Agents and Web Services for Autonomic Computing

Katia P. Sycara

School of Computer Science

Carnegie Mellon University

e-mail: katia@cs.cmu.edu

URL:www.cs.cmu.edu/~softagents

Autonomic Computing

"Computing systems that manage themselves in accordance with high-level objectives from humans."

Kephart and Chess, *A Vision of Autonomic Computing*, IEEE Computer, January 2003

- Four key properties:
  - **Self-Configuration**: Automated (re-)configuration of components in response to changing situations; allows addition/removal of components without disrupting system functionality
  - **Self-Healing:** Automated detection, localization, diagnosis, and repair; proactively circumvent problems that could cause functionality disruption
  - **Self-Optimization:** Automated and continual adaptive and proactive performance monitoring and tuning
  - **Self-Protection:** Automated recognition and defense against threats, attacks or failures

Challenges for AC

The challenges are akin to the challenges of integration of business processes plus require more intelligent autonomous behavior:

Integrate multiple heterogeneous

- Data repositories; Policies; Processes; Applications

Ensure

- Performance
- Dynamism: real time access and reconfiguration works accurately
- Flexibility in self configuration: systems enter and leave
- Intelligible communication (semantic equivalence of similar concepts)
- Establish, maintain and monitor negotiated service level agreements

Three Significant Technological Visions

- Web Services (Service Oriented Architectures)
- Semantic Web
- Multi Agent Systems (MAS)

Web Services - A New Paradigm?

- Web Services heralded as:
  "… self-contained, self-describing, modular applications that can be published, located, and invoked across the Web…"
- Which will allow…
  - …on the fly composition of new functionality through the use of loosely coupled reusable software components
  - …decomposition and distribution of large-scale processing tasks into component tasks executed simultaneously across many devices

"Web services are expected to revolutionize our life in much the same way as the Internet has during the past decade or so." (Gartner)

Service Oriented Computing

- Encapsulation
- Autonomy
- Distribution
- Interaction via messaging
- Interoperability
  - Through common vocabularies
  - Through ontologies for objects and processes

AC is service-based and the WSDM (WS Distributed Management) standard relies on current WS standards

Current State: Web Services Standards

- SOAP: XML based web services communication protocol

Limitations

- Unbounded message format
- Has no communicative speech acts (cannot determine intention of actors or type of the message)
- WSDL: Structured mechanism to describe a WS interface

Limitations

  - No semantics for message sequencing and correlation
  - No semantics for message content

- BPEL: Description of how Web Services are composed

Limitations

  - No IOPEs
  - Allows execution of a manually constructed composition

- UDDI: Directory Service for Web Services

Limitations: keyword searches, limited capability search

Current standards promote interoperability by providing common vocabularies and syntax: good first step

Tackling Semantic Interoperability…

- Current Web Services standards lack formal semantics
- Lack of Semantic Interoperability is a major hurdle for
  - **Discovery**
    - Different terms used for advertisements and requests
  - **Invocation**
    - Different specs for messages and WS interface
  - **Understanding**
    - Interpreting the results returned by the Web service
  - **Monitoring and Repair**
    - Understanding heterogeneous components faults
  - **Composing Services**
    - Reconciling private goals with goals of the WS
  - **Negotiating service agreements, contracts & communications**
    - Different terminology and protocols used

Is this a real problem?

World Wide Annual Integration plus Data Quality Costs: $1 Trillion / year

"The problem is not in the plumbing. It's in the semantics"

(quotation from Michael Brodie's invited talk at ISWC 2003)

Note: some standards committees, e.g. WSDL and UDDI start realizing this truth and planning to incorporate RDF and OWL in these standards

Semantic Web

- Goal of Web Semantics: Development of common representation format, shared ontologies, reasoning mechanisms, and query engines to support improved utilization of Web knowledge
- Semantic Web Standards
  - Rdf
  - OWL: based on Description Logic

From the W3C Semantic Web Activity statement

"The Semantic Web is an extension of the current Web in which information is given a well-defined meaning, better enabling computers and people to work in cooperation. It is the idea of having data on the Web defined and linked in a way that it can be used for more effective discovery, automation, integration and reuse across various applications. The Web can reach its full potential if it becomes a place where data can be processed by automated tools as well as people"

Semantic Web: semantic metadata and ontologies for web content to enable information access, integration, interoperation and consistency.

Semantic Web Services

OWL-S Ontology

- OWL-S is an OWL ontology to describe Web services
- OWL-S has been accepted as a Note by W3C (12/1/2004)
- OWL-S leverages on OWL to
  - Support capability based discovery of Web services
  - Support automatic composition of Web services
  - Support automatic invocation of Web services
  - Support monitoring of the execution of Web services

Complete do not compete

- OWL-S does not aim to replace the Web services standards rather OWL-S attempts to provide a semantic layer
  - OWL-S relies on WSDL for Web service invocation *(see Grounding)*
  - OWL-s Expands UDDI for Web service discovery *(OWL-S/UDDI mapping)*

http://www.daml.org/services/owl-s/

Intelligent Software Agents

Semantic Web Services

Acknowledgements to the **OWL-S Web Services Coalition**

BBN: Mark Burstein

CMU: Katia Sycara, Massimo Paolucci, Naveen Srinivasan

De Montfort University: Monika Solanki

ISI: Jerry Hobbs

U. Of Maryland: Bijan Parsia

Nokia: Ora Lassila

Southampton: Terry Payne

Stanford KSL: Deborah McGuiness

SRI: David Martin

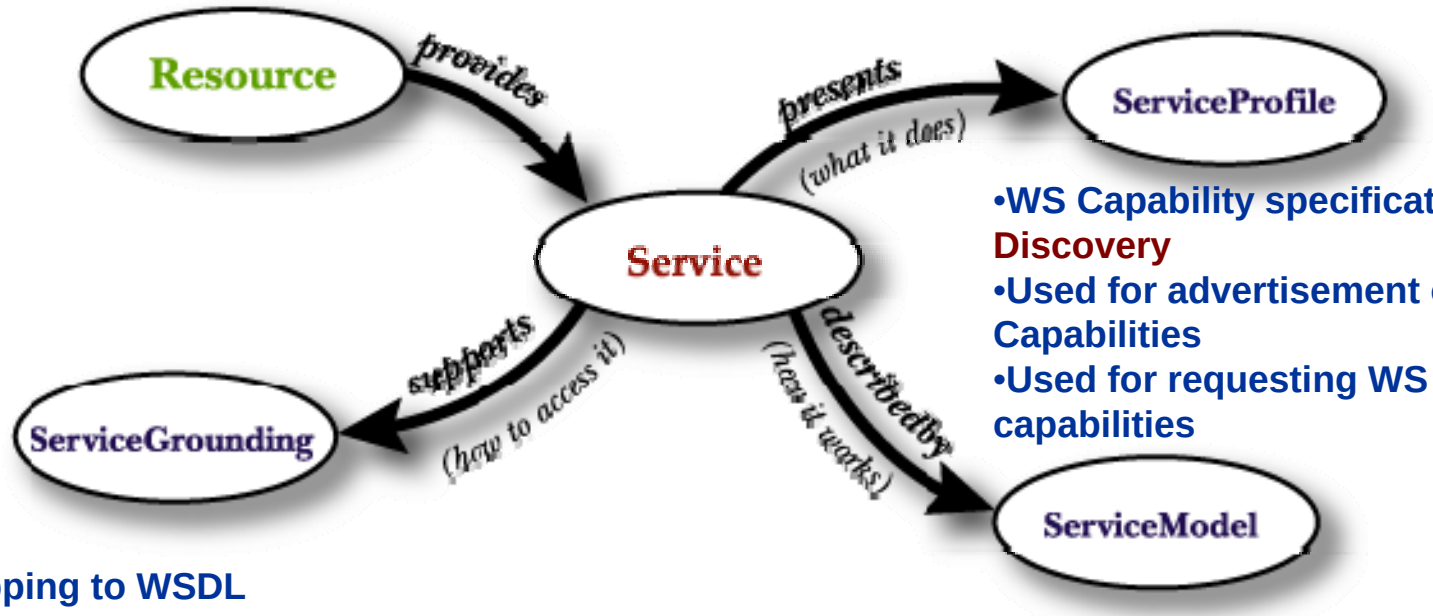
U. Of Toronto: Sheila McIlraith

Vrije Universiteit: Marta Sabou

Yale: Drew McDermott

Semantic Web Services

OWL-S Upper Ontology



- **WS Capability specification to assist Discovery**
- **Used for advertisement of service Capabilities**
- **Used for requesting WS with desired capabilities**

- **Mapping to WSDL**
  - **communication protocol (RPC, HTTP, …)**
  - **marshalling/serialization**
  - **transformation to and from XSD to OWL**

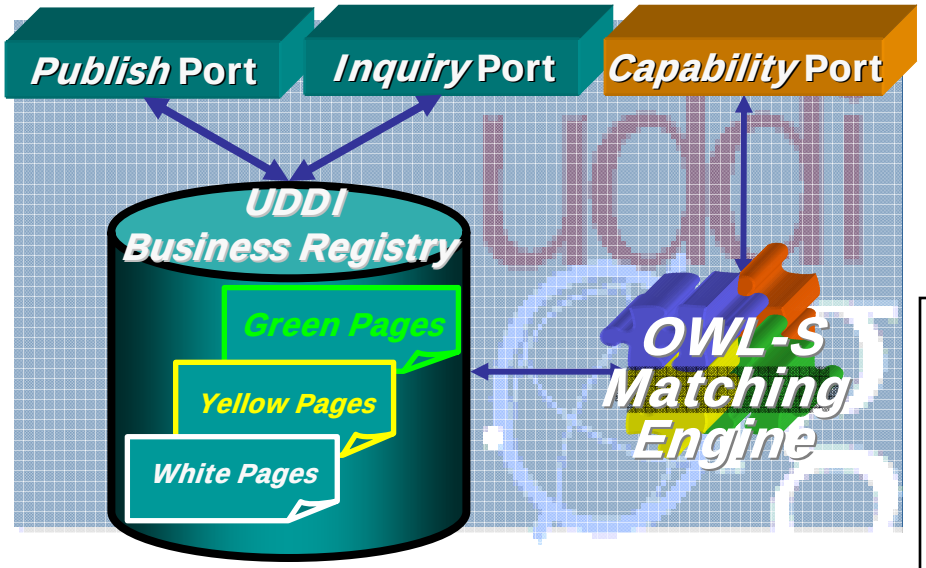
- **Control flow of the service**
  - **Black/Grey/Glass Box view**
- **Protocol Specification**
- **Abstract Messages**

Semantic Web Services

Capability Description

- **Preconditions**
  - Set of conditions that should hold prior to service invocation
- **Inputs**
  - Set of necessary inputs that the requester should provide to invoke the service
- **Outputs**
  - Results that the requester should expect after interaction with the service provider is completed
- **Effects**
  - Set of statements that should hold true if the service is invoked successfully.
- **Service type**
  - What kind of service is provided (eg selling vs distribution)
- **Product**
  - Product associated with the service (eg travel vs books vs auto parts)

Integration of CMU OWL-S Matchmaker and UDDI



- OWL-S Profile has been mapped to UDDI data structure
  - OWL-S Web services can be advertised in UDDI as any other Web service

(see Paolucci et al 2002)

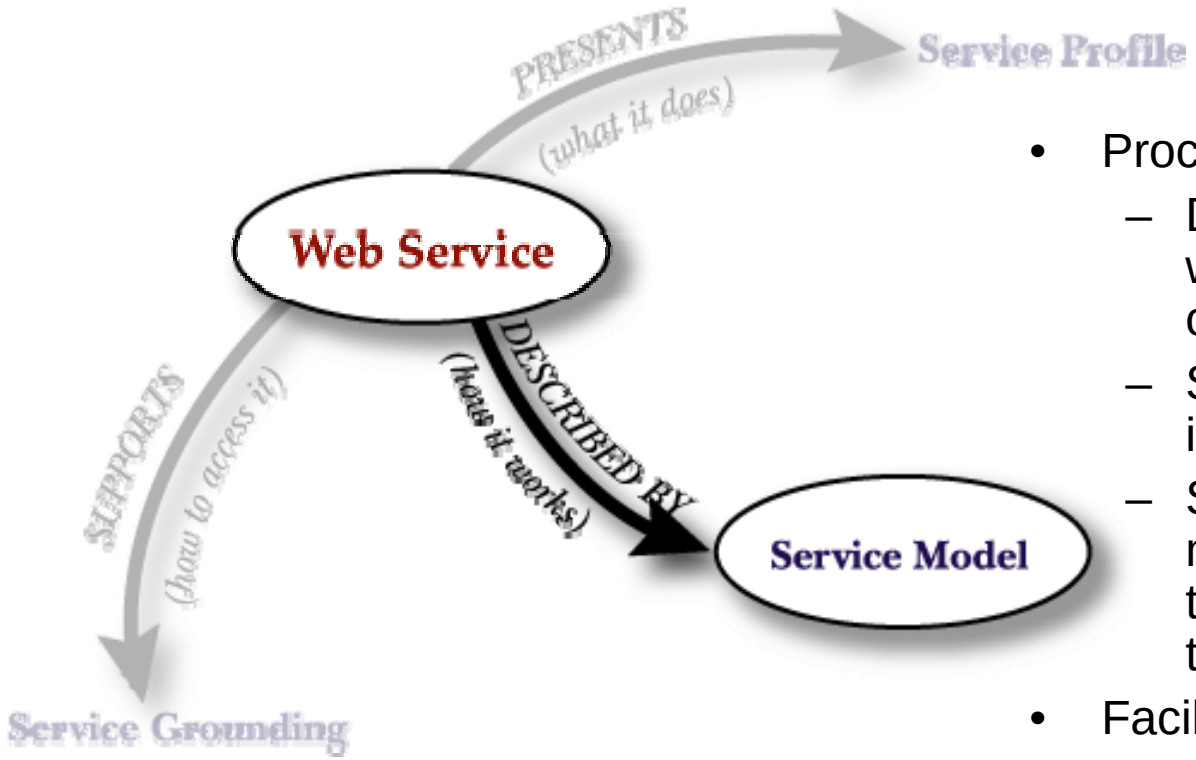
- CMU OWL-S Matching engine has been integrated within UDDI server
- CMU UDDI server provides
  - Normal UDDI Publish/Inquiry ports
    - Complete interoperability with any UDDI Client
  - Capability Port provides OWL-S based capability requests

(see Srinivasan et al 2004)

CMU UDDI is publicly available at
www.daml.ri.cmu.edu/matchmaker
or on SemWebCentral
www.semwebcentral.org
A variant of the CMU UDDI is in use at the NTT UDDI Business Registry (The main public UDDI in Japan) (see Kawamura et al 2003, 2004)

Semantic Web Services

Process Model



- Process Model
  - Describes how a service works: internal processes of the service
  - Specifies service interaction protocol
  - Specifies abstract messages: ontological type of information transmitted
- Facilitates
  - Web service invocation
  - Composition of Web services
  - Monitoring of interaction

OWL-S Definition of Process

A Process represents a transformation (function). It is characterized by four parameters

- ***Inputs***: the inputs that the process requires
- ***Preconditions***: the conditions that are required for the process to run correctly
- ***Results***: a process may have different outcomes depending on some condition
  - ***Condition***: under what condition the result occurs
  - ***Outputs***: the data that results from the execution of the process
  - ***Effects***: real world changes resulting from the execution of the process

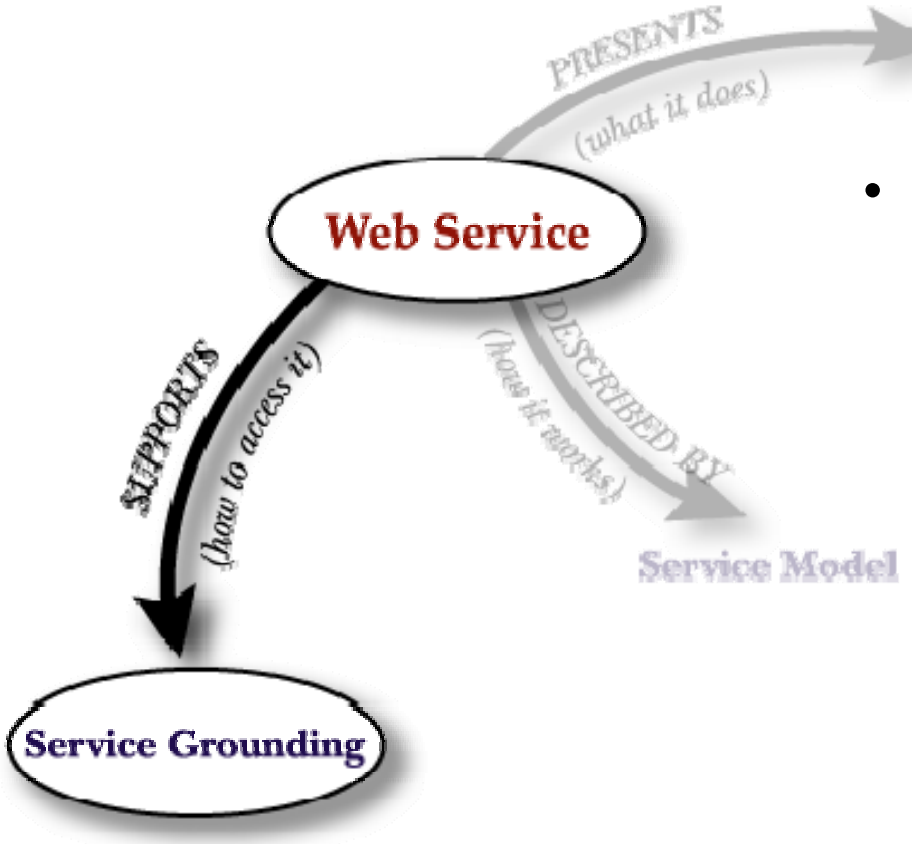
- Composite processes have

  Control Flow (sequence, conditional, loops, ND-choice)

  Date Flow (output-> input, input->input, output-> output)

Semantic Web Services

Service Grounding



- Service Grounding
  - Provides a specification of service access information.
  - Service Model + Grounding give everything needed for using the service
  - Builds upon **WSDL** to define message structure and physical binding layer

Mapping OWL-S to WSDL

- OWL-S invocation is based on the Grounding
  - Map atomic processes into WSDL operations
  - Use XSLT to map between XML Schema data structures and Ontological Information
  - Invocation procedure totally separated from semantic description of Web service
    - Invocation may be modified without changing semantic description
  - Any Web service can be described in OWL-S without modifying the WSDL description of the service
    - Amazon's Web service has been described in OWL-S maintaining Amazon's XML-Schema data types

Problem of Composition

- No single Web service may achieve all goals of an agent
  - Composition is the process of chaining results from different Web services automatically
- Planning problem
  - How do the Web services fit together?
- Interoperation problem
  - How does the information returned fit together?

- **CMU Composition Architecture**: Exploits Retsina Architecture for WS composition
  - OWL-S/UDDI Matchmaker for discovery
  - Retsina planner to control the agent
    - Use interleaving of planning and execution to allow communication while planning
  - OWL Reasoner
  - OWL-S Virtual Machine to communicate and invoke Web Services

Katia Sycara, Massimo Paolucci, Anupriya Ankolekar and Naveen Srinivasan, "**Automated Discovery, Interaction and Composition of Semantic Web services**," *Journal of Web Semantics*, Volume 1, Issue 1, September 2003, pp. 27-46

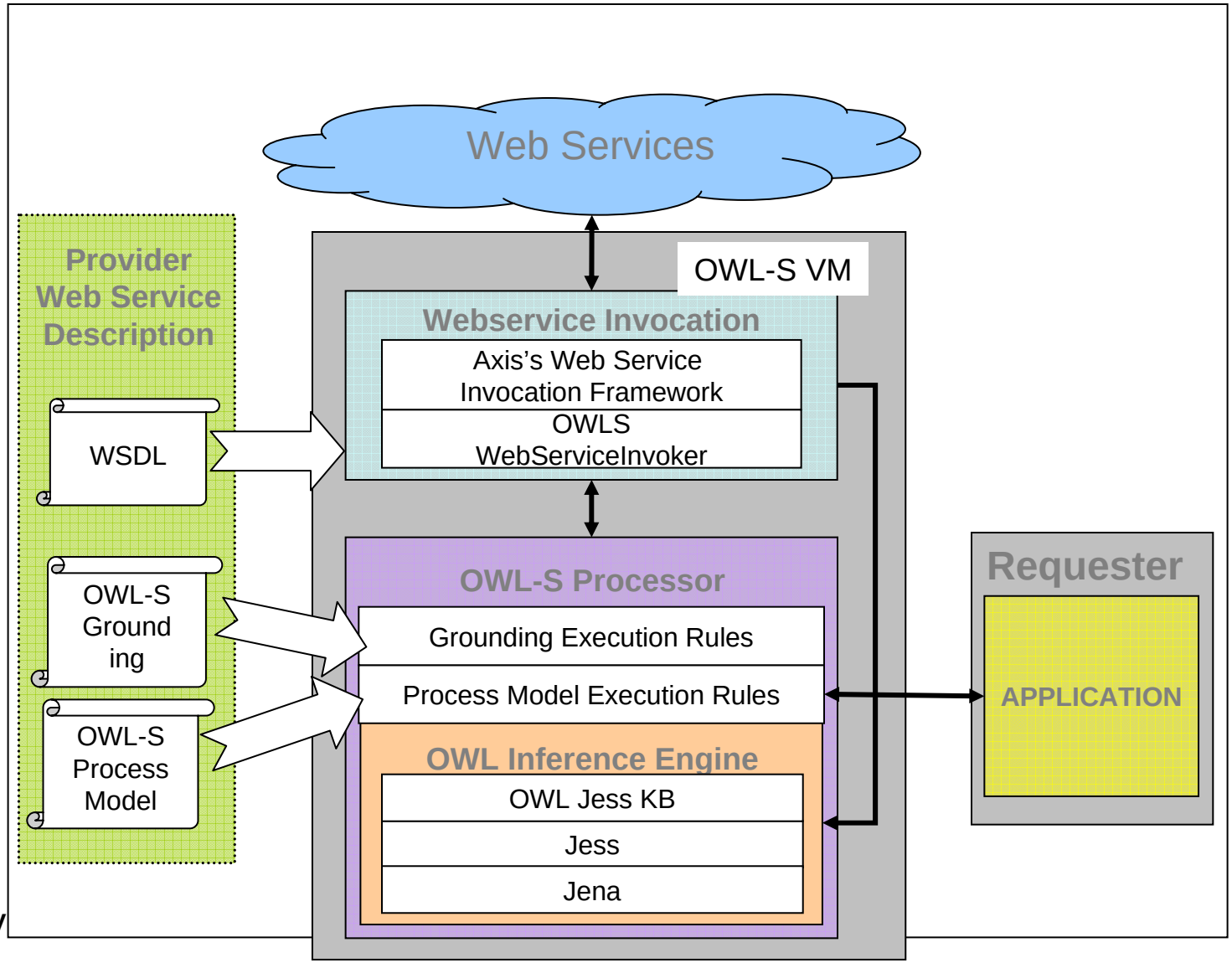
OWL-S Virtual Machine

- OWL-S VM a generic processor for the OWL-S Process Model
  - Uses OWL-S to represent service descriptions
  - It can interact with any OWL-S Web service
  - Based on the Process Model formal semantics (Ankolekar et al 2002)
  - Implements grounding mapping to WSDL
  - Uses OWL to represent information to exchange between Web services
  - Actively adopts logic inference to reason about OWL-S and OWL ontologies
  - Exploits Web services technology such as Axis and WSIF for actual invocation and message exchange

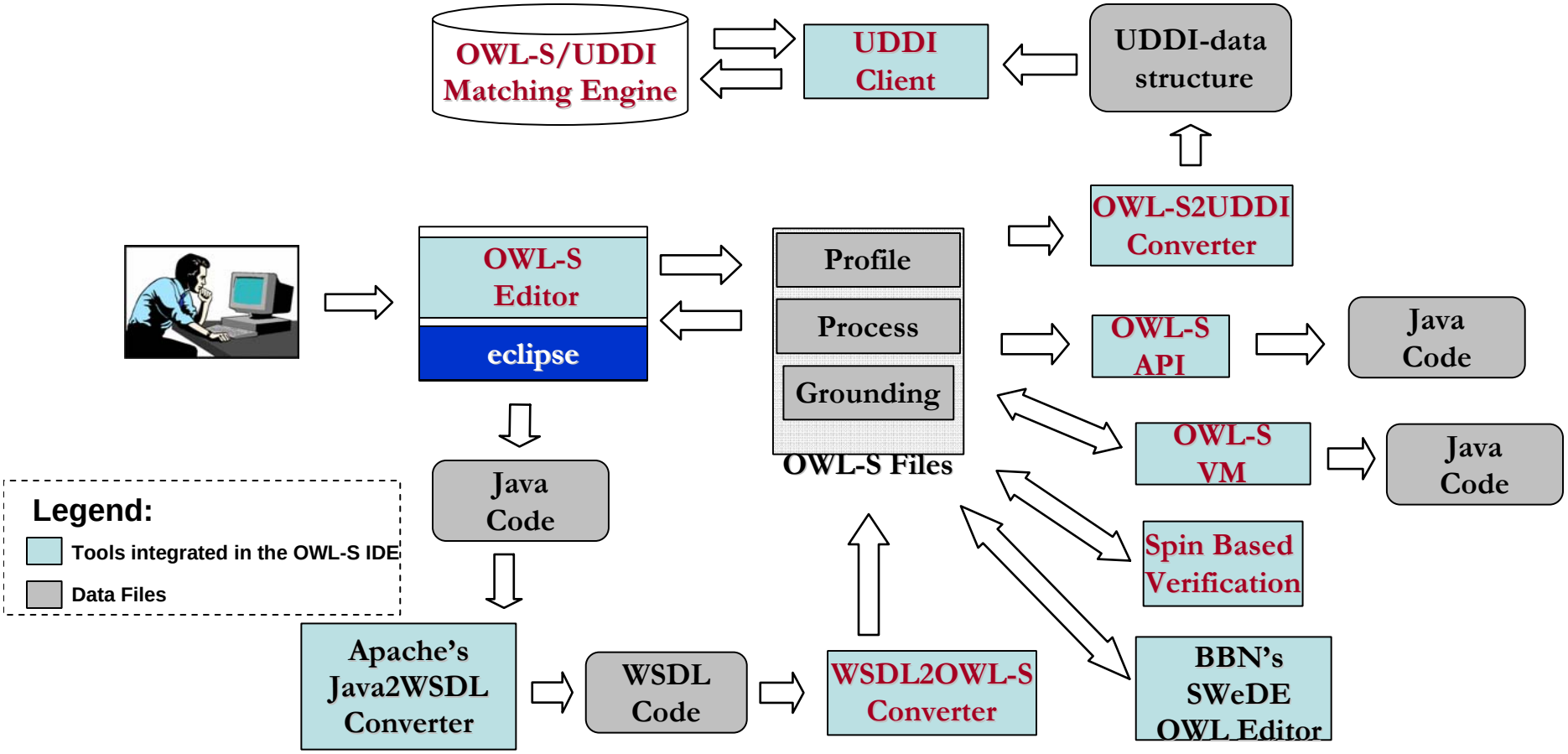
Anupriya Ankolekar, Frank Huch, Katia Sycara. "**Concurrent Execution Semantics for DAML-S with Subtypes**." In *The First International Semantic Web Conference (ISWC)*, 2002.

Massimo Paolucci, Anupriya Ankolekar, Naveen Srinivasan and Katia Sycara, "The DAML-S Virtual Machine," In *Proceedings of the Second International Semantic Web Conference (ISWC)*, 2003,

Architecture of CMU OWL-S VM



CMU OWL-S IDE (Eclipse based)



Agent Characteristics

An agent is a computational entity which is capable (to various degrees of):

Autonomy: agent is goal-directed and has control over its internal state (can reason and act without direct intervention of others)

Interactivity/Sociability: agents interact other agents (including humans) in pursuit of their goals (individual or collective)

Adaptivity: agents respond reactively or proactively to changes in their environment and behavior of other agents

Situatedness: agent receives sensory inputs from its environment and outputs actions (physical environments, Internet)

MAS Coordination

Teamwork: Agents share a common set of goals and each contributes to the fulfillment of these goals through teamwork. No explicit modeling of individual agent utility

Capability-based: Agents discover others with desired functionality to form non-permanent ad hoc groups to solve a current problem; group members may belong to one or more groups simultaneously

Coalitions: Agents seek to maximize individual utility and group utility (coalition stability is an issue)

Coordination: Agents pursue their individual goals and utilities; coordination with others is done only to avoid harmful interactions (e.g. traffic)

Auctions: Agents seek to maximize utility; agents interact through centralized auctioneer.

Negotiation: Agents seek to maximize their individual utility but are willing to compromise; agents interact directly

Game Theoretic Interactions: Agents seek to maximize individual utility while taking into consideration other's options

Adversarial Interactions/Zero Sum Games: Agents seek to maximize own utility while minimizing utility of opponent

There is no single coordination technique that fits all applications

Current IT Systems: sources of complexity

- Large number of agents/components, typically in the thousands
- Large number of asynchronous interactions
- Distributed over widely geographical areas
- Operate in uncertain and changing environments
- Components have limited viewpoint on system behavior
- Components are heterogeneous
- Lack of global control
- Open system: system structure is dynamically changing (information sources, languages, ontologies, communication links, components dynamically vary over time)

These are the characteristics of many Large Multi-Agent Systems

Desiderata of Overall System Behavior

- Flexibility
- Extensibility
- Stability
- Invulnerability
- Performance: As number of agents (and agent interactions) increase, the system performance (e.g. system responsiveness, reasonable bandwidth consumption) should not deteriorate
- Manageability: As the system scales up it should not require considerable increase in administrative complexity

Challenges of Large MAS (AC)

- Unpredictability of Overall MAS Behavior (e.g. chaotic or oscillatory behavior)
- Not possible to verify overall system correctness
- Not possible to determine optimality
- System anomalies and vulnerabilities
- Dealing with agent heterogeneity (e.g. need for intelligibility)
- Scaling up discovery of relevant agents to coordinate with to solve given problems (accomplish goals)
- Density and asynchrony of communications
- Software Engineering considerations (interoperability, platform independence, QoS)

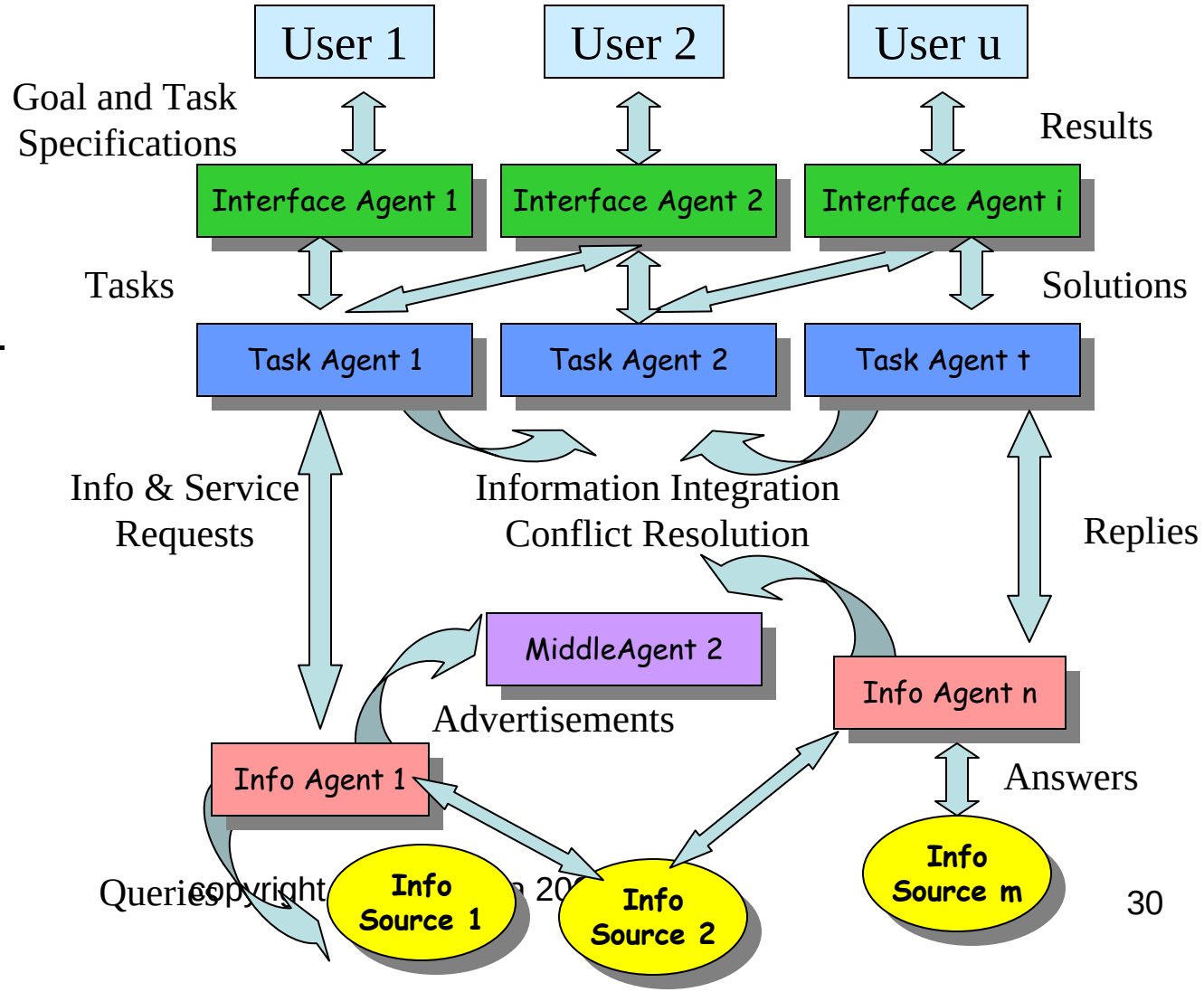
Generic Tasks in Open Environments

Agents must be able to:

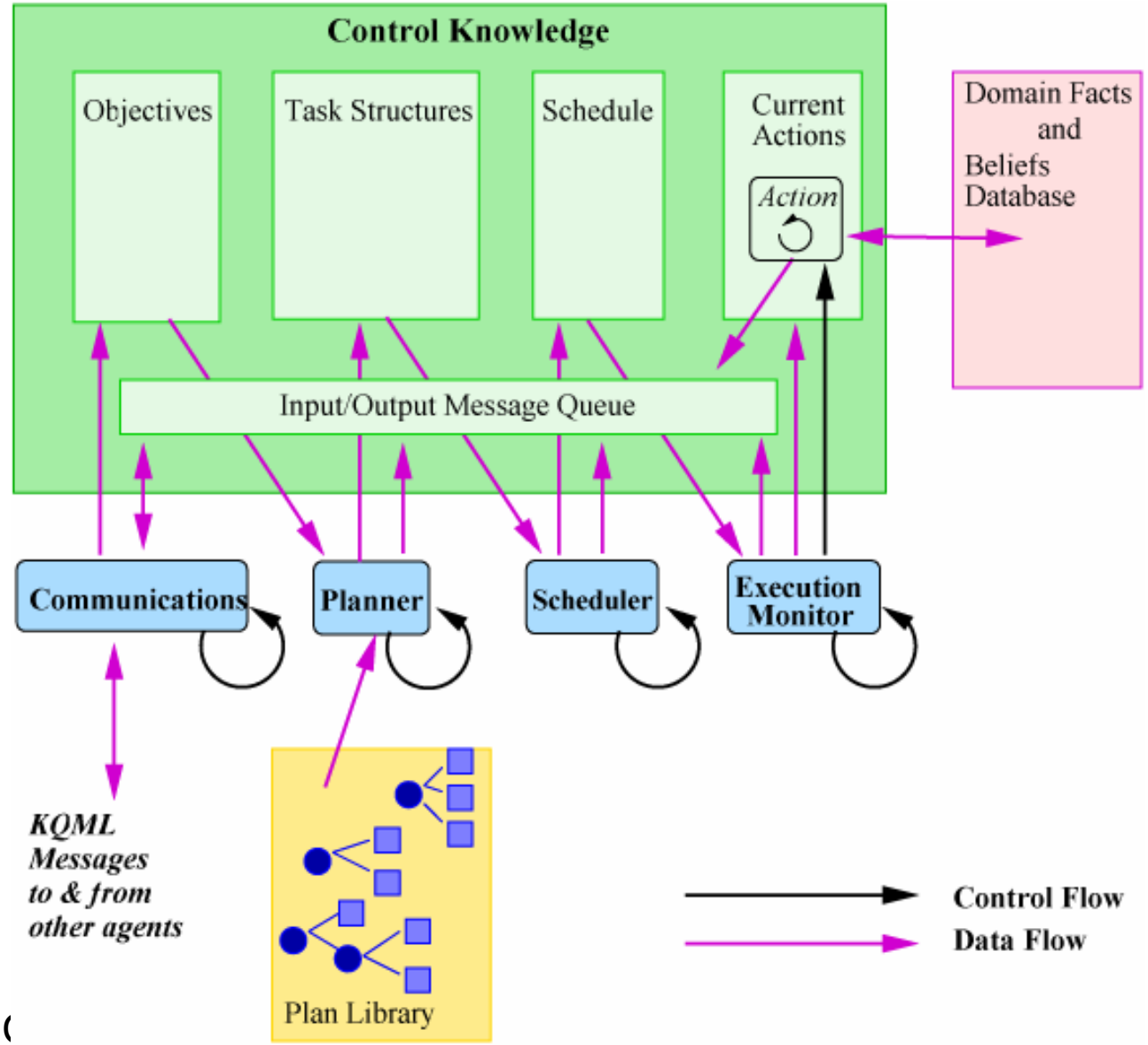
- **discover** each other in order to interact in cooperative or self-interested fashion.
- **allocate tasks and resources**
- **interact/transact** with each other
- **reason** about own and other's tasks
- **compose** results of their reasoning
- **adapt** to the overall situation
- **monitor** progress of delegated tasks
- **form and learn** models of others
- **learn** from their interactions
- **semantically interoperate** with one another

The RETSINA Multi-Agent Architecture

distributed adaptive collections of agents that self organize and self-monitor based on environmental stimuli



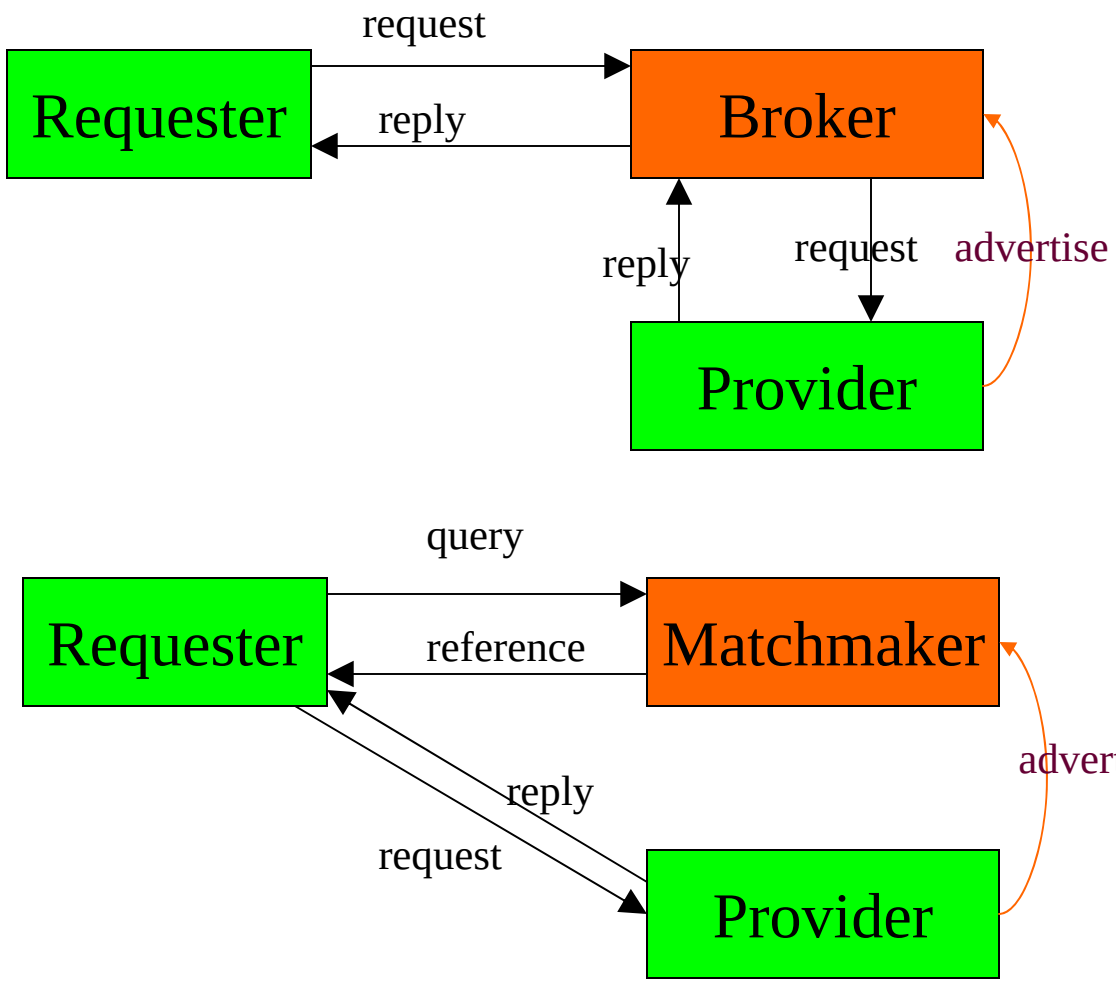
RETSINA Single Agent Architecture



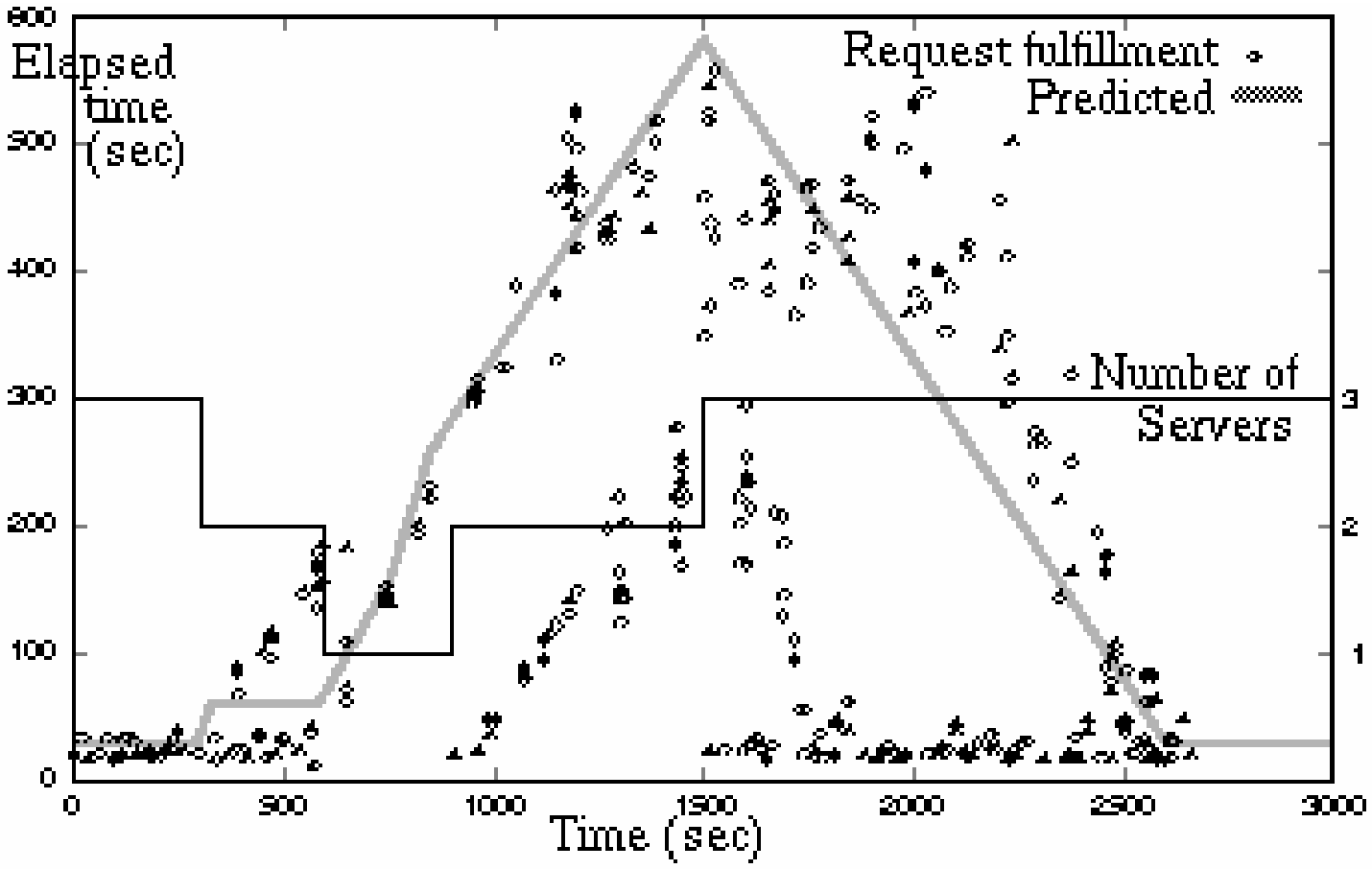
RETSINA - Discovery and Interoperation

- Agents that provide services advertise their capabilities to middle agents
  - Advertisements include functional parameters and non-functional/performance parameters (e.g. availability, response time ..)
  - The non functional parameters are learned by the agents over time through self-monitoring
- Requester agents ask middle agents for agents with particular capabilities
- Middle agents match requests to advertisements and return results
- Communication protocols include formal semantics and ontologies for interoperation
- A2A (Agent to Agent) is an addition to RETSINA to allow discovery of infrastructure (Middle agents and Agent Name Servers) using peer to peer techniques.
- We have identified 28 types of middle agents with different functionalities, protocols and performance tradeoffs [ICMAS 2000]

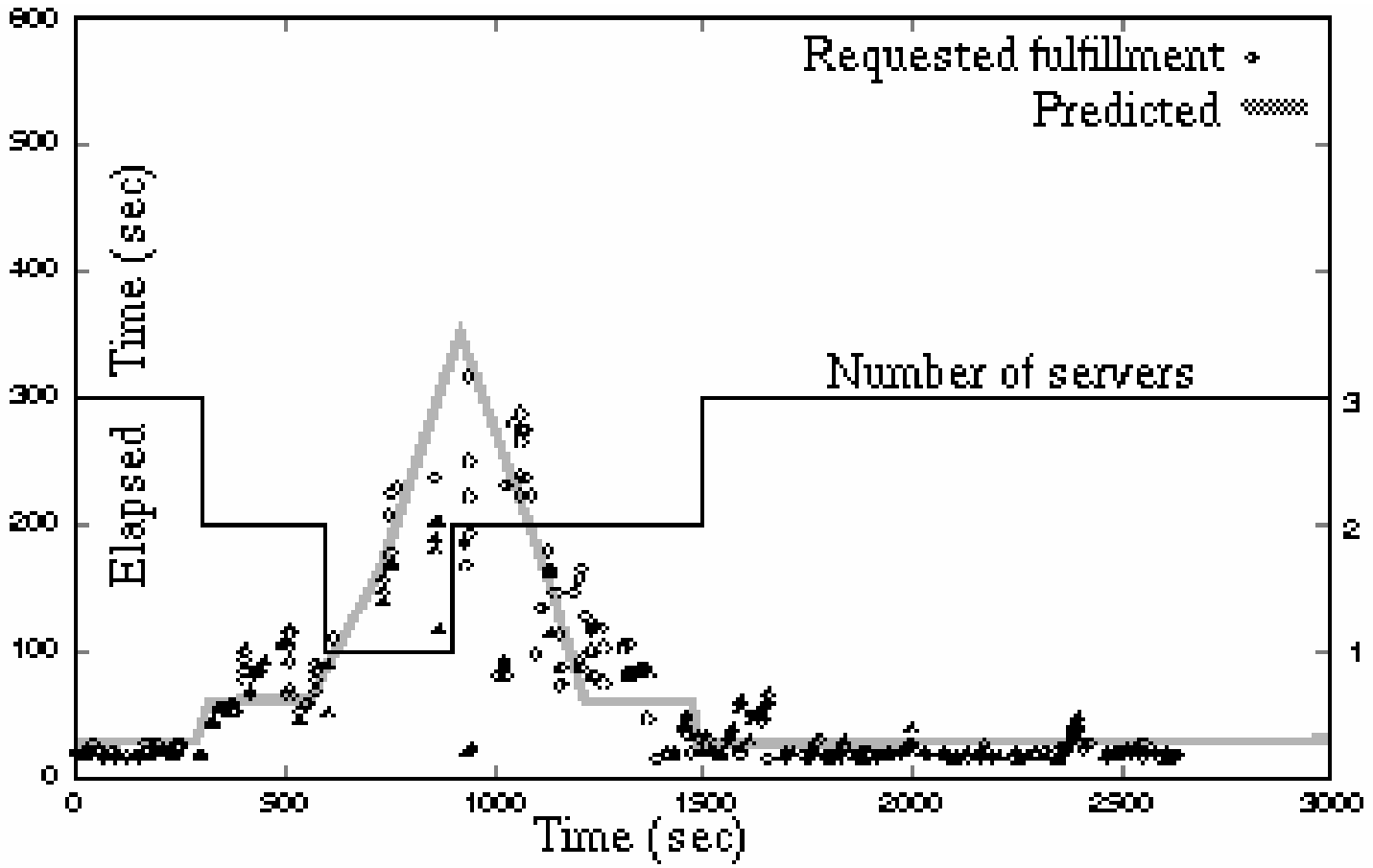
RETSINA – Example Middle Agents



Performance of Match-made System



Performance of Brokered System



Self-Cloning

- Agents that perceive an overload look for other agents to pass tasks to (simple model to predict idle time using learned estimation of task durations)
- When other agents not found:
  - locate a host with resources for cloning
  - create a clone on the host
  - partition tasks and transfer to clone
  - the ``old'' agent updates its advertisement; the ``clone'' agent advertises
- When clone is idle -- consider self-extinction, and shut down if necessary.
- Experimental results comparing systems with and without cloning showed a 25% increase in task completion

Shehory, Sycara, Chalasani, Jha, “Increasing Resource Utilization and Task Performance by Agent Cloning”, Lecture Notes in Artificial Intelligence, Intelligent Agent 5, Rao, Singh, and Woolridge (Eds.) 1999

Large Scale Capability-Based Coordination

- Discovery in open environments (middle agents or p2p)
- Differential coordination (large scale discovery, close interaction once discovered)
  - Agents interact with each other directly, if known to each other a priori
  - Agents may interact through middle agents for various reasons:
    - privacy issues (anonymizing of requesters and providers)
    - trust issues (enforcement of honesty)
- Loose Coupling
- Distributed intelligence reduces vulnerability
- Dynamic self-assembly of agents based on agent and system goals
- Able to deal with heterogeneity
- Functional substitutability contributes to system robustness

[JAAMAS 02] Sycara, K., Klusch, M. Widoff, S. and Lu, J. "LARKS: Dynamic Matchmaking among Heterogeneous Agents in Cyberspace", Journal of *Autonomous Agents and Multiagent Systems,* vol 5, no. 2, July 2002.

[JAAMAS 03] Sycara, K., Paolucci, M., Van Velsen, M. and Giampapa, J. "The RETSINA Multiagent Infrastructure", Journal of Autonomous Agents and Multiagent Systems, vol. 7, nos. 1-2, July/September, 2003.

Large Scale Teamwork

- Teamwork implies closer coupling than capability-based coordination
- Large number of agents in dynamic, open and hostile environments
  - Limited communication channels
  - Local information
  - High uncertainty, high dynamics, complex environment
  - High “failure” rate
  - Impractical to centralize
  - Impractical to have strict organizational structure due to frequency of agent disappearance
- Maintaining team status model
  - Prohibitive volume of communication
  - Can we limit what needs to be known by others?
- Existing approaches only work for small teams (10s)
  - Require accurate models of team activities or a centralized information broker
- Key coordination algorithms are typically NP-Complete (or worse)
  - Can we build scalable, generic algorithms?

Reusable Teamwork Algorithms

- Team member represented in team by a *proxy*
  - Proxy represents that team member in coordination
- Proxies encapsulate generic teamwork algorithms
  - Generic execution of domain specific team plans
- Each team member has a model of teamwork in a TOP template, and a model of team status
  - Communication keeps models of team status in sync
  - Use the model to decide what actions to take
- A team member has a *role(s)* within the team
  - Roles provide bounds on required reasoning
  - How can role allocation be done efficiently in Large Scale MAS?
- Information is locally sensed
  - How to do information sharing efficiently?
- Domains: large scale crisis response

Overview of Approach

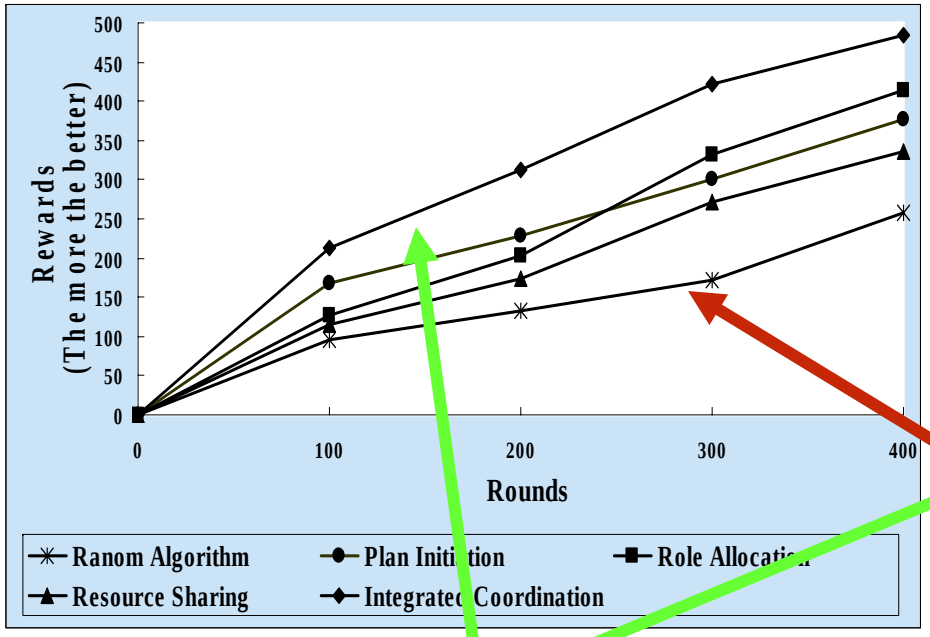
- Tokens for scalable algorithms
  - Software engineering and algorithmic abstraction
  - Self contained packets of information, routed to neighbors
  - A single token for each piece of information, role, resource
  - Movement of tokens around team implements coordination
  - Team members use receipt of tokens to create *local models* of other team members
  - Local models are used to improve the routing of future tokens
- Associates network, “small worlds” property
  - Connects entire team in static, task independent logical network
- Highly scalable token-based implementation of key algorithms
  - Shown coordination of 500+ agents in low fidelity simulation

Important Algorithms

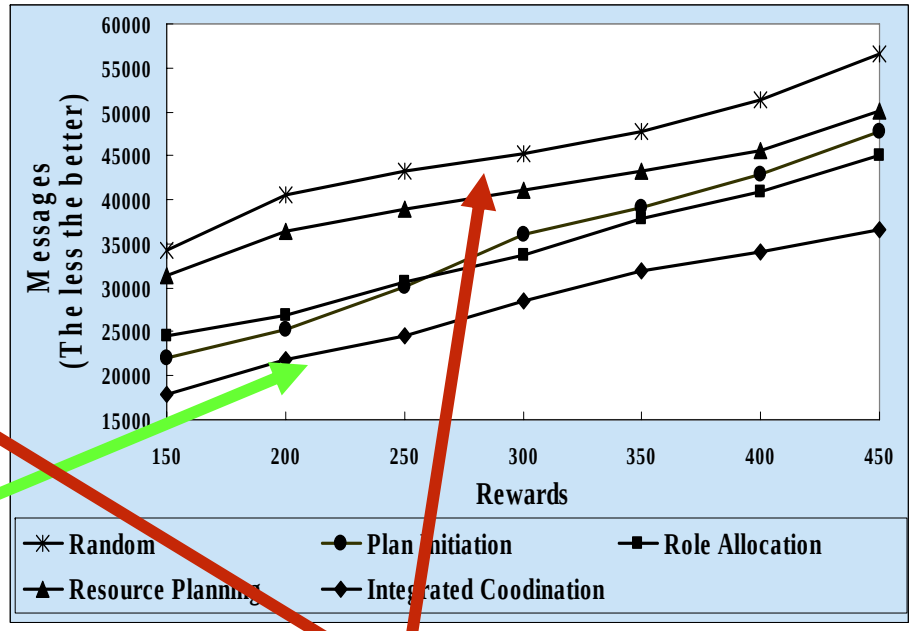
- Role Allocation
  - Uses probability distribution over team state to estimate “high quality” allocation then finds it
    - LA-DCOP has several orders of magnitude less comms than competing approaches (AAMAS’05)
- Sensor Fusion
  - Forward sensor readings to others that might have similar readings X increase in probability of fusion in large networks (AAMAS’06)
- Information Sharing
  - Build simple models of team mates, to probabilistically route information tokens
    - “Small Worlds Principle” (AAMAS’05)
- Movement of tokens for one aspect of coordination can inform routing of tokens for another aspect of coordination
  - Doubles performance (AAMAS’05 short-listed for best paper)

Results

Total Reward



Messages



Fully Integrated

Random

Analysis for Large Scale MAS (AC Systems)

- Operation state space of AC Systems systems is huge
  - Often exponential in number of operating points for individual computational nodes
  - Cannot characterize all possible node interactions
    - Undesirable to do so
- **Need formal methods to analyze AC systems to examine, e.g.**
  - Stability
  - Scalability
  - Invulnerability
- Simulation results are insufficient
  - Simulation unlikely to enter vulnerable zones in system operation state space

Solution

- Consider analysis in the system design process
  - Design function that gives valuation for system states
  - Design probability distribution over valuation function
  - Use probability distribution to calculate expectations over macroscopic system variables

Statistical Mechanics

- Provides mathematical tools for the study of complex large scale systems
- Specific tools
  - Potts Model
  - Percolation Theory
- Provide formal guarantees concerning:
  - System dynamics
  - Emergent behavior
- These in turn relate to system properties such as:
  - Stability
  - Scalability
  - Degree of vulnerability

Potts Model of Complex Systems

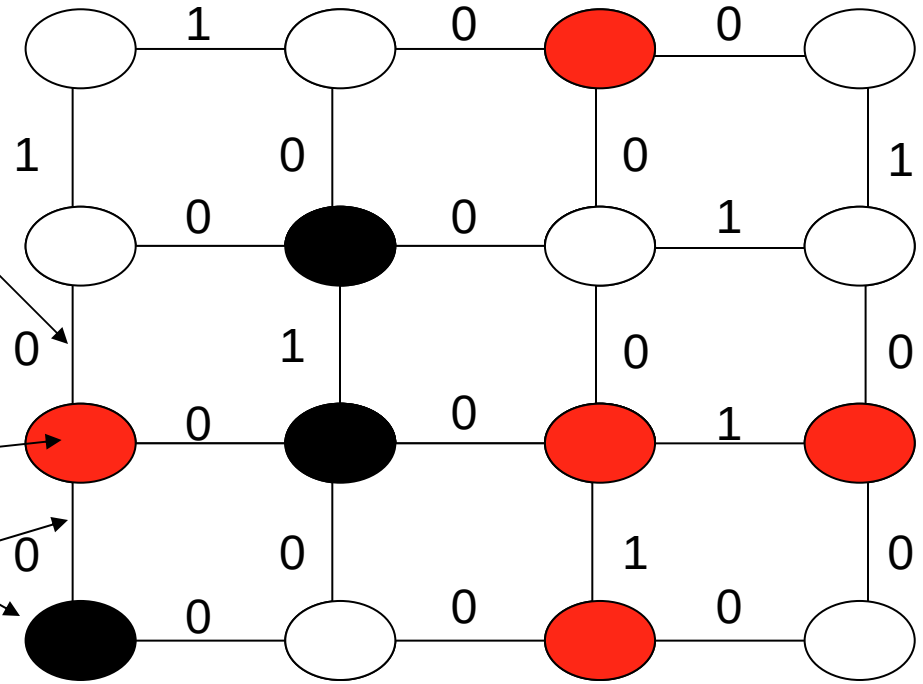
$$H(S) = \sum_{edges} -J\delta_{s_i, s_j} + h\sum_{nodes} \delta_{s_i, s\partial}$$

Hamiltonian gives measure of macroscopic system properties

Bond energy: measure of constraints on node interaction

State of individual computational nodes

Communication Channel



Hamiltonian/Energy Function

Hamiltonian gives measure of macroscopic system properties

Magnetization term captures constraints on node state

Bond energy measure of constraints on node interaction

$$H(S) = \sum_{edges} -J\delta_{s_i,s_j} + h\sum_{nodes}\delta_{s_i,s\partial}$$

$$\delta_{s_i,s_j} = \begin{cases} 1 & s_i = s_j \\ 0 & s_i \neq s_j \end{cases}$$

Potts Model for System Analysis

- *P(S)* the probability of a particular global state *S* is given follows Boltzman distribution:
  - Temperature *T* is a measure of rate of node interaction

$$P(S) = \frac{\exp\left(-\frac{1}{kT}H(S)\right)}{\sum_S \exp\left(-\frac{1}{kT}H(S)\right)}$$

Potts Model for System Analysis

- Can approximate *P(S)* to calculate expectations of global system properties
  - Swendsen-Wang algorithm
- Can answer questions like:
  - How likely is a state *Sc* where macroscopic system properties are outside desirable operation zones
  - What system temperature results in desired operation range for macroscopic system variables?
  - Is this a stable state?
    - Characterized by low energy (high probability)

Example System Safety Monitoring

- Safety monitoring
  - E.g. Power plant monitoring
  - Distributed computational nodes monitor safety variables
  - Each node has uncertain knowledge of a subset of variables
  - Nodes must fuse and filter variables to achieve consensus on system state
    - Detect unsafe states

Example System

$A = \{a_1, \ldots, a_m\}$ Set of computational nodes

$X(t) = \{x_1(t), \ldots, x_n(t)\}$ Computational nodes have state vector to describe system operation

$P^i(X(t), t)$ Computational nodes maintain probability distribution over system state vector

$\Delta^i(X, P^i(X(t), t))$ Divergence metric of difference between belief and true system state

$P^{i'}(X(t), t) = f(P^i(X(t), t), r)$ Filter to incorporate sensor readings

$$\sum_{a_i \in A} \int_{t=0}^{T} C(a_i, \Delta^i(\bullet)).dt + CommCost$$ optimization function

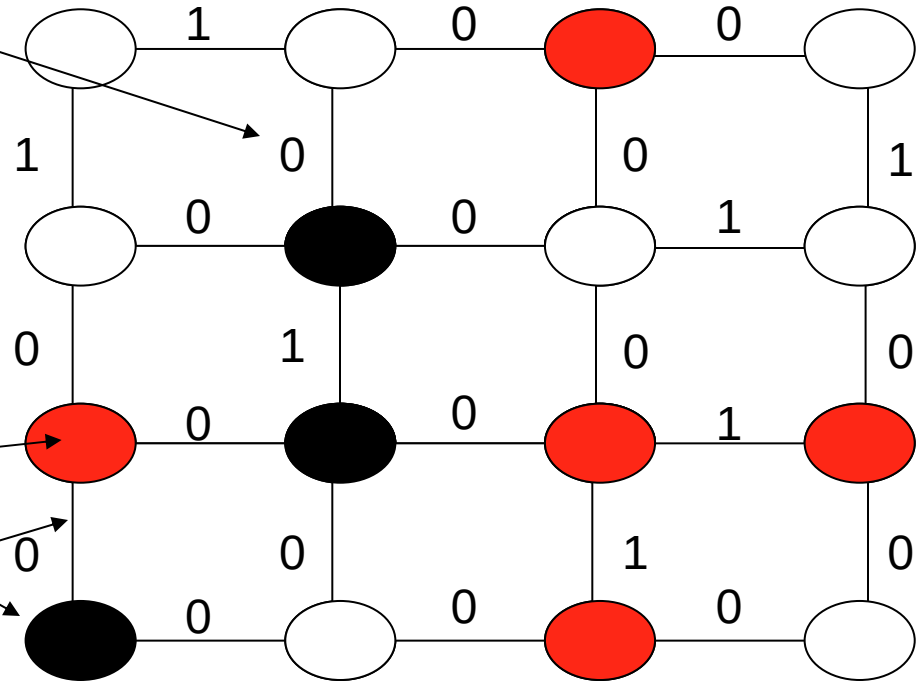
Wish to minimize divergence of computational nodes belief from true system state over time

Pott's Model Example Consensus Formation

Bond energy measure of degree of opinion/belief similarity between neighbors (KL-divergence)

$$H(S) = \sum_{edges} -J\delta_{s_i, s_j} = -7J$$

Hamiltonian gives measure of total system energy/ degree of global consensus



State/belief of individual agents

Communication Channel

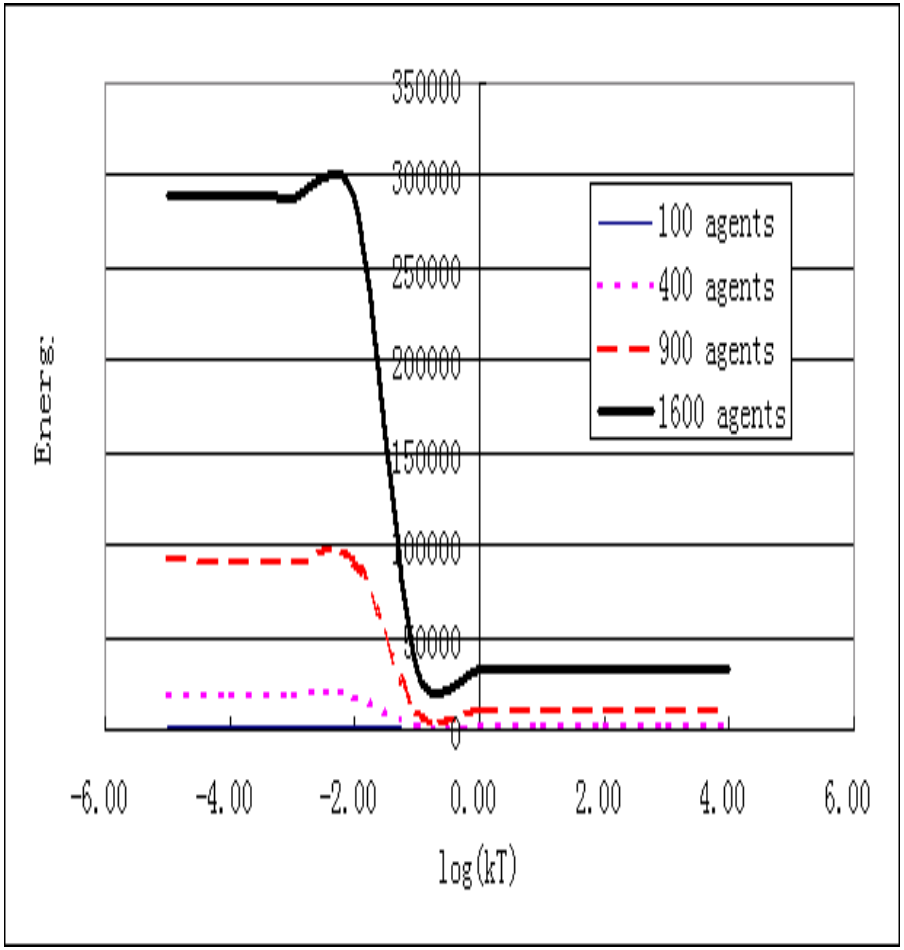
Two communicating agents may switch to the same color (belief)

Experiment

- Investigate stability of consensus formation in a group of up to 1600 agents
  - Agent belief represented as 20 floating point decimals
  - Agents communicate “sensor” readings which are ground truth with added zero mean normal noise
  - Agents maintain a maximum likelihood estimate of decimals based on received readings
  - Independent variable is temperature *T* (a measure of how “eager” agents are to interact)

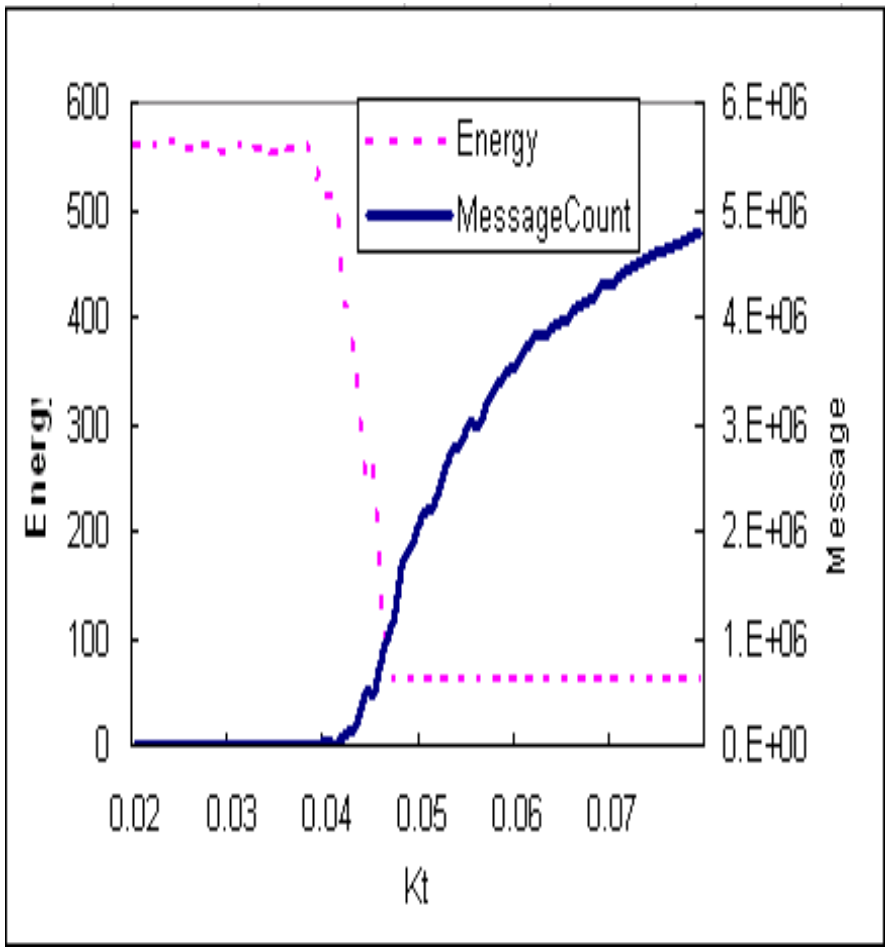
Results

- Energy is a measure of degree of consensus
- Low Energy indicates strong consensus between agents
- Found critical temperature below which consensus never occurs (system unstable)



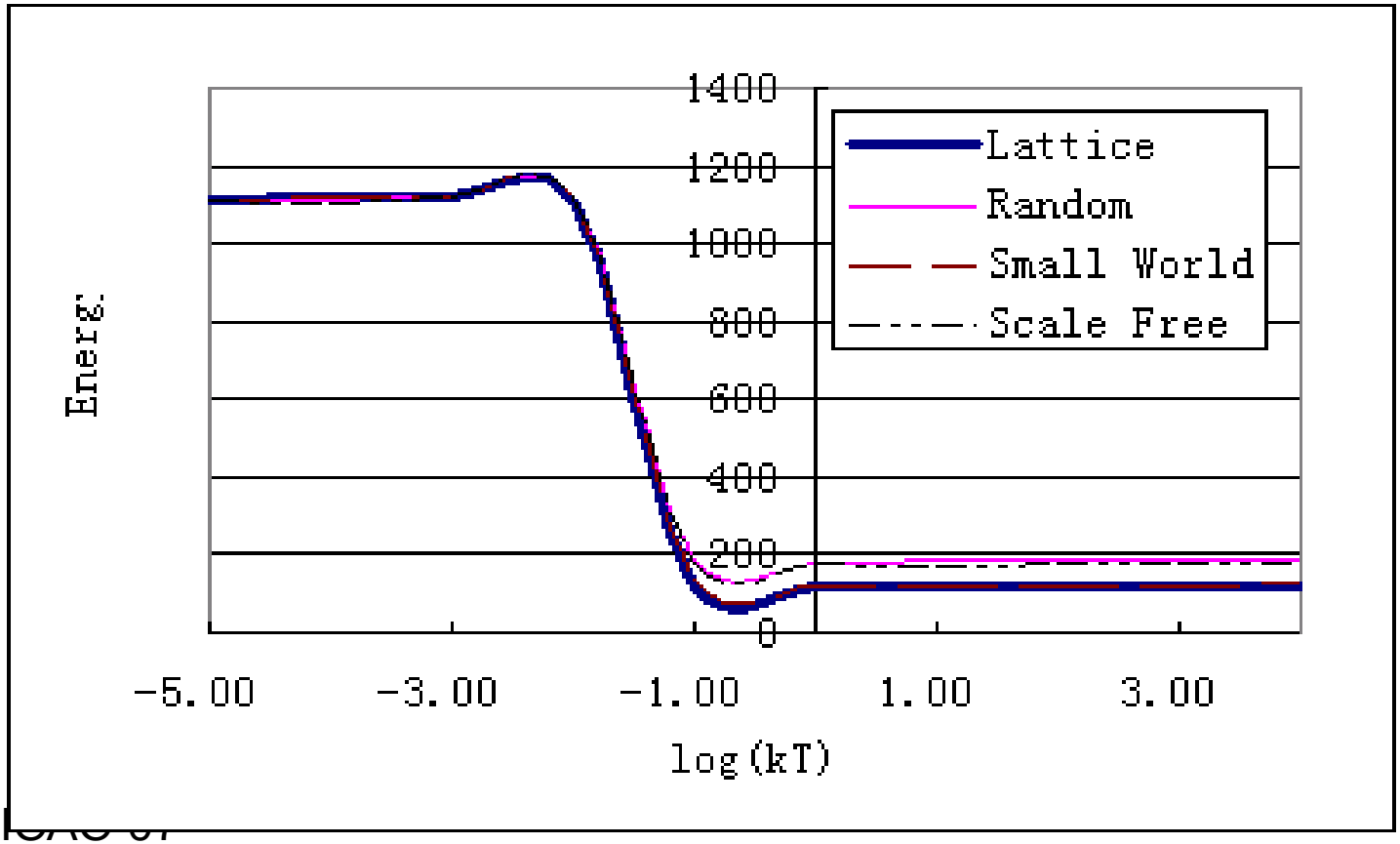
Results

- At critical temperature
  - Optimal balance between
    - Messages passed between agents
    - Degree of consensus
    - Consensus/stability guaranteed with high probability



Results

- Network topology does not seem to effect critical temperature



Research Issues

- **Modeling and analysis of Autonomic Systems (AS)**: better understanding of the relationship between the architecture and outcomes of Autonomic Systems
  - Does dynamic partial centralization help and how?
  - Diversity increases complexity but is useful for exploring the solution space
- **Design and Synthesis: techniques for AS** design (or modification) to obtain desired properties (e.g. reliability, robustness)
  - Can intelligence of nodes help?
  - Where should the intelligence be in the system?
- **Management and Control of AS**: techniques to manage agent creation, deletion, migration as well as overall system behavior
- **Modeling, Simulation and Measurement** techniques for AS
- **Local and global behavior**: techniques for understanding the relationship between local, emergent and global behavior in AS
- **AS life cycle**: techniques for handling SE life cycle concerns
- **Including humans**: techniques for modeling and understanding inclusion of humans in AS not simply as administrators but possibly as system components

References

[JAAMAS 02] Sycara, K., Klusch, M. Widoff, S. and Lu, J. "LARKS: Dynamic Matchmaking among Heterogeneous Agents in Cyberspace", Journal of *Autonomous Agents and Multiagent Systems,* vol 5, no. 2, July 2002.

[JAAMAS 03] Sycara, K., Paolucci, M., Van Velsen, M. and Giampapa, J. "The RETSINA Multiagent Infrastructure", Journal of Autonomous Agents and Multiagent Systems, vol. 7, nos. 1-2, July/September, 2003.

[AAMAS 04] Scerri, Xu, Liao, Lai, Sycara, "Scaling Teamwork to Very Large Teams" in Proceedings of the Third International Conference on Autonomous Agents and Multiagent Systems (AAMAS'04), July 19-23, 2004, New York, New York, USA.

[AAMAS 05] Y. Xu, P. Scerri, B. Yu, S. Okamoto, M. Lewis, and K. Sycara, "An Integrated Token-based Algorithm for Scalable Coordination", *Fourth International Conference on Autonomous Agents and Multi Agent Systems (AAMAS 05)*, Utrecht, the Netherlands, July 25-29, 2005 (*finalist for best paper award*).

[AAMAS 06] Yu B., Xu Y., Scerri P., Sycara K., Lewis M., "Scalable and Reliable Data Delivery in Mobile Ad Hoc Sensor Networks", Fifth International Conference on Autonomous Agents and Multi Agent Systems (AAMAS 06), Hakodate, Japan, May 9-12, 2006.

[AAMAS 07] Velagupundi, Scerri, Sycara, Owens, "Locating RF Emitters with Large UAV Teams", Sixth International Conference on Autonomous Agents and Multi Agent Systems, (AAMAS 07), Honolulu, Hawaii, May 2007.

References

[AAMAS Wkshop1 04] Liao, E., Scerri, P. and Sycara, K. "A Framework for Very large Teams" in *AAMAS'04 Workshop on Coalitions and Teams*, New York, NY., July 19, 2004.

[AAMAS Wkshop2 04] Xu, Y., Lewis, M., Sycara, K. and Scerri, P. "Information Sharing in Large Scale Teams" in *AAMAS'04 Workshop on Challenges in Coordination of Large Scale Multi-Agent Systems,* New York, NY, July 20, 2004

[IJCAI 97] Decker, K., Sycara, K. and Williamson, M. "Middle-Agents for the Internet", *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence* (IJCAI-97), Nagoya, Japan, August 1997 pp. 578-584.

[ICMAS 2000] Wong, C. and Sycara, K. "A Taxonomy of Middle Agents for the Internet" In Proceedings of the Fourth International Conference on Multiagent Systems, July 10-12, Boston MA., 2000 pp. 465-466.

[Security] Wong, H.C., Sycara, K., "Adding Security and Trust to Multi-Agent Systems", Applied Artificial Intelligence, vol 14, No. 9, pp. 927-941, October 2000.

[Humans] Sycara, K. and Lewis, M. "Integrating Agents into Human Teams", In Salas E. (ed.) Team Cognition, Erlbaum Publishers, 2002.

References

[Fusion 06] Yu, B and Sycara, K. "Learning the Quality of Sensor Data in Distributed Decision Fusion", International Conference on Information Fusion (Fusion 06), Florence, Italy, July 9-13, 2006.

[Fusion 07] Velagapudi, P, Prokopyev, O., Sycara, K., Scerri, P.. "Maintaining shared belief in a large multiagent team". In Proceedings of the Tenth International Conference on Information Fusion, Quebec City, CA, July 9-12, 2007.

[Fusion 07] Glinton, R., Scerri, P., Scerri, D., and Sycara, K. "An Analysis and Design Methodology for Belief Sharing in Large Groups" , In Proceedings of the Tenth International Conference on Information Fusion, Quebec City, CA, July 9-12, 2007.

[RETSINA Large Scale] Sycara, K., Giampapa, J. Langley, B. and Paolucci, M. "The RETSINA Multiagent System: A Case Study", in Software Engineering for Large Scale Multi-Agent Systems: Research Issues and Practical Applications, A. Garcia, C. Lucena, F. Zambonelli, A. Omici, J. Castro (eds), Springer Verlag, Vol. LNCS 2603, July 2003, pp. 232-250.

References

[RETSINA Large Scale] Sycara, K., Giampapa, J. Langley, B. and Paolucci, M. "The RETSINA Multiagent System: A Case Study", in Software Engineering for Large Scale Multi-Agent Systems: Research Issues and Practical Applications, A. Garcia, C. Lucena, F. Zambonelli, A. Omici, J. Castro (eds), Springer Verlag, Vol. LNCS 2603, July 2003, pp. 232-250.

[Mech Design] Huang, P., Scheller-Wolf, A. and Sycara, K. "Design of a Multi-Unit Double Auction Market", *Computational Intelligence*, (special issue on Agent Technology for Electronic Commerce), vol. 18, no. 4, November 2002.

[Coalitions 01] Yamamoto, J, and Sycara, K. "A Stable and Efficient Buyer Coalition Scheme for e-Marketplaces" *Proceedings of the Fifth International Conference on Autonomous Agents*, May 28-June 1, Montreal, CA. 2001.

[Coalitions 02]] Li, C. and Sycara, K. "Algorithms for Coalition Formation and Payoff Division in e-Marketplace", *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, Bologna, Italy, July 15-19, 2002.