Intelligent front-ends for scientific problem solving

V. Negru
E-Austria Institute Timisoara

Content

- Intelligent front-end NESS
- MATOPS: Multi-Agent Architecture
- Implementing NESS in MATOPS
- AgentDiscover front-end
- Conclusions

- Intelligent front-end NESS
- MATOPS: Multi-Agent Architecture
- Implementing NESS in MATOPS
- AgentDiscover front-end
- Conclusions

NESS: Overview

- Nonlinear Equation Systems Solver (1998→…)
- Goals:
  - Acquiring and formalizing mathematical expertise (problem solving methods properties, hints for the generation of initial iteration, choosing NESS PSMs, etc)
  - Identifying NESS problems properties
  - Developing and using NESS methods
  - Designing a task-based formalism for modeling the reasoning (using UPML)

Task Reasoning Systems

- Problem decomposition in sub-problems
- Task = problem to solve
- Task based reasoning = solving by problem decomposition
- Advantages:
  - Similar to the human way of solving problems
  - Allows for describing at different abstraction levels
  - Appropriate for describing actions on knowledge: performing tasks

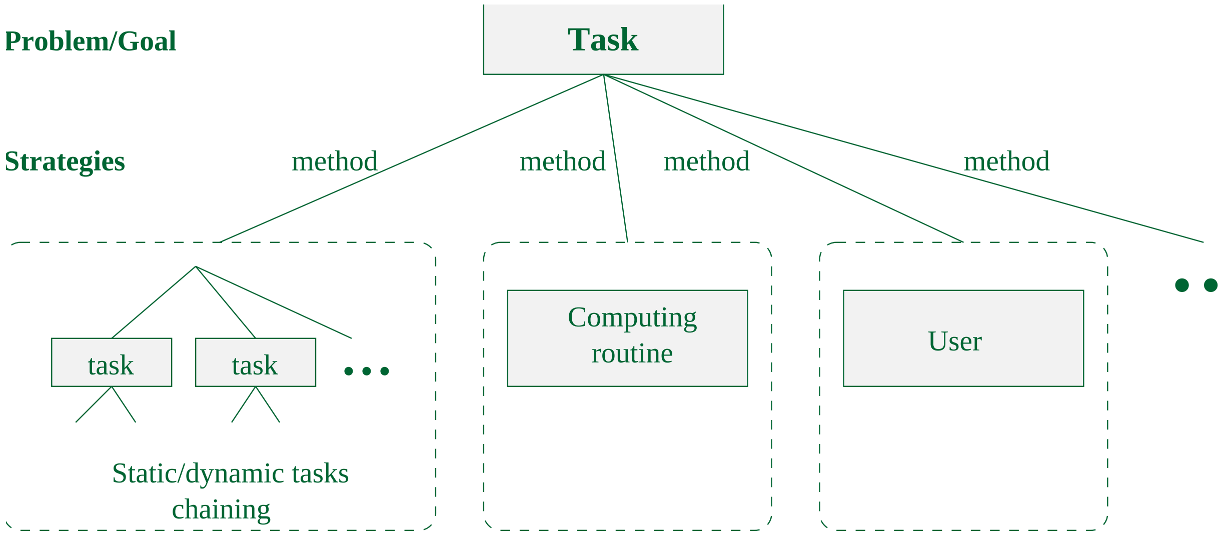
NESS – Reasoning abstraction

- Functional approach
  - generic tasks (Chandrasekaran, 1983)
  - problem solving methods (McDermott, 1988)
  - expertise components (Steels, 1990)
  - task structure (Chandrasekaran, 1990)
  - CSRL(Bylander & al. 1986), HYPER (Johnson & al. 1989), PEIRCE (Punch & al. 1990)

NESS – Reasoning abstraction

- Conceptual approach
  - Conceptual models (task, method, model)
  - Modeling based on multiple points of view
  - Components of expertise (domain, inference, tasks)
  - the inference structures (Clancey, 1985)
  - KADS (Wielinga 1990, 1992), KARL (Landes 1994), PROTÉGÉ (Musen, Genari & al. 1989-2000)

Task – methods relationship



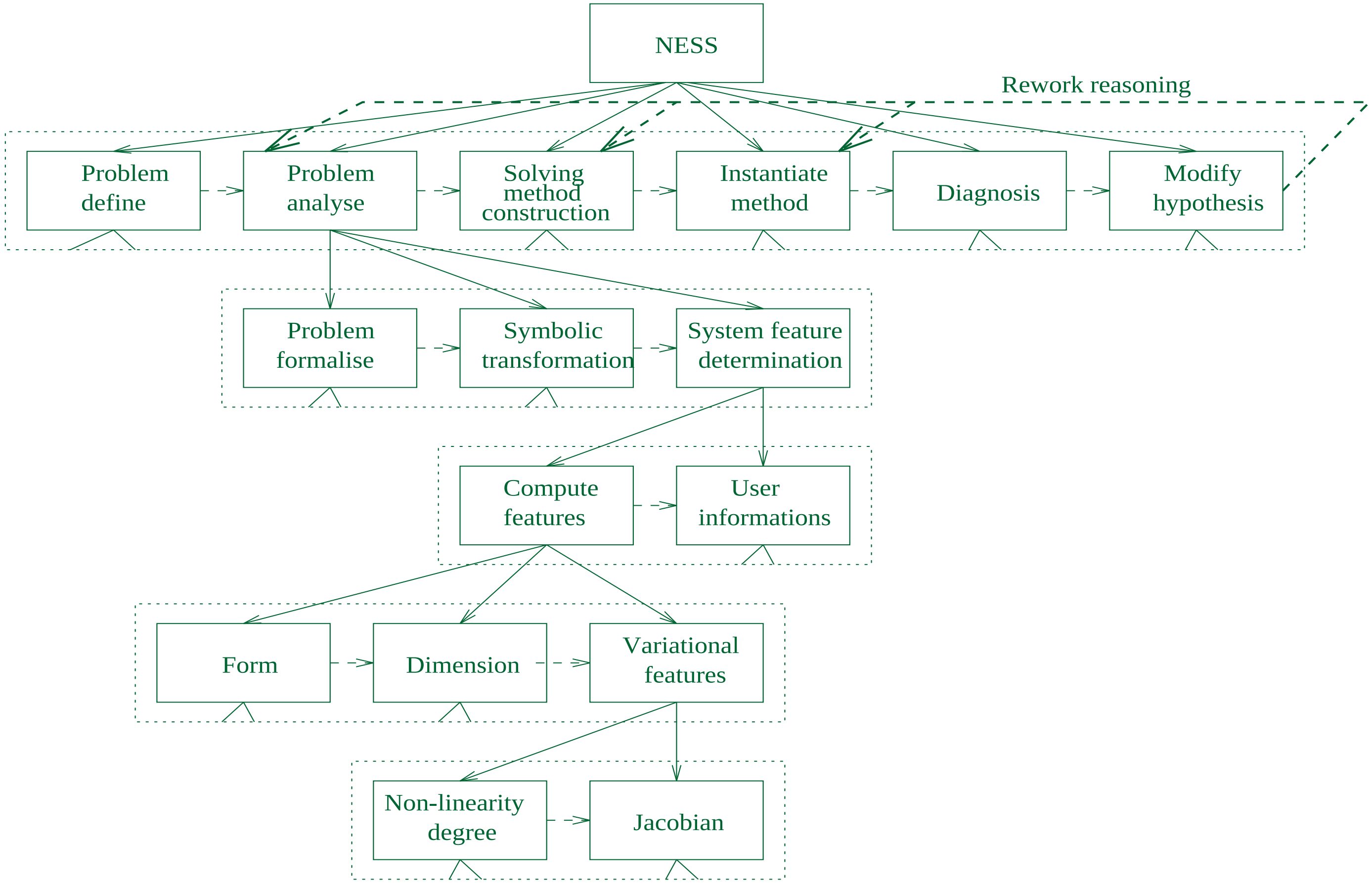
NESS - description

- Problem solving
  - the problem analyze for determinate the system properties;
  - the choose of the most suitable numerical method according with the properties already determinated and with the characteristics of the known methods;
  - after the execution of the numerical method, the expert analyze the intermediary and final results;
  - in the case of failure, he try a diagnosis and a rework of solving, by changing some conditions or by changing the method.
- Equations system features:
  - the system form: general, sparse, sum between a linear and a non-linear and a diagonal part, explicitly diagonal, special, etc.;
  - variational features: linearity degree (non-specific variation, weak non-linear), Jacobian (nonsingular, positive defined, singular);
  - dimension: small ($n \leq 10$), medium ($10 < n \leq 50$), big ($50 < n \leq 500$), very big ($500 < n$).

NESS - description

- Numerical methods characteristics:
  - rate of convergence
  - domain of convergence,
  - computing effort per iteration
  - used memory
  - asymptotically error
  - initial data
  - the sparse conservation
  - the dependency of the discretization step
- We are using classical numerical methods (Newton, Secant etc.), and also recent methods (conjugated gradient, Broyden etc.).

NESS – partial task tree

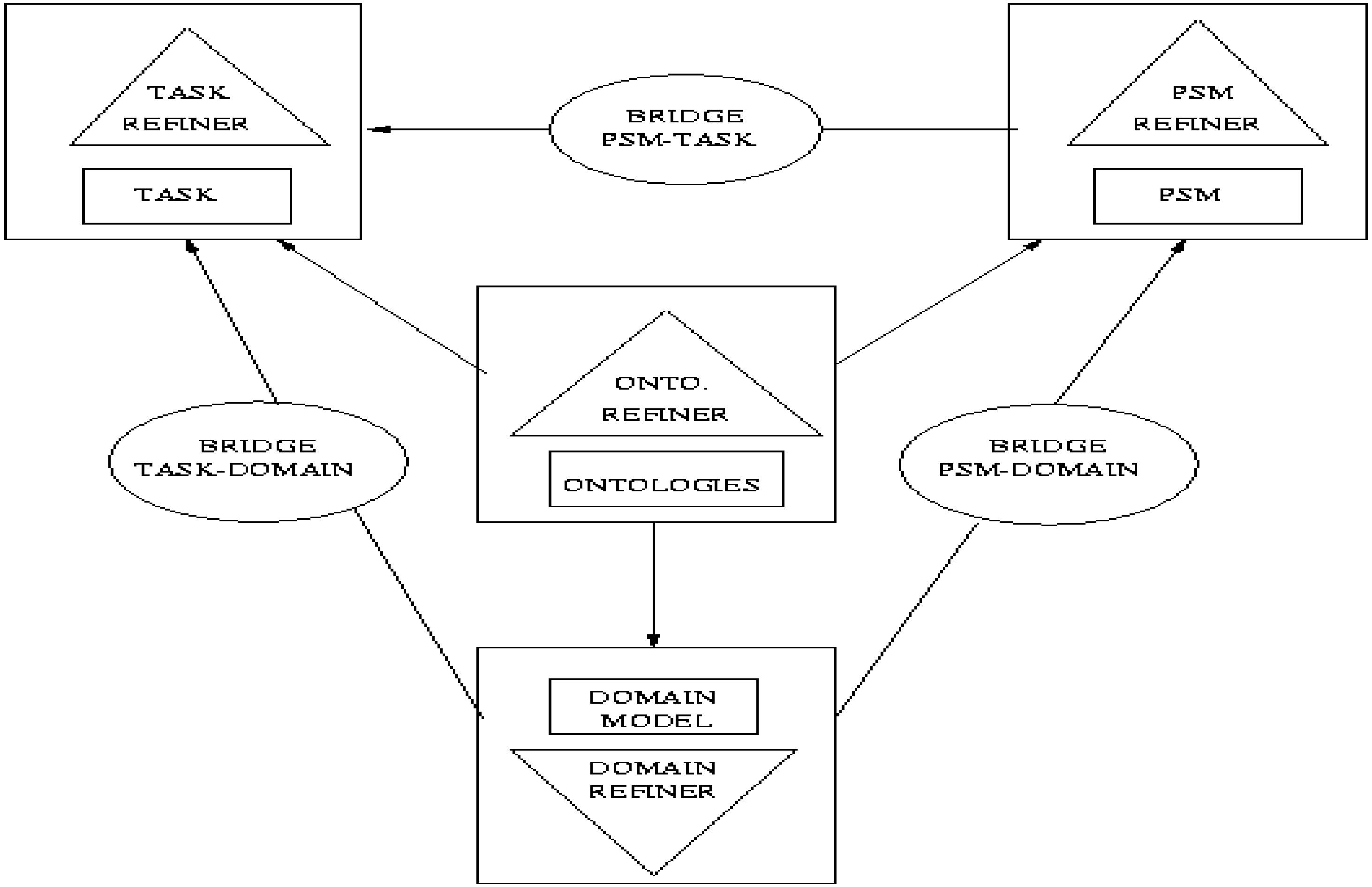


- Intelligent front-end NESS
- MATOPS: Multi-Agent Architecture
- Implementing NESS in MATOPS
- AgentDiscover front-end
- Conclusions

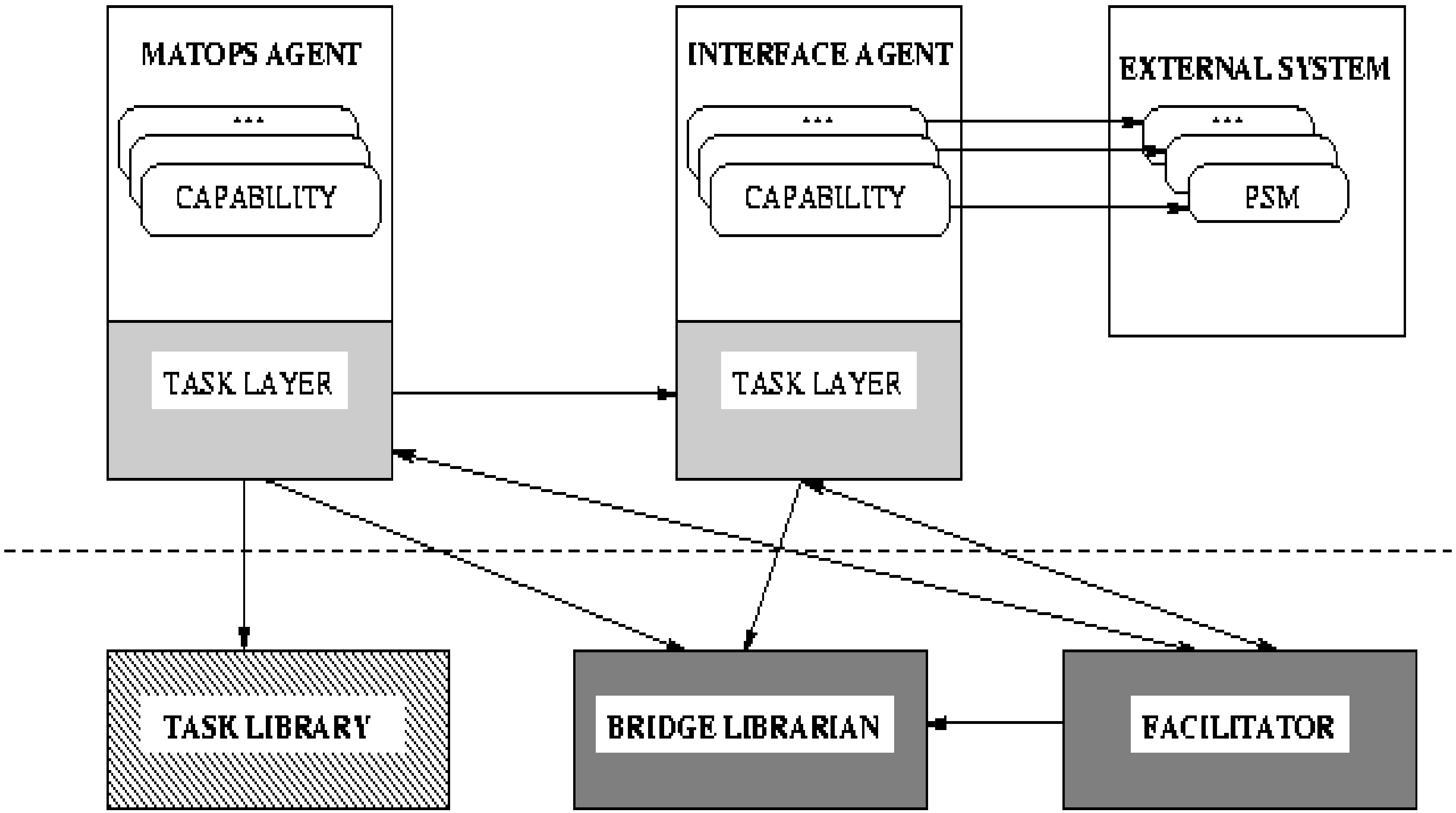
Goal

- To propose a new architecture for problem solving, which should be generic enough to be instantiated for various problems
- Multi-agent based
- Task-oriented
- Based on UPML
- Easy to interface with external modules
- Flexible

Unified Problem-solving Method development Language (UPML)



Multi-Agent Task Oriented Problem Solving Architecture



MATOPS: Overview

- Problem solving methods (PSM) → Agent capabilities
- UPML → Capability Description Language
- Brokering PSM → Facilitating agent capabilities
- Tasks → Descriptions used when facilitating capabilities

UPML → MATOPS

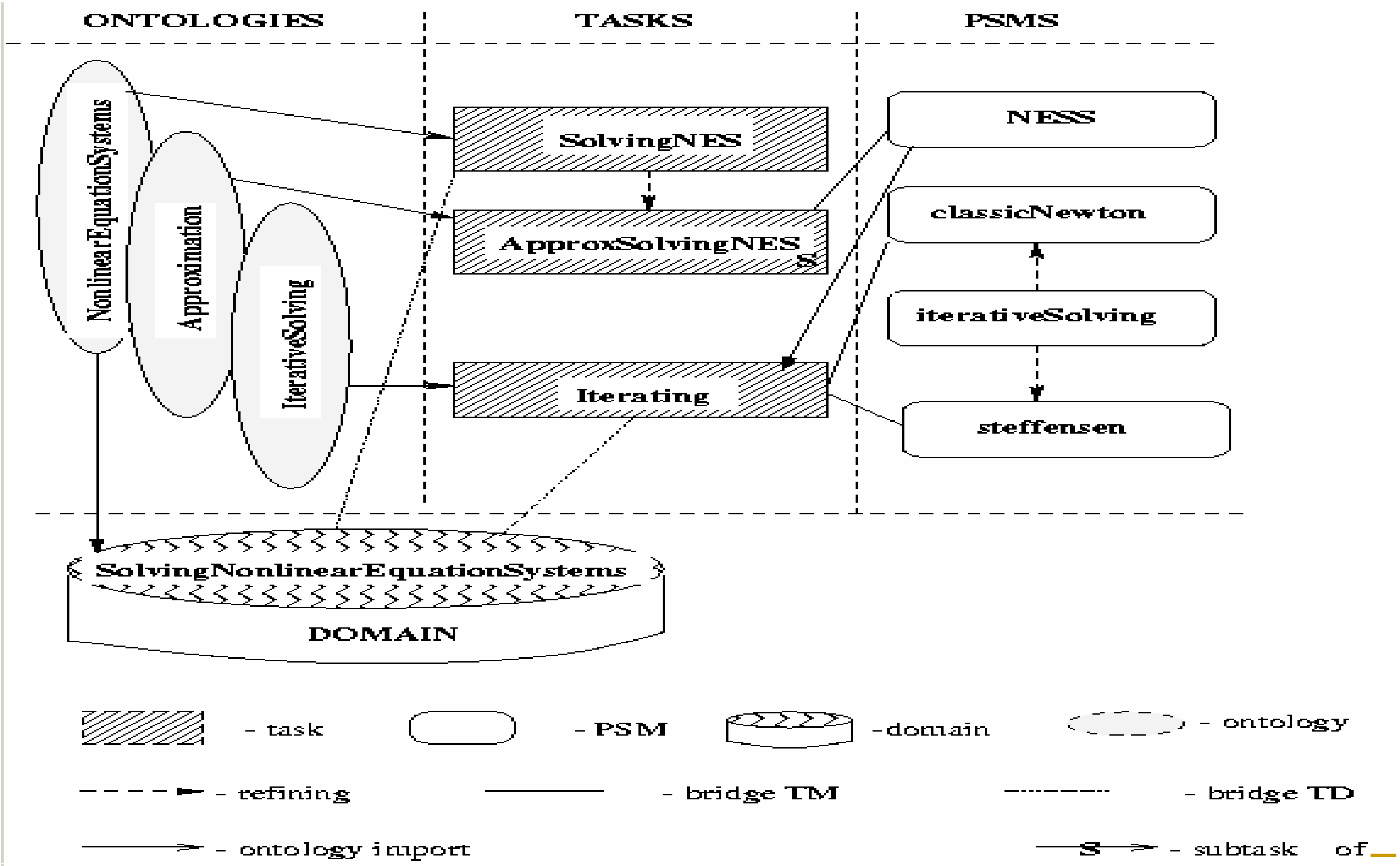
$$
\begin{array}{l}
\textit{AssignedCapability} \\
\hline
name : Attribute \\
roles : Roles \\
preconditions : \mathbb{P}_1\ Formula \\
postconditions : \mathbb{P}_1\ Formula \\
subtasks : \mathbb{P}\ Task \\
control : Program \\
agent : MATOPSAgent \\
executeOperationalProcedure : Program \rightarrow RequestedTask \rightarrow \\
\quad Result \times \mathrm{seq}\ ActualRole \\
\hline
\end{array}
$$

- Different kinds of bridging
  - Bridging tasks and PSMs
    - Specify possible PSMs achieving a task
    - Provide ontological mappings
  - Bridging tasks and tasks
    - Choose PSMs to realize a task – UPML extension
- Need for an agent managing bridges: bridge librarian

MATOPS: Facilitator Agent

- MATOPS-CDL: XML/RDF/RDFS/OWL describing UPML PSMs and tasks
- Brokering algorithm
  1. Candidate methods (capabilities) selection
     - Using task-task bridges
     - Using task-PSM bridges
  2. Competence-based filtering (promoted by UPML)
     - Ontological based role checking
     - Pre-conditions evaluations
     - Subtasks checking
  3. Sorting capabilities
     - Based on capabilities properties
     - Matching with requested properties of a task

NESS: MA Architecture



NESS: MA Benefits

- Easy to add new solving procedures by including different solving environments into the same application
- Heterogeneous agents
- Using *Facilitator* and *Bridge Librarian* agents manage and use the bridging information in order to select agent capabilities to solve a particular task

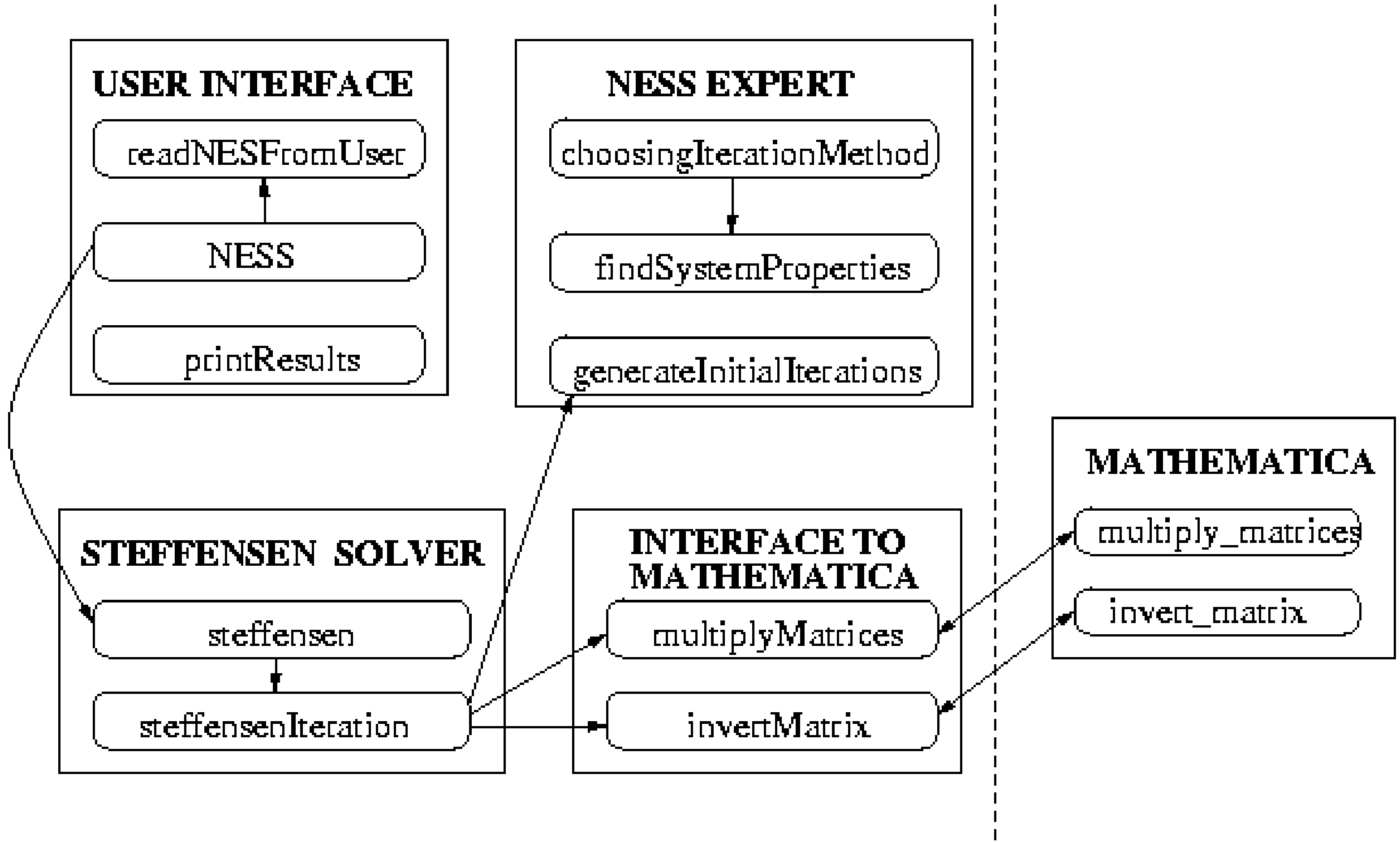
NESS: Method Chooser Task

- Problem: to choose PSMs to realize a task
- competence based vs. domain based selection
- Ex: ChoosingIterationMethod task for the Iterating task

task *Iterating*
  pragmatics
    Generic task executing an iterative process on a function
  ontology *IterativeSolving*
  specification
    roles
      input *function*; **input** *error*; **input** steps;
      output *iteratie*;
    method-chooser-task
      all *x. ChoosingIterationMethod*(*function; x*)

…

NESS: Agents Cooperation



Conclusions - MATOPS

- The model of expertise proposed in UPML has been adapted to a multi-agent context
- Use of existing generic problem solving methods (implemented as capabilities of specific components - agents)
- Supports heterogeneity of the involved technologies in the application
- Using bridging families of terms describing various components of the application can be used together
- Supports the interactions with external systems (CASs in our NESS application)
- Method Choosing Task for another task

- MATOPS:
  - New agent type: the bridge librarian – explicitly manage bridges
  - Particular brokering algorithm based on interaction with the bridge librarian while task properties are used to sort the results
  - Applied on NESS

- FUTURE WORK
  - Validate MATOPS for other problems: mathematical, e-commerce, etc.
  - Identify other kind of bridges affecting agents operation (e.g. task monitoring the execution of a task)
  - Study how to extend UPML capability description in order to be more expressive in a MA context (e.g. capability’s preconditions may be evaluated in the context of the caller agent, executing agent or both)

- Intelligent front-end NESS
- MATOPS: Multi-Agent Architecture
- Implementing NESS in MATOPS
- AgentDiscover front-end
- Conclusions

AGENT DISCOVER – a MAS for Knowledge Discovery from Databases

- Goal
  - For novice users: to obtain, in a short period of time, satisfactory outcomes out of the KDD process based on recommendations proposed by system
  - For advanced users: to easily explore various KDD algorithms
- Objectives
  - To integrate various knowledge representation models (set of rules, decision table, decision trees etc.)
  - Extensibility
  - Transparency and configurability of the whole process
  - Recommendation of solutions

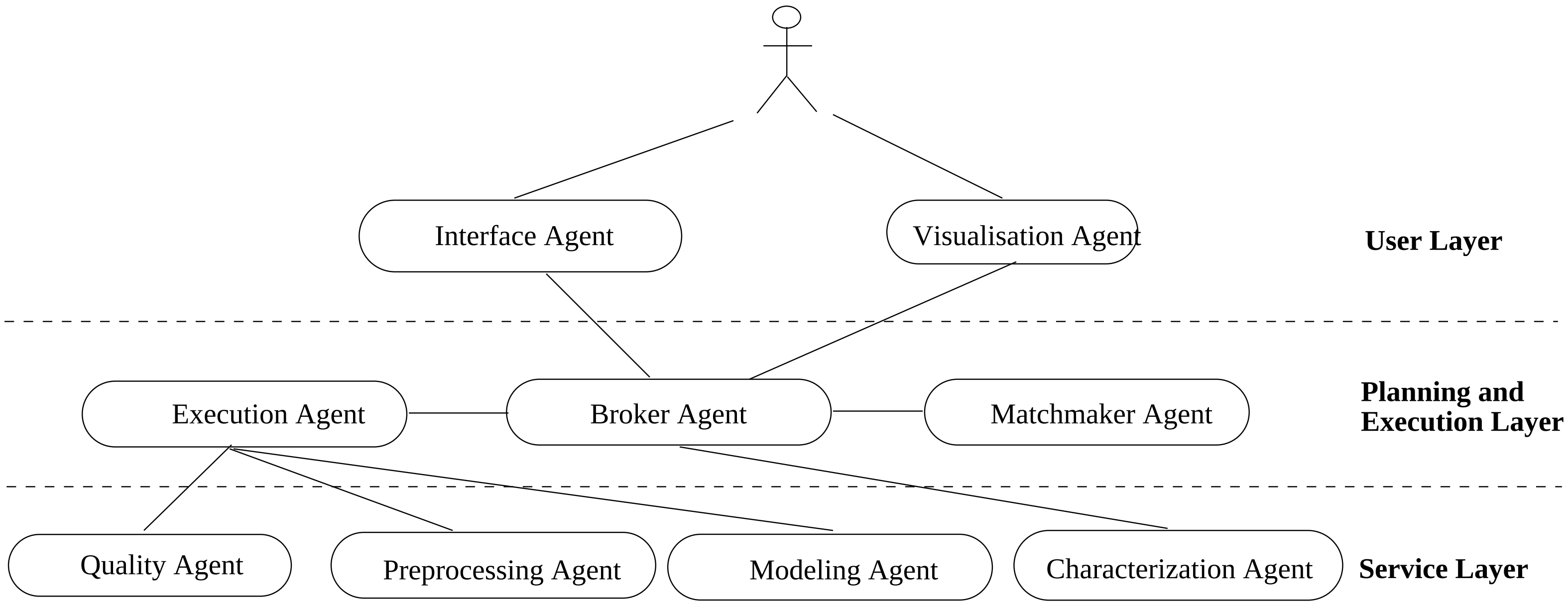
AgentDiscover

- KDD front-ends
  - METAL – offers a Web interface to assist and guide users in KDD process (Bota et al. 2001).
  - AGENT ACADEMY – a multi-agent system for design, implementation and deployment of MAS for data mining (Symeonidis & Mitkas, 2005).
  - MAGE – a middleware to build an execution engine that uses a directed acyclic graph to formalize the representation of KDD process (Z. Shi et al., 2004).
- What's new in AgentDiscover
  - Uses a task-oriented reasoning for problem solving on a multi-agent architecture MATOPS (Negru, Şandru & Pop 1998, 2001, 2005)
  - Assesses the quality of induced knowledge models using a quality model
  - Provides recommendations (scenarios) for solving for a KDD problem

Why MAS?

- Offers a knowledge infrastructure: access to latest implementations of methods and algorithms provided by agents
- Share the computational resources → problems like resource limitations, bottle-necks, system failure are infrequent in MAS
- Concurrent collaboration for problem solving → enhance the computational performances, reliability, extensibility, robustness, response time, scalability, maintainability
- Interconnection and inter-operability with legacy systems
- Offers a more ‘natural’ way of modeling the problem
- a MAS retrieves, filters and coordinates spatially distributed information between several data servers

Agent Discover Architecture



Agents Capabilities

Agent type	Capabilities
Interface	1) Displays the proposed scenarios to the user 2) Creates, modifies and removes scenarios 3) Offers a Web/Swing-based for user interaction
Visualization	1) Visualizes and outputs the knowledge model
Execution	1) Executes, controls and adjust a scenario 2) Saves the scenario
Broker	1) Decomposes the problem in sub-problems and tasks 2) Queries the Matchmaker Agent to find an appropriate agent to handle the tasks 3) Builds an Execution plan and send it to the Execution agent 4) Handles task dependencies
Matchmaker	1) Acts as a “Yellow Pages” service provider for Broker agent 2) Performs task refinements (quality of service) in order to prune the list of candidates agents
(Knowledge) Modeling	1) Builds the knowledge model 2) Saves the knowledge model
Characterization	1) Statistically analyzes the dataset and constructs the feature vector
Preprocessing	1) Selects a method and prepares the dataset for modeling task
Quality	1) Computes the quality metrics and builds the quality model

Scenario

- Scenario - defines the tasks to solve a KDD problem and the pre-conditions for it
- Composed of:
  - ordered list of tasks
  - the feature vector
  - the knowledge model
  - the quality model
- Classification:
  - Generic – tasks are described at generic level (e.g. classification, discretization)
  - Compiled – these are scenarios obtained after running the system for a particular dataset and are composed of individualized tasks (e.g. J48 classifier with a defined list of arguments)

Design and implementation details

- Agent facilitations – using a FIPA-compliant facilitator agent
- External resources (databases, file system, WEKA algorithms etc.) are accessed using transducer agents
- The feature vector contains values of three types: general values (e.g. number of attributes), statistical (e.g. average of coefficient of variation) and informational (e.g. normalized class entropy)
- Knowledge Base
  - Knowledge models are saved in PMML format (Predictive Model Markup Language)
  - Scenarios are saved in XML format
- Use WEKA implementation for DM algorithms
- Scenario classification is based on quality characteristics.
- Supported datasets format: WEKA's ARFF (relational database access is planned to be supported soon)

Testing Environment

- Training datasets: 6 UCI medical datasets: Dermatology, Hepatitis, Liver disorders, Thyroid, Lung cancer, Pima diabetes
- Testing dataset: Maternal (full-term vs. pre-term births)
- Problem type: classification
- Investigating models: a Bayesian network (BN), a multinomial logistic regression model with a ridge estimator (LRM), a multi-layer perceptron trained with backpropagation (MLP), a radial basis function network (RBF), a pruned C4.5 decision tree (J48), a decision table majority classifier (DT) and rules set (PART)

Conclusions - AgentDiscover

- MAS for KDD process with recommendations
- What has been done:
  - prototype implementation on JADE
  - first tests on medical datasets
- What's next:
  - deeply investigate the similarity between datasets
  - feature vector: complete with other metrics
  - continue with system evaluation for different types of datasets and much larger datasets: Weblog files, financial datasets etc.

- Intelligent front-end NESS
- MATOPS: Multi-Agent Architecture
- Implementing NESS in MATOPS
- AgentDiscover front-end
- Conclusions

THANKS FOR YOUR ATTENTION !

Questions ?