Brett Ward Limited
Software and systems engineering

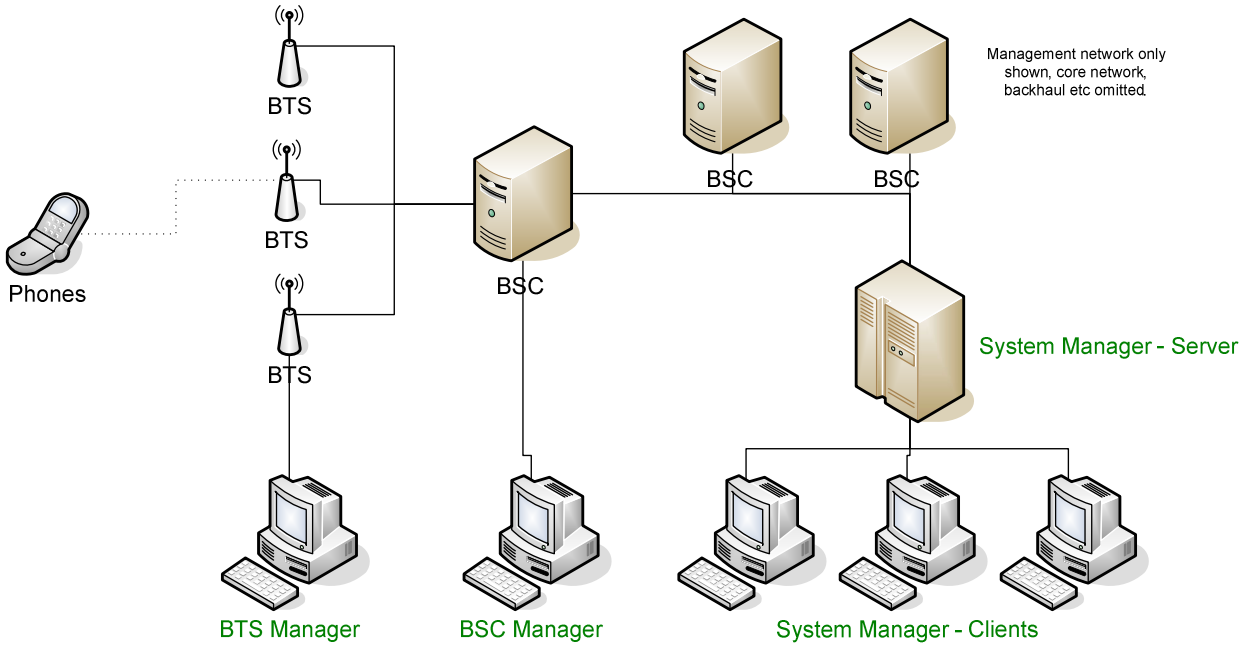
MOBILE PHONE SYSTEMS

Tim Ward, Brett Ward Limited, 11/4/2012

This document gives an outline of Tim Ward's work on mobile phone systems 2002 – 2012.

Details of some work for the security industry are omitted.

2G MANAGEMENT SYSTEMS



The diagram shows a much simplified 2G mobile phone network, showing the base station subsystem only with its associated management systems and omitting the core network (links to the telephone network, the internet, the SMS system and so on).

Components to which I have made a significant contribution are listed below.

(Note: BTS – base transceiver station, what most people think of as a "mobile phone mast" and its associated electronics; BSC – base station controller.)

BTS Manager

Description

The BTS Manager connects to BTSs and can manage one or more BTSs at a time. It allows configuration of the BTS to the point where it understands how to get an IP address and how to talk to a BTS. This involves the management of some dozens of configuration parameters, and a software download function, and a network listen

	function whereby the BTS listens to nearby cells and makes various measurements on the signals received.
Platform	Windows (or any Java host if the crypto functions are not needed)
Languages and technologies	This application is mostly written in Java, with certificate, dongle and SSL handling in C++. The user interface uses Swing.
Communications	Standard GSM protocols to talk to the BTS, with proprietary extensions
Database	Flat file, .ini style
Role	Full responsibility for design, code, test, document and maintain the application over several years.

BSC Manager

Description	The BSC Manager connects to a single BSC and manages that BSC and its BTSs (once the BTS have been configured, using BTS Manager, to the point where they can connect to their BSC). It allows creating and deleting managed objects hosted in the BSC and BTS and viewing and setting their attributes, instructing the objects to perform various actions, and displaying a list of alarms and other events as they occur. The BSC Manager contains a number of Wizards for ease of constructing and modifying complex configurations involving hundreds of managed objects and attributes. It has a full context sensitive Help system much of which is generated by mapping together the XML MIB for the system and the Windows templates for the wizard panes.
Platform	Windows
Languages and technologies	This application is written in C++, with the GUI using MFC. The help system involved coding in HTML, CSS and JavaScript. Converting the MIB from XML to C++ involved coding in XSLT.
Communications	A proprietary telecoms management protocol is used between the BSC Manager and the BSC.
Database	Early versions used a direct connection to an Access database, later versions keep no local data.
Role	Full responsibility for design, code, test, document and maintain the application over several years.

BSC

Description	The BSC is a large complex system with a number of functions. My only contribution is the data design and database interface layer.
Platform	Linux
Languages and technologies	C++
Communications	N/A
Database	MySQL
Role	Evaluated various database systems and recommended MySQL for this product. Data design for managed object and attribute storage. Code and test the data access layer which interfaces between the main BSC application, which works in terms of objects and attributes, and the database, which works in terms of tables and rows, in C++ and SQL. This was a non-trivial exercise, as some attributes are quite complex, for example consisting of an array of structures of individual values.

System Manager

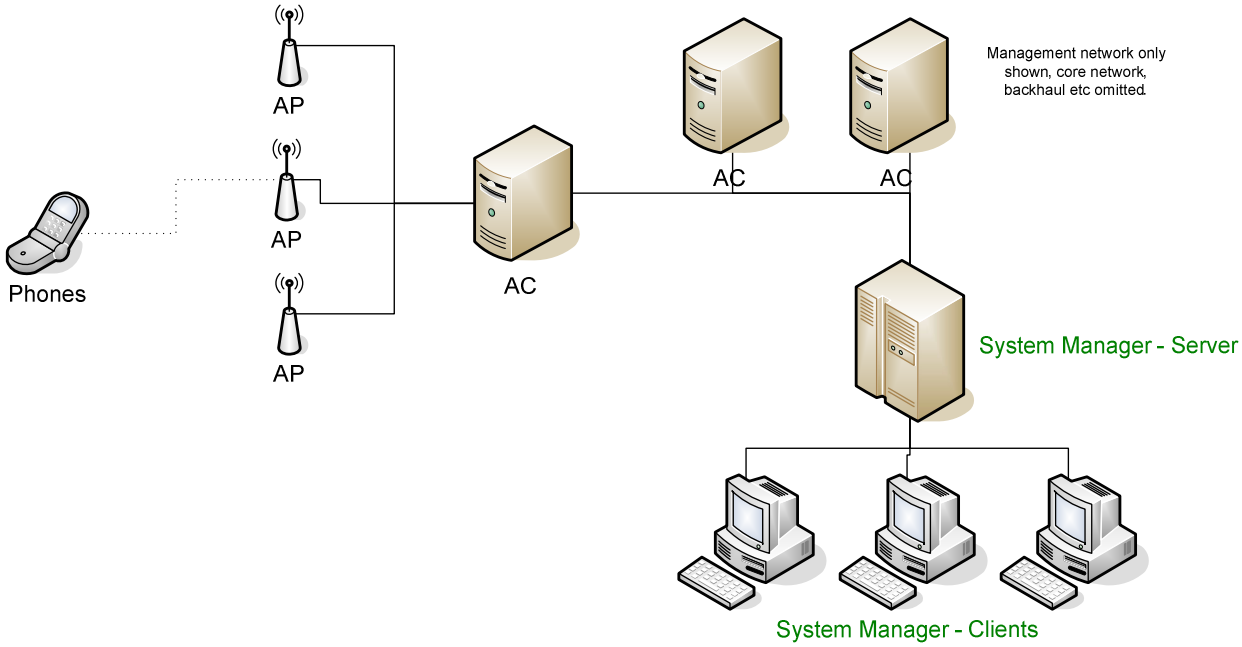
Description	The System Manager manages an entire system consisting of several BSCs. In addition to performing the same management functions as the BSC Manager, but applied to an entire system not just a single BSC, it manages alarms from the various pieces of equipment and processes performance data. Whereas the BTS Manager and BSC Manager are relatively simple applications which can easily be carried around and run on a laptop, the System Manager is a large complex application designed for permanent installation in a NOC (Network Operations Centre). It has a server component which can support several GUI clients.
Platform	Server – Linux. Client – any Java platform, typically Windows.
Languages and technologies	All code is in Java, with a variety of support data in XML, property files, and so on. The server is an EJB (Enterprise JavaBeans) application running in JBoss. The client is a Swing GUI application. Converting the MIB from XML to Java involved coding in XSLT.
Communications	EJB protocols are used between client and server. CORBA is used between the server and the BSCs.

Database	MySQL used primarily as the persistence mechanism for EJB, but with quite a lot of additional data manipulation code as just using JBoss out of the box provided neither the performance nor the flexibility needed.
Role	Over several years I worked, as one of a team, on pretty well all parts of this product, including • low level database access and interaction • bug fixing and performance improvements to low level protocol stack • server logic • client logic • GUI design and implementation. As well as designing and developing new features, which involved negotiating with the Systems team and customer facing functions and writing various specification documents, a lot of the software engineering work was refactoring earlier design in order to improve robustness and maintainability of the code and to improve performance.

System Manager – Web Client

Description	The project involved providing a web client interface to the System Manager Server, using a browser in the client and a web server and web application bolted on to the server. The client was a single page web application with all interactions taking place via Ajax. (This project was cancelled for non-technical reasons before shipping but proof of concept had been established and much of the code written.)
Platform	Server – Linux. Client – web browser.
Languages and technologies	The additional server side layer consisted of servlets written in Java and hosted in Tomcat, serving both data and JavaScript code on demand to the client via Ajax, with Push Ajax used for alarms and notifications etc. Minimal code and data was initially downloaded to the browser for an instant startup, with other components (eg code for dialog boxes) fetched via Ajax in response to user actions. The client code was HTML, CSS and JavaScript, with JSON and XML being used as the data transfer languages.
Communications	Standard HTTP between client and server
Database	N/A
Role	I led this project and designed the system and wrote much of the code, in all the languages used, with the assistance of a small team.

3G MANAGEMENT SYSTEM – earlier versions



This is a similarly simplified diagram for the 3G management system. The AP (access point) replaces the BTS, and the AC (access controller) replaces the BSC. This system had no craft terminal equivalents of the BTS Manager or BSC Manager.

Earlier versions of the System Manager were an evolution of the same code base as used for the 2G System Manager and my role on this project was similar.

Description	The System Manager manages an entire system consisting of several ACs. Its functions are similar to those of the 2G System Manager. The System Manager is a large complex application designed for permanent installation in a NOC (Network Operations Centre). It has a server component which can support several GUI clients.
Platform	Server – Linux. Client – any Java platform, typically Windows.
Languages and technologies	All code is in Java, with a variety of support data in XML, property files, and so on. The server is an EJB (Enterprise JavaBeans) application running in JBoss. The client is a Swing GUI application.
Communications	EJB protocols are used between client and server. Proprietary telecoms

management protocols are used between the server and the ACs.

Database

MySQL used primarily as the persistence mechanism for EJB, but with quite a lot of additional data manipulation code as just using JBoss out of the box provided neither the performance nor the flexibility needed.

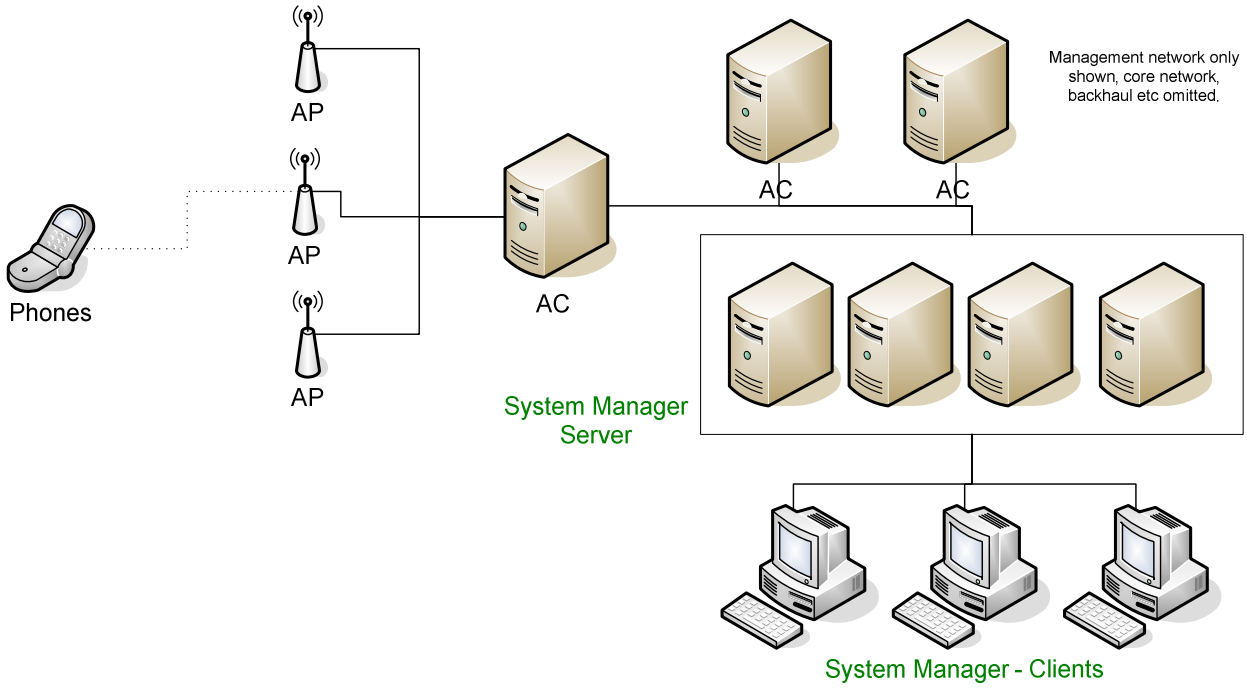
Role

Over several years I worked, as one of a team, on pretty well all parts of this product, including

- low level database access and interaction
- bug fixing and performance improvements to low level protocol stack
- server logic
- client logic
- GUI design and implementation.

As well as designing and developing new features, which involved negotiating with the Systems team and customer facing functions and writing various specification documents, a lot of the software engineering work was refactoring earlier design in order to improve robustness and maintainability of the code and to improve performance.

3G MANAGEMENT SYSTEM – later versions



Later versions of the 3G System Manager have been reworked using different technologies with the server component now consisting of an expandable number of boxes, for scalability, in a high availability configuration, with functions able to fail over from one box to another.

I contributed significantly, as one of a team, to the specification of this new implementation, the architecture, and the design and implementation, in particular of how the various parts of the server running on different boxes interacted and communicated with each other and with the supporting High Availability platform.

Description

The System Manager manages an entire system consisting of several ACs.

Its functions are similar to those of the earlier versions of the 3G System Manager with many additional features.

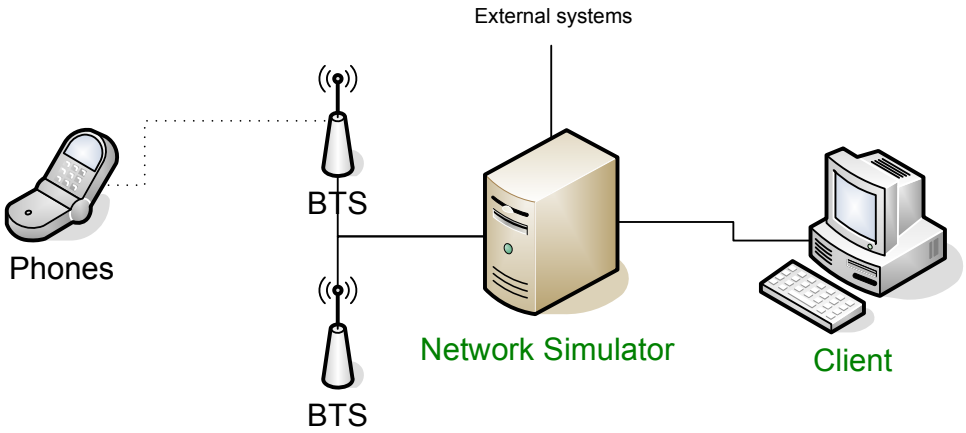
The System Manager is a large complex application designed for permanent installation in a NOC (Network Operations Centre). It has a server component which can support several GUI clients.

The server component can be spread across several physical servers, both for load sharing and scalability and for high availability, as functions can fail over from one

box to another as necessary.

Platform	Server – Linux, with Pacemaker for high availability. Client – any Java platform, typically Windows.
Languages and technologies	All code is in Java, with a variety of support data in XML, property files, and so on. The server consists of Spring and OSGi components running in Glassfish, with Hibernate for persistence. The client is a Swing GUI application.
Communications	Java protocols are used between client and server (RMI, JMS). Proprietary telecoms management protocols are used between the server and the ACs. Various standard telecoms protocols are used between the server and other components of the operator's management setup.
Database	MySQL used as the persistence mechanism for Hibernate.
Role	Major contributions to the design of the architecture for the new system, including • how functionality is to be distributed between servers • communication systems between servers • routing of messaging around the system • design, specification and coding (using Enterprise Architect and XML) of the object modelling for the internals of the server • interaction of the components of the server with Pacemaker, the high availability platform. Detailed design and documentation of large parts of the central infrastructure for the configuration management parts of the system, coded by others. Coding and testing of parts of the system infrastructure, including a component framework, layered on top of OSGi and Pacemaker, to remove the need for boiler plate code in the various application level services.

NETWORK SIMULATOR



The Network Simulator combines emulations of the functions of the BSC and the core network, allowing for example, calls to be made between mobiles. A client application monitors and controls the simulator, and an interface can be used for external systems to interact with the simulator.

The uses of various versions of the simulator include

- testing BTS functions
- demonstrations for marketing purposes
- security applications.

Description	See above
Platform	Windows
Languages and technologies	C++. The client GUI uses MFC.
Communications	Standard GSM protocols (GSM 04.08 and many others) are used between the BTS and the simulator. A proprietary protocol is used between the client and the server, and between the simulator and external systems.
Database	Microsoft Access is used to persist configuration and logging data.
Role	I took over responsibility for developing and maintaining an early version of the simulator, and developed many new features over several years. This included adding several protocol families to those handled by the simulator, including adding a GPRS function which can supply internet service to the phones via various mechanisms. As well as writing emulations for SGSN and GGSN, the GPRS work involved IP

packet rewriting and complex internal routing and interactions with Windows' IP configuration and routing facilities.