Network-Based Distributed Systems Middleware

Hervaldo S. Carvalho
Center For Future Health, University of Rochester,
Rochester, NY, U.S.A. and School of Medicine,
University of Brasilia, DF, Brazil
hervaldo_carvalho@futurehealth.rochester.edu

Amy L. Murphy
Computer Science Dept.
University of Rochester
Rochester, NY, U.S.A.
murphy@cs.rochester.edu

Wendi B. Heinzelman
Electrical and Comp. Eng. Dept.
University of Rochester
Rochester, NY, U.S.A.
wheinzel@ece.rochester.edu

Claudionor J. N. Coelho
Dept. of Computer Science
Federal University of Minas Gerais
Belo Horizonte, MG, Brazil
coelho@dcc.ufmg.br

Abstract: *Middleware is a growing, multidisciplinary area that merges knowledge from diverse areas such as distributed systems, networks and, more recently, embedded systems. This paper presents the results of an extensive review of middleware related literature, and presents an overview of critical features that must be considered during middleware development. These features include: network independence, plug and play operation, quality of service provisioning, service locating and data routing, providing appropriate transactions, scheduling transactions, providing mechanisms for system recovery, and interoperability among multiple languages and middleware systems. We also present a brief overview of our own middleware system, and describe how the above features have influenced its development.*

1. Introduction: In recent years, the development of wireless networking technology and improvements in sensors and embedded devices have made ubiquitous computing (computing available all the time) an appealing environment for many system developers. In order to be available all the time, applications in such an environment should be fault tolerant and demonstrate graceful degradation in the presence of failures. By integrating middleware technologies into ubiquitous applications, the applications themselves can be made more robust, and the development process can focus less on the complexities of the environment, and more on the application itself.

Generally, middleware is software that enables interaction among different nodes in a network. These nodes can supply services (service suppliers) or consume services (service consumers). Today's middleware has a variety of features, and is usually developed for a specific application environment. Nonetheless, the development process often addresses a common set of features with a variety of technologies. In this paper, we present a review of the middleware literature (Section 2) and then enumerate these common features (Section 3). Section 4 briefly presents our own middleware system and how it relates to these features. Finally, Section 5 presents discussion and conclusions.

2. Literature Review: Over the last 12 years, there were 3010 article references related to middleware in [1] and 786 articles according to [2]. We tried to correlate references to middleware with those of distributed systems, networks, and wireless networks by querying the IEEE Explorer Database using the following queries: middleware, distributed systems, network and wireless network. According to the IEEE Explorer Database, the first middleware article was published in 1993 by Desal et al. [3]. Since than, the number of articles increased to 7 in 1994 and to approximately 170 articles/year in the next 3 years after 1988 (see Figure 1).

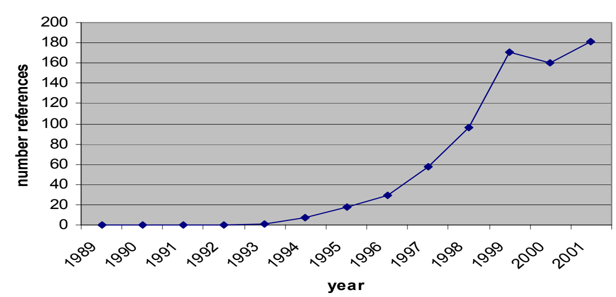

Figure 1: Number of references related to middleware published in the IEEE Explorer Database in the last 12 years.

Although the queries were very simple and different types of biases are related to these results, (for example, the IEEE Explorer Database is more related to engineering literature than to computer science literature), these queries showed the necessity for middleware followed the development of the networks and distributed systems. This positive correlation is very important to the justification of the different views of middleware and its evolution over the last 7 years. First, middleware development was strongly influenced by distributed systems technologies. This period was followed by a transition time where both distributed systems and network technologies influenced middleware design and development. Recently, the improvements in network technologies, embedded hardware, and Internet-based systems, brought further challenges and feature demands to middleware developers.

The articles published in 1994 addressed the necessity of interoperability among distributed systems, the necessity of connections to support community network applications [4,5], and the necessary requirements for a middleware at that time [6]. Mizunuma et al. presented one of the first views of middleware for real-time applications. At that time, the view of middleware was basically related to distributed systems, with little influence from network characteristics.

In 1995, Dlodloat and Bamford stated that middleware should connect the user interface from a distributed environment [7]. In the meantime, a number of articles began to address middleware from the network point of view, building on different platforms such as ATM networks [8]. At that time, Martin introduced the idea of middleware for the future of wireless applications [9]. In 1996, diverse projects based on mobile systems [10], multimedia [11] and Internet-WWW [12,13] contributed to enhance the view of middleware and to add new middleware functionalities based on both distributed systems and networks. In that year, CORBA was established as an important middleware solution [14,15,16,17]. Remote Procedure Calls (RPC) and Java also were presented as middleware alternatives to address some of the necessary functionalities [18,19]. From 1996 to 1999, many different aspects of middleware and new technologies began to appear. These aspects included object oriented programming [20], intelligent agents [21], mobile applications and computing [22,23], quality of service [24,25,26], fault tolerance approaches [27], software components [28], connections to databases [29], and multimedia data management [30], among others.

At the end of the 1990's, middleware was going in the direction of the network point of view [31,32]. Network-based distributed systems were established as the common basis for many applications and Internet use had grown all over the world. The network centric view of distributed systems [33,34,35] was established. At the end of 1998, middleware was predominantly based on the server-client architecture [36,37]. At the same time, many different problems and constraints needed to be addressed, including scalability [38,39], temporal and spatial constraints [40], cost [41], mobility [42,43,45], adaptability [45], multi agents [46], coordination [47], reliability [48], legacy systems [49], customizable services [50], performance [51], security services [52], service management [53], synchronization [54], Quality of Service [55], clustering [56], meta data [57], dynamic configuration [58], handoff [59], bandwidth reservation [60], logical versus physical location [61], semantic [62], and locating services [63].

From 1998 to 1999, new types of middleware appeared. Message-based techniques started to emerge as middleware communication methods [64,65]. Remote Procedure Call [66], event based middleware [67], publish-subscribe middleware [68], shared memory and tuple space middleware [69,70], and Java based middleware [71,72,73] were introduced as well. Most of these technologies had previously existed, but were not integrated into middleware until this time. Based on the fact that there were so many different types of middleware, middleware integration became necessary. Some authors tried to provide middleware integration using Java and CORBA [74,75]. In the attempt to provide interoperability among distributed systems, some exploited markup languages (XML) to achieve this goal [76]. The evolution of wireless networks [77] brought the possibility of developing distributed systems in wireless networks. This signaled the era of ubiquitous computing and sensor networks [78,79,80].

In the following years, the improvement of wireless networks and the necessity of plug and play applications and devices made service discovery algorithms necessary [81,82,83,84,85,86,87]. Some authors refer to these algorithms as middleware because of the importance they have to middleware functionalities. Additionally, quality of service [88,89,90,91], location and routing algorithms [92, 93,94], scheduling and application hand-off [95,96], grid computing [97], mobility [98,99,100], and semantics [101,102,103] have been major areas of research within the middleware field in the last two years.

Various articles from different fields such as distributed systems, parallel processing, optimization algorithms, and network management have contributed to the design and specification of a variety of middleware functionality. The relationship between middleware concepts and network properties is so close that some authors have considered middleware functionalities similar to those of the network layers. Schantz and Schmidt described a middleware stack with the following layers: Applications, Domain Specific Middleware Services, Common Middleware Services, Distribution Middleware, Host Infrastructure Middleware, Operating Systems & Protocols and Hardware Devices [104]. These authors proposed that because middleware had such a tight relationship to the network, its functional architecture should be represented by a middleware stack, similar to the network architecture stack.

Concurrent with middleware evolution, the last 5 years have seen great improvements in hardware technologies such as nano-technology, location devices, and a variety of ever-shrinking sensors. A large number of microprocessor-based systems of small size and great computational power have also become available. Today, microcontrollers have low cost (a few dollars), large memory capacity (on the order of megabytes), and processing capacity equal to a desktop computer from 1985. Further, they can support compact, embedded operating systems, 10 Mbps networks, the TCP/IP stack, web interaction through the HTTP protocol, and wireless communication.

The development of middleware has also been greatly impacted by the development of microelectromechanical systems (MEMS). MEMS has been presented as an important solution for sensing, integrated computation, and communication. MEMS technology is made from novel mechanical structures constructed directly from silicon. One example from a commercial application is the use of MEMS accelerometers for controlling airbags.

Also based on novel available devices, Tags and GPS are devices that are able to achieve the functionalities of location, tracking, and sensing. Tags use radio frequency identification (RFID) for tracking everything from packages to livestock. They now contain onboard memory and have anti-collision mechanisms to allow multiple e-tags to be read in the same space. The global positioning system (GPS) provides high-accuracy location data and can detect an object's presence and its position.

We cannot talk about network-based distributed systems without mentioning the Internet. The Internet is a distributed system (not structured) with a great number of users that has the potential for many styles of interaction. The communication is low cost and has a relatively democratic access system. Because the Internet is based on web servers, the use of embedded web servers on small hardware devices may allow access to the web's basic functionality - enabling client programs and browsers to fetch web pages and display them. Hyperlinks can link other local or remote files to that site and a link may reference a language script. One challenge is to build a compact yet functional web server for use in embedded systems.

We have provided an overview of the evolution of middleware during the last 10 years and the variety of factors that have influenced this evolution. We have also briefly reviewed new technologies that will influence ubiquitous computing in the coming years. Based on these aspects, in the next section we present important features that can be found in different middleware systems.

3. Features for Middleware: In this section we begin with a definition of middleware and then describe features to be considered by middleware designers.

3.1 Definition: Middleware is software that connects service suppliers and service consumers through a network. A service supplier is any type of networked node (device or software) that can offer services. Printers, sensors, databases, and applications are all examples of service suppliers. A service consumer is any type of networked node that requires a service from a supplier. Actuators and applications are examples of service consumers. Applications can simultaneously act as service suppliers and service consumers. A blood pressure analyzer is a service consumer when it receives a blood pressure signal from a blood pressure sensor (service supplier), but it can also send the result of its analysis to a display (service consumer), advising the user that his blood pressure is abnormal. This notation can be applied to all kinds of middleware—client-server, message-based, shared memory-based, and event-based.

3.2 Network independence: Middleware often serves as a bridge among multiple network technologies, or needs to adapt to multiple underlying networks. This includes the ability to connect wireline network technologies such as local ethernet and ATM backbones among themselves and with wireless network technologies such as Bluetooth, IEEE 802.11, or infrared wireless networks. Nonetheless, middleware intended to be flexible in a variety of settings should function independent of the network stack.

3.3 Plug and play functionalities: Another key aspect of middleware is the ability to adapt as the environment changes. Part of this is dependent on the service discovery mechanisms built into the system. These can be completely distributed, completely centralized, or a mixture of the two. The choice of mechanism depends on the size of the network, the communication overhead that can be tolerated, and how frequently the available components change. Yet another approach is to allow the service discovery approach to adapt to the current environment, selecting a centralized or distributed approach based on some aspects of the network itself such as density or traffic. To further increase scalability, mirroring approaches can be introduced.

Service discovery can also increase the flexibility of the middleware by providing an abstraction of the interface in the form of markup languages such as XML. These can effectively incorporate sophisticated matching criteria based on quality of service (QoS). Definition of the specific QoS measurement must be done with care, and most likely will depend on the application environment the middleware is applied to. Another dimension of service discovery is security, but this can be incorporated into the matching protocol (e.g., through password verification) or the transport protocols (e.g., through encryption).

3.4 Quality of Service: Another important feature of many middleware systems is their ability to manage system QoS among service suppliers and service consumers, taking into consideration network constraints. The service supplier's QoS includes the necessary components to be connected (devices), security access (password), authentication and data encryption (if applicable), power constraints in the case of battery-powered systems, and service availability constraints (the service might not be available all the time). The service consumer's QoS should include service and attribute needs over time and space based on reliabilities or other QoS measurements. It should also include the time constraints of the QoS (benefit function). The application should receive the data immediately or with some small delay. This is important because some applications such as real-time systems have strong time constraints, while e-mail applications in general are more relaxed with respect to delay. Identifying this variability across applications is important to properly manage system-wide QoS. Another possibility is spatial QoS, i.e., a quality related to physical location. For example, a user would like to print a file on the nearest and "best matched printer." Some matching algorithms only consider logical location, which is not compatible with spatial QoS. Network QoS is mainly related to bandwidth issues, but network density and traffic patterns can be considered as well. All QoS characteristics should provide to the middleware tools to deal with fault tolerance to provide graceful degradation of the system in the presence of failures. To achieve this goal, QoS specifications should be strongly influenced not only by middleware, but also by service suppliers and consumers.

3.5 Locating and Routing: Many middleware systems, especially those for mobile systems, require a notion of location and a mechanism for communicating across multiple network hops. Rather than build these services on a per-application basis, they should be incorporated into the middleware, or at least the facilities provided by the underlying network should be understood when designing the middleware. Although the service discovery protocols can perform partial location functionality by including this aspect in the service description, they may not have sufficient low-level information about the network layout to make optimal decisions. Therefore, we separate this, allowing the middleware to exploit low-level network information for location and routing when such tight controls are necessary.

3.6 Transactions: At the most fundamental level, middleware enables communication among distributed nodes. We use the word transaction to denote this interaction between a service supplier and a service consumer. A transaction should be established by the middleware based on matching specifications including QoS constraints. Transactions can be classified as continuous, intermittent with some prediction, or on demand scheduling. The transactions represent an important aspect of middleware that is related with how data and functionalities between the service supplier and service consumer are exchanged. Many different technologies have been applied to this goal, including software agents, threads, RPC, RMI, Java applets, and web-based interaction. The chosen technology should not over-burden the network, and should not prohibit the interaction between nodes, i.e., it should provide asynchronous connections, and possibly platform and language independent interactions.

3.7 Scheduling: Some environments place tight constraints on the operations that can be performed in a given time period. Although we have been considering QoS in many parts of the middleware, we address it again here, when concerning the management (scheduling) of interactions. Specifically, the middleware can decide on interaction order based on priority or bandwidth constraints. For example, if a service is about to be discontinued (e.g., a mobile service moving out of range), then the transactions involving it should be either completed, or transferred to different services matching the constraints. These interactions can be scheduled with high priority, and possibly allocated more bandwidth. Similar scheduling concerns arise in grid computing where middleware must consider the scheduling of tasks to processors.

3.8 Recovery system: If middleware works with critical transactions, it must include a recovery system to deal with failures. Sometimes a simple log-based scheme can be used, while other times, sophisticated database recovery mechanisms must be incorporated.

3.9 Interoperability: Some middleware emphasize the need to connect among multiple languages and/or middleware platforms. CORBA can connect most legacy applications based on different programming languages, but there must be a CORBA object and it is sometimes difficult to implement this in small systems. Some authors also question whether it will be successful on the scale of the Internet. Interoperability must be considered; however, the cost must be weighed carefully, especially when considering embedded systems. For non legacy systems, the use of a markup language such as XML or any other language that provides semantic independence is necessary to guarantee interoperability.

3.10 Miscellaneous: There are some functions that should also be considered when designing a middleware, such as applications, programming language, scalability, mobility, and event management, among others. There are different types of applications with different requirements and constraints, from real-time applications to e-mail, including multimedia data streams and data mining management. Different platforms and programming languages increase the complexity of the system. The presence of heterogeneous networks also brings the problem of scalability. A middleware designer should also take into account mobility aspects, including physical and logical location. Ideally, the middleware should react to events from all system components (services suppliers, services consumers and network). All of these aspects should be QoS based, this is a challenge for a middleware designer.

The middleware specifications described here tries to connect all of the papers published in the area and to cover problems introduced by different applications. We should consider these different points of view of the middleware.

4. MiLAN: Middleware Linking Applications and Networks: We are currently in the development of a new middleware platform targeting applications that are characterized by their ability to adapt to changing sets of available components, and their need to further constrain the active components for application-performance reasons [105]. Physical resources (e.g., transmission distance, bandwidth) and minimum application performance limit the input to certain subsets of available components. It is the job of MiLAN to identify these feasible sets and to determine which set optimizes the tradeoff between application performance and network cost (e.g., energy dissipation). MiLAN must then configure the network (e.g., determine which components should send data, which nodes should be routers in multi-hop networks, and which nodes should play special roles in the network, such as Bluetooth masters). A key feature of MiLAN is the separation of the policy for managing the network, which is defined by the application, from the mechanisms for implementing the policy, which is affected within MiLAN.

In terms of the features described above, MiLAN is intended to work in wireless environments, although it is applicable to multiple specific technologies (e.g., Bluetooth or 802.11). Applications themselves are able to adapt to changing sets of components providing input (in a sense, plug and play), and the system incorporates a service discovery mechanism to identify new components. Data is sent from the components to the application (transactions and scheduling). The application-performance we consider is exactly the definition of application QoS, which is specified by the application and maintained by MiLAN as the environment changes. In multi-hop networks, routing can be an important source of network energy management; therefore, we do not exploit any existing routing algorithms, but rather the middleware incorporates this functionality. This choice is different from many middleware platforms, which choose to sit above the network protocols; however, the goal of MiLAN is to increase the lifetime of a network by incorporating low level network functionality not usually manipulated by the application. We are still addressing concerns at the middleware level such as fault tolerance, but we expect this to eventually become a key feature.

5. Discussion and Conclusions: Middleware is an important multidisciplinary area that provides connections among distributed components. There are many different features of the middleware described in the literature. We have described many of them and discussed their necessity.

By providing this list of features, we are not advocating a single middleware that addresses all of them with all possible combinations. Considering the variety of application environments as well as the large number of consortiums and industries developing middleware, it is unlikely that a single middleware can satisfy all needs. However, this list does provide a comprehensive picture of the issues to be addressed in middleware development. Within our own work, we have used these criteria to show the strengths of our own system, and to illuminate future directions for our work.

References:

[1] http://citeseer.nj.nec.com/cs

[2] http://ieeexplore.ieee.org/Xplores/VadvSearch.jsp

[3] Weinrib, A._The need for a software infrastructure to support community networking applications. Proceedings of the 1st International Workshop on Community Networking Integrated Multimedia Services to the Home, 1994.page(s): 141 -143.

[4] *Wells, E.J.* 2 LAN/WAN integration: internetworking application interoperability. Military Communications Conference (MILCOM '94). Conference Record, IEEE , 1994, page(s): 220 -226 vol.1.

[5] Deutsch, J.M. The evolution of customer middleware requirements. Proceedings of the Third International Conference on Parallel and Distributed Information Systems, 1994, page(s): 262 -263.

[6] *Mizunuma, I.; Shimakawa, H.; Takegaki, M.* Real-time middleware based on the rate-monotonic theory. Proceedings of the IEEE Workshop on Real-Time Applications, 1994, page(s): 58 -62.

[7] Dlodlo, N.; Bamford, C. Separating application functionality from the user interface in a distributed environment. EUROMICRO 96. Beyond 2000: Hardware and Software Design Strategies., Proceedings of the 22nd EUROMICRO Conference , 1995. Page(s): 248 -255.

[8] Mizunuma, I.; Shen, C.; Takegaki, M. Middleware for distributed industrial real-time systems on ATM networks.Real-Time Systems Symposium, 1996., 17th IEEE , 1996, page(s):32-38.

[9] *Martin, R.L.* The future of wireless applications. Technologies for Wireless Applications Digest, 1995., MTT-S Symposium on , 1995.

[10] Leiner, B.M.; Ruther, R.J.; Sastry, A.R. Goals and challenges of the DARPA GloMo program [global mobile information systems]. IEEE Personal Communications , Volume: 3 Issue: 6 , Dec. 1996, page(s): 34 -43.

[11]Egelhaaf, C.; Moeller, E.; Schoo, P. Developing distributed multimedia telecommunication applications. IEEE Multimedia , Volume: 3 Issue: 3 , Fall 1996 Page(s): 76 -81.

[12] Pitt, J.V. A World Wide Web interface to agent-based negotiation. IEE Colloquium on Intelligent Agents and Their Applications (Digest No: 1996/101) , 1996, page(s): 8/1 -8/3.

[13] Ciancarini, P.; Rossi, D.; Vitali, F.; Knoche, A.; Tolksdorf, R. Coordination technology for the WWW. Proceedings of the 5th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1996 page(s): 322 -327.

[14] Gokhale, A.; Schmidt, D.C.; Harrison, T.; Parulkar, G. Operating system support for high-performance, real-time CORBA. Proceedings of the Fifth International Workshop on Object-Orientation in Operation Systems, 1996 page(s): 30 -37.

[15] Felber, P.; Garbinato, B.; Guerraoui, R. The design of a CORBA group communication service. Proceedings 15th Symposium on Reliable Distributed Systems, 1996, page(s): 150 -159.

[16] *Bakken, D.E.; Schantz, R.E.; Zinky, J.A.* QoS issues for wide-area CORBA-based object systems. Proceedings of WORDS '96, Second

Workshop on Object-Oriented Real-Time Dependable Systems,1996. Page(s): 110 -112.
[17] *Zhonghua Yang; Vogel, A.*Achieving interoperability between CORBA and DCE applications using bridges. Proceedings of the IFIP/IEEE International Conference on Distributed Platforms: Client/Server and Beyond: DCE, CORBA, ODP and Advanced Distributed Applications, 1996. Page(s): 144 -155.
[18] *Heindel, L.E.; Kasten, V.A.* Highly reliable synchronous and asynchronous remote procedure calls. Conference Proceedings of the 1996 IEEE Fifteenth Annual International Phoenix Conference on Computers and Communications, 1996. Page(s): 103 -107.
[19] *Lewis, T.*Will tiny beans conquer the world again? [Java language]. Computer , Volume: 29 Issue: 9 , Sept. 1996. Page(s): 13 -14.
[20] Hebbel, F. Distributed object oriented systems. ELECTRO '96. Professional Program. Proceedings. , 1996. Page(s): 137 -145.
[21] Pitt, J.V.; Mamdani, E.H.; Hadingham, R.G.; Tunnicliffe, A.J. Agent-oriented middleware for telecommunications network and service management. IEE Colloquium on AI for Network Management Systems (Digest No: 1997/094), 1997.Page(s):3/1-3/4.
[22] Haas, Z.J. Location-independent access in mobile systems. 1996 IEEE International Conference on Communications. ICC '96, Conference Record, Converging Technologies for Tomorrow's Applications. , Volume: 1 , 1996. Page(s): 255 -259.
[23] Blair, G.S.; Coulson, G.; Davies, N.; Robin, P.; Fitzpatrick, T. Adaptive middleware for mobile multimedia applications. Proceedings of the IEEE 7th International Workshop on Network and Operating System Support for Digital Audio and Video, 1997. Page(s): 245 -254.
[24] Conrad, C.; Stiller, B. The design of an application programming interface for QoS-based multimedia middleware. Proceedings 22nd Annual Conference on Local Computer Networks, 1997. Page(s): 274 -283.
[25] Huang, J.; Wang, Y.; Vaidyanathan, N.R.; Cao, F. GRMS: a global resource management system for distributed QoS and criticality support. Proceedings., IEEE International Conference on Multimedia Computing and Systems '97., 1997. Page(s): 424 -432.
[26] *Chao-Ju Hou; Ching-Chih Han; Yao-Min Chen.* Communication middleware and software for QoS control in distributed real-time environments. Proceedings The Twenty-First Annual International. Computer Software and Applications Conference, COMPSAC '97., 1997. Page(s): 558 -564.
[27] Shokri, E.; Hecht, H.; Crane, P.; Dussdault, J.; Kim, K.H. An approach for adaptive fault-tolerance in object-oriented open distributed systems. Proceedings., Third International Workshop on Object-Oriented Real-Time Dependable Systems, 1997. Page(s): 298 -305.
[28]Rader, J.A. Mechanisms for integration and enhancement of software components. Proceedings Fifth International Symposium on Assessment of Software Tools and Technologies, 1997. Page(s): 24 -31.
[29] Ludwig, A.; Becker, P.; Guntzer, U. Interfacing online bibliographic databases with Z39.50. Proceedings International Database Engineering and Applications Symposium, 1997. (IDEAS '97), 1997. Page(s): 379 -387.
[30] McCanne, S.; Brewer, E.; Katz, R.; Rowe, L.; Amir, E.; Chawathe, Y.; Coopersmith, A.; Mayer-Patel, K.; Raman, S.; Schuett, A.; Simpson, D.; Swan, A.; Teck-Lee Tung; Wu, D.; Smith, B. Toward a common infrastructure for multimedia-networking middleware. Proceedings of the IEEE 7th International Workshop on Network and Operating System Support for Digital Audio and Video, 1997.Page(s): 39 -49.
[31] Decina, M.; Trecordi, V. Convergence of telecommunications and computing to networking models for integrated services and applications. Proceedings of the IEEE , Volume: 85 Issue: 12 , Dec. 1997. Page(s): 1887 -1914.
[32] Koyanagi, K.; Saito, T.; Kanada, T.; Kitami, K. Technology initiatives for the near future in internetworking. IEEE Communications Magazine , Volume: 35 Issue: 5 , May 1997. Page(s): 108 -113.
[33] Rakotonirainy, A.; Berry, A.; Crawley, S.; Milosesevic, Z.Describing open distributed systems: a foundation. Proceedings of the Thirtieth Hwaii International Conference on System Sciences, 1997.Page(s): 405 -414 vol.1.
[34] Van Slyke, C.; Prescott, M.B.; Kittner, M. Overcoming barriers to distributed interorganizational systems. Proceedings of the Thirtieth Hwaii International Conference on System Sciences, 1997.Page(s): 69 -78 vol.4.
[35] Eckert, K.-P.; Schoo, P. Engineering frameworks: a prerequisite for the design and implementation of distributed enterprise objects. Proceedings, First International Enterprise Distributed Object Computing Workshop, 1997. Page(s): 170 -181.
[36] Benda, M. Middleware: any client, any server. IEEE Internet Computing , Volume: 1 Issue: 4 , July-Aug. 1997. Page(s): 94 -96.
[37] Wiederhold, G.; Genesereth, M. The conceptual basis for mediation services. IEEE Expert [see also IEEE Intelligent Systems] , Volume: 12 Issue: 5 , Sept.-Oct. 1997. Page(s): 38 -47.
[38] Kordale, R.; Krishnaswamy, V.; Bhola, S.; Bommaiah, E.; Riley, G.; Topol, B.; Torres-Rojas, F.; Ahamad, M. Community Networking Proceedings Fourth International Workshop on Middleware support for scalable services 1997. Page(s): 53 -58.
[39] Gokhale, A.S.; Schmidt, D.C. Measuring and optimizing CORBA latency and scalability over high-speed networks. Computers, IEEE Transactions on , Volume: 47 Issue: 4 , April 1998. Page(s): 391 -413.
[40] West, R.; Schwan, K.; Tacic, I.; Ahamad, M. Exploiting temporal and spatial constraints on distributed shared objects. Proceedings of the 17th International Conference on Distributed Computing Systems, 1997.Page(s): 280 -287.
[41] Kaltabani, A.K.; Tzifa, E.C.; Demestichas, P.P.; Anagnostou, M.E. Cost effective deployment of service logic in future distributed processing environments. Electrotechnical Conference, 2000. MELECON 2000. 10th Mediterranean , Volume: 1 , 1998. Page(s): 144 -147 vol.1.
[42] Hagen, L.; Breugst, M.; Magedanz, T. Impacts of mobile agent technology on mobile communication system evolution. IEEE Personal Communications , Volume: 5 Issue: 4 , Aug. 1998. Page(s): 56 -69.
[43] Kreller, B.; Park, A.S.-B.; Meggers, J.; Forsgren, G.; Kovacs, E.; Rosinus, M. UMTS: a middleware architecture and mobile API approach. IEEE Personal Communications , Volume: 5 Issue: 2 , April 1998. Page(s): 32 -38.
[44] Meggers, J.; Anthony Sang-Bum Park; Fasbeder, A.; Kreller, B. A multimedia communication architecture for handheld devices. The Ninth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 1998. Page(s): 1245 -1249 vol.3.
[45] Fitzpatrick, T.; Blair, G.S.; Coulson, G.; Davies, N.; Robin, P. Software architecture for adaptive distributed multimedia systems. Software, IEE Proceedings- , Volume: 145 Issue: 5 , Oct. 1998. Page(s): 163 -171.
[46] Wallis, A.; Haag, Z.; Foley, R. A multi-agent framework for distributed collaborative design. (WET ICE '98) Proceedings., Seventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1998.Page(s): 282 -287.
[47] Papadopoulos, G.A.; Arbab, F. Modelling electronic commerce activities using control-driven coordination . Proceedings. Ninth International Workshop on Database and Expert Systems Applications, 1998. Page(s): 583 -588.
[48] Curran, K.; Parr, G. Reliable remote isochronous applications over the Internet. 1998 IEEE International Conference on Systems, Man, and Cybernetics, 1998. , Volume: 3 , 1998. Page(s): 2860 -2865 vol.3.
[49] Budenske, J.R.; Ahamad, A.; Chartier, E.R. Agent-based architecture for exchanging data between legacy applications. 1998 IEEE International Conference on Systems, Man, and Cybernetics, 1998. Page(s): 2907 -2912 vol.3.
[50] Hiltunen, M.A. Configuration management for highly-customizable services. Proceedings. Fourth International Conference on Configurable Distributed Systems, 1998. Page(s): 197 -205.
[51] Dumas, S.; Gardarin, G. A workbench for predicting the performances of distributed object architectures. Simulation Conference

Proceedings, 1998. Winter , Volume: 1 , 1998. Page(s): 515 -522 vol.1.

[52] Kang, M.H.; Froscher, J.N.; Eppinger, B.J. Towards an infrastructure for MLS distributed computing. Proceedings. 14th Annual Computer Security Applications Conference, 1998. Page(s): 91 -100.

[53] Kellogg, L.; Zeisler, E. Interface management: a CORBA universal service manager. Proceedings of the IEEE Third International Workshop on Systems Management, 1998. Page(s): 148 -149.

[54] Hurfin, M.; Raynal, M. Asynchronous protocols to meet real-time constraints: is it really sensible? How to proceed? Proceedings. 1998 First International Symposium on Object-Oriented Real-time Distributed Computing, 1998. (ISORC 98). Page(s): 290 -297.

[55] Loyall, J.P.; Schantz, R.E.; Zinky, J.A.; Bakken, D.E. Specifying and measuring quality of service in distributed object systems. (ISORC 98) Proceedings. 1998 First International Symposium on Object-Oriented Real-time Distributed Computing, 1998. Page(s): 43 -52.

[56] Mendiratta, V.B. Reliability analysis of clustered computing systems. roceedings. The Ninth International Symposium on Software Reliability Engineering, 1998. Page(s): 268 -272.

[57] Kutsche, R.-D.; Sunbul, A.Meta data support for evolutionary software systems. SCCC '98. XVIII International Conference of the Chilean Society of Computer Science, 1998. Page(s): 84 -90.

[58] Kim, K.H.; Subbaraman, C.Dynamic configuration management in reliable distributed real-time information systems. IEEE Transactions on Knowledge and Data Engineering, , Volume: 11 Issue: 1 , Jan.-Feb. 1999. Page(s): 239 -254.

[59] Kounavis, M.E.; Campbell, A.T.; Ito, G.; Bianchi, G. Supporting programmable handoff in mobile networks. (MoMuC '99) 1999 IEEE International Workshop on Mobile Multimedia Communications, 1999. Page(s): 131 -141.

[60] Hoo, G.; Johnston, W.; Foster, I.; Roy, A. QoS as middleware: bandwidth reservation system design. Proceedings. The Eighth International Symposium on High Performance Distributed Computing, 1999. Page(s): 345 -346.

[61] Bakkar, A.; Van Steen, M.; Tanenbaum, A. From remote objects to physically distributed objects. Proceedings. 7th IEEE Workshop on Future Trends of Distributed Computing Systems, 1999.Page(s): 47 -52.

[62] Grefen, P.; Vonk, J.; Boertjes, E.; Apers, P. Semantics and architecture of global transaction support in workflow environments. CoopIS '99. Proceedings. 1999 IFCIS International Conference on Cooperative Information Systems, 1999. Page(s): 348 -359.

[63] Pfeifer, T.; Popescu-Zeletin, R. A modular location-aware service and application platform. Proceedings. IEEE International Symposium on Computers and Communications, 1999. Page(s): 137 -148.

[64] Martins, P.; Rigault, C.; Penkler, D. B-ISDN control plane solutions: comparison of the deployment properties of a message based versus an RPC based middleware. Global Telecommunications Conference, 1999. GLOBECOM '99 , 1999. Page(s): 414 -418 vol. 1a.

[65] Do-Guen Jung; Kwang-Jin Paek; Tai-Yun Kim Design of MOBILE MOM: Message oriented middleware service for mobile computing. Proceedings. 1999 International Workshops on Parallel Processing, 1999. Page(s): 434 -439.

[66] Yau, S.S.; Ning Dong; Tarim, F. Autonomous decentralized system with event service for information services. Proceedings. The Fourth International Symposium on Autonomous Decentralized Systems and Integration of Heterogeneous Systems. 1999. Page(s): 182 -189.

[67] Uramoto, N.; Maruyama, H. InfoBus repeater: a secure and distributed publish/subscribe middleware . Proceedings. 1999 International Workshops on Parallel Processing, 1999. Page(s): 260 -265.

[68] Picco, G.P.; Murphy, A.L.; Roman, G.-C. LIME: Linda meets mobility. Proceedings of the 1999 International Conference on Software Engineering, 1999. ,Page(s): 368 -377.

[69] Lehman, T.J.; McLaughry, S.W.; Wyckoff, P. T Spaces: the next wave. Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences. HICSS-32. 1999. Page(s): 9 pp.

[70] Bergner, K.; Rausch, A.; Sihling, M. Casting an abstract design into the framework of Java RMI. Proceedings. 1998 International Conference Software Engineering: Education & Practice, 1998. Page(s): 278 -285.

[71] Keller, A.; Reiser, H. Dynamic management of Internet telephony servers: a case study based on JavaBeans and JDMK. EDOC '99. Proceedings. Third International Enterprise Distributed Object Computing Conference, 1999. Page(s): 135 -146.

[72] Barelos, D.; Pitoura, E.; Samaras, G. Mobile agent procedures: metacomputing in Java. Proceedings. 19th IEEE International Conference on Distributed Computing Systems Workshops on Electronic Commerce and Web-based pplications/Middleware, 1999. Page(s): 90 -95.

[73] Long Yan; Chungmin Chen JAM: high performance Internet computing with massive Java applets. Proceedings. 19th IEEE International Conference on Distributed Computing Systems Workshops on Electronic Commerce and Web-based Applications/Middleware, 1999. Page(s): 3 -8.

[74] Hong An; Hong Li; Ming Wu; Zhihui Chen; Yongchang Ji; Feng Wang; Guoliang Chen A Java/CORBA based universal framework for super server user-end integrated environments. TOOLS 31. Proceedings Technology of Object-Oriented Languages and Systems, 1999. Page(s): 336 -341.

[75] Bouguettaya, A.; Benatallah, B.; Ouzzani, M.; Hendra, L. Using Java and CORBA for implementing Internet databases. Proceedings., 15th International Conference on Data Engineering, 1999. Page(s): 218 -227.

[76] Emmerich, W.; Schwarz, W.; Finkelstein, A. Markup meets middleware. Proceedings. 7th IEEE Workshop on Future Trends of Distributed Computing Systems, 1999. Page(s): 261 -266.

[77] Kikuchi, T. A middleware for improving utilization of wireless data telecommunication and its efficiency. ICCE. International Conference on Consumer Electronics, 1999. Page(s): 324 -325.

[78] Munson, M.; Hodes, T.; Fischer, T.; Lee, K.H.; Lehman, T.; Zhao, B. Flexible internetworking of devices and controls. IECON '99 Proceedings. The 25th Annual Conference of the IEEE Industrial Electronics Society, 1999. Page(s): 1139 -1145 vol.3.

[79] Schmidt, D.C. Middleware techniques and optimizations for real-time, embedded systems . Proceedings. 12th International Symposium on System Synthesis, 1999. Page(s): 12 -16.

[80] Gokhale, A.; Schmidt, D.C. Techniques for optimizing CORBA middleware for distributed embedded systems. Proceedings. IEEE INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies.1999. Page(s): 513 -521 vol.2.

[81] Woodrow. E. Data-centric Routing for Mobile Ad Hoc networks. Master of Science Thesis. Department of Electrical and Computer Engineering, University of Rochester, Rochester, N.Y., 2002.

[82] Gryazin, E. A. Service Discovery in Bluetooth. www.hpl.hp.com/techreports/2002/HPL-2002-233.pdf

[83] Glasser, U., Gurevich, Y., Veanes, M. Universal Plug and Play Machine Models. Technical report MSR-TR-2001-59.

[84] Waldo, J. The JINI Architecture for Network-Centric Computing. Communications of the ACM, July 1999/vol. 42, No 7. pg 76-82.

[85] Guttman, E. Service Discovery Protocol: Automatic Discovery of IP Network Services. IEEE Internet Computing, July-August, 1999, pg 71-80.

[86] Perkins, C. E. Service Location Protocol for Mobile Users. www.iprg.nokia.com/~charliep/

[87] Czerwinski, S. E., Zhao, B. Y., Hodes, T D., Joseph, A. D. Katz, R. H. An Architecture for a Secure Service Discovery Service. ?

[88] Tsetsekas C., Maniatis S., Lecture I. S. Rting QoS for Legacy Applications. Supporting QoS for Legacy Applications Notes in Computer Science (2001).

[89] Vass, J.; Zhuang, S.; Xinhua Zhuang Scalable, error-resilient, and high-performance video communications in mobile wireless environments. IEEE Transactions on Circuits and Systems for Video Technology, , Volume: 11 Issue: 7 , July 2001. Page(s): 833 -847.

[90] Nahrstedt, K.; Dongyan Xu; Wichadakul, D.; Baochun Li QoS-aware middleware for ubiquitous and heterogeneous environments. IEEE Communications Magazine , Volume: 39 Issue: 11 , Nov. 2001. Page(s): 140 -148.

[91] Kim, K.H.K.; Juqiang Liu QoS-driven resource management in real-time object based distributed computing systems. FTDCS 2001.

Proceedings. The Eighth IEEE Workshop on Future Trends of Distributed Computing Systems, 2001. Page(s): 222 -230.
[92] Dadarlat, V.; Ivan, C.; Coffey, T. A middleware based approach for designing routing protocols. IEEE CCECE 2002. Canadian Conference on Electrical and Computer Engineering, 2002. , Volume: 3 , 2002. Page(s): 1436 -1441.
[93] Harter, A., Hopper, A., Steggles, P., Ward, A., Webster, P. The Anatomy of a Context-Aware Application. Wireless Networks 1(2001) 1-16.
[94] Awduche, D. O., Ganz, A., Gaylord, A. An Application of Statistical Search Theory to Location Management in Wireless networks. Submitted o IEEE Journal onSelected Areas in Communications, 1996.
[95]Gill, C., Cytron, R., Schmidt, D. Middleware Scheduling Optimization techniques for Distributed Real-Time and Embedded Systems. IEEE Proceedings of the Seventh International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS 2002).
[96]Phan, T., Xu, K., Guy, R., Bagrodia, R. Handoff of Application Sessions Across Time and Space. http://pcl.cs.ucla.edu
[97] Fox, G., Gannon, D. Computational Grids. Computing in Science & Engineering, July/August 2001, pg 74-77.
[98] Capra, L.; Mascolo, C.; Zachariadis, S.; Emmerich, W. Towards a mobile computing middleware: Synergy of reflection and mobile code techniques. FTDCS 2001. Proceedings. The Eighth IEEE Workshop on Future Trends of Distributed Computing Systems, 2001. , 2001. Page(s): 148 -154.
[99] Capra, L. Mobile computing middleware for context-aware applications ICSE 2002. Proceedings of the 24rd International Conference on Software Engineering, 2002. Page(s): 723 -724.
[100] Murphy, A.L.; Picco, G.P.; Roman, G.-C. LIME: a middleware for physical and logical mobility Distributed Computing Systems, 2001. Page(s): 524 -533.
[101] *Chester, T.M.* Cross-platform integration with XML and SOAP. IT Professional , Volume: 3 Issue: 5 , Sept.-Oct. 2001. Page(s): 26 -34.
[102] Mascolo, C.; Capra, L.; Emmerich, W. An XML-based middleware for peer-to-peer computing. Proceedings. First International Conference on Peer-to-Peer Computing, 2001. ,Page(s): 69 -74.
[103] Grefen, P.; Vonk, J.; Boertjes, E.; Apers, P. Semantics and architecture of global transaction support in workflow environments. CoopIS '99. Proceedings. 1999 IFCIS International Conference on Cooperative Information Systems, 1999. Page(s): 348 -359.
[104]Schantz, R. E., Schmidt, D. C. Middleware for Distributed Systems Evolving the Common Structure for Network-centric Applications. (2001) www.cs.wustl.edu/~schmidt/PDF/middleware-chapter.pdf .
[105] A. Murphy and W. Heinzelman, "MiLAN: Middleware Linking Applications and Networks," *TR-795*, University of Rochester, Computer Science, Nov. 2002.