Using Method Engineering for the Construction of Agent-Oriented Methodologies

Giancarlo Fortino, Alfredo Garro , Wilma Russo
g.fortino@unical.it, {garro, russow}@deis.unical.it

D.E.I.S. - Università della Calabria, ITALY

Outline

❶ Introduction and Motivations

❷ Two approaches for the construction of agent-oriented methodologies by using method fragments integration: meta-model-driven and process-driven

❸ An example of application of a methodology built-up using the process-driven approach

Introduction and motivations

- Recently the agent-oriented approach has been widely recognized as very suitable for the development of **complex software systems**.
- In particular:
  - the **agent-oriented decompositions** are an effective way of partitioning the problem space;
  - the **key abstractions of the agent-oriented mindset** (agents, interactions, and organizations) are a natural means of modeling;
  - the **agent-oriented** philosophy for modeling and managing **organizational relationships** is appropriate for dealing with existing dependencies and interactions.

Introduction and motivations

- **The development of complex software systems by using the agent-oriented approach requires suitable agent-oriented modelling techniques and methodologies which provide explicit support for the key abstractions of the agent paradigm.**

- Existing development methodologies have different advantages when applied to specific problems. However…
  …it seems to be widely accepted that an unique methodology cannot be general enough to be useful to everyone without some level of customization… so

  an approach that combines the designer's need of defining her/his own methodology with the advantages and the experiences coming from the existing and documented methodologies is highly required.

Introduction and motivations

•A possible solution to this problem is to adopt the **method engineering** paradigm so enabling designers of MAS to use phases or models or elements coming from different methodologies to build up a customized approach for their own problems, in particular:

- •the development methodology is constructed by the developer assembling pieces of methodology (**method fragments**) from a repository of methods (**method base**);
- •the method base can be built up by taking pieces coming from existing methodologies (ADELFE, Gaia, MESSAGE, PASSI, Tropos, etc.) or ad hoc defined.

•It is therefore crucial to define guidelines for methods integration in order to both construct the methodology (retrieving the method fragments from the method base and integrating them) and apply it in the actual development life cycle.

Introduction and motivations

In this direction, this work proposes two approaches for the construction of agent-oriented methodologies by using methods integration:

(i) meta-model driven;

(ii) development process driven;

The MAS meta-model driven approach

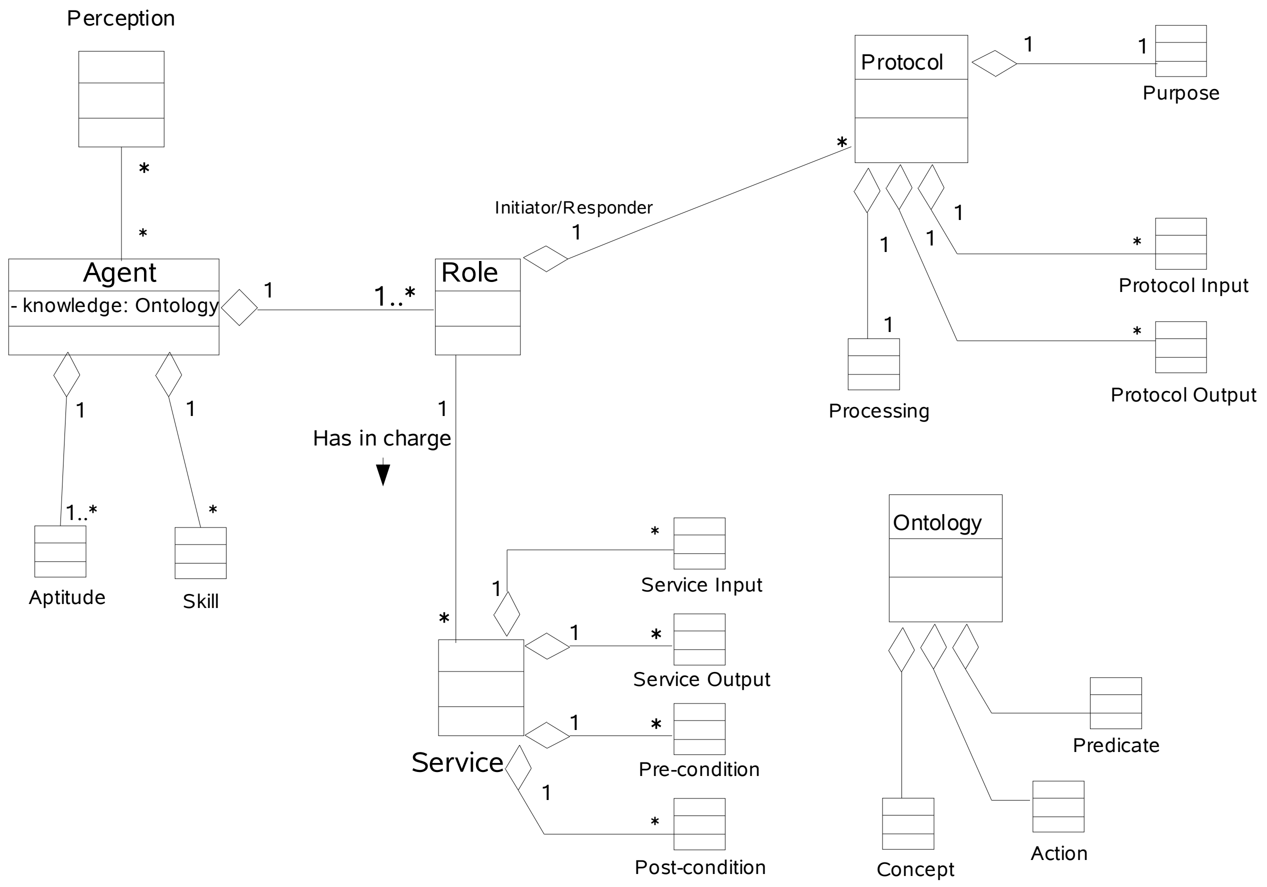
To build her/his own methodology by exploiting the ***meta-model driven*** approach, the designer “must”:

- choose or define a MAS meta-model suitable for the specific problem and/or the specific application domain;
- identify the elements that compose the meta-model of the MAS under development;
- choose the method fragments that are able to produce the identified meta-model elements;
- defining a development process characterized by a method fragments execution order on the basis of the relationship existing among the meta-model elements produced by each fragment.

Hence, the obtained methodology is able to completely “cover” the MAS meta-model for a given problem in a specific application domain.

The MAS meta-model-driven approach: an example

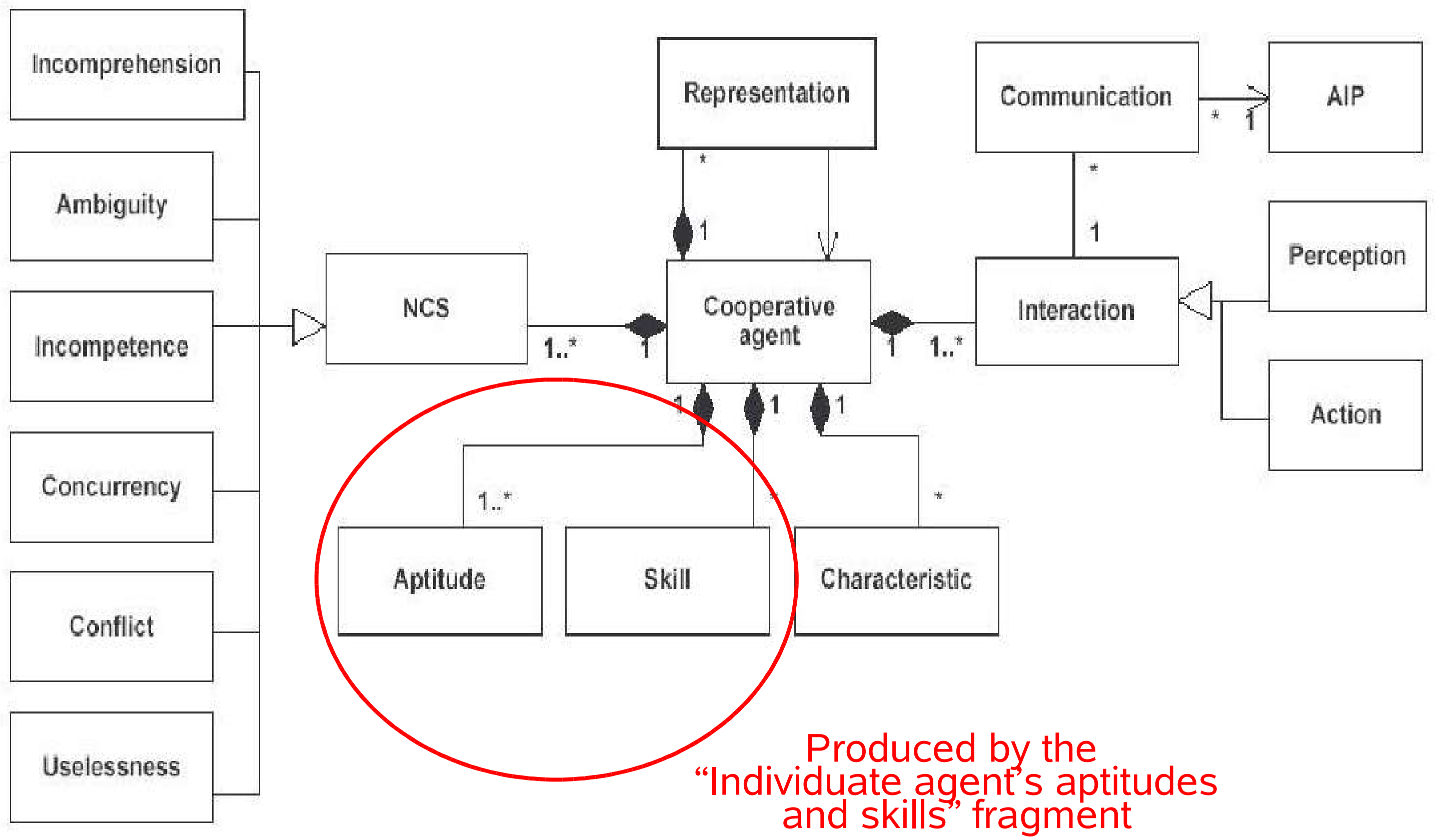
Let us consider the following MAS (meta) model that we want to adopt for solving a specific problem in a specific application domain:



The MAS meta-model-driven approach: an example

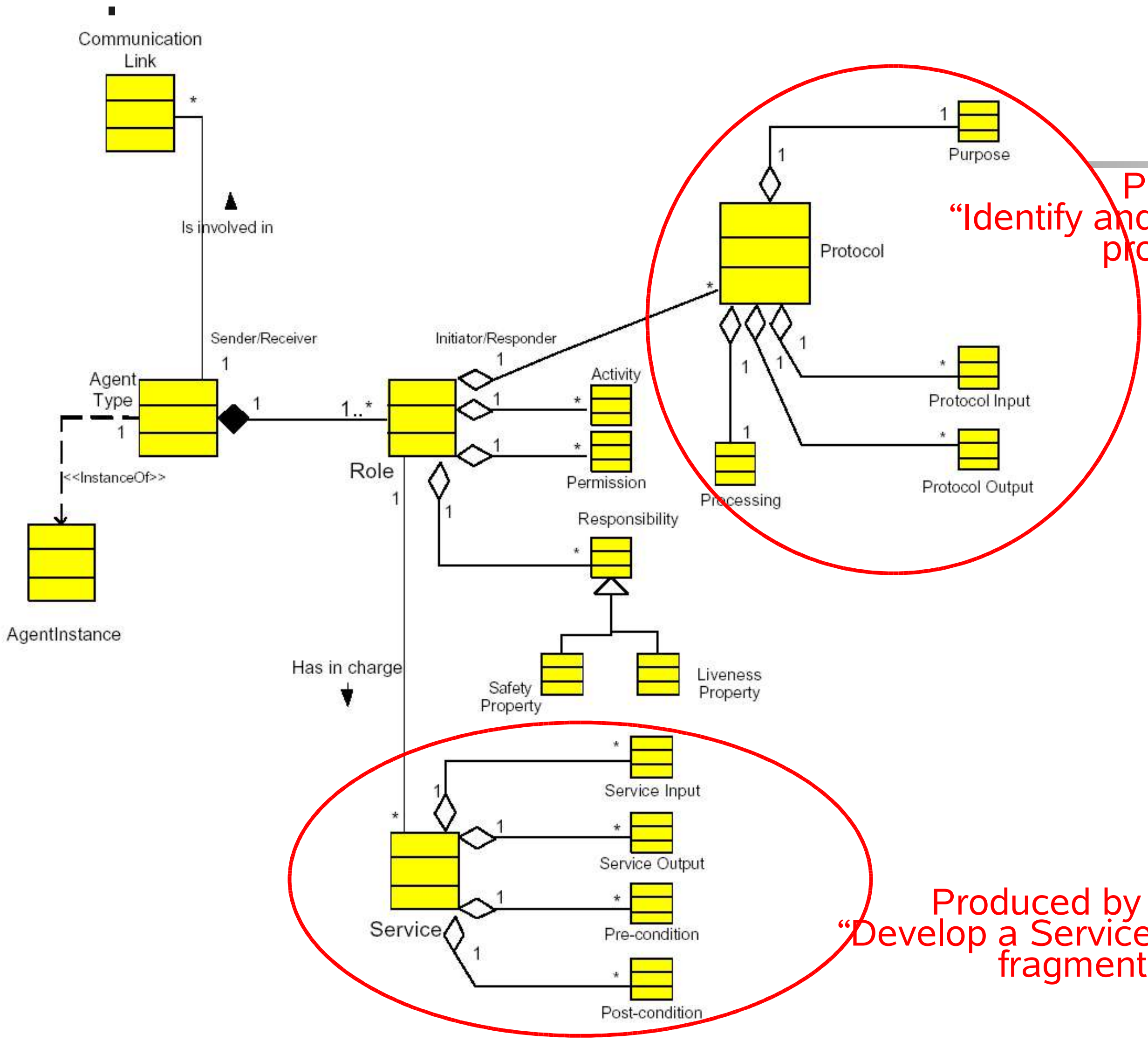
Referring to the MAS meta-models adopted by the **Adelfe**, the **Gaia** and the **Passi** methodologies, a set of method fragments that are able to produce "***pieces***" of the showed MAS meta-model can be chosen…

ADELFE MAS (meta)model



Produced by the
“Individuate agent’s aptitudes and skills” fragment

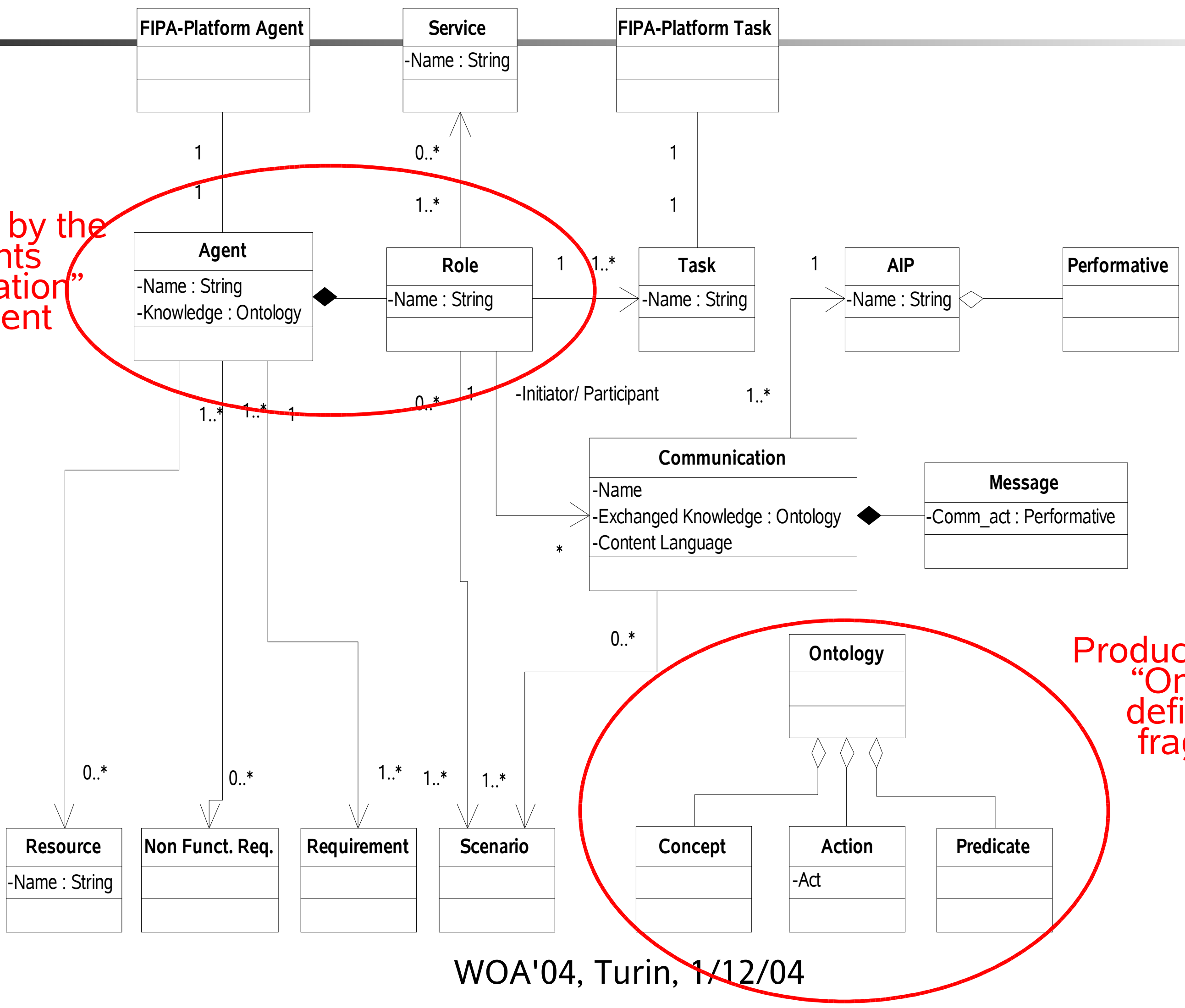
GAIA MAS (meta)model



Produced by the "Identify and document the interaction protocols" fragment

Produced by the "Develop a Services Model" fragment

PASSI MAS (meta)model

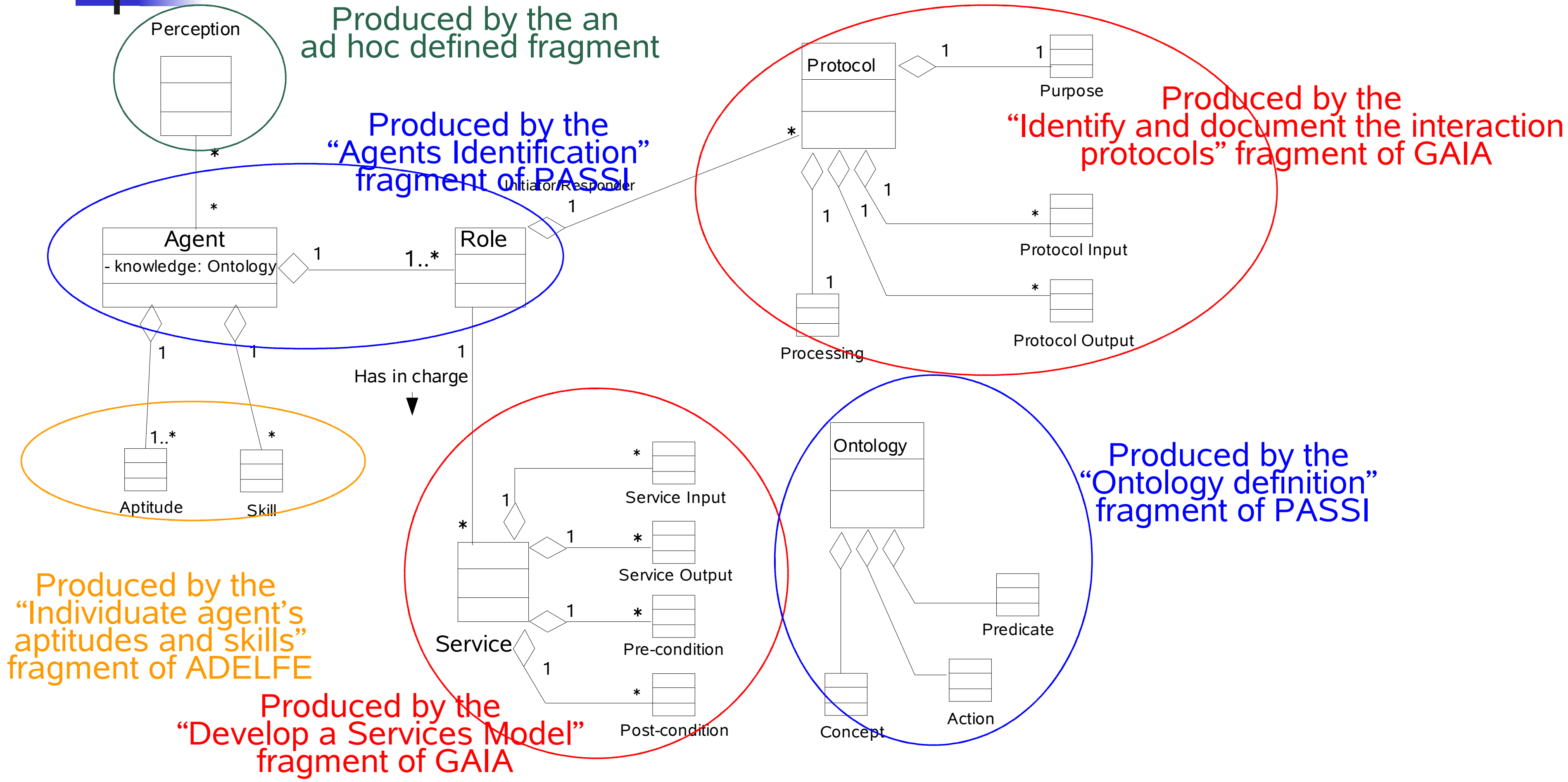


Produced by the "Agents Identification" fragment

Produced by the "Ontology definition" fragment

The MAS meta-model-driven approach: an example

To completely cover the MAS meta-model that we want to adopt for solving a specific problem in a specific application domain the selected fragments can be combined and, if necessary, new fragments can be defined:


Perception
Produced by the an ad hoc defined fragment
Produced by the "Agents Identification" fragment of PASSI
Initiator/Responder
Agent
- knowledge: Ontology
Role
1
1..*
*
Has in charge
Protocol
Purpose
Produced by the "Identify and document the interaction protocols" fragment of GAIA
Protocol Input
Protocol Output
Processing
Aptitude
Skill
1..*
Service
Service Input
Service Output
Pre-condition
Post-condition
Ontology
Predicate
Action
Concept
Produced by the "Ontology definition" fragment of PASSI
Produced by the "Individuate agent's aptitudes and skills" fragment of ADELFE
Produced by the "Develop a Services Model" fragment of GAIA

The development process-driven approach

The ***development process-driven*** approach focuses on the *instantiation* of a software development process that completely covers the development of MAS.

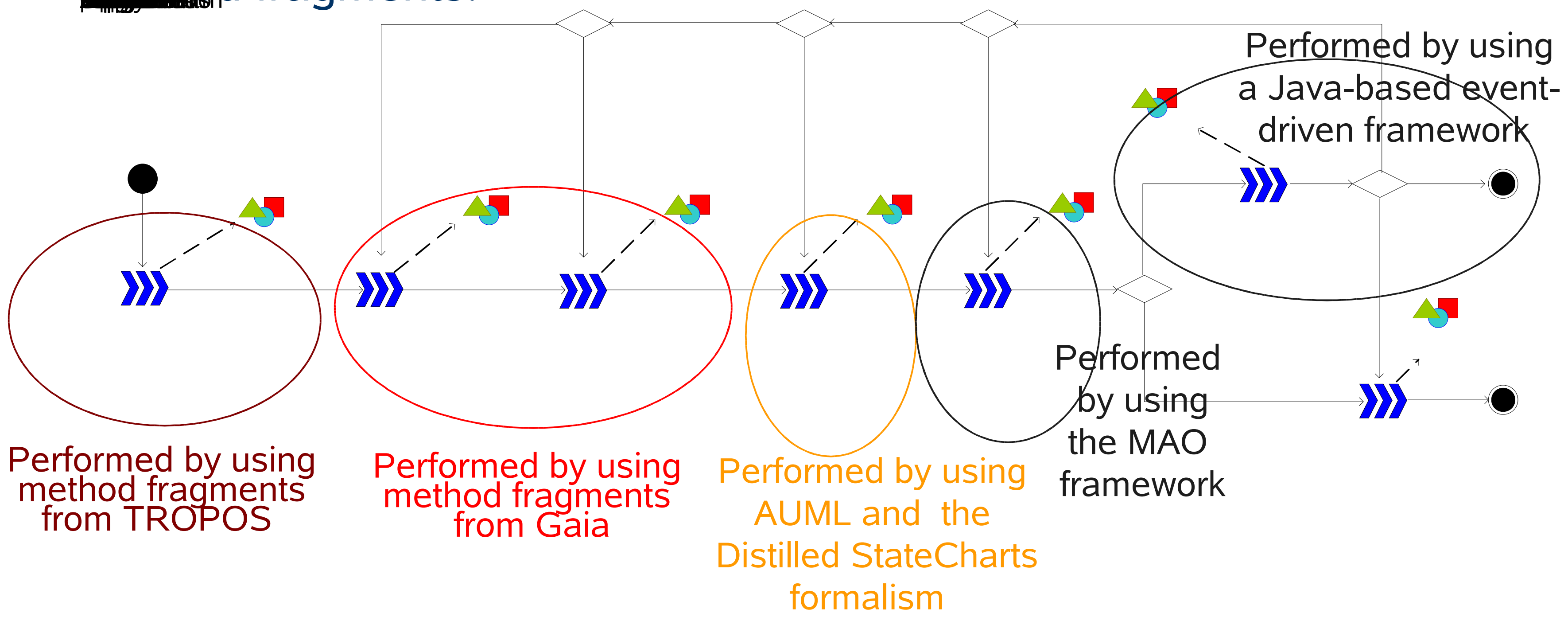
To build her/his own methodology by exploiting the development process-driven approach, the designer must:

- •choose or define a software development process suitable for the specific problem and for the specific application domain;
- •instantiate the development process by selecting, for each phase, *suitable method fragments*, chosen from agent-oriented methodologies proposed in the literature or ad-hoc defined.

The development process-driven approach: an example

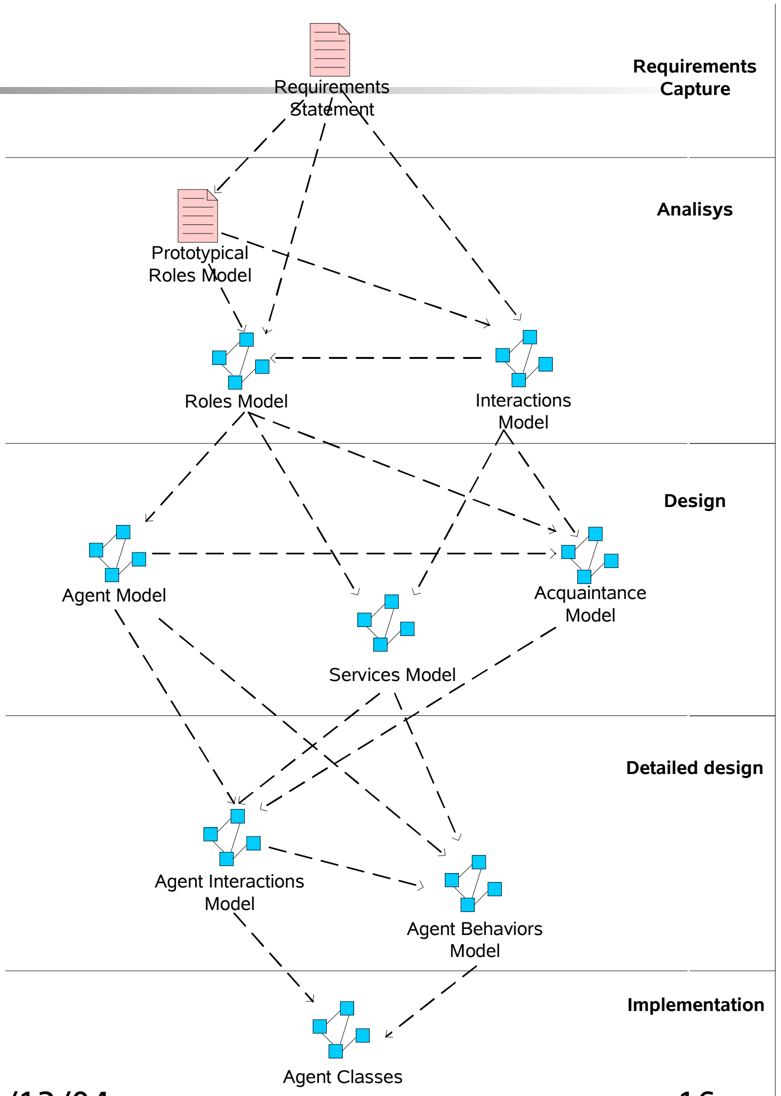
Let us consider the following software development process:

we can instantiate this process by selecting, for each phase, suitable method fragments:


Performed by using a Java-based event-driven framework
Performed by using the MAO framework
Performed by using method fragments from TROPOS
Performed by using method fragments from Gaia
Performed by using AUML and the Distilled StateCharts formalism

The development process-driven approach: an example

Using this approach, the integration between the fragments is achieved by individuating and/or defining dependencies among work products produced by the fragments of the instantiated process.



An example: a methodology for the modeling and the validation through simulation of MAS

- None of the emerging and well documented methodologies supports **validation through simulation** of the MAS under-development as it is required for complex MAS.
- A possible solution is to exploit the ***development process driven*** approach to method fragments integration for the construction of a methodology for the modeling and the validation through simulation of MAS.
- The constructed methodology centers on the instantiation of a software **development process** which specifically includes a **simulation phase** which makes it possible the validation of a MAS under-development before its actual deployment and execution.

A methodology for the modeling and the validation through simulation of MAS

- The constructed methodology was applied for the definition and the validation through simulation of a **consumer-driven e-Marketplace** model which offers mobile agent-based services for searching and buying goods.
- The results obtained from the simulations of the defined consumer-driven e-Marketplace allowed to:
  - evaluate which task execution model is more appropriate with respect to a set of buying policies and for the characteristics of the e-Marketplace, and
  - validate the analytical model proposed by Wang, Tan, and Ren regarding the sequential and parallel dispatching of mobile agents.

The MAS meta-model-driven approach: pros & cons

The meta-model-driven approach provides the following advantage:

flexibility for the definition of methodologies and meta-models of the MAS to be developed

However, it has some drawbacks:

- difficulty of integration of different fragments due to different semantics of the meta-model concepts;
- selection and/or definition of the meta-model to adopt for the specific problem and/or application domain.

The development process-driven approach: pros & cons

- The development process-driven approach is characterized by the following advantages:

simplicity for the construction of methodologies by means of the instantiation of each stage of the development process

However, the disadvantages are the following:

- process rigidity;
- low flexibility of the system meta-model since the meta-model of an adopted methodology must be used;
- adaptation among the work products which is sometimes difficult to achieve;
- choice and definition of the process to instantiate for the specific problem and/or application context

Conclusions & Future Work

- This paper has proposed two "*approaches*" to the integration of methods fragments: *meta-model-driven* and *development process-driven*.
- These approaches are not mutually exclusive; rather, hybrid approaches containing features of both of them might be defined as well.

Conclusions & Future Work

- On going research activity is focused on:
  - definition of adaptation techniques among work products produced by different methods and/or method fragments;
  - extraction from and definition of method fragments of already existing methodologies and the mutual adaptation among the defined method fragments. This activity is was carried out in the context of the FIPA Methodology TC and is being in the context of Agentlink III, AOSE TC;
  - the experimentation of the two presented approaches for the e-Commerce application domain.

Method Fragment

A method fragment [18] is a portion of methodology which is composed of the following parts:

- A process specification, defined with a SPEM diagram [20], which defines the procedural aspect of the fragment;
- One or more deliverables such as AUML/UML diagrams and text documents [3];
- Some preconditions which represent a kind of constraint since it is not possible to start the process specified in the fragment without the required input data or without verifying the required guard conditions;
- A list of elements (which is a part of the MAS meta-model subsumed by the methodology from which it was extracted) to be defined or refined through the specified process;
- Application guidelines that illustrate how to apply the fragment and related best practices;
- A glossary of terms used in the fragment in order to avoid misunderstandings if the fragment is reused in a context that is different from the original one;
- Composition guidelines which describe the context/problem that is behind the methodology from which the specific fragment is extracted;
- Aspects of fragment which are textual descriptions of specific issues such as platform to be used, application area, etc;
- Dependency relationships useful to assemble fragments.

It should be noted that not all of these elements are mandatory; some of them (for instance notation or guidelines) could be not applicable or not necessary for some specific fragment.