ECCO: Event Brokering in a Distributed Adaptive Mobile Environment

Jean Bacon Jean.Bacon@cl.cam.ac.uk

University of Cambridge Computer Laboratory

Part 1: Previous research and track record

This proposal seeks funding for a PhD student to carry out research into an Event Brokering System on messaging middleware in a distributed mobile environment. Below, the background of the Opera group in the University of Cambridge Computer Laboratory and the proposed PhD student (Eiko Yoneki) are described. Part 1 references (1-8) are included with Part 2.

Jean Bacon, Opera Group, CL

Jean Bacon is a Reader in Distributed Systems at the University of Cambridge Computer Laboratory. She co-leads the Opera research group where the focus is on the design and deployment of open, large-scale, widely distributed systems.
Two major thrusts are in the areas of asynchronous middleware, originating with the Cambridge Event Architecture (CEA), and an open access control architecture for secure interworking services (OASIS). An overview of the work is given in [1, 2].

A recent grant in the area of this proposal is Global Computing Using Events, EPSRC GR/M22413. Our *publish, register, notify* paradigm was incorporated into standard CORBA, using CORBA-IDL to specify events. Subsequently, we added event storage to the architecture and used ODL for event typing and OQL for querying event stores. The software is available from our web site, http://www.cl.cam.ac.uk/Research/SRG/opera/projects /index.html. Its application in a pervasive computing environment is demonstrated in [3].
The focus in CEA and its extensions was to augment object-oriented middleware (then prevalent), which is based on synchronous method invocation, with asynchronous event-based communication. Even so, all communicating parties have to be mutually aware and simultaneously active which is appropriate for a localised pervasive computing environment or a federation of such environments. We went on to develop Hermes [4] in which we have investigated the use of XML and an event-broker system for wide-area, scalable event transmission, while retaining CEA's notion of event type, type hierarchy and super-type subscriptions. Using typed events and type hierarchies helps in federating event systems; we use gateway event services to negotiate contracts for cross-domain event registration and notification [6]. A scalable routing algorithm using an overlay routing network avoids global broadcasts by creating rendezvous nodes. Fault-tolerance mechanisms that can cope with different kinds of failures in the middleware are integrated with the routing algorithm resulting in a scalable and robust system. A list of our publications in the area of event-based systems can be found at
http://www.cl.cam.ac.uk/Research/SRG/opera/projects/ This proposal is to provide support for mobile senders and receivers.

In addition to grant-focused research, Jean Bacon has supervised graduate students at the Computer Laboratory in distributed systems since 1985, leading to 19 PhD awards and a further seven in their first to fourth years. She has examined 19 other internal and 12 external, international PhDs in related topics. Employment of her recent graduate students includes Citrix Research UK, AT&T Research UK, University Lecturers in Cambridge, Singapore and Egypt, Telcordia US, Cisco US, Fore UK, FutureTV Cambridge, Microsoft Research Cambridge.

Jean Bacon was Editor-in-Chief of IEEE Concurrency during 1999 and 2000. From mid 2000, she was EIC of the IEEE Computer Society's first online magazine, Distributed Systems Online, http://computer.org/dsonline. She was recently elected to the IEEE-CS Board of Governors for 2002-2004. Recent conference participation includes:

- ACM SIGOPS European Workshop Programme Chair 1998
- ACM Symposium on Operating Systems Principles (SOSP), Program Committee 1999
- IEEE International Conference on Distributed Computing Systems, ICDCS, PC 1998 Vice chair middleware 2004
- IFIP/ACM Middleware PC 1998, 2000, 2001, 2003
- POS9 PC 2000,
- Data Engineering PC 2001
- ACM SACMAT PC 2002, 2003
- Proposer and organizer of ICDCS workshop on event-based systems (DEBS02), Vienna, July 2002
- VLDB-TES02 Technologies for E-Services, PC 2002
- Percom (Pervasive Computing) PC 2003

Eiko Yoneki (please see appended work experience details)

Eiko has extensive experience as a software engineer. Through her work with IBM she had several opportunities to collaborate closely with researchers at IBM Almaden Research, IBM Zurich Research, IBM Heidelberg Network Laboratory and IBM Yorktown Heights Research (see enclosed Work Experience). All of these projects were shipped as strategic IBM products. She has experience in software engineering with extensive work in networking and communications, client/server software, object-oriented programming and product development. She is an experienced software developer and is fluent in most mainstream object oriented and structured programming languages as well as in network protocols (OSI, TCP/IP, NETBIOS, HTTP, etc.), frameworks, and APIs.

In the WebExpress project (1996-8), she worked on a pioneering project (Java API) that includes a notification mechanism between a C++ server and a Java Client over different address spaces and structured event subscriptions for event traffic control. An IBM networking division technical award was given for this achievement. Since 1996, she has been working intensively on Java-based Internet middleware over mobile/wireless and network computer environments with optimization of protocol streams, security, and asynchronous messaging models.

She received a postgraduate Diploma in Computer Science from the University of Cambridge in 2002. Her dissertation “A MobileGateway with a Publish/Subscribe Paradigm over a Wireless Network”, is a middleware system for mobile applications with messaging as a basis. Her system **Pronto** [7] uses the Java Message Service (JMS) [34] with extensions and provides a solution for mobile application-specific problems such as resource constraints, network characteristics, and data optimisation. The publish-subscribe paradigm was integrated into the mobile environment and an ***intelligent gateway*** was introduced as a ***message hub*** to transmit information using store-and-forward messaging. The dissertation obtained an award and two papers [7,8] have been accepted for publication

This proposal is for support for Eiko to carry out PhD research in the Opera group, under Jean Bacon’s supervision, on current challenges in event-based distributed systems including mobile/wireless environments. Her previous experience, both in industrial research and through her Cambridge Diploma dissertation work, makes her ideally suited to this endeavour.

Part 2: Proposed research
2.1 Introduction and Background

Client-end computing devices are becoming increasingly mobile and are based on different communication models. Large scale distributed systems should offer load sharing and load reduction for good performance. These key issues are more complex for mobile/wireless-based applications and web-based services. Characteristics of mobile environments and wireless networks and corrsponding requirements on middleware are as follows:

- Mobile devices have small ROM/RAM footprints and low CPU and power usage. Remote processing and resource sharing should be offered by the middleware.
- Wireless networks are increasingly packet-oriented. With a bearer such as GPRS (General Packet Radio Service) or UMTS (Universal Mobile Telecommunications System) users typically pay only for the time they communicate data. Reducing data size for transmission is important.
- Because of low bandwidth, high latency, and frequent disconnections, a middleware should provide an interface to applications that allows maintenance of communication during disconnected operation.
- A data source can be interpreted in different formats and semantics depending on the specifications of mobile devices and wireless networks. Semantic transcoding technology [29] provides better efficiency in data flow.
- There are various bearers such as 2G, 2.5G, 3G, Bluetooth, and IEEE 802.11, and many devices are non-programmable. A middleware needs to offer an interface with a communication abstraction and a gateway/proxy for the clients’ devices.
- An ad-hoc network, a feature of some mobile/wireless networks, is a dynamically re-configurable wireless network without a fixed infrastructure that does not require the intervention of a centralised access point.

The architecture of a distributed system at this level needs careful consideration, and it is essential to provide core functionality for such a system as a ***semantics-based middleware***; that is, resources must be tailored to the required services and applications to give seamless access and unrestricted mobility.

Evolution of communication for loosely coupled, mobile components

As discussed in Part 1, closely coupled (synchronous) communication, as provided in object oriented middleware (OOM) such as Java RMI and CORBA, is not appropriate for ***loosely coupled***, and/or ***mobile***, senders and receivers. OOM-providers added (asynchronous) message passing extensions but static dependencies remained and none supported wide-area, large-scale interworking where participants might be anonymous, might detach or move. The minimum requirement for these is Message Oriented Middleware (MOM), underlying ***event-based systems***, so that heterogeneous, distributed, and dynamically changing components of large information systems can interoperate. This is acknowledged in the business domain through the provision of Sun’s JMS API [26] as a common interface for Java applications to IBM’s MQSeries [25], Microsoft Message Queue (MSMQ), TIBCO’s TIB/Rendezvous [39], Softwired’s iBus [37], and BEA’s WebLogic [41].
In the research domain, in addition to Hermes [4] described in Part 1, the Siena project [10,11,12] created a distributed event service. The three main components, event sources, event sinks, and event brokers can be interconnected in an arbitrary topology. Gryphon [24] is a distributed topic-based and content-based message brokering system that uses an information flow approach to model its behaviour. The Grid Event Service [14] is designed for a large, decentralized network of services. It groups event brokers into a hierarchical cluster topology and supports reliable delivery of events by using persistence features. Narada [30] is based on the Grid Event Service and implements the JMS API. None of these systems support mobile sources and sinks with substantial consideration of the mobile computing specific environment.

Peer-to-Peer (P2P), a new paradigm

A complementary type of distributed system, based on the peer-to-peer network, emerged independently of the enterprise domain. The peer-to-peer style interaction model facilitates sophisticated resource sharing environments between peers over the edges of the Internet. Peer interactions involve advertising resources, search and subsequent discovery of resources, request for access to these resources, responses to these requests, and exchange of messages between peers. Scribe [42] from Microsoft is a topic-centric publish/subscribe messaging system based on the peer-to-peer platform Pastry [33]. Pastry utilizes routing mechanisms to achieve great scalability. Scribe depends on Pastry to route messages to the destinations. Siena [10,11,12], Gryphon [24], Hermes [4] and Narada [30] are also built over peer-to-peer network.

Some initiatives were recently made towards building libraries or frameworks for deploying peer-to-peer applications. Sun Microsystems' JXTA [38] is a library specification for peer-to-peer computing, defining three layers: a core layer, a service layer, and an application layer. The application layer wraps all the applications that are developed by JXTA programmers. The service layer contains services simplifying the development task for the programmer. The JXTA community currently implements various services such as protocols for service discovery and many-to-many communication. The core JXTA layer consists of protocols ensuring basic communication between peers, message routing or peer group creation. We intend to start from JXTA, extending it to support mobility.

With the emergence of peer-to-peer infrastructures, new forms of decoupled interactions are needed. It is challenging to devise a simple abstraction that can directly fit these architectures and potentially be supported by future Internet-wide operating systems. Creating a ***grid of event brokers*** is an interesting key technique to provide reliability and scalability in an event-based distributed system. Event brokers could form a group to provide scalability at the cluster level; which may then be linked together for geographic scaling through link bundles that provide redundant connections between groups.

Integration of event-based systems and mobile computing is immature

In the enterprise domain, messaging is becoming the dominant style of communication among components, because it supports decoupled interactions. However, intranets with centralised administration are still predominant. Recent extensions to mobile computing are done by adding an edge server to manage mobile devices, as in Softwired iBus/Mobile [37]. iBus/Mobile is designed as an extension of J2EE application servers. It includes a messaging middleware client library compatible with the JMS standard as well as a middleware gateway used to connect mobile applications to J2EE application servers. It supports mobile communication-specific protocols such as GPRS, UMTS, and CDPD. This approach is not fully scalable to more dynamically extended business over the Internet via web services or mobile computing.

In the research domain, peer-to-peer network based messaging has been evolving. Because of the power of the publish/subscribe paradigm, this will be the way to establish communication platform over wide area networks. However, support for mobile computing is not yet sufficient. Most research is focusing on integrating ad-hoc networks into peer-to-peer networks, but this will not support mobile computing in its full generality. IBM released the WebSphere Event Broker with MQ Everyplace for the integration of event-based systems with mobile applications. These decentralised event-based systems are still immature and fragmented; integrating a mobile environment requires architecting event-based systems on both wired and mobile networks from a unified viewpoint [16,17, 28]. Thus, there is a strong need for research on how we can abstract messaging in hybrid mixed environments.

Pronto (by **Eiko Yoneki** [7]) addresses design issues for a messaging system in mobile environments and provides a lightweight MOM client, a gateway as a message hub with store-and-forward messaging, and Serverless, MOM-based IP multicast over JMS environments. Pronto offered a similar concept to an event broker over a JMS environment addressing data optimization through semantic transcoding and cascading gateways. The proposed research builds on this start and will create an event-based distributed system **Ecco** over a peer-to-peer network using a multi event broker model. Other event-based systems will be integrated within a unified interface and, in particular, we will adapt Ecco to mobile computing environments and web services. Our aims are as follows; how we will meet them is described in Section 2.2.

- To design an event-based system providing publish-subscribe functionality and implementation based on JXTA
- To design and implement algorithms for dynamic, event-broker grid formation with context-awareness and infrastructure for constrained mobile computing environments
- To design and implement service discovery over an event broker system
- To integrate other event-based systems (e.g., Java Message Service -JMS)
- To design and implement a gateway for non-programmable mobile devices and provide optimised data flow
- To provide a security framework
- To provide an engine for processing composite events for an active information framework (see the detail for2.2.3)
- To evaluate the performance of the system

2.2 Methodology and work programme

An overview of the project is given in the enclosed workplan which comprises six main phases. First, components of a JXTA-based system will be designed and integrated in **workunits 1 and 2**. Creating an event broker grid (**Unit 2.1**) is a key factor in providing reliability and scalability in an event-based distributed system. The grid consists of event brokering agents (called also *broker agent* or *agent*), and a broker agent performs routing, receiving events, and sending events. Broker agents can form a group to provide scalability at the cluster level; a group of agents can then be linked together in a flexible, fault-tolerant, efficient, high-performance fashion in a publish-subscribe model. *Dynamic* grid formation is essential, including context-awareness and an infrastructure such as hierarchy and grouping for better performance. The proposed system is for wide area networks and requires service discovery, various communication schemes, fault tolerance, and security (**workunits 3 and 4).**

The event broker approach is attractive for the integration of generic event-based systems with mobile computing and web services, because it allows heterogeneous messaging systems to be combined under a unified interface (**workunit 3).** In particular, constrained mobile devices can take advantage of resource sharing from middleware systems. Non-programmable devices will be integrated with event broker systems by means of a gateway (**unit 3.2**).

Another important goal in this research is to adapt an Event Condition Action (ECA) Rule Engine (**workunit 5**) for running components that are required to perform active rule computations within broker agents. Combining ECA-rules, complex event composition, and detection with event-based systems should provide a sound active information framework for workflow construction. We also aim at establishing a unified semantics-based API across event broker implementations including messaging across brokers.

Ecco uses JXTA as a base, which provides a general peer-to-peer platform (**workunit 2**). Ecco constructs an event broker agent system on top of JXTA and the distributed event-based interaction, which provides the abstraction to hide the mechanism of a low-level JXTA peer-to-peer library, enables preservation of the decoupled flavour of peer-to-peer applications. JXTA offers a solid peer-to-peer platform supporting routing and security, a membership service, a discovery service, a local cache, and pipes for communication endpoints for peers. Although JXTA is not a mature peer-to-peer platform, and several performance issues have been raised, it is the most promising starting point. We expect this research to contribute to the evolution of JXTA. Alternatively, Ecco could construct a native mode of event broker agent grid with Pastry. An overview of an event broker agent system is shown in Figure 1. Broker agents are linked together to form a grid of event broker systems. Connectors provide a proxy to communicate with different event-based systems such as a JMS based messaging system. Gateway connectors provide an edge point to those mobile devices that are not capable of communicating directly with the event broker agent system. Broker agents triggered by event condition action may spawn mobile agents, which perform temporal broker agent functions at the specified location. Mobile agents are autonomous code entities that can stop, move themselves to another agent-enabled host on the network, and continue execution, deciding where to go and what to do along the way.

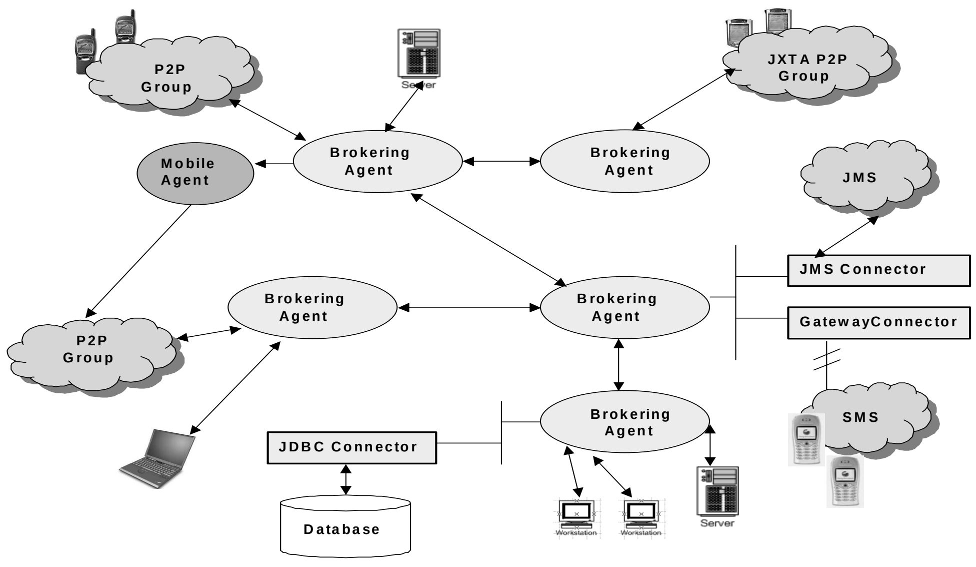

Figure 1. Overview of Ecco - an Event Broker Agent System

Ecco consists of the following two main components (**workunit 1**):

- Event Broker Agent (detailed in section 2.2.1) consists of Discovery, Reliability, Security, QoS, Communication Transport, and Publish-Subscribe services as sub-components.
- ECA Rule Engine (detailed in section 2.2.2) is embedded in Event Broker Agent.

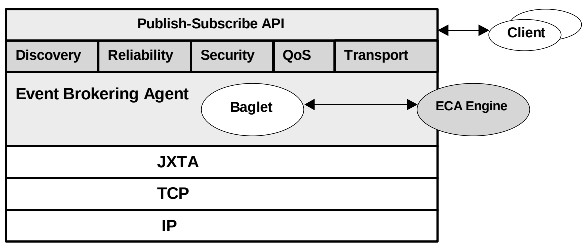

Figure 2. Ecco Components

The base of Ecco is a peer-to-peer grid comprising resources such as relatively static clients, high-end resources and a dynamic collection of multiple peer-to-peer systems. For a hybrid environment, services can be hosted on such a peer-to-peer grid with peer groups managed locally and arranged into a global system supporting core servers. Access to services can then be mediated either by the agent or alternatively by direct peer-to-peer interaction between machines. The distributed broker approach scales best globally, but is less suitable for managing the rich structure of the transient service characteristics of complex tasks.

2.2.1 The Event Broker Agent using JXTA pub/sub (workunit 2)

In the Event Broker Agent a transient component named ***baglet*** (see Figure 2) is used for the subscription and distribution of events. A baglet is responsible for receiving an event, either asynchronously or synchronously, performing any ECA rule defined for the event, and generating new events that are published back to the agent. A baglet is activated (by the agent) to handle one particular event, after which it terminates. A connector is a specific type of baglet that has slightly different run-time semantics. The connector is permanently active in the agent where it is deployed and is used to interface with JMS, JXTA groups, and the gateway to the specific network environment. Design issues on the functions of an event-broker agent are listed below:

Topology and Dynamic Grid Formation (unit 2.1): A peer-to-peer topology is fully decentralized, with no central servers or routers involved. It allows many network nodes, because connections and workload are shared among peers. The disadvantage is high latency, because messages have usually to traverse multiple peers. In addition, slow peers on the route degrade performance. The peer-to-peer cluster topology is a more recent approach. The network is divided into clusters that have knowledge of the peers. Clustering helps to lower unnecessary traffic but still lacks efficient group communication mechanisms. A heterogeneous and dynamic service for peer-to-peer networks when the grid is created or changed is especially important for the integration of a mobile computing environment.

- ***Adding infrastructure to the topology*** Adding a hierarchy facilitates route aggregation to decrease network load, which is imposed by traditional broadcast-style techniques. Another possibility would be a virtual overlay network for each topic group, or specific device groups, independently of the existing peer-to-peer network, allowing a more efficient set up of the topology. In addition, the topology could also be adjusted dynamically to respond to changes and optimise delivery. Only the bandwidth of involved peers is consumed since the virtual network includes only peers interested in a topic. Introduction of hierarchy and grouping should also provide performance benefits, since they reduce broadcasts significantly.
- ***Location and Context-awareness*** - allows peer-to-peer applications to exploit information on the underlying network context for better performance and better group organisation [18,35]. Information such as location, availability of resource, battery power, services in reach, and relative distances can be used to improve the routing structure of the peer-to-peer network, thus reducing the routing overhead.

Routing Algorithm and Subscription Language (unit 2.2): Topic-based addressing is an abstraction of numeric network addressing schemes. With content-based subscription as used in Siena [10], Gryphon [24], and Elvin [36], delivery depends only on message content, and applications can therefore select different combinations of messages without changing the addressing structure. Content-based subscription extends the capability of event notification with more expressive subscription filters compared to topic-based subscription. The most advanced and expressive form of subscription language is content-based with pattern matching; such a language is important for event notification.
Common topic-based systems arrange topics in hierarchies, but a topic cannot have several super topics. Type-based subscription, introduced in [4, 19, 20, 21], provides a natural way of doing this if the language offers multiple sub-typing, thus avoiding any explicit message classification through topics. This works well with typed languages, but it is complex to deploy this degree of serialisation of objects. Also, mobile applications may not have the concept of objects or typing. It is therefore better to provide an abstraction of an addressing scheme based on the current name-based addressing. Supporting hierarchy and wildcards in the topic allow subscribers flexibility in subscriptions. The combination of hierarchical topics and high speed content filtering provides the flexibility necessary to allow applications to evolve beyond their initial design. In Pronto [7], besides topic-based routing, a filtering function that selects messages based on their content is supported. The content filter language is based on the WHERE syntax of SQL92 over XML messages. Content-based publish/subscribe is essential for better (filtered) data flow in mobile computing. An event broker agent serves the event service to satisfy the information needs of all connected event consumers by topic and content-based routing of notifications. Embedding routing decisions at each agent, depending on the content, can be expensive but offers several advantages. The combination of topic and content-base routing is best for mobile computing.

Communication Transport (unit 2.3): Wireless data networks have more constrained communication environments than wired networks, due to mobile computing. Thus, communication has to be extremely efficient. [13,22,27]. In general, a messaging system deployed on a public network cannot depend on homogeneous router technology, and it should not rely on LAN or IP multicast technologies. Instead, universally adopted standards such as TCP/IP or HTTP should be used for all communication. However, multicast can provide an efficient way of sending data to many destinations, and recent developments such as Pragmatic General Multicast (PGM) [23,31] brought a new dimension to multicast [9]. If a suitable network communication is available, it should be used to get the best performance for data transmission. On the other hand, messaging systems require broader functionality for various QoS parameters like message priority and expiration time. These parameters directly influence the message distribution process. Messages with high priorities are forwarded earlier than concurrent messages with low priority. Handling QoS parameters has to be in the overlay network layer to be efficient. Thus, there should be a dynamic configuration function of communication [15]. Ecco will provide different communication protocols between client and agent and between agents in a well-hidden manner. JXTA offers Pipes for communication endpoints for peers, and by using the JXTA Pipes service framework several protocols can be implemented such as Application layer multicast, Reliable multicast over IP multicast, PGM, TCP/IP, SSL, HTTP (HTTPS), RMI and SOAP.

Remote processing and Mobile Agents (unit 2.4): In an event broker agent, we plan to use mobile agent technology to activate event reaction objects, so that mobile devices can take advantage of resources from the event broker agent grid to execute processes remotely and automatically obtain results from the devices. Mobile agents are autonomous, intelligent programs that travel to the target agent and apply corrective actions. Agents act as key components within an event broker agent grid, and combining ECA and mobile agent technology offers an effective computational framework for comtemporary business practices.

2.2.2 Federation (workunit 3) and large-system extensions (workunit 4)

JMS Integration (unit 3.1): A bridge function to JMS is designed as a connector. A JMS connecter is instantiated for a JMS connection, and list of sessions will be managed. Integrating JMS will transparently replace a single server or a limited server JMS systems with a large scale distributed solution which has failure resiliency, dynamic load balancing and scaling.
JXTA Integration (unit 3.1): We plan to implement a bridge function to JXTA by a proxy node sitting between Ecco and a JXTA group. JXTA interactions that would be routed by Ecco are fed through the proxy node, which also serves as a rendezvous peer. JMS clients can also interact with JXTA peers.
Gateway Integration (unit 3.2): In Pronto, we have introduced Gateway to obtain good performance in messaging over a mobile/wireless network. Gateway is a message hub to transmit information with store-and-forward messaging that provides powerful optimisation and data transformation. SmartCaching is an intelligent cache function with *Pull, Subscribe,* and *Snapshot* services. Gateway provides a disconnect operation by using a cache. Figure 3 shows the gateway connector as a bridge between Ecco and Pronto to integrate them.

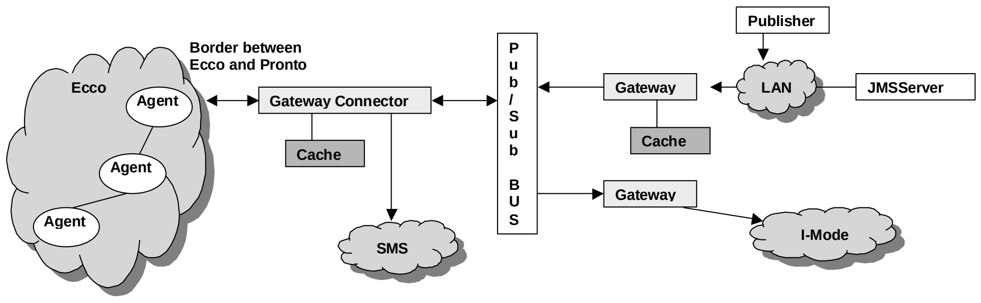

Figure 3. Gateway Connector

Web services Integration (unit 3.3): A web service is a computer program running on either a local or remote machine with a set of well-defined interfaces (ports) specified in XML (WSDL). Web services communicate using messages. Support for JMS and JXTA in Ecco will ensure that the web services run seamlessly within the same API to either classic 3-tier (client-middleware-server) or peer-to-peer environments.

Message Interoperability (unit 3.1-3.3): The above integration of different event-based systems requires message translation, because it is unlikely that there will be a single event service standard. Establishing a common message format for events and messages will be important in gaining interoperability on the grid.

Service Discovery (Naming) (workunit 4, unit 4.1): In distributed peer-to-peer networks, service discovery is complex as hosts frequently join and leave the overlay network. In mobile systems service discovery may become even more complex especially in ad-hoc or mixed systems. Current widely used peer-to-peer networks rely on central directory servers or massive message flooding, clearly not a scalable solution. In such a system, a service is an entity that can be used by a program or another service. Ecco will combine that functional concept and will create a unique service discovery framework using JXTA. In Ecco, Agent Locator takes care of service discovery, and clients do not need to keep track of Agent Locators. The Agent Locator has properties and constraints based on which it decides on the agent that a client will connect to.

Security (unit 4.2): A messaging system deployed over public networks must provide security and privacy features to a degree not mandated over private secured networks. These features must include client authentication, access controls, and encryption/integrity of messages. Because of decentralized environments with frequent disconnection, it is hard to provide robust authentication. Grid Security can be deployed on Kerberos/PKI through JXTA security services, but the following aspects require consideration:

- Message level: support encryption and cryptography for integrity of messages
- Authentication/authorization: tunnelling from clients allows public agents to be reached from within intranets.
- End-to-end encryption: allows content-based routing without excessive cost for cryptographic operations.

Reliability and Fault-tolerance (unit 4.3): The architecture of event-based systems should be tolerant to error and network fallout. It should provide an application programmer with the ability to acknowledge and identify such conditions and pursue alternative means to continue.

2.2.3 The ECA Rule Engine (workunit 5)

The new requirements for event-based systems, high availability auditing, and tracking requirements from enterprise application integration, have shown the increasing importance of databases in event-based computing, especially in large-scale application integration [32]. The need to track changing business information reflected in databases has driven the development of active database technology and continuous query management. The Semantic Web [40] offers a research perspective for automating business processes. Rule mark-up languages [34] allow modular business rules to be expressed as stand-alone units in a declarative way. Also, publishing and exchanging them between different systems and tools will play an important role for facilitating business-to-customer (B2C) and business-to-business (B2B) interactions over the web. Reaction rules allow the specification of the reactive and communicative behaviour of a system or agent (in other contexts these are called event-condition-action (ECA) rules or triggers).

In event-based distributed systems, composite events represent complex patterns of activity from distributed sources. Although composite events have been a useful modelling tool in active database research and telecommunications network monitoring, little progress has been made in using them in large-scale, general-purpose distributed systems. This new requirement will require further work on event composition; ongoing Opera group work in this area is described in [5] and the approach remains to be evaluated.

The Event Condition Action (ECA) paradigm underpins the way in which large distributed systems are constructed and we propose to create an ECA Rule Engine in Ecco. ECA triples can be combined to express rules based on event detection, the application of conditions, and the execution of actions. That is, when something occurs, and an ECA rule can determine what occurred, an appropriate action is triggered. An ECA Rule Engine is a set of interfaces and semantics that describe the components required to build ECA rules; it supports the definition, execution and deployment of ECA rules that are applied to events. Its goal is to run the components that are required to perform active rule computations expressed through the ECA rule paradigm. Scalable multi-threaded rule sessions for monitoring complex event sequences will reside in the Event Broker Agent. This is consistent with the Semantic Web approach, since the events are most likely to be exchanged among all clients, and the decision logic should be described in a unified way..

In **workunit 6** the system will be integrated and tested and evaluation will be carried out. Our methodology is to test the earlier workunits as they develop (see the **milestones** in the workplan), and this latter stage is to evaluate the integration of the already tested system components.

2.3 Beneficiaries

This research aims to create a comprehensive software platform for extending Internet-based applications to a wide range of network-edge (pervasive) devices that can enable end-to-end solutions across multiple market segments. Device and server-side software can help device manufacturers, application developers, and platform integrators to streamline the production and rollout of small-footprint applications that will generate new businesses, and differentiate their products with innovative service packages. As an example, it could be used for pervasive computing, thus providing personalised, instantaneous access to information (stock quotes, news, email, calendar-alerts, etc.), in real-time and in a reliable manner.
Integrating an Event Condition Action (ECA) engine with event broker systems is the way to construct large distributed systems where events are the common currency. Events reflect the movement and flow of information around a system. At the highest level, the business events describe the processes that define a business and how it works. At the lowest level, they reflect the internal processes within a system and how the individual parts interact with each other.

2.4 Dissemination and exploitation

The Opera group pioneered event-based systems in the mid 1990s as the paradigm for pervasive and mobile computing. We will continue to participate as organisers and programme committee members of workshops and conferences in the area. We will publish our work (Eiko has had two papers on her Diploma dissertation accepted for an ICDCS workshop and Middleware 2003). As milestones are met we will make the software available for download by other researchers.

2.5 Resources

We request support for travel and subsistence to present our work (for example at Middleware, ICDCS, ACM SIG Mobile, DEBS) and to maintain contact with other groups working in the area, for example TU Darmstadt (Rebeca) and IBM TJW (Gryphon). Opera group members spent three month periods with both of these groups in 2002. Also we request support for ten percent of a Computer Officer, to support the extensive experimental work, and equipment for implementing Ecco including trials with small mobile devices and a substantial simulation to evaluate the large-scale system.

Summary

In the enterprise domain, messaging is becoming the dominant communication mechanism, through its support for decoupled interactions. However, intranets with centralised administration are still predominant. Some attempts have been made to integrate mobility but these have been ad hoc; no-one is providing a solution that fully integrates mobile computing and traditional messaging.

In the research domain, P2P based messaging, with the publish/subscribe paradigm, has emerged as the most promising approach to communication over wide area networks. But it has not yet integrated mobility. The alternative research focus for mobile computing, Ad-Hoc networking, does not fully address wide-area, large-scale solutions.

There is therefore a need for research on how a P2P substrate can be built on to provide unified support for mobile sources and sinks for large-scale, widely distributed environments. Ecco addresses this need.

References

[1] J. Bacon, K. Moody, J.Bates, R.Hayton, C.Ma, A.McNeil, O.Seidel, and M.Spiteri, Generic Support for Distributed Applications, IEEE Computer, pp.68-77, Mar. 2000.

[2] J. Bacon, K.Moody, and W. Yao, Access control and trust in the use of widely distributed services, In Middleware 2001, Vol.2218 of Lecture Notes in Computer Science, pp.300-315, November, 2001.

[3] A. Hombrecher and A. McNeil, The Active House Project, http://www.cl.cam.ac.uk/Research/SRG/opera/projects/active-house, 1998.

[4] P. Pietzuch and J. Bacon, Hermes: A Distributed Event-Based Middleware Architecture, *Proceedings of the 1st International Workshop on Distributed Event-Based Systems (DEBS'02) with IEEE ICDCS,, July 2002.*

[5] P. Pietzuch, B Shand and J. Bacon, A Framework for Event Composition in Distributed Systems, accepted for Middleware 2003.

[6] J. Bacon, A. Hombrecher, C. Ma, K. Moody, and W. Yao, Event Federation and Storage using ODMG, *Proceedings of the 9th International Workshop on Persistent Object Systems: Design, Implementation and Use (POS9),* pp.265-281, 2000.

[7] E. Yoneki and J. Bacon, Pronto:A MobileGateway using the Publish/Subscribe Paradigm over Wireless Networks, Technical Report UCAM-CL-TR559, Computer Laboratory, University of Cambridge, also to appear in ACM/IFIP/USENIX International Middleware Conference (WiP session), June 2003.

[8] E. Yoneki and J. Bacon, Gateway: a Message Hub with Store-and-forward Messaging in Mobile Networks, to appear in *Proceedings of the 23rd International Conference on Distributed Computing Systems – Workshops on Mobile Computing Middleware (ICDCS – MCM03), May, 2003.*

[9] G. Banavar, T. Chandra *et al*, An Efficient Multicast Protocol for Content-based Publish-Subscribe Systems, *Proceedings of IEEE International Conference on Distributed Computing Systems (ICDCS),* 1999.

[10] A. Carzaniga et al., Content-Based Addressing and Routing: A General Model and its Application, Technical Report, University of Colorado, 2000.

[11] A. Carzaniga, D. Rosenblum, and L. Wolf, Design and Evaluation of a Wise-Area Event Notification Service, ACM Trans. On Computer Systems, 19(3) pp.332-383, Aug. 2001.

[12] A. Carzaniga, D. Rosenblum, and L. Wolf, Architecturing Scalability and Expressiveness in an Internet-Scale Event Notification Service, *Proceedings of the 19th ACM Symposium on Principles of Distributed Computing (PODC '00),* July, 2000.

[13] C. Chiang, M. Gerla, and S. Lee, On-Demand Multicast Routing Protocol", *Proceedings of IEEE WCNC '99,* pp.1298-1302, 1999.

[14] G. Fox *et al.*, An Event Service to Support Grid Computational Environment, Concurrency and Computation: Practice and Experience, 2000.

[15] J. Crowcroft *et al*., Channel Islands in Reflective Ocean: Large-Scale Event Distribution in Heterogeneous Networks, IEEE Communications Magazine, Vol.40 No.9, pp.112-115, September, 2002.

[16] G. Cugola and E. Nitto, Using a Publish/Subscribe Middleware to Support Mobile Computing, Middleware for Mobile Computing Workshop, 2001.

[17] G. Cugola, E. Di Nitto, and A. Fuggetta, Exploiting an event-based infrastructure to develop complex distributed systems, *Proceedings of the 19th Int. Conf. On Software Engineering (ICSE98), 1998.*

[18] R. Gold *et al.*, Use of Context-Awareness in mobile Peer-to-Peer Networks, 8th IEEE Workshop on FTDCS, 2001.

[19] P. Eugster, and R.Gueraoui, Content-Based Publish/Subscribe with Structural Reflection, *Proceedings of the 6th USENIX Conference on Object-Oriented Technologies and Systems (COOTS01), Jan. 2001.*

[20] P. Eugster, P. Felber, R. Guerraoui and J. Sventek, Type-Based Publish/Subscribe, Technical Report, EPFL, Lausanne, 2000.

[21] P. Eugster *et al.*, The Many Faces of Publish/Subscribe, *Technical Report TR-DSC-2001-04, EPFL*, 2001.

[22] S. Floyd *et al*., ``A Reliable Multicast Framework for Light-weight Sessions and Application Framing'', *ACM SIGCOMM'95.*

[23] J. Gemmell *et al*., The PGM Reliable Multicast Protocol, IEEE Network special issue on "Multicasting: An Enabling Technology", 2003.

[24] Gryphon: Publish/Subscribe over public networks, IBM Research, http://researchweb.watson.ibm.com/grypohn/Gryphon/gryphon.html.

[25] IBM MQ Series, http://www.ibm.com/software/ts/mqseries/.

[26] Java Message Service (JMS) Specification, http://java.sun.com/products/jms/.

[27] M. Kadansky, 'Tree-based Reliable Multicast Protocol', http://research.sun.com/techrep/authors/Kadansky,+Miriam.html.

[28] C. Mascolo, L. Capra, and W. Emmerich, Mobile Computing Middleware, the 2nd IFIP-TC6 Networking Conference, Networking Tutorial, 2002.

[29] K. Nagao, 'Semantic Transcoding: Making the World Wide Web More Understandable and Usable with External Annotations', IBM Research, 2000.

[30] The Narada Event Brokering System, http://grids.ucs.indiana.edu/ptliupages/projects/narada/.

[31] RFC3208, PGM Reliable Transport Protocol Specification, http://www.rfc-editor.org/rfc/rfc3208.txt.

[32] I. Rouvellou, L. DeGanaro, H. Chan *et al.*, Combining Different Business Rules Technologies: A Rationalization, *Proceedings of the OOPSLA 2000 Workshop on Best-practices in Business Rule Design and Implementation*, October, 2000.
[33] A. Rowstron and P. Druschel, Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems, *Proceedings of the 3rd Middleware conference,* 2001.
[34] RuleML, http://www.dfki.uni-kl.de/ruleml/.
[35] B. Schilit *et al.,* Context-Aware Computing Applications, *Proc. of IEEE- the Workshop on Mobile Computing*, pp.85-90, 1994.
[36] B. Segall *et al*., Elvin has left the Building: A Publish/Subscribe Notification Service with quenching, Proc. of AUUG, 1998.
[37] Softwired iBus Messaging, http://www.softwired-inc.com/.
[38] Sun Microsystems, Project JXTA, http://www.jxta.org/.
[39] TIBCO. TIB/Rendezvous White Paper, http://www.rv.tibco.com.
[40] W3C Semantic Web Activity, http://www.w3.org/2001/sw/.
[41] WebLogic, http://www.bea.com/products/index.shtml.
[42] Scribe, http://research.microsoft.com/~antr/SCRIBE/