Microsoft Office Automation

We have been **automating** the **Microsoft** product range from it's **inception**.

Most users are completely unaware that when they create a **macro**, they are actually adding Visual Basic for Applications code (known as **VBA**) in the background. Whilst the **code created** in this fashion might **not be as efficient** as writing it by hand, it gives a good starting point to understanding the fundamentals of programming in VBA.

Microsoft Developers have other options to extend functionality such as:

- using **Visual Studio** and **Microsoft Office Developer Tools** (known as **VSTO**) to create **.NET Framework** applications (also named **Office solutions)**

- with newer versions of **Microsoft Office** create apps (as an **Office Add-in**) using almost any web language such as **HTML**, **JavaScript**, **CSS3** and **XML**

For **many tasks** and a **rapid turnaround**, **VBA** is all that is needed to create a working **Add-in** which can then be installed by the user with little effort.

Individual examples of our own work include:

- A complete **Excel** system including a separate logging form for the **license brand holders** of **Pink Lady™** apples and **Tenderstem™** broccoli using a combination of VBA coding, Pivot Tables, Charts and Slicers for reporting product quality from data collected via random inspections at distribution hubs.

- A **Timesheet Input** system written entirely in **VBA** code for **Excel** as a separate **Add-in**. It integrates with our own commercially available **Time Recording** software using the **Client**, **Staff** and **Work Type** references held therein.

- An **Final Accounts Production** add-in for **Excel** which enables Accountants to produce a full set of final accounts for clients including pre-formatted sets of accounts, working papers, table of contents, numerous additional buttons for specialist accountancy styling of data and various printing options. An additional add-in for **iXBRL** tagging of data to conform with **HMRC** filing requirements was created using the custom field properties, so that this new data was hidden from the user.

- An **Excel** button to create a new **Working Paper** with references, specific formatting and sorted into correct order within existing worksheets for accounting purposes.

- An **Excel** button to show or hide ranges in a worksheet based on criteria held in the workbook elsewhere or indeed within the range itself.

- **Excel** buttons for **single**, **double-underlining** and **formatting** of **currency amounts** in a pre-defined style.

- Data cleansing routines in **Excel** including de-duplication, matching, auto-fill for missing data, re-organisation/re-formatting, normalisation, removal of erroneous data, auto-correction and other facilities as required by the end user.

- In **Word** having extra **print** options for **client letters** so that **originals** have a first page on **letterhead** style paper and **copies** are automatically printed on plain paper only where more than one **print tray** are loaded appropriately.

- In **Outlook** when viewing an email message, automatically show the **sender's address details**.

- In **PowerPoint** automatically creating a **slideshow** made up of a **list of company names** registered at your address from an **Excel** workbook, so that they can be displayed in a continuous loop on a display screen.

Of course you can extend Excel with new features but you can also create entire business processes with very little knowledge, which is one of the reasons why Excel is so popular amongst end users.

If there is a task which might benefit from **automation** in any of the **Microsoft product range**, then just let us know. It might **cost less than you think**, especially if such a feature might be of use to others. You might even **recoup** your initial outlay by a sharing of **commission arrangement** for selling on your idea.