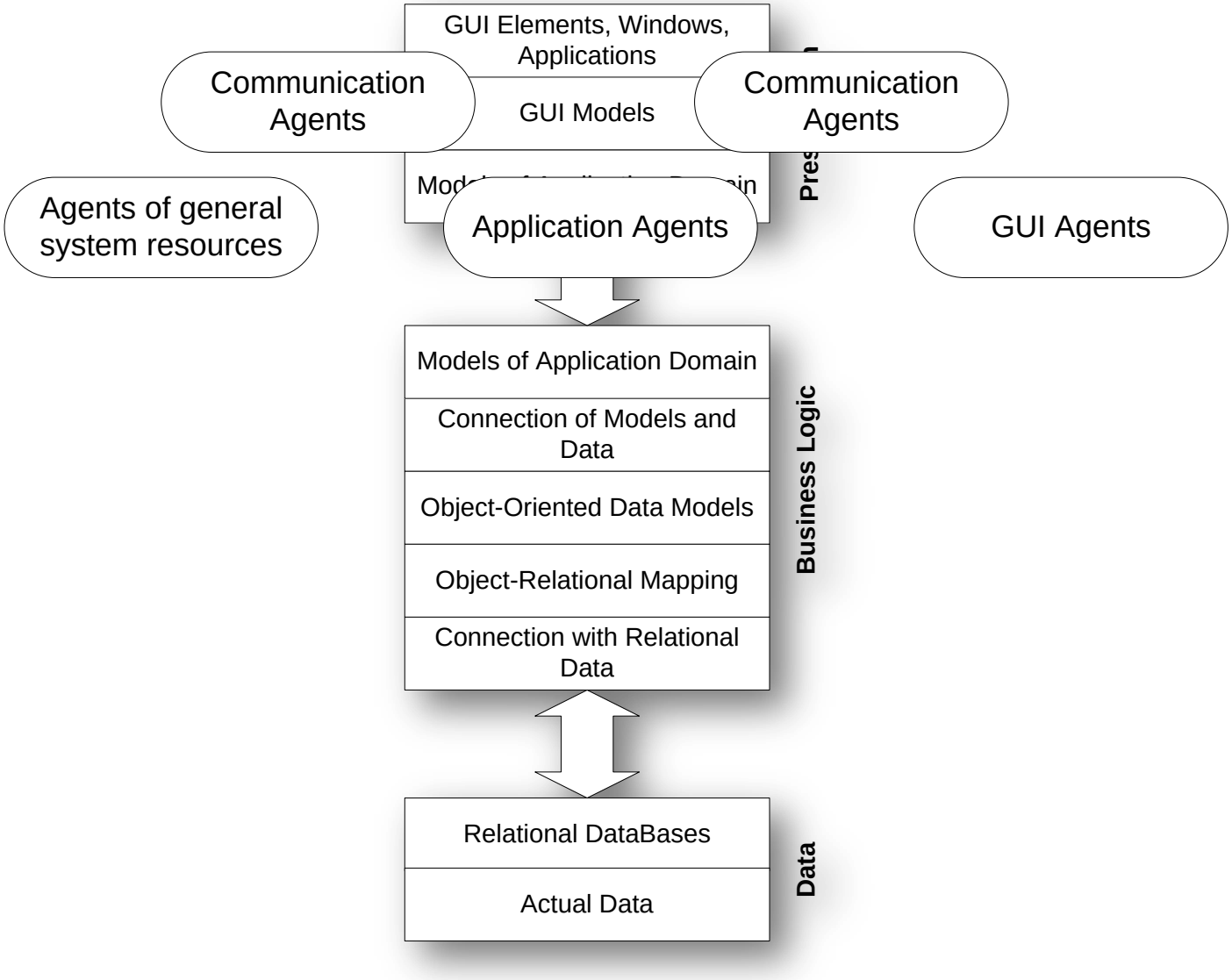
Wooldridge and Jennings (1995):

Agent is a software or hardware system with the following characteristics:

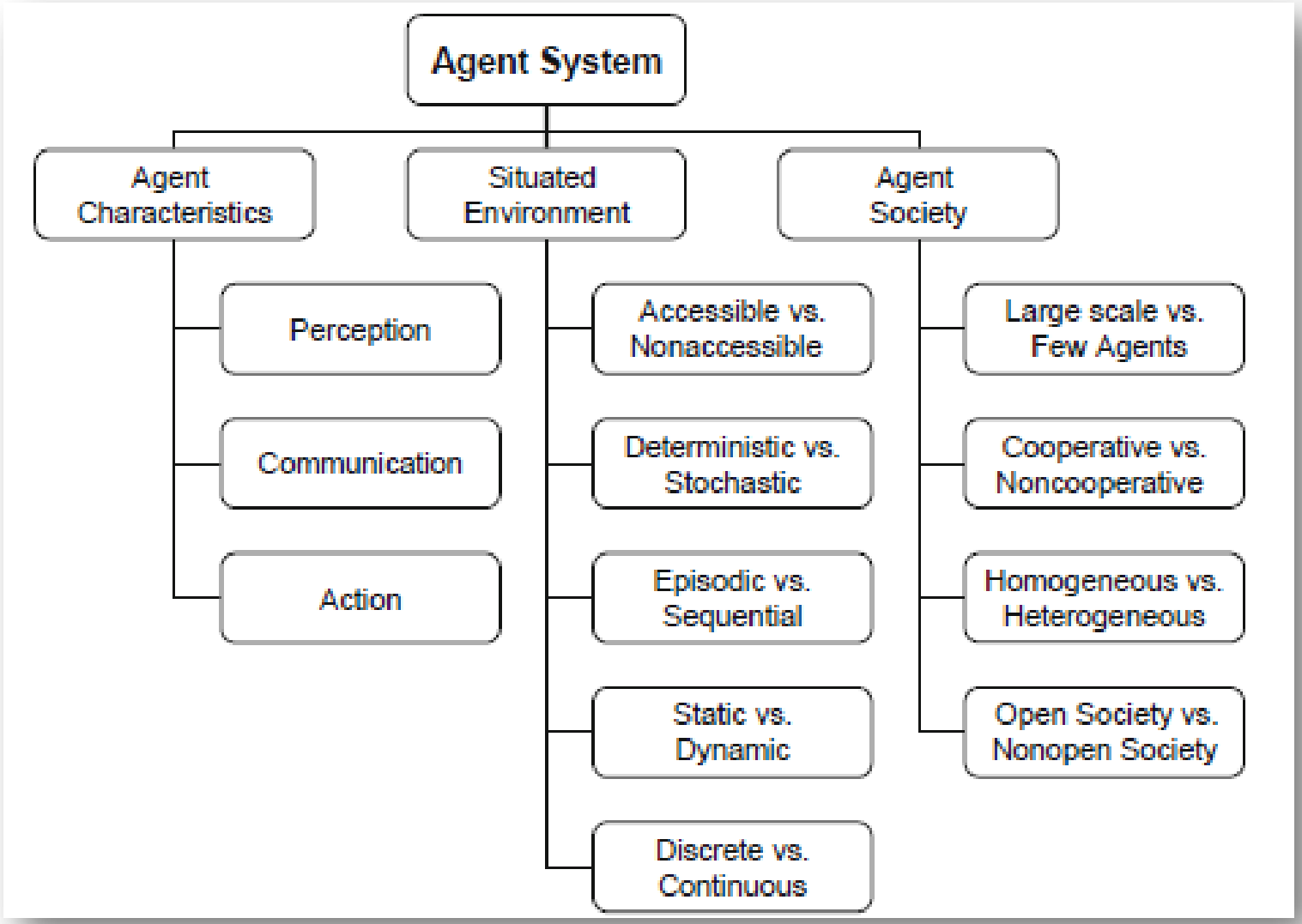
- *Autonomy*
- *Social ability*
- *Responsiveness*
- *Proactiveness*

Multi-Agent System (MAS) is a dynamic, multi-level system where complex social interactions are modeled by formal A2A communication protocols.



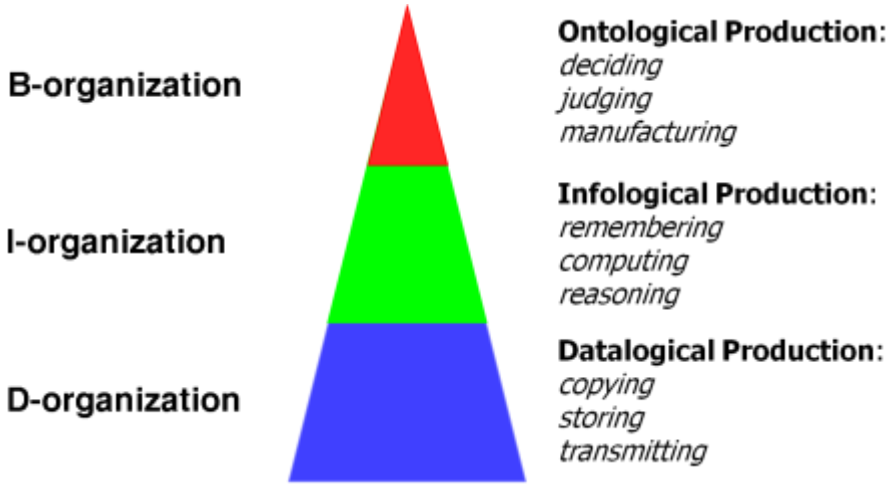

GUI Elements, Windows, Applications
GUI Models
Communication Agents
Communication Agents
Agents of general system resources
Application Agents
GUI Agents
Models of Application Domain
Connection of Models and Data
Object-Oriented Data Models
Object-Relational Mapping
Connection with Relational Data
Business Logic
Relational DataBases
Actual Data
Data

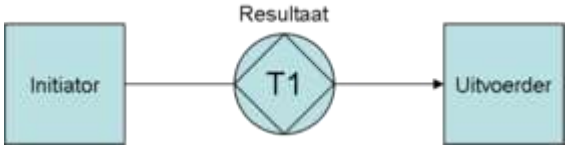
Further Specialization is Needed



Business-Process Control in terms of DEMO* and Multi-Agent paradigm

- Separation of concerns
- Universal transaction pattern
- Language-action Perspective (J. Habermas):
  - request
  - promise
  - state
  - accept





*** J. Dietz, Enterprise Ontology, Springer, 2006**

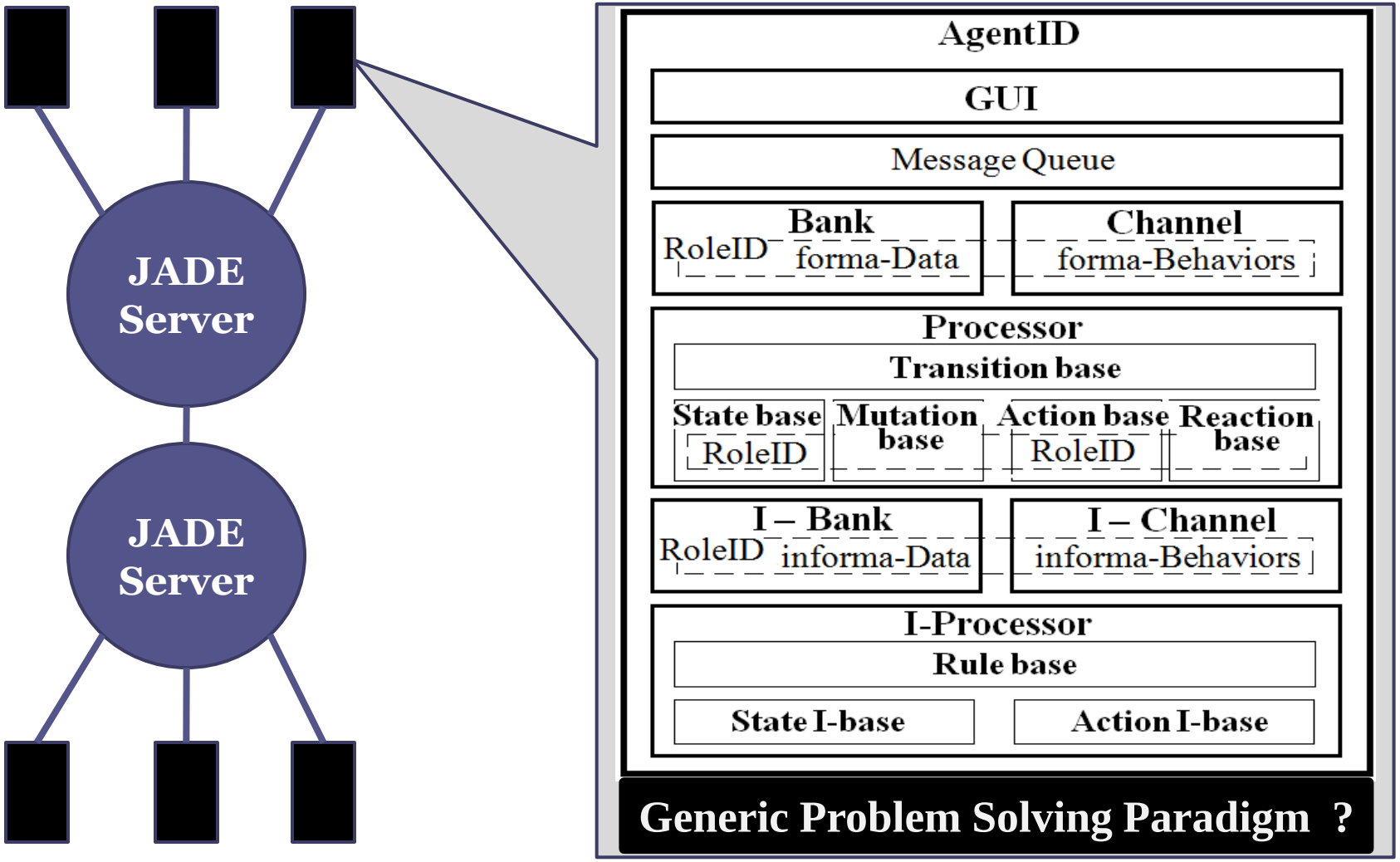
Conformance with Standards

- Proposed Design of programming agents is based on the CRISP meta-model (meta-model for modelling organizations) and DEMO methodology (www.demo.nl)*.
- Design fits industrial MAS standard (FIPA – ww.fipa.org)
- Software Implementation uses FIPA-ACL compliant libraries (JADE –jade.tilab.com)

CRISP Essentials

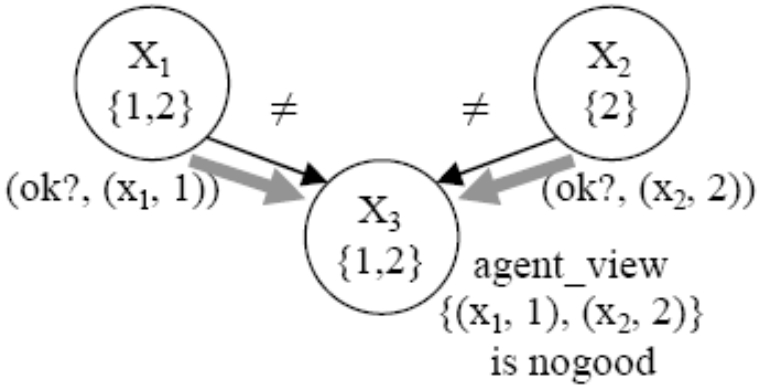
- Organizations operate in a discrete linear time
- State of organization consists of
  - C-facta
  - C-stata
  - P-facta
  - P-stata
- At every moment organization disposes an Agenda (agendum – C-factum to deal with)

Proposed high-level design



DSCP for Problems Solving

- DSCP represents generic framework for distributed cooperative problems solving (multiple DCSP algorithms: (i.e. asynchronous BackTracking (ABT) by M. Yokoo et al.).



- Data Model Integration as an instance of DCSP problems with some new specifics*

- Many application domains affected

* **Hassan Aït-Kaci, Data Models as Constraint Systems**, Constraint Programming Letters 1 (2007) 33–88

Formal DCSP Framework

- CSP P = <X, D, C> :
- X = $\{x_1, \ldots, x_n\}$
- D = $\{d_1, \ldots, d_n\}$
- C = $\{c_1, \ldots c_p\}$, $c_l(x_i, \ldots, x_k) : d_i(x_i) \times \ldots \times d_k(x_k) \rightarrow \{0, 1\}$.

- The main task is to find a value of each variable x_i from the permitted domain d_i in such a way that all constraints are satisfied:
- $(\forall i)$, $v_i \in d_i$ and $(\forall c_l(x_i, \ldots, x_k) \in \text{C})\ c_l(x_i, \ldots, x_k) = 1$.

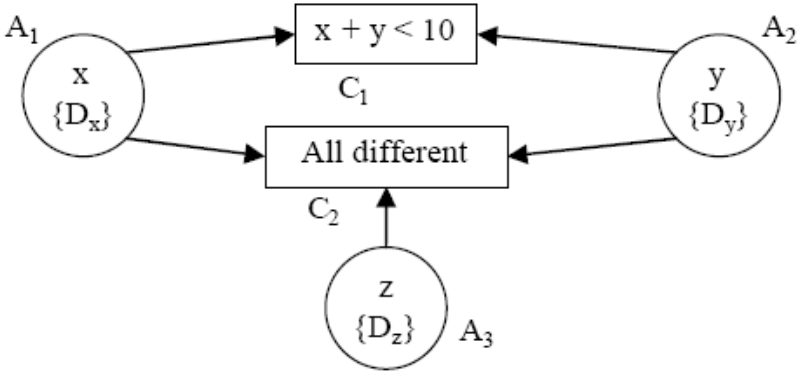
- DCSP additionally determines distribution of X,D,C among the agents

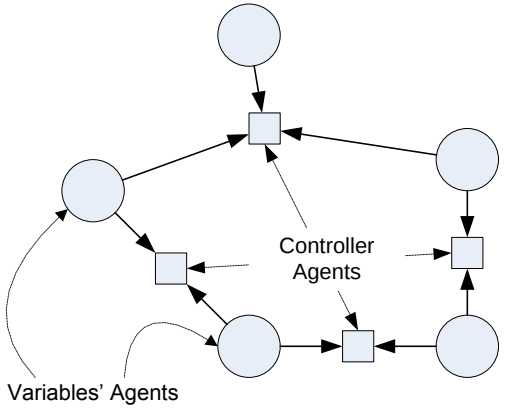
Existing DCSP Research Problems

- Common properties and challenges exist:
  - complex and dynamic structure
  - multiple types of constraints
  - privacy considerations
  - inefficiency of centralized control
- General distributed methods and software frameworks are needed

Our Proposals - Controller-Agents for Constraints Solving (CACS)

- A modification of DCSP
- Key principles:
  - fusion of MAS and traditional CSP
  - 2 agent categories are distinguished:
    1. Variable Agent (VAgent) – owns local variables
    2. Controller Agent (CAgent) – manages inter-agent constraints

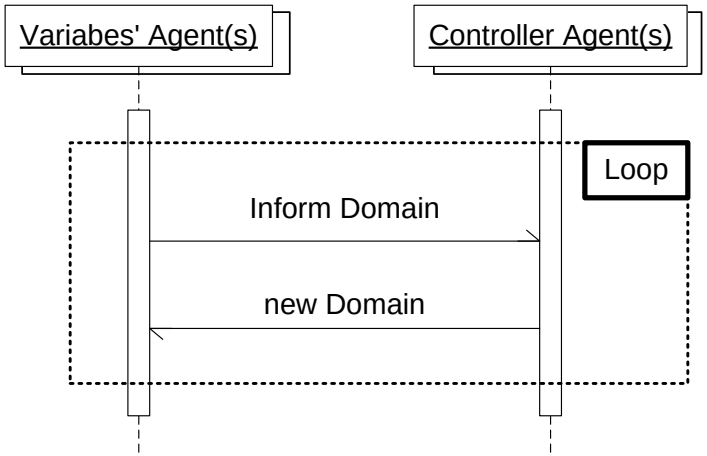




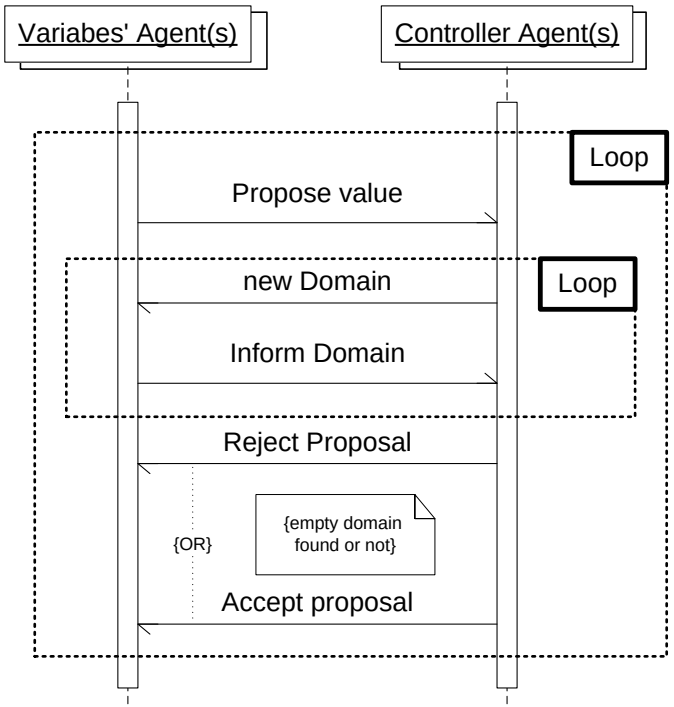
  - specific interaction protocol is proposed

CACS: protocol of interaction

- Interaction protocol with 2 stages:
  1. domain reducing stage
  2. value proposing and validating stage

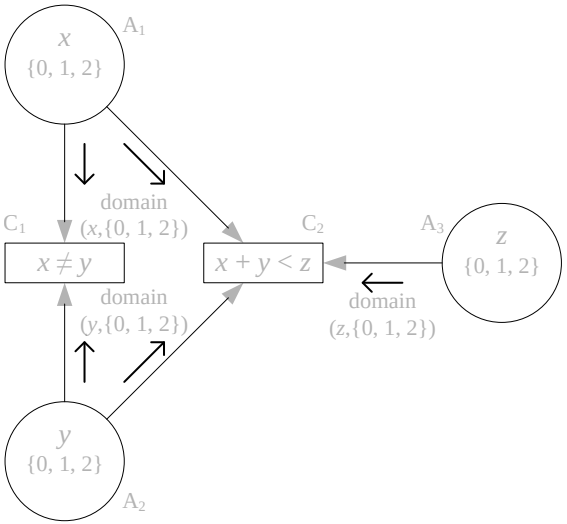


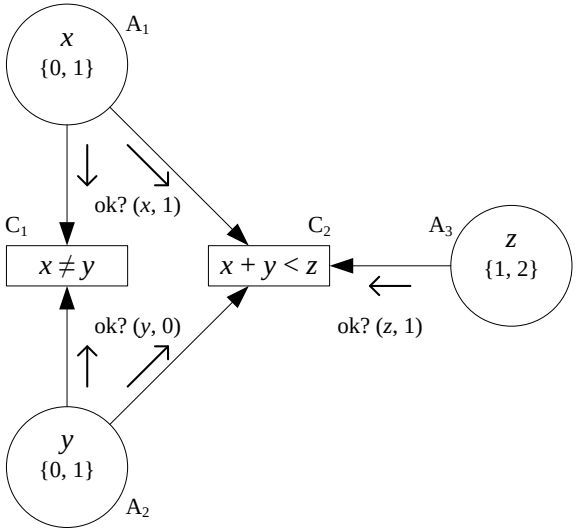
Stage 1

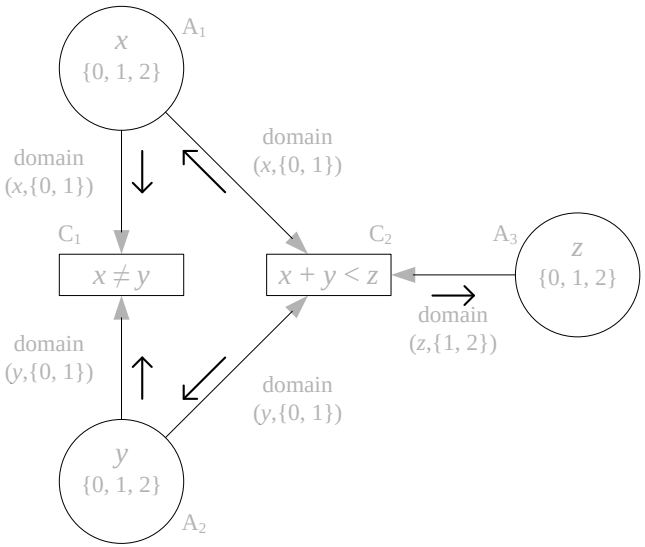


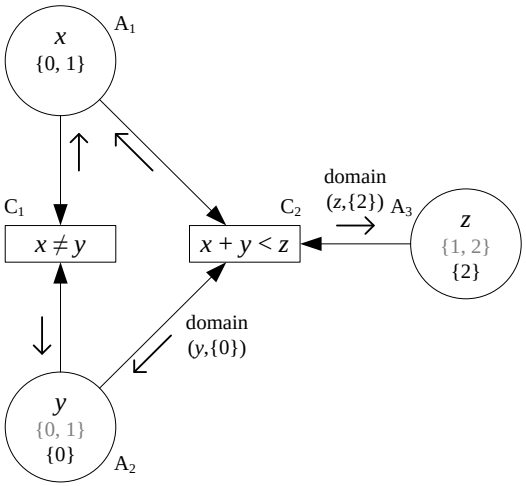
Stage 2

Example of interaction



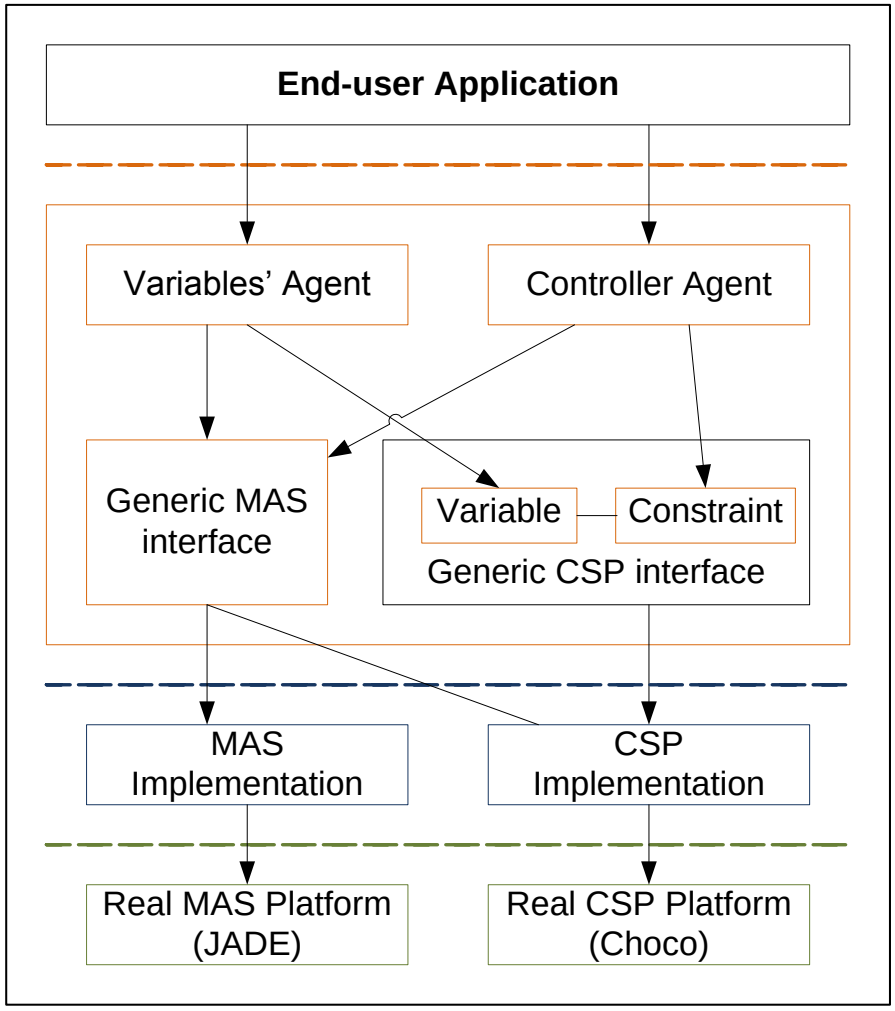




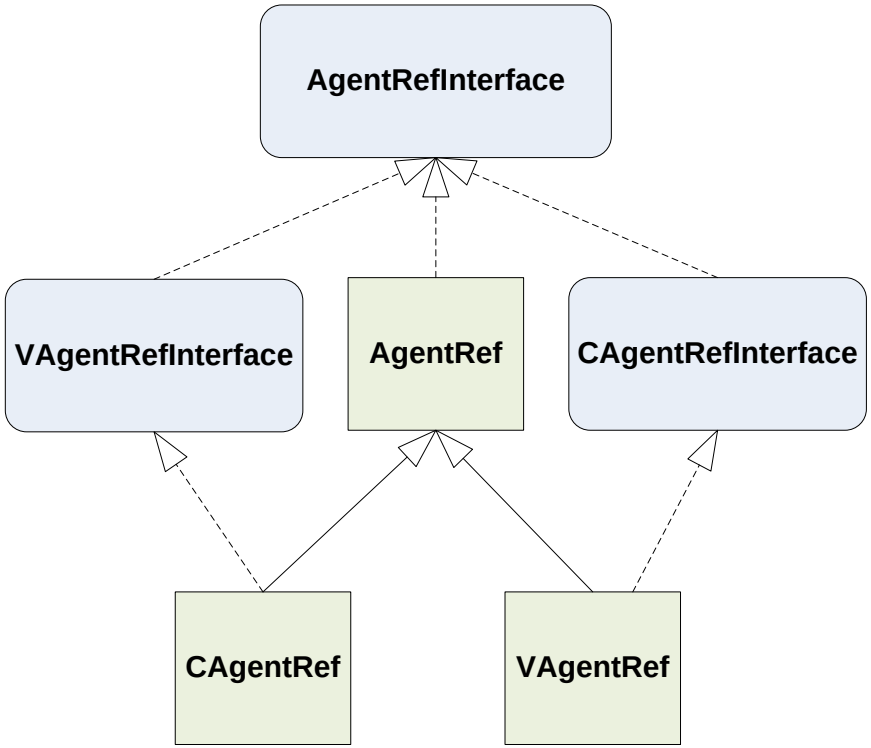


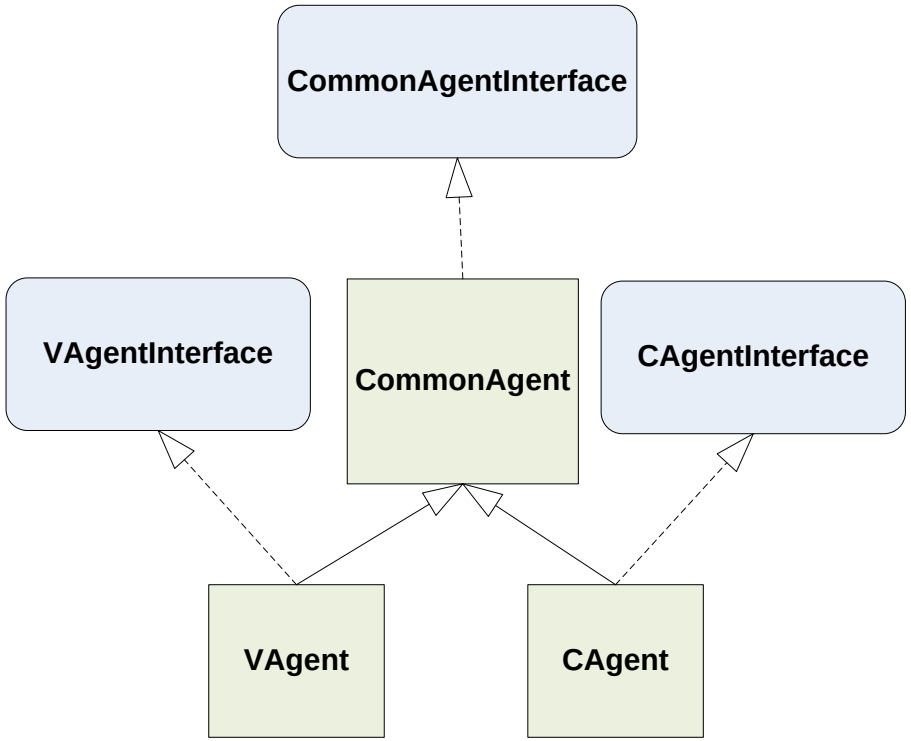
CACS: Software prototype

- Layered software architecture
- Java –based implementation
- Integration of 3d parties solutions:
  - JADE – open-source MAS platform
  - Choco – open source constraints solver



Agent-oriented Implementation





Flexibility of the API

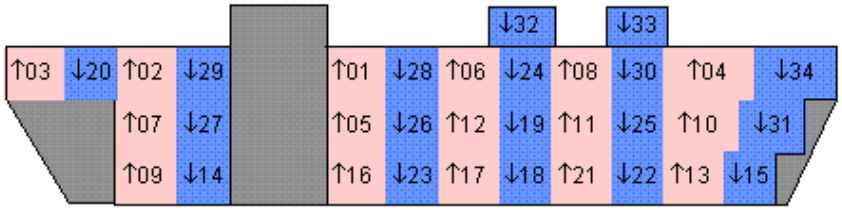
Java API

```
DCSP p = new DCSP("example");

VAgentRef v1 = p.makeVAgent ("v1");
VAgentRef v2 = p.makeVAgent ("v2");
VAgentRef v3 = p.makeVAgent ("v3");
CAgentRef c1 = p.makeCAgent ("c1");
CAgentRef c2 = p.makeCAgent ("c2");
CAgentRef c3 = p.makeCAgent ("c3");

VID x = v1.makeBoundedIntVar ("x", 1, 100);
VID y = v2.makeBoundedIntVar ("y", 1, 100);
VID z = v3.makeBoundedIntVar ("z", 1, 100);

c1.post(new AllDifferent(
                    new VID[]{x,y,z}
                         )
        );
c2.post(new GreaterOrEqual(x, y));
c3.post(new GreaterOrEqual (y, z));

p.solve();
```

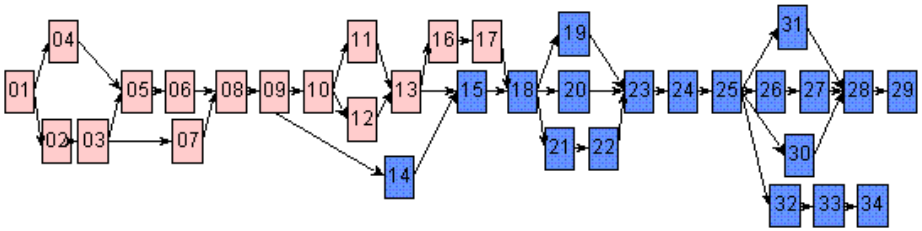
XML-based declarative description of the problem

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE dcsp SYSTEM "dcsp.dtd">
<dcsp>
<name>example</name>
<vagent><name>v1</name>
     <var><name>x</name>
      <inf>1</inf><sup>100</sup>
     </var>
</vagent>
<cagent>
  <name>c1</name>
       <constraint><alldiff>
       <vid><name>x</name>
     <owner>v1</owner></vid>
       <vid><name>y</name>
     <owner>v2</owner></vid>
       <vid><name>z</name>
     <owner>v3</owner></vid>
       </alldiff></constraint>
</cagent>
</dcsp>
```

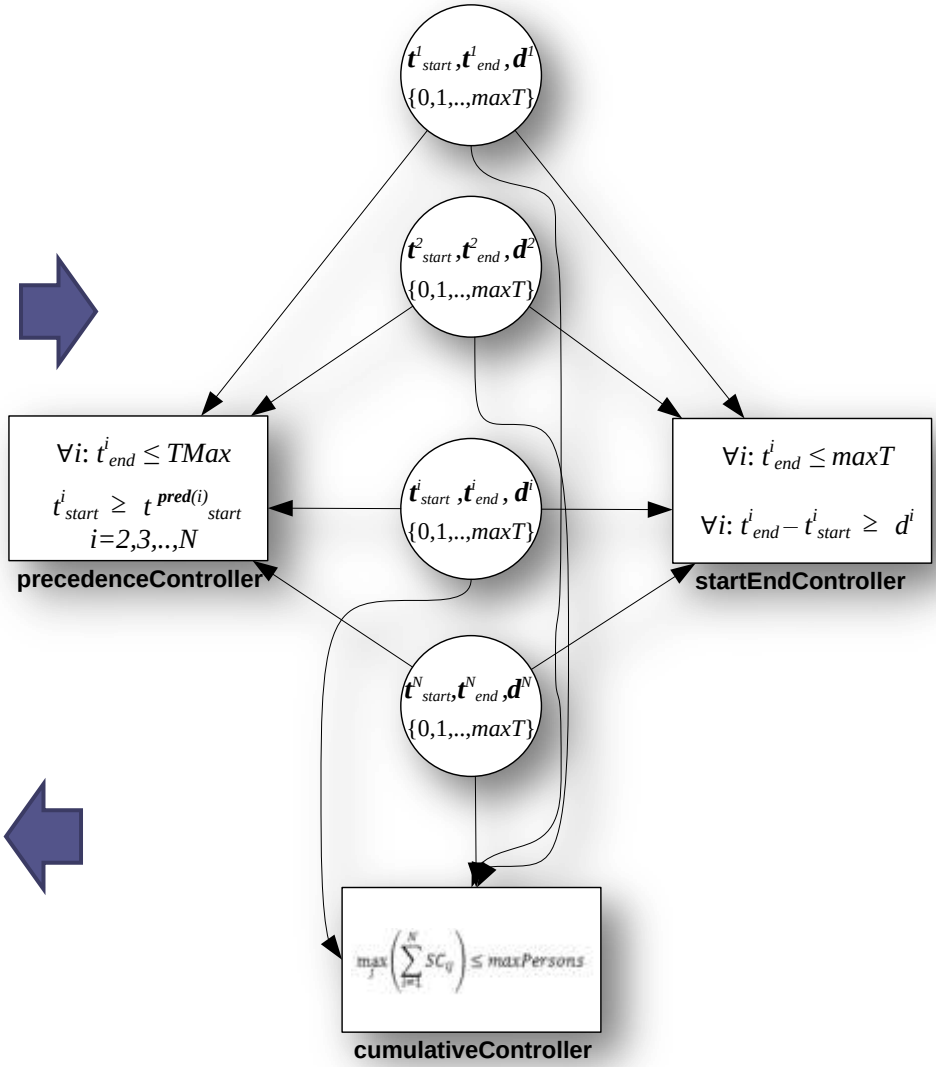
Simple Application Case



Ship model



Loading Sequence



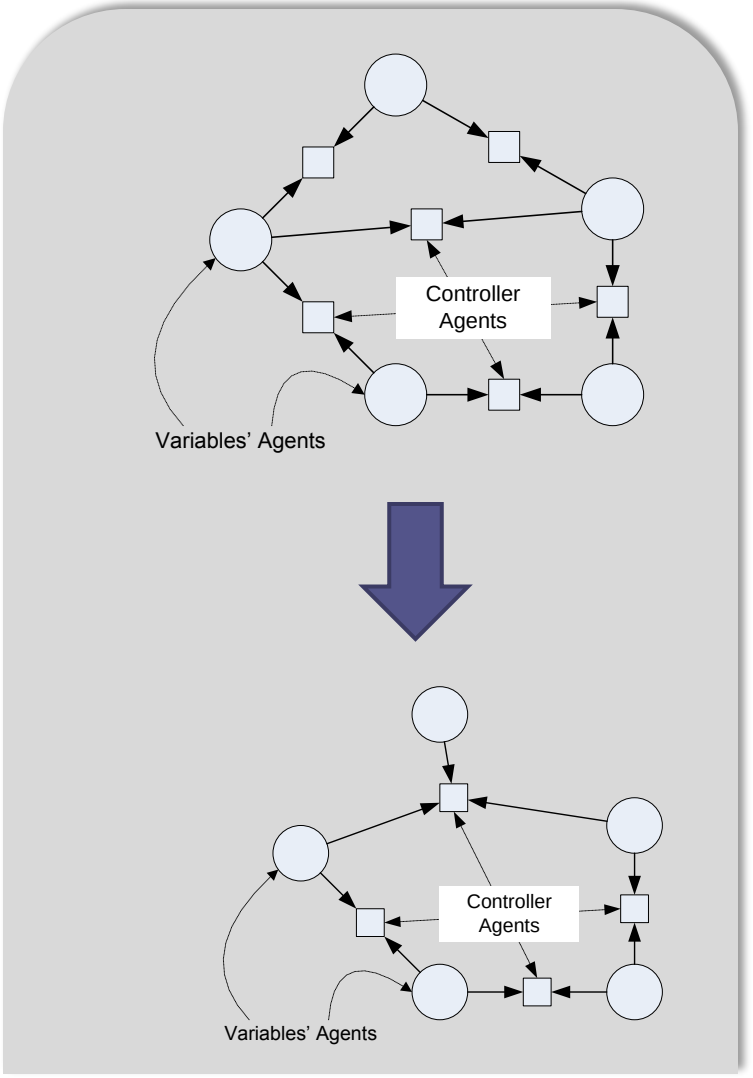
CACS-based MAS

CACS Roadmap

- ✓ Processing non-binary constraints easily and directly
- ✓ Dynamic Fusion/Decomposition of constraints inside a Controller Agent
- ✓ Java-based prototype
- ✓ Integration with CSP Solvers
- ✓ Several large-scale OR problems were solved

- !! Integration with CRISP-Agents

- !! Optimization Add-On to CACS



Application in the context of traceability

- JADE agents collect RFID data
- CRISP Agents perform DEMO transactions
- CRISP Agents determine inconsistencies in D- and I-transactions and their influence on B-transactions (business capabilities, business model, business strategy, stakeholders' interests)
- CACS-Agents try to resolve discovered problems by using DCSP methods

Next Seminar

- Semantic Technologies for Complex Logistic Operations:
  - Relevant Ontologies and Standards
  - Semantic Interoperability Challenge
  - Details of CRISP and DEMO
  - Fusion of DEMO, ISO 15926 and BORO
  - Relevant Software Technologies

Thank you for your attention