Sonar/Oredi

A Tool for Creation and Deployment of Organisation Models

Michael Köhler-Bußmeier   Endri Deliu

Department for Informatics
University of Hamburg

MATES'10

Agent vs. Organisation

Agents

- local autonomy
- inteligence:
  goal-orientation, learning
- social awareness:
  cooperation, competition

Problems

- bounded rationality:
  sub-optimal solutions
- scaling problem for large agent populations

Organisations

- global coherence
- organisational learning,
  data mining
- teams

Solutions

- social structures:
  roles, protocols, norms
- reduction of complexity:
  interaction networks,
  e.g. hierarchies

Agent vs. Organisation

Agents

- local autonomy
- inteligence:
  goal-orientation, learning
- social awareness:
  cooperation, competition

Organisations

- global coherence
- organisational learning,
  data mining
- teams

Problems

- bounded rationality:
  sub-optimal solutions
- scaling problem for large agent populations

Solutions

- social structures:
  roles, protocols, norms
- reduction of complexity:
  interaction networks,
  e.g. hierarchies

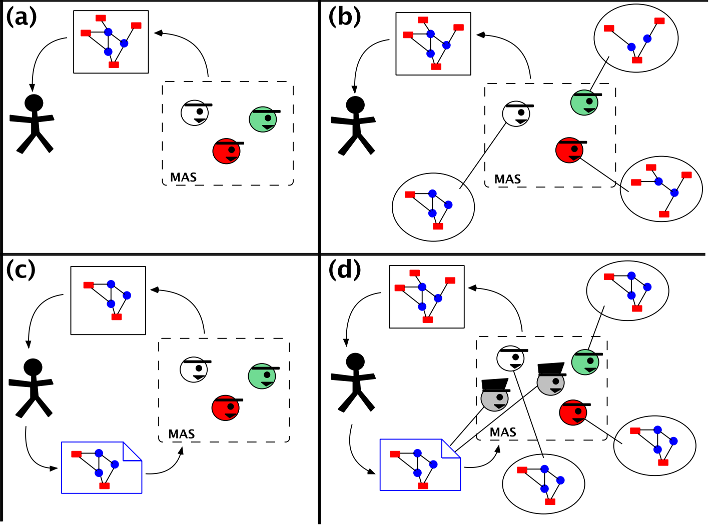
Matrix: Organisation Awareness

agents without org. knowledge

agents with org. knowledge

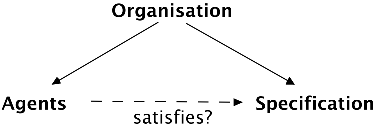
agent−centered perspective
=
no explicit formal organization

organization−centered perspective
=
formalization of organization



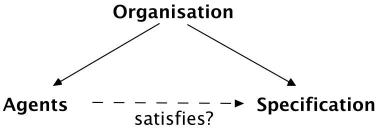
The Formal Organisation as an Artefact

Organisation considered as a Specification:



The Formal Organisation as an Artefact

Organisation considered as a Specification:



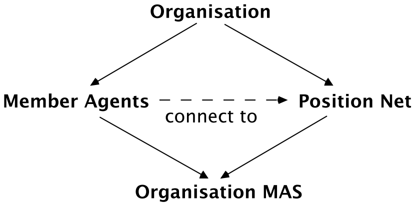
Organisation

Agents

Specification

satisfies?

SONAR: Organisation considered as a Multi-Agent System:



Organisation

Member Agents

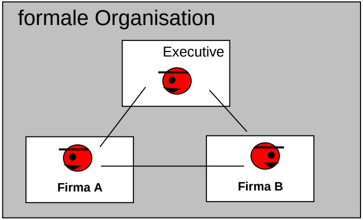
Position Net

connect to

Organisation MAS

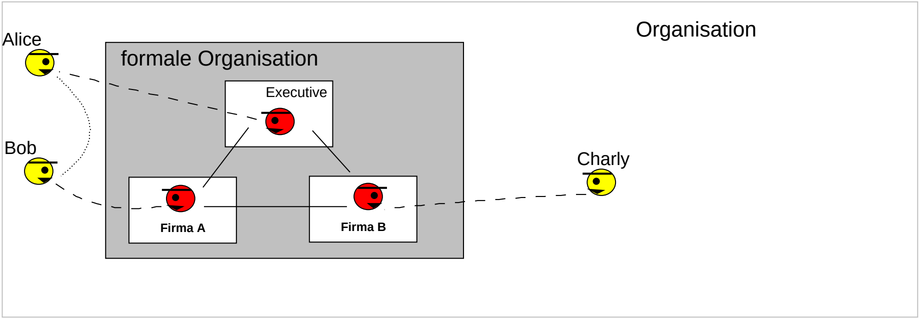
Formal vs. informal Organisation

SONAR: The organisation as a MAS.



Formal vs. informal Organisation

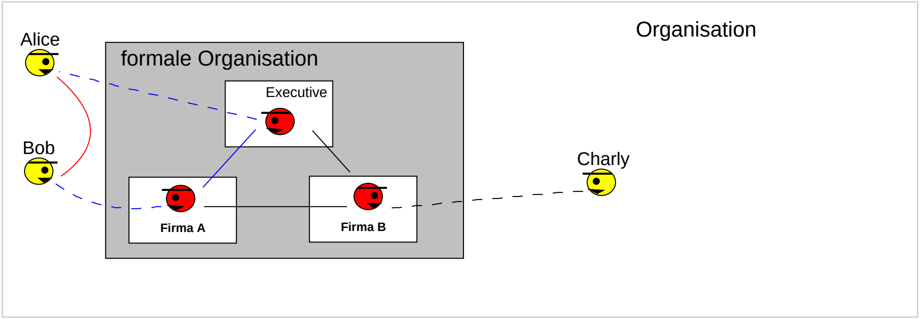
SONAR: The organisation as a MAS.



Membership:
An agent becomes a member of the organisation via binding to a position.

Formal vs. informal Organisation

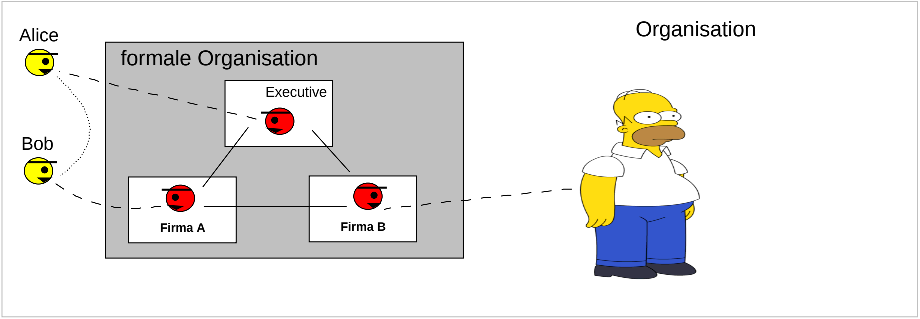
SONAR: The organisation as a MAS.



inofficial vs. official communication

Formal vs. informal Organisation

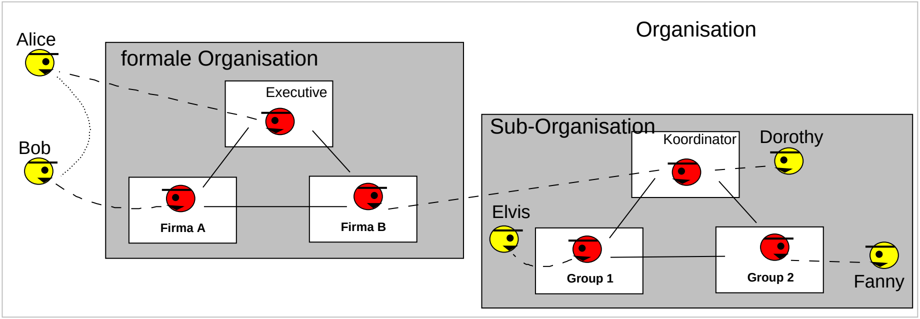
SONAR: The organisation as a MAS.



Hybrid Organisation: Artefacts + Humans

Formal vs. informal Organisation

SONAR: The organisation as a MAS.



Suborganisation, Holons

Outline

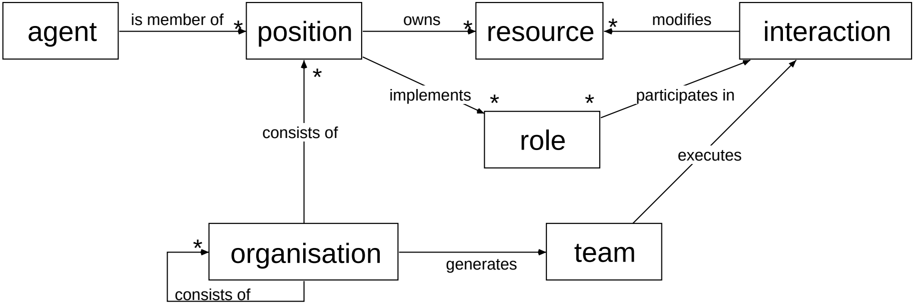
1. Sonar
   - Static Aspects: Position Network, Roles, AIP
   - Dynamic Aspects: Teamwork
   - Theoretical Aspects
2. Mulan4Sonar
   - Deployment of SONAR-Models
   - The Organisation Agent
   - The Position Agent
3. The Sonar/Oredi Compiler
   - Middleware: Design Objectives
   - The Mulan4Sonar Prototype
   - Compilation as Graph Transformation
4. Conclusion

Outline

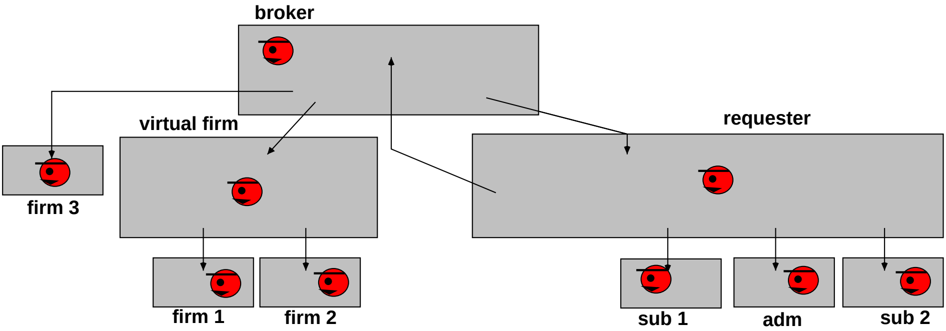
1. Sonar
   - Static Aspects: Position Network, Roles, AIP
   - Dynamic Aspects: Teamwork
   - Theoretical Aspects
2. Mulan4Sonar
   - Deployment of SONAR-Models
   - The Organisation Agent
   - The Position Agent
3. The Sonar/Oredi Compiler
   - Middleware: Design Objectives
   - The Mulan4Sonar Prototype
   - Compilation as Graph Transformation
4. Conclusion

Overview of Sonar Concepts

Basic SONAR concepts:

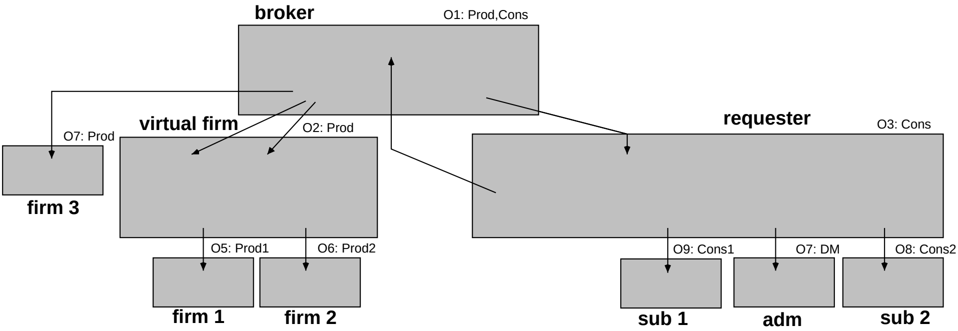


Static Aspects: Position Network, Roles, AIP



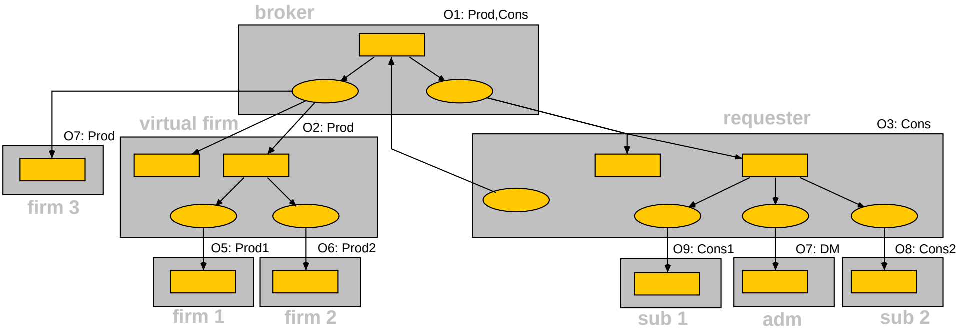
Agent Network

Static Aspects: Position Network, Roles, AIP



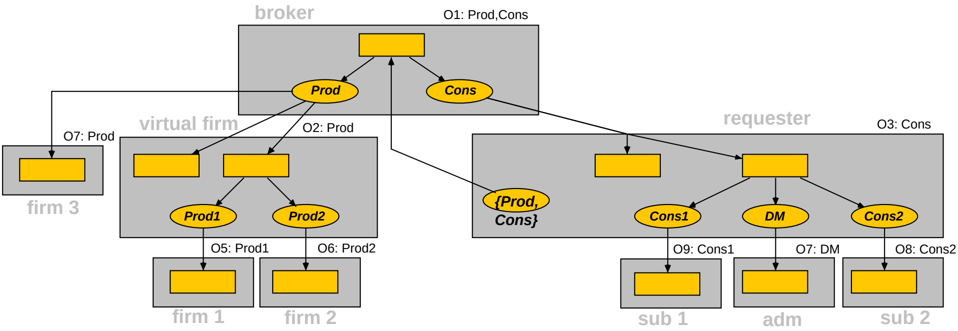
Interface Description

Static Aspects: Position Network, Roles, AIP



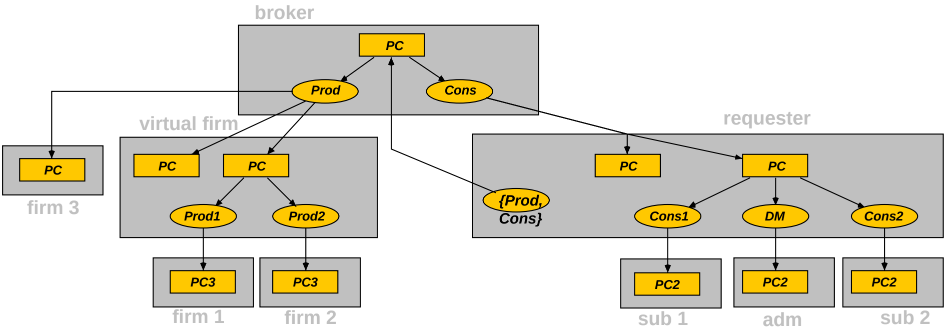
Petri Net
→ place = (sub)-task
→ transition = delegation and/or refinement

Static Aspects: Position Network, Roles, AIP



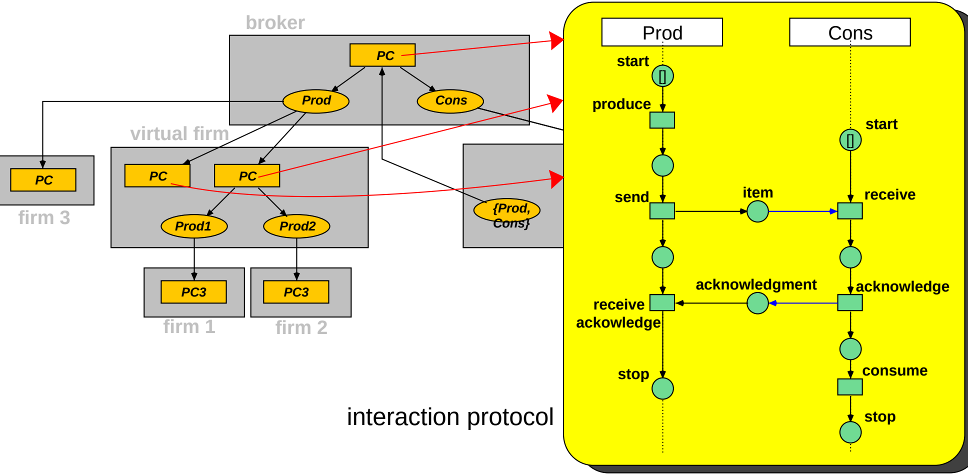
Roles

Static Aspects: Position Network, Roles, AIP



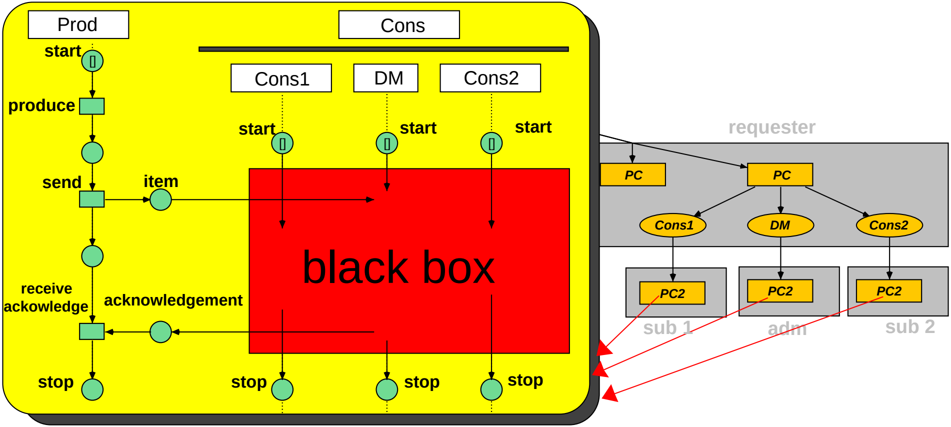
Interaction Protocols

Static Aspects: Position Network, Roles, AIP

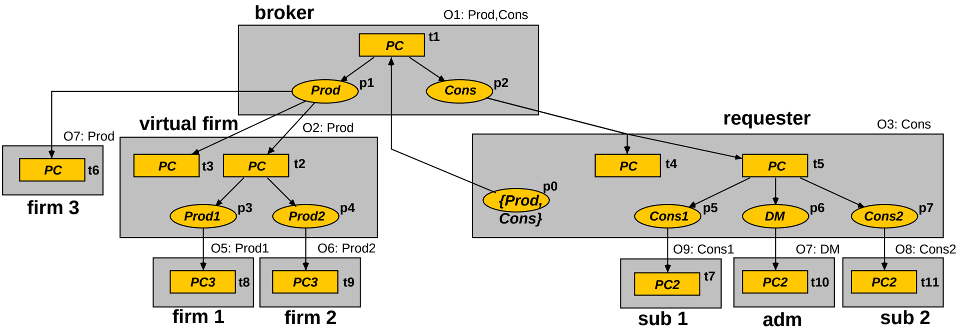


interaction protocol

Static Aspects: Position Network, Roles, AIP



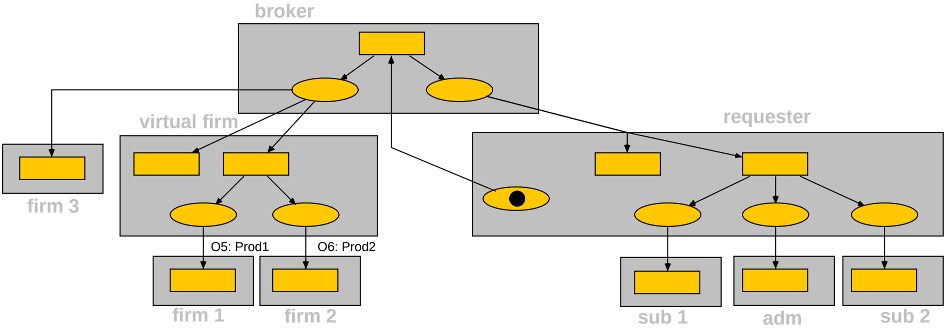
Static Aspects: Position Network, Roles, AIP



This is the static model structure.
What about the Petri Net dynamics?

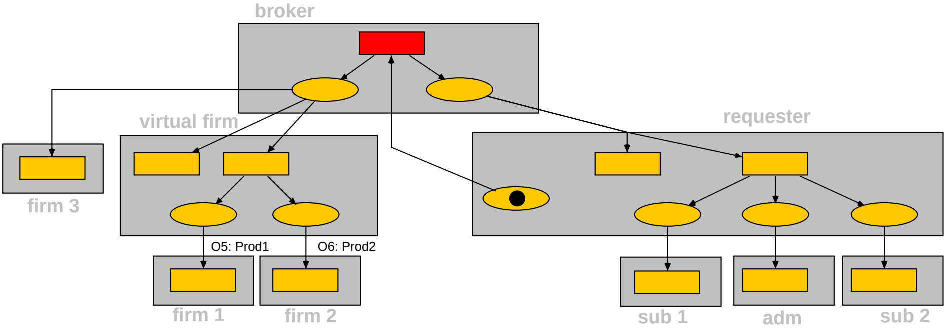
Dynamic Aspects: Teamformation

Petri Net Token Game = Team Formation



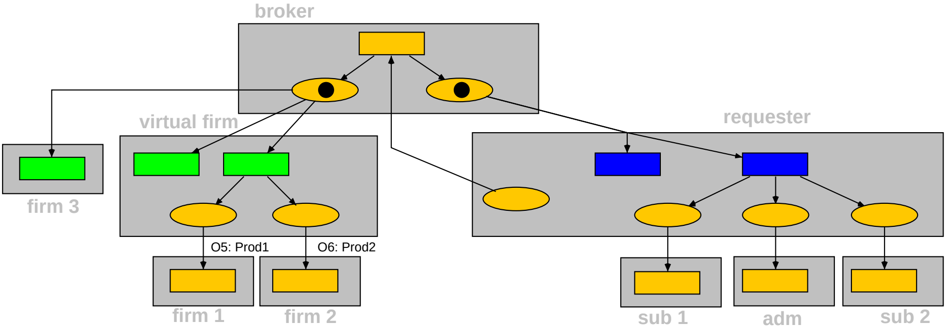
Dynamic Aspects: Teamformation

Petri Net Token Game = Team Formation



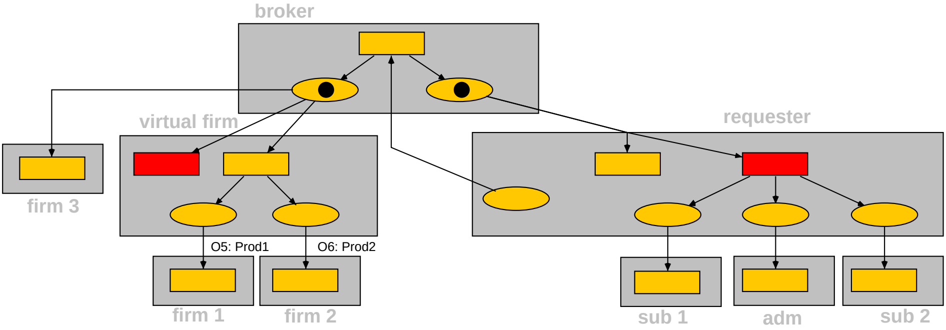
Dynamic Aspects: Teamformation

Petri Net Token Game = Team Formation



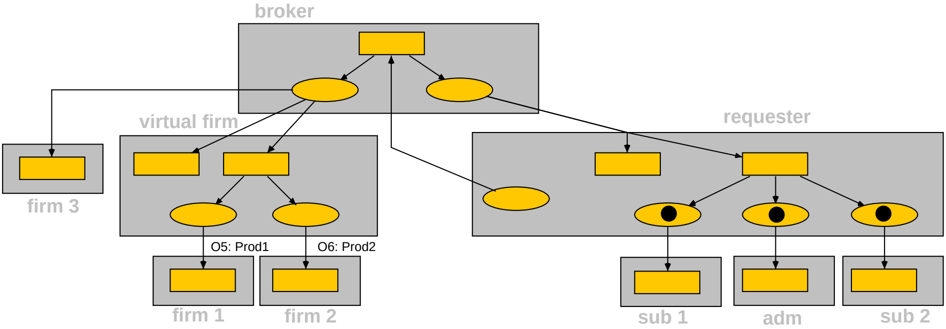
Dynamic Aspects: Teamformation

Petri Net Token Game = Team Formation



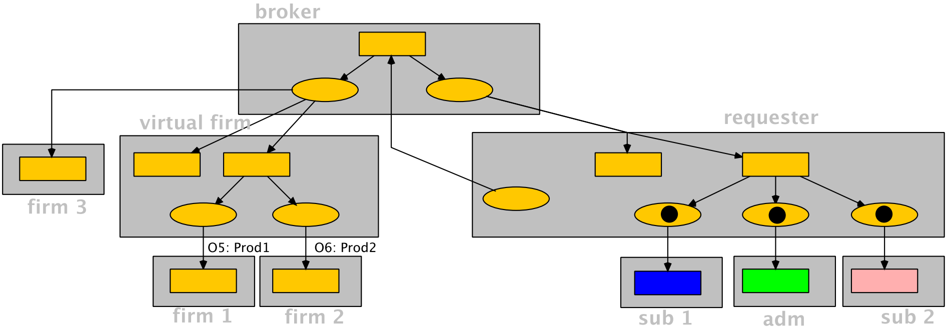
Dynamic Aspects: Teamformation

Petri Net Token Game = Team Formation



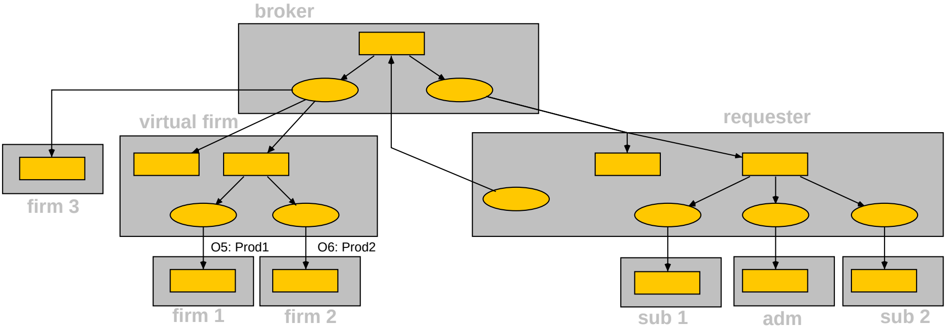
Dynamic Aspects: Teamformation

Petri Net Token Game = Team Formation



Dynamic Aspects: Teamformation

Petri Net Token Game = Team Formation



Exploiting Petri Net Theory

Definition

An organisation net N is **sound** if each initial task can be performed via delegation.

This can easily expressed as a Petri net property:

Definition

1. A marking m is **processible** if the net can be emptied: $\mathbf{0} \in RS(m)$.
2. The net N is **processible** if **all** markings m are.

Problem: To prove processibility for N inifinitely many markings m have to be considered.

Fortunately, this property can be expressed in terms of the net topology:

Theorem

A net N is processible iff
(i) each place is reachable via a path from a initial place and
(ii) for each place there is a path leading to a final transition.

Exploiting Petri Net Theory

Definition

An organisation net N is **sound** if each initial task can be performed via delegation.

This can easily expressed as a Petri net property:

Definition

1. A marking m is **processible** if the net can be emptied: $\mathbf{0} \in RS(m)$.
2. The net N is **processible** if **all** markings m are.

Problem: To prove processibility for N inifinitely many markings m have to be considered.

Fortunately, this property can be expressed in terms of the net topology:

Theorem

A net N is processible iff
(i) each place is reachable via a path from a initial place and
(ii) for each place there is a path leading to a final transition.

Exploiting Petri Net Theory

Definition

An organisation net N is **sound** if each initial task can be performed via delegation.

This can easily expressed as a Petri net property:

Definition

1. A marking m is **processible** if the net can be emptied: $\mathbf{0} \in RS(m)$.
2. The net N is **processible** if **all** markings m are.

Problem: To prove processibility for N inifinitely many markings m have to be considered.

Fortunately, this property can be expressed in terms of the net topology:

Theorem

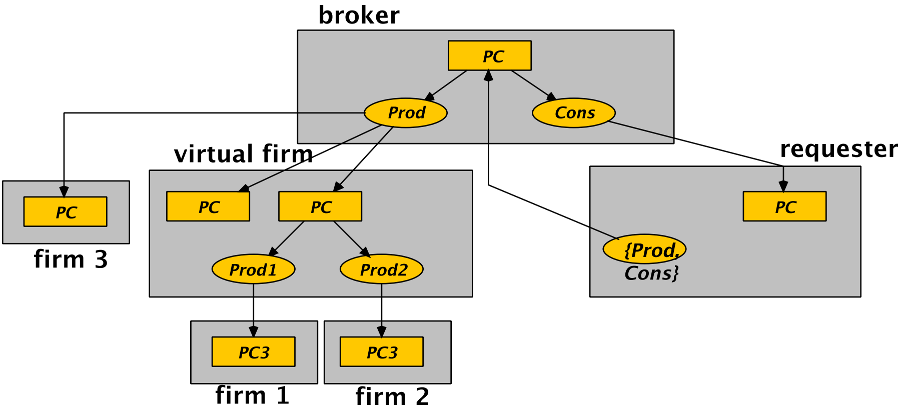
A net N is processible iff
(i) each place is reachable via a path from a initial place and
(ii) for each place there is a path leading to a final transition.

Outline

1. Sonar
   - Static Aspects: Position Network, Roles, AIP
   - Dynamic Aspects: Teamwork
   - Theoretical Aspects
2. Mulan4Sonar
   - Deployment of SONAR-Models
   - The Organisation Agent
   - The Position Agent
3. The Sonar/Oredi Compiler
   - Middleware: Design Objectives
   - The Mulan4Sonar Prototype
   - Compilation as Graph Transformation
4. Conclusion

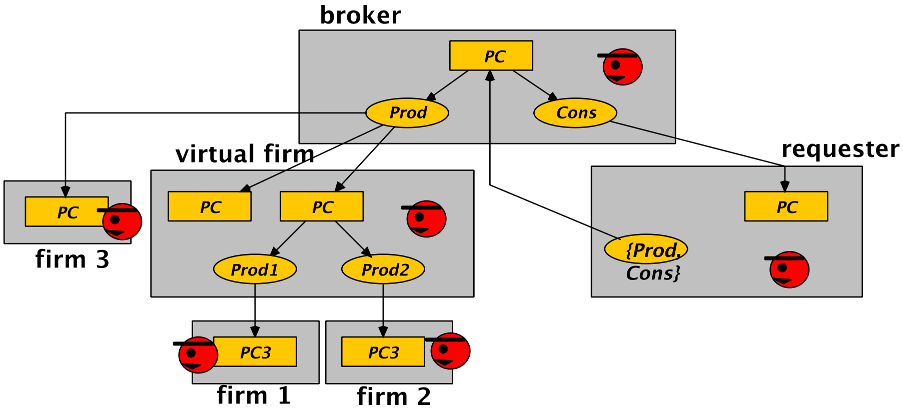
The Mulan4Sonar Middleware

Represent a SONAR-Model as a MAS:



The Mulan4Sonar Middleware

Represent a SONAR-Model as a MAS:

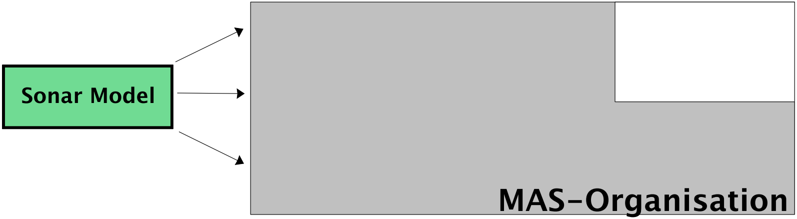


Deployment of SONAR-Models

SONAR-Modell ↦ MAS

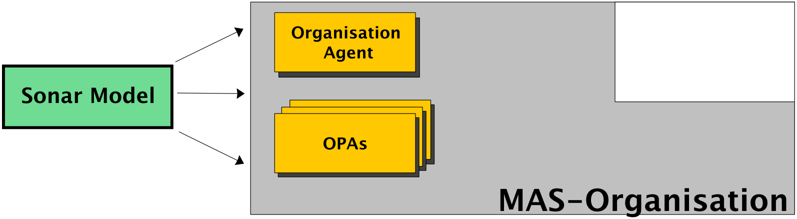
Deployment of SONAR-Models

SONAR-Modell ↦ MAS



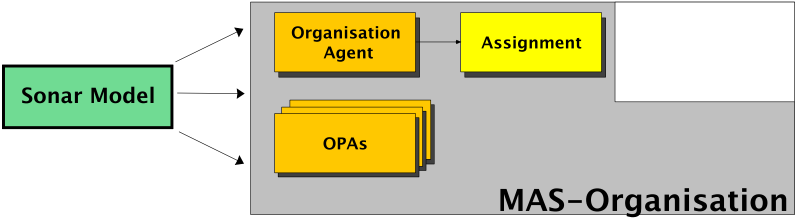
Deployment of SONAR-Models

SONAR-Modell ↦ MAS



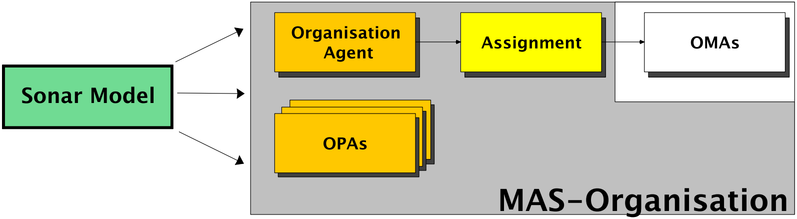
Deployment of SONAR-Models

SONAR-Modell ↦ MAS



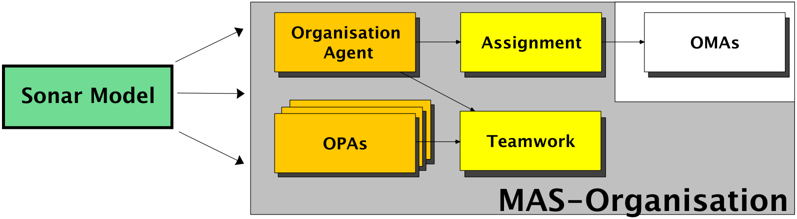
Deployment of SONAR-Models

SONAR-Modell ↦ MAS



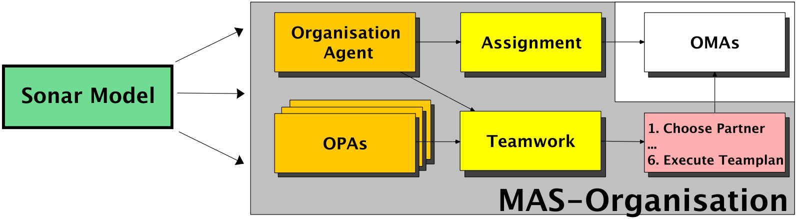
Deployment of SONAR-Models

SONAR-Modell ↦ MAS



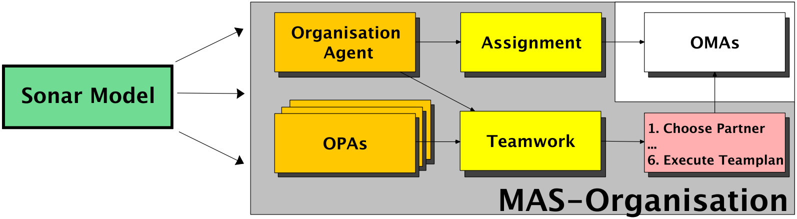
Deployment of SONAR-Models

SONAR-Modell ↦ MAS



Deployment of SONAR-Models

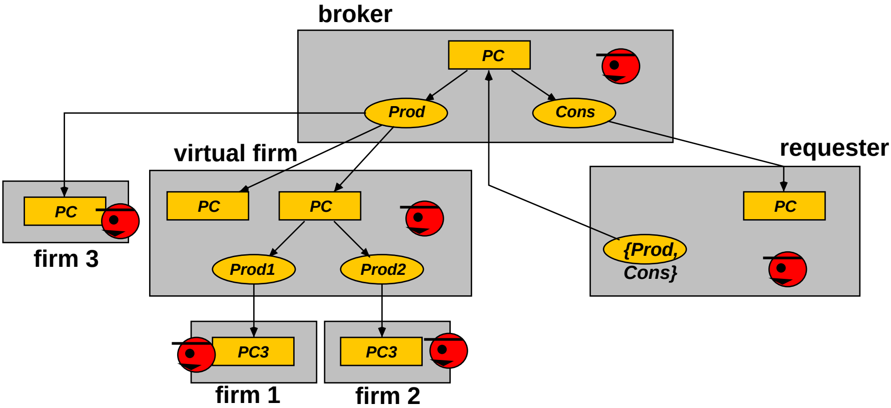
SONAR-Modell ↦ MAS



- Organisation Agent
- Organisation Position Agent (OPA)
- Organisation Member Agent (OMA)

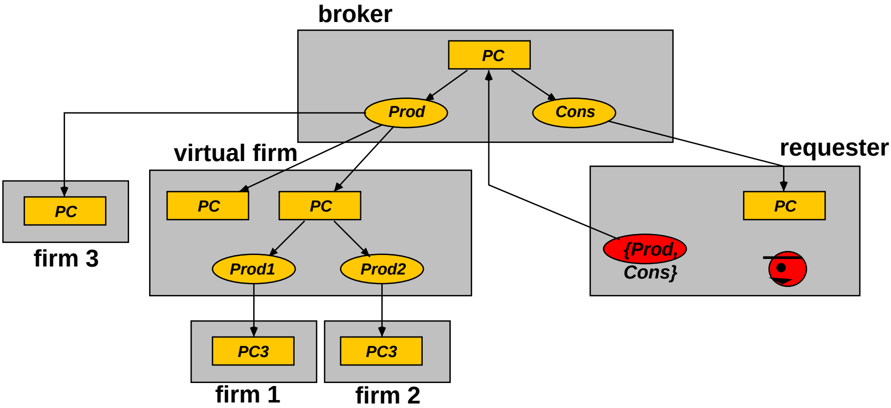
Example: Team Formation

Team Formation is the first of six teamwork phases.



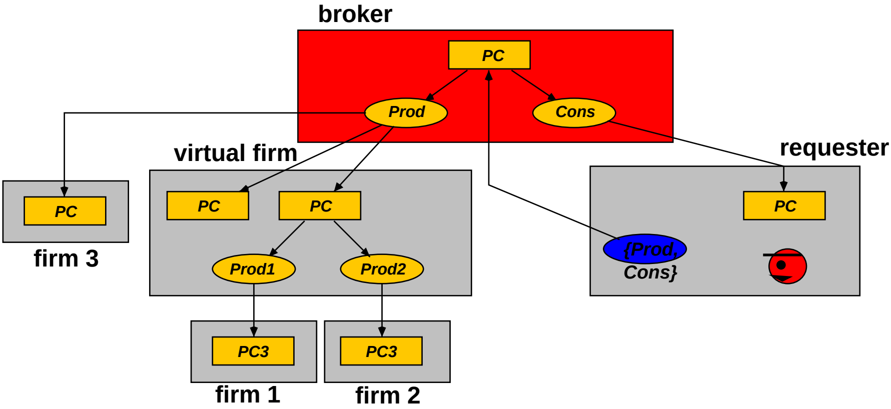
Example: Team Formation

Team Formation is the first of six teamwork phases.



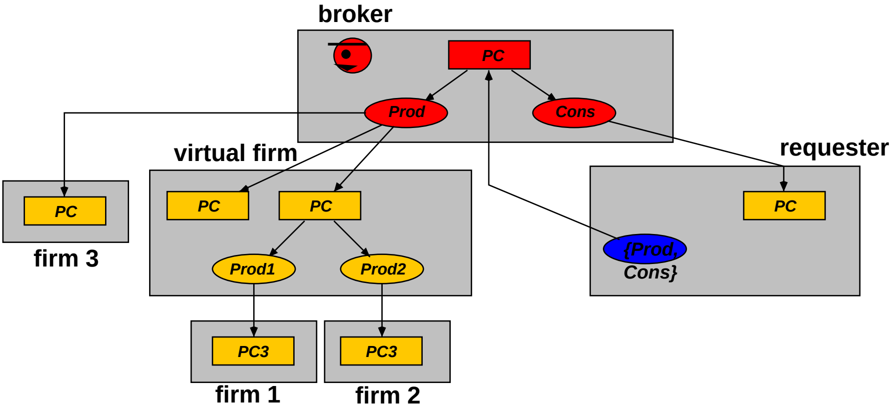
Example: Team Formation

Team Formation is the first of six teamwork phases.



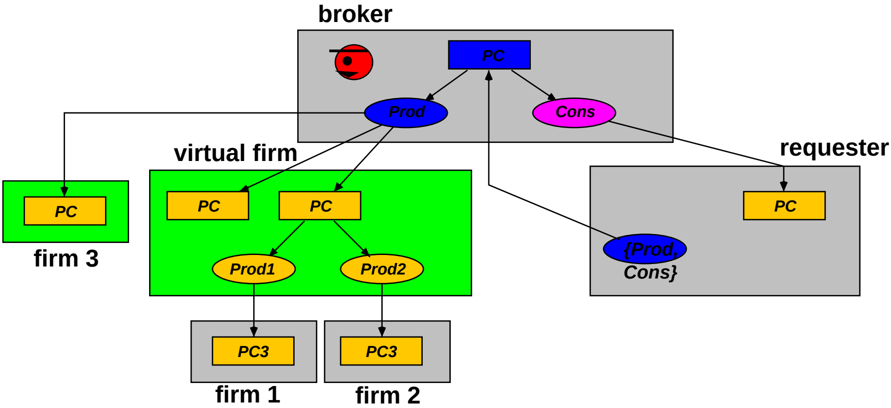
Example: Team Formation

Team Formation is the first of six teamwork phases.



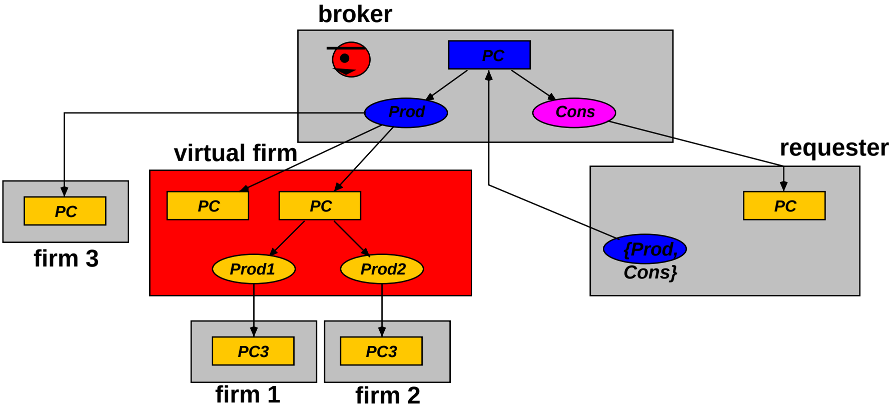
Example: Team Formation

Team Formation is the first of six teamwork phases.



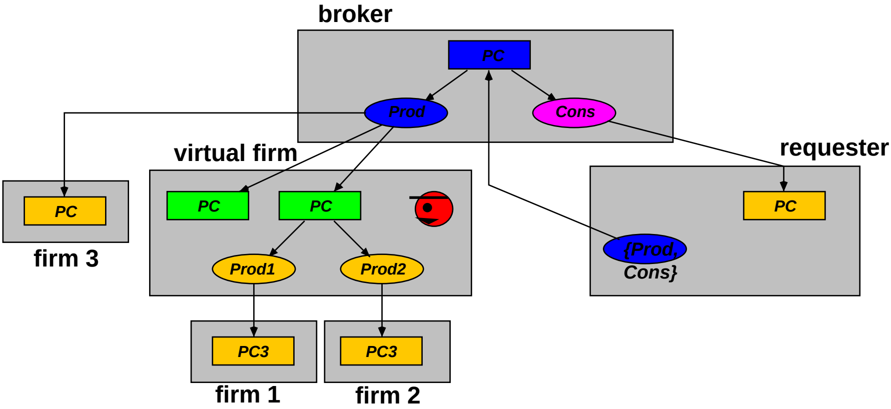
Example: Team Formation

Team Formation is the first of six teamwork phases.



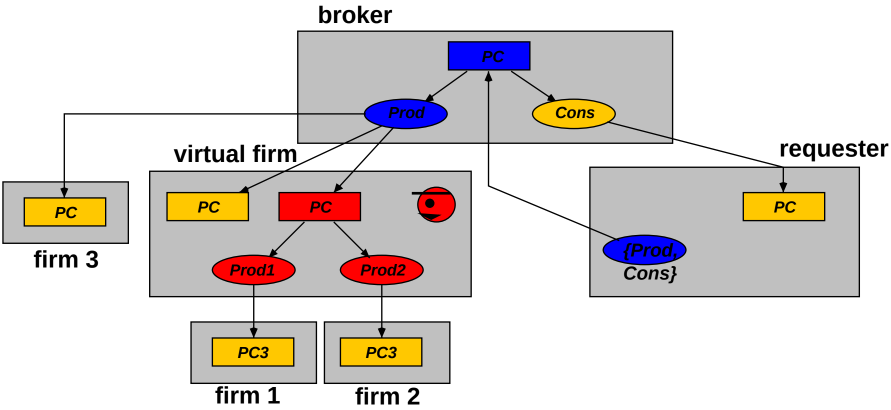
Example: Team Formation

Team Formation is the first of six teamwork phases.



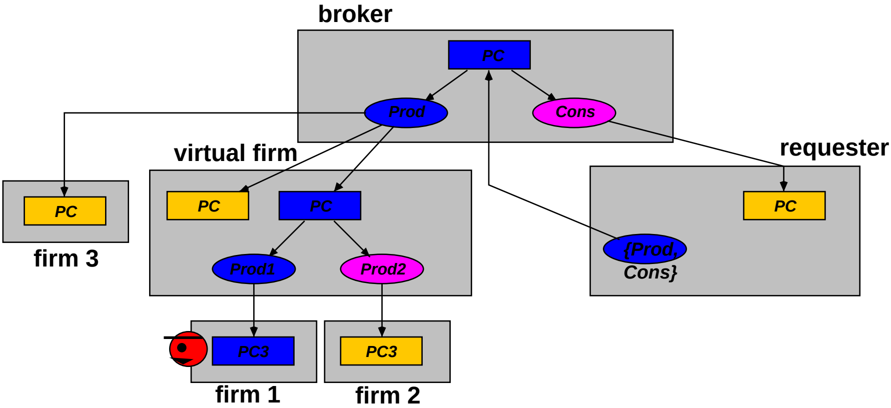
Example: Team Formation

Team Formation is the first of six teamwork phases.



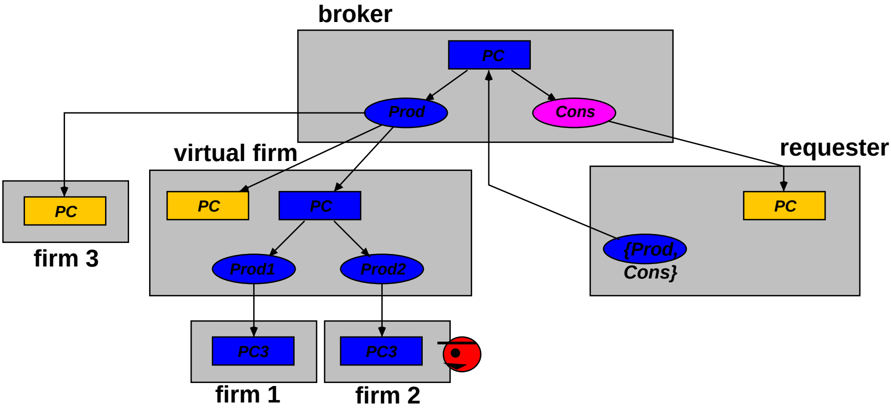
Example: Team Formation

Team Formation is the first of six teamwork phases.



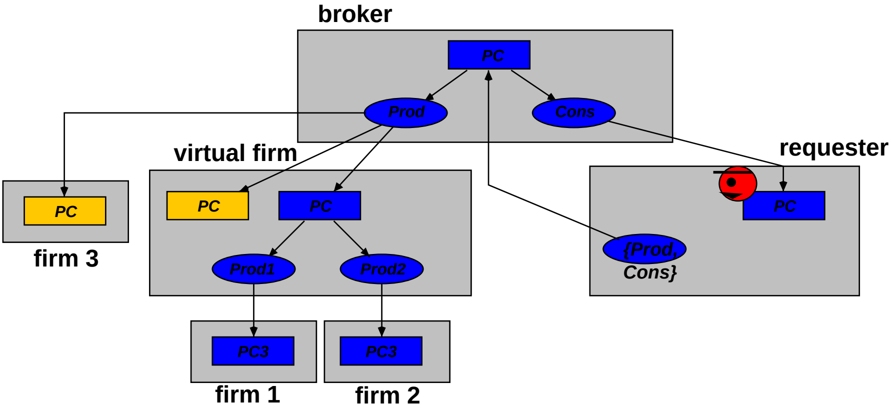
Example: Team Formation

Team Formation is the first of six teamwork phases.



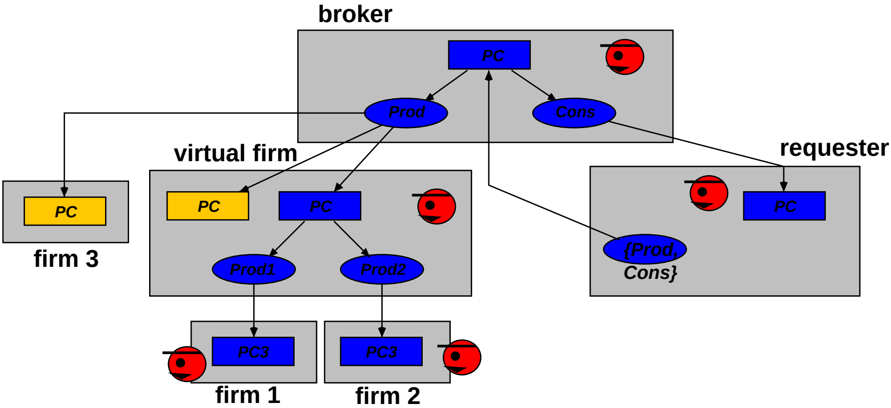
Example: Team Formation

Team Formation is the first of six teamwork phases.



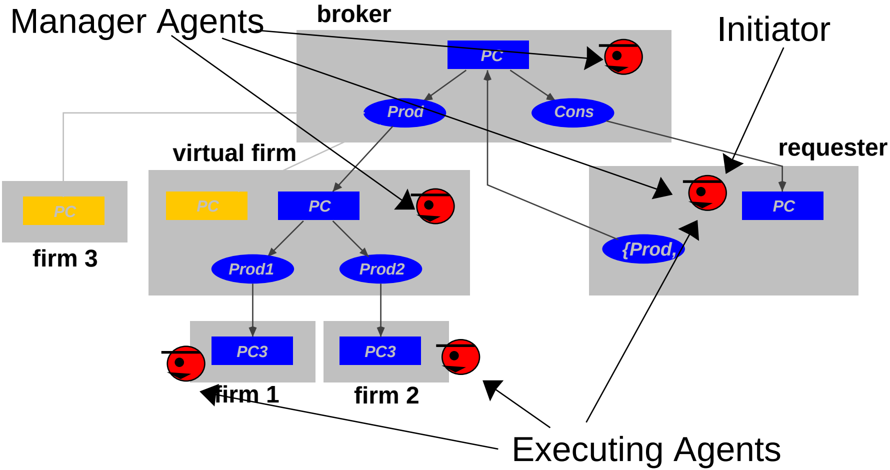
Example: Team Formation

Team Formation is the first of six teamwork phases.



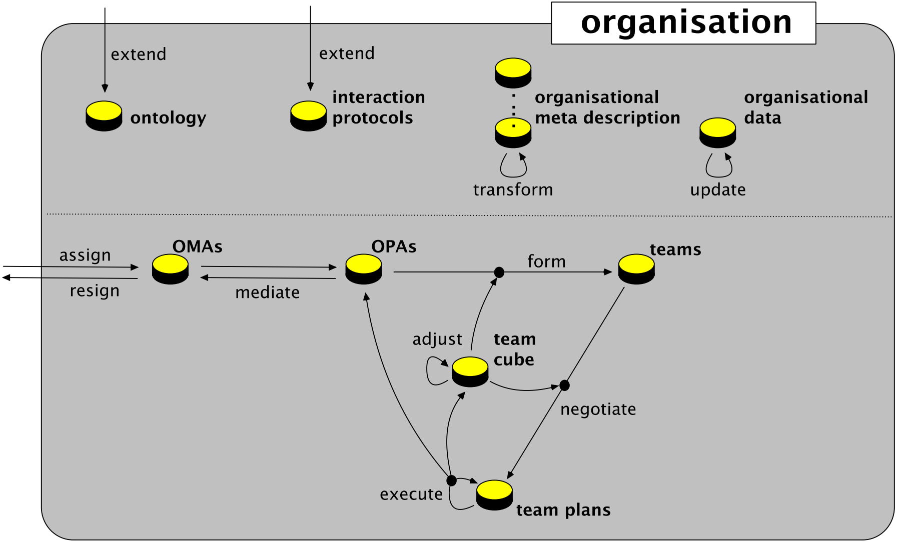
Example: Team Formation

Team Formation is the first of six teamwork phases.



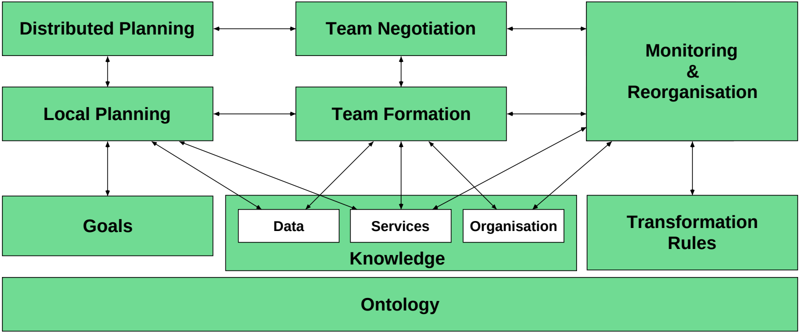
Generic Mulan4Sonar-Architecture: Organisation

Generic Organisational Agent:



Generic Mulan4Sonar-Architecture: OPA

Generic Organisational Position Agent (OPA):



Outline

1 Sonar

- Static Aspects: Position Network, Roles, AIP
- Dynamic Aspects: Teamwork
- Theoretical Aspects

2 Mulan4Sonar

- Deployment of SONAR-Models
- The Organisation Agent
- The Position Agent

3 The Sonar/Oredi Compiler

- Middleware: Design Objectives
- The Mulan4Sonar Prototype
- Compilation as Graph Transformation

4 Conclusion

Middleware: Design Objectives

1. Rapid development. Need for a specification language that inherently supports powerful high-level features like pattern matching and synchronisation patterns.
2. Small gap between the specification and implementation. Ideally, middleware specifications are directly executable.
3. Well established analysis techniques to study the prototype's behaviour.
4. As close as possible to the original SONAR model.
5. Automatic generation of the middleware specifications from the SONAR model.
6. Easy transfer of the prototype into an agent programming language.

Compilation into Mulan4Sonar

MULAN4SONAR: From a SONAR-model we compile a high-level Petri net, namely a reference net, that implements the organisation.

Main benefits

1. The translation result is very close to the original specification, since the prototype directly incorporates the main Petri net structure of the SONAR-model;
2. The prototype is immediately functional as reference nets are directly executable using the open-source Petri net simulator RENEW.
3. Various well-supported analysis techniques for Petri nets: model-checking, structural reduction, linear algebraic approaches, etc.
4. We can easily integrate the prototype into MULAN – our MAS framework based on Java and reference nets.

The SONAR/OREDI compiler is implemented as a RENEW-plugin.

It is part of the P*AOSE approach: `http://www.paose.net/`

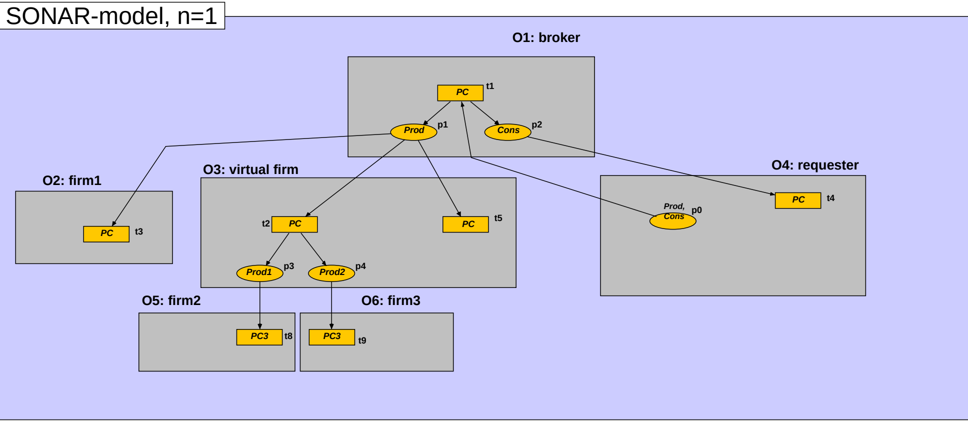
The Mulan4Sonar Prototype

Support the six phases of SONAR-Teamwork in MULAN4SONAR:

1. Select the team agents.
2. Generate the team as an object of discourse.
3. Announce the team among the team agents.
4. Recursively process partial plans.
   The executing team agents construct partial local plans.
   These partial plans are recursively processed via negotiation by the ancestors in the team tree, resulting in a global plan.
5. Localise global team plan by the executing team agents.
6. Instantiate a team interaction protocol and starts the execution.

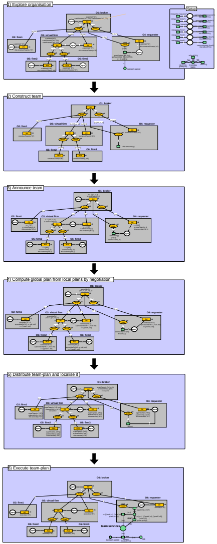
A Mulan4Sonar Example

Compiler input: The following SONAR model



A Mulan4Sonar Example

Compiler output: MULAN4SONAR middleware = 6 Petri Net modules

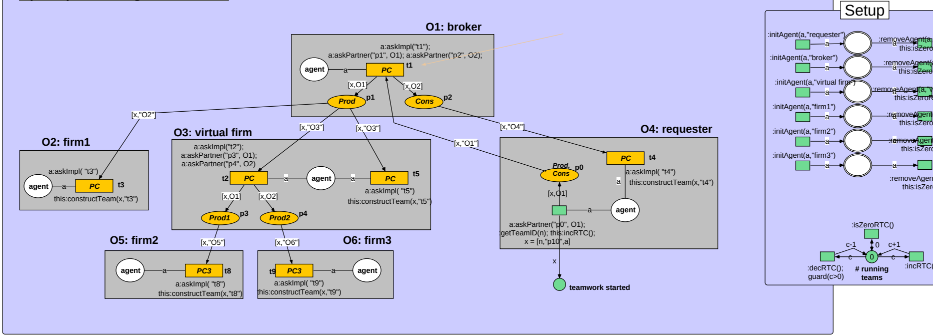


1. Select the team agents.
2. Generate the team.
3. Announce the team.
4. Negotiate: Recursively process partial plans.
5. Localise global team plan.
6. Instantiate a team interaction protocol.

6 Petri Net modules = 6 teamwork phases

A Mulan4Sonar Example

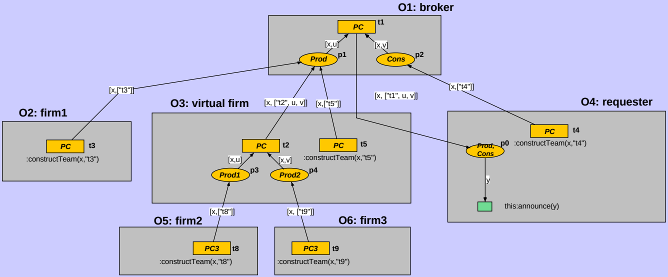
1) Explore organisation



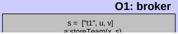
2) Construct team

A Mulan4Sonar Example

2) Construct team

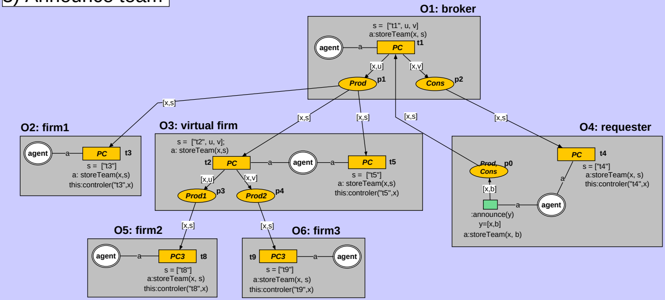


3) Announce team

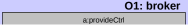


A Mulan4Sonar Example

3) Announce team

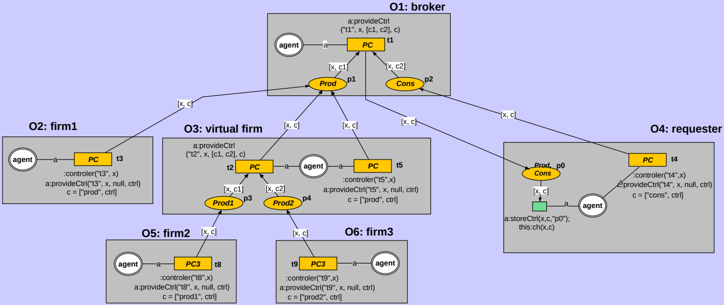


4) Compute global plan from local plans by negotiation



A Mulan4Sonar Example

4) Compute global plan from local plans by negotiation



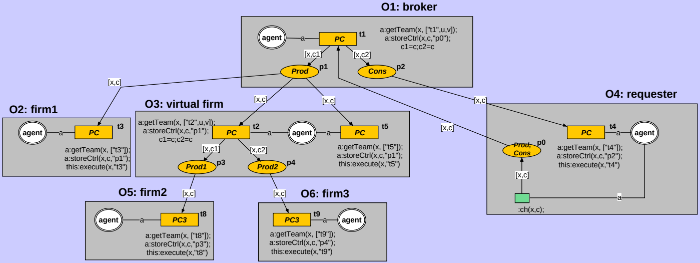
5) Distribute team-plan and localise it

O1: broker

A Mulan4Sonar Example

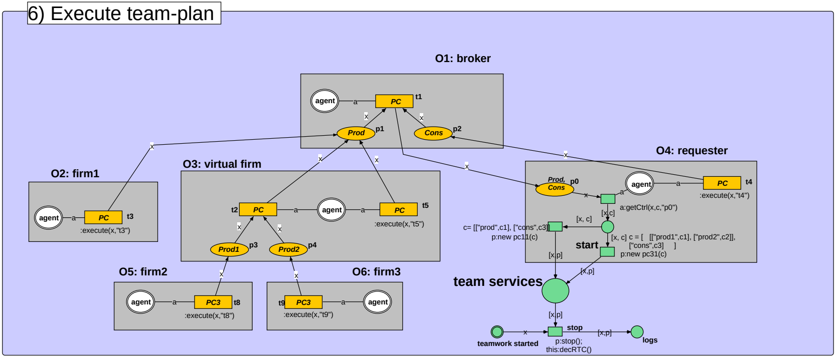
c = ["prod1", ctrl]   c = ["prod2", ctrl]

5) Distribute team-plan and localise it



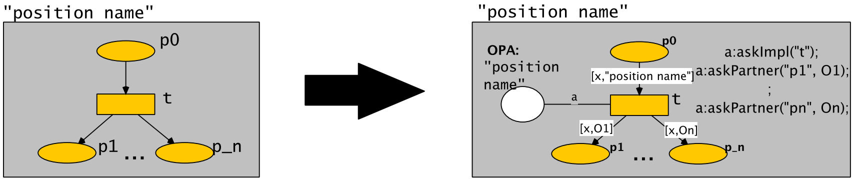
A Mulan4Sonar Example

6) Execute team-plan



Compilation as Graph Transformation

Example: One graph transformation rule for phase 1:



Sonar Model

Prototype

- add the place OPA which contains the position agent
- ask OPA to choose among its internal implementations for the input place via channel `a:askImpl(..)`
- ask OPA to choose delegation partners for each output place via channel `a:askPartner(..)`

Outline

1. Sonar
   - Static Aspects: Position Network, Roles, AIP
   - Dynamic Aspects: Teamwork
   - Theoretical Aspects
2. Mulan4Sonar
   - Deployment of SONAR-Models
   - The Organisation Agent
   - The Position Agent
3. The Sonar/Oredi Compiler
   - Middleware: Design Objectives
   - The Mulan4Sonar Prototype
   - Compilation as Graph Transformation
4. Conclusion

Conclusion

- Sonar
  - Static Aspects: Position Network, Roles, AIP
  - Dynamic Aspects: Teamwork
  - Theoretical Aspects: Petri Net analysis
- The Mulan4Sonar Middleware
  - The Organisation Agent
  - The Position Agent
  - Member Agents and External Agents
- The Sonar/Oredi Compiler
  - The Mulan4Sonar Prototype based on Reference Nets
  - Compilation as Graph Transformation
- P*AOSE approach: `http://www.paose.net/`

- Outlook: *Higher-Order Teamwork* modifies the organisation.

Conclusion

- Sonar
  - Static Aspects: Position Network, Roles, AIP
  - Dynamic Aspects: Teamwork
  - Theoretical Aspects: Petri Net analysis
- The Mulan4Sonar Middleware
  - The Organisation Agent
  - The Position Agent
  - Member Agents and External Agents
- The Sonar/Oredi Compiler
  - The Mulan4Sonar Prototype based on Reference Nets
  - Compilation as Graph Transformation
- P*AOSE approach: `http://www.paose.net/`

- Outlook: *Higher-Order Teamwork* modifies the organisation.