

# **A new methodology, and proposed usability evaluation framework, for delivering models to users via desktop software and Web services**

**Mills, Kathryn<sup>1</sup> and Peter Fitch<sup>1</sup>**

<sup>1</sup> *Commonwealth Scientific and Industrial Research Organisation (CSIRO), Land and Water Division, Water for a Healthy Country Flagship, Canberra, ACT 2601*

*Email:* [katie.mills@csiro.au](mailto:katie.mills@csiro.au)

**Abstract:** There has long been a disconnect between the resources of the web and desktop commodity software, such as Microsoft (MS) Excel. We attempt to unite the world of the Web (through Web services) with the familiarity of desktop applications – in this case Microsoft Excel – and to test the effectiveness of this strategy through a usability evaluation framework.

Excel is a widely used spreadsheet package and offers powerful data manipulation features for environmental managers. However until recently there has been no easy way to access, from within Excel, world-wide environmental data and specialised statistical services. Excel has also been shown to give inaccurate results in some circumstances when directly used for statistical analysis (McCullough & Heiser 2008).

Web applications are the traditional way to use Web services and this was our first approach. Yet the Web was never originally intended to support the lengthy workflows inherent in tools for environmental data analysis, such as that required by our project (to make Australian water quality data available for analysis over the Web). The web client application (also referred to as the “Web client”) we developed also faced the overwhelming restrictions of slowness in data transmission; difficulty of state persistence; and rigidity in configuration of data source options. However, some of these problems resulted from our inexperience in Web application development and insufficient project time to explore other Web client application examples.

Our desire to showcase a range of client options to our stakeholders (and our perceived limitations of the Web) motivated the requirements for an alternative: an Excel tool. We used Visual Studio Tools for Office (VSTO) technology to develop an Excel Add-in (also referred to as the “Excel client”) rich in desirable functionality such as data source configuration and workflows, with the ability to access world-wide environmental data and statistical processes through Web services.

One of the challenges for a software developer working in a research environment is the development of software applications in anticipation of users’ needs. The impetus for developing the Excel tool was to overcome the afore-mentioned restrictions of our Web client. So this assumption that a desktop environment provides superior usability should be evaluated. In response to usability problems with the Web client, we developed a usability evaluation framework (built on a previous in-house scoping study) and considered it during development. When our product is released, this framework could be used for evaluating its usability and the results used to inform subsequent client architectural decisions.

Development of the Excel tool suggests that while the web is ideal for information delivery and presentation to a wide international audience, the traditional desktop client (when combined with technology for accessing world-wide data and statistics) may have significant usability advantages for the environmental management user sector. And now, with the aid of the usability evaluation framework presented in this paper, this proposition can be tested.

**Keywords:** *VSTO, Excel Add-in, Excel client application, Web client application, usability evaluation*

## **1. INTRODUCTION**

### **1.1 Background of the project**

The Excel tool was developed as part of a project to make Australian water quality data available for inspection and analysis via web-services and statistical models. This project, Water Quality Statistical Analysis and Reporting (WQSAR), aims to empower those needing to report on water quality condition and trends; as well as assessing compliance to, or progress towards targets. Data<sup>1</sup> are available for water quality variables such as electrical conductivity, pH, total nitrogen, total phosphorus, turbidity and total suspended solids.

Traditionally, such users as those in local government tasked with assessing water quality in their catchment area, needed to obtain the water quality data and then send it off to statistical experts for analysis. The tools, or client applications, delivered as part of the WQSAR project, give users access to Australia-wide water quality data and enable them to have that data analysed within minutes. Supported statistical methods include seasonal Kendall's Tau, linear regression and the Generalised Additive Models.

The project divides into two main areas: that of the tools/client applications (i.e. the software with which the user interacts, often residing on the user's own computer) and the server side, with software to obtain and analyse data via the use of web-services. The web service for obtaining data contacts either Web Feature Services (WFS) or Hydrological Information Services (HIS) to retrieve that data. The Web Processing Service (WPS) sends that data off to statistical processes<sup>2</sup> for analysis.

### **1.2 Scope**

We will be discussing the functionality and main software components involved with the client-side application development and referring to the project requirements when discussing our usability evaluation framework.

Usability is defined by Nielson, 1993, as a measure of a system's acceptability to its users – how well it meets their goals. Nielson (1993, p26) identified five usability attributes (learnability, efficiency, memorability, errors and satisfaction) but others have since been developed and used, for example by Cuddy et al (2007).

Given the usability focus of this paper and space restrictions, we won't be going into details of the software design or of other admittedly important non-functional requirements (such as robustness, maintainability, security etc). We won't be discussing details of how the Web services work or describing the statistical processes. The only mention of the web client will be in comparison with the Excel tool. The discussion of the web client and its limitations occurs in a pre-Web 2.0 context given our lack of project time to truly understand the Web 2.0 world.

The users involved in usability testing or their work environments will not be mentioned as these are currently unknown (but once identified will be involved in the evaluation). We will be discussing some general strategies for formulating tests but not the tests themselves, although these tests (and their results) may be mentioned in our Modsim conference talk.

### **1.3 History of development**

The first client we developed was a Web application (also referred to as the "Web client"), to communicate with our web services. It exhibited significant usability problems though – slowness in data transmission/reception (the latency of the Web); difficulty of persisting data in between sessions due to the stateless nature of Web transactions; difficulty of coordinating complex workflows; suboptimal interface responsiveness, due to the necessity of posting back user choices to the server for in-session persistence; and rigidity in configuration of data sources. These problems may have been overcome with further research, however our deadlines precluded that and we were contracted to deliver a variety of clients.

---

<sup>1</sup> measurements taken by water quality monitoring ["gauging"] stations and stored on servers in databases accessible by Web services

<sup>2</sup> written in the R programming language <http://www.r-project.org/>

The rich and intuitive functionality of desktop client applications, (especially Excel, as it was already employed for statistical analysis work by our user group) then looked alluring. The Microsoft Office suite of products has long been a daily aspect of many professionals' lives and supports complex workflows<sup>3</sup> (through Windows forms applications), persistence, as well as ease of importation of, user's own data. For example, in a spreadsheet a user has many data importation options, from using a local database to typing in the data directly. Graphs are easily embedded or generated from templates; tables are inherently supported by the spreadsheet environment. Spreadsheets can be locally and reliably saved to be worked on later, thus solving the issue of data persistence.

However traditional purely desktop clients, e.g. MS Office applications, are unable to easily take advantage of what the web offers, because they are only intended to be used in a desktop or locally networked environment. Excel in particular is not designed to run on a server as a Web service in its own right and the infrastructure doesn't exist to call Web services directly from Excel (without any supporting VBA code) (Bullen et al, 2005).

Having all the code directly inside the application itself (e.g. formulae hard-coded into Excel spreadsheets) also presents maintenance and version control issues as you can have multiple copies of spreadsheet with embedded formulae and no-one is sure which the current or official one is. It is also difficult to update the embedded program code without destroying what extra information the user has inputted into the worksheet, for example they may have tailored or improved the macros (Bullen et al, 2005).

Microsoft delivered Visual Studio Tools for Office (VSTO) as part of its Office 2003 release. VSTO enables .NET developers to link .NET framework code to Office documents. Thus rich and powerful applications can become part of Excel via the development of Automation Add-ins. We took advantage of this technology when developing our Excel client application.

#### 1.4 Add-in application technology

The add-in consists of two Windows Forms applications, one for retrieving the data and one for analysing the data. These will be discussed in "The Workflows". Behind these application interfaces are C# classes, which interact with C# .NET 3.5 libraries we developed for contacting the Web services to retrieve and analyse the data and for writing images and data to the Excel spreadsheet.

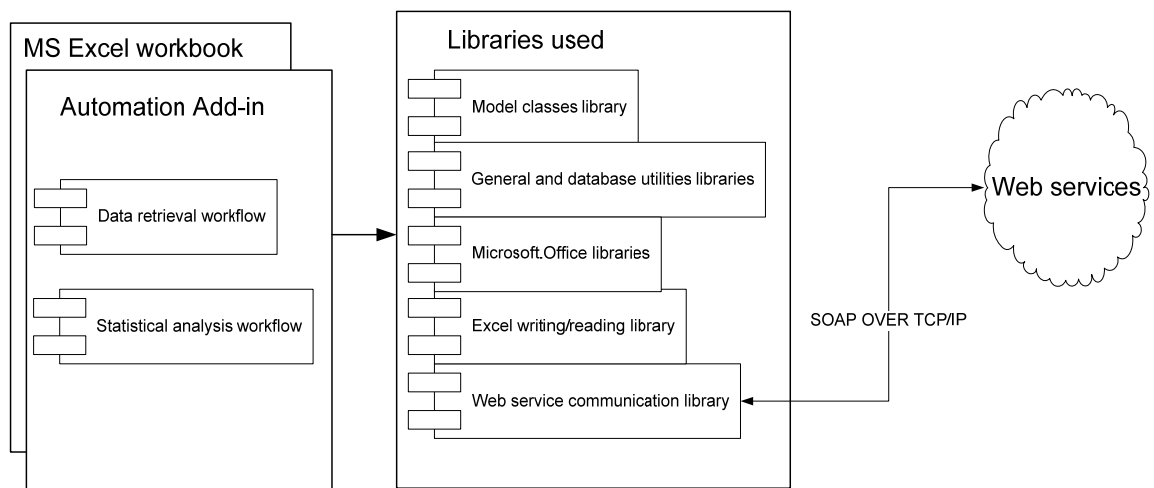


Figure 1. Main system components

<sup>3</sup> Workflow being a series of steps needed to be undertaken in a certain order to reach an end goal. In a software application, such as a Wizard, each step can be represented by one user interface page or tab and will consist of a set of tasks or information to be supplied in order to complete that step.

## **2. THE WORKFLOWS**

When Excel 2007 is opened, the “WQSAR” add-in is accessible within the top menu of the Excel Workbook under “Add-ins”. The add-in delivers two main Windows forms workflow applications (or wizards). The first in the menu is “Retrieve Data”, the second, “Run Statistical Analysis”.

The first obtains the water quality data from a Web service (or just locally) and imports it into the first worksheet of the workbook (as well as downloading all the individual data inspection elements, such as graphs and data). The second workflow involves setting up the data analysis and getting the results back from the statistical Web processing service (WPS) into the next worksheet (and downloading a .pdf report). Persistence of choices as the user steps through the workflow will ensure that they can go back and change their choices without having to re-enter everything. Navigation is forwards, via the NEXT buttons or backwards via BACK buttons. Non-fatal errors return the user to the previous screen so they can try again.

The advantage of having the two workflows separate is the flexibility of choice in which to run. The disadvantage is that there is no state persistence between them (so the user may have to supply information which a workflow can't ascertain from the spreadsheet contents).

### **2.1. “Retrieve Data” workflow: obtaining the data**

This workflow is optional; as the advantage of an Excel client (Workbook) (as opposed the Web client) is that the user doesn't even have to use the Web service to import data. They could import it from a local database by setting up a connection, import it from another spreadsheet or just copy and paste it in.

#### **Step 1: Introduction**

Before they embark on something that may be new, users need a summary and explanation of what they will be choosing as they step through the workflow. The introduction is dismissible on subsequent uses (Bollaert, 2001).

#### **Step 2: Choosing a Web service**

Choose either from a list of Web services or to set-up a new service (entering a description and the service type, for example WFS or HIS), which will be remembered by the Web service it is registered with.

#### **Step 3: Choosing the station and its data**

Choose a water quality monitoring station from the list, after which the water quality variables monitored by the station appear. From these a water quality variable, and optionally a flow variable, must be chosen. When the user selects a water quality variable, the “from” and “to” dates of the duration of measurements appear. They can either be accepted or changed.

Select a data download destination.

Finally, select where in the Excel application the data is to be inserted, either in the currently selected cell or in a new worksheet at the first cell (A1).

#### **Step 4: Finish: summary of choices and initiating the data retrieval.**

This displays all the data options chosen, with the option of navigating backwards to change anything configured. Once the "FINISH" button is pressed, the add-in contacts the data Web service, requesting the specified data.

#### **Results displayed in the Excel spreadsheet**

The first blank worksheet of the spreadsheet (or other user-selected location) is populated with the returned data and [data inspection] graphs of it showing its suitability for analysis. Local links to the data and graphs (download to the stipulated folder) are displayed.

Mills et. al., *A new methodology, and proposed usability evaluation framework, for delivering models to users using commodity tools and web services.*

## 2.2. “Analyse Data” workflow: statistically analysing the data

### Step 1: Introduction

See the introduction page of the first workflow (Step 1).

### Step 2: Choosing a Web processing service and data-source

The statistical processing service options are displayed. For the foreseeable future we will only offer one Web processing server (which will send the data off to the statistical analysis application). The raw data range (either “All” or start and end Excel cell locations) is selected through a third party range selection control (reallyBig\_J, 2009).

### Step 3: Describing raw data

The user describes the some of raw data’s attributes (such as source station name, station latitude/longitude) and some, such as variable names, are auto-populated in the GUI from spreadsheet fields.

### Step 4: Setting statistical analysis options

Choose the location to download results to and analysis options, including:

- statistical trend analysis type (linear, seasonal Kendall’s Tau, Generalized Additive Model)
- whether to operate on the natural or logarithmic (transformed) scale
- whether to adjust for changes in flow or not
- the flexibility of any nonlinear trend.

### Step 5: Summary of choices and initiating the data retrieval.

This displays all the analysis parameters chosen, with the option of navigating backwards to change anything configured. Once the "FINISH" button is pressed, the add-in contacts the statistical web service and calls the analysis method chosen, passing it the data it has packaged up from the spreadsheet.

### Results displayed in the Excel spreadsheet

A new worksheet in the Workbook is populated with the returned analysis results (graphs and tables). Local links to a comprehensive pdf report containing a summary of all the analysis information (and a zip file of all its contents) are displayed.

## 3. DOES THE EXCEL CLIENT OFFER A SUPERIOR USER EXPERIENCE TO THAT OF THE WEB CLIENT? A USABILITY FRAMEWORK FOR EVALUATING THIS CLAIM

The point of coming up with some guidelines, or heuristics, about good user interface design is to prevent problems and evaluate built interfaces for usability. Our guidelines, as expressed in the usability evaluation framework below, build on a previous in-house scoping study that identified usability attributes and evaluation indicators relevant to water quality software (Cuddy et al, 2007). We consider a subset of these attributes, relevant to our project’s original non-functional requirements, and associate them with measurable goals and metrics.

We have designed the Excel client in accordance with the values of this framework. However the application’s usability will only be revealed when we can apply this evaluation to real users of the software in their actual environments.

Core desirable usability attributes for which we are claiming Excel client superiority (over our Web client – not Web clients in general):

- *Familiarity/learnability* – the application should be consistent with the user’s prior experience of other Windows forms applications and more usable as a result. Most people would have used a wizard-like interface for installing software, for example.
- *Effort (degree of)* – it should be less effort than the web application to obtain, setup and analyse data.
- *Availability* – with the client application on the desktop rather than the Web, availability should be greater. There should always be some functionality even when the Internet is unavailable.
- *Explanation* – a benefit of this application is that the user group does not need to know the inner workings of the statistical algorithms to use it. However they will need sufficient understanding of the data inputs describing the analysis and how to interpret the results. Unfortunately time constraints for

the Web client development meant that the help available for non-statistically savvy users was inadequate so this makes it an essential goal for the Excel client.

- *Operability* – we claim that the built-in support for workflows in Windows applications (the wizard approach), and the stand-alone nature of each workflow, will result in the user being better able to operate it to achieve their goal.

### 3.1 Goals of applying usability attributes to the Excel client application design

(based on findings from Cuddy et al, 2007 and on usability heuristics by Nielsen, 1993).

- Attribute: Familiarity/learnability

*Objective:* Users feel at ease with navigating through the system and can easily learn how to use it.

*Measurable Goal:* Observation will determine whether users can navigate through the main workflow without being instructed. Monitoring could include the number of times the BACK button is clicked.

*Methods of ensuring attribute quality (in product):* Use of the windows forms GUI components in the interface. Use of the “workflow” or “wizard” metaphor in GUI design. Use of clear and traditional navigational aids, for instance “NEXT” buttons to navigate to the next workflow screen and “BACK” buttons to return.

- Attribute: Effort (degree of)

*Objective:* Reduced effort required to obtain and analyse data (compared to the Web application). The user will easily be able to take advantage of the multiple ways in which to obtain their raw data for analysis.

*Measurable Goal:* Observation will determine whether users can take advantage of the multiple ways in which they can set up their data sources.

Timing will determine how long it takes, in comparison with the Web client, to complete a task (such as setting up all the parameters for data analysis and obtaining the analysis).

*Ensuring attribute quality:* Design application so that user can import data from a variety of sources (i.e. the Data Retrieval Workflow, a local database or just copy and paste). Ensure they are aware, through design and help text on the GUI, of their data source options. (See also “Explanation” as that also reduces effort).

- Attribute: Availability

*Objective:* Provides some basic functionality when web services are unavailable or in non-fatal error situations, so that the system can still operate in a local Office environment.

*Measurable Goal:* The web client allows some basic functionality even if the web services are unavailable (such as allowing the user to input their own data).

*Ensuring attribute quality:* On start up the Excel application should check the availability of Web services. If unavailable the application should inform the user of such and their options for off-line data input (such as a local database connection or even just copy and paste). Robustness built in through recoverable error handling when Web services are unavailable.

- Attribute: Explanation

*Objective:* Users can easily find answers to questions about the application’s use and domain knowledge.

*Measurable Goal:* When asked the user can explain: the purpose of the workflows; data source options; how to select a station and data; how to inspect the initial data to ensure it is fit for analysis; what inputs are required for statistical analysis; how to understand and access the results.

*Ensuring attribute quality:* Contextual help, glossary, explanations (in layperson’s language) of statistical terms and algorithms are provided and easily accessible. With involved procedures, help will be in a step-by-step format, task-oriented and remain open in a separate window throughout the operation. Examples of algorithms in use will explain their real-world relevance.

- Attribute: Operability

*Objective:* Users should feel in control of the application and the choices they make – not the other way around. Both expert and novice users should be catered for.

*Measurable Goal:* Interviewing users to determine satisfaction with workflow operability. Low error rates (determined through the number of errors written to log files) and little time taken to recover from error states.

*Ensuring attribute quality:* Workflows will allow users to view their choices before actions are performed and to navigate back to change options (achieved through state persistence of choices). Users will be able to abandon workflow when they wish. Novice users will have access to a basic workflow which reflects the most common operation. Error messages to be clear (in non-technical language at the highest level, then technical details later), precise about the exact system artifact that caused the problem and help the user try to solve the problem.

#### **4. DISCUSSION/FUTURE WORK**

Short term priorities will involve devising tests (to be given to users in South Australia) from the usability evaluation matrix, both observational/qualitative and quantifiable, using proven usability testing methods, metrics and targets. We hope to present the results of these tests at Modsim09. For example, Nielson (1993) has determined optimal system response times. "Customer evaluation" software, such as Morae by Techsmith, could record user interaction/satisfaction (Techsmith, 2009), especially when users are separated by great distances, which precludes being personally there at each user site.

Long term priorities are to research technologies and strategies that could improve the Web client's performance and usability. These could include taking a Web 2.0 approach; using Google docs to bring office-like functionality in; a Workflow engine to better accommodate complex workflows; and individual logons/asynchronous message passing to enable returning later on to retrieve results during lengthy calculations (this would also benefit the Excel client).

#### **5. CONCLUSION**

This paper proposes that the traditional desktop client (when combined with VSTO technology for providing feature-rich Windows forms interfaces and accessing world-wide data/statistics) has significant usability advantages for the environmental management user sector.

The Excel client application enables water quality data and statistical algorithms to be accessible by a broader range of users than would have been the case if detailed knowledge of the algorithms was required. It opens up a world-wide source of environmental data. Given the familiarity to users of Excel, Windows Forms and wizard-like workflows, as well as its faster response times (due to not having to post back user selections to the server), it should also be more usable than the Web client.

However, only with evaluation against the framework discussed in this paper with real users will we be able to prove or disprove these claims of usability. And only software testing will prove whether the application has superior response times to that of the web application.

Primarily, though, it is our hope that these investigations, for which our paper has prepared the way, will aid our future decision-making when considering the appropriate client technology to suit project requirements. Ideally, usability considerations will become central to our software development culture.

#### **5. ACKNOWLEDGEMENTS**

I gratefully acknowledge Peter Fitch the original concept of the Excel client. I am indebted to Susan Cuddy and Nick Murray for advice on usability evaluation (see Cuddy et al, 2007 in References) and reviews of this paper. Many thanks also to my project leader, Brent Henderson, and to my colleagues Dave Penton and Jamie Vleeshower for their reviews of this paper. Thanks to Qifeng Bai for his advice about my diagram.

This paper arose out of the Water Quality Statistical and Reporting (WQSAR) project, jointly funded by the National Heritage Trust / Caring for our Country, the Bureau of Meteorology and CSIRO. The WQSAR project is part of CSIRO's Water for a Healthy Country Flagship.

#### **REFERENCES**

- Bollaert, J (2001), Crafting a Wizard, <http://www.ibm.com/developerworks/library/us-wizard/?n-us-9131>
- Bullen et al (2005), Professional Excel Development, Addison-Wesley.
- Cuddy, S.M; Farry, K; and Murray, N (2007), Usability Indicators for eWater Products. A scoping study report for the eWater CRC, CSIRO Land and Water
- Fowler, M (2003). Patterns in Enterprise Architecture. Addison Wesley.
- McCullough, B.D. and Heiser, D.A. (2008), On the accuracy of statistical procedures in Microsoft Excel 2007, Computational Statistics and Data Analysis, 52, 4570–4578
- Nielson, J (1993), Usability Engineering, Morgan Kaufmann, San Diego, USA.
- reallyBig\_J(2009),VS.NET Excel Addin Refedit Control, <http://www.codeproject.com/KB/office/RefeditControl.aspx>
- Six, J (2009), Writing Usability Requirements and Metrics, <http://www.uxmatters.com/mt/archives/2009/02/writing-usability-requirements-and-metrics.php>, Techsmith (2009), [www.techsmith.com](http://www.techsmith.com)