Applying Enterprise Architectures and Technology to The Embedded Devices Domain

Ken Taylor and Doug Palmer
CSIRO Mathematical and Information Sciences Division & The Australian National University
GPO Box 664, Canberra, ACT, 2601
Australia
Ken.Taylor@csiro.au Doug.Palmer@csiro.au

Abstract

Large-scale networks of embedded devices present a challenge in terms of performance management and other quality of service issues, such as security or transaction management. Current commercial off-the-shelf software designed for use in the Internet environment has already had to face the same challenges in a general-purpose computing environment. We provide an overview of existing enterprise architectures. We then propose an architecture for the embedded device domain that adapts existing enterprise technologies by separating functionality and services into a embedded device world and general-purpose computing world that communicate via *device proxies.* We also discuss its application to an architecture that uses the GPRS mobile phone network as a network layer.

Keywords: Embedded Device, Enterprise Architecture, Middleware, Clustering, Failover, SCADA, GPRS, GSM.

1 Introduction

There is considerable interest in very small, lightweight, easily networked computing and sensing devices that should lead to ubiquitous computing where in future most appliances are networked.

Gelsinger (2002) argues there is an expectation that Moore's Law can be expanded beyond just more, faster, cheaper transistors to a much broader set of potential types of devices that can be integrated directly into silicon and bring the volume economics associated with silicon to other types of functionality, in particular, sensors, wireless, and optical components. Gelsinger is describing a project marketed as Radio Free Intel where the goal is to drive radios directly into silicon so that every processor that is built has a directly integrated radio capability. This aim is driving wireless connectivity to the point where it's essentially an incremental cost of zero to have radio or communications capabilities.

As cost decreases, it becomes an attractive proposition to completely instrument an environment. The CSIRO Smart Spaces project (2002) envisions millions of small, embedded devices permeating an environment, such as a building or vehicle, and providing large amounts of fine-grained information.

The issue then arises as to how this multitude of devices with limited processing power and huge variety in functionality can be harnessed without complexity limiting the capacity of humans to take advantage of the opportunities created. One approach is to think of this new application space as a blank slate requiring the development of software support systems, architectures and multi agent systems aimed at reducing complexity and making it easy to construct useful applications using multiple cooperating devices. Another is to look at existing solutions to similar problems in the Internet and enterprise domains, where commercial off-the-shelf (COTS) software is common, and consider ways that existing technologies can be adapted and deployed to this new application area.

The principal differences in the embedded devices domain from the internet domain are likely to be interaction with the physical world, low bandwidth communications, unreliable/intermittent communications, and low processing power due to the requirement for minimal cost of embedded devices. We identify existing technologies, architectures and approaches that can be adapted to provide solutions in the embedded devices domain. Adaptations of some existing technologies for the embedded devices domain are proposed and presented in the context of an infrastructure we have built that utilises the GSM mobile phone network for communicating with simple programmable devices. This architecture allows complex supervisory control and data acquisition (SCADA) applications to be built utilising shared network infrastructure to communicate with distributed devices that monitor and act on the world.

2 Current Enterprise Architectures and Technology

Current enterprise systems are designed to handle large workloads using strategies such as replication and clustering. The world-cup website handled 106,000,000 hits in a day (BBC, 2002). Chiu (2002) reports that the Wimbledon website handled 430,000 hits in a minute. Google (2002) routinely handles 150,000,000 requests a day.

Figure 1 shows a typical web-site configuration for e-business. Incoming requests are routed to web servers, with the router handling load balancing for the web servers. The web servers use a cluster of application servers to handle the complex (non file-serving aspects) of the web pages that they produce. Business logic, such as how to handle orders and queries, resides in the application servers. The application servers also manage resources, such as threads, caching and database connections to provide the best throughput. The application servers use a database to store persistent data. Key attributes of this architecture are:

- Replication and Failover. Critical components, such as the database and router are replicated, so that if a component fails, there is a "hot backup" waiting to take over with the same information as the failed component.
- Clustering. Components that manage many requests are clustered. Clusters group together multiple copies of a component into a single logical unit. Clients talk to the cluster and the cluster balances load between the clustered components. If a component fails, the cluster redistributes work to other components in the cluster giving failover.

Testing by Brebner et. al. (2002) of application servers shows that a cluster of two application servers (4x700 MHz Xeon CPUs) and a database server (8x700 MHz Xeon CPUs) shows a throughput of 3500 transactions per second. Such hardware is readily available – and already somewhat out of date – making a million sensors producing a reading every five minutes achievable, using existing systems on commodity hardware.

Figure 2 shows an alternative architecture to the application server architecture shown in Figure 1. In this architecture, messages are sent to a cluster of message oriented middleware (MoM) servers. The MoM cluster then publishes the messages to interested applications – one of which may save it to a database. Clustering in MoM serves a similar purpose to clustering in application servers: load balancing and failover.

Testing by Greenfield et. al. (2001) of (unclustered) MoM servers shows that throughputs of 5000 messages per second are possible on a single 700 MHz CPU system.

2.1 Quality of Service Attributes

Enterprise systems are largely designed to ensure adequate *quality of service* (QoS) in the face of unreliable and loaded components. In SCADA systems, QoS attributes will often need to be built out of collections of unreliable, intermittent embedded devices and their control systems. Typical QoS attributes available in enterprise systems. and their application to SCADA systems. are:

- Guaranteed Delivery. Data from sensors needs to be delivered, even in the face of a temporary

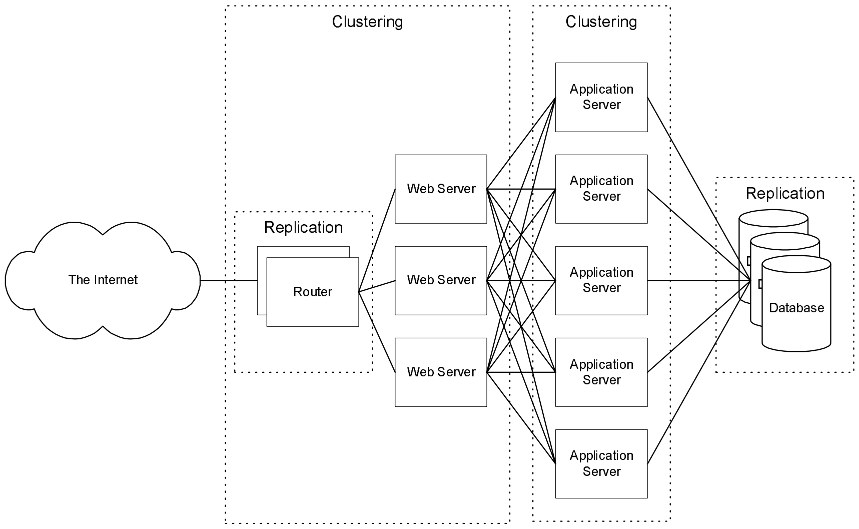

Figure 1 Example Application Server ArchRelational dat

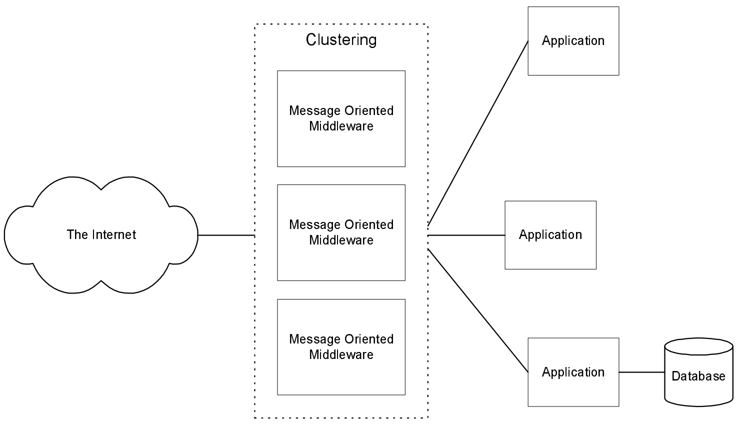
Clustering
Message Oriented Middleware
Message Oriented Middleware
Message Oriented Middleware
The Internet
Application
Application
Application
Database

Figure 2 Example Message-Oriented Middleware Architecture

component or network failure.

- Availability. Servers for sensor data are always available to receive and process data.
- Transactions. Groups of sensors and actuators need to achieve a consistent state, or roll-back to the initial state if a component fails.
- Performance. Latency times for sensor readings are bounded. Sufficient throughput to handle the volume of data flowing through the central host is provided. In overloaded conditions, messages or requests are queued and managed efficiently.
- Security. Authentication, permission policy and data security is maintained in an environment where mobile sensors can enter and leave the SCADA space.

Not all of these attributes may be required in a specific SCADA system and all of these attributes can be built into a SCADA system during design and programming. There are also some QoS attrributes, such as the integration of poor quality data, which are not encountered in the enterprise domain and are not part of the normal QoS attributes provided by such systems. However, enterprise COTS systems tend to provide a number of QoS attributes as configurable options, easing the software development load.

Naturally, supplying improved QoS exacts a price. In an extreme case, in one of the MoM servers tested by Greenfield et al. (2001), changing from non-guaranteed to guaranteed delivery reduced throughput from 5700 to 160 messages per second. However, the same price can be expected in any system and COTS software is usually carefully tuned to extract maximal performance.

3 Applying Current Technologies to the Embedded Devices Scenario

How can these existing technologies be applied to support large numbers of embedded devices?

Consider the architectural model shown in Figure 3. The system is divided into worlds: the *embedded system world* (ESW) and the *general purpose system world* (GPSW). The general purpose system world is for software that runs on general purpose computers (such as an office PC or better) and has access to low cost medium to high speed permanent network connections (128K and up). The embedded system world is for embedded devices that cannot operate directly in the internet world due to resource limitations:– small memory spaces (8K – 128K of code memory, 0.5K – 32K of data memory) and low to medium speed, high cost networks.

Gabel and Litz (2001) have also recognised the desirability of separating the ESW from the GPSW and have suggested an architecture where devices contain an embedded web server that communicates only with a main server. All device interactions are via the main server. Their proposed architecture allows device polling but doesn't provide for event driven communications from devices to applications. Similarly Bergstrom et. al. (2001) describe an architecture aimed at home automation and which also aims to achieve a layering between the ESW and the GPSW. They describe a three tier architecture with sensors and actuators communicating with a controller they call the home gateway which in turn communicates with a hosted service they call the global home server. The home server manages up to 100,000 home gateways and allows home owners configure their home gateway and monitor any

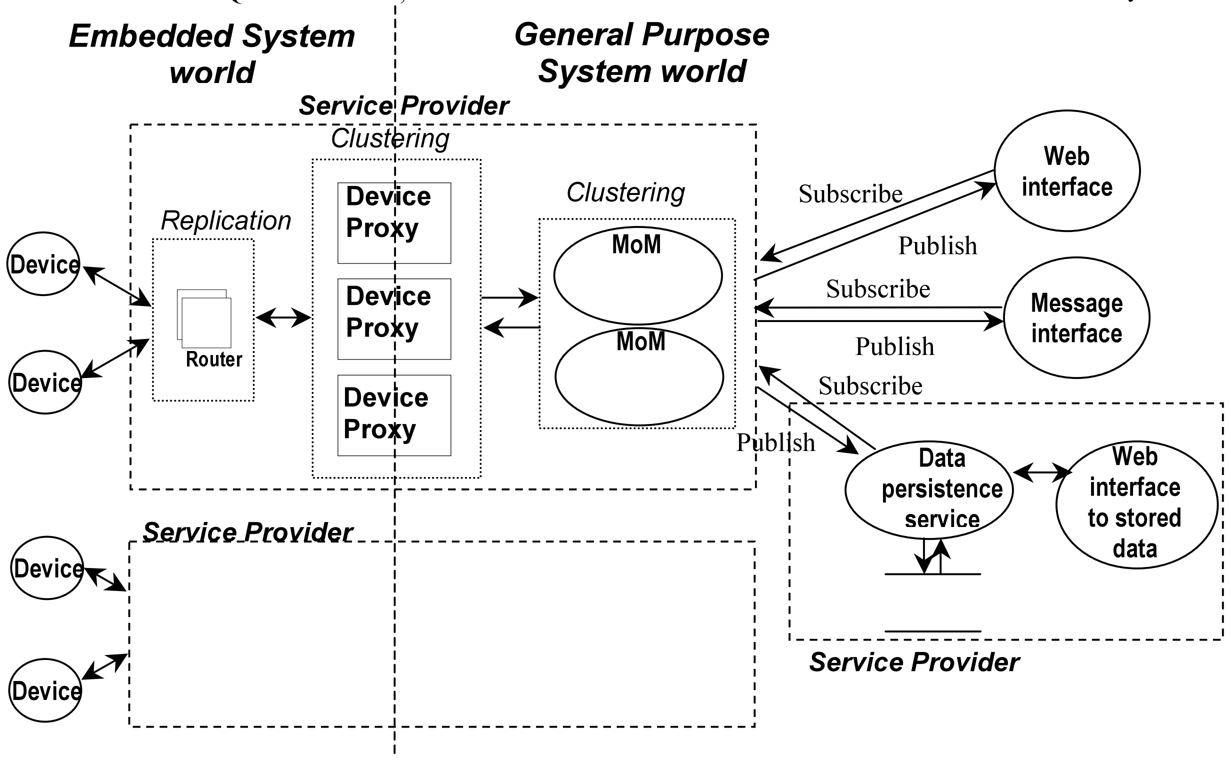

Figure 3 Example Device Proxy Architecture

activity in the home that is reported by the home gateway to the hosted service. This architecture is specifically aimed at home automation applications. The home gateway communicates to sensors and switches using residential network protocol or other non-IP protocols. If the IP protocol was used to communicate directly with sensors and switches, the home gateway could be removed. However, the home gateway also hosts control logic, which if it were located on the global home server would cease to be available when, communications from the home were not functioning.

The *device proxies* provide the link between the ESW and the GPSW. The purpose of the device proxy is to offload functionality from the device to the proxy so that the normal GPSW authentication, encryption, multithreading, compiling, application management etc can be done in an environment with the processing power to support it. This enables devices to have an absolutely minimal functionality. The device must interface to the physical environment to sense or act on it as well as provide sufficient computing power to provide local control loops and to summarise sensor data so that only summarised data is transferred across the communications network. For example a temperature sensor might sample the temperature 10 times a second but only communicate the temperature when it had changed by more than one degree since previously advised or a device might be commanded to move an actuator to a set point but the actual position control is provided by a programmable device control loop with a frequency of kilohertz. Low cost micro controllers operating in sleep mode most of the time can provide this level of functionality. Where a device is sufficiently powerful to provide the functionality otherwise provided by a device proxy it can operate without a proxy and is represented in the same way as the device proxy.

In the architecture proposed a proxy can connect to a device using a range of different device protocols running on top of IP protocol through loadable device interfaces. Similarly, a proxy supports a range of general-purpose protocols through loadable interfaces. The diagram shows a proxy providing a messaging interface to a message orientated middleware system. The MoM provides a publish/subscribe interface. Applications that wish to receive data subscribe to a particular channel. Data is made available by publishing it to that channel. As shown on the diagram multiple applications can subscribe concurrently to a single data message channel.

Three possible applications are shown. At the bottom is a data persistence service. This service subscribes to selected data channels and records published data in a relational database. It has a web protocol based interface that allows humans or programs to query the persistence service.

In the middle is an application that is using the live data from the devices. It subscribes to the desired channel and processes the data as it is published.

At the top is a web interface to the live data. This requires a script to be running on the client to accept the published data and render it.

Similar approaches to those currently used for enterprise information systems can be used to scale this architecture to enable a single service provider to support hundreds of thousands of devices. Replicated routers, such as the Cisco 12000 series routers, handle the interface to devices. The router function provides a high reliability single point of interface for the devices. The required messaging throughput is obtained from a cluster of device proxies running on a multiple hardware system. Due to the router, the devices appear to see a single device proxy. If a particular device proxy hardware platform fails, the router function and device proxy cluster re-route activity to other device proxies. Typical MoM systems have a limited throughput capacity (of the order of 1000s of messages per second). This rate can be scaled, by using multiple processors for the MoM and by clustering MoM servers. Sophisticated MoM systems, such as TIB/Rendezvous or IBM WebSphereMQ, provide this facility out of the box.

A service provider may choose to offer the data persistence service as part of the service. Again this can be scaled using software technology already deployed in other areas.

Many of the software technology/infrastructure issues on the GPSW part of the architecture relate to applying existing technologies in a slightly different manner. The major design task in the ESW is to determine if and how concepts from the GPSW can be applied, given the resource constraints. It would be highly desirable if the software running on the devices was easily able to take advantage of services such as:

- Message passing facilities: automatic selection/reselection of proxies, store and forward of message during communication outages, transactions covering sensing of data from multiple sensors and/or requests requiring multiple actions.
- Means to address authentication of devices, authorisation for access to device services (such as re-programming) and confidentiality of message data.

At the application level, methodologies for the construction of distributed applications running across many levels of scale, power and hardware need to be developed. In the enterprise application area, applications are often deployed into containers that supply system management and QoS attributes. Ideally, an application written for the model shown in Figure 3 would be written to be independent of the exact configuration in the same way that a program written in a high-level language is hardware and platform independent. The system itself would be responsible for partitioning and distributing a deployed application across the ESW and the GPSW. The deployed application would also be instrumented (to prevent hostile or buggy code damaging the integrity of the system) and augmented with code aspects that provide QoS. Multiple applications may need to be run simultaneously, with the system managing the packaging of multiple software components into a single device. The development of such a level of sophistication is an area of ongoing research.

protocol we have developed running over a permanently maintained TCP/IP connection.

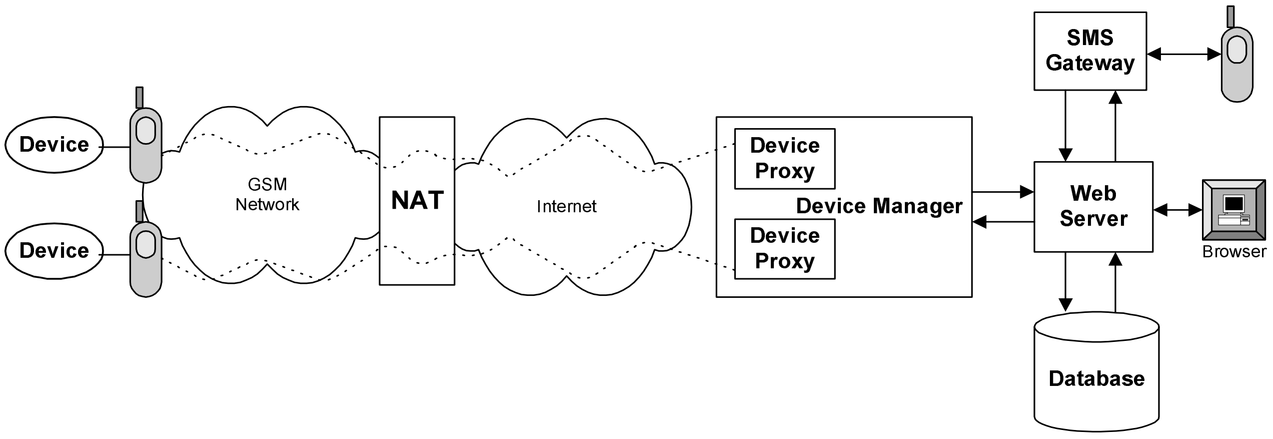

Figure 4 GPRS-Based Architecture

3.1 An Implementation Using the Mobile Phone Network.

Wireless connectivity at zero incremental cost as discussed in the introduction is not yet available. Probably the closest service currently available is the GPRS packet switched data service provided by the GSM mobile phone network and often referred to as a 2.5G mobile phone network. It is a precursor to the higher speed packet switched services that will become available over the next few years and which are often referred to as 3G mobile phone networks. Ultimately the 3G network is likely to be extended to ad hoc sub networks to achieve the low power RF communications required for long life battery powered embedded devices.

We have implemented a subset of the architecture described above, shown in Figure 4, which is available at http://mobile.act.cmis.csiro.au. Ethernet networks and the GPRS network are used to communicate between device proxies and low cost micro controllers that interact with the physical world. The elements in Figure 3 currently implemented are:

- The devices with differing versions connected by ethernet and GPRS. The GPRS connected devices are described by Mayer and Taylor (2002) and are reprogrammable over the air.
- An instance of the device proxy based on a PC and able to support large numbers of devices also described by Mayer and Taylor (2002).
- A data persistence service based around a relational database accessible via HTTP and providing data in XML format.
- Additional services, for example an SMS message sending facility and an incoming phone call monitoring service.

HTML and XML over HTTP interfaces are provided to interface from the GPSW to the device proxy and to configure and interact with networked support services. Communication from the device proxy to devices is via a

A device connected to the Internet via GPRS is assigned a private IP address that is not 'visible' to other nodes on the Internet. This is not a requirement of GPRS but is the way telecommunication providers generally implement GPRS due in part to the number of mobile phone handsets exceeding addresses available in the IPv4 address space. The invisibility of devices from the internet and the excellent inbuilt security provided by the GSM network in the RF portion of the network provides a high level of security between a device and its proxy without adding additional layers of security being required. When a device initiates a connection to a public node, traffic passes through a network address translator (NAT). The remote node sees the NAT, not the originating node as the source of the connection. The gateway acts as a conduit between the public Internet, and all of the devices that have private IP addresses. Thus the device is required to make the connection outwards, rather than waiting for a proxy to contact it. Once the connection is established, data can be sent in both directions (since TCP channels are full duplex).

A sufficiently large proportion of the architecture has been implemented to make it possible to construct useful applications and various applications have been constructed with this infrastructure. One of these applications, constructed to demonstrate a large proportion of the available functionality, is control of a twin water tanks model. The model, described by Tan and Taylor (2002) has valves controlling water flow into one of two linked tanks. The valves are controlled through a web browser interface (shown in Figure 5), available at http://mobile.act.cmis.csiro.au/sms2/tanks/SCADA-Control-Compact.asp. As an alternative, the valves can be controlled by dialling numbers on a mobile phone and receiving status information via SMS messages.

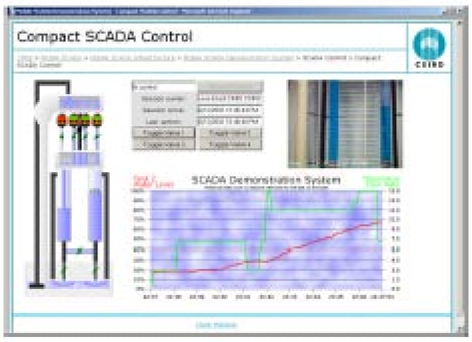

Figure 5 Browser interface for twin tanks model

4 Conclusions and Future Work

Separating large scale networks of embedded devices into an embedded systems world and general purpose systems world allows the use of enterprise COTS software in the form of application servers, message-oriented middleware and web servers. Enterprise COTS software provides high levels of performance and useful quality of service attributes, easing the development of networks of embedded systems and, potentially, managing millions of embedded devices.

An example implementation provides a prototype of this approach. The implementation uses the GSM network to connect to programmable devices and uses device proxies linked to COTS web servers and databases. While promising, this implementation does not yet make full use of the possible COTS software available and development is continuing. The architecture described in Section 3 is currently being implemented and interfaces to various middleware technologies are being developed. Once completed, the architecture can be tested against the sort of loads implicit in the large numbers of devices that the Smart Spaces project envisages.

There are still a number of yet undeveloped areas. Enterprise COTS software provides a number of quality of service attributes, but there are a number of attributes inherently part of SCADA-like applications that are not part of enterprise middleware. The middleware approach of configurable quality of service is an attractive approach to the design and management of such software. Provision of additional quality of service attributes by building on existing middleware services is needed. At a broader level, using the power of general-purpose computing systems to manage and deploy applications in the embedded systems world, so that applications can be transparently deployed, represents a considerable challenge. However, developing such a methodology is necessary if the Smart Spaces vision of millions of embedded devices is to come to fruition.

5 References

BBC NEWS (2002): *World Cup website breaks record*, 10 June, 2002. Available at http://news.bbc.co.uk/1/hi/sci/tech/2036238.stm

BERGSTROM, PETER., DRISCOLL, KEVIN. and KIMBALL, JOHN. (2001) Making Home Automation Communications Secure. *IEEE Computer Magazine October 2001* pp 50-56 IEEE

BREBNER, PAUL, CHEN SHIPING, GORTON IAN, *et al* (2002): *Evaluating J2EE Application Servers*, version 2.1, CSIRO Software Architectures and Component Technologies Group, June 2002.

CHIU, WILLY (2001): *Design for Scalability – An Update*, September 2001. Available at http://www7b.boulder.ibm.com/wsdd/library/techarticles/hvws/scalability.html

CSIRO (2002): *Smart Spaces* November, 2002. Available at http://www.cmis.csiro.au/smartspaces/

GABEL, OLIVER. and LITZ, LOTHAR. (2001): Automatic Control Markup Language – An Opportunity for Standardized Remote Control and Remote Maintenance? *The 5th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI 2001,) July 22-25, 2001, Orlando, Florida USA*

GELSINGER, PATRICK (2002) Keynote Speech *Infoworld CTO Forum 2002* San Francisco, Calif. April 9, 2002 Available at http://www.intel.com/pressroom/archive/speeches/gelsinger20020409.htm

GOOGLE (2002): *Google at a Glance*. Available at http://www.google.com/corporate/facts.html

GREENFIELD, PAUL, TRAM, PHONG, TRAN, HUU, COLTON, JOHN and GORTON, IAN (2001) *Performance Evaluation of Message Oriented Middleware Technology*, version 1.01, CSIRO Software Architectures and Component Technologies Group, September 2001

MAYER, KEVIN B and TAYLOR, KEN (2002) An Embedded Device Utilising GPRS for Communications. *International Conference On Information Technology & Applications (ICITA 2002),* Bathurst, Australia 25-28 November 2002

TAN, LUKAS and TAYLOR, KEN (2002) Mobile SCADA with Thin Clients – a Web Demonstration. *International Conference On Information Technology & Applications (ICITA 2002),* Bathurst, Australia 25-28 November 2002