TOWSON UNIVERSITY
COLLEGE OF GRADUATE EDUCATION AND RESEARCH

BIBLIOGRAPHY OF MIDDLEWARE

BY

JYOTSNA VODELA

A PROJECT PRESENTED TO THE FACULTY OF

TOWSON UNIVERSITY

IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE DEGREE

MASTER OF SCIENCE IN COMPUTER SCIENCE

SEPTEMBER 2004

TOWSON UNIVERSITY
TOWSON, MARYLAND 21252

Bibliography of Middleware

Abstract

The need for middleware has been moving from phases of client/server applications to multiple stages of application integration and expanding its wings into areas such as embedded systems, real-time systems, multimedia and security. It has initiated development of diversified middleware applications and solutions. This paper provides a thorough survey of middleware solutions, by grouping several published papers based on a particular area, discipline or its application, summarizing each of them, and understands further research directions suggested by each author.

1. Introduction:

The role of middleware in broad is to ease the task of designing, programming and managing distributed applications by providing an integrating programming environment. It abstracts over the complexity and heterogeneity of a distributed environment and covers a multitude of underlying technologies, operating systems and programming languages. Nonetheless, there are still many uncovered areas and wide-open issues to be addressed.

2. Bibliography:

There are several middleware solutions in the market and various new platforms and directions established as suggested by many prominent researchers. Our objective is to understand the different perspectives, existing trends and thus propose scope for improvement and a seamless integration and standardization of middleware product structures.

We have studied many middleware papers and broadly grouped them into the following categories, based on a particular area, discipline and application, as follows:

1. Aspect Oriented Programming
2. Components/Frameworks
3. Middleware Solutions
4. Embedded & Real-Time Systems
5. CORBA/ORB
6. ADA
7. DB-Query Middleware
8. Enterprise Architecture & Solutions
9. QoS
10. Ontology
11. Actors
12. Client/Server
13. Agents
14. Issues

15. Mediators
16. Multimedia
17. AOA
18. XML
19. Security
20. Distributed Processing & Computing
21. Tutorials/Whitepapers

The summaries of all such papers and related references, are given below:

I. Aspect Oriented Programming

Quantifying aspects in middleware platforms [147]

In today's world, middleware systems are getting widely adopted and more functionally mature. It is also becoming increasingly difficult for the architecture of middleware to achieve a high level of adaptability and configurability, primarily because of the limitations of traditional software decomposition methods. The author describes the aspect oriented programming approach as one, which brings new perspectives to software decomposition techniques and such new concepts allows us to compose, with respect to the primary decomposition model, the modular solution for each orthogonal design requirement. The author also says "we still need to apply quantitative analysis to justify our motivation." They use AspectJ for an empirical study to re-factor a number of aspects identified through the mining process and implementations, which exist in multiple places of the original code, are grouped within a few aspect units. The authors are also trying to gain more experience and define new horizontal decomposition procedures.

A Scalable, Distributed Middleware Service Architecture to Support Mobile Internet Applications [108]

To improve the scalability of the existing middleware architecture, the authors have designed an efficient distributed middleware service layer that properly maintains application session handoff semantics while being able to service a large number of clients. They state that this service layer improves the scalability of general client-to-application server interaction as well as specific case of application session handoff and thus provide scalability as a response to increased workload.

II. Components/Frameworks

The Role of Software Architecture in Constraining Adaptation in Component-bases Middleware Platforms [9]

A number of technologies are emerging to support different levels of configurability and re-configurability, mostly for those middleware platforms that are based on the concepts of open implementation and reflection. The author in this paper discusses software

architecture role in maintaining the overall integrity of the system in such an environment. The author also discusses precisely about the extensions to the Aster framework to support the re-configuration of a reflective (component-based) middleware platform.

More specifically, they have explored the role of the Aster framework in supporting dynamic adaptation in the context of the Open-ORB middleware platform. Inspite of the benefits of this approach with respect to software system robustness, it should be demonstrably efficient for it to support for middleware adaptation regarding both the implementation of the Aster extensions and OpenORB infrastructure.

Towards Software Plug-and-Play [11]

'The shift towards developing systems from components has been more evolutionary than revolutionary. It has its roots in accepted architectural principles such as layering, modularization, and information hiding'. The authors in this paper discuss research ideas and technologies that will facilitate the transition toward component-based software development by leveraging object-oriented middleware technologies such as CORBA and OLE.

In particular, the paper focuses on the following:

- A component based architecture model which provides the required interfaces, binding or interconnection of component, and system configuration. An architecture specification language (ASL) has been developed to support this model.
- A component specification analysis tool that check the compatibility of interfaces between components and generates adapters to bridge incompatible interfaces.
- A component integration tool that offers capabilities to transform abstract component-based architecture specifications into executable code that can run on distributed communication infrastructures such as CORBA or OLE.

The Anatomy of Component [17]

In this paper, the author discusses the following main points:

- A component should implement meaningful amount of business logic.
- A component should be extendible and configurable.
- A component should be available to future applications.
- A component should make use of open based standard middleware protocols
- A component should be managed through open based standard management packages.
- A component should run on a wide range of platforms.
- A component should be reliable.

The author states "Microsofts COM+(DCOM) and MTS together can be used to provide components, and when judge such components they could well and up with good marks."

Integrating Meta-Information Management and Reflection in Middleware [24]

The authors in this paper present an approach to represent and maintain middleware meta-information and show how it is integrated with their reflective architecture for middleware, which is based on the idea of multi-model reflection. The result of this is a principled way to control the use and evolution of meta-information in the platform, also offering a means to harness the power of reflection. The authors relate how reflective meta-objects integrate with the type repository in order to allow for dynamic reconfiguration, preserving consistency properties defined in the repository.

Thus they propose an extensible way for specifying and reusing middleware configurations, based on existing technologies for meta-information management and repositories and, an approach for the dynamic evolution of middleware configurations, based on a multi-model reflective framework, in order to enable adaptation in the face of new requirements and environmental changes. They also present a case study of interoperability between platforms with different systems and define a scheme to enable CORBA objects to be meaningfully communicated to entities in a COM environment.

A Configurable Multimedia Middleware Platform [25]

The author in this article focuses mainly on dynamic QoS management issues and describes low level concurrency and communications mechanisms designed to maximize performance and predictability in multimedia middleware. He implemented these mechanisms in a multimedia middleware platform called GOPI (Generic Object Platform Infrastructure) and attempts to deliver performance and predictability in a configurable manner in a standard operating environment.

The main dimensions of configurability in GOPI are as follows:

Threads

- Configure by choosing appropriate ASC and scheduling parameters.
- Change ASC and scheduling parameters of threads at runtime.
- Add new ASCs dynamically.
- Configure ASCs with preemption, no preemption, or timeslicing.

Bindings/communications

- Configure by choosing appropriate ASPs and QoS
- Renegotiate ASPs and binding QoS at runtime
- Add new ASPs dynamically
- Configure ASPs with I/O polling or I/O notification.

Exploiting ADLs to Specify Architectural Styles Induced by Middleware Infrastructures [33]

In this paper, the authors provided an evaluation of Architecture Definition Languages (ADLs) as to their suitability for defining middleware-induced architectural styles. They

identified new requirements for ADLs, and highlighted the importance of existing capabilities. As middleware has demonstrated its usefulness and effectiveness in a number of practical cases, the software architecture community has now the potential to formalize these achievements in expressive and usable ADLs and, more generally, to coordinate the definition of support technology for the development of middleware-based applications.

The authors have a long-term objective to implement an environment that supports the definition of architectures by providing a library of styles induced by specific middlewares.

Introduction to the Computing Surveys' Electronic Symposium on Object-Oriented Application Frameworks [40]

The emerging focus on object-oriented enterprise and application frameworks (OOAFs) in the OO community offers software developers both a new vehicle for reuse and a way to capture the essence of successful architectures, components, policies, services, and programming mechanisms. By providing reusable skeletons on which to build new applications, frameworks can save countless hours and lot of development costs.

A Modular Middleware Flow Scheduling Framework [43]

Flow Scheduling Framework (FSF) is an extensible set of classes that provide basic synchronization functionality and composition mechanisms to develop data-stream processing components. In this dataflow approach, applications are implemented by specifying data streams and their path through processing nodes, where they can undergo various manipulations.

The authors in this paper describe, the FSF data and processing model to support stream synchronization in a concurrent processing framework, and shown how to build component-based applications in this model. They have also demonstrated a prototype FSF implementation with a real-time video processing and composting application.

Applying model-integrated computing to component middleware and enterprise applications [46]

Component middleware has emerged as a promising solution to many limitations with OO application frameworks. This type of middleware consists of reusable software artifacts that can be distributed or co-located throughout a network. A proliferation of component middleware technologies have emerged recently to address various requirements of enterprise applications. These types of applications are increasingly being assembled from components belonging to disparate middleware technologies, which increases the effort required to integrate and deploy semantically compatible and interoperable components across multiple middleware platforms.

The authors also proposed a MIC toolsuite called MIAO which extends the popular GME

Modeling and synthesis tools to support the development, assembly and deployment of QoS-enabled enterprise applications using component middleware. To ensure these QoS requirements can be realized in the middleware layer, the authors propose a QoS-aware CCM implementation called CIAO which allows MIC tools to specify these QoS requirements of components in the accompanying metadata.

A multi-tier framework for accessing distributed, heterogeneous spatial data in a federation based EIS [53]

Geo spatial data and GIS components play a leading role in any EIS and have to be integrated into various systems of the EIS. Therefore, a framework, called GIStermFramework, has been developed that allows the integration of GIS components into systems or service based architectures and hide the distributed, heterogeneous geo-spatial data sources. The components of that framework provide a systems and data abstraction that use concepts of the upcoming OpenGIS standardization efforts. Another aim of the framework is that all GIS components can be easily integrated into network based information systems architectures. The framework is implemented in the programming language Java and uses many modern concepts like Java/Beans component technology and a middleware(Java/RMI) based multi tier architecture.

Systematic Aid for Developing Middleware Architectures [55]

Middleware services consist of a number of elements that provide certain nonfunctional properties. Combining more than one middleware service amounts to composing the elements of each service. Addressing all such various middleware issues, the authors have developed an environment that facilitates the design and quality analysis of flexible and configurable middleware architectures, providing the following 3 main features:

- Architecture Description Language(ADL)
- Repository of architectural descriptions.
- Automated tool support.

The authors have been experimenting with the environment on real-world cases in the context of the European Commission's Dependable Systems of Systems project.

An Approach to Designing Reusable Service Frameworks via Virtual Service Machine [60]

Virtual Service Machine is a new service-computing platform that provides developers with a reusable service-based frameworks and applications. It works by taking incoming service requests and converting them to the executable tasks of the underlying middleware or machine that it is running on. The author emphasizes on presenting the functionality and major components inside VSM.

The author finally states VSM reduces cost and time of developing service-based frameworks and applications, and increased their flexibility. It distinguishes itself as one of the first innovative approaches for service-computing platform. It provides service

developers with an alternative method to manufacture their service-based frameworks and applications.

The Design of a Flexible Communications Framework for Next-Generation Middleware [76]

The authors in this paper introduces a flexible, object-oriented communication framework, called BOSSA NOVA which can be integrated into middleware platforms to better support domains such as soft real-time, multimedia, and adaptive mobile systems. This framework provides flexibility in terms of protocol structure and composition, protocol granularity, and concurrency structure. The use of reflective interfaces explicitly facilitates management and dynamic reconfiguration as well as QoS specification and negotiation together with associated resource management in protocol graphs.

The author also states that BOSSA NOVA promotes a reactive style of processing through carrying incoming messages to the application and the receive operation is neither required nor supported. It is also easy to implement the notion of active bindings by outlets that own a thread and notify the application via inter-thread communication or upcalls whenever the next message is due. He also describes how it requires only relatively few abstractions, which are repeatedly applied throughout the architecture to decrease its complexity and increase its usability.

Frameworks for Component-Based Client/Server Computing [81]

The author discusses two major component technologies, CORBA that is an Object Management Group's proposal for a distributed object framework, and DCOM, Microsoft's system for creating and using objects on remote machines while maintaining the common paradigm for interaction among libraries, applications, and systems software provided by COM.

The author states that leveraging CORBA and Java provides the most notable advantages of component technology in the domain of client/server computing. It works well with Java applications, which make very good downloadable clients because of their small size and java's mobile code system.

A Reflective Component-Based & Architecture Aware Framework to Manage Architecture Composition [100]

The author proposes a reflective component-based framework with architecture style awareness for managing architecture composition and constraining adaptation. This framework provides the necessary tools to generate and manipulate the programming model abstractions. It offers a principled way to deal with both introspection and adaptation of basic and composite components. It provides the developers with the ability to choose, extend and modify architecture style managers.

The design of the framework makes intensive use of variants of well-proven and recurrent software development design patterns. It is almost completely implemented but some aspects are still under development and is necessary to finish the transactional adaptation process, based on the architectural restrictions.

Patterns, Frameworks, and Middleware: Their Synergetic Relationships [118]

Patterns, frameworks and middleware are increasingly popular techniques for addressing major challenges the software industry is facing.

Patterns codify reusable designing expertise that provides time-proven solutions to commonly occurring software problems that arise in particular contexts and domains. Frameworks provide both a reusable product-line architecture for a family of related applications and on integrated set of collaborating components that implement concrete realizations of the architectures. The authors in this paper presents an overview of patterns, framework and middleware and describes how these technologies complement each other to enhance reuse and productivity. These technologies provide

- Open Standards
- Strategic focus
- Design reuse
- Implementation reuse

for developing and evolving application software.

Customization of Component-based Object Request Brokers Through Dynamic Reconfiguration [132]

The new component architecture solves the problems of current component architectural knowledge. The author proposes an integrated development environment(IDE) for building component-based applications which presents a virtual development and deployment workspace that is structured according to 5 different views offered to ORB architects, ORB builders, applications programmers, and system integrators. The software architecture view supports the ORB architect in developing a generic ORB builder by providing a visual view to create a new ORB implementation by providing one or more component instances together with their respective component descriptors for each component type. In applying the non-functional view, application programmers specify application-specific policies for the non-functional requirements that were identified during the use-cases analysis of their applications. The component view offers the system integrator the possibility to adapt the ORB at runtime by selecting and replacing component instances. Finally, the interaction refinement view, allows the integration of new protocols and services that were not anticipated for during design of the ORB architecture.

Mediation In Information Systems [138]

Today, the many tools provided by vendors of client-server systems are aiding rapid construction of information systems. There are aggressive efforts in place for establishing standards for middleware, since adoption of standards is expected to direct much future business into one camp of vendors or another. Candidate middleware interface conventions go by names as CORBA (Common Object Request Broker Architecture), DOE (Distributed Objects Everywhere), ILU (Inter-Language Unification), KQML (Knowledge Query and Manipulation Language), OLE (Object Linking and Embedding), Opendoc (Open Document Exchange), PDES (Product Data Exchange suing STEP) and many others.

The author also states that mediated architecture is also attractive to academia. Large, realistic databases are hard to manage in an academic setting. Building effective user interfaces is also an area where industry has excelled and, with a few exceptions, little progress can be made with the background available there. As Mediators, being mainly code, are easier subjects for academic research, the author proposes the formal knowledge in the form of resource schemas, user models and functional specification, can be used to generate mediators or at least help in maintenance.

III. Middleware Solutions

Managing Complexity: Middleware Explained [14]

Several distributed object platforms have recently become popular and extend distributed technologies like Open Group's DCE, Sun's remote procedure call (RPC) interface, and Novell's Netware. Now three major middleware technologies are largely used of which, CORBA is the most comprehensive and explicitly addresses both the general and vertical market aspects of middleware's potential uses.

The Object Request Broker(ORB) is CORBA middleware that establishes the client-server relationships. ORBs allow the developer to choose the most appropriate operating system, execution environment and language to use for each component under construction. In contrast, the Java RMI system assumes the homogeneous environment of the Java virtual machine, which mean the system can take advantage of the Java object model whenever possible, but also means your are largely confined to Java.

The success of the middleware concept has naturally led to the desire to deploy the technology in more diverse and demanding application areas than in traditional, networked environment.

Software Engineering and Middleware: A RoadMap [39]

In this article, the author says "In order to build distributed systems that meet the requirements, software engineers have to know what middleware is available, which is best suited to the problems at hand, and how middleware can be used in the architecture, design, and implementation of distributed systems."

Existing middleware products enable software engineers to build systems that are distributed across a local-area network but state-of-the-art middleware research aims to push this boundary towards Internet-scale distribution, adaptive and reconfigurable middleware for dependable and wireless systems. The author reviews this research and uses this knowledge to derive a software engineering research agenda that will produce the principles, notations, methods and tolls that are needed to support all activities during the life cycle of a software engineering process.

IV. Embedded and Real Time Systems

A New Development Framework Based on Efficient Middleware for Real-Time Embedded Heterogeneous Multicomputers [58]

Application software development for an embedded multi-computing target is complex enough and becomes more complex by considering portability and technology insertion issues. The author proposes a solution to solve such problems by using middleware-based technology and complementary design methodologies and tools. One other recent approach that is proving viable and effective is the Talaris application configuration middleware framework with an application development tool that layers on top of it known as PeakWare for RACE. Applying these concepts to the embedded multiprocessor signal processing domain is both novel and promising. The middleware and tool are introduced and then illustrated by a simplified STAP(Space-time adaptive processing) application that was rapidly prototyped and run on fifteen compute nodes.

A Novel CoDesign Methodology for Real-Time Embedded COTS Multiprocessor-Based Signal Processing Systems [59]

Powerful frameworks exist for each individual phase of this canonical design process, but no single methodology exists which enables these frameworks to work together coherently allowing the output of a framework used in one phase to be consumed by a different framework used in the next phase. A specification and design methodology (SDM) known as 'Search-Explore-Refine' (SER) was developed for an application and technology domain that is different from that of real-time embedded signal processing systems implemented with commercial-off-the shelf multiprocessing hardware and software. The author now proposes a new SDM developed and prototyped based on SER known as MAGIC SDM. It effectively allows the designer to evaluate candidate

architectures and technologies before committing to a technology and be confident that an optimum design has been obtained.

Middleware for Distributed Industrial Real-Time Systems on ATM Networks [99]

The author in this paper addresses the problem of middleware design for constructing ATM LAN based distributed industrial plant monitoring and control systems. In order to achieve on-demand transmission of plant data, the author has developed a concept called selective real-time channels to be supported by MidART(Middleware and network Architecture for Real-Time Industrial applications). MidART is one of the first efforts in introducing the emerging ATM technology into distributed industrial plant control applications. Besides the evaluation of its real-time performance, some of the open issues that we are continuing to address include signaling algorithms at the ATM layer to facilitate the efficient implementation of selective real-time channels, as well as effective QoS mapping algorithms for the RT-ATM component of MidART.

Middleware for Real-Time and Embedded Systems [117]

Distributed real-time (DRE) and embedded systems play an increasingly important role in modern application domains. Some of the most challenging requirements for new and planned DRE systems can be characterized as follows:

- Multiple QoS properties must be satisfied in real time
- Different levels of service are appropriate under different configurations, environmental conditions, and costs
- The levels of service in one dimension must be coordinated with and/or traded-off against the levels of service in other dimensions to meet mission needs.
- The need for autonomous and time-critical application behavior necessitates a flexible distributed system substrate that adapts robustly to dynamic changes in mission requirements and environmental conditions.

The author states “middleware is a strategic element in developing DRE systems, bridging the gap between application programs and the underlying operating systems and network protocol stacks to provide reusable services critical to these systems.”

V. CORBA/ORB

Designing an Efficient and Scalable Server-side Asynchrony Model for CORBA [13]

The author describes in this paper, a strategy for implementing server-side asynchrony model for CORBA. They outline the key design challenges faced when developing an Asynchronous Method Handling(AMH) model for CORBA and how they are resolving these challenges in TAO, a high performance real-time CORBA ORB.

AMH allows servants in middle-tier server applications to store response handlers in a container maintained by the servant application code and return control to the ORB

immediately. This design allows the ORB to start handling a new request while it is waiting for a response from a sink server without requiring a thread for each request/response pairing in a middle-tier server. Any response from a sink server can invoke an immediate response to the client.

Extending CORBA Interfaces with Protocols [15]

The author in this paper stresses the importance of incorporating protocol information into object interface descriptions. They propose extending traditional IDLs with protocol descriptions described using Pi calculus. The information needed for object reuse is now available as part of their interfaces and more precise interoperability checks can be achieved when building up applications. The authors also researched how to add protocol information to CORBA IDLs and the sort of benefits that can be obtained from it.

Exploiting Reflection in Mobile Computing Middleware [16]

Mobile devices face temporary and unannounced loss of network connectivity when they are moved, they are likely to have scarce resources, and they are required to react to frequent changes in the environment. To accommodate such requirements imposed by middleware platforms for mobile computing must be capable of both deployment–time configurability and run-time configurability. The authors show how reflective techniques can be exploited by middleware designers to address these requirements. They describe two projects CARISMA, where reflection is used to allow applications to dynamically customize middleware behavior, and ReMMoC, where reflection is used to deal with heterogeneous middleware technology in mobile environments.

An Original View Mechanism for the CORBA Middleware [18]

The authors proposed a new architecture to implement a complete view mechanism and also described the following advantages

- It doesn't require any modifications at the level of data sources
- Such view mechanisms for the CORBA middleware are easy to implement.
- We just need to give clients a new IDL file, corresponding to the new external schema, and to update the new policy file.
- We don't need to modify neither second-tier(the server objects) nor third-tier of the 3-tier client-server architecture in order to update external schemas.

On server side, new methods may have been added and does not necessarily have the ability to update all the clients but the new proposed architecture has the ability for all old clients and new clients to communicate with the new server.

What is Reflective Middleware? [26]

The author describes reflection to the capability of a system to reason about and act upon itself. More specifically, he defines reflective system as one that provides a representator of its own behavior which is amenable to inspection and adaptation, and is causally connected to the underlying behavior. 'Causally connected' means that changes made to

the self-representation are immediately mirrored in the underlying system's actual state and behavior and vice versa. It can therefore be said that a reflective system is one that supports an associated causally connected self representation.

Distributed Object Computing Platform [34]

The commercial state-of-the-art practice in addressing the interoperability of DBMSs is to build gateways. This approach is quite restricted and provides only a partial solution. None of these systems properly deal with semantic or structural heterogeneity of the stored data. The introduction of object-oriented technology into data management has lifted this restriction. Object-orientation, with its encapsulation and abstraction capabilities, enables the development of wrappers that encapsulate a particular repository and provide a common DBMS-like interface to the rest of the system. The components of the interoperable system may be DBMSs, other types of repositories, or legacy systems. The author describes the usage of CORBA, which provides implementation transparency allowing a client to access an object through its interface defined in IDL, independent of the hardware and software environment in which the object resides. This solves the platform heterogeneity problem. On the other hand, CORBA provides location transparency that allows clients to access objects using their object identifiers independent of their location and communication protocols between the client and the object.

A CORBA Service Enabling Programmer-Friendly Object-Oriented Real-Time Distributed Computing [70]

In this paper the authors discusses a cost-effective approach for facilitating CORBA-complaint TMO-structured RT distributed application programming. It is realized via provision of a middleware and does not require change in the ORB core. The initiation and scheduling of the executions of TMO methods are managed by TMOES/AnyORB which is a CORBA-complaint derivation from the TMO support middleware(TMOSM) model. Its instantiation must reside in every computing node involved in TRdC applications.

The Case for Reflective Middleware [74]

In the past years, software developers created middleware technology to facilitate development of software systems, most notably distributed and Internet-based, to support activities as diverse as scientific computation, information discovery and dissemination, and e-commerce. Middleware mediates interaction between the application and the operating system. CORBA, or Common Object Request Broker Architecture, and Java also hide the differences in the underlying software and hardware platforms, increasing portability and facilitating maintenance, as new version of operating systems are released.

In reflective models, middleware is implemented as a collection of components that can be configured and reconfigured by the application. The middleware interface remains unchanged and may support applications developed for traditional middleware. In

addition, the systems and application code may inspect the internal configuration of the middleware and, if needed reconfigure it to adapt to changes in the environment through meta-interfaces.

Integration of QoS-Enabled Distributed Object Computing Middleware for Developing Next-Generation Distributed Application [77]

QoS-enabled DOC middleware facilitates ease of development and deployment of applications that can leverage the underlying networking technology or end-system QoS architecture. Distributed applications are increasingly being composed out of flexible and modular reusable software components and services, instead of being programmed entirely from scratch via lower-level, proprietary tools. The QoS-enabled CORBA ORB and Audio/Video Streaming Service middleware developed using ACE and TAO and the QuO middleware help to simplify and coordinate such applications.

The Performance of a Real-time I/O Subsystem for QoS-enabled ORB Middleware [79]

The design and performance of a real-time I/O(RIO) subsystem optimized for QoS-enabled TAO ORB endsystems that support high-performance and real-time applications running on off-the-shelf hardware and software. RIO is designed to operate with high-performance interfaces such as the 1.2 Gbps ATM port interconnect controller. The author mentions that the following lessons were learnt from the integration of RIO and TAO:

- Vertical integration of ORB endsystems is essential for end-to-end priority preservation.
- Schedule-driven protocol processing reduces jitter significantly.

The TAO research effort has influenced the OMG Realtime CORBA specification, which was recently adopted as CORBA standard.

A Combat Management System Middleware Based on CORBA [80]

The author discusses a new middleware as part of a generic CMS development project named GENESIS. This middleware satisfies the requirements of CMS applications and runs in a heterogeneous distributed computing environment. This middleware is implemented by sing CORBA and its Event and Naming services. It uses a portable operating system wrapper framework named Adaptive Communication Environment (ACE). During the development process, it has been observed that the right mixture of OTS and in-house software cuts down the development time to a great extent without sacrificing functionality, performance and reliability requirements.

Design of a Middleware Service for Scalable Wide Area Network Applications [87]

The authors in this paper presents a middleware design that can adapt to changes in its environment better than current systems. He says, “it includes services not present in

current implementation of commercial ORBs or PRC systems." He also states that it provides applications with shared common facilities to achieve the desired scalability, fault tolerance, adaptability and load balancing properties, and need to be defined on a per application basis. This design is not tied to any particular distributed paradigm but allows transparent interoperability between applications originally written for them, minimizing the performance penalty incurred upon the application.

An Object Infrastructure For Internet Middleware [88]

As the industry is moving towards object technology as a development and operations model to help control and simplify the application environment, and its dependency on the Internet, the demand for middleware architectures for strategic solutions is increasing. The convergence of the web and object technology have led to 'object web' which assumes universal connectivity, broadly distributed environments, and cross-platform interoperation of both infrastructure and applications.

The authors also describes some key technical concepts of Component broker as follows:

- The Component Broker programming model
- Single-level store
- Composed business objects
- System management
- Managed objects
- Workload balancing, and
- Legacy integration

On the Role of Middleware in Architecture-based Software Development [89]

Middleware technologies such as CORBA, COM and RMI provide a set of predefined services for enabling component composition and interaction. However, the potential role of such services in the implementations of software architectures is not well understood. So the author has investigated a set of techniques and conducted preliminary case studies involving a particular architectural style, C2, and its implementation infrastructure. This approach directly exploits architectural constructs and provides a principled, repeatable solution to the problem of bridging middleware.

They have employed sets of both commercial and research technologies to test the hypothesis that software connectors are the proper mechanisms for supporting middleware-based implementation of architectures. The results indicate that it is possible to provide compositional inter-middleware connectors such that pairwise solutions can be avoided.

Middleware for Building Adaptive Systems Via Configuration [102]

Large and adaptive systems can be created by integrating Commercial off-the-shelf devices(COTS). The basic integration method is configuration and the finite set of

integration parameters can be set to a definite values. These parameters are static and unlike the device dynamic state, they do not change during the device's normal operation.

The authors in this paper propose a technique called Service Grammer for developing middleware for a given domain. It is based on the idea that algorithms for implementing end-to-end requirements can be expressed as the enforcement of a set of fundamental requirements in a Requirements Library. These requirements are upon the existence of definite high-level relationships between configuration parameters of various devices.

Applying a Scalable CORBA Event Service to large-scale Distributed Interactive Simulations [104]

The authors in this paper make the following five different contributions to the design, implementation and performance measurement of distributed interactive simulation systems:

- Describing how the CORBA event service can be implemented to support key QoS features.
- Illustrating how to extend the CORBA Event Service so that it is better suited for distributed interactive simulations.
- Describing how to develop efficient event dispatching and scheduling mechanisms that can sustain high throughput.
- Describing how to use multicast protocols to reduce network traffic transparently and to improve system scalability.
- Illustrating how an Event Service framework can be strategized to support configurations that facilitate high throughput, predictable bounded latency, or some combination of each.

An Overview of the Real-Time CORBA Specification [119]

In complex real-time systems, the authors propose the use of CORBA to help decrease the cycle time and effort required to develop high-quality systems by composing applications using reusable software component services rather than building them entirely from scratch. The Real-time CORBA specification includes features to manage CPU, network and memory resources. The RT-CORBA specification identifies capabilities that ORB end systems must integrate and manage vertically, from network interface to application layer and horizontally from peer to peer, to ensure end-to-end predictable behavior for activities that follow between CORBA clients and servers.

The author lists the following explicit capabilities of RT-CORBA:

- Communication infrastructure resource management
- OS Scheduling mechanisms
- Real-time ORB end system
- Real-time services and applications

A CORBA-based Application Service Middleware Architecture and Implementation [133]

The authors in this paper, present an architecture and initial implementation of a middleware CORBA based Application Service middleware(CASM) that satisfies user requests according to expected requirements, distributes networked resources lad beyond the local environment, and respects the logical constraints of different domains. The authors state that the implementation of CASM is governed by the following three goals:

- To serve as a better than best effort middleware to the application layer, selecting the most appropriate resource to satisfy a request among the variety of resource of the same type in the network environment.
- To reflect the application requirement with respect to usage/control of resources, in accordance with the user/task priority
- To require only minor changes to existing applications in order to achieve benefits from CASM.

Applying Reflective Middleware Techniques to Optimize a QoS-enabled CORBA Component Model Implementation [136]

As conventional CORBA Component model (CCM) focus on functionality rather than adaptive quality-of-service, it is unsuitable for next-generation applications with demanding QoS requirements. Based on ongoing research on CORBA and the CCM, authors believe the application of reflective techniques to component middleware will provide a dynamically adaptive and (re) configurable framework for COTS software that is well-suited for the QoS demands of next-generation applications. This paper presents three contributions to the study of middleware for QoS-enabled component-based applications.

VI. ADA

An Object-Oriented, Distributed Architecture for Large Scale Ada Systems [78]

This paper presents an architectural model that ideally suites for the description of large, distributed command and control systems. This model is organized around multiple dimensions of software architecture and is used to describe the software architecture of a family of automated air traffic control systems. The proposed model is composed of four perspectives:

- The logical view, which is the object model of the design.
- The dynamic view, which captures the concurrency and synchronization aspects of the design.
- The physical view, which describes the mapping of the software onto the hardware and reflects its distributed aspect.
- The static view, which describes the organization of the software in the development environment.

VII. DB-Query Middleware

Use of a Component Architecture in Integrating Relational and Non-Relational Storage Systems [5]

A fundamental service provided by the products offered by various software vendors is that of managing the persistent storage of application data. Rather than attempting to build one monolithic storage engine, which address the needs of all application, why not use several engines together, each tuned to a particular kind of data.

Toward Scalability and Interoperability of Heterogeneous Information Sources [27]

In this paper, an algorithm that transforms a relational database schema to an object oriented database schema is presented. The algorithm is capable of generating typical OODB schemas that may contain class hierarchies, composition hierarchies, set attributes and subtuple constructs.
There are at least two issues that need further investigation.

- The creation of methods in transformed OODB schema, especially when update queries are also allowed to be issued by the users of OODB front end.
- To improve the applicability of the transformation rules.

A Middleware System which Intelligently Caches Query Results [31]

This paper describes, how caching is used to improve performance in the Accessible Business Rules framework (ABR) for IBM's Websphere. ABR is a middleware system, which enables application writers to build applications business rules such as time and situation - variable parts of their business logic are externally applied. General – Purpose Software cache (GPS cache) and Data Update Propagation (DUP) are used in ABR. The techniques for caching queries in ABR can be deployed in other query based environments as well.

Guest Editor's Introduction to the Special Issue on Heterogeneous databases [38]

This paper summarizes the HDB work for the readers of computing Surveys. The HDB work is divided into 3 layers, at the is the 'access' layer, dealing with query languages and data model transaction, second is the operation layer deals with the data – related heterogeneity or schema integration, the bottom layer, the TP layer, focuses on transaction processing, in particular updating autonomous HDBS. This includes work on concurrency control and crash recovery. The challenges in the HDB work are:

- Maintenance of global serializability.
- The preservation of autonomy in the component databases.
- Semantic heterogeneity of operation layer.

Extended Data Type Support in Distributed DBMS products: A Technology Assessment and Forecast [51]

This report addresses support for extended data types and distributed data management in deployable database management systems and provides a general technology assessment and forecast of DBMS support, addressing capabilities of both relational DBMSs and object-oriented DBMS and includes a discussion of issues in supporting extended data types and distribution, the current state of the art in DBMS products, and expected developments in those systems. Finally, it provides a rough timetable for inclusion of the needed technology in commercially - available database systems.

Middleware Object Query Processing with Deferred Updates and Autonomous Sources [69]

This paper presented a query processing algorithm called DECAF for use in middleware object query systems that use an object cache and in which updates to the underlying databases that don't through the object cache are supported. The DECAF algorithm attempts to push down query predicates to the underlying DBMSs to take advantage of the query processing capabilities and to reduce the amount of data transferred from these systems to the object cache. Preliminary performance measurements show that the DECAF algorithm outperforms a naïve approach in a variety of workload combinations.

Enterprise Objects Framework: a second generation object – relational enabler [73]

The Enterprise Object Framework product is a second-generation product bringing the benefits of object-oriented programming to relational database application development. It maintains a clear separation between the user interface, business objects and the database server. The architecture provides the framework for quickly deploying object-oriented business applications on a wide-scale basis. Its open architecture allows for easy customization and extension to non relational data storage engines. It provides a robust and flexible infrastructure for ad-hoc as well as production workflow applications. In future it will be extended in many dimensions.

The State of the Art in Distributed Query Processing [75]

This paper presents the state of the art of query processing for distributed database and information systems and shows how this works in different kinds of distributed systems such as client-server middleware (multi-tier) and heterogeneous database systems. The paper presents the 'textbook' architecture for distributed query processing and a series of techniques that are particularly useful for distributed database systems. These techniques include special join techniques, techniques to exploit intra-query parallelism, to reduce communication costs and to exploit caching and replication of data. Most of this paper was focused on structural (i.e. relational) data.

An Evaluation of Object – Oriented DBMS Developments 1994 Edition [84]

This report begins by describing the limitations of conventional relational DBMSs that led to the emergence of ODBMS technology. In this paper the author indicates that future information systems will increasingly be driven by requirements for the integration of heterogeneous components, both programs and databases, and the easy the easy reuse of such components, to address new requirements.

Distributed Object Management [85]

In this paper, the concepts of distributed object management, and identify its role in the development of these, interoperable systems. The authors have briefly described the requirements for DOM technology, the aspects of object technologies requiring extension in this new environment, and some work being performed at GTE laboratories to address some of the related technical issues.

A 'RISC' Object model for Object System Interoperation: Concepts and Applications [86]

This report describes a 'reduced instruction set' or 'RISC' object model, to address the problem of object model interoperability. Serving in a descriptive role, the RISC object model could be the basis for understanding the differences among object models, for defining mappings between different object models, or even for defining new, application-specific models.

Addressing Component Interoperability in the OMG Object Model [103]

The proliferation of new object models and programming languages leads to believe that the 'one model fits all' doctrine of interoperability may not be adequate for integrating future information systems. To avoid future interoperability problems, it is important to address component interoperability issues explicitly. In this paper, the authors discussed the OMG object model's 'core + components' approach in addressing component interoperability and introduced a 'reduced set' or 'RISC' variant to the 'core + components' approach that addresses component interoperability in a more explicit way.

Construction of a Relational Front-end for Object-Oriented Database Systems [92]

In this paper, authors have proposed a solution for the construction of a relational front-end for Object-Oriented database systems. In particular, discussed the transformation of OODB schemas to relational schemas and the translation of relational queries to OODB queries. By introducing relational predicate graphs and OODB predicate graphs to facilitate the query translation, the authors have shown that OODB predicate graphs contain more information than the corresponding query graphs.

Processing Hierarchical Queries in Heterogeneous Environment [93]

The authors have developed a generic hierarchical database system, which contains many common features of existing hierarchical database systems and some object-oriented

database systems. Based on GHDBS as receiving database system, developed rules that correctly translate each relational query to one or more GHDBS queries and also provided the rules that can guarantee a minimum number of target queries in case when a (user submitted) source query needs to be translated to multiple target queries.

Transformation of Relational Schemas to Object-Oriented Schemas [94]

In this paper, the authors have presented an algorithm that transforms schemas of relational databases to schemas of object-oriented databases (OODB). This algorithm has the following special features:

- It transforms schemas between two commercial database systems.
- It initiates limited interactive sessions between the schema transformer and its user when additional input to the schema transformer is necessary.
- The transformed schemas are typical OODB schemas.

MOCHA: A Self-Extensible Database Middleware System for Distributed Data Sources [114]

A new self-extensible database middleware system designed to interconnect distributed data sources; MOCHA is presented in this paper. It's designed to scale to large environments and is based on the idea that some of the user-defined functionality in the system should be deployed by the middleware system itself. MOCHA has been implemented in Java and runs on top of Informix and Oracle. Experiments with the MOCHA show that selecting the right strategy and right site to execute the operators can increase query performance and also demonstrated that the Volume Reduction Factor (VRF) is a more accurate cost metric for distributed processing than the standard metric based on selectivity factor and result cardinality because VRF also considers the volume of data movement.

Report on the Workshop on Heterogeneous Database Systems held at Northwestern University [116]

Research into the interoperability of heterogeneous database systems plays an important role in the development of high-level open systems. The objective of this workshop was to explore current approaches to interoperability of autonomous information systems and to identify the most important research directions to be pursued in this area. This report summarizes authors' discussions and broadly classifies the issues into the following categories: (a). Semantic Heterogeneity and Schema Integration. (b). The Role of Object-Oriented Approach. (c). Transaction Processing. (d). Query Optimization. (e). Standardization Efforts.

Adaptable Query Optimization and Evaluation in Temporal Middleware [122]

This paper proposes a new approach for temporal query optimization and evaluation: using a middleware component on top of a conventional DBMS. This component accepts

temporal SQL statements and produces a corresponding query plan consisting of algebraic as well as regular SQL parts. The algebraic parts are processed by the middleware, while the SQL parts are parts are processed by the DBMS. The paper describes the architecture and implementation of the temporal middleware component, termed TANGO. Experiments with the system demonstrate the utility of the middleware's internal processing capability and its cost-based mechanism for distributing the processing between the middleware and the underlying DBMS.

Amalgamating Knowledge Bases [126]

In this paper author presents a uniform theoretical framework, based on *annotated logics*, for amalgamating multiple knowledge bases when these knowledge bases contain inconsistencies, uncertainties, and non-monotonic modes of negation and shows that annotated logic may be used, with some modifications, to mediate between different knowledge bases.

Some Practical Advice for Dealing with Semantic Heterogeneity in Federated Database Systems [134]

Integrating semantically heterogeneous databases is costly, since it's a largely intellectual process involving extracting semantics, expressing them as metadata, and resolving inconsistencies among them for overlapping data. Thus, it's important to avoid expending the effort wherever possible. Authors suggest two ways to make this effort practical, based on experience with very large system integration efforts and discussions with others engaged in these efforts.

- Narrow the scope of the problem and prioritize the information to be federated, based on requirements specification for federated users.
- Exploit the various tools and enabling technologies that are available, including semantic and object models, Repository, CASE tools, and standards.

A Middleware Implementation for Active Rules for ODBMS [145]

Utilizing the recent development of CORBA and ODMG standards, a middleware approach to provide active rules systems for heterogeneous ODBMS is presented in this paper. Comparing to other architecture of rule systems authors reduced the overhead involved in runtime monitoring all the changes of database states by compiling database programs. But, still there are many things left for the further optimization.

Translation of Object -Oriented Queries to Relational Queries [146]

In this paper, authors proposed a formal method for translating OODB queries to equivalent relational queries. The translation is achieved through the use of the relational predicate graphs and the OODB predicate graphs. The OODB query language, which is used in the paper, allows path expressions with possibly multiple quantifiers. Path expressions are the key construct in many proposed OODB query languages.

VIII. Enterprise Architecture & Solutions

Proxy – Server Architectures for OLAP [65]

Data warehouses are not properly served their purpose because of the growth of the Internet & WWW. The proposed architecture for OLAP cache server (OCS) (≅ Of a proxy server for web documents, but designed to accommodate data from warehouses & support OLAP operations) , by considering the performance issues against the existing systems, is considered to offer increased autonomy at remote clients, substantial network traffic services, better suitability, lower response times and is complement any both to existing OLAP cache systems & distributed OLAP approaches. Currently working on different issues like connecting to the underlying DBMS; support multiple unrelated data warehouses by same OCS.

An Efficient Middleware Architecture Supporting Time-Triggered Objects and an NT-based Implementation [71]

TMO is a natural & syntactically small but semantically powerful extension of the OO design & implementation techniques & been established to remove the limitation of the conventional object structuring techniques in developing applications containing real time (RT) distributed computing components. An efficient & cost-effective middleware architecture called TMOSM, which can be adapted to various COTS platforms, has been proposed. A prototype implementation of TMOSM on WinNT is developed to validate the proposition that it can support highly efficient and economic development of complex distributed RT applications with action timings of the precision in the range of around ten msec.

Enterprise Information Architectures - They're Finally Changing [91]

Because of the changes in the business environment, aggressive introductions in business process reengineering, new IT architecture has been introduced which demands revised assumptions about the design and deployment of databases. This paper reviews the components of the architectural shift now in process, and offers strategic planning assumptions for database professionals.

IX. QoS

The Design of an Application Programming Interface for QoS-based Multimedia Middleware [23]

Research at the boundary between multimedia middleware and application has to follow two aspects:

- Homogeneous and adjusted multimedia communication interface is required that offers unaltered performance characteristics with minimal overhead as the underlying communication subsystem delivered it from the network.

- The interface is required to be independent of multimedia applications as far as possible the communication subsystem.

The author proposes a developed and implemented API supports flexible communication services handling QoS specification, transparently hiding communications relevant features from applications. This API is independent of the underlying communication and multimedia middleware, which results in its portability on other communication architectures.

General Framework for the Description of QoS in UML [32]

A QoS framework provides support to ensure consistency in modeling various qualities of service. It supports a general categorization of different kinds of QoS including those that are fixed at design time as well as ones that are managed dynamically. The authors state that the general QoS framework must provide support for description of

- QoS characteristics that define the vocabulary of QoS expressions.
- QoS forces for the description of QoS constraints.
- QoS levels that support the component & system functions.
- QoS adaptation behavior that allows updating quality levels.

Communication Middleware and Software for QoS Control in Distributed Real-Time Environments [54]

In this paper, the author discusses how to develop an integrated set of network service components, admission control & resource management techniques, and a suite of middleware services to support multi-dimensional QoS as an end-to-end basis & exporting a unified, well defined API for distributed real-time environments. The author also proposes to develop a representative real-time multimedia application to demonstrate the key functional, highly predictable, timely & dependable characteristics of the network software implemented. It not only improves QoS of a Tm-based distributed system, but also expands applications domains to real-time controls and digital continuous-media transmission.

QualProbes: Middleware QoS Profiling Services for Configuring Adaptive Applications [83]

Middleware layer may assist distributed systems by controlling behavior of the applications so that they adapt and reconfigure themselves. The author proposes QualProbes, a set of middleware QoS probing & profiting service to discover such relationships at runtime.

QualProbes services are able to precisely capture the effects made to the critical performance criterion when resource availability varies and thus enable more effective control of the application to adapt to resource variations. It also details QualProbes services like application model, dependency tree model for application QoS parameters, and the QoS profiling algorithm implemented in the QualProbes services kernel. In

addition they presented experimental results with the omnitrack, a complex multimedia application to verify the different approaches effective in understanding of the application.

Using Analytic Models for Predicting Middleware Performance [107]

As interoperability is facilitated by middleware on heterogeneous systems, the authors focus on analytic modeling of middleware-based systems that can be used in the performance engineering of CORBA-based client server applications. The authors applied the layered queuing network model to 2 types of middleware interaction: Handle driven & forwarding architectures. The authors state, "The LQN model predictions are reasonably acceptable, with most of the errors less than 12 %". The bottleneck in the system was also correctly predicted. The largest error of –18% occurred when both the services demand at the servers and the population was large. In such cases, the analytic model tends to predict shorter response times because it cannot capture a convoy effect that appears in the real system due to its deterministic behavior.

Evaluating and Optimizing Thread Pool Strategies for Real-Time CORBA [110]

This paper primarily focuses on contributing towards the evaluation of techniques for improving the quality of implementation of Real Time CORBA thread pools.
The authors stress on the fact that, RT-CORBA users need to understand the trade-offs between the different strategies for optimizing for certain applications. He details the Half Sync/Half Async and the leader/followers strategies for implementing RT-CORBA thread pools. They evaluate these strategies using several different factors & present results that illustrate empirically how different thread pool implementation strategies perform in different ORB configurations.

Towards Automatic Synthesis of QoS Preserving Implementations from Object-Oriented Design Models [115]

In this article, the author addresses the problem of synthesizing QoS preserving implementations from object-oriented design models. He presents an overview of his research in making it applicable to uniprocessor systems with hard real time constraints. The automatic synthesis process releases a designer from the burden of selecting various implementation artifacts, in the same way automatic code generation releases a designer form the burden of deciding how to implement modeling abstractions.
The author states, "synthesis problem gets tougher as we go from uniprocessor to multiprocessor & distributed systems."

Challenges Designing Next-Generation Middleware Systems [131]

Distributed applications nonfunctional requirements are related to the applications quality of service (QoS). The traditional notion of QoS largely pertains to performance measures like throughput & latency. Increasingly complex QoS requirements in distributed applications require policy-driven middleware services. Dynamic security requirements,

user & component mobility, ad hoc networking of devices, and context-aware computing pose new challenges motivating this approach.

Stuck in the Middle: Challenges and Trends in Optimizing Middleware [144]

There has been increasing market for the middlewares over the past few years due to the following two major reasons:

- It greatly increases the productivity by encapsulating common tasks in a set of frameworks & components.
- Many recent business trends require a great deal of application integration to succeed, and middleware is the technology of choice when it comes to integration.

As the industries shifting from using a homogeneous programming environment to heterogeneous programming environments, there is an increasing need for new IDEs to provide visual metaphors for composition & contain pallets of components that can be used for common integration tasks. The author finally states that there is a great deal of research that is required to come up with better models & tools for understanding compusystems (systems containing several computational components), new programming models to construct, maintain & optimize them & adaptive middleware to optimize based upon the dynamic state of the system.

X.Ontology

WebODE: a Scalable Workbench for Ontological Engineering [3]

As there is a need for a common workbench to ensure a wide acceptance & use of ontological technology, the authors describe three main areas of focus:

- Ontology development, management & support.
- Ontology middleware services.
- Ontology-based applications development suites.

They also present to us WebODE as a scalable ontological engineering workbench that gives support to activities from the first two areas of the workbench previous identified. It provides varied ontology related services & supports ontology development process. It's built on an application server basis, which provides high extensibility & usability by allowing the addition of new services & the use of existing services like webpicker, ontomerge & ontocatalog.

This workbench has also been successfully used in B2B &B2C ontology creation and reengineering projects.

An Extensible Approach for Modeling Ontologies in RDF (S) [123]

The author defines RDF (S) as "an abstract data model that defines relationships between resources called RDF, in a similar fashion as semantic nets." It consists of two closely related parts: RDF & RDF schema. The foundation of RDF (S) is laid out by RDF, which defines basic entities, like resources, properties & statements. Anything in RDF (S) is a resource.

The authors have presented a new approach towards engineering ontologies extending the general arguments made for ODE & ontolingua in the web formats, RDF & RDF (S). They have thus defined a methodology that classifies axiom s into axiom types according to their semantic meaning. Each type receives an object representation that abstracts from scooping issues and is easily representable in RDF (S).

XI. Actors

Customization and Composition of Distributed Objects: Middleware Abstractions for Policy Management [4]

As approaches such as CORBS & RMI lack the tools to verify semantic compositions, stubs & skeletons do not generate proper invoking of interfaces; researchers have introduced the concept of architecture description languages (ADLs). Such ADL specifications define software architecture in terms of a collection of connectors, which describe how components are integrated into the architecture. While connectors define the mechanisms by which clients are matched to the server, the nature of the execution environment requires policies like cluster management & secure interactions.
In this paper, the author describes Distributed Connection Language (DCL), which is an Actor based architecture description language for specifying distributed systems. They also provide a high level description of semantics of DCL abstractions.

XII. Client/Server:

The Architecture of Global Access [7]

The author in this paper, defines five component parts of the Internet as follows:

- User and real-world interfaces
- Applications
- Data
- Delivery system
- Middleware

These five components result in new designs which are developed in a distributed manner with local decisions benefiting, the interest of the self-organization. The author also states that the internet needs a comprehensive distribute dfr4oamework and the object management group (OMG)'s framework, Microsoft's DCOM and Sun's Java Bean or IBM's arabica support such framework.

Selecting the Appropriate Middleware for Your Web-To-Database Application [49]

The author discusses the following popular and emerging methods for webpage-database communication on a NT platform:

- Perl Scripting – It is a standard programming language and runs relatively quick. Also standard scripts can be slightly modified to meet the needs. The negative side of it being, it requires programming changes for every change to the database or the interface.
- IDC files – IDC files are simple to create and update and laos doesn't need a programmer. The negative side being, poor documentation, limited to MS-IIS-Web Server and no client-state management.
- Tango – It comes with a development environment with use wizards, drag-and-drop database table field names and built-in error checking. The negative side being, it needs a specialist to design, no paper documentation and need to run a Tango server on the web server.
- Cold Fusion: It doesn't require any special editor and can use any HTML editor. It is extremely flexible and powerful and allows reuse of code/HTML. The negative side being, it needs an expert to write sophisticated queries and the server runs on NT platform only. Documentation accompanying the product is not comprehensive.

OFTT: A Fault Tolerance Middleware Toolkit for Process Monitoring and Control Windows NT Applications [50]

OFTT(OLE Fault Tolerance Technology) is meant to provide application developers with a reusable, easily-integrated, OPC – 33 complaint middleware toolkit such that applications can be made fault tolerant with minimal modifications.
The OFTT engine described in this article is Microsoft Windows NT based and performs the following functions:

- Role Management: Whether it is a primary or the backup.
- Failure Detection: Monitoring the status of all software components that are linked with the fault tolerance interface module.
- Recovery Management: Recovery from a detected failure.

The fault tolerance interface module takes care of check-pointing the application state, monitoring the status of the application and communicating with the OFTT engine.

Development of Distributed and Client/Server Object-Oriented Applications Industry Solutions [52]

The author defines distributed application as one which can be divided into parts each of which can run on a different computer within a network and communicate with each other. He describes three kinds of distributed applications:

- Point-to-point that allows at most two parts of each other to communicate with each other at any one point in time
- Client/Server application that consists of two types of parts, the Server and Client

- Collaborative application that has more than two parts and a varying of them interacting with each other.

He identifies some important characteristics of distributed application as:

- Locality
- Environment
- Interoperability
- Integration
- Parallelism
- Mobility
- Complexity

And broadly classifies them into inter-application and intra-application distribution, based on whether they support development of the application as a whole or integration of systems.

Where is Client/Server Software Headed? [82]

The author states in this article that the Industrial software is being driven by document-centric design, reusable component implementation based on industry standards, and end user programming. He gives a taxonomy in the industrial flavor of distributed computing and broadly categorizes into two principal parts:

- Decision support systems
- On-line transaction processing

He also explains how middleware proves to be the success maker, as it becomes the infrastructure of all applications ranging from home computing to large scale enterprise computing. He further describes the two essential architectural approaches:

- OLE/COM from Microsoft
- OpenDoc/DSOM from component Integration laboratories

He also highlights the importance of CORBA and describes that it is a model of distributed object-oriented computing and has been followed by SOM/DSOM and not OLE/COM.

Intermediary Architecture: Interposing Middleware Object Services between Web Client and Server [128]

The author proposes a Intermediary architecture which is a pattern for combining two of the component architecture frameworks, Web or ORBs. The main components of this architecture are

- URL Interceptor which enables insertion of side effect behaviors before or after client send, server receive, server send or client receive operations.
- Intermediary services which implement some behavior that a third party can specify remotely

- Service support infrastructure that may share a common service support infrastructure comprising of a
    a. System Management
    b. Metadata repository
    c. Trader
    d. Security infrastructure
    e. Management GUI

XIII. Agents

DBIS-Toolkit: Adaptive Middleware for Large Scale Data Delivery [2]

The two major features of a DBIS framework are

- It incorporates a number of different options for data delivery, including traditional request-response, publish/subscribe, broadcast disks, and on-demand broadcast.
- It's based on the notion of network transparency, which allows different data delivery mechanisms to be mixed-and-matched with in a single application.

The DBIS tool kit consists of four main components:

- Data source (DS) library
- Client library
- Information Booker (IB)
- Information Booker Master

In addition, it contains a flexible performance monitoring capability that can be used to graphically display real time performance metrics such as bandwidth & CPU utilization and response times on a per-IB basis.

Agents and GUIs from Task Models [36]

As GUIs are meant to support some particular set of human tasks, the author states that a designer of a GUI is expected to elicit & formalize task models.
A wide spectrum approaches have been explored to incorporate task models into the GUI design process, ranging from informal, non computational to implementations automatically generated from a formal task model.

The author proposes such a task-centered GUI editor called VAMPIRE (for Visual Model-based Pick-and-place Interface Editor), which generates an explicit task-GUI mapping as output, and captures the relationship between a task model & a GUI as a natural by-product of using the tool.

He also proposes a code generated BATS (for bridging agents, tasks & softwares), which is a batch program that takes 3 XML documents as inputs: a task model, a task-GUI mapping, a GUI specification & outputs a Java implementation of the agent interface needed for COLLAGEN agent.

A Comparison of Mobile Agent and Client-Server Paradigms for Information Retrieval Task in Virtual Enterprises [57]

With the emergence of next generation enterprises and their dependencies on wireless technologies & devices, it is essential that we automate information retrieval tasks for such users.

Middleware being important part of the software and IT infrastructure of next generation enterprises, the authors argue that, it is required to design information systems with the assumption that mobile & wireless users on the norm rather than the exception. However, the author is concerned about the fact that existing commercial middleware platforms, like CORBA & DCOM, do not provide the functions commonly requires.

Thus, the author proposes a newly developed analytical model to qualify the performance benefits of using mobile agent technology rather than traditional client server techniques to retrieve information on behalf of a mobile user. He also continues to investigate integration of mobile agent technology with CORBA-based Mobitrix platform.

Global Change Master Directory: Object-Oriented Active Asynchronous Transaction Management in a Federated Environment Using Data Agents [96]

Global change Master Directory (GCMD) is a repository that contains the information on the changing environment collected by various agencies including the United States government agency global change data center (GCDC) at NASA. The GCMD was initially meant to support the catalog interoperability effort (CI), which strives at facilitating interactions among science data catalyzing systems.

The author describes a new system called MD8 whose objective is to promote the GCMD from a client-server architecture to a federated architecture that is totally automated by providing sites participating in the IDN network the necessary autonomy needed to maintain their system. The MD8 system implements an asynchronous transaction protocol that addresses the issues of dual ownership of entries, looking for updates and temporary inconsistency. The asynchronous transaction model resolves the issues of network failure, availability & allows IDN partners to share their metadata.

Architecture and Performance Evaluation of a Massive Multi-Agent System [143]

Multi-agent systems have been studied, addressed, by several researchers and applied to the area of electronic commerce (EC) in the recent times. The author in this paper describes such a multi-agent system ‘Tabi Can’, which is a commercial service on the Internet. He thus proposes a mechanism for controlling the memory & CPU in multi-agent systems where thousands of systems interact with each other on a single server. He states “the scheduling activities of agents in a specific way that the throughput of agent interactions can be kept to a constant value in response to an increase in the consumer agents.”

An Agent Framework Based Distributed Object [148]

In order to facilitate openness & resource sharing in distributed systems, the author refers to an agent communication language – KQML that has been implemented in a framework.
As many researchers focused on using agent & multi-agent system theories, the author proposes a framework, which is designed and implement on starbus, which is a computing platform compliant with CORBA 2.3 specification. He also describes how KQML can be used for querying and manipulating knowledge & bring distributed system the ability of open interoperability.

XIV. Issues

Adaptive Middleware [1]

The author here defines middleware as an abstract interface that gives an application dev eloper a uniform view of low-level operating systems and networks. He also says that in traditional view, middleware is a means for gluing together applications components that comply with certain interoperability requirements. He highlights 2 important aspects of the new generation middleware:

- Dynamic customizability
- High-level abstractions

Dynamic customizability provides the flexibility required to adapt the middleware for resource-aware mobile applications, whereas, high-level abstractions simplify the problem of expressing complex interaction patterns by enabling application developers to specify interactions.

As middleware technologies mature over the years, it shall facilitate seamless integration of the physical world and the computational world, where every physical object is computationally active and networked.

Mastering the Middleware Muddle [12]

Enterprise application integration issues, the nuances of competing middleware products from lots of vendors and several overlapping standards activities all contribute to the confusion and uncertainty. To understand the middleware muddle, the author reviews the current best practice in each of the following areas.

Processes: Development has taken place in three primary areas.

- Modeling notations.
- Methods for designing distributed systems.
- Life cycle approaches.

Infrastructure: Major development has transpired in

- Component models: The two most popular models evolved are Microsoft based COM (Component Object Model) & CORBA on different platforms.
- Component services: Most recently a Java based component model for enterprise applications has also been developed. The ETB specification defines a server-side component model for Java.

Tools: Broadly speaking the various developments happening in tools are based on the readers supplying tools.

They can be categorized as:

- Smaller venders offering focused solutions
- Infrastructure vendors making their infrastructure products more usable.
- Independent enterprise application-development tool vendors

Issues in Supporting Event-Based Architectural Styles [19]

The author in this paper states, "event-based style presents interesting characteristics for the development of distributed, highly decoupled systems." He explains how each of the infrastructures, make specific assumptions on the structure of notifications, on the mechanism that allows components to declare in their interest in same event; and on the way scalability of architectures is supported.

After analyzing the advantages & drawbacks in each approach, he thinks that the knowledge of these styles can be profitably used when the architecture of an application is defined. He therefore thinks that defining them explicitly in terms of architectural elements, so that they can provide guidelines to application developers and support transition of an architecture into an implementation on top of a selected infrastructure.

Next Generation Middleware: Requirements, Architecture, and Prototypes [37]

With the changes in the industry and need for new applications, including support for multimedia, real time requirements, and mobility, there is an increasing need for defining a new architecture for open distributed systems and next generation middlewares.

The paper focuses on reporting the new middleware manifesto and proposes the different prototypes accommodating the changing needs. The new manifesto proposes the next generation middleware be run-time configurable and allow inspection and adaptation of underlying software which can be achieved based on the marriage of reflections and components. He relates to the following research prototypes:

- MULTE-ORB: This is an adaptive multimedia ORB that is based on flexible protocol framework Da CaPo. This allows dynamic selection, configuration and reconfiguration of protocol modules to dynamically shape the functionality of a protocol.
- OPEN-ORB: The open-ORB Python prototype implements prototype of the architecture under development of the Open-ORB project. It is a reflective

middleware based on components. It supports a binding framework providing explicit open binding.

- Flexinet: The Ansa flexinet platform is a Java based toolkit for creating and reconfiguring ORBs. It allows programmers to tailor the platform for a particular application domain or deployment scenario. It is focused at operational interaction.
- GOPI: GOPI (Generic Object Platform Infrastructure) is a distributed object based middleware platform that supports multimedia applications using stream interfaces, explicit bindings, third party binding, and QoS(Quality of Support) support.
- TAO ORB: TAO is a freely available open source Corba implementation with special efforts on high performance and predictable real-time QoS.

Issues in the design of Adaptive Middleware Load Balancing [106]

Many load balancing middlewares, used in distributed systems are currently geared towards specific usecases & environments. This paper primarily addresses all the service inadequacies, such as server side transparency centralized load balancing, sole support for stateless replication, fixed load monitoring granularities, lack of fault tolerant load balancing, non – extensible load balancing algorithm, and simplistic replica management. The author believes that the above features are essential to implement a generalized, highly effective & optimized adaptive CORBA load balancing service. He also states

- CORBA portable interceptors mechanisms can support transparent server side load balancing for stateless replicas & also different load monitoring granularity levels.
- CORBA persistent state service can support state migration & also hierarchical load monitoring.
- CORBA fault tolerance service implementation can support basic fault tolerance.
- TAO can support extensible load balancing strategies.
- TAO's runtime control of replica life times will capitalize on the interface provided by the Fault Tolerance specification.

Developing Middleware for Webaware Systems: Lessons Learned [120]

As application sharing has become more popular &prevalent in the current global markets, the author discusses the challenges concerning design & development of middleware for web – aware systems.
He cites the practical example of a web based middleware Go Web, which is a object – oriented distributed software that gathers objects into a web enabled application & can be accessed from any platform that supports JVM (Java Virtual Machine). He also states the underlying technology includes

- Object orientation
- Thin client principle
- Multithreading
- Code Portability

He highlights the major difficulties that developers of a web –based systems have to deal with such as, unstable web technology, design of user interface, accessibility, security, & performance issues. Nevertheless, he thinks swift development & confirmed effort shall solve most problems near future.

He thinks that, though complex to develop, web –aware systems like Go Web shall radically reduce information dissemination & delivery of various products and documents to the user desktop.

Applications Thrive on Open Systems Standards [125]

Application development is taking a new course with the changing trends & maturity of its tools & environments. New standards are evolving from appropriate standard organizations. The task of the application developer now, has become, to retrofit existing applications into new energy standards. From the perspective of users & vendors alike, an open systems environment, which consists of a computing support infrastructure to facilitate applications interoperability, portability, and scalability across networks of heterogeneous hardware/software platforms is the answer. Such an environment forms an extensible framework that allows interfaces, services, protocols and supporting data formats to be defined in terms of non – proprietary specifications that evolve through open, consensus based forums.

Several group of standards like RDA, SQL/3, ODMA, SDTS, SAIF, & DEN seem to be future for application development.

Avoiding a Middleware Muddle [129]

As the essence of middleware is enabling communication, it is very important to identify whether it is

- The right product for application
- Constrained to a single application or stored over multiple applications within the boundaries of the business.
- Constrained to the intranet of the business or across company boundaries via the Internet.

Therefore, the author here tries to analyze position such communication middleware & make recommendations. He describes the 5 middleware models proposed by the Gartner Group as follows:

- Conversational – This model fits in real – time programming such as process control system.
- Request/Reply – This model applies when a program requires the capacity to invoke another program's function.
- Message Passing – This model performs a simple one-way transfer of information in the form of a message.

- Message queuing - This model also transfers information via messages however, provides the ability to queue the messages.
- Publish/Subscribe – This model used non directional communication; a program simply publishes a message to the middleware. Programs subscribe to selected message types and notified when any message of a subscribed type is published.

Middleware Design Issues for Application Management in Heterogeneous Networks [130]

In this paper, the authors present a middleware architecture and initial implementation of that middleware that satisfies applications according to expected requirements, application service level agreement & distributed networked resources in and beyond local environment.

In this framework, the whole network – computing environment is constructed as a hierarchical resource tree. Depending on the size of the network - computing environment, this resource can have 2 or more levels. At the leaves of the resource tree, the resources are grouped according our standard grouping criterion and the resource location. Each resource is managed by a local entity called Resource Inspector.
This system allows decisions that incorporate resource information across multiple domains, facilitating the application's task, enabling QoS requirements to be met, and resulting in load balancing for a variety of resource types.

Evolving Hypermedia Middleware Services: Lessons & Observations [141]

The Hypermedia system architectures have been evolving quite a bit from the monolithic systems of 1980s to the middleware – oriented component – based open systems of today. The author evaluates various challenges faced by moving more towards the middleware – oriented approach.

The following is the summary of the general observations:

- Increased Flexibility: The move from a single link server in OHSs to an open set of hypermedia structure servers in CB – OHSs has greatly increased the flexibility of the systems.
- Better Maintainability: The more from homemade frameworks in 1G CB – OHSs to off – the –shelf frameworks in 2G CB – OHSs allowed for much easier maintainability.
- Increased front end interoperability: The move from single overloaded link servers in OHSs to an open set of servers in CB – OHSs has allowed standardization efforts to go forward at a much quicker pace by partitioning the standardization task.
- Inability to address legacy application integration: The migration to middleware approach has not helped addressing issues of integrating legacy third party applications into hypermedia environments

- Inability to address backend application integration fully: The migration has made it more difficult to decide what kind of support should be provided by the backend of a hypermedia system.

Hypermedia Research Directions: An Infrastructure Perspective [140]

The author reviews the hypermedia infrastructure research, and focuses on 2 particular threads of such research named 'multiple open services' & 'structured computing' Multiple Open Services: a multiple open service environment has 3 characteristics that allow it to provide generally applicable middleware services. They are as follows:

- It is based on a 'plug &play' for changes to the architecture allowing: insertion of new components with new services; replacement of existing components by new; and, removal of existing components.
- An open service is available to an open set of applications in the computing environment. It is orthogonal to other services used by participating applications like storage and display services.
- Middleware services are available on major computing platforms such as Suns, PCS, and Macintoshes, for most major OS, like Unix, Windows, and Mac OS and to applications written in many programming languages.

Structural Computing: It describes the view that structural abstractions should constitute the fundamental building block of CB – OHS. It posits additionally, however that the design & implementation of arbitrary services accrue these same benefits on such an infrastructure.

XV. Mediators

Mediators in the Architecture of Future Information Systems [139]

The available data has to be processed intellectually so that technology advances are utilized efficiently which requires a class of software modules that mediates between the workstation applications and the databases. Mediation simplifies, abstracts, reduces, merges, and explains data. Mediators form a distinct middle layer there by plays an active role in driving transformations with knowledge structure.

The Problems: This article discussed on the two types of problems that exist:

- For single databases, the prime hindrance for end-user access is the volume of data that is being available, the lack of abstraction, and the necessity to understand the representation of data.
- The major concern, when information is combined from multiple databases is the mismatch encountered in information representation & structure.

A Taxonomy and Current Issues in Multidatabase Systems [10]

By integrating information from preexisting, heterogeneous local databases in a distributed environment, Multidatabase systems give users a common interface and provide transparent methods to use the total information

This article presents global information-sharing systems taxonomy and discusses where multidatabase systems best fit in to the spectrum of solutions.

Global information-sharing systems can be categorized in to distributed databases, Global Schema multidatabases, Federated databases, Multidatabase language systems, Homogeneous multidatabase language systems, Interoperable systems.

Issues in multidatabases: site autonomy, differences in data representation (name, format, structural, missing or conflicting data), heterogeneous local databases, global constraints, global query processing, concurrency control, security and local node requirements.

Design approaches: global schemas and multidatabase languages.

Problems involved in making multidatabase systems more useful & efficient:

- Global system requirements equal distribution should not be there among all nodes.
- Given the possible huge amounts of data available globally, there is a lack of support for identifying specific data in a system.

XVI. Multimedia

A Standard for Multimedia Middleware [35]

PREMO (Presentation environments for Multimedia Objects) is a new ISO standard to be published in 1998, is the result of conflicts that are emerged over the design of programming interfaces allow the creation, manipulation, and presentation of multimedia information. It defines a middleware framework encompassing synchronization, the management of distributed media resources, and the seamless integration of data and processes from disparate application areas. This paper gives an overview of the concepts underlying PREMO and the approach by which the standard utilizes these concepts to address key aspects of multimedia system design.

The authors are currently developing a reference implementation of PREMO with the intention that resulting toolkit will be used in a computer animation project, which combines traditional animation techniques with event-based synchronization requirements. Because PREMO has its own object model and assumptions about its environment, the task of implementing PREMO is non-trivial. Each object oriented programming language is also equipped with its own object model and assumptions and such a model has a variety of features and constraints, which may or may not be compatible with those of the standard .As PREMO objects are potentially distributable, is

decided that, in addition to the language binding, it would also need to define an environment binding, this includes remote method invocation, which could be realized using the specialized services such as the CORBA technology of OMG, or the RMI package of the Java library.

Using Fagin's Algorithm for merging ranked results in multimedia middleware [142]

Fagin gives an algorithm for efficiently merging multiple ordered streams of ranked results, to form a new stream ordered by a combination of those ranked results and this algorithm is for an actual multimedia middleware system. In this paper the implementation of Fagin's algorithm in an actual multimedia middleware system, including a novel, incremental version of the algorithm that supports dynamic exploration of data is described.

XVII. AOA

Application-oriented Object Architecture: A Revolutionary Approach [66]

The author in this paper proposes a new and revolutionary architecture called Application-oriented Object Architecture which is based on Application Object(AO), self managed hardware units and focuses on applications instead of computing environments. This architecture is based on primitive computer applications such as sending a message, editing a document, sorting data, computing FFTs, and performing a purchase online. The author states that, with proper understanding of such primitive applications, an Application object can be designed and run in a hardware without a need for a particular environment, thus offering total flexibility in AOs and stability in hardware and software evolution.

He proposes that the AOs be implemented as executable modules by application-specific vendors and distribute the objects through a network server. He has demonstrated such AO implementation using C++ and developed AO interfaces using assembly language. He states that, this architecture requires a different mindset to design and implement computer hardware and software applications and its impact is not eminent at this point.

Application-oriented Object Architecture: Concepts and Approach [67]

The authors in this paper, view information technology in the following eight different dimensions i. Vendors ii. Hardware releases iii. Software releases iv. Operating Systems v. Programming languages and tools vi. Application environments vii. Applications and viii. Secure systems.

They propose a Application-oriented Object Architecture (AOA) solution with one program unit with a sole knowledge of its creation, execution and retention, thus bundling the application programs with operating system and environment related programs. Such a program can also run on a bare machine and does not require other

program loaded prior to running the application program. The architecture proposed has only two layers; application layer implemented using an application object (AO) and hardware layer implemented using application units (AU). The two layers communicate through a standard interface.

XVIII. XML

XML-Based Info Mediation for Digital libraries [6]

Distributed prototype architecture for digital libraries is demonstrated which is based on XML-based modeling of metadata, use of an XML query language, and associated mediator middleware, to query distributed metadata sources, and the use of a storage system middleware to access distributed, archived data sets.

On wrapping Query languages and Efficient XML Integration [22]

Though XML enables fast wrapping and declarative integration, query processing in XML-based integration systems is penalized by lack of algebra with adequate optimization properties and the difficulty to understand source query properties. Finally develop the new optimization techniques for XML-based integration systems by defining general purpose algebra suitable for XML, and to also wrap more structured query languages such as OQL or SQL.

Using XML with ODBC and JDBC: WHITE PAPER [29]

This white paper discusses DataDirect Technologies' latest advances in using XML with ODBC and JDBC standards, and includes basic technology explanation and benefits of using XML in a standards-based data access model. It is emphasized that DataDirect Connect Wire Protocol ODBC drivers are the only drivers on the market that let the users easily create XML from data stored in relational databases by persisting query results as XML data files. It's also mentioned that the flexible and powerful jxTransformer, a developed product allows programmers to retrieve data from relational databases and transform that data into complex XML structures using a Java API.

Managing your data the XML way: Data transformation, exchange and integration [30]

In summary, by giving the different aspects and solutions of Business-to-Business transactions (B2B), e-Business, data exchange and integration initiatives, the author asserted that XML based technologies are quickly becoming the industry standard for integrating, transforming and flowing data and structured content over Internet or internal networks connected to enterprise databases in a secure environment. This paper considers the role of XML and Java technologies in today's enterprise and provides an overview of DataMirror Corporation's XML software solutions including DB/XML Transform and DB/XML Vision. The author insists on considering new EAI and B2B tools that support open standards like XML.

Efficient Evaluation of XML Middle-ware Queries [41]

The paper addressed the problem of efficiently constructing materialized XML views of relational databases and focuses on how best the SQL queries can be chosen, without having control over the target RDBMS .To do this, they used SilkRoute, a declarative query language of a middleware-system.

Middleware for Software Leasing over the Internet [56]

The author defines MMM (Middleware for Method Management) as "an infrastructure for managing the deployment, integration, distribution, and use of application services (may range from a simple database access to a full fledged application package) via the WWW." The MMM implementation is based on standard Web technologies, such as HTML, XML, and MetaHTML; distributed object computing frameworks, such as CORBA; and database technology, such as ODBC. In this paper a technical account of the MMM architecture is given and its primary features are discussed.

MMM system is a kind of middleware that implements a software infrastructure in which users make their input data available to a service that performs the necessary computations remotely & sends the results back to the user, and offers a component leasing based business model. Some research related issues those are to be answered- Licensing and pricing of software deployed on a leasing basis, certification of services to ensure customers a certain 'quality of service', and simplicity of user-system interaction.

XIX. Security

Secure Virtual Enclaves: Supporting Coalition Use of Distributed Application Technologies [121]

The primary goal was to study software mechanisms to support coalitions in collaborative computing efforts. To meet coalition requirements, a prototype security infrastructure is designed and implemented. The problem of limited trust relationships is addressed by providing significant resource access policy definition and enforcement autonomy to individual member enclaves, and support for the changing nature of a collaborative arrangement is provided by a dynamic policy update mechanism.

Secure Virtual Enclaves (SVE) collaboration infrastructure is transparent to applications, which is implemented in middleware, with no modifications to COTS operating systems or network protocols allows multiple organizations to share their distributed application objects, while respecting organizational autonomy over local resources. While the prototype demonstrates fine-grained access control for secure collaborative computing, they have identified significant issues that remain to be addressed, in the area of policy development before such collaboration will be convenient. The SVE platform infrastructure offers a platform and a conceptual basis for further exploration of these problems and experimentation with new solutions.

Intrusion detection for distributed applications [124]

In this article, the organization, Odyssey Research Associates (ORA), briefly reviewed intrusion detection and computer immunology, and then described their approach to providing intrusion detection in distributed object applications. The heart of the approach involves an empirical characterization of the application, which is called application's 'self'; the essential components of such a characterization are discussed. A prototype intrusion detection system (IDS) to protect applications that are based on the CORBA is built and results are obtained. The system collects training data and subsequently uses the data to detect intrusions

XX. Distributed Processing and Computing

Middleware: A model for Distributed System Services [8]

Most organizations have a wide variety of heterogeneous hardware systems, including personal computers, workstations, minicomputers, and mainframes. Because, these systems run on different OSs and rely on different architectures, integration is difficult and its achievement uneven. One way to solve this is by supporting standard protocols and another way is to offer distributed system services that have standard programming interfaces and protocols. These services are called middleware services as they sit 'in the middle' in a layer above the OS and networking software and below industry – specific applications. In this paper, middleware services and issues related to them are discussed in detail.

A Distributed Object Computing Architecture for Leveraging Software Reengineering Systems [21]

In this paper, a distributed object computing architecture for allowing existing commercial software reengineering systems to integrate with the customer's heterogeneous distributed environments is presented. The architecture was implemented with the architectural design patterns (Adapter, Broker & Façade) through CORBA technology. The product value is achieved through the leverage of legacy commercial software reengineering systems and adapted the product to new market needs.

Using Off-The-Shelf Middleware to Implement Connectors in Distributed Software Architectures [28]

Software interconnection and middleware technologies such as RMI, CORBA, ILU, & ActiveX provide a valuable service in building applications from components. The relation of such services to software connectors in the context of software architectures, however, is not well understood. To understand the trade offs among these technologies with respect to architectures, several off –the – shelf middleware technologies are evaluated and key techniques for utilizing them in implementing software connectors are identified.

The Gateway System: Uniform Web Based Access to Remote Resources [42]

A new system, Gateway is designed by exploring their experience developing the webflow system, to provide seamless & secure access to computational resources at ASC MSRC. The Gateway is implemented as a modern three –tier architecture i.e. each standard is implemented in order to strictly conform to the standards. Tier 1 is a high level front end for visual programming steering, run – time data analysis and visualization that is built on top of the web & OO commodity standards. Distributed object – based, scalable, and reusable web server & object broker middleware using CORBA & Java Beans forms Tier 2. Tier 3 comprises back – end services. In particular, access to high - performance computational resources is provided by implementing the emerging standard for metacomputing API.

Tools and Approaches for Developing Data – Intensive Web Applications: A Survey [44]

This paper investigates the current situation of web development tools, both in the commercial and research fields, by identifying and characterizing different categories of solutions, evaluating their adequacy to the requirements of Web application development, enlightening open problems, and exposing possible future trends.

A Windows CE Implementation of a Middleware Architecture Supporting Time – Triggered Message – Triggered Objects [45]

The time – triggered message – triggered object (TMO) programming scheme has been establishing to remove the limitations of conventional object programming techniques & tools in developing applications containing real – time (RT) distributed computing components. As a cost – effective facility for supporting TMO – structured distributed RT programming, a middleware architecture that can be adapted to various well – established commercial software/hardware platforms like windows CE has been established and been named TMO support middleware (TMOSM). In this paper, the internal structure of & some implementation techniques used in the prototype TMOSM/CE are discussed. In addition, it is said that experimental efforts will continue toward making assessments of the case of RT distributed programming enabled by TMOSE/CE.

Early Adopters An Internet 2 Middleware Project [47]

Early Adopters, the campus test-bed phase of early Harvest, is a group of eleven institutions of higher education working to provide a test-bed for the deployment of middleware technologies. Early Adopters project has some primary and secondary goals to be achieved are listed in the paper.

An Object-oriented Middleware for our Metasystem on Internet [48]

A metasystem is composed geographically distributed heterogeneous resources that can be reached over the network. In this paper, an object – oriented middleware WaPC, for the metasystem based on Internet is described in detail & presented the way of implementing it on top of a heterogeneous platform composed of the x86/Linux, Sun Sparc/Solarsis and IBM power PC/AIX & it's compatibles. It also masks the hardware details, giving the user a single – image space system with transparent data transfer & uniform program interface. Further research has to be done to reduce the effort of rewriting the application programs, which is required to deal with different communication & processing performance of different architectures.

Towards a Universal Service – Computing Platform via Virtual Service Machine [61]

By lifting up traditional focus of software development from the level of applications to that of services, Service computing is a new paradigm for manufacturing IT artifacts. For this we need a new breed of middleware that can relieve the developers of low-level platform concerns, e.g. hardware, OS or even middleware. Virtual service machine is a novel way toward more effective service – computing development. This paper proposes an approach of designing VSMs, which can be used as a development platform for service – computing frameworks & applications and emphasizes on presenting the functionality & components inside the VSM.

Experience with Group Communication Middleware [62]

Group Communication is a widely studied paradigm for building fault – tolerant distributed systems. This paper describes the experienced results in implementing a fault-tolerant distributed radar tracking system, and discusses how could they able to simplify the design implementation by utilizing a service library containing additional abstractions built on top of the group communication model.

A Survey of Energy Efficient Network Protocols for Wireless Networks [63]

As wireless networks became an integral component of the modern communication infrastructure and continue to add more capabilities such as multimedia & QoS, low power design remains one of the most important research areas with in wireless communication. Since the power conservation in the hardware design has already been achieved, the key to energy conservation in wireless communication lies with in higher levels of the wireless protocol stack. This paper describes research completed at the data link, network, transport, OS/middleware, and application protocol layers that have addressed energy efficiency for wireless networks. However, power conservation with in the wireless protocol stack remains a very crucial research area for the viability wireless services in the future.

Design of MOBILE MOM: Message Oriented Middleware Service for Mobile Computing [64]

Message oriented middleware (MOM) is a specific class of middleware that operates on the principles of message passing or message queuing. In the near future, mobile computing environment & wireless network environment will be spread & an essential element of computing environment. Therefore, not only H/W supporting mobile computing but also ADE based on such H/W needs to be developed. The concept of MOM lends itself mobile computing environment. This paper has presented the model of MOM that is compatible to mobile computing environment. According to results of tests, MOBILE MOM has fault –tolerance supporting features of wireless communication.

Deadlock Detection in Distributed Object Systems [68]

Distribute object & component middleware is used by an increasing number of distributed systems. This increase calls for support of developers by systematic software engineering methods. The research concentrates synchronization behavior in object – oriented middleware systems. By utilizing the fact that modern object middleware offers only a few built in synchronization and threading primitives by suggesting UML stereotypes to represent each of these primitives in distributed object design. The semantics of the stereotype are designed using a process algebra and used in translating UML diagrams into behaviorally equivalent process algebra representations and then can be used for model checking techniques to find potential deadlocks. The paper also shows how the model checking results can be related back to the original UML diagrams.

Object-oriented Real-Time Distributed Programming and Support Middleware [72]

Time triggered message triggered object (TMO) programming scheme is used to make specific illustration of the issues and potentials of OORT programming. The desirable features of middleware providing execution support for OORT distributed programs, the issue of fault – tolerant execution of distributed RT objects & that of RT distributed/parallel simulation are discussed. The research community dealing with this technology area is expected to grow continuously and consequent accelerations of the technology advances will in return accelerate the development of many new types of sophisticated RT distributed computing applications.

Experiences with Building Distributed Debuggers [90]

A number issues that are encountered in building distributed debuggers are presented and briefly described an approach to addressing them. The architecture of the prototype is described. The prototype supports the debugging multi – threaded, multi – process, multi – language applications that use multiple middlewares while executing in a heterogeneous distributed environments. Finally, the implementation of some of the distributed primitives that make debugger particularly suited to debugging distributed applications is described.

Introduction to the Electronic Symposium on Computer – Supported Cooperative work [97]

The DARPA IC&V program explored the premise that significant advances in computer supported cooperative work can best be achieved by mixing together ideas from the most talented researchers across a range of disciplines including networking, multimedia, artificial intelligence, visualization and human computer interaction. In the context of a military setting, the paper describes particular challenges for CSCW researchers. While most of these challenges might seem specific to military environments, many others might have already found similar challenges, or soon will, when attempting to collaborate through networks of computers. To support this claim, the paper includes a military scenario that might hit fairly close to home for many, certainly for civilian emergency response personnel.

Towards Universal Software Substrate for Distributed Embedded Systems [101]

This paper proposes universal software substrate that can be used to build various types of distributed embedded systems. The universal software substrate contains an operating system and several middleware components and offers universal application programming interface (Universal API) that increases application's portability extremely. The most important issue of research is how to control the level of abstraction when designing distributed systems. The authors described several software components towards realizing universal software substrate, and some research topics for achieving the goals.

The Design and Performance of a Pluggable Protocols Framework for Real – time Distributed Object Computing Middleware [105]

OO middleware must preserve communication-layer QoS properties of applications end-to-end, for an effective development platform for performance – sensitive applications. It is essential therefore to define pluggable protocols frameworks that allow custom inter - ORB messaging & transport protocols to be configured flexibly & transparently by CORBA applications.

This paper provides 3 contributions to research on pluggable protocols framework for performance sensitive distributed object computing (DOC) middleware.

- The key design challenges faced by pluggable protocols developers are depicted.
- How these challenges are resolved by developing a pluggable protocol framework for TAO, which is high performance, real – time (ORBA – compliant ORB) is also described.
- The results of benchmark that pinpoint the impact of TAO's pluggable protocols framework end –to – end efficiency & predictability are presented.

The results obtained from the performance of TAO's pluggable protocols framework when running CORBA applications over high speed interconnects, such as VME, illustrates that (a). DOC middleware performance is largely an implementation on detail

& (b). The next generation of optimized, standards – based CORBA middleware can replace ad hoc & proprietary solutions.

A Scalable, Distributed Middleware Service Architecture to Support Mobile Internet Applications [108]

To improve the scalability of the old architecture, which has middleware to perform a new capability application session handoff using a single middleware server to provide all the functionality, the authors designed an efficient distributed middleware service layer that properly maintaining application session handoff semantics while being able to service a large number of clients protocols involved in performing handoff explained in detail and analyzed an implementation of the architecture that supports the use of a real medical teaching tool. From experimental results, it can be seen that the middleware service effectively provides scalability as a response to increased workload.

General Framework for Fault Tolerance from ISO/ITU Reference Model for Open Distributed Processing (RM – ODP) [109]

Fault tolerance is an important system property of a distributed system to enable a robust, reliable, and stable system. The ISO Reference Model for Open Distributed Processing (RM – ODP) is a software engineer enabler for software architecture of distributed systems, which addresses the non functional properties of distributed systems in terms of a framework of concepts, structuring rules, semantics and mechanisms. One such property Fault Tolerance. This paper provides a view of how RM –ODP addresses the fault tolerance (FT) framework that consists of precise terminology, precise concepts for fault tolerance and mechanisms of infrastructure services and policies to govern the behavior of the system. A brief comparison of this with the emerging proposal for a fault tolerant CORBA is also provided.

A Pattern language for Efficient, Predictable, Scalable, and Flexible Dispatching Mechanisms for Distributed Object Computing Middleware [111]

In many application domains, distributed object computing (DOC) middleware is responsible for dispatching upcalls to one or more application objects when events or requests arrive from clients in particular. Dispatching mechanisms must be prepared to dispatch upcalls to multiple objects, to handle recursive requests originated from application-provided upcalls, and must often collaborate with the application to control object life cycle. This paper presents two contributions to the design and implementation of efficient, predictable, stable, & flexible DOC middleware and applications.

- It shows how patterns can be applied to the object-oriented systems to capture key design & performance characteristics of proven dispatching mechanisms.
- It presents a pattern language that describes successful solutions that are appropriate for key dispatching challenges that arise in various real time DOC middleware and applications.

The pattern language documented in this paper has been applied to TAO real time.

MIMO: An Infrastructure for Monitoring and Managing Distributed Middleware Environments [112]

MIMO is based on a new multi layer monitoring application for middleware systems, which classifies collected information using several abstraction level and the key features of this are its openness, flexibility, extensibility. This paper presents the MIMO middleware monitoring system, an infrastructure for monitoring and managing distributed middleware environments, and the MIVIS visualization tool demonstrating basic MIMO functionality, and which serves as a framework further tool extension. MIMO's research contribution is to enable the integration of different middleware platforms which is reached by introducing a standardized intruder – monitor – interface, it's asserted that, no other monitoring infrastructure reaching this high degree of flexibility over several dimensions has been developed up to now.
The main research contribution is motivated by the fact that common middleware environments & tool requirements are too diverse to be handled by a single, static monitoring system. Instead, an open, flexible monitoring and management infrastructure, which only provides basic monitoring services, but is open to be extended easily is proposed and implemented.

Mobile Streams: A Middleware for Reconfigurable Distributed Scripting [113]

The design of a Middleware for building reactive, extensible, reconfigurable distributed systems, based upon an abstraction called Mobile Streams is presented. Using this system, a distributed, event – driven application can be scripted from a single point of control, dynamically extended and reconfigured while it's in execution. Also this system is suitable for building a wide variety of applications; for ex, distributed test, conferencing and control oriented applications.

The authors are currently building a data-management application that manages data collection & visualization in a distributed, 'collaboratory'.

Strategies for integrating Messaging and Distributed Object Transactions [127]

Integrating messaging into distributed object environments and in particular with distributed object transactions, describes a novel & complex software design problem. This paper details this problem, presenting first results from the project of developing a messaging & transaction integration facility and introduced a comprehensive messaging classification framework that serves for this purpose. The framework defines messaging properties and property values organized around three models: message delivery model, message processing model, and message failure model. Also derived four strategies for integrating messaging & distributed object transactions, each serving for a specific integration objective & following a distinct flavor of messaging: MQ – integrating transactions, message delivery transactions, message processing transactions, and full messaging transactions. Overall, this paper advances understanding of the motivation for,

the problems of, the current state-of-the-art in and future models for integrating messaging & distributed object transactions.

Distributed, Scalable, Dependable Real-Time Systems

Some classes of real-time systems function in environments which cannot be modeled with static approaches. This paper presents middleware services that support such dynamic real time systems through adaptive resource management middleware that provides integrated services for fault tolerance, distributed computing & real time computing. The middleware services have been implemented and employed for components of the experimental Navy system. Experiments show that the services provide bounded response times, scalable services and low intrusiveness.

XXI. Tutorials/Whitepaper

Java Servlets and EJB's In Enterprise Architectures: Friends or Foes

Though both the servlets and the EJB technology are suitable platforms for an enterprise application, the servlets use would be sub-optimal.

Comparison of Servlets & EJBs:

Application Topology:

Servlets	EJBs	Comments
Web-Centric model	Web-centric model LAN-based model Local	EJB provides much more flexible deployment

Client Access:

HTTP only	HTTP RMI and/or CORBA IIOP HTTP tunneling Value added proprietary protocols such as t3.	RMI & CORBA are optimized than HTTP.

Transactional Support:

Through integration with JTS.	All transaction functions are Performed by the container.	EJB is better choice.

Similarly when we take properties like Scalability, Load balancing, Business logic hosting, Business agility, Integrity, High availability, Fail over, Deployment, Portability, Management and administration, Security, persistence, Object Identification and Context Association, Environment Access, Naming and Directory Services, Life-cycle Management, State Management, Resource Binding, Resource Sharing, Relationship Multiplicity, Database Integration, Invocation Arguments, Threading Model, Metaphor, Chaining and compare the Servlets and EJBs can state that EJBs are better choice.

It seems to be a perfect match in using a lightweight servlets as a Web-exposing front end to EJB-encapsulated business functionality.

Scalability Issues in CORBA-based Systems [135]

OMG Specifications and The implementation choices made by middleware providers and application developers affect CORBA application scalability. Performance is only one quality among many like Flexibility, Scalability and Maintainability, that distributed application architects & designers must consider.

3. Conclusions:

In this paper, we developed a comprehensive bibliography of all the middleware related articles, to be used by other researchers for further advancement and study. Broadly, we understood that, with the rapid pace of information technology changes and increasing demands of customers for better services, there is a need to bring new systems and applications on stream and to open new service and distribution channels and that, middleware plays a pivotal role in opening up such channels to enterprise IT environments and integrating application systems. During such study and preparation, we have also learnt and understood several new middleware classifications and solutions and noted suggestions pertaining to further advancement and refinement proposed by several authors.

References based on categories:

Aspected Oriented Programming:

[1] Zhang, C., Jacobsen, H., Quantifying aspects in middleware platforms. *Proceedings of the 2nd international conference on Aspect-oriented software development,* March 2003 , pg 130 - 139.

Components/Frameworks:

[1] Blair, G.S., et al, The role of software architecture in constraining adaptation in component –based middleware platforms, *IFIP/ACM International Conference on Distributed systems platforms,* 2000, pg 164 – 184.

[2] Bronsard, F., et al, Toward Software Plug and Play, *ACM SIGSOFT Software Engineering Notes, Proceedings of the 1997 Symposium on Software Reusability, Vol. 22, Issue 3*, May 1997, pg 19 – 29.

[3] Carew, M., The Anatomy of a Component.

[4] Costa, F. and Blair, G., Integration Meta-Information Management and Reflection in Middleware*, Proceedings of the IEEE International Symposium on Distributed Objects and Applications (DOA)*, Sep 2000.

[5] Coulson, G., A Configurable Multimedia Middleware Platform, *IEEE Multimedia*, Jan/Mar 1999, pg. 62 – 76.

[6] Di Nitto, E. and Rosenblum, D., Exploiting ADLs to Specify Architectural Styles Induced by Middleware Infrastructures, *ACM Proceedings of the 1999 International Conference on Software Engineering*, May1999, pg 13 – 22.

[7] Fayad, M., Introduction to the Computing Surveys' Electronic Symposium on Object-Oriented Application Frameworks, *ACM Computing Surveys (CSUR)*, Mar 2000.

[8] Francois, A.R.J., Medioni, G.G., A Modular Middleware Flow Scheduling Framework, Proceedings *of the eighth ACM international conference on Multimedia,* October 2000,pg 371 – 374.

[9] Gokhale, A., et al, Applying model-integrated computing to component middleware and enterprise applications, *Communication of the ACM, Vol. 45, Issue 10*, October 2002,pg 65 – 70 .

[10] Hofmann, C., A Multi-Tier Framework for Accessing Distributed, Heterogeneous Spatial Data in a Federated Based EIS, *Proceedings of the 7th International Symposium on Advances in Geographic Information Systems*, Nov 1999, pg 140 – 145.

[11] Issarny, V., Kloukinas, C., and Zarras, A., Systematic Aid for Developing Middleware Architectures, *Communication of the ACM, Vol. 45, No. 6*, Jun 2002, pg 53 – 58.

[12] Jeng, J., An Approach to Designing Reusable Service Frameworks via Virtual Service Machine, *ACM SIGSOFT Software Engineering Notes, Proceedings of SSR'01 on 2001 Symposium on Software Reusability, Vol. 26, Issue 3*, May 2001, pg 58 – 66.

[13] Kramp, T., Coulson, G., The Design of a Flexible Communications Framework for Next-Generation Middleware, *IEEE International Symposium on Distributed Objects and Applications,* September 2000.

[14] Lewandowski, S.M., Frameworks for Component-Based Client/Server Computing, *ACM Computing Surveys, Vol. 30, No. 1,* March 1998.

[15] Moreira, R.S., et al, A Reflective Component-Based & Architecture Aware Framework to Manage Architecture Composition, *Proceedings of the Third International Symposium on Distributed-Objects and Applications (DOA'01),* IEEE 2001.

[16] Schmidt, D.C., Buschmann, F., Patterns, Frameworks, and Middleware: Their Synergetic Relationships, *25th International Conference on Software Engineering*, May 2003, pg 694 – 704.

[17] Truyen, E., et al, Customization of Component-based Object Request Brokers through Dynamic Reconfiguration, *Technology of Object-Oriented Languages and Systems (TOOLS 33) IEEE,* June 2000.

[18] Wiederhold, G., Mediation in Information Systems, *ACM Computing Surveys, Vol. 27, No, 2,* June 1995.

Middleware Solutions:

[1] Campbell, A., Coulson, G., and Kounavis, M., Managing Complexity: Middleware Explained, *IT Pro*, Sep/Oct 1999, pg 22 – 28.

[2] Emmerich, W., Software Engineering and Middleware: A Roadmap, *ACM Proceedings of the Conference on the Future of Software Engineering*, May 2000, pg 119 – 129.

Embedded & Real – time Systems:

[1] Janka, R., A New Development Framework Based On Efficient Middleware for Real-Time Embedded Heterogeneous Multicomputers, *IEEE Conference and Workshop on Engineering of Computer-Based Systems*, March 1999, pg 261 – 268.

[2] Janka, R.S., Wills, L.M., A Novel Codesign Methodology for Real-Time Embedded COTS Multiprocessor-Based Signal Processing Systems, *Proceedings of the eighth international workshop on Hardware/software codesign***,** May 2000,pg 157 – 161.

[3] Mizunuma, I., Shen, C., and Takegaki, M., Middleware for Distributed Industrial Real-Time Systems on ATM Networks, *Proceedings of the 17th IEEE Real-Time Systems Symposium (RTSS'96)*, pg 32 – 38.

[4] Schmidt, D., Middleware for Real-Time and Embedded Systems, *Communications of the ACM, Vol. 45, No. 6*, Jun 2002, pg 43 – 48.

CORBA/ORB:

[1] Brunsch, D., O'Ryan, C., and Schmidt, D., Designing an Efficient and Scalable Server-side Asynchrony Model for CORBA, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg 223 – 229.

[2] Canal, C., et al, Extending CORBA Interfaces with Protocols, *British Computer Society 2001, Vol. 44, No. 5*, pg 448 – 462.

[3] Capra, L., Exploiting reflection in mobile computing middleware, *ACM SIGMOBILE Mobile Computing and Communications Review, Vol. 6, Issue 6*, 2002, pg 34 – 44.

[4] Caron, O., Carré, B., and Debrauwer, L., An Original View Mechanism for the CORBA Middleware*, IEEE Proceedings of the Technology of Object-Oriented Language and Systems (TOOLS 33)*, Jun 2000.

[5] Coulson, G., What is Reflective Middleware? , RM article.

[6] Dogac, A., Dengi, C., and Öszu, M.T., Distributed Object Computing Platforms, *Communications of THE ACM*, September 1998/Vol. 41, No. 9,pg 95 – 103.

[7] Kim, K., et al, A CORBA Service Enabling Programmer-Friendly Object-Oriented Real-Time Distributed Computing, *IEEE 5th International Workshop on Object-Oriented Real-Time Dependable Systems*, Nov 1999.

[8] Kon, F., et al, The case for reflective middleware, *Communications of the ACM, Vol. 45, Issue 6*, June 2002, pg 33 – 38.

[9] Krishnamurthy, Y., et al, Integration of QoS-Enabled Distributed Object Computing Middleware for Developing Next-Generation Distributed Applications, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg 230 – 237.

[10] Kuhns, F., Schmidt, D.C., and Levine, D.L., The Performance of a Real-time I/O Subsystem for QoS-enabled ORB Middleware.

[11] Kutlusan, A., et al, A Combat Management System Middleware Based on CORBA, *The Proceedings of the International Symposium on Distributed Objects and Applications (DOA'00)*, IEEE 2000.

[12] Marrón, P., Design of a Middleware Service for Scalable Wide Are Network Applications, *Proceedings of the IEEE International Symposium on Distributed Objects and Applications (DOA),* Sep 2000, pg 1 - 10.

[13] McFall, C., An Object Infrastructure for Internet Middleware, *IEEE Internet Computing*, Mar/Apr 1998, pg 46 - 51.

[14] Medvidovic, N., On the role of middleware in architecture-based software development, *ACM International Conference Proceeding Series-Proceedings of the 14th*

international conference on Software engineering and knowledge engineering, 2002, pg 299 – 306.

[15] Narain, S., et al, Middleware for Building Adaptive Systems via Configuration, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg 188 – 195.

[16] O'Ryan, C., et al, Applying a Scalable CORBA Event Service to Large-scale Distributed Interactive Simulation, *IEEE 5th International Workshop on Object-Oriented Real-Time Dependable Systems*, Nov 1999, pg 1 - 8.

[17] Schmidt, D.C., Kuhns, F., An Overview of the Real-Time CORBA Specification, *Computer*, June 2000, pg 56 – 63.

[18] Tsaoussidis, V., A CORBA-based Application Service Middleware Architecture and Implementation.

[19] Wang, N., Kircher, M., and Schmidt, D., Applying Reflective Middleware Techniques to Optimize a QoS-enabled CORBA Component Model Implementation, *Proceedings of the 24th Annual International Computer Software and Applications Conference (COMPSAC)*, Oct 2000, pg 1 - 8.

ADA:

[1] Kruchten, P.B., Thompson, C.J., An Object-Oriented, Distributed Architecture for Large Scale Ada Systems, *Proceedings of the conference on TRI-Ada '94,* Nov 1994, pg 262 – 271 .

DB – Query – Middleware:

[1] Atkinson, R., Use of a Component Architecture in Integrating Relational and Non – Relational Storage Systems, *ACM SIGMOD Record , Proceedings of the 1995 ACM SIGMOD international conference on Management of data, Volume 24 Issue 2*, May 1995, pg 454.

[2] Dao, S., Toward Scalability and Interoperability of Heterogeneous Information Sources, *11th International Conference on Data Engineering*, pg 65 – 67.

[3] Degenaro, L., et al, A Middleware System which Intelligently Caches Query Results, *IFIP/ACM International Conference on Distributed Systems Platforms*, 2000, pg 24 – 44.

[4] Elmagarmid, A.K., Pu, C., Guest Editors' Introduction to the Special Issue on Heterogeneous Databases, *ACM Computing Surveys, Vol. 22, No. 3*, September 1990, pg 175 – 178.

[5] Heiler, S., Extended Data Type Support in Distributed DBMS Products: A Technology Assessment and Forecast, *GTE Laboratories Inc*. pg 1 – 44.

[6] Kiernan, J., Carey, M.J., Middleware Object Query Processing with Deferred Updates and Autonomous Sources, *ACM SIGPLAN Notices , Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, Volume 35 Issue 10* , October 2000, pg 118 – 129.

[7] Kleissner, C., Enterprise Objects Framework: a second generation object-relational enabler, *ACM SIGMOD Record , Proceedings of the 1995 ACM SIGMOD international conference on Management of data, Volume 24 Issue 2*, May 1995, pg 455 – 459.

[8] Kossmann, D., The State of the Art in Distributed Query Processing, *ACM Computing Surveys, Vol.32, No. 4*, December 2000,pg. 422 – 469 .

[9] Manola, F., An Evaluation of Object-Oriented DBMS Developments 1994 Edition, *GTE Laboratories Inc*,pg 1- 6.

[10] Manola, F., et al, Distributed Object Management, *GTE Laboratories Inc*,pg1 – 41.

[11] Manola, F., Heiler, S., A 'RISC' Object Model for Object System Interoperation: Concepts and Applications, *GTE Laboratories Inc.*

[12] Nayeri, F., Addressing Component Interoperability in the OMG Object Model, *ORB Implementer's Workshop*, June 1993, pg 1 – 5.

[13] Meng, W., Construction of a Relational Front-end for Object-Oriented Database Systems,IEEE 1993, pg 476 – 483.

[14] Meng, W., Processing Hierarchical Queries in Heterogeneous Environment .

[15] Meng, W., Kamada, A., and Chang, Y., Transformation of Relational Schemas to Object-Oriented Schemas .

[16] Rodriguez-Martinez, M., and Roussopoulos, N., MOCHA: A Self-Extensible Database Middleware System for Distributed Data Sources, *ACM 2000 1-58113-218-2/00/05*, May 2000, pg 213 – 224.

[17] Scheuermann, P., et al , Report on the Workshop on Heterogeneous Database Systems held at Northwestern University, *SIGMOD RECORD, Vol. 19,No. 4*, December 1990,pg 23 – 31.

[18] Slivinskas, G., Jensen, C., and Snodgrass, R., Adaptable Query Optimization and Evaluation in Temporal Middleware, *Proceedings of the 2001 ACM SIGMOND International Conference on Management of Data on Management of Data, May 2001, Vol. 30 Issue 2*, May 2001, pg 127 – 138.

[19] Subrahmanian, V.S., Amalgamating Knowledge Bases, *ACM Transactions on Database Systems, Vol. 19, No. 2* , June 1994, pg 291 – 331.

[20] Ventrone, V., Heiler, S., Some Practical Advice for Dealing with Semantic Heterogeneity in Federated Database Systems, pg 1 – 20.

[21] Yoo, S.B., Kim, K.C., and Cha S.K., A Middleware Implementation of Active Rules for ODBMS.

[22] Yu, C., et al, Translation of Object – Oriented Queries to Relational Queries, IEEE 1995, pg 90 – 97.

Enterprise Architecture & Solutions:

[1] Kalnis, P., Proxy-Server Architectures for OLAP, *ACM SIGMOD Record, Proceedings of the 2001 ACM SIGMOD international conference on Management of data, Volume 30 Issue 2*, May 2001, pg 367 – 378.

[2] Kim, K.H., Ishida, M., and Liu, J., An efficient Middleware Architecture Supporting Time-Triggered Objects and an NT-based Implementation.

[3] Melling, W.P., Enterprise Information Architectures – They're Finally Changing, *ACM SIGMOD Record , Proceedings of the 1994 ACM SIGMOD international conference on Management of data, Volume 23 Issue 2* ,May 1994,pg 493 – 504 .

QoS:

[1] Conrad, C., Stiller, B., The Design of an Application Programming Interface for QoS-based Multimedia Middleware, *Proceedings of the 22nd IEEE Conference on Local Computer Networks (LCN'97)*, pg 274 – 283.

[2] De Miguel, M.A, General Framework for the Description of QoS in UML, *Proceedings of the Sixth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'03)*.

[3] Hou, C., Han, C., and Chen, T., Communication Middleware and Software for QoS Control in Distributed Real-Time Environments, *COMPSAC'97 – 21st International Computer Software and Applications Conference*, Aug 1997.

[4] Li, B. and Nahrstedt, K., QualProbes: Middleware QoS Profiling Services for Configuring Adaptive Applications, *IFIP/ACM International Conference on Distributed System Platforms*, Apr 2000, pg 256 - 272.

[5] Petriu, D., et al, Using Analytic Models for Predicting Middleware Performance, *ACM Proceedings on the 2nd International Workshop on Software and Performance*, Sep 2000, pg 189 – 194

[6] Pyarali, I., et al, Evaluating and Optimizing Thread Pool Strategies for Real-Time CORBA, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg. 214 – 221.

[7] Saksena, M., Towards Automatic Synthesis of QoS Preserving Implementations from Object-Oriented Design Models.

[8] Tripathi, A., Challenges Designing Next-Generation Middleware Systems, *Communications of the ACM, Vol. 45, No. 6*, Jun 2002, pg 39 – 42.

[9] Yellin, D., Stuck in the Middle: Challenges and Trends in Optimizing Middleware, ACM *SIGPLAN Notices*, Aug 2001, pg 175 – 180.

Ontology:

[1] Arpirez, J.C., et al, WebODE: a Scalable Workbench for Ontological Engineering, *Proceedings of the international conference on Knowledge capture*, October 2001, pg 6 – 13.

[2] Staab, S., et al , An extensible Approach for Modeling Ontologies in RDF (S), *ECDL 2000 Workshop on the Semantic Web*, pg 1 – 10 .

Actors:

[1] Astley, M. and Agha, G., Customization and Composition of Distributed Objects: Middleware Abstractions for Policy Management, *ACM SIGSOFT Software Engineering Notes, Proceeding of the ACM SIGSOFT 6th International Symposium on Foundations of Software Engineering,,Vol. 23, Issue 6*, Nov 1998, pg 1 - 9.

Client/Server:

[1] Benda, M., The Architecture of Global Access, *IEEE Internet Computing*, Jan/Feb 1997, pg 78 – 80.

[2] Gutzman, A.D., Selecting the Appropriate Middleware for Your Web-To-Database Application, *ACM SIGUCCS XXV 1997*, pg 365 – 367.

[3] Hecht, M., et al, OFTT: A Fault Tolerance Middleware Toolkit for Process Monitoring and Control Window NT Application, *Proceedings of the IEEE International Conference on Dependable Systems and Networks*, Jun 2000.

[4] Heuser, L., et al, Development of Distributed and Client/Server Object-Oriented Applications, *Proceedings of the 9th Annual Conference on Object-Oriented programming Systems, Language, and Applications, Vol. 29, Issue 10*, Oct 1994,pg 317 – 323.

[5] Lewis, T., Where is Client/Server Software Headed?, *IEEE Computer, Vol. 28, Issue 4*, Apr 1995, pg 49 - 55.

[6] Thompson, C., et al, Intermediary Architecture: Interposing Middleware Object Services between Web Client and Server, *ACM Computing Surveys (CSUR)*, Jun 1999.

Agents:

[1] Altinel, M., et al, DBIS-Toolkit: Adaptive Middleware for Large Scale Data Delivery, *Proceedings of the 1999 International Conference on Management of Data (SIGMOD '99), Vol. 28, Issue 2,* May 1999, pg 544 - 546.

[2] Eisenstein, J., Rich, C., Agents and GUIs from Task Models, *Proceedings of the 7th international conference on Intelligent user interfaces*, January 2002, pg 47 – 54.

[3] Jain, R., Anjum, F., and Umar, A., A Comparison of Mobile Agent and Client-Server Paradigms for Information Retrieval Task in Virtual Enterprises, *IEEE*, 2000.

[4] Miled, Z.B., et al, Global Change Master Directory : Object-Oriented Active Asynchronous Transaction Management in a Federated Environment Using Data Agents, *Proceedings of the 2001 ACM symposium on Applied computing* **,**March 2001, pg 207 – 214 .

[5] Yamamoto, G., Nakamura, Y., Architecture and Performance Evaluation of a Massive Multi-Agent System, *Proceedings of the third annual conference on Autonomous Agents* , April 1999, pg 319 – 325.

[6] Zhou, J., Zhou, M., and Wu, Q., An Agent Framework Based on Distributed Object, *IEEE Technology of Object-Orient Languages and Systems (TOOLS 33)*, Jun 2000.

Issues:

[1] Agha, G., Adaptive Middleware, *Communication of the ACM, Vol. 45, No. 6*, Jun 2002, pg 31 – 32.

[2] Brown, A., Mastering the Middleware Muddle, *IEEE Software*, Jul/Aug 1999, pg 18 – 21.

[3] Carzaniga, A., et al, Issues in Supporting Event-Based Architectural Styles, *Proceedings of the 3rd International Workshop on Software Architecture*, Nov 1997, pg 17 – 20.

[4] Charles, J., Middleware Moves to the Forefront, *Computer*, May 1999, pg 17 - 19.

[5] Eliassen, F., et al, Next Generation Middleware: Requirements, Architecture, and Prototypes, *7th IEEE Workshop on Future Trends on Distributed Computing Systems*, Dec 2000.

[6] Othman, O. and Schmidt, D., Issues in the Design of Adaptive Middleware Load Balancing, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg 205 – 213.

[7] Serbedżija, N., Developing Middleware for Web-aware Systems: Lessons Learned, *Proceedings of the Australasian Computer Science Conference*, Feb 2000.

[8] Strand, E.J., Mehta, R.P., and Jairam, R., Applications Thrive on Open Systems Standards, *StandardView Vol. 2, No. 3, September/1994*, pg 148 – 154.

[9] Thompson, J., Avoiding a Middleware Muddle, *IEEE Software*, Nov/Dec 1997, pg 92 - 96.

[10] Tian, Y., Middleware Design Issues for Application Management in Heterogeneous Networks, *The Proceedings of the IEEE International Conference on Networks (ICON'00)*. September 2000.

[11] Wiil, U.K., Evolving Hypermedia Middleware: Lessons and Observations, *Proceedings of the 1999 ACM symposium on Applied computing***,** February 1999, pg 427 – 436.

[12] Wiil, U., Nürnberg, P., and Leggett, J., Hypermedia Research Directions: An Infrastructure Perspective*, ACM Computing Survey (CSUR), Vol. 31, Issue 4*, Dec 1999, pg 1 – 9.

Mediators:

[1] Bright, M.W., Hurson, A.R., and Pakzad, S.H., A Taxonomy and Current Issues in Multidatabase Systems, *Computer*, IEEE 1992, pg 50 – 59.

[2] Wiederhold, G., Mediators in the Architecture of Future Information Systems, *Computer*, IEEE 1992, pg 38 – 59.

Multimedia:

[1] Duke, D.J., Herman, I., A Standard for Multimedia Middleware, *ACM Multimedia'98*, Aug 1998, pg 381 – 390.

[2] Wimmers, E.L., Using Fagin's Algorithm for Merging Ranked Results in Multimedia Middleware.

AOA:

[1] Karne, R.K., Gattu, R., Dandu, R., Zhang, X., Application-oriented Object Architecture: A Revolutionary Approach, Poster Paper, *6th International Conference on High Performance Computing*, December 16-19, Bangalore, India, 2002.

[2] Karne, R.K., Gattu, R., Dandu, R., Zhang, X. Application-oriented Object Architecture: Concepts and Approach, IASTED *International Conference, Networks, Parallel and Distributed Processing and Applications (NPDPA 2002)*, Tsukuba, Japan, October 2002.

XML:

[1] Baru, C., XML-based Information Mediation for Digital Libraries, *Proceeding of the 4th ACM Conference on Digital Libraries*, Aug 1999, pg 214 – 215.

[2] Christophides, V., Cliet, S., and Simèon, J., On Wrapping Query Languages and Efficient XML Integration, *Proceedings of the 2000 ACM SIGMOD on Management of Data, Vol. 29, Issue 2*, May 2000, pg 141 - 152.

[3] Datadirect Whitepaper, Using XML with ODBC and JDBC, pg 1 – 12.

[4] DataMirror Whitepapers, Managing your Data the XML Way: Data Transformation, Exchange and Integration, http://www.datamirror.com/resourcecenter/download/dbxmlvision/pdfs/dbxmltransform/XML_solutions.pdf, found 2/14/2002.

[5] Fernanadez, M., Morishima, A., and Suciu, D., Efficient Evaluation of XML Middle-ware Queries, *ACM SIGMOD Record , Proceedings of the 2001 ACM SIGMOD international conference on Management of data, Volume 30 Issue 2* , May 2001,pg 103 – 114.

[6] Jacobsen, H. and Günther, O., Middleware for Software Leasing Over the Internet, *Proceedings of the First ACM Symposium on Electronic Commerce*, Nov 1999, pg 87 – 95.

Security:

[1] Shands, D., et al, Secure Virtual Enclaves, *ACM Transactions on Information and System Security (TISSEC), Vol. 4, Issue 2*, May 2001, pg 103 – 133.

[2] Stillerman, M., Marceau, C., and Stillman, M., Intrusion Detection for Distributed Applications, *Communications of the ACM, Vol. 42, Issue 7*, Jul 1999, pg 62 – 69.

Distributed Processing & Computing:

[1] Bernstein, P.A., Middleware: A model for Distributed System Services, *Communications of the ACM, Vol. 39, No. 2*, February 1996, pg 86 – 98.

[2] Chiang, C., A Distributed Object Computing Architecture for Leveraging Software Engineering Systems, *Proceeding of the 16th ACM SAC2001 Symposium on Applied Computing*, Mar 2001, pg 653 – 657.

[3] Dashofy, E., Medvidovic, N., and Taylor, R., Using Off-The-Shelf Middleware to Implement Connectors in Distributed Software Architectures, *Proceedings of the 1999 International Conference on Software Engineering (ICSE '99)*, May 1999, pg 3 - 12.

[4] Fox, G., et al, The Gateway System: Uniform Web Based Access to Remote Resources, *Proceedings of the ACM 1999 conference on Java Grande*, June 1999, pg 1 – 8.

[5] Fraternali, P., Tools and Approaches for Developing Data-Intensive Web Applications: A Survey, *ACM Computing Survey (CSUR), Vol. 31, Issue 3*, Sep 1999, pg 227 – 263.

[6] Gimenez, G. and Kim, K., A Windows CE Implementation of a Middleware Architecture Supporting Time-Triggered Message-Triggered Objects, *Proceedings of the 25th Annual International Computer Software and Applications Conference (COMPSAC01)*, Jun 2000.

[7] Graham, J., Cepull, J., Early Adopters an Internet 2 Middleware Project, *Proceedings of the 28th annual ACM SIGUCCS conference on User services: Building the future*, October 2000, pg 86 – 91.

[8] Gui, X., An Object-oriented Middleware for our Metasystem on Internet, *Proceedings of the 36th International Conference on Technology of Object-Oriented Languages and Systems*, October 2000.

[9] Jeng, J., Towards a Universal Service-Computing Platform via Virtual Service Machine, *Proceedings of the 16th ACM SAC2001 Symposium on Applied Computing*, Mar 2001, pg 663 – 667.

[10] Johnson, S., et al, Experiences with Group Communication Middleware, *Proceedings of the IEEE International Conference on Dependable Systems and Networks (DSN)*, Jun 2000.

[11] Jones, C., et al, A Survey of Energy Efficient Network Protocols for Wireless Networks, *Wireless Networks, Vol. 7, Issue 4*, Sep 2001, pg 343 - 358.

[12] Jung, D., Paek, K., and Kim, T., Design of MOBILE MOM: Message Oriented Middleware Service for Mobile Computing, *IEEE International Workshops on Parallel Processing*, Sep 1999.

[13] Kaveh, N., Emmerich, W., Deadlock Detection in Distributed Object Systems, *ACM SIGSOFT Software Engineering Notes , Proceedings of the 8th European software engineering conference held jointly with 9th ACM SIGSOFT international symposium on Foundations of software engineering, Volume 26 Issue 5* ,September 2001, pg 44 – 51.

[14] Kim, K.H., Object-Oriented Real-Time Distributed Programming and Support Middleware, *7th International Conference on Parallel and Distributed Systems (ICPADS'00)*, July 2000.

[15] Meier, M.S., Miller, K.L., and Pazel, D.P., Experiences with Building Distributed Debuggers, *Proceedings of the SIGMETRICS symposium on Parallel and distributed tools*, January 1996, pg 70 – 79.

[16] Mills, K., Introduction to the Electronic Symposium on Computer-Supported Cooperative Work, *ACM Computing Surveys (CSUR), Vol. 31, Issue 2*, Jun 1999, pg 105 - 115.

[17] Nakajima, T., Towards Universal Software Substrate for Distributed Embedded Systems, *Proceedings of the 6th International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS'01)*, January 2001.

[18] O'Ryan, C., et al, The Design and Performance of a Pluggable Protocols Framework for Real-Time Distributed Object Computing Middleware, *IFIP/ACM International Conference on Distributed System Platforms*, Apr 2000, pg 372 – 393.

[19] Phan, T., Guy, R., and Bagrodia, R., A Scalable, Distributed Middleware Service Archicture to Support Mobile Internet Applications, *Proceedings of the First Workshop on Wireless Mobile Internet*, Jul 2001, pg 27 – 33.

[20] Putman, J., General Framework for Fault Tolerance from ISO/ITU Reference Model for Open Distributed Processing (RM-ODP) , *Workshop on Object-Oriented Real-Time Dependable Systems (WORDS)*; Nov 18 – 20, 1999, pg 1 – 8.

[21] Pyarali, I., O'Ryan, C., and Schmidt, D., A Pattern Language for Efficient, Predictable, Scalable, and Flexible Dispatching Mechimisms for Distributed Object Computing Middleware, *3rd IEEE/IFIP International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC)*, Mar 2000, pg 1 - 8.

[22] Rackl, G., et al, MIMO - An Infrastructure for Monitoring and Managing Distributed Middleware Environments, *IFIP/ACM International Conference on Distributed System Platforms*, Apr 2000, pg 71 - 87.

[23] Ranganathan, M., et al, Mobile Streams: A Middleware for Reconfigurable Distributed Scripting, *Proceedings for the 1st International Symposium on Agent Systems and Applications & 3rd International Symposium on Mobile Agents*, Oct 1998, pg 1 – 14.

[24] Tai, S., Rouvellou, I., Strategies for Integrating Messaging and Distributed Object Transactions, *Middleware 2000*, pg 308 - 330.

[25] Welch, L.R., Distributed, Scalable, Dependable Real-Time Systems: Middleware Services and Applications.

Tutorials/Whitepapers:

[1] Dolgicer, M., Bayer, G., and Bardash, M., Java Servlets and Enterprise Java Beans In Enterprise Architectures, *International Systems Group (ISG), Inc*, 2003.

[2] Vinoski, S., Scalability Issues in CORBA-based Systems, *Proceedings of the 22nd international conference on Software engineering***,** June 2000, pg 826.

References in alphabetical sequence:

[1] Agha, G., Adaptive Middleware, *Communication of the ACM, Vol. 45, No. 6*, Jun 2002, pg 31 – 32.

[2] Altinel, M., et al, DBIS-Toolkit: Adaptive Middleware for Large Scale Data Delivery, *Proceedings of the 1999 International Conference on Management of Data (SIGMOD '99), Vol. 28, Issue 2,* May 1999, pg 544 - 546.

[3] Arpirez, J.C., et al, WebODE: a Scalable Workbench for Ontological Engineering, *Proceedings of the international conference on Knowledge capture*, October 2001, pg 6 – 13.

[4] Astley, M. and Agha, G., Customization and Composition of Distributed Objects: Middleware Abstractions for Policy Management, *ACM SIGSOFT Software Engineering Notes, Proceeding of the ACM SIGSOFT 6th International Symposium on Foundations of Software Engineering,Vol. 23, Issue 6*, Nov 1998, pg 1 - 9.

[5] Atkinson, R., Use of a Component Architecture in Integrating Relational and Non – Relational Storage Systems, *ACM SIGMOD Record , Proceedings of the 1995 ACM SIGMOD international conference on Management of data, Volume 24 Issue 2*, May 1995, pg 454.

[6] Baru, C., XML-based Information Mediation for Digital Libraries, *Proceeding of the 4th ACM Conference on Digital Libraries*, Aug 1999, pg 214 – 215.

[7] Benda, M., The Architecture of Global Access, *IEEE Internet Computing*, Jan/Feb 1997, pg 78 – 80.

[8] Bernstein, P.A., Middleware: A model for Distributed System Services, *Communications of the ACM, Vol. 39, No. 2*, February 1996, pg 86 – 98.

[9] Blair, G.S., et al, The role of software architecture in constraining adaptation in component –based middleware platforms, *IFIP/ACM International Conference on Distributed systems platforms,* 2000, pg 164 – 184.

[10] Bright, M.W., Hurson, A.R., and Pakzad, S.H., A Taxonomy and Current Issues in Multidatabase Systems, *Computer*, IEEE 1992, pg 50 – 59.

[11] Bronsard, F., et al, Toward Software Plug and Play, *ACM SIGSOFT Software Engineering Notes, Proceedings of the 1997 Symposium on Software Reusability, Vol. 22, Issue 3*, May 1997, pg 19 – 29.

[12] Brown, A., Mastering the Middleware Muddle, *IEEE Software*, Jul/Aug 1999, pg 18 – 21.

[13] Brunsch, D., O'Ryan, C., and Schmidt, D., Designing an Efficient and Scalable Server-side Asynchrony Model for CORBA, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg 223 – 229.

[14] Campbell, A., Coulson, G., and Kounavis, M., Managing Complexity: Middleware Explained, *IT Pro*, Sep/Oct 1999, pg 22 – 28.

[15] Canal, C., et al, Extending CORBA Interfaces with Protocols, *British Computer Society 2001, Vol. 44, No. 5*, pg 448 – 462.

[16] Capra, L., Exploiting reflection in mobile computing middleware, *ACM SIGMOBILE Mobile Computing and Communications Review, Vol. 6, Issue 6*, 2002, pg 34 – 44.

[17] Carew, M., The Anatomy of a Component.

[18] Caron, O., Carré, B., and Debrauwer, L., An Original View Mechanism for the CORBA Middleware*, IEEE Proceedings of the Technology of Object-Oriented Language and Systems (TOOLS 33)*, Jun 2000.

[19] Carzaniga, A., et al, Issues in Supporting Event-Based Architectural Styles, *Proceedings of the 3rd International Workshop on Software Architecture*, Nov 1997, pg 17 – 20.

[20] Charles, J., Middleware Moves to the Forefront, *Computer*, May 1999,pg 17 - 19.

[21] Chiang, C., A Distributed Object Computing Architecture for Leveraging Software Engineering Systems, *Proceeding of the 16th ACM SAC2001 Symposium on Applied Computing*, Mar 2001, pg 653 – 657.

[22] Christophides, V., Cliet, S., and Simèon, J., On Wrapping Query Languages and Efficient XML Integration, *Proceedings of the 2000 ACM SIGMOD on Management of Data, Vol. 29, Issue 2*, May 2000, pg 141 - 152.

[23] Conrad, C., Stiller, B., The Design of an Application Programming Interface for QoS-based Multimedia Middleware, *Proceedings of the 22nd IEEE Conference on Local Computer Networks (LCN'97)* , pg 274 – 283 .

[24] Costa, F. and Blair, G., Integration Meta-Information Management and Reflection in Middleware*, Proceedings of the IEEE International Symposium on Distributed Objects and Applications (DOA)*, Sep 2000.

[25] Coulson, G., A Configurable Multimedia Middleware Platform, *IEEE Multimedia*, Jan/Mar 1999, pg. 62 – 76.

[26] Coulson, G., What is Reflective Middleware? , RM article.

[27] Dao, S., Toward Scalability and Interoperability of Heterogeneous Information Sources, *11th International Conference on Data Engineering*, pg 65 – 67.

[28] Dashofy, E., Medvidovic, N., and Taylor, R., Using Off-The-Shelf Middleware to Implement Connectors in Distributed Software Architectures, *Proceedings of the 1999 International Conference on Software Engineering (ICSE '99)*, May 1999, pg 3 - 12.

[29] Datadirect Whitepaper, Using XML with ODBC and JDBC, pg 1 – 12.

[30] DataMirror Whitepapers, Managing your Data the XML Way: Data Transformation, Exchange and Integration, http://www.datamirror.com/resourcecenter/download/dbxmlvision/pdfs/dbxmltransform/XML_solutions.pdf, found 2/14/2002.

[31] Degenaro, L., et al, A Middleware System which Intelligently Caches Query Results, *IFIP/ACM International Conference on Distributed Systems Platforms*, 2000, pg 24 – 44.

[32] De Miguel, M.A, General Framework for the Description of QoS in UML, *Proceedings of the Sixth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'03)*.

[33] Di Nitto, E. and Rosenblum, D., Exploiting ADLs to Specify Architectural Styles Induced by Middleware Infrastructures, *ACM Proceedings of the 1999 International Conference on Software Engineering*, May1999, pg 13 – 22.

[34] Dogac, A., Dengi, C., and Öszu, M.T., Distributed Object Computing Platforms, *Communications of THE ACM*, September 1998/Vol. 41, No. 9,pg 95 – 103.

[35] Duke, D.J., Herman, I., A Standard for Multimedia Middleware, *ACM Multimedia'98*, Aug 1998, pg 381 – 390.

[36] Eisenstein, J., Rich, C., Agents and GUIs from Task Models, *Proceedings of the 7th international conference on Intelligent user interfaces*, January 2002, pg 47 – 54.

[37] Eliassen, F., et al, Next Generation Middleware: Requirements, Architecture, and Prototypes, *7th IEEE Workshop on Future Trends on Distributed Computing Systems*, Dec 2000.

[38] Elmagarmid, A.K., Pu, C., Guest Editors' Introduction to the Special Issue on Heterogeneous Databases, *ACM Computing Surveys, Vol. 22, No. 3*, September 1990, pg 175 – 178.

[39] Emmerich, W., Software Engineering and Middleware: A Roadmap, *ACM Proceedings of the Conference on the Future of Software Engineering*, May 2000, pg 119 – 129.

[40] Fayad, M., Introduction to the Computing Surveys' Electronic Symposium on Object-Oriented Application Frameworks, *ACM Computing Surveys (CSUR)*, Mar 2000.

[41] Fernanadez, M., Morishima, A., and Suciu, D., Efficient Evaluation of XML Middle-ware Queries, *ACM SIGMOD Record , Proceedings of the 2001 ACM SIGMOD international conference on Management of data, Volume 30 Issue 2* , May 2001,pg 103 – 114.

[42] Fox, G., et al, The Gateway System: Uniform Web Based Access to Remote Resources, *Proceedings of the ACM 1999 conference on Java Grande,* June 1999, pg 1 – 8.

[43] Francois, A.R.J., Medioni, G.G., A Modular Middleware Flow Scheduling Framework, *Proceedings of the eighth ACM international conference on Multimedia,* October 2000,pg 371 – 374.

[44] Fraternali, P., Tools and Approaches for Developing Data-Intensive Web Applications: A Survey, *ACM Computing Survey (CSUR), Vol. 31, Issue 3*, Sep 1999, pg 227 – 263.

[45] Gimenez, G. and Kim, K., A Windows CE Implementation of a Middleware Architecture Supporting Time-Triggered Message-Triggered Objects, *Proceedings of the 25th Annual International Computer Software and Applications Conference (COMPSAC01)*, Jun 2000.

[46] Gokhale, A., et al, Applying model-integrated computing to component middleware and enterprise applications, *Communication of the ACM, Vol. 45, Issue 10*, October 2002,pg 65 – 70 .

[47] Graham, J., Cepull, J., Early Adopters an Internet 2 Middleware Project, *Proceedings of the 28th annual ACM SIGUCCS conference on User services: Building the future*, October 2000, pg 86 – 91.

[48] Gui, X., An Object-oriented Middleware for our Metasystem on Internet, *Proceedings of the 36th International Conference on Technology of Object-Oriented Languages and Systems*, October 2000.

[49] Gutzman, A.D., Selecting the Appropriate Middleware for Your Web-To-Database Application, *ACM SIGUCCS XXV 1997*, pg 365 – 367.

[50] Hecht, M., et al, OFTT: A Fault Tolerance Middleware Toolkit for Process Monitoring and Control Window NT Application, *Proceedings of the IEEE International Conference on Dependable Systems and Networks*, Jun 2000.

[51] Heiler, S., Extended Data Type Support in Distributed DBMS Products: A Technology Assessment and Forecast, *GTE Laboratories Inc*. pg 1 – 44.

[52] Heuser, L., et al, Development of Distributed and Client/Server Object-Oriented Applications, *Proceedings of the 9th Annual Conference on Object-Oriented Programming Systems, Language, and Applications, Vol. 29, Issue 10*, Oct 1994,pg 317 – 323.

[53] Hofmann, C., A Multi-Tier Framework for Accessing Distributed, Heterogeneous Spatial Data in a Federated Based EIS, *Proceedings of the 7th International Symposium on Advances in Geographic Information Systems*, Nov 1999, pg 140 – 145.

[54] Hou, C., Han, C., and Chen, T., Communication Middleware and Software for QoS Control in Distributed Real-Time Environments, *COMPSAC'97 – 21st International Computer Software and Applications Conference*, Aug 1997.

[55] Issarny, V., Kloukinas, C., and Zarras, A., Systematic Aid for Developing Middleware Architectures, *Communication of the ACM, Vol. 45, No. 6*, Jun 2002, pg 53 – 58.

[56] Jacobsen, H. and Günther, O., Middleware for Software Leasing Over the Internet, *Proceedings of the First ACM Symposium on Electronic Commerce*, Nov 1999, pg 87 – 95.

[57] Jain, R., Anjum, F., and Umar, A., A Comparison of Mobile Agent and Client-Server Paradigms for Information Retrieval Task in Virtual Enterprises, *IEEE*, 2000.

[58] Janka, R., A New Development Framework Based On Efficient Middleware for Real-Time Embedded Heterogeneous Multicomputers, *IEEE Conference and Workshop on Engineering of Computer-Based Systems*, March 1999, pg 261 – 268.

[59] Janka, R.S., Wills, L.M., A Novel Codesign Methodology for Real-Time Embedded COTS Multiprocessor-Based Signal Processing Systems, *Proceedings of the eighth international workshop on Hardware/software codesign***,** May 2000,pg 157 – 161.

[60] Jeng, J., An Approach to Designing Reusable Service Frameworks via Virtual Service Machine, *ACM SIGSOFT Software Engineering Notes, Proceedings of SSR'01 on 2001 Symposium on Software Reusability, Vol. 26, Issue 3*, May 2001, pg 58 – 66.

[61] Jeng, J., Towards a Universal Service-Computing Platform via Virtual Service Machine, *Proceedings of the 16th ACM SAC2001 Symposium on Applied Computing*, Mar 2001, pg 663 – 667.

[62] Johnson, S., et al, Experiences with Group Communication Middleware, *Proceedings of the IEEE International Conference on Dependable Systems and Networks (DSN)*, Jun 2000.

[63] Jones, C., et al, A Survey of Energy Efficient Network Protocols for Wireless Networks, *Wireless Networks, Vol. 7, Issue 4*, Sep 2001, pg 343 - 358.

[64] Jung, D., Paek, K., and Kim, T., Design of MOBILE MOM: Message Oriented Middleware Service for Mobile Computing, *IEEE International Workshops on Parallel Processing*, Sep 1999.

[65] Kalnis, P., Proxy-Server Architectures for OLAP, *ACM SIGMOD Record , Proceedings of the 2001 ACM SIGMOD international conference on Management of data, Volume 30 Issue 2*, May 2001, pg 367 – 378.

[66] Karne, R.K., Gattu, R., Dandu, R., Zhang, X., Application-oriented Object Architecture: A Revolutionary Approach, Poster Paper, *6th International Conference on High Performance Computing*, December 16-19, Bangalore, India, 2002.

[67] Karne, R.K., Gattu, R., Dandu, R., Zhang, X. Application-oriented Object Architecture: Concepts and Approach, IASTED *International Conference, Networks, Parallel and Distributed Processing and Applications (NPDPA 2002)*, Tsukuba, Japan, October 2002.

[68] Kaveh, N., Emmerich, W., Deadlock Detection in Distributed Object Systems, *ACM SIGSOFT Software Engineering Notes , Proceedings of the 8th European software engineering conference held jointly with 9th ACM SIGSOFT international symposium on Foundations of software engineering***,** *Volume 26 Issue 5* ,September 2001, pg 44 – 51.

[69] Kiernan, J., Carey, M.J., Middleware Object Query Processing with Deferred Updates and Autonomous Sources, *ACM SIGPLAN Notices , Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, Volume 35 Issue 10* , October 2000, pg 118 – 129.

[70] Kim, K., et al, A CORBA Service Enabling Programmer-Friendly Object-Oriented Real-Time Distributed Computing, *IEEE 5th International Workshop on Object-Oriented Real-Time Dependable Systems*, Nov 1999.

[71] Kim, K.H., Ishida, M., and Liu, J., An efficient Middleware Architecture Supporting Time-Triggered Objects and an NT-based Implementation.

[72] Kim, K.H., Object-Oriented Real-Time Distributed Programming and Support Middleware, *7th International Conference on Parallel and Distributed Systems (ICPADS'00)*, July 2000.

[73] Kleissner, C., Enterprise Objects Framework: a second generation object-relational enabler, *ACM SIGMOD Record , Proceedings of the 1995 ACM SIGMOD international conference on Management of data, Volume 24 Issue 2*, May 1995, pg 455 – 459.

[74] Kon, F., et al, The case for reflective middleware, *Communications of the ACM, Vol. 45, Issue 6*, June 2002, pg 33 – 38.

[75] Kossmann, D., The State of the Art in Distributed Query Processing, *ACM Computing Surveys, Vol.32, No. 4*, December 2000,pg. 422 – 469 .

[76] Kramp, T., Coulson, G., The Design of a Flexible Communications Framework for Next-Generation Middleware, *IEEE International Symposium on Distributed Objects and Applications,* September 2000.

[77] Krishnamurthy, Y., et al, Integration of QoS-Enabled Distributed Object Computing Middleware for Developing Next-Generation Distributed Applications, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg 230 – 237.

[78] Kruchten, P.B., Thompson, C.J., An Object-Oriented, Distributed Architecture for Large Scale Ada Systems, *Proceedings of the conference on TRI-Ada '94,* Nov 1994, pg 262 – 271.

[79] Kuhns, F., Schmidt, D.C., and Levine, D.L., The Performance of a Real-time I/O Subsystem for QoS-enabled ORB Middleware.

[80] Kutlusan, A., et al, A Combat Management System Middleware Based on CORBA, *The Proceedings of the International Symposium on Distributed Objects and Applications (DOA'00)*, IEEE 2000.

[81] Lewandowski, S.M., Frameworks for Component-Based Client/Server Computing, *ACM Computing Surveys, Vol. 30, No. 1,* March 1998.

[82] Lewis, T., Where is Client/Server Software Headed?, *IEEE Computer, Vol. 28, Issue 4*, Apr 1995, pg 49 - 55.

[83] Li, B. and Nahrstedt, K., QualProbes: Middleware QoS Profiling Services for Configuring Adaptive Applications, *IFIP/ACM International Conference on Distributed System Platforms*, Apr 2000, pg 256 - 272.

[84] Manola, F., An Evaluation of Object-Oriented DBMS Developments 1994 Edition, *GTE Laboratories Inc*,pg 1- 6.

[85] Manola, F., et al, Distributed Object Management, *GTE Laboratories Inc*,pg1 – 41.

[86] Manola, F., Heiler, S., A "RISC" Object Model for Object System Interoperation: Concepts and Applications, *GTE Laboratories Inc.*

[87] Marrón, P., Design of a Middleware Service for Scalable Wide Are Network Applications, *Proceedings of the IEEE International Symposium on Distributed Objects and Applications (DOA),* Sep 2000, pg 1 - 10.

[88] McFall, C., An Object Infrastructure for Internet Middleware, *IEEE Internet Computing*, Mar/Apr 1998, pg 46 - 51.

[89] Medvidovic, N., On the role of middleware in architecture-based software development, *ACM International Conference Proceeding Series-Proceedings of the 14th international conference on Software engineering and knowledge engineering*, 2002, pg 299 – 306.

[90] Meier, M.S., Miller, K.L., and Pazel, D.P., Experiences with Building Distributed Debuggers, *Proceedings of the SIGMETRICS symposium on Parallel and distributed tools*, January 1996, pg 70 – 79.

[91] Melling, W.P., Enterprise Information Architectures – They're Finally Changing, *ACM SIGMOD Record , Proceedings of the 1994 ACM SIGMOD international conference on Management of data, Volume 23 Issue 2* ,May 1994,pg 493 – 504 .

[92] Meng, W., Construction of a Relational Front-end for Object-Oriented Database Systems,IEEE 1993, pg 476 – 483.

[93] Meng, W., Processing Hierarchical Queries in Heterogeneous Environment .

[94] Meng, W., Kamada, A., and Chang, Y., Transformation of Relational Schemas to Object-Oriented Schemas.

[95] Middleware White Paper, Middleware – The Essential Component for Enterprise Client/Server Applications, *International Systems Group (ISG), Inc*, 1997.

[96] Miled, Z.B., et al, Global Change Master Directory : Object-Oriented Active Asynchronous Transaction Management in a Federated Environment Using Data Agents,

Proceedings of the 2001 ACM symposium on Applied computing **,**March 2001, pg 207 – 214 .

[97] Mills, K., Introduction to the Electronic Symposium on Computer-Supported Cooperative Work, *ACM Computing Surveys (CSUR), Vol. 31, Issue 2*, Jun 1999, pg 105 - 115.

[98] Milojičić, D., Middleware's Role, Today and Tomorrow, *IEEE Concurrency, Vol. 7, Issue 2*, Apr/Jun 1999, pg 70 - 80.

[99] Mizunuma, I., Shen, C., and Takegaki, M., Middleware for Distributed Industrial Real-Time Systems on ATM Networks, *Proceedings of the 17th IEEE Real-Time Systems Symposium (RTSS'96)*, pg 32 – 38.

[100] Moreira, R.S., et al, A Reflective Component-Based & Architecture Aware Framework to Manage Architecture Composition, *Proceedings of the Third International Symposium on Distributed-Objects and Applications (DOA'01),* IEEE 2001.

[101] Nakajima, T., Towards Universal Software Substrate for Distributed Embedded Systems, *Proceedings of the 6th International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS'01)*, January 2001.

[102] Narain, S., et al, Middleware for Building Adaptive Systems via Configuration, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg 188 – 195.

[103] Nayeri, F., Addressing Component Interoperability in the OMG Object Model, *ORB Implementer's Workshop*, June 1993, pg 1 – 5 .

[104] O'Ryan, C., et al, Applying a Scalable CORBA Event Service to Large-scale Distributed Interactive Simulation, *IEEE 5th International Workshop on Object-Oriented Real-Time Dependable Systems*, Nov 1999, pg 1 - 8.

[105] O'Ryan, C., et al, The Design and Performance of a Pluggable Protocols Framework for Real-Time Distributed Object Computing Middleware, *IFIP/ACM International Conference on Distributed System Platforms*, Apr 2000, pg 372 – 393.

[106] Othman, O. and Schmidt, D., Issues in the Design of Adaptive Middleware Load Balancing, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg 205 – 213.

[107] Petriu, D., et al, Using Analytic Models for Predicting Middleware Performance, *ACM Proceedings on the 2nd International Workshop on Software and Performance*, Sep 2000, pg 189 – 194 .

[108] Phan, T., Guy, R., and Bagrodia, R., A Scalable, Distributed Middleware Service Archicture to Support Mobile Internet Applications, *Proceedings of the First Workshop on Wireless Mobile Internet*, Jul 2001, pg 27 – 33.

[109] Putman, J., General Framework for Fault Tolerance from ISO/ITU Reference Model for Open Distributed Processing (RM-ODP) , *Workshop on Object-Oriented Real-Time Dependable Systems (WORDS)*; Nov 18 – 20, 1999, pg 1 – 8.

[110] Pyarali, I., et al, Evaluating and Optimizing Thread Pool Strategies for Real-Time CORBA, *ACM SIGPLAN Notices, Vol. 36, Issue 8*, Aug 2001, pg. 214 – 221.

[111] Pyarali, I., O'Ryan, C., and Schmidt, D., A Pattern Language for Efficient, Predictable, Scalable, and Flexible Dispatching Mechimisms for Distributed Object Computing Middleware, *3rd IEEE/IFIP International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC)*, Mar 2000, pg 1 - 8.

[112] Rackl, G., et al, MIMO - An Infrastructure for Monitoring and Managing Distributed Middleware Environments, *IFIP/ACM International Conference on Distributed System Platforms*, Apr 2000, pg 71 - 87.

[113] Ranganathan, M., et al, Mobile Streams: A Middleware for Reconfigurable Distributed Scripting, *Proceedings for the 1st International Symposium on Agent Systems and Applications & 3rd International Symposium on Mobile Agents*, Oct 1998, pg 1 – 14.

[114] Rodriguez-Martinez, M., and Roussopoulos, N., MOCHA: A Self-Extensible Database Middleware System for Distributed Data Sources, *ACM 2000 1-58113-218-2/00/05*, May 2000, pg 213 – 224.

[115] Saksena, M., Towards Automatic Synthesis of QoS Preserving Implementations from Object-Oriented Design Models.

[116] Scheuermann, P., et al, Report on the Workshop on Heterogeneous Database Systems held at Northwestern University, *SIGMOD RECORD, Vol. 19,No. 4*, December 1990,pg 23 – 31.

[117] Schmidt, D., Middleware for Real-Time and Embedded Systems, *Communications of the ACM, Vol. 45, No. 6*, Jun 2002, pg 43 – 48.

[118] Schmidt, D.C., Buschmann, F., Patterns, Frameworks, and Middleware: Their Synergetic Relationships, *25th International Conference on Software Engineering*, May 2003, pg 694 – 704.

[119] Schmidt, D.C., Kuhns, F., An Overview of the Real-Time CORBA Specification, *Computer*, June 2000, pg 56 – 63.

[120] Serbedżija, N., Developing Middleware for Web-aware Systems: Lessons Learned, *Proceedings of the Australasian Computer Science Conference*, Feb 2000.

[121] Shands, D., et al, Secure Virtual Enclaves, *ACM Transactions on Information and System Security (TISSEC), Vol. 4, Issue 2*, May 2001, pg 103 – 133.

[122] Slivinskas, G., Jensen, C., and Snodgrass, R., Adaptable Query Optimization and Evaluation in Temporal Middleware, *Proceedings of the 2001 ACM SIGMOND International Conference on Management of Data on Management of Data, May 2001, Vol. 30 Issue 2*, May 2001, pg 127 – 138.

[123] Staab, S., et al, An extensible Approach for Modeling Ontologies in RDF (S), *ECDL 2000 Workshop on the Semantic Web*, pg 1 – 10.

[124] Stillerman, M., Marceau, C., and Stillman, M., Intrusion Detection for Distributed Applications, *Communications of the ACM, Vol. 42, Issue 7*, Jul 1999, pg 62 – 69.

[125] Strand, E.J., Mehta, R.P., and Jairam, R., Applications Thrive on Open Systems Standards, *StandardView Vol. 2, No. 3, September/1994*, pg 148 – 154.

[126] Subrahmanian, V.S., Amalgamating Knowledge Bases, *ACM Transactions on Database Systems, Vol. 19, No. 2* , June 1994, pg 291 – 331.

[127] Tai, S., Rouvellou, I., Strategies for Integrating Messaging and Distributed Object Transactions, *Middleware 2000*, pg 308 - 330.

[128] Thompson, C., et al, Intermediary Architecture: Interposing Middleware Object Services between Web Client and Server, *ACM Computing Surveys (CSUR)*, Jun 1999.

[129] Thompson, J., Avoiding a Middleware Muddle, *IEEE Software*, Nov/Dec 1997, pg 92 - 96.

[130] Tian, Y., Middleware Design Issues for Application Management in Heterogeneous Networks, *The Proceedings of the IEEE International Conference on Networks (ICON'00)*. September 2000.

[131] Tripathi, A., Challenges Designing Next-Generation Middleware Systems, *Communications of the ACM, Vol. 45, No. 6*, Jun 2002, pg 39 – 42.

[132] Truyen, E., et al, Customization of Component-based Object Request Brokers through Dynamic Reconfiguration, *Technology of Object-Oriented Languages and Systems (TOOLS 33) IEEE,* June 2000.

[133] Tsaoussidis, V., A CORBA-based Application Service Middleware Architecture and Implementation.

[134] Ventrone, V., Heiler, S., Some Practical Advice for Dealing with Semantic Heterogeneity in Federated Database Systems, pg 1 – 20.

[135] Vinoski, S., Scalability Issues in CORBA-based Systems, *Proceedings of the 22nd international conference on Software engineering***,** June 2000, pg 826.

[136] Wang, N., Kircher, M., and Schmidt, D., Applying Reflective Middleware Techniques to Optimize a QoS-enabled CORBA Component Model Implementation, *Proceedings of the 24th Annual International Computer Software and Applications Conference (COMPSAC)*, Oct 2000, pg 1 - 8.

[137] Whetten, B., Message-Based Computing: The Fourth Wave of Integration, http://www.messageq.com/communications_middleware/whetten_1.html, found 2/14/2002.

[138] Wiederhold, G., Mediation in Information Systems, *ACM Computing Surveys, Vol. 27, No, 2,* June 1995.

[139] Wiederhold, G., Mediators in the Architecture of Future Information Systems, *Computer*, IEEE 1992, pg 38 – 59.

[140] Wiil, U., Nürnberg, P., and Leggett, J., Hypermedia Research Directions: An Infrastructure Perspective*, ACM Computing Survey (CSUR), Vol. 31, Issue 4*,Dec 1999, pg 1 – 9.

[141] Wiil, U.K., Evolving Hypermedia Middleware: Lessons and Observations, *Proceedings of the 1999 ACM symposium on Applied computing***,** February 1999, pg 427 – 436.

[142] Wimmers, E.L., Using Fagin's Algorithm for Merging Ranked Results in Multimedia Middleware.

[143] Yamamoto, G., Nakamura, Y., Architecture and Performance Evaluation of a Massive Multi-Agent System, *Proceedings of the third annual conference on Autonomous Agents* , April 1999, pg 319 – 325.

[144] Yellin, D., Stuck in the Middle: Challenges and Trends in Optimizing Middleware, ACM *SIGPLAN Notices*, Aug 2001, pg 175 – 180.

[145] Yoo, S.B., Kim, K.C., and Cha S.K., A Middleware Implementation of Active Rules for ODBMS.

[146] Yu, C., et al, Translation of Object – Oriented Queries to Relational Queries, IEEE 1995, pg 90 – 97.

[147] Zhang, C., Jacobsen, H., Quantifying aspects in middleware platforms. *Proceedings of the 2nd international conference on Aspect-oriented software development,* March 2003, pg 130 - 139.

[148] Zhou, J., Zhou, M., and Wu, Q., An Agent Framework Based on Distributed Object, *IEEE Technology of Object-Orient Languages and Systems (TOOLS 33)*, Jun 2000.