SOFT COMPUTING AGENTS FOR DYNAMIC ROUTING

Georgi KIROV

Introduction

The strength of the distributed systems[1] as a new approach derives not only from its ability to allow people to communicate across big distances and at different times, but also from the ability of machines to help people communicate and manage information.

The pace of change in the software industry is so great that the traditional distributed solutions may run out of steam in the not too distant future. The forces of competition, regulation, convergence and globalization are driving change. Each one of these forces is changing the way we realize interconnected applications and the nature of networks and services. At present, the existing software technologies and network management infrastructures are getting old and information distributed systems cannot cope with the speed of response required in tomorrow's world. It is questionable whether traditional computing technologies can cope with the total information management demands of global network applications that companies will need to field in the early 21st century in order to remain competitive.

Increasingly, a great variety of different software applications and worldwide information services, such as WWW servers, databases, software packages, distance learning, video on demand, are being connected through the Internet, intranets, and other network systems. With the arrival of new technologies it is necessary to attempt to coordinate the action of these disparate entities within a cohesive framework. Standard distributed systems rely on message-based exchange with fixed connections, and as such, these systems are of limited efficiency when one attempts to use a large number of applications. Situations like these are prohibitively expensive in both time and resources, unless network applications cooperate effectively through an appropriate communication infrastructure. The field of distributed network systems is in a critical need of intuitive and innovative approaches and novel algorithms to

address the growing complexity in all of its different aspects: performance, stability, security, connectivity, efficiency, routing.

Current lines of research give the promise of stopgap solutions that will suffice for the next five to ten years. Areas such as distributed artificial intelligence (agent technology) appear to offer good term solutions for distributed network applications and service management. Current engineering technologies could breathe a breath of fresh air into management information systems. But whilst these newer approaches will partner humans in dealing with the forthcoming explosion in scale and complexity, they only offer better, proactive, access to information stores and expert system solutions that we enjoy today. Soft computing could multiply the benefits of such systems many times.[2,3]

Distributed information systems can be viewed as two level structures:

- A network level that deals with network traffic, management, and control; and
- A service level that deals with applications, inter-application communications, and service provided to the customers.

Soft computing technologies have had an impact on these two levels to a varying degree. One of the most popular soft computing technologies, fuzzy logic, has been applied to the network level for network routing, traffic modeling, and congestion control.[4]

The service level, the area of inter-application communications in particular, creates an opportunity to address problems within the Artificial Intelligence (AI) domain, e.g. within the intelligent distributed information systems and the intelligent multi-modal interfaces. Soft computing in conjunction with other AI techniques and software agents[5] can be used for knowledge representation and reasoning, information retrieval, search and optimization to make the resulting systems more robust, flexible and adaptive.

In an attempt to resolve some of the above-mentioned problems in network communications the author proposes an approach that combines the Bee-gent agent technology with the fuzzy logic representation.

The Bee-gent Technology

Basic Concept

Bee-gent is a communication framework based on the multi-agent model.[6] It has been developed by the Toshiba Corporation. It provides applications with autonomous

network behavior by "agentifying" them. Bee-gent supports agent-based inter-application communication, facilitating co-operation and problem solving.

This environment is based on two types of agents – Agent Wrappers (AW) and Mediation Agents (MA). The main function of the agent wrappers is to agentify existing software applications, while the mediation agents are responsible for inter-application coordination by handling all communications. The MA can move from an application to another, interacting with the AW. The AW themselves manage the state of the applications they are wrapped around. The Bee-gent applications are suitable for many software fields: distributed databases, management systems, and system optimization.[7] Figure 1 illustrates the relationships between existing applications, the agent wrappers, and the mediation agents.[8]

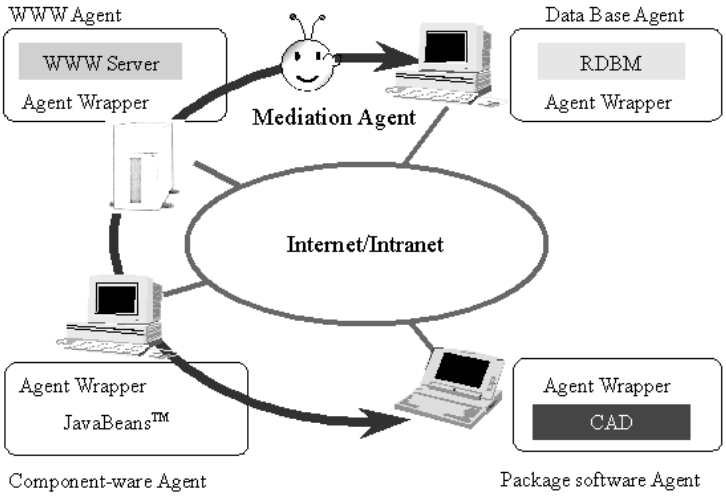

Figure 1: Relationships between Agent Wrappers and Mediation Agents.

The main characteristics of the Bee-gent technology can be defined as follows:

- Capabilities to connect distributed network applications via Internet, Intranet, and other network types;
- Ability the standard network users to receive requested data by information retrieval from databases distributed across the network.

Motivation to Use the Bee-gent Technology

The motivation to use the Bee-gent agent-based technology is inspired by the new capabilities built in the two agent types of the framework. The mediation agent controls the interaction protocol between applications in a standardized manner. It makes it easy to add to and modify the system configuration and the coordinating interactions. The mediation agent can migrate from one application to another and can preserve its present state – program code and data. Compared to the traditional message-based distributed technologies the network load decreases. The main reason for this is that the mediation agent communicates with the applications locally and the communication links can be disconnected after migration. The agent wrapper realizes interoperability between the applications. It provides a common communication interface. The communication between the agents is very appropriate for Internet use due to the fact that it is based on the XML/ACL representation format. Figure 2 shows the Bee-gent's system architecture.[9]

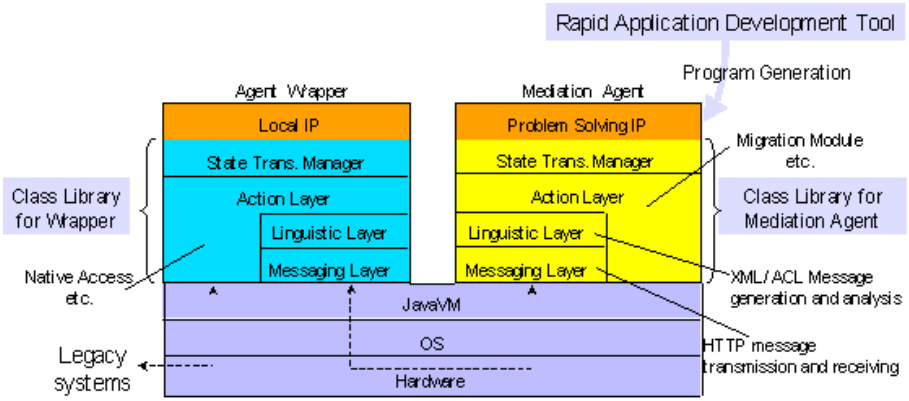

Figure 2: Bee-gent's System Architecture.

Comparison of the Bee-gent Technology with the Conventional Distributed Development Framework

The reason to have distributed object technologies is to create powerful, yet maintainable, applications. The "distributed" nature makes the system powerful, enabling scaleable solutions that can overcome the limits of the single machine performance, as well as the geographical boundaries. This section will describe the existing technologies, as well as a comparison of them. These technologies include the Object Management Group's (OMG) Common Object Request Broker

Architecture (CORBA), Microsoft's Distributed Common Object Model (DCOM), and the Java Bee-gent solutions.

The main objective of CORBA is to enable interoperability between objects on distributed systems. With the newest specification of CORBA, different vendors can now communicate, creating the "intergalactic object bus." Thus, CORBA provides a good architecture for working with distributed objects on a heterogeneous network; this aids greatly the integration of legacy applications. CORBA works by allowing clients and servers to communicate without worrying about network protocols and other communication aspects. In this specification of CORBA, the "client" is the process that uses an object and the "server" is the process that contains the instantiated object. The client does not need to know where the server is; the client can work just as if the object it works with exists in the same process space. A CORBA implementation achieves this by creating client stubs, server skeletons, and standard communication interfaces.

Distributed Component Object Model (DCOM) allows applications to work with objects on other computers. DCOM is a good infrastructure for compiled programs that need to make use of objects. Each interface in DCOM is compiled into a code, similar to CORBA stubs. DCOM interfaces, however, do not provide as much flexibility and speed for dynamic invocations. Unlike DCOM, it is not important for CORBA how implementations handle the objects. It is only required that a CORBA ORB be accessible through code created with the IDL. DCOM, however, is a binary specification. It matters very much what the compiled code looks like. DCOM works with pointers and arrays of pointers. A disadvantageous side effect of this fact is that executables created with Microsoft's Visual C++ will work with DCOM, whereas those created with other compilers may not.

There are no individual communication units programmed in the communication model of Bee-gent, in the "mediation agents," but procedures how to communicate and with which destinations. The mediation agents move to the individual destinations and communicate with them according to the procedures.

Soft Computing Agents for Dynamic Routing

This section presents a fuzzy distributed approach for dynamic network routing based on the Mobile Software Agents (MSA) paradigm.[10,11] The proposed routing technique combines Soft Computing Technologies (SCT)[12], more precisely fuzzy logic, with Software Agents (SA). The suggested solution is a distributed information system that allows interoperability between applications distributed across the network. The system consists of three parts located on different network nodes (see Figure 3):

- User application;

- Database applications that store the routing tables;
- Mobile Agent.

Taking into consideration the above-mentioned parts, it is appropriate to use the Bee-gent agent framework as a working shell. Agent wrappers agentify the first two constituents of the distributed system, while mediation agent realizes the third.

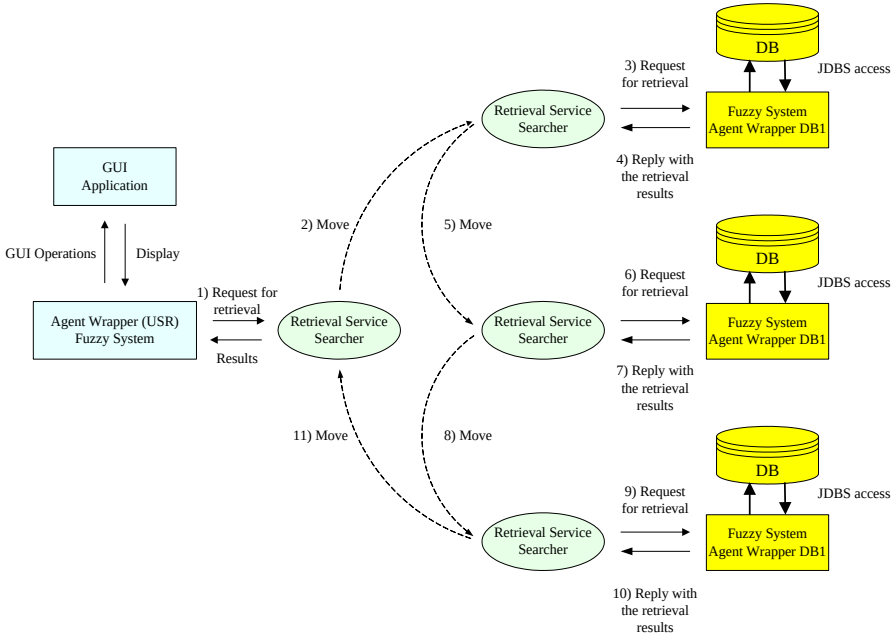

Figure 3: Soft Computing Agents for Dynamic Routing.

The main function of the user application is to evaluate user preferences. The application provides a user-friendly interface for the definition of some parameters and a fuzzy system for decision-making. The task of the fuzzy routing system is to generate a request for retrieval of the best routing path to a given destination based on the user's preferences.

The mobile mediation agent defines the interaction protocol between the user and the database applications in a standardized manner. The MA migrates from one application to another and can preserve its present state – program code and data. The mobile agent transfers the user's request to the destination databases and sends the result back.

A Fuzzy Routing Model

The main idea of the approach is to use more than one routing parameter by means of a fuzzy system in order to produce affordable link quality.[13] Fuzzy routing is treated as a multi-criteria optimization task that depends on the following input variables: time delay of a packet (*del*.), waiting queue (*que*.), cost of proposed routing (*cos*.), link capacity (*cap*.), priority of transmitted messages (*pri*.). Each variable is defined as a fuzzy linguistic variable with three inputs corresponding to small (*s*), middle (*m*) and large (*l*). It is assumed that all the necessary information about the investigated routing path is available. All variables are normalized in the [0, 1] interval:

$$p_{reli,j} = \frac{p_{absi,j}}{p_{\max i,j}} \qquad \rho_{reli,j} = \frac{\rho_{absi,j}}{\rho_{\max i,j}} \qquad t_{reli,j} = \frac{t_{transi,j} + t_{servi,j}}{t_{\max i,j}} \qquad l_{reli,j} = \frac{l_{absi,j}}{l_{\max i,j}}$$

where p_{rel} is the relative cost of the proposed routing, $\rho_{reli,j}$ is the relative link capacity, $t_{reli,j}$ is the relative time delay, t_{trans} is the transmission time, t_{trans} is the processing time, and $l_{reli,j}$ is the relative queue length.

Figure 4 shows a fuzzy routing system of the Sugeno type.

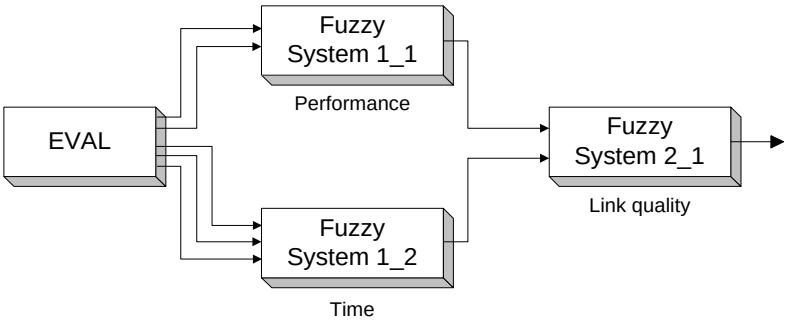

Figure 4: A Fuzzy Routing System of Sugeno Type.

The system consists of three fuzzy subsystems arranged in two hierarchical levels. This leads to:[14]

- Decreasing of fuzzy rule base size since the original system is decomposed into several similar fuzzy subsystems,
- Improving expert assessment by acknowledging intermediate results.

The system output is interpreted as a crisp value of link quality in the [0, 1] interval: the bigger the value – the better the link quality. A number of experiments have been performed in order to assess the usefulness of the fuzzy routing approach. All fuzzy systems have been implemented in FuzzyJava. Figures 5, 6, and 7 show the experimental results for the following user preferences: Cost = 0.5, Capacity = 0.5, Delay = 0.35, Queue_length = 0.73, and priority = 0.53. The fuzzy systems produced crisp results: Performance = 0.5, Time=0.524, and Link_quality = 0.495. The value of 0.495 means that the searched path has to have Link quality more than 0.495.

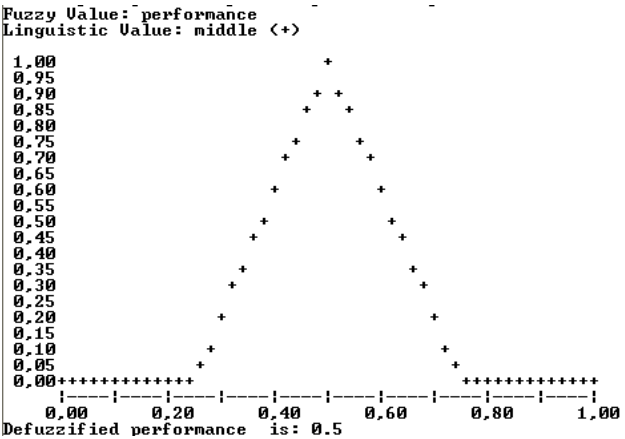

Figure 5

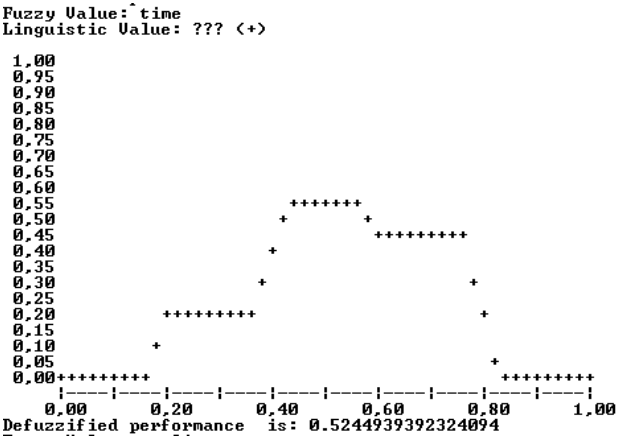

Figure 6

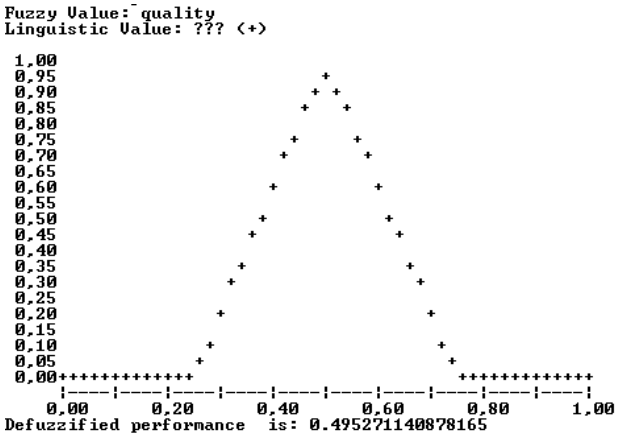

Figure 7

Implementing the Fuzzy Bee-gent Development Infrastructure for Dynamic Routing

The first step in the development procedure is to define the services provided by the distributed applications and the way in which these services are realized. The second step is to design the communication between the applications. The communication procedure is realized by the MA (see Figure 8).

In the proposed fuzzy Bee-gent system, the MA receives fuzzy request from the AW that agentifies the user interface application. The MA migrates then to the database application and requests the local database agent to perform a fuzzy search. The AW of the database application processes this request.

The MA and the agentification of applications using AWs turn every component of the Bee-gent distributed system into an agent. This facilitates the autonomous behavior of each system component and provides flexibility in such a way that if problem solving fails alternative procedures can be activated. The behaviors of the MA and the AW are individually described in the form of state transition diagrams. The agent functions (see Figure 3) can thus be represented in fixed form. The behavior of the mediation agent is described by the GUI tool of the Bee-gent framework (see Figure 8). In addition, the interaction between the MA and the AW is described by state transition diagrams.

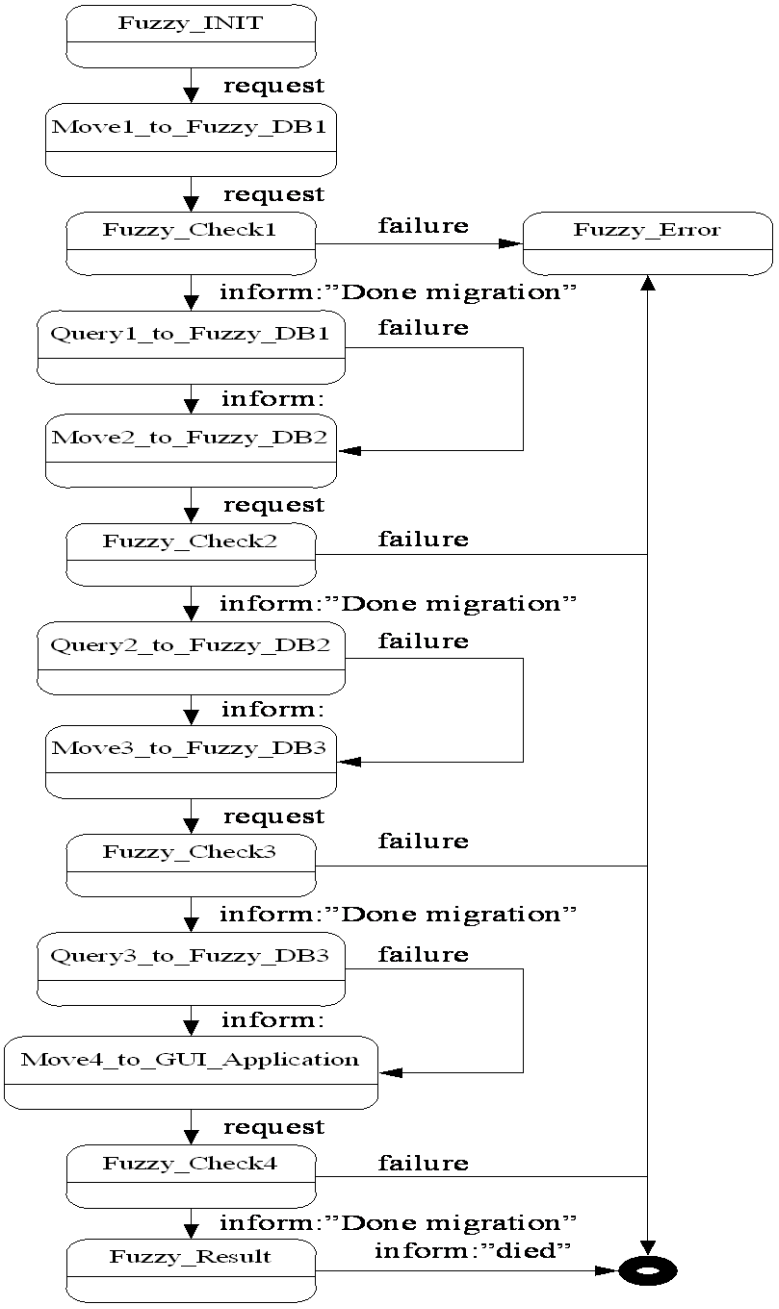

Figure 8: The Behavior of the Mediation Agent.

An Example

The user interface of the fuzzy routing system is shown in Figure 9. The user can define his/her preferences for the five input criteria and can select the destination node. The fuzzy system estimates user's preferences and generates fuzzy value that is a criterion for the best routing path. The resulting best routing path is shown in the control window.

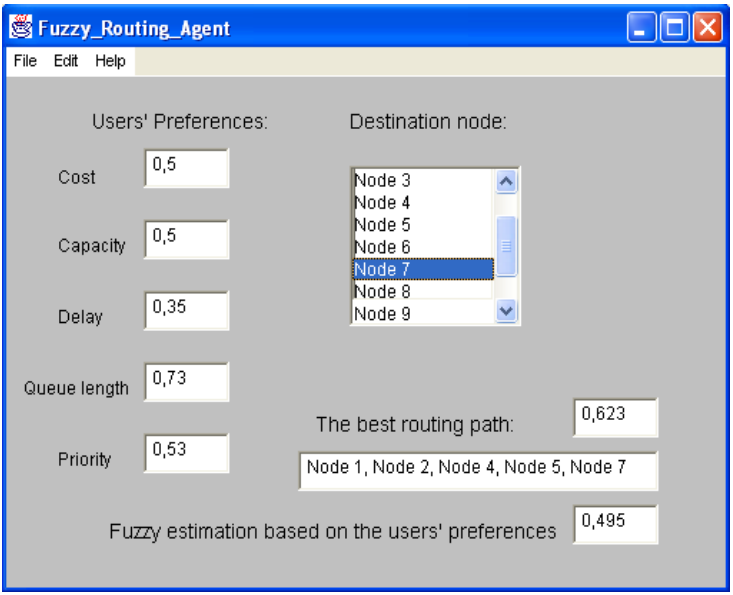

Figure 9: User Interface of the Fuzzy Routing System.

Conclusions

The most important conclusions of this study can be summarized as follows:

- The main idea of the proposed system is to use more than one routing parameter and a fuzzy system to obtain a crisp value for link quality. The fuzzy routing approach estimates five important criteria for network routing.
- Every routing strategy depends on individual user requirements. The fuzzy routing system can be easily adapted to different routing criteria. Tuning the linguistic variables and the rule bases might be promising and a different set of input parameters, which act according to other goals of the routing strategy, could be tested. In this sense the system is open and flexible.

- Compared to the traditional message-based distributed technologies, the Bee-gent technology decreases network load. The main reason for this is that the mediation agent communicates with the applications locally and the communication links can be disconnected after migration.

Notes:

[1] Randall Davis and Reid G. Smith, "Negotiation as a Metaphor for Distributed Problem Solving," *Artificial Intelligence* 20, 1 (1983): 63-109.

[2] Georgi Kirov, "Fuzzy Approach for FDDI Network Performance Improvement" (paper presented at the First International IEEE Symposium on Intelligent Systems, Student Session, Varna, Bulgaria, September 10-12, 2002), 17-23.

[3] Dimitar Lakov and Georgi Kirov, "Information Soft Computing Agents" (paper presented at the International Conference on Automatics and Informatics'2000, October 24-26, 2000), 135-140.

[4] Douglas E. Comer, ed., *Internetworking with TCP/IP*, Volume 3 (Prentice-Hall, 1991).

[5] Dimitar Lakov and Georgi Kirov, "Information Soft Computing Agents in Network Management" (paper presented at the International Conference on Automatics and Informatics, Sofia, Bulgaria, May 30 – June 2, 2001), A177- A183.

[6] Takahira Kawamura, Yasuyuki Tahara, Tetsuo Hasegawa, Akihiko Ohsuga, and Shinichi Honiden, "Bee-gent: Bonding and Encapsulation Enhancement Agent Framework for Development of Distributed Systems," *Systems and Computers in Japan* 31, 13 (2000): 42-56.

[7] Takahira Kawamura, Tetsuo Hasegawa, Akihiko Ohsuga, and Shinichi Honiden, "Bee-gent: Bonding and Encapsulation Enhancement Agent Framework for Development of Distributed Systems," in *Proceedings of the 6th Asia-Pacific Software Engineering Conference (APSEC 99)*, (IEEE, 1999), 260-267.

[8] *Tutorial for Bee-gent*, <http://www2.toshiba.co.jp/beegent/tutorial/tindex.htm> (7 October 2003).

[9] "System Development Example of Bee-gent, Environmental Information System," <http://www2.toshiba.co.jp/beegent/_example/index.htm> (7 Oct. 2003).

[10] Peter Dömel, Anselm Lingnau, and Oswald Drobnik, "Mobile Agent Interaction in Heterogeneous Environment," in *Proceedings of the First International Workshop on Mobile Agents* (Berlin: Springer-Verlag, LNCS 1219, 1997).

[11] Mercedes Garijo, Andrés Cancer, and Julio J. Sánchez, "A Multi-Agent System for Cooperative Network-Fault Management," in *Proceedings of the First International Conference and Exhibition on Practical Applications of Intelligent Agents and Multi-Agent Technology* (London, 1996), 279-294.

[12] Richard A. Bellman and Lotfi A. Zadeh, "Decision-Making in a Fuzzy Environment," *Management Science* B 17, 4 (1970): 141-164.

[13] Emad H. Aboelela and Christos Douligeris, "Fuzzy Multiobjective Routing Model in B-ISD," *Computer Communications* 21, 17 (November 1998): 1572-1585.

[14] Dimitar Lakov and Georgi Kirov, "Routing of Computer Nets via Fuzzy Logic" (paper presented at the Youth Science Session, Sofia, Bulgaria, in Bulgarian, July 1-2, 1999), 100-105.

GEORGI KIROV: Born 1973; M.Sc (1996, Computer Technologies) from the Technical University of Sofia, Bulgaria; Ph.D. (2002, Soft Computing models in computer networks) from the Institute of Computer and Communication Systems at the Bulgarian Academy of Sciences. Dr. Kirov is currently a research fellow at the Department of Knowledge Based Control Systems, Institute of Control and System Research in the Bulgarian Academy of Sciences (BAS) with publications in the fields of intelligent technologies in telecommunication network, software agents and programming languages.
E-Mail: kirov@icsr.bas.bg.