AdaptAgent: Integrated Architecture for Adaptive Workflow and Agents

By

N.C. Narendra

Technology Cell, E-Solutions Center (ESC)

Hewlett-Packard India Software Operations Ltd; 29 Cunningham Road, Bangalore – 560 052; INDIA

Phone: +91-80-2251554; Fax: +91-80-2200196

Email: ncnaren@india.hp.com

Abstract

Adaptive Workflow and Software Agent technology have the potential to revolutionize B2B E-Commerce in the 21st century. Adaptive workflow allows for dynamism in business process definition and enactment, while Agent technology helps to automate decision making among the entities executing the adaptive workflows. Hence integrating Adaptive Workflow and Agents is highly essential for effectively facilitating B2B E-Commerce. However, efforts to integrate the two have been lacking so far. In this paper we describe our work on developing and implementing an integrated architecture called *AdaptAgent* where Adaptive Workflows and Multi-Agent Conversations are modeled, executed and adapted together seamlessly. *AdaptAgent* extends our previous work on a 3-tier adaptive workflow architecture [Narendra1]; it also builds on our earlier work on what we call "flexible workflow support" [Narendra3], which provides a means for increased flexibility in defining and managing adaptive workflows.

Keywords: agents, adaptive workflow, agent-oriented workflow, agent-oriented architectures for B2B

1. Introduction

There are two new technologies that have the potential to revolutionize B2B E-Commerce in the 21st century: Adaptive Workflow, which aims to manage the ever-changing business processes automated by information systems, and Software Agent technology, which promises to endow information systems with the requisite autonomy and decision-making capability so as to make them adaptable to the constantly changing business scenario that is a part of the 21st century business environment.

Workflow and Agent technologies are complementary to each other, and there has been a lot of work on integrating the two [Chen, Griss, Zeus]. However, there is one thing that stands out when reviewing the research work in this area, and it is that there has been little work integrating Adaptive Workflow and Agents. In B2B E-Commerce, adaptive workflows are essential for allowing dynamism in business process definition and enactment. Over and above this, Agent technology is useful in automating cross-organizational decision making, viz., determining new business processes, adapting existing business processes, negotiating (resp. renegotiating) new (resp. existing) contracts, which would result in workflow creation (resp. adaptation). Hence the need to integrate Adaptive Workflow and Agents in order to effectively facilitate B2B E-Commerce.

In this paper, we present an integrated architecture called *AdaptAgent*, where the adaptive workflow processes and multi-agent conversations are integrated and modeled seamlessly. Apart from this, the unique contribution of our work, is the idea of "flexible workflow support", drawing on our earlier work [Narendra3]. This approach classifies workflow processes and agent conversations on the basis of a 3-tiered control level approach consisting of loose, medium and tight control levels. The "tighter" the workflow or agent conversation, the more centralized is its definition and adaptation. Using this approach, it is possible to provide extra flexibility in supporting and managing agent-oriented adaptive workflow processes.

This paper is organized as follows. We present definitions and concepts in the next section. In Section 3, we present our 3-tier adaptive workflow architecture. Section 4 discusses our flexible workflow support approach, which also extends our 3-tier architecture. In Section 5, we present our integrated *AdaptAgent* architecture. The paper concludes in Section 6 with suggestions for future work.

2. Preliminaries

2.1 Workflows

We borrow our workflow definitions from our earlier work on adaptive workflow [Narendra1]: a workflow is modeled as a directed graph whose nodes are the tasks with edges denoting the flow of control between the nodes. Nodes are of two types – *work nodes* and *route nodes*. In work nodes actual task execution takes place, while route nodes are decision nodes that route the workflow information to the appropriate work nodes based on evaluation of certain boolean conditions. Edges are of four kinds – *forward edge, loop edge, soft-sync edge and strict-sync edge*. Forward edges depict the normal workflow execution, which is in a forward direction. Loop edges are backward pointing edges that are used to depict the repeated execution of loops.

The “sync” edges are used to support synchronizations of tasks from different parallel branches of a loop, and they are of two types:

- A “soft-sync” edge is used to signify a “delay dependency” between two nodes n_1 and n_2, i.e., n_2 can only be executed if n_1 is either completed or cannot be triggered anymore. This type of synchronization does not require the successful completion of n_1.
- A “strict-sync” edge between n_1 and n_2 requires that n_1 successfully complete before n_2 executes.

2.2 Agents

Currently, there is not much consensus on what an "agent" is, and many definitions abound. For our purposes, we will combine the following definitions:

- [MASIF]: " An agent is a computer program that acts autonomously on behalf of a person or organization"
- [Griss]: "an autonomous software component that interacts with its environment and with other agents"

Agents can be characterized as "weak" or "strong" agents [Wooldridge]. Weak agents possess the following characteristics: *autonomous* (having goals and plans for achieving them), *social* (can interact with other agents and their environment), *reactive* (can perceive their environment and respond to changes that occur) and *pro-active* (affect their environment rather than passively allowing their environment to affect them). In addition to the above characteristics, strong agents also possess the following characteristics: *mentalistic notions* (have beliefs, desires and intentions), *rationality* (can reason about their actions and perform actions which further their goals in line with their beliefs, desires and intentions), *learning* (have the ability to learn from their actions and their environment and other agents). In this paper, we will restrict our attention to weak agents.

A 7-axis characterization of agents can be found in [Griss], and has the following dimensions: Adaptability, Autonomy, Collaboration, Intelligence, Mobility, Persistence, Personality/Sociability. For our purposes, we require our agents to have high adaptability, autonomy, collaboration and persistence. The other 3 axes, i.e., intelligence, mobility and personality/sociability, are not applicable for us, since we are concerned with "weak" agents. The Federation for Intelligent and Physical Agents (FIPA), has been developing standards for agents and multi-agent systems. Their reference architecture for agent platforms is accessible from [FIPA].

There are many interesting linkages between workflow and agents, which we will be exploiting in this paper [Griss]:

- *Agents can collaborate to perform a workflow,* e.g., telecom provisioning, service provisioning, scheduling
- *Agents can be used to make workflow more intelligent,* e.g., by adding negotiation, reasoning or decision points
- *Workflow can be used to choreograph a set of agents,* e.g., application management
- *Workflow can be used to coordinate interaction between people and agents, having agents delegate to people or other agents*, e.g., telecom management system alerting a human operator, or assigning a repair or provisioning engineer

For our purposes, we recognize that the first two linkages represent *agent-enhanced workflow* (using agents to enhance workflow systems, i.e., agents representing workflow systems) and the last two represent *agent conversations* (i.e., using workflow concepts to model agent interactions in multi-agent systems). Later in this paper, we will realize that both types of linkages are essential for our *AdaptAgent*. Agent-enhanced workflow will be used to define what we will call "macro-workflow"; and agent conversations will be used to model what will be termed "micro-workflow". Hence (based on how we have defined Adaptive Workflow and Agents in Section 1) "macro-workflow" will model Adaptive Workflows, whereas "micro-workflow" will model Multi-Agent interactions.

3. Three Tier Approach for Adaptive Workflow Management

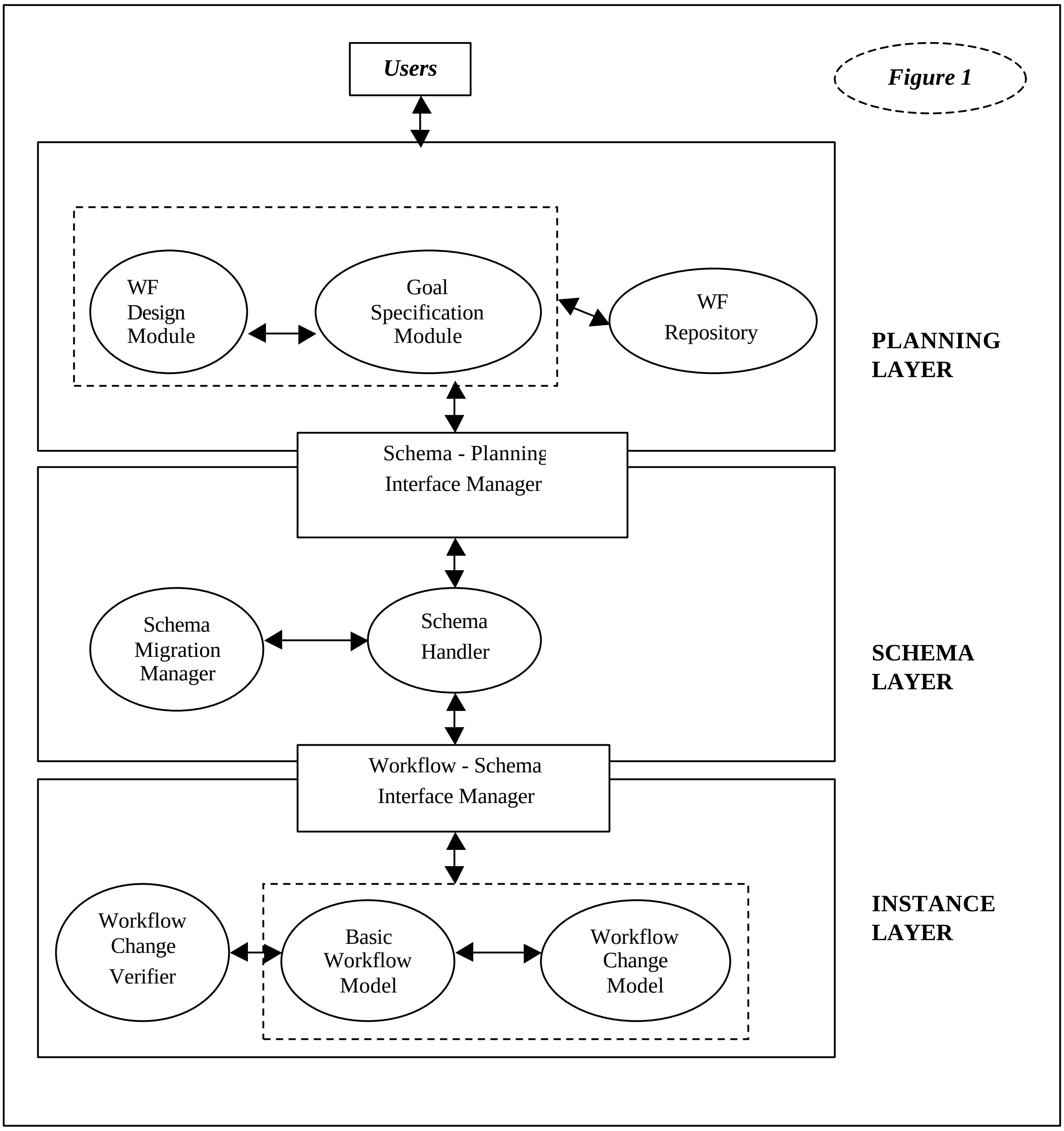
In [Narendra1], we have developed a 3-tier approach to adaptive workflow management, which is based on the graph-based workflow model described in Section 2.1. We have identified that adaptivity is basically of three types:

- **Adaptivity at instance level**: here, only the workflow instances need to be modified, perhaps to make them more efficient, or to make them easier to execute. The modules in our architecture that implement this, are:

    - *Basic Workflow Model* - stores the instances of our workflow model
    - *Workflow Change Model* - stores the constructs for specifying dynamic workflow changes as graph transformations
    - *Workflow Change Verifier* - performs the syntactic and semantic checking of the graph transformations
    - *Workflow-Schema Interface Manager* - is an interface module that interacts with the next higher layer, which is the Schema Layer
- **Adaptivity at schema level**: here, the workflow schema will need to be modified - this would arise if certain "ways of doing things" change, and will necessitate radical modifications to all the workflow instances of the schema in question. The modules that implement this, are:
    - *Schema Handler* - creates, versions and stores the workflow schema
    - *Schema Migration Manag*er - handles migration of workflow instance between schemas, and interfaces with the Schema Handler
    - *Schema-Planning Interface* Manager - interface module that interacts with the layer above, i.e, Planning Layer
- **Adaptivity at planning/goal level**: here, the goals of the workflow may themselves have to be modified in response to changing environmental conditions - this could cause radical changes in the schema themselves, which could cause disruptive changes in the workflow instances. The modules that implement this, are:
    - *Organizational Workflow Repository* - organization-wide repository of all workflow processes stored into the system; stores the workflow schemas and their instances, and also stores the goals and sub-goals that led to the creation of the workflow in the first place.
    - *Goal Specification Modul*e - through this module, the workflow designer first enters the goals and sub-goals that the workflow has to satisfy. He/she can then look into the *Organizational Workflow Repository*, for reusing any workflows from the past that can satisfy – either fully or partially – the goals that he/she has specified.
    - *Workflow Design Module* - with this module, and with the goal and workflow information from the *Organizational Workflow Repository*, the workflow designer can design the workflow schema.

He/she will then check it into the *Organizational Workflow Repository*. The schema will then be transferred into the *Schema Handler* via the *Schema-Planning Interface Manager*, where it will be appropriately versioned and stored. The appropriate version number is then communicated to the user.

The pictorial representation of this architecture is given in Figure 1 below.



4. Flexible Workflow Support

Traditionally, workflow management has been regarded as being centralized, monolithic, rigid and not easily amenable to adaptation; once a workflow process was defined, it was regarded as more or less "cast in stone," and to be executed with little change. However, this does not meet the needs of most B2B E-Commerce systems today, due to their inherent dynamism. To this end, in our previous work, we extended ideas from [IBM] and developed an architecture for what we called "flexible workflow support" [Narendra3], containing the following ideas:

- Any workflow can be classified to belong to one of the following three "control levels" – loose, medium or tight
- Loose workflows can be defined and started by any user (not necessarily the Workflow Administrator) in collaboration with the other participants in the workflow. Such a workflow can adapt itself as much as possible, depending on the need and on the participants.
- Medium workflows are also started by any user, but need approval from the Workflow Administrator, in order to ensure that they adhere to a particular workflow schema, if they are adapted.
- Tight workflows are the traditional workflows, which are defined and modified centrally by the Workflow Administrator.
- Hence our extensions to Figure 1 to accommodate flexible workflow support, are the following:
    - Logically separate the WF Repository for representing and storing these three types of workflows.
    - Have a Schema Discovery Module as part of the Goal Specification Module, which will assist the Workflow Administrator in observing the execution of several instances of a loose workflow and then "upgrade" it to medium level by designing a schema for it; all future executions of this process will then have to be at medium level, adhering to the schema.
    - Similarly, the Workflow Administrator can, using the Goal Specification Module, also observe the execution of several instances of medium processes and "upgrade" them to the tight level; the assumption behind this is that processes that are executed sufficient number of times can be better controlled centrally.
- Synchronization and interaction between different workflow processes can also be at the three control levels, and would depend on the control levels of the individual workflows – i.e., they can be either peer-to-peer or centralized interaction

- Extending this to the distributed workflow case means that there are two categories of workflows, which can be at different control levels – the overall workflow and its constituent workflows (which will be executed at different workflow servers) – resulting in nine different combinations for the three control levels. Essentially they are the following:
  - If the overall workflow process is at loose level, this means that it is defined among the workflow servers in a peer-to-peer fashion. However, each constituent workflow can be at any of the three levels, and managing the constituent workflows becomes an internal matter for the individual workflow servers.
  - If the overall workflow process is at medium level, this will need a Central Coordinator (CC) that will need to approve the workflow before it is executed. The individual workflow processes can be at any control level; however, the CC will ensure – from the viewpoint of the overall workflow process – that the individual workflow processes are represented as "black boxes" with predefined inputs and outputs, which the individual workflow servers will need to satisfy.
  - If the overall workflow process is at tight level, then the CC itself is in charge of defining and imposing the overall workflow process on the individual workflow servers. Here also, the individual workflow processes are specified as "black boxes" with predefined inputs and outputs, and it will be the responsibility of the individual workflow servers to design workflow processes to satisfy the inputs and outputs.
    - Hence the CC in this case is also an adaptive workflow system just as the one described in Section 3 and pictorially depicted in Figure 1, with the important difference that it supports and manages the overall distributed workflow process, with the constituent workflows being managed by (probably geographically separated) individual workflow servers.

5. *AdaptAgent* Architecture

The *AdaptAgent* architecture is envisioned as an extension of our 3-tier adaptive workflow and the flexible workflow-derived Central Coordinator (CC) ideas described above, in order to accommodate multi-agent interactions. That is, the CC not only coordinates the workflow processes but also the conversations among the agents that represent the workflow servers. Our basic premise is that agents execute workflows as part of meeting their goals; in addition, they would also need to interact with each other via conversations that would involve message passing among them in a language such as KQML [KQML].

An example of this is a supply chain scenario, where the agents represent an automobile company and its suppliers (which are also agents), working together to produce an automobile. All the agents will execute the overall workflow process needed for automobile production (consisting of its constituent workflows which are executed by the individual supplier agents), which they may either define themselves (i.e., the overall workflow process is at either loose or medium level) or which can be defined by the automobile company – the automobile company will act as the CC in this case (i.e., the overall workflow process will be at tight level). In addition, the supplier agents will also interact with each other and their CC via multi-agent conversations that would not necessarily involve the data transfer or product delivery that is common with workflow process execution. One such example of a multi-agent conversation, would be a negotiation [Wooldridge] over prices or delivery schedules, whereas delivery of an automobile component would be an example of a workflow execution by a supplier agent.

Hence as defined in Section 2, workflow processes are modeled as "macro-workflows", with agent conversations being modeled as "micro-workflows". The reason for this, is that micro-workflows can be modeled as single tasks within the macro-workflows with predefined entry and exit criteria (which are essentially boolean conditions); within a micro workflow, its related macro workflows are represented within these entry and exit criteria. We therefore have to extend our CC architecture of Section 4 in order to accommodate **two** types of distributed workflows – macro and micro workflows, with the following additional characteristics:

- A micro workflow is defined after its respective macro workflow has been created. As explained in the supply chain example above, the micro workflow is a conversation conducted by the agent executing the macro workflow, jointly with the other agents involved in the conversation. The conversation is started after the agent's macro workflow execution has crossed the point represented by the predecessor of the task representing the micro workflow. After the conversation is completed, the agent will then resume its macro workflow execution from the successor of the task representing the micro workflow. This will be the same for every other agent participating in the conversation. Naturally, the outcome of the conversation could determine the course of the future macro workflow executions of the agents, as per the macro workflow definitions.

- Hence, macro and micro workflow processes can be adapted relatively independent of each other; however, when a macro (resp. micro) workflow is adapted, the effect of the adaptation on its related micro (resp. macro) workflows will need to be considered before going ahead with the change. This is done as follows:
  - If the change is to a macro workflow instance or schema, then the affected micro workflow instance or schema will be syntactically and semantically checked. If, as a result of this, a change in the micro workflow schema is needed, then it must also be adapted as per the procedures described in Sections 3 and 4.
  - If the change is to the micro workflow instance or schema, then if there is a corresponding change in its input or output to any of its related macro workflows, the macro workflow may also need to be modified as per the procedures described in Sections 3 and 4.
- Macro and micro workflows can be at different control levels; however, this should not cause any issues, due to the highly modular way in which the micro workflows have been modeled as "black boxes" within macro workflows. Hence both the macro workflow and micro workflow are created, and their adaptivity managed, just like the overall distributed workflow process and its constituent workflow processes as described in Section 4. This is because both types of workflows will have components that are executed by different agents representing individual workflow servers. The only major difference, is that the micro workflow is created **after** the macro workflow is created.
- Since multi-agent conversations are structurally different from the macro workflow processes, they can be represented as multi-graphs using techniques such as Dooley Graphs [Singh]. However, from the viewpoint of the macro workflows, they are merely single tasks with predefined entry and exit criteria. Similarly, the presence of the macro workflow is modeled in the micro workflow by means of the entry and exit criteria.
- Agent Communication Languages (ACLs) such as KQML [KQML] can be used for implementing the multi-agent conversations. Each edge in the multi-graph will represent a performative sent by an agent

to its recipients, which would either be a broadcast message to the recipients, request for information, reply to an earlier message, etc.

6. Conclusions

In this paper we have presented an integrated approach for Adaptive Workflow and Agents. Adaptive Workflow is implemented within each agent by means of a 3-tier workflow architecture that we have previously developed [Narendra1]. This architecture is augmented with our later idea of "flexible workflow support" [Narendra3], which provides user flexibility in defining, executing and adapting workflow processes by classifying them into loose, medium and tight control levels. This augmented architecture has been further enhanced by representing agent-oriented workflow as a combination of "macro-workflow" (agent-enhanced workflow) and "micro-workflow" (conversations among the agents representing individual workflow systems).

There are several opportunities for future work:

- *Detailed Architecture Specification and Implementation* – due to lack of space, we have only sketched the broad outlines of our approach towards agent-oriented adaptive and flexible workflow; further details will be the subject of a forthcoming paper. Once detailed architecture specification is done, it needs to be implemented and experimentally evaluated. Another important research issue is (semi-) automatic derivation of the appropriate adaptive workflows and agent conversations to meet the goals specified in the Goal Specification Module. We have made a minor beginning in [Narendra2], but much more work remains to be done in this area. Work also needs to be done to investigate the extent to which our architecture can model a "Contractual Agent Society" as defined in [Dell].
- *Security and Trust Management* – issues such as Security and Trust Management [IBM2] become crucial in multi-agent interactions, hence the Goal Specification Module functionality should be enhanced to implement Security and Trust Management.
- *Distributed Service Management* – the Planning Tier should be augmented to implement distributed Service Management, in a manner similar to that described in [Sahai].

7. Acknowledgments

The author wishes to thank his manager, Srivatsa Krishnaswamy, and the ESC Center Manager, Padma Ravichander, for supporting his work.

8. References

[Chen] Q. Chen, P. Chundi, U. Dayal and M. Hsu, "Dynamic Agents," International Journal of Cooperative Information Systems, 1999

[Dell] C. Dellarocas, "Contractual Agent Societies: Negotiated shared context and social control in open multi-agent systems," Workshop on Norms and Institutions in Multi-Agent Systems, 4th International Conference on Multi-Agent Systems (Agents-2000), Barcelona, Spain, June 2000; also available from http://ccs.mit.edu/dell/aa2000/paper13.pdf

[FIPA] FIPA 97 Specifications, available from http://www.fipa.org/spec/FIPA97.html

[Griss] M.L. Griss, ""My Agent Will Call Your Agent … But Will It Respond?," Software Development Magazine, 2000

[IBM] K. Whittingham, "OpenWater - White Paper", IBM Research Division, Zurich Research Laboratory, 1999

[IBM2] A. Herzberg, Y. Mass, J. Mihaeli, D. Naor and Y. Ravid, "Access Control Meets Public Key Infrastructure, Or: Assigning Roles to Strangers," available from http://www.hrl.il.ibm.com/index.asp

[KQML] T. Finin, Y. Labrou and J. Mayfield, "KQML as an Agent Communication Language," in *Software Agents*, Jeffrey Bradshaw (editor), AAAI/MIT Press, 1997; also available from http://www.csee.umbc.edu/~jklabrou/publications/mitpress96.pdf

[MASIF] OMG's MASIF (Mobile Agent System Interoperability Facilities) specification; available from ftp://ftp.omg.org/pub/docs/orbos/98-03-09.pdf

[Narendra1] N.C. Narendra, "Adaptive Workflow Management: An Integrated Approach and System Architecture," ACM Symposium on Applied Computing 2000

[Narendra2] N.C. Narendra, "Goal-based and Risk-based Creation of Adaptive Workflow Processes," American Association for Artificial Intelligence (AAAI) Spring Symposium 2000

[Narendra3] N.C. Narendra, "Flexible Support and Management of Adaptive Workflow Processes," submitted to International Journal of Cooperative Information Systems, 2001

[Sahai] A. Sahai, J. Ouyang, V. Machiraju and K. Wurster, "End-to-End E-service Transaction and Conversation Management through Distributed Correlation", HP Labs Technical Report HPL-2000-145, also available from http://lib.hpl.hp.com/techpubs/2000/HPL-2000-145.pdf

[Singh] M.P. Singh, “Synthesizing Coordination Requirements for Heterogeneous Autonomous Agents,” Autonomous Agents and Multi-Agent Systems. volume 3, number 2, June 2000, pages 107-132; also available from http://www.csc.ncsu.edu/faculty/mpsingh/papers/mas/aamas-synthesis.pdf

[Wooldridge] M. Wooldridge and N. R. Jennings, "Intelligent Agents: Theory and Practice," In Knowledge Engineering Review 10(2), 1995; also available from http://www.csc.liv.ac.uk/~mjw/pubs/ker95.ps.gz

[Zeus] The Zeus Agent Building Toolkit Home Page, available from http://www.labs.bt.com/projects/agents/zeus/index.htm