Agents and Service-Oriented Computing for Autonomic Computing

Prepared by Faisal Sibai for CS 895

Presentation Agenda

- Objective of the paper
- Some definitions and clarification
- Autonomic need
- Similarities between MAS, SOC and Autonomic systems
- Autonomic in early papers
- Benefits of Agent and SOC technology towards Autonomic computing:
  - Individual Agents
  - Multi Agent Systems (MAS)
  - Service-Oriented Computing (SOC)
- Conclusion and Discussion
Primary Objective

Exploring the relationship between three paradigms: Autonomic computing, Multi-Agent Systems (MAS), and Service-Oriented Computing. The paper also presents what current autonomic research lacks and how Autonomic systems can better benefit from the previous research in MAS and SOC.
Definitions

**Multi Agent Systems (MAS):** A system composed of multiple interacting intelligent agents which are used to solve complex and difficult problems which are difficult or impossible for individual agents to handle.

- **Examples:** online trading, disaster response and modeling structures
- The agents in a multi-agent system have several important characteristics:
  - **Autonomy:** the agents are at least partially autonomous
  - **Local views:** no agent has a full global view of the system, or the system is too complex for an agent to make practical use of such knowledge
  - **Decentralization:** there is no designated controlling agent (or the system is effectively reduced to a monolithic system)

Source: [wikipedia](http://en.wikipedia.org/wiki/Multi-agent_system)
Definitions

- **Individual Agent Systems**: Are a smaller implementation of a multi-agent where usually only one agent interacts with the system, so all processing is contained in one agent and which interacts with a centralized systems.

- **Service-Oriented Computing (SOC)**: Design paradigm that specifies the creation of automation logic in the form of services. It is a strategic goal in developing Service-oriented architecture (SOA).

- **Autonomic computing**:
  - **Self-Configuration**: Configuring themselves automatically when computing resources are added or removed.
  - **Self-Healing**: self-repair and fault correction operations.
  - **Self-Optimization**: Monitoring and controlling resources to ensure optimal function with respect to defined requirements (SLA).
  - **Self-protection**: Proactively identifying and protecting themselves from arbitrary or malicious attacks or cascading failures.

- Autonomic systems can achieve this in both:
  - **Infrastructure level**: manages processing capacity, storage and communication bandwidth.
  - **Application level**: uses the functions the infrastructure level provides.

Why Autonomic systems?

• Increasing complexity in IT environments

• Individual components of computing systems, such as workload managers and database management systems, are becoming more difficult to configure, manage, and maintain as each release includes ever more features and tuning parameters

• With the advent of service-oriented computing (SOC), computing environments have become open and distributed, and components are no longer under a single organization’s control

• Typical enterprise computing environment is a heterogeneous, irregular, multivendor pastiche that’s difficult to configure, maintain, and trouble-shoot
Similarities

- Commonly cited definitions:
  - **Autonomic computing**: “computing systems that can manage themselves given high-level objectives from administrators.”
  - **Software agents**: “an encapsulated computer system, situated in some environment and capable of flexible, autonomous action in order to meet its design objectives”

- Individual agents can be identified with autonomic elements:
  - Autonomously
  - Adaptive entities representing resources or services that act both reactively and proactively
  - Sensing and responding to the system environment and interacting with each other to satisfy individual goals.

- They both manage:
  - Their own behavior.
  - They interact actions with other entities to achieve system-wide goals.

- Agent paradigm includes interactions between services and agents
- Most autonomic computing systems involve multiple systems, some of which will be services and most agent systems involve multiple agents
- Treatment of services overlaps with agents and MAS in this paper
- Analogies among autonomic computing, services, and MAS are strong and obvious yet surveys show very little ideas and technology has been transferred from MAS and SOC communities.
- Communication failure among the communities deprives the emerging autonomic computing field
Early papers describe autonomic systems:

- self-management that are strongly agent-oriented
- interacting collections of autonomic elements representing self-managing components such as computing resources
- Similar to a software agent in that it manages its own behavior by acting on data obtained from its sensors in accordance with policies and agreements established with other autonomic elements
- System-level autonomic behavior arises from interactions among the autonomic elements
- MAS behavior arises from interactions among individual software agents
- interactions are dynamic and flexible in pattern
- relationships among agents are established via negotiation and maintained via agreements created during the negotiation process
- Agreements between agents and service providers are, in fact, SLAs
Benefits

- **Individual Agents:**
  - Several technologies developed for Individual agents yet not applied to autonomic computing which include:
    - **Knowledge and Reasoning:**
      - Essential capabilities which govern behavior and interactions with environment and other agents.
      - Agents possess knowledge of its environment and own abilities and other agents (for example metalevel knowledge).
      - Lets the agent determine and reason about its own environment which policies to use.
      - Example self-healing: normal and abnormal behavior.
      - Reasoning can be applied on both infrastructure and application level. Infrastructure level considers the following:
        - **Structure:** making sure right components and general structure are in the right place.
        - **Function:** semantics are correct? Like input, outputs and preconditions are correct.
        - **Behavior:** meeting QoS and SLA.
      - **Benefit for autonomic systems:** enables agents to engage the external world and do their job better.
    - **Planning and Scheduling:**
      - Determine a partial order of actions that have a specific goal over time.
      - Planners must take several real-world issues such as coping with open environments.
Benefits

- **Benefit for autonomic systems**: present-day planning engines capabilities provide functionality needed by AC. In addition AC systems must also be able to assess plan execution when plans go awry.

- **Interagent communication**:
  - Formal agent communication languages and interaction protocols govern the content and sequence of messages that agents exchange with one another.
  - **Benefit for autonomic systems**:
    - Messages are relatively simple and inflexible in form
    - The mapping to the core application application’s functionality is clear-cut
    - With evolution from web services to agents their interactions will evolve from one-shot messages to extended multi-messages interactions

- **Multiagent Systems (MAS)**:
  - Transition many multiagent paradigms and technologies to autonomic computing should be straightforward due to the high similarity.
  - Many issues with multiparty service and resource negotiation research has been explored in (MAS).
  - Virtual emergent organizations, auctions and brokering are designed for this purpose.
Benefits

• Has modeled, explored and implemented multi-level commitments between multiple agents, services and organizations.
• Markets and auctions constitute major a major sub branch of multi-agent technology.
• Naïve approach used in autonomic computing to achieve self-management is by the usage of feedback loops which covers data center’s operations, this fails in an open environment with external services because of its uncontrollable nature.
• swarm (intelligence, insect based agents) a model which provides no direct message exchange but by jointly making and sensing changes to a shared environment
• Individual autonomy prototype components for agent-like systems have been built but have not exploited agent platforms such as JADE agent development or some others.
  o Benefit for autonomic systems:
    • Risk assessment of global system behavior with new configurations of system components or services is unexplored
    • In markets and auctions several authors have explored using such economic mechanisms to allocate resources, very few of these mechanisms have been used in AC systems
    • Interaction models by using MAS models would better evolve
Benefits

- **Service-Oriented Computing (SOC):**
  - The existence of service-oriented architecture (SOA) development tools such as the Eclipse “enable the design, configuration, assembly, deployment, monitoring, and management of software designed around a service oriented architecture” make it relatively easy for vendors to use an incremental approach to create Web services from their existing products.

- Two strategies for transitioning agent concepts into autonomic computing:
  - Services that respond to requests (www) is an evolutionary step towards a world of agents, services acquire agent capabilities.
  - SOA tool emergence suggest at least two strategies for creating agent toolkits for AC usage:
    - Create new agent toolkits similar to SOA tool kits which AC systems can use (Hitiachi’s Autonomous Decentralized Control architecture)
    - Add agent functionality and communication capabilities progressively to existing SOA toolkits and enterprise service busses (ESBs).

  - **Benefit for autonomic systems:**
    - Mostly reusable toolkits and the use of incremental approaches to create web services from existing products which eliminates creating stuff from scratch.
Conclusion

• There is a clear value in bringing autonomic systems, MAS and SOC communities together.

• The agent community has a good deal of technology that's relevant for autonomic computing.

• Autonomic computing has a set of problems that are important to agent researches

• Hosting cross cultural workshops

• Involving the SOA and SOC community into the autonomic discussion

• Holding autonomic computing competition at one of the main agent or semantic web conferences