# Guy Pender

# 18 Oakwood Drive, Angmering, West Sussex, BN164GB

# guy.pender@me.com

# A highly motivated, respected and technically competent software engineer writing C# code using the Microsoft .NET Framework. I am currently part of a small team writing code for the Parker SSD AC30 motor controller and the sole member writing tools software based on the CoDeSys PLC framework. I am responsible for all aspects of the tool design process from idea generation, code development and testing through to deployment to the Company website. As the only non-embedded software engineer I engage effectively and considerately with the other members of my team, whilst challenging the norms to improve and simplify our customers’ experience. I communicate regularly with customers and I am always keen to make a positive and friendly impression. I keep myself up to date with the latest web and mobile technologies and wherever possible introduce these ideas to the team for consideration.

# Recent Software experience

* 10 years plus of C# experience
* WPF UI design using MVVM design patterns. Some experience writing custom controls.
* Microsoft TFS source control and work item management
* Unit testing with some TDD and BDD
* Working in a Scrum (Lite) environment with 3 week sprints
* Microsoft SQLSERVER. Writing procedures for importing business system data.
* AngularJS development. Creating components and services for rich client side web pages served from the AC30 onboard webserver.
* Webpage design exposing business system data using both classic ASP and ASP.NET
* *Organized an IOT workshop for the engineering department using Raspberry PIs and the Node-Red IOT programming tool to push data up to an Exosite Web portal.*
* HPUX Unix Server administration

# 2008 – Parker Hannifin (Parker SSD) R&D Software Engineer, member of a team responsible for motor controller tools development.

Joined the R&D software team comprising 7 firmware engineers, writing C code for motor control, communications, system architecture and the simple UI, and 3 tools engineers including myself writing C# for the .NET framework. The team have been writing code for the AC30 which is the latest motor control product with built-in Ethernet, onboard SD card, numerous Fieldbus options and a powerful PLC application engine.

The tools software is used by the customer during setup and commissioning and can be used to create sophisticated industrial motor control solutions with or without a separate system PLC. The AC30 firmware and tool use an IEC 61131 PLC solution called CoDeSys V3 provided by 3S in Kempten Germany. The CoDeSys platform uses a plugin framework which makes it highly customizable and a good fit for the AC30 tool requirements. The framework is written for Microsoft .NET using C# and technologies from both WinForms and WPF (Windows Presentation Foundation).

During the development I have worked closely and effectively with the embedded software team as the drive firmware has evolved. I have been responsible for the design, development and delivery of the following projects with the exception of the PDQ which was originally conceived by my line manager, though it has undergone a major overhaul since then.

* Parameter Database Editor
* Parker Drive Quicktool (PDQ)
* Parker Drive Developer (PDD)
* Drive Configuration Tool (DCT) development tool for easy FW installation
* AC30 Scope Tool. A non-Codesys Oscilloscope tool
* AC30 File System Browser. A non-CoDeSys tool for accessing the SD card file system

Parker Parameter Database Editor

The AC30 motor controller uses over 1000 parameters which setup, define and control its operation. It was clear at the start of the development that a database to capture and store these parameters was required for both the firmware and the tool development. SQLSERVER was initially considered but a more standalone solution using the Microsoft SQL CE was eventually selected: SQLCE is file based and could be more easily archived with the firmware in the source code management system, Team Foundation Server. I developed the Parameter Database Editor to use the SQL CE files and built my code using Microsoft LINQ and Entity Framework technologies. The database tables, editor and editor features evolved over time as the firmware engineers developed their code. The software has the following features

* FE Parameter Hierarchy Editor. The firmware was developed within a FE (Functional Entity) framework with parameters belonging to a specific FE, eg Motor Control. For the Parameter Database Editor I provided a tree based editor where FEs could be defined and parameters added.
* Context Sensitive Parameter Editor. I created an editor for parameters which would only show attributes required by the particular type of parameter being edited. For example an Enum parameter does not have Min/Max defined.
* Customer Menu Editor. A drag and droppable tree based editor for creating menus and and adding parameters to define how the user interacts with the parameters inside the drive. The menus are visible to the user using both the drive’s keypad and the tool.
* Dependancy Editor. For many power related parameters in a drive it is necessary to define relationships between a parent parameter and several child parameters whereby the values of the children are set by the value of the parent. The Parameter Editor has within it a Dependancy editor where these relationships can be defined. In previous drive developments these dependency relationships defined solely in the firmware, using the Parameter Editor they were captured outside the firmware and available to the tool software as well.
* Validity Rule Mechanism. I created the concept of validity rules whereby the visibility of parameters to the user could be switched on or off based on the state of other parameters. So for example CANOPEN fieldbus parameters can hidden or shown based on whether the CANOPEN option is selected or not and this is defined as a validity rule. The validity rules can be quite complex and use logical operators, brackets and parameter ids.
* Translation Management. An editor is provided to collate all strings in use throughout the various database entities and provide a mechanism for creating and editing language translations. The translations again are available to both the firmware and tool.
* Wizard Page Editor. A simple wizard is available on the drive keypad which guides the user through questions about the various areas of the drive from Motor setup through Comms Option and IO Option. The Parameter Database Editor provides a simple Wizard Page editor to create these questions.
* Outputs to Firmware and Tool. Outputs to the firmware build chain are provided automatically through the use of command line version of the Parameter Database Editor. Output to the tool is achieved using an xml file, the format of which is defined by the Codesys IEC PLC engine.

Parker Drive Config Tool (DCT)

This tool was originally developed as a diagnostic and test-bed tool during the early stages of the firmware Ethernet development. It provides a mechanism for scanning using a UDP Ethernet broadcast with an internally defined message format. Drives which reply to the message are shown in a list which can then be used to edit the drive’s Ethernet properties; enable DHCP, change IP address, drive name etc.. Additionally the tool can be used to change the drive’s installed firmware and also gain quick link access to the built in web server.

Parker Drive Developer Tool (PDD)

The Parker Drive Developer tool is the largest project I have embarked upon since I joined R&D. It comprises several plugins which together provide the following functionality.

* Drive Scan and Selection. This provides a more graphical means of finding AC30 drives on an attached Ethernet network and indicates to the user key parameters such as firmware version, IP address, power stack size and which option cards are fitted. A drive identify feature allows the user to make the drive’s keypad display flash and the presence or otherwise of an SD card is also shown.
* Login Manager. The login manager identifies differences between the actual hardware fitted (eg. the option cards) and what has been set in the project and gives the user a simple way to resolve these. For other parameters it provides parameter synchronization management to allow the user to choose the project or drive values.
* Drive Customization. Drive customization allows the tool user to create their own drive parameters with their own names, units etc., link them to their IEC PLC code and make them appear on their own custom Menus, visible through the drives keypad display, the drive’s webpages and the tool. The customization is made possible using various editors including a Soft Parameter Editor which borrows heavily from the work I did on the Parameter Database Editor and a Soft Menu Editor.
* Oscilloscope. An oscilloscope tool is available which uses the firmware fast trace functionality to capture parameter data. The tool provides the configuration for the fast trace feature inside the drive firmware which can be setup for continuous or triggered data capture on the PWM or a 1ms T1 tick. It provides 7 channels of data and provides Rolling or Triggered capture modes.
* Parameter Editor. The parameter editor provides online monitoring and viewing of parameters inside the drive. It supports the validity rule mechanism and so is context sensitive to the setup of the drive. A number of custom controls were written to deal with the various special parameter types such as Bitfields and Prefs.

# 1999 -2008 PCB Design Team Leader. Team of 3 PCB Layout Engineers.

Role responsibilities

* Unix System administration
* PCB Design Process improvement
* PCB Design Team management

The PCB Design tools Unix based Mentor Graphics Boardstation. I learned how to administer the Unix operating system and use the Boardstation tools to layout boards. Armed with this knowledge I was able to devote most of my time in improving the process in the following ways.

When I took on the role most designs used a so called ‘design library’ which was a full copy of all geometries used and as such completely decoupled from the master library. I identified this as a maintenance and design control issue and my first job was to switch to the use of the master library for all designs.

Part of reason this had come about was the difficulty in creating and adding new parts in the design capture process. To help solve this I created menus driven by simple text files into the Schematic Design Capture tool using the Awk scripting language. Previously the menus were hard coded Awk script. These text files with rows for each part number allowed other design data to be added such as test point count; a low value resistor (eg. <47R) ideally needs 2 test points per node. The extra part data was used in automatic checking scripts.

In the Layout tools I introduced automatically generated colour coded voltage maps for checking clearances on high voltage circuit boards. This was driven by data added at the Schematic capture stage by the Design Engineer in the form of net type names (eg: DC\_LINK and SELV).

I wrote tools for creating collated pdf drawings of schematics, coloured voltage maps and PCB Manufacturing drawings.

During my tenure as the PCB Design Team Leader I embarked on a number of other related engineering projects.

2003 ?? Intranet access to Company Business System

Following the introduction by our IT department of a SQLSERVER with tables of data extracted from our business system (eg Part Numbers, BOMs etc) I identified the potential for bringing all sorts of engineering, purchasing, supplier, and sales data to our then very limited Intranet. Using Microsoft IIS and Asp web page technologies I was able to create a web based environment for searching, viewing and in some cases maintaining part information.

I also identified that our then paper based drawing office should be converted to electronic pdf format which then became an integral part of the growing company intranet. I played the key role in making this happen in terms of defining SQL tables and providing pdf creation and publishing tools.

The pages I created are still very much in use today. They shorten the learning curve for accessing data in many areas of the company and in some cases eliminate the need for expensive Business System licences.

2008 JDE: New Business System Introduction

I played a key role in the introduction of JDE as the new company business system, liasing with JDE experts from Parker House in Hemel Hempstead.

JDE intranet data. I was asked to provide mechanisms for extracting data from the new business system with the brief to replicate the Infoflo intranet. I achieved this through the use of generic import procedures written in SQLSERVER Tsql driven by templates held in other easily editable tables. In addition to this I was able to build in half hourly updates for more time critical data such as stock movements.

With the introduction of JDE, a number of older business processes had to be replaced and the standard JDE solutions did not meet our needs; I was therefore asked to provide solutions.

WebSphere base queue process for transmitting work order completions to the main JDE computer system in Hemel Hempstead.

I wrote a Virtual Print Server (using the Line Printer Daemon protocol) to capture spool files from JDE from various Business processes. The Virtual Print Server was capable of handling the spool files for various types of labels including simple receive data and print (eg Pick labels) through to more complex labels such as Product Labels with many calculated data fields and using a C# class library I created for the purpose.

# 1989 -1999 Manufacturing Engineer, various projects involving Smt Assembly, Wave Soldering and PCB Design for Manufacture.

# 1989 Graduated from Loughborough University with 2:1 in Electronics Manufacturing Engineering