Software Projects

Due to the commercial-in-confidence and sensitive classification of the software, no source code is available for public viewing.

# Air Defence Training Software (2022)

## Objective

The system allows the trainees to practise using the air defence system by creating immersive battle scenarios in various terrains and weather conditions.

## Role

Work across functional teams to support and enhance the system.

## Outcome

Fixed all reported bugs via code refactoring. Released builds successfully in accordance with the contract.

## Technology

Support and enhance the system using C++ (main), C#, JavaScript, PHP, and Unix shell scripts.

# New ERP Deployment (2010)

## Objective

Replacing the monolithic and antiquated legacy ERP system with more modern in-house developed ERP system. The new ERP system utilise Open-Source frameworks and sub-systems, including Linux and PostgreSQL.

## Role

Provide operation and system expertise to the development team.

## Outcome

Developed the deployment strategy to run the new ERP graphical user interface (GUI) on Windows. Thus, saving implementation cost by using existing hardware.

## Technology

Using Windows batch script to deployed X servers and GUI startup script on all the Windows workstations. The bash startup script authenticates the user and startup the GUI using Remote X.

# ERP Systems Integrator (2010)

## Objective

Replacing the monolithic and antiquated legacy ERP system with more modern in-house developed ERP system. The new ERP system utilise Open-Source frameworks and systems, including Linux and PostgreSQL.

## Role

Support the development team in synchronising data between the legacy ERP and new ERP systems.

## Outcome

The ERP Systems Integrator synchronised live data between the two ERP systems. So, different parts of the business can be “business as usual”. This allowed the new ERP system can be continuous developed and deployed.

## Technology

The Integrator was developed on Windows and ran on Linux. The Integrator uses Pentaho Data Integration to develop ETL data pipelines between Ingres DBMS and PostgreSQL DBMS.

# New Proof of Delivery Document Imaging System (2009)

## Objective

The new Document Imaging System replaced the legacy imaging system, which ran on Windows 95 and used Powerbuilder. The new system can be deployed on the Allied Express network or remote sites.

## Role

Develop the system using Open-Source frameworks and sub-systems.

## Outcome

The system automatically scans the documents, allowing the operator to manually index documents without barcodes. Compared to the old system, this reduced processing time by more than 70%.

The imaged documents get sent to the SFTP server when an internet connection becomes available.

## Technology

The new Document Imaging System ran on Ubuntu Linux. It was written in C++ and can utilise any locally connected image scanners.

# Live Vehicle Tracking (2008) – The First in Australia

## Objective

Provide real-time visibility of vehicles and delivering parcels on map.

## Role

Design and implement the system using Open-Source frameworks and sub-systems.

## Outcome

The Live Vehicle Tracking system allows operators to track the travelled path and current position on the map for all the vehicles in the fleet in all states. The system also allows customers to view the position of their parcels while in transit.

## Technology

The system used PHP for server-side rendering and JavaScript for frontend rendering. It integrates with OpenStreetMap.org to display the map and uses Pentaho Data Integration to create an ETL data pipeline for the GPS information from Microsoft SQL Server.

# Scan To Revenue System (2008)

## Objective

Recover lost revenue of parcels in the distribution network.

## Role

Develop the system using Open-Source framework and sub-system to identify and recover the lost revenue.

## Outcome

The system recovered $2 million annually in lost revenue due to human and customer system errors.

## Technology

The Scan-To-Revenue System cleans the parcel tracking and proof of delivery data to automatically capture lost revenue for the overnight freight business unit. The system utilises Pentaho Data Integration to create an ETL data pipeline that cleans the source data, searches for customers, and allocates charges to them.

# Automatic Service and Network Monitor System (2007)

## Objective

Provide real-time systems and network monitor and alert supporting technical staff.

## Role

Implement the system using Open-Source framework and sub-systems.

## Outcome

Reduced labour required to continuous monitor of systems and network.

Improved problem resolution time.

## Technology

I deployed Nagios, an Open-Source monitor system, to more than 60 network nodes on the company's network.

The system ran on Linux. Instead of relying on third-party services, it utilised a simple 2G mobile phone locally connected to send SMS alerts.

I developed many bespoke scripts and plugins, using Unix shell scripts, C, and C#, to provide stats for the in-house-developed services.

# Extend & Migrate the Automatic Delivery Booking System (2003)

## Objective

Extend the functionality of the Courier Delivery Booking System to include freight booking as well. Migrate the system from AIX Unix to Solaris.

## Role

I ported the Electronic Freight Delivery Booking System to Solaris. The migration moved all the data structures from the in-house developed library to Glibc.

## Outcome

The extended function generated addition revenue with multi-million contracts.

## Technology

Used Glibc to replaced purpose-built data structure algorithm, e.g., binary tree, linked list, etc.

# Collins Class Submarine Combat System Training Software (2002)

## Objective

Provide onshore training facility for the Collins Class Combat System.

## Role

Design and implement the training system on Windows operation system.

## Outcome

The training system completed within project schedule.

## Technology

The training software used C++ and Microsoft Foundation Class on Windows 95 and 98.

# Hard Disk Cloning Software (2001)

## Objective

Provide an easy-to-use hard disk cloning over the network for the Collins Class submarine combat system.

## Outcome

The software ran within the familiar combat system environment. So, the operator could clone the hard disks while in operation such that reduce the downtime of the combat system.

## Technology

Utilise existing functionality of the Collins Class Combat System, I developed the software in Ada.

# Parcel Tracking System: Australia's first real-time parcel tracking system (1999).

## Objective

Provide tracking of parcels within the distribution network.

## Role

Design and develop the system within the constraint of the technology used by the company.

## Outcome

Captured the parcels transition between different parts of the network. Allowing clear delivery progress.

## Technology

The Parcel Tracking System consists of multiple deployed components:

**Docking Station** – A Windows-based desktop application that allows barcode scanners, Symbol PDT-3000, to dock and download scanned records via serial communication. The application enables the site supervisor to search for tracked parcels. The application transfers the data via FTP to the central repository for processing.

The application ran on Windows 95/Windows 98, using Powerbuilder.

**Scan Processor** – A Unix daemon retrieves the scan data from the FTP server, cleans it, and matches it with the master record.

The processor ran on AIX Unix. I developed the processor in embedded SQL C.

# Automatic Courier Delivery Booking System: Australia's first courier booking via EDI (1997)

## Objective

Provide automatic courier delivery booking using EDI data from customers’ systems.

## Role

Design and develop the system within the constraint of the technology used by the company.

## Outcome

Won multimillion dollar contracts with customers like Ingram Micro, Foxtel and Telstra.

## Technology

The system retrieves EDI data from the customer's system on the FTP server and books the deliveries. The process includes charging, dispatching to the local depot, and allocating drivers.

The system ran on AIX Unix.

Developed the system using embedded SQL C.