



OpenAIR@RGU

The Open Access Institutional Repository at Robert Gordon University

<http://openair.rgu.ac.uk>

Citation Details

Citation for the version of the work held in 'OpenAIR@RGU':

WOOD, D. C., 2007. Research and development of enhanced, integrated and accessible flow metering software for industry. Available from <i>OpenAIR@RGU</i> . [online]. Available from: http://openair.rgu.ac.uk

Copyright

Items in 'OpenAIR@RGU', Robert Gordon University Open Access Institutional Repository, are protected by copyright and intellectual property law. If you believe that any material held in 'OpenAIR@RGU' infringes copyright, please contact openair-help@rgu.ac.uk with details. The item will be removed from the repository while the claim is investigated.

RESEARCH AND DEVELOPMENT OF ENHANCED,
INTEGRATED AND ACCESSIBLE FLOW METERING
SOFTWARE FOR INDUSTRY

DAWN COLLEEN WOOD

A thesis submitted in partial fulfilment of the requirements of The
Robert Gordon University for the degree of Master of Research

This research programme was carried out in collaboration with

KELTON® and TUV NEL

December 2007

Acknowledgments

I would like to thank Graeme Ryan, academic supervisor at TUV NEL, for his continual encouragement and advice throughout this project and for his interest in the overall project. I would also like to thank Christian Guest, company supervisor at KELTON[®], for his continual advice throughout the project and for introducing me to the use of n-tier applications, XML and web applications.

The research project was supported by KTP through TUV NEL and KELTON[®], without this collaboration the project would not have been possible.

I would like to thank David Davidson, Senior Lecturer at The Robert Gordon University for his advice on the structure of the thesis.

Finally, I would like to thank John for his encouragement and support throughout this project and to Isabel for holding off arriving into the world.

Abstract

This project was an investigation to find improvements required in the delivery of software for the flow metering industry. The project has resulted in the repackaging of existing software using appropriate technologies. This included developing software that is accessible via the web and extending functionality whereby a user can import and export information in a variety of data formats. The software was successfully revised and is now commercially accessible to the flow metering industry.

The project was performed in the context of a KTP (Knowledge Transfer Partnership) programme with academic supervision provided by TUV NEL (the academic partner) on the premises of KELTON® (the commercial partner) who provided day-to-day project management supervision. The project was in collaboration between the two organisations with the joint aims of facilitating knowledge transfer between the organisations and enhancing the market performance of the commercial partner.

The main objective of the study was to gain a full understanding of the needs of the flow metering industry in terms of software and delivery via web or standalone application. Web based applications are new to KELTON® so it was necessary to investigate the methods of delivery. The work concentrated on investigating techniques to modularise code, allowing flexible access to data between applications and on data presentation.

At an early stage of the project an online market survey program was developed and appropriate questions were used to get customer feedback. The results were analysed and used to prioritise work.

Following the review, the current software architecture was found to be unsuitable so new approaches were investigated. The software was created using an n-tier architecture which is a method of splitting common code into separate components.

Web based applications were found to be slower than standalone applications. However, web applications benefited from not having to fully install software on individual user PCs therefore allowing access from anywhere that users have access to the network.

List of Abbreviations

The following is a list of abbreviations used within the text of this report, along with the relevant descriptions. When initially used in the report the abbreviation appears in brackets after the description, after which only the abbreviation is used. For ease of reading the abbreviations have been separated into software and industry terms.

Software

ADO	Active X Data Objects
AJAX	Asynchronous JavaScript and XML Applications
API	Application Program Interface
ASP	Active Server Page
BES	Back End Server
COE	Common Operating Environment
COM	Component Object Model
CSS	Cascading Style Sheet
DAO	Data Access Objects
DCOM	Distributed Component Object Model
DHTML	Dynamic HTML
DLL	Dynamic-Link Library
DOM	Document Object Model
DTD	Document Type Definitions
FES	Front End Server
GUI	Graphical User Interface

HTML	Hypertext Markup Language
HTTP	Hypertext Transport Protocol
IDE	Integrated Development Environment
IIS	Internet Information Service
JET	Microsoft Joint Engine Technology (JET) Database Engine
JSP	Java Server Page
MDI	Multiple Document Interface
MIME	Multiple Internet Mail Extensions
OCX	OLE Custom Control
OLE	Object Linking and Embedding
PC	Personal Computer
SAX	Simple API for XML
SGML	Standard Generalized Markup Language
SQL	Structured Query Language
UI	User Interface
UML	Unified Modelling Language
URL	Uniform Resource Locator
WWW	World Wide Web
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
XSD	XML Schema Definition
XSL(T)	Extensible Stylesheet Language (Transformation)

Industry

API	American Petroleum Institute
AGA	American Gas Association
ANSI	American National Standards Institute
BS	British Standard
DP	Differential Pressure
DTI	Department for Trade and Industry
FLOCALC [®]	Flow Metering Calculation Software
FM ² P	Fiscal Metering Management Package
GOMB	Gas and Oil Measurement Branch
HMRC	Her Majesty's Revenue and Customs
IP	Institute of Petroleum
ISO	International Standards Organisation
KAMS [®]	KELTON [®] Audit Management System
KCCL [®]	KELTON [®] Common Calculation Library
KIMS [®]	KELTON [®] Instrument Management System
KITS [®]	KELTON [®] Interactive Training Software
K-LOG [®]	KELTON [®] Logging system
K-TRAC [®]	KELTON [®] Tracking System for monitoring day-to-day performance of proving systems
K-VIEW [®]	KELTON [®] monitoring system for output of differential pressure transmitters during calibration
LLE	Liquid – Liquid Equilibrium
LPG	Liquid Petroleum Gas
NGL	Natural Gas Liquid
PPDS	Physical Property Data Service

SLE	Solid – Liquid Equilibrium
UKCS	United Kingdom Continental Shelf
VLE	Vapour – Liquid Equilibrium
VLLE	Vapour – Liquid – Liquid Equilibrium

List of Tables

Table 5.1 XML Technologies	46
Table 5.2 Web Technologies	47
Table 5.3 Web Languages	48
Table 5.4 Web Servers	49
Table 5.5 Protocols	49
Table 5.6 Standards	50
Table 5.7 Other	51
Table 10.1 FLOCALC [®] Test Procedure	140

List of Figures

Figure 2.1 Liquefied Gases	13
Figure 2.2 UK Pipelines	18
Figure 4.1 Structure of Online Survey	34
Figure 4.2 ASP using FES object.....	35
Figure 4.3 Survey Responses to Questions 1-3.....	38
Figure 4.4 Survey Responses to Question 4.....	38
Figure 5.1 Structure of Layers within Web Applications.....	45
Figure 5.2 HTML Form.....	54
Figure 5.3 'Unit Convertor' Window in FLOCALC®	55
Figure 5.4 JavaScript to validate and submit HTML Form	56
Figure 5.5 HTTP request/response Protocol.....	60
Figure 5.6 HTML URL Request.....	61
Figure 5.7 Communication Sub Layers.....	63
Figure 5.8 JavaScript call to ASP.....	63
Figure 5.9 FES default ASP	64
Figure 5.10 FES dll code to process request	65
Figure 5.11 BES default ASP	66
Figure 5.12 BES dll code to pass back response object	66
Figure 6.1 Structure of Standalone Layers.....	72
Figure 6.2 UI Logic Graphical Interface	74
Figure 6.3 MDI form containing a single object.....	75
Figure 6.4 MDI form containing a collection.....	77
Figure 6.5 Handle closing event of a single object.....	78

Figure 6.6 Handle closing event of an object in a collection.....	80
Figure 6.7 Termination procedure for closing the UI Logic.....	82
Figure 6.8 User control initialise, terminate and property code	87
Figure 6.9 Unit Converter Interface.....	89
Figure 6.10 Calculation List Interface.....	90
Figure 6.11 Calculation Header Interface.....	91
Figure 6.12 Calculation initialisation code	92
Figure 6.13 Calculation property code	92
Figure 6.14 Setting a workbook object in a calculation user control.....	93
Figure 6.15 Structure of MVC Pattern.....	95
Figure 7.1 KELTON® Online Activation.....	97
Figure 7.2 License Information.....	98
Figure 7.3 FLOCALC® Calculation Interface	100
Figure 7.4 n-Tier FLOCALC® Software Architecture Model	105
Figure 7.5 n-Tier KITS® Software Architecture Model.....	109
Figure 8.1 XML document.....	114
Figure 8.2 XSD document.....	116
Figure 8.3 DTD Document	116
Figure 8.4 XML License details.....	117
Figure 8.5 XSL document	118
Figure 8.6 Transfer of XML using HTTP	121
Figure 9.1 Server Communication with Core Data.....	125
Figure 9.2 Basic UML Diagram.....	126
Figure 9.3 Top Level Object Model	127
Figure 9.4 FLOCALC Calculation Workbook Object Model.....	129

Figure 9.5 KITS [®] Libraries Object Model	131
---	-----

Contents

Abstract	iii
List of Abbreviations	v
List of Tables	ix
List of Figures	x
1 Introduction	1
1.1 Background	1
1.2 Purpose of Analysis	4
1.3 Main Project Objectives	5
1.4 Project Plan	5
1.5 Review of Previous Work.....	7
1.6 Thesis Outline.....	8
2 Background to the Flow Metering Industry.....	12
2.1 Introduction	12
2.2 UK Oil & Gas	12
2.3 Authorities, Regulations & Standards	14
2.3.1 Fiscal Metering Requirement.....	14
2.3.2 Gas and Oil Measurement Branch (DTI – GOMB)	15
2.3.3 Operator Standards and Codes of Practice	15
2.4 Export Pipeline Systems	16
2.4.1 Typical Oil Export Pipeline Systems	16
2.4.2 Pipeline Transportation Agreements	18
2.4.3 Role of Pipeline Operator	19
2.5 Requirements for Software	20

3	Software Needs of the Flow Metering Industry	21
3.1	Introduction.....	21
3.2	Existing Software	21
3.2.1	FLOCALC®	22
3.2.2	K-TRAC®	23
3.2.3	KAMS®	24
3.2.4	KITS®	24
3.2.5	KIMS®	25
3.2.6	K-LOG®	26
3.2.7	K-VIEW®	26
3.2.8	FM ² P®	27
3.2.9	PPDS.....	27
3.3	KELTON® Infrastructure	28
3.4	Deployment Methods.....	29
4	Industry Survey and Scope of Work.....	30
4.1	Introductions	30
4.2	Background	30
4.2.1	Introduction.....	30
4.2.2	Method	31
4.2.3	Delivery	32
4.2.4	System Architecture	33
4.3	Survey Content.....	35
4.4	Survey Findings.....	37
4.4.1	Introduction.....	37
4.4.2	Overall Responses	37

4.4.3	Software Tools.....	39
4.4.4	Analysis of Software Network Issues.....	40
4.4.5	Software Functionality	40
4.4.6	Software Delivery	41
4.5	Industry Requirements.....	41
4.6	Conclusions	42
5	Overview of Technologies used in Web Applications.....	43
5.1	Introduction.....	43
5.2	General Requirements.....	43
5.3	Overview of Web Applications	44
5.3.1	Web Layers	44
5.4	Overview of Technologies	45
5.5	GUI Layer	52
5.5.1	Display Of Web Pages	52
5.5.2	DHTML	55
5.5.3	Display of Files	57
5.5.4	JavaScript.....	58
5.5.5	AJAX	58
5.6	Communication Layer.....	59
5.6.1	HTTP	59
5.6.2	ASP Component.....	60
5.6.3	XML Data Handling	62
5.6.4	Communication Sub Layers.....	63
5.7	Middleware Layer	67
5.7.1	IIS	67

5.7.2	COM Objects	68
6	Overview of Technologies used In Standalone Applications.....	69
6.1	Introduction.....	69
6.2	General Requirements.....	69
6.3	Overview of Standalone Applications	70
6.3.1	Standalone Layers.....	71
6.4	UI Logic	72
6.4.1	Interface	73
6.4.2	Interaction with the UI Presentation Layer.....	74
6.4.3	Presentation Interface Objects	75
6.4.4	Handling Object Events	78
6.4.5	Common Functionality.....	83
6.5	UI Presentation Layer.....	84
6.5.1	User Control	85
6.5.2	Components	88
6.5.3	Interfaces.....	88
6.5.4	Limiting Access.....	91
6.6	Business Logic.....	93
6.7	Model View Controller.....	94
7	Software Design - Software Specifications	96
7.1	Introduction.....	96
7.2	Common Requirements.....	96
7.2.1	Licensing Requirements	97
7.2.2	Standalone Interfaces.....	98
7.2.3	Web Interfaces	99

7.3	Flow Metering Calculation Specifications (FLOCALC®)	99
7.3.1	Main Requirements	99
7.3.2	Interface Requirements	101
7.3.3	Calculation Requirements.....	102
7.3.4	Design Specification	102
7.3.5	Licensing Requirements	105
7.4	KELTON® Interactive Training (KITS®)	106
7.4.1	Main Requirements	106
7.4.2	Interface Requirements	107
7.4.3	KITS® Library	108
7.4.4	Design Specification	108
7.4.5	Licensing Requirements	109
8	Software Design - Data Transfer between Interfaces	111
8.1	Introduction	111
8.2	Data Store	111
8.2.1	Software Databases	111
8.2.2	Alternatives to Database Software	112
8.2.3	Web Based Data Transfer Considerations	112
8.3	XML	113
8.3.1	XML Data.....	113
8.3.2	DOMDocument versus SAX	114
8.3.3	Schema	115
8.3.4	Transform	117
8.3.5	Off The Shelf XML Software	119
8.3.6	XML within the Project.....	120

8.4	Additional Data Types Used	122
8.4.1	KITS® Data File	122
8.4.2	KITS® Library	122
8.4.3	KCCL	123
9	Software Design - Object Model Requirements	124
9.1	Introduction	124
9.2	n-Tier Architecture	124
9.3	Object Model	125
9.3.1	UML Diagram	126
9.3.2	Top Level Application Object Model	126
9.4	Core	128
9.4.1	FLOCALC®	128
9.4.2	KITS®	130
10	Conclusions and Future Work	133
10.1	Introduction	133
10.2	Market Survey	133
10.3	Web Based Applications	133
10.3.1	Benefits from Web Application Development	135
10.3.2	i.FLOCALC®	135
10.3.3	i.KITS®	135
10.4	Standalone Applications	136
10.4.1	Benefits from Standalone Application Development	136
10.4.2	FLOCALC®	136
10.4.3	KITS®	137
10.5	Other Key Findings	137

10.5.1	Common Code	137
10.5.2	AJAX	138
10.5.3	Testing.....	138
10.6	Benefits to Industry - FLOCALC® and i.FLOCALC®	141
10.7	Future Work.....	142
10.7.1	FLOCALC®	142
10.7.2	KITS®	143
10.7.3	KAMS®	144
10.7.4	KIMS®	144
10.7.5	AJAX	144
10.8	Summary	145
10.8.1	Flow Calculation Software	145
10.8.2	Training Software	145
Appendix A – Market Survey Report.....		147
Appendix B – Module List using AJAX.....		170
Appendix C – Calculation List		179
Appendix D – KELTON® License Agreement.....		180
Bibliography		185
References.....		186

1 Introduction

1.1 Background

Flow meters are used in all major sectors of UK industry, including Offshore Oil & Gas, Petrochemical, Water and Food & Drink.

This thesis mainly considers the Oil & Gas sector, where the primary flow measurement requirements are for fiscal (single- phase crude oil or natural gas) applications, custody transfer (between seller and buyer) and allocation (where pipelines are shared between different operators).

Changes taking place in industry have meant that efficient sharing of resources, with flow metering playing a key part, has become more prevalent in recent years, as discussed in the following journal extract [1] which presents a brief background to issues relating to metering of hydrocarbons: -

“It has become normal practice to share transportation systems and, as a consequence, we need to focus much more clearly on the allocation of hydrocarbon products, both in quantity and quality.

Operators want to develop fields with as little as 7 million barrels of recoverable reserves. They want them to be profitable and, in order to ensure our survival as a net producer, the UK needs to encourage such developments. Reducing the cost of designing, building and operating metering/allocation systems can make a significant contribution to the process.”

Owing to the very high intrinsic value of the metered product, there are very strict fiscal standards for measurement accuracy. The Department for Trade and Industry (DTI)'s [10] Offshore Regulator requires that all metering

installations should be built and operated to agreed specifications, since meter readings are the basis on which tax is determined. (Chapter 2 provides more background information on the flow metering industry.)

Depending on the type of meter and product (oil or gas) there are industry standards such as International Standards Organisation (ISO) [2] which are used to calculate outputs. These involve very complex calculations which would be inefficient to perform manually, hence, it is necessary to find methods that are fast, reliable and meet industry standards, i.e., utilising software.

Historically, software in the flow metering industry has been developed for standalone computers. As part of this research, new solutions have been investigated arising out of advances in technology such as increased hardware capacity and the expansion of the World Wide Web (WWW). The research has led to solutions that are both portable (data can be accessed by different software) and accessible in any location.

As part of the project, an online market survey was developed and used to investigate industry needs. This included investigating the use of standalone versus web technologies for the delivery of flow measurement software.

As a result of the findings a flow metering calculation software (FLOCALC[®]) package, was redeveloped, and i.FLOCALC[®] was developed as an intranet application. The redevelopment of the KELTON[®] Interactive Training

Software (KITS[®]), a flow metering training package for flow metering professionals, and i.KITS[®], the corresponding intranet application, was commenced.

Methods for storing data were investigated to ensure portability between different software packages. For FLOCALC[®] there was a need to store calculations and calculation sets. Within KITS[®] there was a requirement to store training libraries, training records and user information.

Because of the necessity to move data between users and software applications, technologies that would check and validate the data were identified and researched. This was done so that data that had been tampered with or corrupted, and therefore could prevent the application from performing as expected, could be rejected by the software.

Methods to display data within different user interfaces such as on web pages and within standalone programs were also investigated.

A major feature of the work was the adaptation of an n-tier software architecture [3] to ensure flexibility between the object models, business logic and interfaces. (See section 9.2 for more details.)

1.2 Purpose of Analysis

Flow meters used in the Oil & Gas industry must perform to the very fine tolerances specified by DTI [10], the industry regulator. This ensures the accurate evaluation of product quantity and quality when selling.

Software is used to ensure flow calculations are performed correctly, audits are accurately recorded without loss of data and logging of information is accurately recorded. (Section 3.2 describes software currently used in industry.)

The expansion of the WWW has prompted the development of software using web technology alongside standalone solutions. The use of web technology increases access to data both on-site as well as remotely in an office environment.

With much functionality in common between different applications, new methods were reviewed to streamline the development of products that would allow the reuse of existing code. A spin-off benefit from this was reduced testing due to the reuse of previously tested components.

As multiple user interfaces were to be developed (web and standalone) for multiple packages such as FLOCALC[®] and KITS[®], common code was created and used for displaying data. This gave the interfaces a consistent look and feel and also meant that future changes to the corporate image could be confined to common components.

In order to satisfy the requirements discussed above, formal software documentation was drafted which was complete, clear and in-line with findings from the analysis of industry needs.

1.3 Main Project Objectives

The following is a summary of the main objectives of this research project:

1. To conduct a critical analysis of industry needs in terms of flow measurement software leading to a proposal for the re-development of KELTON's existing software packages.
2. To research and evaluate different software tools to allow the integration of improved flow measurement software and to facilitate delivery of the software across the WWW.
3. To define and develop a software architecture which exploits the technology and techniques identified at objective 2.
4. To develop a fully integrated range of product incorporating state-of-the-art flow measurement software into a common system architecture.
5. To make the software accessible to industry and with flexible data interfaces for input and output that can easily and/or automatically be adapted to individual client preferences.

1.4 Project Plan

The project was conducted in the context of KTP (Knowledge Transfer Partnership) programme 4243. Academic supervision was provided by TUV NEL (the academic partner). The work was undertaken on the premises of

KELTON® (the commercial partner) who provided day-to-day project management supervision. The collaboration achieved the joint aims of facilitating knowledge transfer between the two organisations and enhancing market performance of the commercial partner.

The work carried out during this project is summarised below:

1. An online market survey program was created and used to issue a questionnaire to the flow metering industry in order to obtain feedback on current software needs. The program was developed using an n-tier [3] software architecture and Extensible Markup Language (XML) [19] technologies as a test bed for future projects.
2. Responses to the market survey were analysed to investigate software demands for flow metering tools, e.g., platforms for hosting the software such as internet and intranet (see section 3.3).
3. A formal specification of the software was prepared based on the investigation. Software documentation was created which mirrored the results of the investigation. The documentation was written to define the user requirements and software functionality. Software tools and technologies for handling data in multiple interfaces using a common data format were also reviewed (see Chapter 7).
4. Software was developed using an architecture that was flexible and could be easily changed. The design and code of the software was developed against the software documentation created. Licensing methods were also investigated to protect copyright while ensuring ease and flexibility for users (see Chapter 7).

Two unique software products and various user interfaces were redeveloped during the course of the project. The software products redeveloped were:

- FLOCALC[®] (a flow metering calculation package), using KELTON[®] Common Calculation Library (KCCL) provided by KELTON[®] to run on an intranet, standalone Personal Computer (PC) and as an MS Excel Add-In.
- KITS[®] (KELTON[®] Interactive Training Software) to run on a standalone PC and on the Intranet. The KITS[®] redevelopment was commenced during the project and is ongoing.

1.5 Review of Previous Work

KELTON[®] previously deployed software locally on standalone PCs for the flow metering industry (see section 3.2). Each software package had been created independently but using shared components where appropriate, e.g., for licensing and error handling.

Earlier versions of software were created as complete programs and not separated into smaller components. This meant that the same code often had to be written twice, such as in the case of FLOCALC[®] and KELTON[®] Instrument Management System (KIMS[®]), where both applications use the same flow metering calculations. The User Interface's (UI) were also embedded within the software applications resulting in further duplication within each software package.

As all previous versions were created as standalone applications, there was a requirement to create new common interfaces for web applications to give them a standard look and feel. This was achieved using a common Cascading Style Sheets (CSS) [4] and Extensible Style sheet Language (XSL) [5] which can be easily adapted and stored in one location to ensure all web based software looked the same.

1.6 Thesis Outline

Chapter 2 provides more general background information on the metering industry, with an emphasis on the UK Oil and Gas sector. This chapter reviews the standards required for pipeline systems and the requirements between pipeline operators and producers. It also looks at the authorities, regulations and standards in the oil industry.

Chapter 3 provides an appraisal of current software tools available to the flow metering industry against possible software alternatives. Details are also provided of KELTON's infrastructure and deployment methods.

Chapter 4 describes the results of the Market Survey covering five software applications and three deployment methods. The survey results were used along with in-house expert knowledge at KELTON® to assess the software products that would most benefit from redevelopment. The chapter also discusses the strengths and weaknesses of options available such as using online services in remote locations and compares strengths and weaknesses of standalone software versus web applications.

Chapter 5 reviews and appraises current technologies employed to develop web applications. This chapter describes the layers required for communication between the user and the application. This includes a review of Hypertext Markup Language (HTML) [24] and scripting, security requirements and methods for handling and presenting data.

Chapter 6 reviews and appraises current technologies employed to develop standalone applications. This chapter describes the layers required for communication between the user and the application. This includes a review of UI logic required for creating user interfaces, security requirements and methods for handling and presenting data.

Chapter 7, 8 and 9 describes the software design.

Chapter 7 describes the specifications of each software package. These include the specification for web and standalone versions. The chapter also looks at the user requirements, functionality and implementation documentation. The implementation documentation describes all the object models and the n-tier [3] software architecture of the software.

This chapter also describes the requirements for developing common programs to store the business logic of the individual products, which is included in the n-tier architecture and licensing requirements for each software package.

Chapter 8 describes requirements of data management within applications. Due to the development of multiple interfaces for software applications, such as web and standalone, there was a need to assess data transfer requirements for importing and exporting data between different applications, and to consider storage of data for later use.

This chapter looks at the uses of XML [19] and the data structures created for easy transfer of data. It also reviews the different forms of data representation, e.g. using XSL transforms [5] for representing data in web applications and the Document Object Model (DOM) [6] used to create and translate data into a format used by applications and the use of XML Schema Definition (XSD) [7] for validating imported data.

Chapter 8 also discusses the different off-the-shelf software tools that can be used for working with XML technologies.

Chapter 9 describes the object model used to create software for use by multiple applications. The object model was created in such a way as to reduce duplication and to ease data transfer between different user interfaces using an n-tier [3] software architecture.

The public properties and methods for communication between external programs and handling of object pointers are also reviewed in this chapter.

Chapter 10 presents the overall conclusions drawn from the work undertaken and recommends potential future research.

2 Background to the Flow Metering Industry

2.1 Introduction

In this chapter, the metering industry is described and in particular pipeline and allocation systems. There is also a review of the authorities, regulations and standards that apply in the oil industry.

Sections 2.2 to 2.4 summarise KELTON's oil and gas training course which was attended by the author.

2.2 UK Oil & Gas

The UK sector reserves of crude oil and petroleum liquids are 5% of the World's proven reserves compared to 55% in the Middle East. Despite this relatively small percentage the reserves are extremely valuable to the UK Exchequer and beyond, as they are located in a politically stable part of the world, close to major European markets.

Many fields in the North Sea have a high proportion of "associated gas", i.e., gas associated with oil, either as a gas cap above an oil reservoir or dissolved in the oil. The associated gas is liberated during crude oil production and both processed separately and exported as gas in separate gas pipelines, or liquefied under pressure and exported along with the crude oil.

Liquefying a natural gas can produce Liquid Petroleum Gas (LPG) or Natural Gas Liquid (NGL), see Figure 2.1, dependent on the predominant hydrocarbon gasses being liquefied and the process(es) used.

Exporting LPG by injecting into the crude oil poses technical problems. The live crude must be kept and measured at high pressure to prevent it “gassing off”, which would cause difficult conditions for flow measurement. The measurement of LPG streams themselves pose special difficulties as the lubroscity and viscosity of LPG’s is typically very low at very low temperatures.

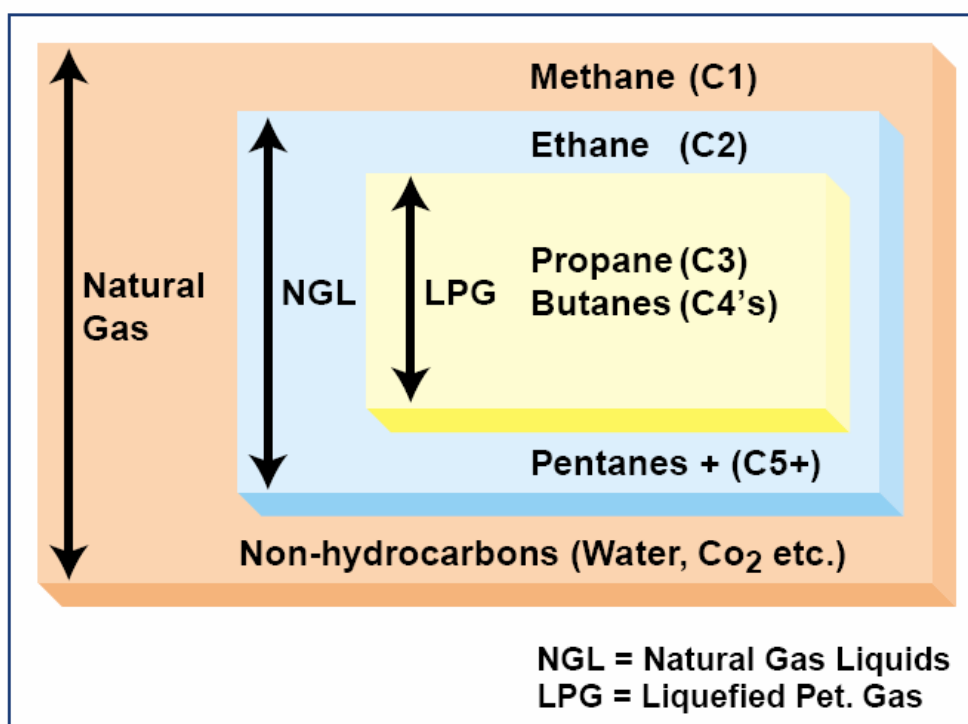


Figure 2.1 Liquefied Gases

“Non associated” gas comes from reservoirs that do not contain crude oil, which are commonly found in the Southern North Sea.

2.3 Authorities, Regulations & Standards

2.3.1 Fiscal Metering Requirement

There is both a commercial and legal requirement to measure quality and quantity of pipeline fluids to work out the value of the oil.

Each operator must comply with the “Petroleum Production Reporting Regulations 1976”. The act places a responsibility to measure hydrocarbons “won and saved” from the facility. The government requires this for two reasons, to levy tax and to estimate the extent of hydrocarbon reserves remaining in the United Kingdom Continental Shelf (UKCS).

In the UK, there are a number of bodies who impose their standards and regulations such as: -

- UK Department for Trade and Industry (DTI), Gas and Oil Measurement Branch (GOMB)
- Her Majesty’s Revenue and Customs (HMRC)
- Pipeline Transportation Agreements and Statement of Requirements for Third Party Metering Stations
- British Standards (BS)
- International Standards Organisation (ISO)
- Institute of Petroleum Standards (IP)
- American Petroleum Institute (API)

2.3.2 Gas and Oil Measurement Branch (DTI – GOMB)

The DTI is responsible for all aspects of the measurement of hydrocarbons won and saved from developments within the UKCS and UK mainland.

They are responsible from the conceptual design stage of a metering system through detailed design and construction and eventually to commissioning and testing. They also maintain an interest throughout the metering system's life and make regular visits especially during major component re-certification.

HMRC have an interest at the point of sale or disposal, however, DTI [10] act as an agent for HMRC for oil and gas systems. HMRC regulations cover aspects of oil and gas measurement.

The DTI GOMB regulates design and operation of hydrocarbon metering stations and specifies the minimum design and operation standards which they expect to be applied.

They also cover design features necessary to ensure security, calibration and maintenance requirements are met.

2.3.3 Operator Standards and Codes of Practice

Engineering standards and codes of practice are issued by the Operators Engineering Group. They are applied to construction, maintenance and

operation of a pipeline system, including the processing facilities and the terminals.

These are mainly covered by the following:

- British Standards (BS)
- International Standards Organisation (ISO)
- Institute of Petroleum (IP).
- American Gas Association (AGA)
- American National Standards Institute (ANSI)

There are also standard units of measurement and reference conditions (ambient temperature and pressure) which are applied to all liquid flowrate calculations. These are:

Metric Conditions

- Pressure 1.01325 bar
- Temperature 15⁰C

Imperial Conditions

- Pressure 30in of Mercury
- Temperature 60⁰F

2.4 Export Pipeline Systems

2.4.1 Typical Oil Export Pipeline Systems

Having found a crude oil deposit, a development plan referred as the “Annex B submission”, needs to be drawn up. The plan needs to be approved by the

DTI. The plan takes into account the type of hydrocarbon deposit, type of subsea strata where the hydrocarbons lie and proximity of other developments.

These factors as well as the project budget largely determine the nature of the development, such as:-

- Number of subsea wellheads
- Number of platforms
- Process export route

Most oil in the North Sea is exported using a pipeline which carries the oil to shore-based processing, storage, tanker loading and/or refining facilities. Some fields still retain offshore tanker loading facilities.

There are four major common carrier oil pipeline systems used to export oil and liquid petroleum products in the North Sea, landing oil at three separate onshore terminals, as shown in Figure 2.2.



Figure 2.2 UK Pipelines

2.4.2 Pipeline Transportation Agreements

There are many instances where third party pipeline liquids are transported in “common carrier” pipelines throughout the North Sea, therefore a “Transportation Agreement” is made between the pipeline operator and each third party to protect all their interests.

Transportation agreements are typically concerned with the following: -

- Quality and quantity of fuel.
- Minimum and maximum quantities an operator is allowed to export in a given time period
- Entry specifications such as water content and pressure.
- Means of calculating the transportation charge or “tariff” to be paid by the producer for use of the pipeline system and a share of the operating costs associated with the processing and export facilities at the terminals.

2.4.3 Role of Pipeline Operator

A carrier pipeline system can be owned by numerous companies and agreement will be reached between them on who will be the “Operator” of the pipeline system.

The operator is responsible for the following:

- Setting up agreements and monitoring the quantity and quality measurements of all processed fluid transported in the pipeline system.
- Setting minimum standards required for design and operation of each metering system located at the inlets and outlet points of the pipeline.
- Operating the pipeline safely and efficiently.
- Hydrocarbon allocation, accounting and billing.
- Periodic monitoring inspections of onshore and offshore systems.
- Initiating audits of newly commissioned metering systems.

2.5 Requirements for Software

Due to the complexities of correctly allocating quantities and quality of fuel for the individual producers, there is a need to use software which can accurately calculate the flow of fuel and record readings from the meters working in the system. The use and recording of the meters must be accurate and this means that industry professionals need to be well trained and aware of the complexities. To ensure the industry is well trained, software such as KITS[®] can be used to test operator competencies.

During the design and build of a meter, there is a need to ensure it is the correct size for the system and is built to industry standards. A flow metering calculation package such as FLOCALC[®] is required to check the readings from the meters are producing results to the accuracy levels specified by the DTI.

A calculation package is also necessary. Suppose, for example a wrong measurement occurs such as the temperature gauge being one degree Celsius out. This would imply that all the results would need to be recalculated by re-entering the corrected temperature as an input into a calculation package. Software such as the FLOCALC[®] Excel Add-In could be utilised as this has the facility to perform multiple calculations.

3 Software Needs of the Flow Metering Industry

3.1 Introduction

In this chapter, software packages available to the flow metering industry at the commencement of this research through KELTON[®] and TUV NEL are reviewed.

All the existing software packages were designed to work on standalone PCs. As part of this project, web based technologies were investigated and used to redevelop the software.

This chapter also presents the KELTON[®] infrastructure.

3.2 Existing Software

This section reviews the following software provided by KELTON[®] [8] based on the versions available at the outset of this work:

- FLOCALC[®] (KELTON[®] Flow Metering Calculation Software)
- K-TRAC[®] (KELTON[®] Tracking System)
- KAMS[®] (KELTON[®] Audit Management System)
- KITS[®] (KELTON[®] Interactive Training Software)
- KIMS[®] (KELTON[®] Instrument Management System)
- K-LOG[®] (KELTON[®] Logging System)
- K-VIEW[®] (KELTON[®] monitoring system for output of differential pressure transmitters during calibration)
- FM2P[®] (Fiscal Metering Management Package)

Physical Property Data Service (PPDS) [9] provided by TUV NEL is also reviewed.

3.2.1 FLOCALC[®]

FLOCALC[®] V2 [8] is a flow metering calculation package used by metering engineers and technicians.

FLOCALC[®] was designed to solve flow measurement calculations and is used in the design of pipeline systems. The software allows users to create and store a set of calculations about a metering system. A metering system calculation package consists of multiple flow calculations using different standards and options within the standard. See Appendix C – Calculation List.

At the outset of this work, the software performed 27 flow metering calculations inline with industry standards such as ISO 5167 [2, 10]. The software was developed to work on standalone PCs and is a single program.

Some of the features of FLOCALC[®] are as follows:

- 3 main categories of calculations - Gas, Oil and Common. (Common calculations are not fuel specific.)
- Inbuilt Unit Converter.
- Release dates of calculations are documented for audit purposes.

- Details about a calculation such as system reference, options and inputs can be saved to a data file containing multiple calculations for the metering system. These can be reopened by the program at a future date.
- Data feed tool allows user to update the inputs from one calculation using the results from a previous calculation, thereby saving time.
- Constant checker allows users to see details of the constant values and units used in a calculation.

3.2.2 K-TRAC[®]

The KELTON[®] Tracking System (K-TRAC[®]) [8] is developed to carry out statistical analysis to determine the accuracy of a flow meter calibration in accordance with the ISO for Crude Oil Flow Measurement.

The measurement of liquid products flowing in a closed conduit is often measured by way of a turbine meter or other pulse generating device. When the value of the product is high, such as in crude oil, it is important to accurately calibrate the meter. The meter is usually calibrated by passing a known volume of fluid through the meter and counting the number of pulses generated by the meter. The result is known as a "meter factor" or "K-Factor".

To confirm the calibration is accurate the factor is compared against previous factors using statistical analysis. By using statistical analysis, all the

parameters that affect the accuracy of the measurement, such as fluid properties and mechanical wear on the meter, can be taken into account.

3.2.3 KAMS[®]

The KELTON[®] Management Audit System (KAMS[®]) [8] is designed to allow an auditor to set up audits and audit criteria for generating reports. KAMS[®] can be used for any type of auditing but is primarily used to audit flow metering systems. Audits ensure meters are performing correctly by being calibrated at agreed intervals and that the correct techniques are being used for measurement and calibration.

KAMS[®] contains a database which allows users to store, manage, sort and review audit information. The software can also be used to find records relating to a piece of equipment and to show common failures or errors of equipment in a metering system.

3.2.4 KITS[®]

KITS[®] [8] is an interactive training package that uses training materials developed by KELTON[®]. It is a training aid to help trainees visualise how a metering system works and can test the competency of workers in the industry.

KITS[®] guides trainees through a training course with the use of text, images and animations. It assesses their level of competency through questions and answers and is used by trainee engineers, technicians and allocation accountants.

The software is divided into the following versions

- **Player Edition**
Designed to work from a CD. The user is given the course to run and to work through.
- **Standard Edition**
Designed for the single user, who logs on and then chooses the required course.
- **Developer Edition**
Designed to be used by an unlimited numbers of trainees. The client buys the modules under license to use and adapt in order to create their own, dedicated flow measurement and analysis courses.

3.2.5 KIMS[®]

KELTON[®] Instrument Management System (KIMS[®]) [8] is used for recording, verifying and saving calibration data of instruments. Instruments need to be calibrated at different intervals depending on the type and age of the instrument to ensure the instrument is giving accurate readings. It can be set to let users know when a calibration of an instrument is due by pre-setting the timing and frequency of each calibration. KIMS[®] also keeps records of the calibration, enabling users to look at the history of the instrument.

KIMS[®] uses KCCL, which is a collection of flow metering calculations for use in KELTON's software applications.

3.2.6 K-LOG®

The KELTON® Logging System (K-LOG®) [8] is designed to allow users to store information about events in a consistent, and user definable format. This way each entry not only includes a reference, a date and time stamp of when it happened, and when and who it was recorded by, but it also always includes all relevant/required information.

It is a legal requirement to keep a logbook of all events that might affect a fiscal metering system, such as stream on/offline events, calibrations and change outs of any meters and transmitters, or the changing of any system variable which would affect the recorded production total. All data is stored and can be archived to ensure nothing is lost. This is required by the DTI [10] to ensure an adequate audit trail for their auditors to check that the metering system is being fully compliant with the DTI regulations and reporting its production figures correctly. It also helps in case of a mis-measurement.

3.2.7 K-VIEW®

The KELTON® monitoring system (K-VIEW®) [8] is designed for viewing and optionally recording the output of any analogue or digital transmitter such as a Differential Pressure (DP) transmitter during calibration. It records the output in real time and displays the results graphically. This means that signals which are subject to continuous periodic fluctuation even when 'stable' can be averaged accurately. This is particularly of use when calibrating a DP Transmitter with a dead-weight tester. It is also much easier

to tell when the signal has stabilised and so reduces errors and can reduce the overall calibration time.

3.2.8 *FM²P[®]*

A Fiscal Metering Management Package FM2P[®] [8] provides a single logon and integration between all of the KELTON[®] software packages. It also links the packages to the Metering Supervisory Computer to provide a complete Fiscal Metering Management Package.

3.2.9 *PPDS*

PPDS [9] is a software package for generating numerical data describing the thermo-physical properties (thermodynamic and transport) of fluids and fluid mixtures over a wide range of temperatures and pressures.

PPDS is used by engineers and scientists. The software has access to a database of chemical properties. It has the capability of returning data on 42 constant properties and 20 temperature and pressure dependent properties of fluids (liquid and gases) and mixtures.

PPDS is designed to answer three fundamental questions on fluid streams based on a range of industry standard models and calculation types.

1. What phases are present and are stable?
 - Vapour – Liquid Equilibrium (VLE)
 - Liquid – Liquid Equilibrium (LLE)
 - Vapour – Liquid – Liquid Equilibrium (VLLE)
 - Solid – Liquid Equilibrium (SLE)

2. What are the concentrations of each component in each stable phase?
3. What are the thermodynamic and transport properties of each stable phase?

3.3 KELTON[®] Infrastructure

The established IT infrastructure at KELTON[®] needed to be taken into account when further developing software to avoid compatibility issues with external software and to ensure development issues were reduced wherever possible.

The main considerations were as follows: -

1. KELTON[®] use Microsoft Windows 2003 server.
2. Data is encrypted using CAPICOM [11] due to compatibility with MS Windows 2003 servers.
3. KELTON[®] use Microsoft Visual Basic 6.0 to write all their common code, reference Dynamic-Link Library's (dll) [12] and Object Linking and Embedding (OLE) [48] Custom Control (OCX) components. This has resulted in a Component Object Model (COM) [13] software architecture structure being used.
4. A common component used in KELTON[®] software is KSync. This component is used to import/export and archive data in an encrypted file format.

3.4 Deployment Methods

The software marketed by KELTON® at the commencement of this research was designed to work on standalone PCs and was tailored to industrial customer requirements.

The range of software tools increased over time from the original software developed as KITS® followed by FLOCALC®, KIMS®, K-TRAC®, KAMS®, K-LOG®, K-VIEW®, and finally FM2P®. The products were developed independently without code reuse with the exception of FM2P which was developed to integrate all the packages.

With the expansion and accessibility of the WWW, it was necessary to respond to industry demands for software to be deployed on the web.

This led to an investigation of whether to reuse existing code to develop web applications or to completely rewrite software for both standalone and web applications. It was also important to consider the future impact changes would have on the software. Although rewriting software would be more time consuming and expensive in the short term, if developed correctly using an n-tier [3] software architecture, new software products and functionality could be added more easily in the future as opposed to using existing code that would be less adaptable to change.

4 Industry Survey and Scope of Work

4.1 Introductions

This chapter describes the work done to identify industry demands via a Market Survey. The responses are analysed and based on the survey findings and discussions with KELTON® management, the focus of this project in terms of software development is presented.

4.2 Background

4.2.1 Introduction

A Market Survey was carried out to determine the software demands of the flow metering industry.

It was necessary to decide what questions needed to be answered, how to conduct the survey and also who to target in order to get as clear a picture as possible of the industry requirements. The industry includes flow metering engineers, technicians and allocation accountants, all of whom have a requirement to produce accurate information on quantity and quality of fuel passing through a pipeline.

The day to day users are technicians who record readings made from the meters. Engineers are responsible for ensuring meters are performing accurately and that the metering systems are calibrated within industry standards. Allocation accountants need accurate readings to calculate the

value of the fuel flowing through a pipeline for both selling and for taxation purposes.

As part of the process, questions needed to be prepared and be brief to ensure a maximum of responses. The key questions were on flow software needed and on deployment such as standalone or web. The questions asked in the Market Survey were agreed through discussion with KELTON® management.

A report was prepared showing the findings from the survey.

4.2.2 Method

Several options were considered for gathering information on industry demands. Information could be solicited via a postal survey, online survey or by cold questioning. The industry is global but relatively small so cold questioning was ruled out due to the difficulty of contacting the target audience. It was considered more likely that responses would be returned online than by post as it would be less time-consuming for busy professionals.

However, the online method posed other problems such as how to inform people of the existence of the online survey and how to motivate them to participate. The survey was targeted at users registered on the KELTON® database. An email containing a link to the survey was delivered with an explanation of the aims. This posed a second problem due to laws on spam

emails [14] as there were 1,729 addresses on the distribution list. The emails were carefully worded to notify people that this was a survey and allow them to choose not to open the email if they so wished.

Finally, it was essential that the personal data (i.e., email addresses of respondents) were held securely and stored for no longer than necessary to create the report, as required under the Data Protection Act [15].

4.2.3 Delivery

An online Market Survey application was developed as part of this project and used to gather responses to the prepared questions.

The questions were presented in Dynamic Hypertext Markup Language (DHTML) [16] format so that JavaScript [17] could be included to check user's responses were valid before submitting the HTML form [18]. The respondent also received an HTML [24] page with a summary of their responses for confirmation.

The results were stored in a secure location, not accessible from outside the organisation but with easy access for KELTON[®] to analyse by storing the data in a Back End Server (BES).

The Market Survey software was also designed so that KELTON[®] could use it in potential future surveys.

The online survey program was developed using an n-tier architecture [3] in order to separate processing and storage of data from the client application. XML [19] technologies were used for storing and presenting data.

This approach was new to KELTON® and used as a test bed as it was envisaged that the main project (see Chapter 7) would be developed in a similar manner. Important lessons were learned and experience gained to give an indication of the skills required. This approach was ultimately adopted for the remaining project applications.

The survey responses were stored in XML in order to ease delivery between the communication layer and the Graphical User Interface (GUI) layer (see section 5.2) in a web program. XML is an industry standard for storing data. It is ideally suited for web applications due to the fact that the data can be separated from the schema (used to validate the data) and representation (web page) making it more portable between different web layers.

4.2.4 System Architecture

The user responded to questions on the survey web page. A server program for handling the user responses and for transforming XML data into a web page was developed.

The user has access to the Front End Server (FES) program. This program does not process or store data, it transforms data using stylesheets [4], [5] into HTML pages.

A BES program was developed to handle data and deal with requests. An illustration of how this works is shown in Figure 4.1.

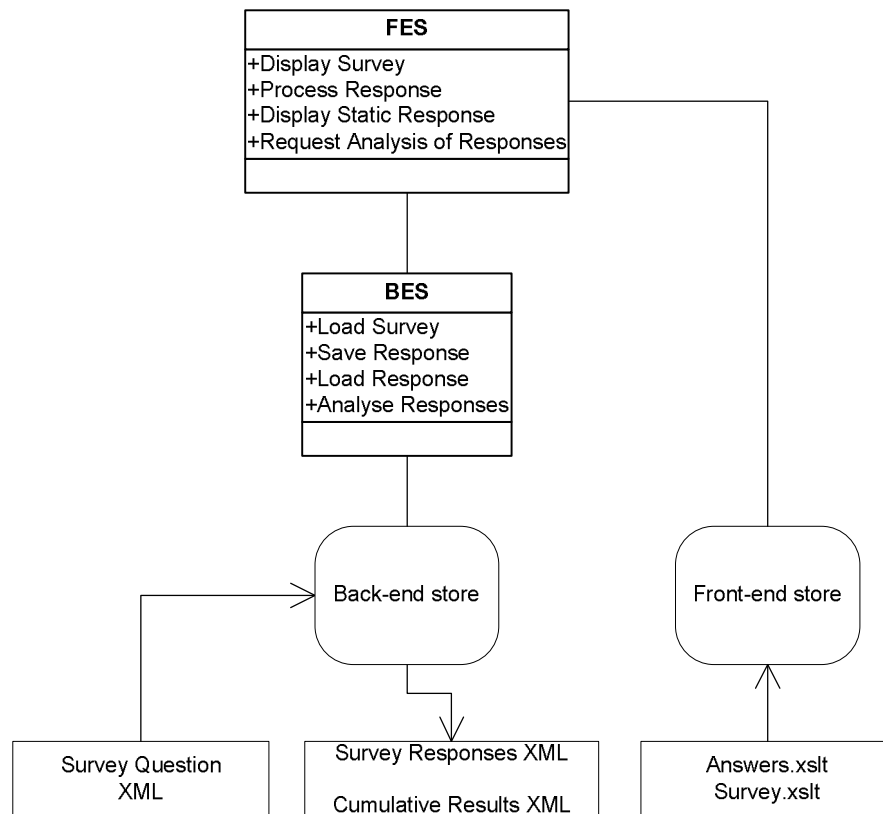


Figure 4.1 Structure of Online Survey

The FES and BES programs were initially developed in an Active Server Page (ASP) [20] which is an HTML page that contains embedded script and allows for the creation of interactive sessions with the user. However in ASP, error handling was found to be awkward and suffers from a lack of data types which can lead to unnecessarily complicated software design. The initial design was improved by transferring the majority of the code to a Visual Basic program so that error handling could be more efficiently implemented. This also allowed for the use of data typing of variables.

An ASP was still required for the HTML to communicate with the dll [12] but it was redesigned to pass the application, session, server, request, and response objects to the FES and BES objects as shown in Figure 4.2.

```
<% Tag required to show script is inside
Get the FES object from the KMSP_FES program (dll)
  dim oWC: set oWC=server.createObject("KMSP_FES.FES")
Pass the Application, Session, Server, Request and Response
objects to the public method - Units
  oWC.ProcessResonse
  Application,Session,Server,Request,Response
  set owc = nothing
%>
```

Figure 4.2 ASP using FES object

The communication between the HTML documents, FES dll and BES dll used ASP pages to GET/POST data and receive/read responses.

4.3 Survey Content

The software tools investigated were products already used in the flow metering industry. The survey gathered information from engineers and technicians currently using these tools and calculated the numbers of users that would be interested if the tools were made available in different formats. The numbers interested in software was compared against sales to determine the software which would be most valuable to industry. This information was used to prioritise software development.

The survey questions were carefully chosen in consultation with KELTON[®] management to elicit as many responses as possible. The questions are summarised below (see Appendix A – Market Survey Report for full question set).

1. Which of the following software tools do you have or would like to have?
2. Do you or your company have a business need for a fully functional intranet, extranet or internet based version of these tools?
3. Do you or your company have a business need for management functions via an intranet, extranet or internet, i.e. reporting and data analysis?
4. Where would you envisage the software residing?

The software tools which featured in the Market Survey (Questions 1-3) were as follows (the relevant KELTON[®] software package is shown in brackets): -

- Instrument Management & Flow Computer Validation (KIMS[®])
- Turbine Meter Performance Monitoring (K-TRAC[®])
- Interactive Training (KITS[®])
- Electronic Logbook (K-LOG[®])
- Engineering Flow Calculations (FLOCALC[®])
- Audit Management System (KAMS[®])

The methods for deployment proposed by the survey (Question 4) were as follows: -

- Local (Standalone)
- Internet
- Intranet

- Unknown

The final part of the survey provided users with the opportunity to make any additional comments.

4.4 Survey Findings

4.4.1 Introduction

This section presents analysis and discussion of the responses received to the survey. Full details are available in Appendix A – Market Survey Report.

4.4.2 Overall Responses

A total of 49 responses were received to the Market Survey out of 1729 emails sent giving a 3% return. This was considered an adequate response rate for this method of survey and the sample was sufficient for the analysis.

A plot of the responses to Questions 1 to 3 is presented in Figure 4.3. The responses to Question 4 are plotted in Figure 4.4.

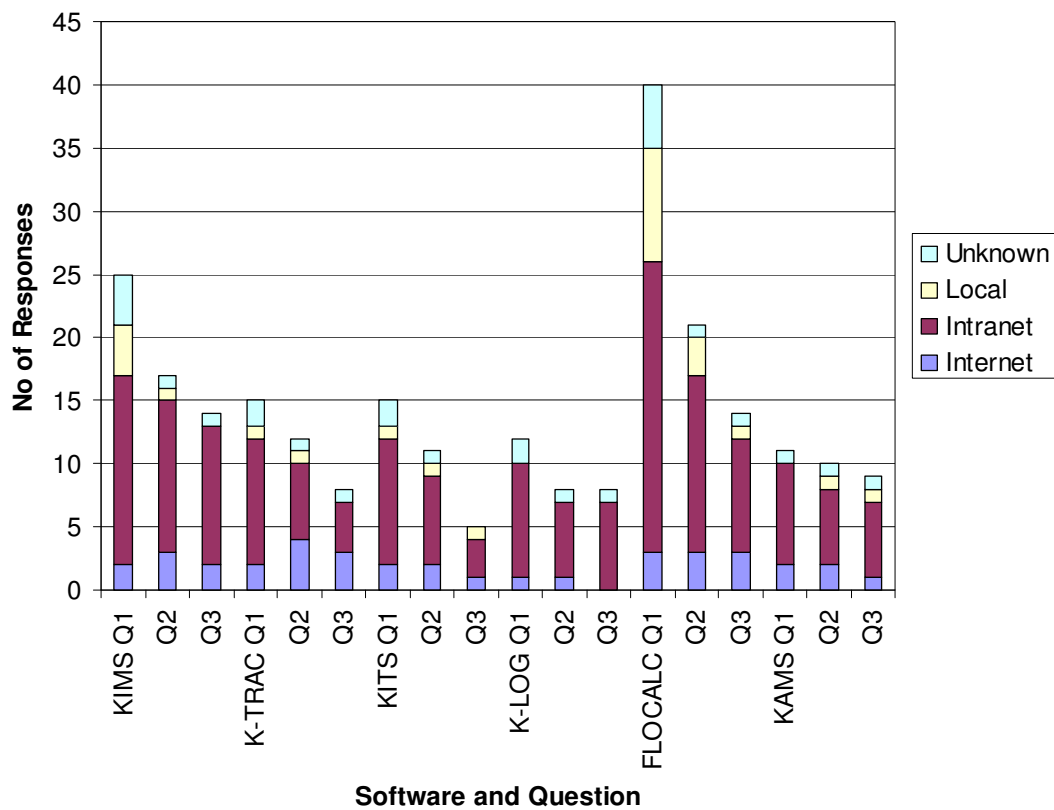


Figure 4.3 Survey Responses to Questions 1-3

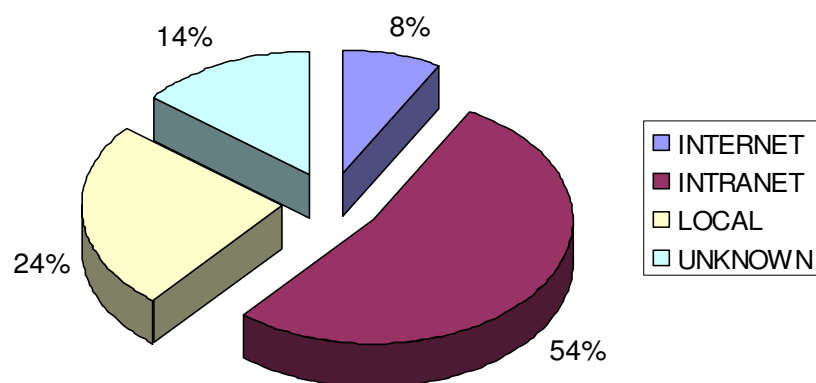


Figure 4.4 Survey Responses to Question 4

More in-depth analysis based on the survey findings and meetings with KELTON[®] management to discuss the commercial considerations associated with the project are presented in the following subsections.

4.4.3 Software Tools

From the survey it was found that FLOCALC[®] and KIMS[®] were the most widely used of KELTON's existing standalone packages. This conclusion was drawn from the number of positive responses and the number of current users.

Commercial considerations were also taken into account when reviewing the results, especially in the case of KITS[®]. KITS[®] is regularly used by KELTON[®] to provide in-house training courses for technicians, trainee engineers and allocation accountants within the oil and gas industry. There is little demand by outside users to buy the software and license modules, but there is a greater demand to attend courses at KELTON[®]. KITS[®] was identified as a high priority due to its value to KELTON[®] but not necessarily as a package for external sales. KELTON[®] also felt that deployment of a web based package would be of value to them allowing multiple users access to modules when attending in-house training courses.

Also highlighted in the current KITS[®] software was the awkwardness to alter or add new modules to the library database for users. There was also a problem with duplication of training materials due to the naming conversions required of files used in the library. A new development would address this

problem and therefore make updates and additions to the KITS[®] library easier. This would mean that an increased number of training courses could be made available faster as new metering technology becomes available and that training materials such as text and animations could be shared more easily between modules.

4.4.4 Analysis of Software Network Issues

K-TRAC[®], K-LOG[®] and KIMS[®] are commonly used at remote field locations for tracking instrumentation. At these locations there is often no network connection. The instrumentation must be “instrument secure” which imposes restrictions on the use of technology that can cause a spark and create a combustion hazard. For example wireless connections are banned from use in such locations. Due to the problems of connecting to a network, it was decided that these packages would not currently be priorities for redevelopment as web applications.

4.4.5 Software Functionality

Fully functional software as opposed to software with management functions only via the web was appraised. The management functions are to allow a general overview of the data rather than access to the primary functionality, and for using software reporting mechanisms. For example, within KIMS[®] (see section 3.2.5) the management functions allow the user to see scheduling of calibrations and information on trends in instrument readings but do not allow a user to record instrument calibrations.

From the survey, FLOCALC[®] and KIMS[®] fully functional software was found to have the most value to industry.

Management functionality only via the web was appraised and found to be of less interest.

4.4.6 Software Delivery

From the analysis of the survey it was concluded that respondents would prefer an intranet solution rather than internet and standalone options. As all the software was deployed already as a standalone version, it was clear that new techniques for the delivery of software as web based applications would need to be investigated.

4.5 Industry Requirements

The project concentrated on creating increased accessibility for a calculation package (FLOCALC[®]) and an interactive training package (KITS[®]). The software was to be developed to use common functionality within both web and standalone versions.

Based on the Market Survey, 82% of respondents had or would like to have a calculation package (FLOCALC[®]). 51% of respondents had or would like to have an instrument management and flow computer validation (KIMS[®]) but it was decided not to pursue this. See section 4.4.4 for discussion on hazards with wireless connections.

4.6 Conclusions

Based on a combination of the Market Survey findings, commercial considerations of KELTON[®] management and instrument safety requirements, it was concluded that FLOCALC[®] and KITS[®] should be redeveloped within this project to work on both standalone PCs and as web applications.

5 Overview of Technologies used in Web Applications

5.1 Introduction

Based on the market research findings, one of the key objectives of this project was to redevelop the KELTON® software to run via the web.

This chapter reviews technologies for communication in web applications and their potential uses in web packages created for the flow metering industry.

5.2 General Requirements

All applications must have a common look and feel with a corporate image for KELTON®. The requirements for the web applications are summarised below:

- Where possible all common code should be developed so it can be reused by other applications and interfaces.
- Data must be held securely on a server and not be accessible to users by any means other than through the application interfaces.
- Users need to be able to interact with the applications and get accurate responses which are displayed so as to be easy to read. Also data that is imported should be checked for tampering and/or corruption.
- Files such as RTF's within the training application need to be displayed.

5.3 Overview of Web Applications

Web applications can be developed using thin interfaces where HTML [24] is generated and reloaded whenever the user makes a request but these can suffer from poor functionality.

They can also be developed with rich interfaces using applets [21, 22] but these can have problems as the developer does not always know what client is accessing the application and they can be slower.

In building web applications there were various technologies to consider. After the success in developing the Market Survey program, an n-tier [3] software architecture (see Figure 4.1) was used for developing the new software so that code could be shared between various interfaces. A review of the different layers that were required is provided in the following subsections.

5.3.1 *Web Layers*

A typical web application is composed of the following four layers [23], see Figure 5.1.

1. GUI Layer: Used to present information.
2. Communication Layer: Used to translate information between the user and the web server.
3. Middleware Layer: Used to process requests using the web server resources.

4. Data Layer. Used to store and query data.

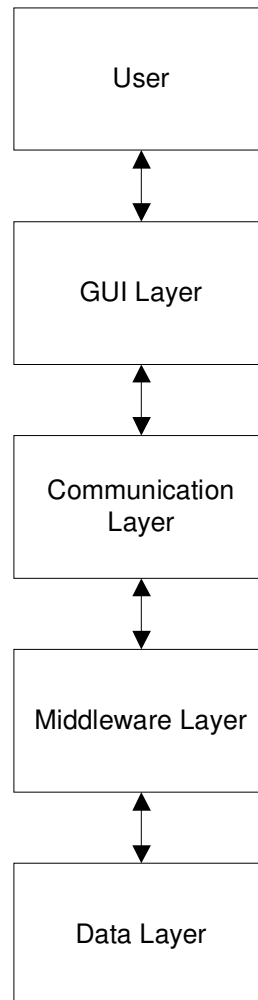


Figure 5.1 Structure of Layers within Web Applications

The GUI, Communication and Middleware Layers are discussed in sections 5.5 to 5.7 and a brief overview of the available web technologies considered for use in this project in section 5.4.

5.4 Overview of Technologies

Table 5.1 to Table 5.7 provide brief details on technologies considered for use in this project.

The technologies marked by an asterisk (*) were not used in the project.

Table 5.1 XML Technologies

Technology	Layer(s)	Description
XML	All	<u>Extensible Markup Language</u> XML [19] contains data in a hierarchy using a tag structure similar to the Hypertext Markup Language (HTML) tag structure; however, whereas HTML defines how elements are displayed, XML defines what those elements contain.
XSLT	Communication	<u>Extensible Stylesheet Language Transformation</u> Widely used to convert XML to HTML for screen display. XSLT [5] is also used to convert an XML document to text and PDF.
DTD*	Communication	<u>Document Type Definition</u> DTD [31, 43] is a language that is used to describe the contents of an XML document.
XSD	Communication	<u>XML Schema Definition</u> The informal name for the XML schema [7] from the W3C. It is a superset of DTD and is written in XML syntax.
AJAX	GUI	<u>Asynchronous JavaScript and XML</u> Allows a Web page to retrieve small amounts of data from the server without reloading the entire page.

Table 5.2 Web Technologies

Technology	Layer(s)	Description
ASP	Communication Middleware	<u>Active</u> <u>Server</u> <u>Page</u> An ASP [20] is a Web page that contains HTML and embedded programming code written in VBScript or JavaScript.
JSP*	Middleware	<u>Java</u> <u>Server</u> <u>Page</u> A JSP extension to the Java servlet technology from Sun that allows HTML to be combined with Java on the same page.

Table 5.3 Web Languages

Technology	Layer(s)	Description
HTML	GUI	<u>HyperText Markup Language</u> The document format used on the Web. Web pages are built with HTML [24] tags (codes) embedded in the text. HTML defines the page layout, fonts and graphic elements as well as the hypertext links to other documents on the Web. Each link contains the URL, or address, of a Web page residing on the same server or any server worldwide, hence "World Wide" Web.
DHTML	GUI	<u>Dynamic Hypertext Markup Language</u> DHTML [16] is a combination of HTML enhancements, scripting language and interface that are used to deliver animations, interactions and dynamic updating on Web pages.
XHTML	GUI	<u>Extensible Hypertext Markup Language</u> XHTML [25] is a reformulation of HTML 4.0 as an application of XML. It can be adapted and used for describing the appearance of web pages.
CSS	GUI	<u>Cascading Style Sheets</u> CSS [4] is a style sheet format for HTML documents endorsed by the World Wide Web Consortium.
JavaScript	GUI Communication	JavaScript [17] adds interactive functions to HTML pages, which are otherwise static, since HTML is a display language, not a programming language.

Table 5.4 Web Servers

Technology	Layer(s)	Description
IIS	Middleware	<u>Internet Information Services</u> IIS [32] is Microsoft's Web server which adds the full HTTP capacity to the Windows operating system.
Apache*	Middleware	Unix-based Web server from the Apache Software Foundation [33].

Table 5.5 Protocols

Technology	Layer(s)	Description
HTTP	Communication	<u>HyperText Transfer Protocol</u> HTTP [29] is a request/response protocol used to communicate between clients and servers.

Table 5.6 Standards

Technology	Layer(s)	Description
COM	Middleware	<u>Component Object Model</u> A component software architecture [13] from Microsoft, which defines a structure for building program routines (objects) that can be called up and executed in a Windows environment.
DCOM*	Middleware	<u>Distributed Component Object Model</u> DCOM [50] is based on COM, Microsoft's component software architecture, which defines the object interfaces. DCOM defines the remote procedure call that allows those objects to be run remotely over the network.
RMI*	Middleware	<u>Remote Method Invocation</u> RMI is a remote procedure call (RPC), which allows Java objects (software components) stored in the network to be run remotely.
CORBA*	Middleware	<u>Common Object Request Broker Architecture</u> A software-based interface from the Object Management Group (OMG) that allows software modules (objects) to communicate with each other no matter where they are located on a private network or the global Internet.

Table 5.7 Other

Technology	Layer(s)	Description
Java applet / servlet*	Middleware	<u>Java</u> <u>Applet</u> A Java program that is downloaded from the server and run from the browser. <u>Servlet</u> A Java application that runs in a Web server or application server and provides server-side processing such as accessing a database and e-commerce transactions.
Active X components	GUI	A software module [27] based on Microsoft's COM architecture.

Most of the technologies used were chosen in part due to the KELTON® IT infrastructure (see section 3.3). KELTON® have a policy of using Microsoft products wherever possible because of compatibility with their other products. Also KELTON® is a Microsoft certified partner and therefore has a wealth of experience with the technology as well as easy access to full documentation, tools and even Microsoft engineers to resolve problems if necessary.

With XML [19] technologies, XSD [7] was chosen over Document Type Definition (DTD) [31, 43] for validating XML due to its enhanced capabilities; see section 8.3.3 for a more detailed discussion.

The technologies used in the project are described in more detail within the context of the web layers presented in the following subsections.

5.5 GUI Layer

Information generated by a web server is presented to the user by returning a HTML [24] or DHTML [16] document which can be rendered by the web browser. The web browser knows how to interpret HTML and display it according to HTML formatting instructions as a web page.

The web browsers most commonly used in industry are Internet Explorer and Netscape, although newer browsers such as Firefox and Opera are becoming more established.

Technologies were evaluated to ensure all the functionality of the web application was met. The main tasks the web pages must perform were also assessed.

5.5.1 *Display Of Web Pages*

Web pages generally benefit from having a consistent look and feel. This can be achieved by formatting text and page backgrounds using common formatting rules. These rules are enforced by inserting instructions on the fonts and colours for all the paragraphs and headers and by inserting a background to each page.

If a company wished to have their own corporate image it would mean that all the tags would have to be manually updated and this could prove tedious and error prone. A CSS [4] is an alternative option that allows formatting instructions to be set in one file and work for all tags. A CSS file was supplied

by KELTON® and used in the web pages for the applications developed in this project to present pages with the corporate image for KELTON®.

Information should be presented in a meaningful manner on the web browser and user responses passed to the web server. An example of passing user responses to the web server was found in the online Market Survey program (see section 4.2.4) where the user responses were sent to the web server and stored for analysis. It is also necessary for a user to select options and enter inputs for a calculation in a calculation application (see section 3.2.1) and to let users answer questions in the training application (see section 3.2.4) which are processed by the web server.

User responses are collected by including components such as text fields and option boxes in the HTML document inside an HTML form [18]. The components are included in a form (see Figure 5.2) within the document which can be read by the web browser when a user submits a request. The web server uses POST and GET methods in the communication layer to convey form data.

Figure 5.2 is an example of including a form within the body tag of an HTML page. This was needed whenever details entered by the user were sent to a server program using the POST method.

Form created for changing unit type i.e. scalar to type length which opens the Unit.asp page and passes the value ChangeUnits to the Mode parameter

```
<form id="frm1" name="frm1" method="POST"
action="Unit.asp?Mode=ChangeUnits" xmlns:msxsl="urn:schemas-
microsoft-com:xslt" >
  <table align="center" width="60%"> Create a table
    <tr>
      <td>Convert: </td>
      <td>
        combo box containing unit types, the Unit.asp program will
        get the unitTypeID value when the form is submitted
      </td>
      <td>
        <select onchange="JavaScript:frm1.submit()"
        name="unitTypeID">
          <option value="0" selected="selected">Scalar</option>
          <option value="1">Factor</option>
          <option value="2">/Length</option>
        </select>
      </td>
    </tr>
  </table>
</form>
```

Figure 5.2 HTML Form

The HTML that the user is presented with is shown in Figure 5.3.

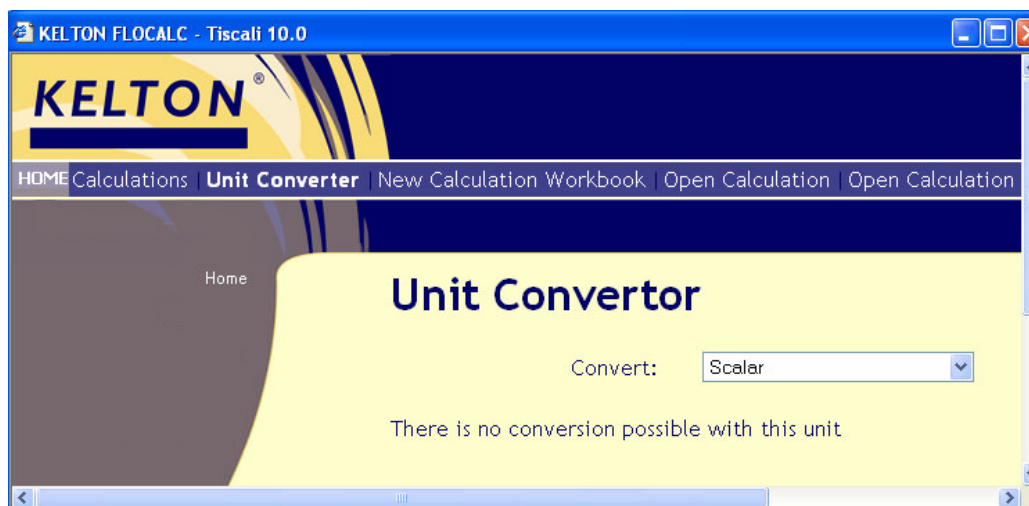


Figure 5.3 ‘Unit Converter’ Window in FLOCALC®

Any web browser can render standard HTML tags on the browser’s display but there are only a limited number of tags. Therefore, it was necessary to investigate technologies which can be incorporated in the web document to enhance the web browsers display and interaction capabilities.

5.5.2 DHTML

DHTML [16] provides a way to combine plain HTML with scripting languages and uses style sheets such as CSS [4] to add style attributes to each element in the document such as fonts and layouts.

The majority of pages created in this project used DHTML. JavaScript [17] was used to check user inputs were valid prior to submitting responses. The online survey program (questionnaire) presented a DHTML document which included components for respondents to enter their preferences to preset questions. The document included JavaScript to ensure all mandatory questions were answered before posting the document to the server.

Figure 5.4 is an example of an option box used for validating a text string as a floating point number using an HTML form, Submit command and JavaScript:

```

JavaScript tag included in the body tag
<script language="JavaScript">
  <!-- Function executed on submit of the form
  function pSubmit() {
    var oForm=document.forms("frm2"); Get Form frm2
    var pSubmit=true;
    if (pSubmit) {
      Return true if the inputs on the form are numbers
      pSubmit=(pSubmit) && (pCheckInput(oForm.Input));
    }
    return pSubmit;
  }
  Function checks text is a number
  function pCheckInput(oText){
    var bText;
    if (oText.value=="") { if text box is empty
      alert("Please enter a number in the input field");
      oText.focus();
      bText=false;
    } else {
      bText=true;
    }
    if (isNaN(oText.value)) { if text box is not a number
      bText=false;
      alert("Please enter a number in the input field");
    } else {
      oText.value = parseFloat(oText.value);
    }
    return bText;
  } //
--></script>

```

Figure 5.4 JavaScript to validate and submit HTML Form

5.5.3 Display of Files

The web browser cannot display certain types of data such as audio and video.

The RTF's are displayed as a server side transformation object which returns Extensible Hypertext Markup Language (XHTML) [25]. This is then returned by the browser and displayed natively.

The audio and animation files are sent as they are and it is expected that the browser has the appropriate plug-in [26] or Active X installed.

A plug-in [26] can be installed on the user's PC to allow them to view pages containing items not included in the web browser. The plug-in extends the browser's capabilities by plugging in components which will translate files on a web page.

An Active X [27] component can display different types of media or use IE native playback which also uses Active X components.

Plug-ins are used by browsers other than internet explorer, whilst Active X is a Microsoft Windows only technology.

It is possible to provide advice on where to download plug-ins and Active X components and which version is required.

5.5.4 JavaScript

JavaScript [17] requires a plug-in to allow added features to be displayed on the web pages. The language is extremely small and can be run on any machine that has a Java virtual machine. JavaScript can be used to store the history of a user's choices, check validity of inputs and give alerts to help users when using the browser to ensure all fields are filled in correctly before submitting the form.

5.5.5 AJAX

A web application can be very limited in its functionality by comparison to a standalone application. Speed is also an issue if it is required to make constant requests to the web server which results in the creation of a new web page at each response.

There are techniques available to provide increased functionality and to handle repeated requests in an efficient manner. One way is to use JavaScript as mentioned in section 5.5.4. There is also Asynchronous JavaScript and XML (AJAX) [28], which uses an enhancement in JavaScript that allows Web pages to be more interactive and behave like local applications.

AJAX creates interactive web applications using the XMLHttpRequest object to get current XML [19] documents which are then used to update named tags in the HTML [24] document. The XMLHttpRequest gets an XML document returned via the communication layer from the web server.

This technique was useful as it allows the web page to retrieve small amounts of data from the server without reloading the entire page following user input.

An example of AJAX being used is in the KITS[®] module list. There is a hierarchy of modules and module groups within a module group. The HTML page containing the list is designed to be updated as a user requests the modules and groups within a module group (see Appendix B – Module List using AJAX).

5.6 Communication Layer

The communication layer conveys information from the web browser to the web server. The key issues are secure and accurate transfer of data. Data must be transferred from one software package on a machine to another software package on another machine.

5.6.1 HTTP

Multiple file types such as text, images and audio are passed between the browser and server. Browsers and servers use the HyperText Transport Protocol (HTTP) [29] protocol to exchange web documents. HTTP is a “request/response” protocol (see Figure 5.5) that defines the means of transport and describes file formats using Multiple Internet Mail Extensions (MIME) [30] specifications.

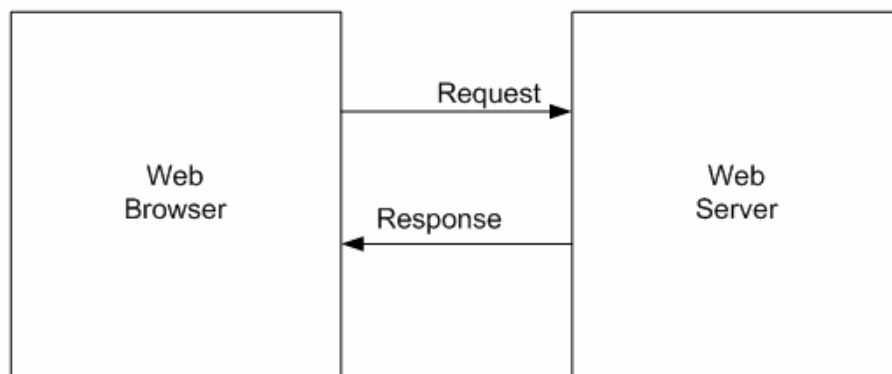


Figure 5.5 HTTP request/response Protocol

5.6.2 ASP Component

An ASP [20] component is a server side Active X [27] component which runs on a server and sends HTML information across the internet allowing for the creation of dynamic, interactive sessions with the user.

The FES and BES dll's [12] have a reference to the Active Server Component. The Active Server Component contains the following objects: -

- Application
- Session
 - Gets the user session
- Server
- Request
 - Requests the components in the web browser and any parameters passed in
- Response
 - Response passes back the response from the BES ASP page to the FES dll.

The GET method only allows a limited number of data (1024 characters) to be appended to Uniform Resource Locator (URL) as encoded request data and the user can see all of this data.

The web server also implements a method for a web browser to send optional search parameters as well.

The browser gets the name/value pairs to put into the HTTP message body through the HTML form [18].

Figure 5.6 is an example of a URL being submitted in an HTML page.

`FES.asp?RequestOption="NewCalc"&calcid="calcID"`

Figure 5.6 HTML URL Request

The FES.asp page is called with values for parameters “RequestOption” and “calcid” being passed in.

Often large amounts of data are passed through the communication layer such as a calculation inputs and options. It is therefore necessary to allow the server to read the components directly. This is done using the POST method which posts an HTML form on submit.

Data has to be protected in some instances because of the Data Protection Act, such as personal details gathered in the Market Survey (see section 4.2.2). The data was protected using anonymous queries consisting of a

unique ID and email address. The data was stored on a secure server and removed when no longer required.

In the case of KITS[®], user names and passwords used for logging in were not visible. This is handled by passing in the login form on submit with the components being read by the ASP request object.

5.6.3 XML Data Handling

There was a need to pass data created dynamically to the browser. The FES dll [12] can be coded to get data and create the HTML. It is also possible to separate data from the representation with the use of XML [19] technologies. This was accomplished by getting an XML document from the BES and transforming it with the use of an XSLT [5] in the FES (see section 8.3.4).

The XSLT files are created at design time and stored in the FES. It is also possible to include any kind of tag that would normally be found in an HTML page such as JavaScript [17] language tags, pointers to CSS [4] files and forms. The XSLT translates XML into HTML using templates to generate the final document.

XML documents are often sent back to the server, especially when importing data. These documents can be validated against schema files such as XSD [7] or DTD [31, 43]. The schemas are created at design time.

The XSD defines the structure of an XML document (see section 8.3.3).

5.6.4 Communication Sub Layers

The methods used in the applications divide the communication layer into the three distinct sub layers shown in Figure 5.7.

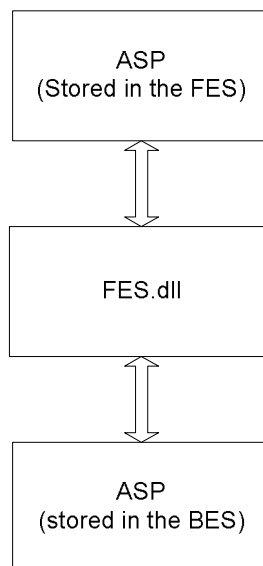


Figure 5.7 Communication Sub Layers

The following example taken from the project work illustrates how this works.

The browser calls the ASP[20] page in the FES to make requests. This is carried out on submit using JavaScript [17], as previously discussed in section 4.2.4, with a command shown in Figure 5.8.

```

Post the Unit Converter form with the action
"FES.asp?RequestOption=ConvertUnit"
  
```

Figure 5.8 JavaScript call to ASP

This calls the FES.asp document and adds a request parameter RequestOption=ConvertUnit to be dealt with in the communication layer

The ASP page shown in Figure 5.9 calls the public method “RequestDocument” in the FES object and includes the Request parameter created in Figure 5.8.

```
<%  
    dim oWC: set oWC=server.createObject("FLOCALC_FES.FES")  
    oWC.RequestDocument  
    Application,Session,Server,Request,Response  
    set oWC = nothing  
%>
```

Figure 5.9 FES default ASP

The FES dll [12] shown in Figure 5.10 handles the front end server request and transforms data into HTML [24].

The parameter “RequestOption” is read from the ASP Request object “oRequest” to perform a unit conversion.

```

Public Sub RequestDocument(oApp As ASPTYPELibrary.Application,
oSession As ASPTYPELibrary.Session, oServer As
ASPTYPELibrary.Server, oRequest As ASPTYPELibrary.Request,
oResponse As ASPTYPELibrary.Response)
    Variables required to get requests and response values
    Dim strReq As String
    Dim lUnitTypeID As Long
    Dim lInputUnits As Long
    Dim lOutputUnits As Long
    Dim dValue As Double
    Select Case oRequest("RequestOption")
        Case "ConvertUnit"
            Get the request values from the components in the form
            lUnitTypeID = oRequest("unitTypeID")
            lInputUnits = oRequest("InputUnits")
            lOutputUnits = oRequest("OutputUnits")
            dValue = CDBl(oRequest("input"))
    End Select
    Create the request string to be passed to the BES ASP page
    with parameters and values required
    strReq = "BES.asp?unitTypeID=" & lUnitTypeID & "&inputID=" &
lInputUnits & "&inputValue=" & dValue & "&outputID=" &
lOutputUnits
    Get an XML document from the BES program using a function
    pGetXML
    Dim oXMLDoc As MSXML2.DOMDocument40: Set oXMLDoc =
pGetXML(strReq)
    Display the unit by transforming the returned XML document
    with the Unit XSLT [5] Stylesheet into DHTML
    pDisplayUnit oXMLDoc, msiUnitXSL
End Sub

```

Figure 5.10 FES dll code to process request

The request to the BES is created in the FES dll code as a string - "strReq" in Figure 5.10.

The request object is passed to the BES asp in Figure 5.11, which is then read by the BES object in Figure 5.12.

```
<%
  dim oWC: set oWC=server.createObject("KITS_BES.BES")
  oWC.Units Application, Session, Server, Request, Response
  set oWC = nothing
%>
```

Figure 5.11 BES default ASP

The generated XML [19] document is saved in the response object which the BES asp references. The response object is then used by the FES object to create a new HTML page by transforming the XML document saved in the response object.

```
Public Sub Units(oApp As ASPTypelibrary.Application, oSession
As ASPTypelibrary.Session, oServer As ASPTypelibrary.Server,
oRequest As ASPTypelibrary.Request, oResponse As
ASPTypelibrary.Response)
  'Create a converted unit XML document to be passed to the
front end server
  Dim oXMLDoc As MSXML2.DOMDocument40: Set oXMLDoc =
  pConvertUnit(oRequest)
  Clear the response object
  oResponse.Clear
  oResponse.contentType = "text/xml"
  Save the XML document to the response object which will be
required by the FES program for transformation to a new
DHTML page
  oXMLDoc.save oResponse
End Sub
```

Figure 5.12 BES dll code to pass back response object

5.7 Middleware Layer

The web server is the entry point to the middleware layer. The purpose of the middleware layer is to accept incoming requests and process them, using the web server resources.

The ASP [20] page in the BES requests data from the web server. The web server in this instance is a BES dll [12] which accepts requests and responds to them by returning data through the response object.

The BES processes the requests in the Market Survey program and creates a file to be returned to the FES. The BES may also get and set data through the business logic layer for more complex applications such as FLOCALC® and KITS®. This is discussed in section 6.6.

5.7.1 IIS

Internet Information service (IIS) [32] is a web server that provides manageable and scaleable web application infrastructure. It is used to increase web site and application availability. The IIS ensures security of the web site and its data. There are other server applications available such as Apache [33] but IIS was used due to KELTON's experience and existing infrastructure with Windows Server 2003 (see section 3.3). Apache is more appropriate for Unix and Linux servers.

5.7.2 COM Objects

ASP can use COM [13] objects to separate functionality into Active X components [27]. This means that data such as XML [19] can be dealt with in a COM object and validated.

COM provides the following features: -

- A common mechanism for applications to access and perform operations on objects
- A mechanism for keeping track of whether the object is in use and deleting it if no longer needed
- A standard error reporting mechanism and set of error code values.
- A mechanism for applications to exchange objects
- A way to identify objects and associated objects with applications that understand how these objects are implemented

6 Overview of Technologies used In Standalone Applications

6.1 Introduction

Based on the market research, only a minority of respondents (24%) wanted software to run on standalone PCs.

Although the feedback indicated a preference for web-based systems, not all of the respondents knew the technical restrictions this imposes on when and where it can be used. Since KELTON® is aware of the usage patterns of its software both internally and amongst its clients, it wanted to develop the software using a common business logic core that could be used by both a web and standalone version, reducing the amount of maintenance required and providing compatibility between the two.

This chapter reviews the technologies required in standalone applications and potential uses in the flow metering industry.

6.2 General Requirements

The requirements for standalone applications matched those outlined for web applications (refer to section 5.2), except the data is held on the user's PC and secured against tampering.

6.3 Overview of Standalone Applications

Applications written for standalone PCs are intended to work without an internet connection. All required functionality must be incorporated into the application and be accessible via a “rich” user interface. The existing versions of KELTON® software at the start of the project were designed for standalone deployment.

Standalone applications can be written in a variety of programming languages, e.g., C++, Visual Basic, .Net, Fortran and Pascal. For this work, all applications were written in Visual Basic 6 due to KELTON's policy of using Microsoft product wherever possible and availability of existing code at KELTON®.

Standalone applications have direct access to data. Interaction with the user can be dealt with more easily than in a web application. It is also possible to install components directly to a user's PC. This means there is more flexibility to develop rich interfaces with increased functionality, usability and speed.

The main design issues are how best to deliver software that is easy to update when changes are requested and how to exploit common functionality found in different standalone applications.

In building standalone applications there were various technologies to consider. After the success in developing the Market Survey program, an n-tier [3] software architecture (see Figure 4.1) was used for sharing code

between the web application and standalone. A review of the different layers required in standalone applications using n-tier is presented in section 6.3.1.

6.3.1 Standalone Layers

An n-tier layered architectural pattern has been adopted to separate the presentation from the logic, as described below and illustrated in Figure 6.1.

1. UI Logic Layer: Used to handle opening and closing interfaces. This layer also deals with handling events from interfaces that affect other interfaces and common UI functions such as licensing.
2. UI Presentation Layer: Used to present the information. This layer holds the interfaces needed for interacting with the software, getting data and processing.
3. Business Logic Layer: Used to store the classes/objects for the application. This layer is where data is collected and processed with objects holding the results returned to the UI.
4. Data Layer: Used for dealing with the data store, see Chapter 8.

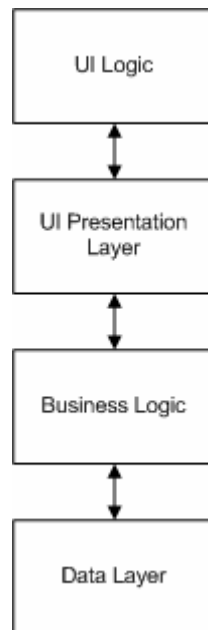


Figure 6.1 Structure of Standalone Layers

This is related to the widely used Model-View-Controller (MVC) [37] architectural design pattern, which is briefly discussed in section 6.7.

The UI Logic, Presentation and Business Layers are discussed in sections 6.4 to 6.6. The technologies used in the project are described in the context of each of these standalone layers. (A full list of technologies was presented in Table 5.1 to Table 5.7)

6.4 UI Logic

The UI Logic coordinates access to user interfaces which in turn accesses the data layer through the business logic. This layer can be an Active X [27] dll [12] or exe file.

6.4.1 Interface

A standard graphical interface which allows users access to the application interfaces created in the UI Presentation Layer and other common functionality such as licensing and version information was needed. It is important to consider carefully the design of the interface and the possible use of common code so that the overall look and feel of the application is consistent with other applications in the company's software product range.

The UI logic controls how the user interacts with the application and its design must be carefully tailored to ensure easy access to all functionality.

This was achieved by creating an executable program which generates a graphical interface giving the user access to the rest of program within the n-tier architecture see Figure 6.2.

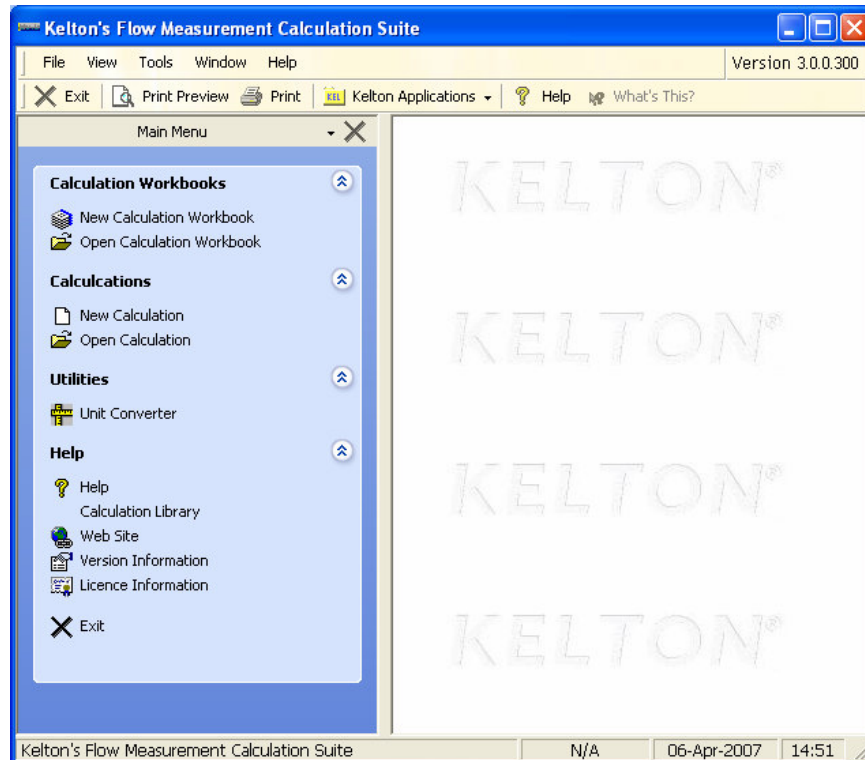


Figure 6.2 UI Logic Graphical Interface

6.4.2 Interaction with the UI Presentation Layer

The following aspects of the UI Logic's interaction with the UI Presentation Layer needed to be considered.

- It needs to store details of the Presentation Layer interfaces being opened and closed.
- It needs to know about events occurring in the Presentation Layer interfaces.
- It needs to be able to complete requests from the Presentation Layer interfaces to prevent duplication of code.

6.4.3 Presentation Interface Objects

As each interface is opened, a Multiple Document Interface (MDI) [53] form object is created in the UI Logic Program and stored in a number of ways by the main application.

The MDI form can be stored as a single object. This method is used when there is only ever one interface object required for a particular interface type in the Presentation Layer such as a Unit Converter.

An empty interface object is created on start up and then assigned with the MDI object as required. Example code is shown in Figure 6.3.

```

'Create a single object to use for handling the Unit Converter
Private m_oUnitConverter as DirectOCXTypes.UnitConverterDirect

Private Function pShowUnitConverter() As Boolean
    If m_oUnitConverter Is Nothing Then
        'Create a new MDI form for displaying the interface
        Set m_oUnitConverter =
            MDI.NewForm("UNITCONVERTER_UI.UnitConverter",
                "UnitConverter", "Unit Converter")
        'Pass in the KCCL Unit types so that the interface can
populate 'itself with units
        With m_oUnitConverter
            Set .UnitTypes = Session.KCCL.UnitTypes
        End With
    End If
    'Show the Unit Converter
    MDI.Form(m_oUnitConverter).Visible = True
End Function

```

Figure 6.3 MDI form containing a single object

An interface may need to be opened several times, e.g., in FLOCALC® several calculation workbook interfaces may be opened at the same time. To allow users to open multiple interfaces of a single type, a collection (or array) of objects for storing the MDI objects is used.

With an array, it is necessary to know exactly which interface item is being accessed. There can be difficulty if removing objects from the “middle” of the array as objects after the item removed need to be carefully placed in the correct position in the array.

Visual Basic collections can be used to store objects. There is the ability to assign a key to each object so that objects can be removed without affecting other items within the collection. The key requires a unique identifier.

In the case of a calculation workbook in FLOCALC® a unique ID is created at the creation of the workbook interface object and used as the key.

An example of a collection of objects is given in Figure 6.4.

```

'Create a weCollection (special collection for handling with
event objects) to 'store calculation workbooks
Private WithEvents m_colWorkbooks as WECollection

Private Function pShowWorkbook(oWorkbook As
FLOCALC_Core.CalculationWorkbook) As Boolean
    If oWorkbook Is Nothing Then 'Check there is a workbook
object
        pShowWorkbook = False
    Else
        If Not m_colWorkbooks.KeyExists(oWorkbook.ID) Then
            'Create a new UI object
            Dim oWorkbookUI As
            DirectOCXTypes.CalculationWorkbookDirect
            'Set the UI object type, key and name used in title
            Set oWorkbookUI =
            MDI.NewForm("FLOCALC_UI.CalculationWorkbook",
            oWorkbook.ID, oWorkbook.Name)
            'Set the UI Core workbook object used to populate the
'interface
            With oWorkbookUI
                Set .CalculationWorkbook= oWorkbook
            End With
            'Add the UI to the workbook collection
            m_colWorkbooks.Add oCoursesUI, , oWorkbook.ID
        End If
        'Show the calculation workbook interface
        MDI.Form(oWorkbook.ID).Visible = True
    End if
End Function

```

Figure 6.4 MDI form containing a collection

6.4.4 Handling Object Events

There are a number of events that are raised in the UI Presentation Layer which are passed to the UI Logic program for handling, such as closing.

A presentation interface raises a closing event which the MDI [53] single object or object in a collection can detect.

In the case of a single interface object there is no real problem as the UI Logic can handle late bound events in Visual Basic raised by the interface objects using WithEvents [34]. This means that a closing event will tell the UI Logic program to set the interface object to null.

An example of how to handle a single object event is given in Figure 6.5.

```
'Create a single object to handle the Unit Converter and
events raised
Private WithEvents m_oUnitConverter as
DirectOCXTypes.UnitConvertorDirect

'Unit converter closing event raised
Private Sub m_oUnitConvertor_Closing()
    'Close the Unit Converter UI and empty the object
    MDI.Form(m_oUnitConvertor).Close
    Set m_oUnitConvertor = Nothing
End Sub
```

Figure 6.5 Handle closing event of a single object

With VBA Collection, there is no facility for knowing when an event has been raised by the presentation interface object. It is therefore necessary to use a

“with-events” collection (WECollection) The WECollection has been created as a common module for handling these special collections. The WECollection was made available using freely available code [35].

An example of how to handle events in a collection of objects is given in Figure 6.6.

```

Private WithEvents m_colWorkbooks As weCollection

'Sub routine for handling the events raised by objects within
a collection
'Index
'Source -UI Object that has raised the event
'RaisedEvent - String indicating the event raised such as
closing
'aParams - Parameters passed by the object when an event is
raised

Private Sub m_colWorkbooks _EventRaised(ByVal Index As Long,
ByVal Source As Object, ByVal RaisedEvent As Variant,
aParams() As Variant)
    'Get the WB UI object
    Dim oWB As DirectOCXTypes.CalculationWorkbookDirect
    Set oWB = Source

    'Get the event raised
    Select Case RaisedEvent
        Case "Closing"
            Dim m As IKForm 'Object required for the MDI form
            'This UI is identified by the workbook id
            Dim ID as String: ID = oWB.CalculationWorkbook.ID
            'Close the workbook using the collection key
            pCloseWorkbook m_colWorkbooks.ItemByKey(ID)
        Case "OpenCalc"
            'Get the calculation selected in the workbook and open UI
        Case "ExportWorkbook"
            'Get the Core workbook object to use an internal interface
            to export the data
    End Select
End Sub

```

Figure 6.6 Handle closing event of an object in a collection

The UI Logic needs to dispose of objects in order to prevent memory leakage when leaving the application. When an interface in the UI Layer is closed an event is raised as previously discussed. However, the application can also be closed directly. A fatal error would occur if the objects or collections were not closed correctly and in the correct order as an object being closed may reference objects which need to be closed first e.g. closing a FLOCALC[®] workbooks needs to close the calculation interfaces associated with the workbook first.

It is therefore necessary to have termination code which will close objects correctly and allow users to save items where appropriate before disposing of the interface object. An example of such code is presented in Figure 6.7.

```

'Termination code called when the class is closed
Private Sub Class_Terminate()
    pTermForm m_oUnitConverter 'Terminate a single MDI form
    pTermForms m_colCalculations 'Terminate a collection of MDI
forms
    Set m_oUnitConverter = Nothing
    Set m_colCalculations = Nothing
End Sub
'Sub routine to close an MDI form then set the object to
nothing
Private Sub pTermForm(oCtrl As stdole.IUnknown)
    If Not oCtrl Is Nothing Then ' Get the MDI form and close
        Dim oFrm As IKForm: Set oFrm = MDI.Form(oCtrl)
        If Not oFrm Is Nothing Then oFrm.Close
    End If
    Set oFrm = Nothing 'Set the form object to nothing
End Sub
'Sub routine to terminate all the MDI forms in a collection
Private Sub pTermForms(oCol As weCollection)
    If Not oCol Is Nothing Then
        While oCol.Count > 0
            ' Get the ID and object of the last item in the collection
            Dim strKey As String: strKey = oCol.Key(oCol.Count)
            Dim oCtrl As stdole.IUnknown
            Set oCtrl = oCol.ItemByKey(strKey)
            pTermForm oCtrl 'Terminate the form
            Set oCtrl = Nothing
            'Remove the object from the collection
            If oCol.KeyExists(strKey) Then oCol.RemoveByKey strKey
        Wend
    End If
    Set oCtrl = Nothing
End Sub

```

Figure 6.7 Termination procedure for closing the UI Logic

6.4.5 Common Functionality

It is preferable for some tasks to be completed by the UI Logic to avoid duplication of code. An example of this is importing and exporting data through a shared interface using a common dialog control [36] component to access an external file.

On start up there are a number of issues to resolve before the user can have access to the main program functions, for example:-

- Licensing
 - Is the user licensed to use the software and if so what level of access is permitted? Licence controls were needed in KITS[®] where a user can license selected modules and the license can impose constraints on the level of access, e.g., whether the user is allowed to create their own modules or restricted to using existing modules only (see section 7.2.1, 7.3.5 and 7.4.5 and for details of software licensing).
- User Rights
 - It may be necessary to include a user login screen to restrict access to data according to user rights. This is handled in the UI Logic which has access to the data file created to store user information (see section 7.4.1 for an example used in KITS[®] of administrator access).

Common functionality between the UI Logic and Presentation Layer was coded in the UI Logic to prevent duplication of code such as printing and

opening help files. This was done because the UI Logic is automatically created on start-up whereas the UI Presentation Layer is created as objects are required by the UI Logic, and can be closed at any time. There are also multiple UI presentation objects created, whereas there is only one UI Logic object created and this is used to control the user interaction.

For enhanced functionality and for fast development, common components such as Active X [27] controls and COM [13] objects were identified using a variety of methods and components such as toolbars and menus.

A user interface for importing data must be created in the UI Logic as an interface is not created for displaying the data until the data has been imported and data object created. Importing and exporting use the same user interface and a module is created to handle the logic. It is therefore preferable to export data by raising an event from the UI Presentation Layer to the UI Logic to allow a user to export data.

6.5 UI Presentation Layer

The user interfaces required specifically for the application are developed in the UI Presentation Layer such as an interface for presenting a FLOCALC[®] calculation. This layer presents data generated by the business logic and allows the user to interact with it. The business logic is separated from the presentation so that the business logic can be reused within other applications, e.g., web applications.

The interfaces needed to be developed in an intuitive and user friendly manner. Interfaces are implemented as user controls separate from the UI logic. A user control provides the means to create a custom interface that can be reused. It can consist of one or more components with blocks of code to extend the functionality for modifying display properties.

6.5.1 User Control

The UI Presentation Layer was created as a Visual Basic Active X [27] .OCX and includes the user controls required in the standalone software.

User controls are created as graphical interfaces and are similar to forms but, unlike forms can be included within other forms/user controls. A user control is created in preference to a form because the interface can be kept separated from the UI Logic but can also have information passed to it such as a calculation object or file name.

Another advantage is that a user control is an object, therefore can be reused by other user controls. An example of this is a metering header user control was created containing header information such as site and company. The control was encapsulated within the FLOCALC[®] calculation and FLOCALC[®] calculation workbook user control.

The UI Presentation Layer dll [12] is referenced in the UI Logic project. This allows the UI Logic to call up and show an interface from the UI Presentation Layer as required (see Figure 6.3 and Figure 6.4). When the UI Logic creates a UI presentation object it initialises the object with suitable information and is informed of events that have occurred in the objects such as closing.

A user control object is public so that the UI logic can have access to the interface object. Public let/set properties are included to allow the UI logic to pass and alter information as necessary. Finally, the user control has public events required by the UI Logic to act upon.

Examples of user controls within the project are :-

- Unit converter display
- The list of KCCL calculations displayed in FLOCALC[®]
- A training module in KITS[®]

A sample of the code required in the Unit Converter user control is given in Figure 6.8.

```

'A public events required by the UI Logic.
Public Event Closing

'The unit converter needs the unit types object from the KCCL
library
Private m_oUnitTypes as KCCL.UnitTypes

Private Sub UserControl_Initialize()
    pInit 'Set the user control background and other components
End Sub

Private Sub UserControl_Terminate()
    'Make objects = Nothing to prevent memory leakage
    Set m_oUnitTypes = Nothing
    'Tell the UI Logic that the user control is closing
    RaiseEvent Closing()
End Sub

'Public properties to get and set objects from the UI Logic
code
Public Property Set UnitTypes(oUnitTypes As KCCL.UnitTypes)
    If m_oUnitTypes Is Nothing Then
        Set m_oUnitTypes = oUnitTypes
    End If
    pPopulate 'Populate the components with unit type object
End Property

Public Property Get UnitTypes() As KCCL.UnitTypes
    Set UnitTypes = m_oUnitTypes
End Property

```

Figure 6.8 User control initialise, terminate and property code

6.5.2 Components

For the user interfaces to be easy to use, a consistent “look and feel” can be created in the program by using buttons, menus, toolbars, grids, etc., which are common to other interfaces within the application. This is achieved by using Active X [27] components which come from the same source. It is necessary to identify the components that are required and how the user will interact with them before coding.

The components can be from Microsoft or specially designed such as the KELTON[®] components used in the FLOCALC[®] and KITS[®] software. These have additional functionality and also ensure consistency and the KELTON[®] corporate “look and feel”.

6.5.3 Interfaces

Two types of interfaces, static and dynamic, are discussed below along with examples of their use.

- A static interface is developed with all the components on a user control. This approach is followed if no changes to the look of the interface are anticipated. In practice a small number of component properties can be changed.
 - For example, in the case of a Unit Converter the arrangement of the interface stays consistent no matter which unit is chosen. This is therefore a good method to use, although a combo box that displays the units still needs to be dynamically populated

depending on the unit type chosen, e.g., for pressure the unit could be bara, Pascals or mmHg, see Figure 6.9.

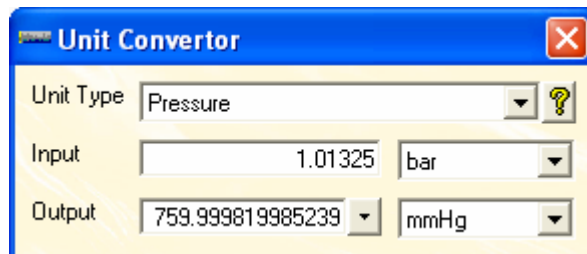


Figure 6.9 Unit Converter Interface

- A dynamic interface can be developed initially with a basic user control. New components are added, changed or removed dynamically at run time. This means that the interface can be reused under a number of different circumstances.
 - For example the KELTON[®] common calculation library (KCCL) currently implements 47 calculations. However new calculations will inevitably be added as new standards are defined by the flow measurement industry. It has therefore been necessary to develop a dynamic interface for accessing KCCL. The interface uses the list of calculations from KCCL to build a list of calculations on the user interface, see Figure 6.10.

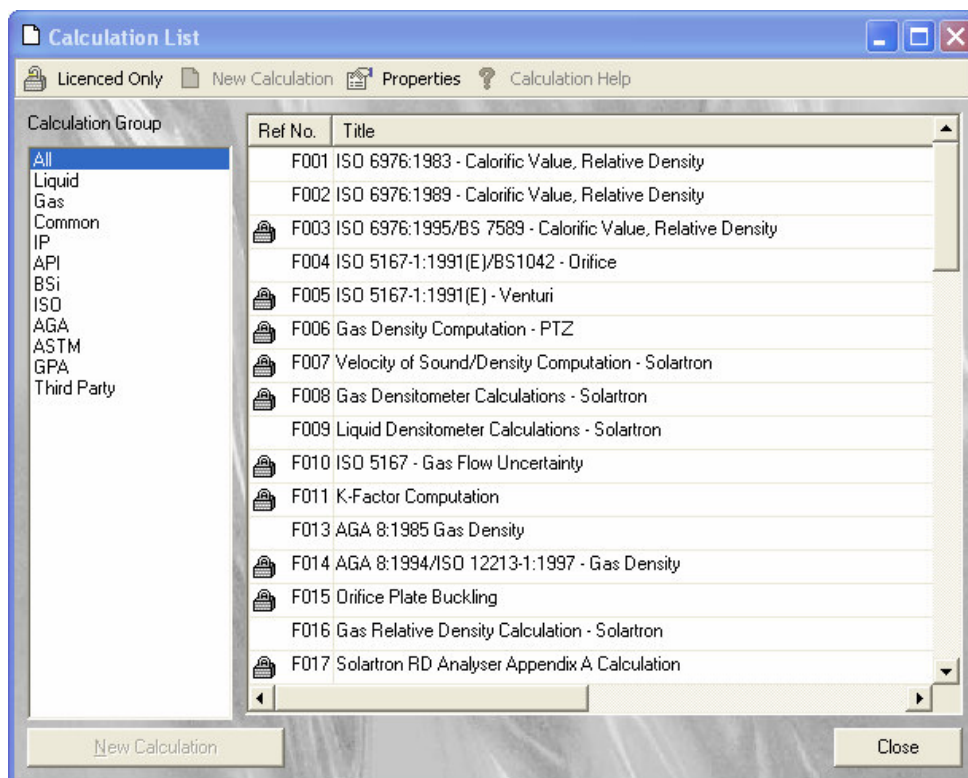


Figure 6.10 Calculation List Interface

A mix of both static and dynamic interfaces can also occur.

For example the FLOCALC[®] Calculation interface is created dynamically depending on the selected KCCL Calculation and the options chosen, but also contains a static header user control, see Figure 6.11.

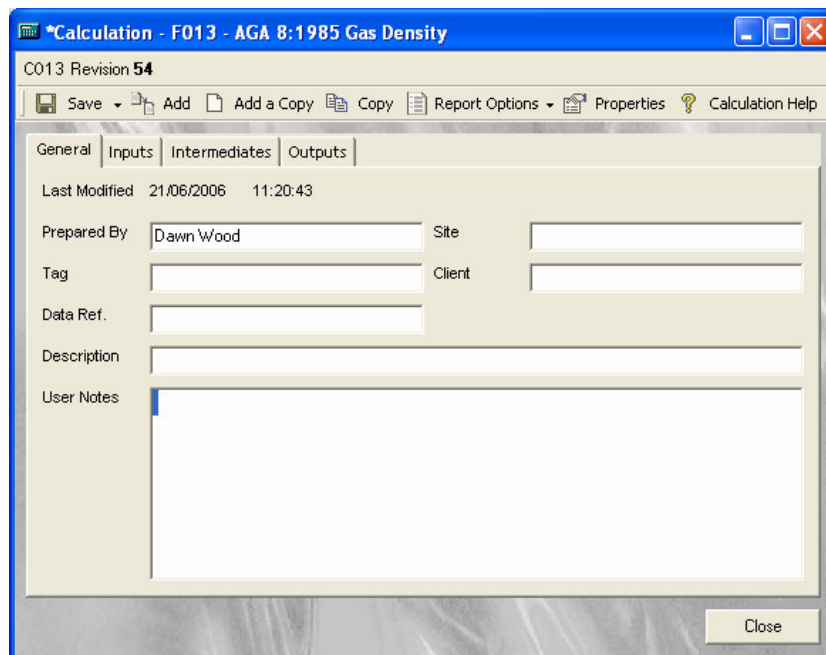


Figure 6.11 Calculation Header Interface

The development and specifications of user Interfaces are discussed further in sections 7.2.1 and 7.3.2.

6.5.4 Limiting Access

Access from the UI Logic to the UI Presentation Layer is limited to ensure data handled within the UI Layer is protected and only modified by the user where appropriate. For example, input values in a FLOCALC[®] calculation can only be modified within the UI Presentation Layer.

The UI Logic needs to initialise the interfaces with data. This can be done by creating a public Init subroutine which is only accessed during start-up. This means objects, such as a calculation object, cannot be replaced with a different Core calculation object at a later stage as this would lead to corruption of the data within the interface. An example is in Figure 6.12.

```

Private m_oCoreCalculation as FLOCALC_Core.Calculation

Public Sub Init(oCalculation as FLOCALC_Core.Calculation)
    Set m_oCoreCalculation = oCalculation
    pInit 'Initialise the UI components
End Sub

```

Figure 6.12 Calculation initialisation code

There is also the possibility of only allowing an object to be set once, as illustrated by the second example shown in Figure 6.13.

```

Private m_oCoreCalculation as FLOCALC_Core.Calculation

Public Property Set Calculation(oCalc as
FLOCALC_Core.Calculation)
    If m_oCoreCalculation is Nothing then
        m_oCoreCalculation = oCalc
    End if
End Property

```

Figure 6.13 Calculation property code

It is possible that a value or object needs to be changed at run time. In the case of FLOCALC[®], a calculation can be part of a workbook. A calculation can be removed / added from the workbook. The calculation user control needs the workbook object to be updated when this occurs. This is done by setting the workbook object to null or to a workbook object in the calculation interface code, as shown in Figure 6.14.


```

Private m_oCoreCalcWorkbook as
FLOCALC_Core.CalculationWorkbook
'setting a workbook object that the calculation object is part
of
Public Property Set Workbook(o as
FLOCALC_Core.CalculationWorkbook)
    If m_oCoreCalcWorkbook is Nothing then
        m_oCoreCalcWorkbook = o
    End if
End Property

```

Figure 6.14 Setting a workbook object in a calculation user control

6.6 Business Logic

The business logic implements domain-specific functionality and handles data persistence. The code for this layer could be incorporated in the UI Presentation Layer for the standalone applications (or the middleware layer in the web applications) but it is preferable to create the layer independently in its own program to ensure consistency, to prevent duplication of code and for compatibility between different types of applications.

The UI Presentation Layer can have direct access to data when there is no need for storage of requests. This would be the case for the Unit Converter where the unit does not need to be saved for future use and additional information does not need to be saved about the request.

The business logic stores information about data such as a unique identifier for allowing multiple objects to be created of the same data. Other

information may need to be stored such as header information about the creator of a calculation, options chosen and inputs which can be stored in the data layer.

The business logic was implemented using an object model developed to incorporate data and other information such as details about the user session. The object model is discussed in Chapter 9.

6.7 Model View Controller

A related approach to the standalone layers described in this chapter is the widely used MVC [37] software architecture for interactive applications, which is a method of separating an application, or an application's GUI, into three parts:

- Model
- View
- Controller

Each layer handles specific tasks and has specific responsibilities to the other areas, as illustrated in Figure 6.15.

A model represents the business logic and data. It provides access for the controller to the application functionality encapsulated by the model. This layer is represented by the business logic layer used in this project (see section 6.6).

A view displays the contents of the model and has access to the data in the business logic. This layer is represented by the UI Presentation Layer (see section 6.5).

A controller allows users to request and select views for presentation. It interprets user inputs and states actions to be performed by the model for tasks such as button clicks. The work done by this layer is shared by the UI Logic (see section 6.4) and the UI Presentation Layer (see section 6.5).

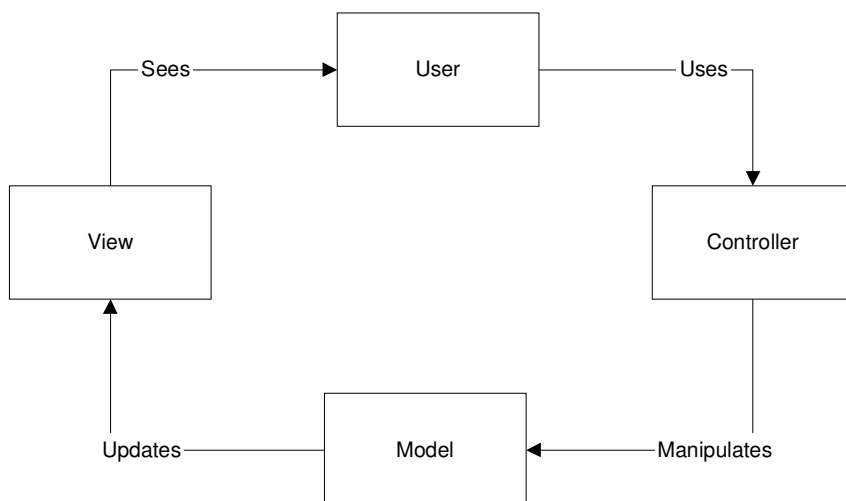


Figure 6.15 Structure of MVC Pattern

7 Software Design - Software Specifications

7.1 Introduction

This chapter discusses the specifications of each software package designed / redesigned in this project based on the findings of the Market Survey, namely:

- Flow Calculation Packages
 - i.FLOCALC[®] - Web Version
 - FLOCALC[®] - Standalone Version
- Interactive Training Packages
 - i.KITS[®] - Web Version
 - KITS[®] - Standalone Version

The discussion covers the specification for web and standalone applications.

The chapter reviews the user requirements, functionality and implementation documentation created as part of the project. The implementation documentation includes the object model and the n-tier [3] structure for the software, which is discussed further in Chapter 9.

7.2 Common Requirements

This section looks at the generic requirements for all KELTON[®] applications developed by KELTON[®] and used within the project applications.

7.2.1 Licensing Requirements

Licensing is required to limit access to software and data. This ensures that users get what they have paid for but no more.

In the past, users licensed KELTON[®] software by telephone or email to obtain a license number for access to software. With all KELTON[®] software, online licensing has been introduced to make licensing easier for users, see Figure 7.1. The data collected from licensing is stored at KELTON[®] and is useful for KELTON[®] so that they know what version a user has, especially when the user contacts KELTON[®] with a problem.

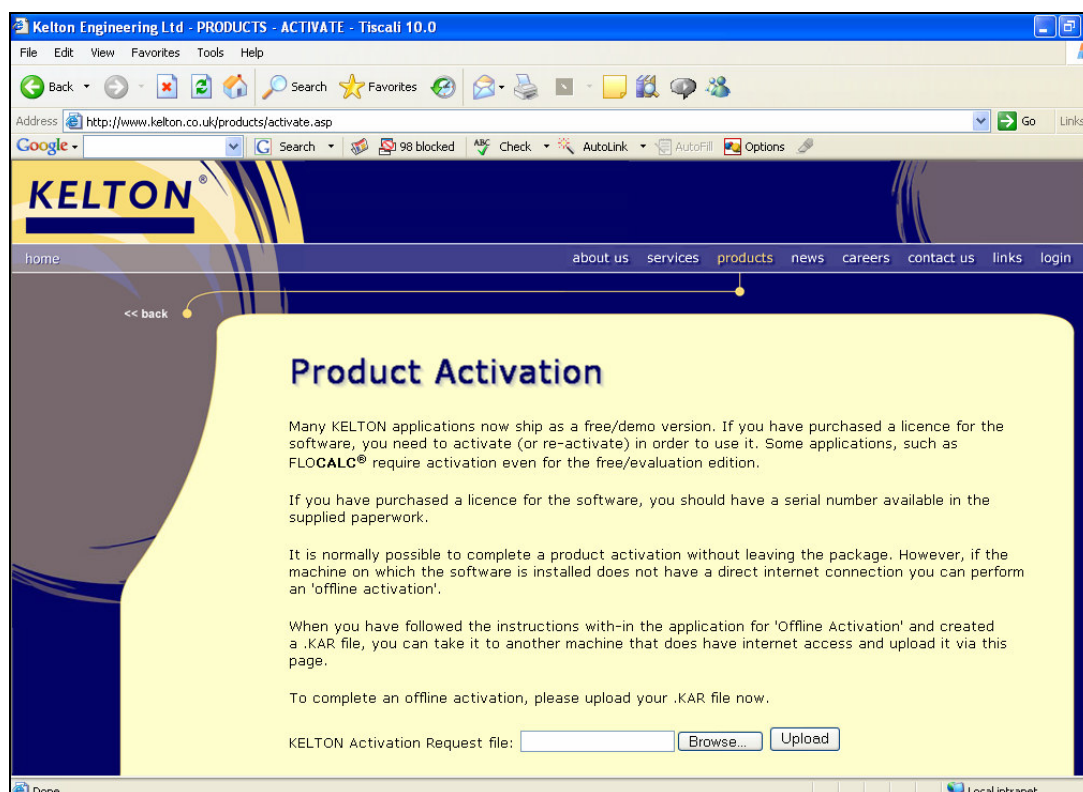


Figure 7.1 KELTON[®] Online Activation

The license data is an encrypted file created by obtaining the user's machine ID and user name. This prevents users from copying the licence on multiple machines. Online activation requires the user to log in, checks the email address and activates the software on a given PC.

Standalone licenses can be used on two machines - a desktop and a laptop. The web version is licensed for a number of concurrent users using a server machine ID. This means that the licenses are floating whereas the standalone licenses are licensed per user, therefore fixed.

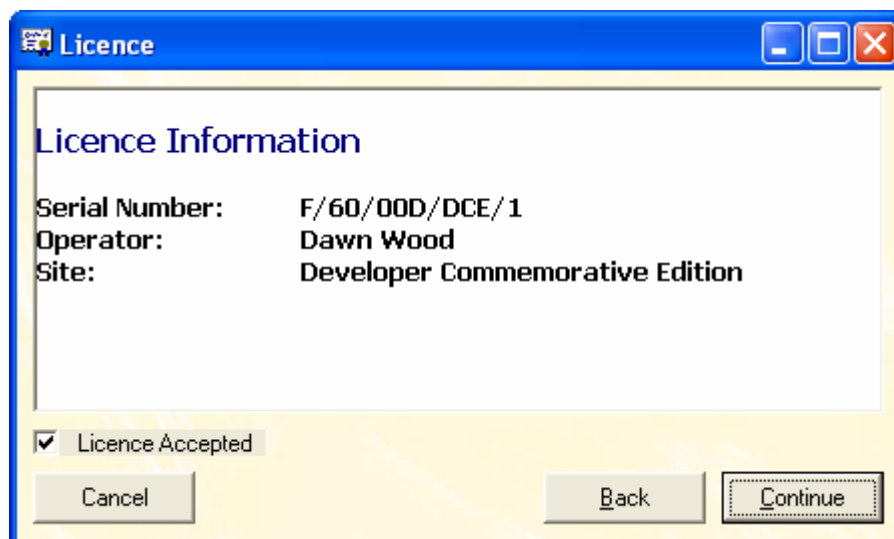


Figure 7.2 License Information

7.2.2 Standalone Interfaces

The standalone interfaces incorporate KELTON® common components such as text boxes. These components incorporate all the base functionality of the Microsoft components [27] plus company-specific extensions. Using these

components creates the impression of a consistent look and feel for all the standalone applications, see Figure 7.3.

7.2.3 Web Interfaces

The web interfaces use HTML [24] components and a KELTON[®] CSS [4] file to ensure consistent rendering. There is also a frame XSLT [5] so that the appearance of each page is the same no matter what page is being viewed, e.g., start up page, FLOCALC[®] calculation page or KITS[®] training module page. The CSS file describes fonts, paragraph spacing, underlining etc. The XSLT file generates the page header, background and main menu/links, see Figure 5.3.

7.3 Flow Metering Calculation Specifications (FLOCALC[®])

7.3.1 Main Requirements

Previously, FLOCALC[®] included all the code required for calculations. In the redesigned FLOCALC[®] software, calculations are performed by KCCL (KELTON[®] Common Calculation Library). KCCL is a component developed by KELTON[®] and can include 3rd party calculations which are wrapped with KCCL by agreement with the end user. This means that FLOCALC[®] only ever accesses calculation data generated by KCCL.

KCCL implements the logic required to perform all calculations. In addition, the end user provides inputs and/or selects from lists of options which determine the response and outputs delivered by KCCL.

FLOCALC[®] allows the user to interact with KCCL. An example of the FLOCALC[®] option, input and output screens developed in this project for the pressure calculation (gauge and absolute) is shown in Figure 7.3.

The figure displays three screenshots of the 'Calculation - F018 - Pressure Calculation' software interface, showing different tabs of the same window.

General Tab: The window title is 'Calculation - F018 - Pressure Calculation - ...'. The version is 'C130 Revision 45'. The menu bar includes 'Save', 'Add', 'Add a Copy', 'Copy', and 'Report Options'. The 'General' tab is selected. It contains three dropdown menus: 'Pressure mode' (set to 'Gauge'), 'Reference standard' (set to 'Digital pressure indicator or gauge'), and 'Output type' (set to 'Digital'). There are also checkboxes for 'Calibration correction' (unchecked) and 'Correct for Head' (unchecked), with an 'Apply' button. A 'Close' button is at the bottom right.

Inputs Tab: The 'Inputs' tab is selected. It features a table with the following data:

Input Name	Value	Units
Applied pressure	0	Pa

A 'Close' button is at the bottom right.

Outputs Tab: The 'Outputs' tab is selected. It features a table with the following data:

Output Name	Value	Units
Pressure	-1.013250	bar g

A 'Close' button is at the bottom right.

Figure 7.3 FLOCALC[®] Calculation Interface

Additional information about a calculation is stored in the FLOCALC[®] Core business logic as follows: -

- Unique identifier (ID)

- Header Information
- Units for parameters (Inputs, Intermediates, Constants and Outputs)
- Display resolution (decimal places) for Intermediates, Constants and Outputs
- Report Templates

A FLOCALC[®] calculation object wraps a KCCL calculation and also stores the additional information about the calculation.

Calculation workbook objects are created to contain multiple calculations, header information and a unique identifier.

7.3.2 Interface Requirements

There was a requirement to develop two versions of FLOCALC[®]. A web based version accessible via a user company's intranet called i.FLOCALC[®] and a standalone version, which includes an option to purchase an Excel Add-In for access to KCCL. The Excel Add-In utilises the interfaces developed in the standalone version.

In addition it was required to list calculation details from KCCL. This included the KCCL title, revision number and date last revised.

Help files were created to show how to use the two different interfaces and designed to be intuitive depending on the users actions, e.g., working with calculation workbooks.

7.3.3 Calculation Requirements

There are currently 156 flow metering calculations available in KCCL of which 47 are available in FLOCALC[®]. Each of these calculations can be altered in FLOCALC[®] and in addition FLOCALC[®] is able to create multiple instances of a KCCL calculation, allowing it to be used simultaneously.

The software was redeveloped as part of this project to allow FLOCALC[®] calculations to be saved with the above information as a template (.fct file), calculation (.fcc file) or as part of a calculation workbook (.fcw file), which is a set of calculations. A template is saved as a FLOCALC[®] calculation without its ID so that it can be opened as a blank calculation that can be assigned a new ID and modified as required. A workbook can contain zero to many FLOCALC[®] calculations.

Calculations can also be imported into any of the interfaces created, web, standalone and Excel.

Due to the possibility of updates to the KCCL calculation library there may be incompatibility issues. This was solved by prohibiting a user from performing a revised calculation in conjunction with an “older” version of KCCL. There is however a facility to open the calculation in a report.

7.3.4 Design Specification

A modular design was preferred to facilitate code reuse between the different versions, e.g., standalone and web. This was accomplished by using an n-

tier [3] architecture, see Figure 7.4. The programs required were as follows:-
 (Note: All the programs were developed as part of this project with the exception of KCCL which was supplied by KELTON®.)

- KCCL

KCCL which contains the logic required for calculating flow metering calculations and unit conversions. KCCL may include 3rd party calculations such as PPDS [9] by wrapping them within KCCL. KCCL calculation objects are wrapped by the Core and available provided there is adequate licensing. KCCL is also supplied with an extensive help file for each calculation showing the standards used for user traceability.

- Core

The Core wraps KCCL and has all the business logic required for providing access to FLOCALC® calculation and workbook objects. The Core contains additional information about the object such as header details discussed above (also see section 7.3.1). Finally, it contains the common functionality required by the intranet and standalone versions, such as importing and exporting data.

- BES

The back end server program used in the intranet version accesses the Core by requesting calculation and workbook Core objects and updating them following user requests. The back end server responds to requests made by the front end server.

- FES

The front end server program used in the intranet version makes

requests to the back end server program. This program is accessed by the client and transforms data using XSLT [5] style sheets to present web pages to the client.

- WIN32

The WIN32 application deals with the common functionality required in the standalone versions such as opening a calculation, licensing and importing/exporting. The program also has code for accessing Excel. This program stores standalone interface objects and passes the Core calculation/workbook objects to the interfaces.

- Standalone

The Standalone program contains the interfaces required in the standalone version for displaying calculations, lists of calculations available and workbooks. This program also handles the user interactions with the Core calculation and workbook objects.

- Excel

The Excel Add-In creates a FLOCALC[®] menu in Excel to allow users access to FLOCALC[®] calculations within Excel. The Add-In uses the standalone interfaces as necessary and works by allowing users to link cells in a spreadsheet with the input and output values of a FLOCALC[®] calculation.

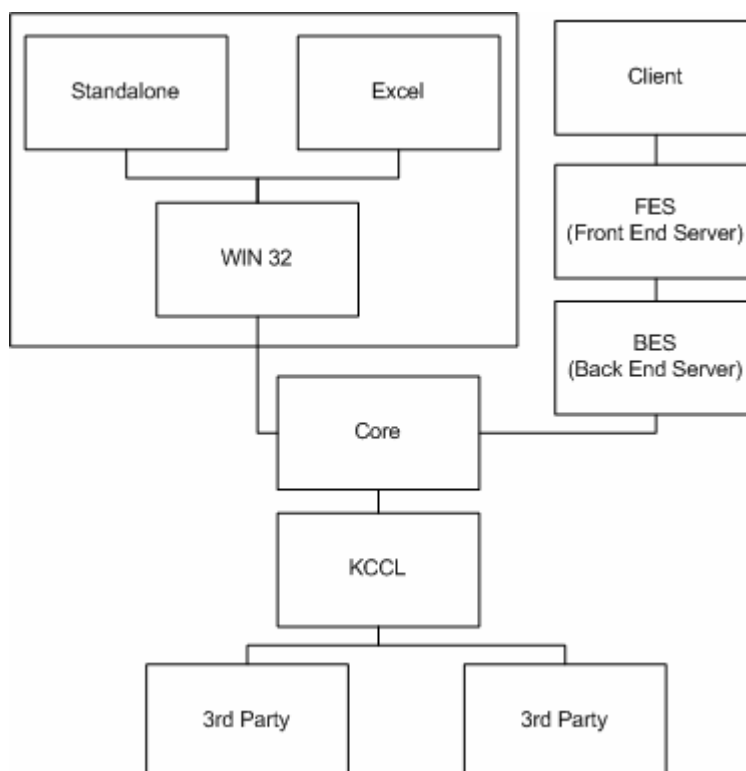


Figure 7.4 n-Tier FLOCALC® Software Architecture Model

7.3.5 Licensing Requirements

The standalone version of FLOCALC® is user specific and licensed to an individual PC. The web version is licensed to a web server with concurrent users.

There are various licensing requirements in terms of:

- Software version (standalone or web-based, see section 7.2.1).
- Level of functionality (standalone with or without an Excel Add-In).
- Access to KCCL calculations.

The Excel Add-In is licensed in addition to the standalone version.

FLOCALC[®] calculations are sold per calculation so it is necessary to know what calculations the user has purchased. KELTON[®] supply eight calculations for free but these are activated online. The user needs to register with KELTON[®] to allow KELTON[®] to know what each user has licensed and currently installed.

7.4 KELTON[®] Interactive Training (KITS[®])

7.4.1 Main Requirements

A KITS[®] library is organised to allow developers to create training libraries, courses and modules for the flow measurement industry.

Trainees can run modules and save training records on modules worked on. It is important that data is organised in a manner that allows a trainee to progress with a training module in a logical order but also so that training material is easy to find to allow reuse in other training modules. The KITS[®] library data is organised as follows: -

- Courses (contains in sequence Courses, Modules, Frames and hyperlinks)
- Modules (contains sequenced frames)
- Module Groups (contains Module Groups and Modules)
- Frames (contains documents such as RTF's, images, animations, audio or questions)
- Frame Groups (contains Frame Groups and Frames)

These all include a name and description to allow a developer easy co-ordination of data.

The training records are stored in the user database and can be transported in an encrypted file to other user database files, either within the same company but a different site or to another company. They log the name of a module completed so that they can be viewed in another location, even if they do not have the library.

Full administrator rights are available. This allows an administrator to set pass marks and specify maximum time limit that a pass is valid on a module, allowing different sites/companies to enforce different levels of competency. They can also set up other administrators and users in the system.

7.4.2 Interface Requirements

There was a requirement to develop two versions of the KITS®:

- A web based version accessible via a company intranet. This is limited to users running training modules and users with administration rights accessing training records to allow the setting of users and pass marks. This version was called i.KITS®.
- A standalone version which incorporated additional functionality to allow developers to create new KITS® libraries, courses, modules, etc.

Help files also needed to be created to show how to use the two different interfaces and be intuitive depending on what the user is doing.

7.4.3 KITS[®] Library

From KELTON's experience with the old version of KITS[®], it was found that keeping all the data in one library incurs a large storage penalty and is complicated to co-ordinate. The new software was developed to allow the use of multiple libraries which can be opened simultaneously and designated by a unique identifier.

7.4.4 Design Specification

As with FLOCALC[®] it was preferable to construct the applications as separate programs illustrated in Figure 7.5 to maximise opportunities for code reuse between different versions.

The programs required were as follows:

- Core
 - The Core has all the business logic required for providing the KITS[®] library objects. The Core has a collection of Library objects, therefore can have zero to many KITS[®] libraries open at any time. Each Library object contains Courses, Modules, Module Groups, Frames and Frame Groups (see section 7.4.1).
 - There is also an object containing training records.
- BES as per FLOCALC[®] (see section 7.3.4).
- FES as per FLOCALC[®] (see section 7.3.4).
- WIN 32

- The WIN 32 application deals with common functionality required in the standalone version for opening libraries and licensing. This program stores interface objects and passes library objects to the interfaces.
- Standalone
 - The standalone program contains all the interfaces required in the standalone version for working with libraries. There is an interface for running modules and for organising libraries in developer mode. This program also handles the user interaction with the Core library objects.

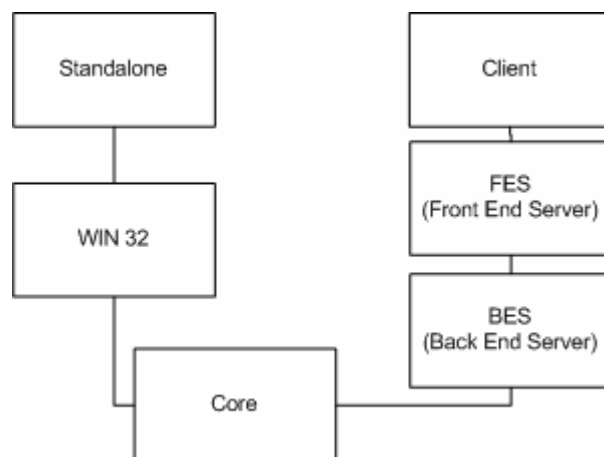


Figure 7.5 n-Tier KITS® Software Architecture Model

7.4.5 Licensing Requirements

KITS® can be user specific or site specific, e.g. sold to a department.

There are various licensing requirements in terms of:

- Software version (standalone or web-based, see section 7.2.1)
- Level of functionality (Developer, Standard or Player)
- Access to individual training modules.

The standalone version is licensed in 3 ways:-

1. A license for a Developer edition which allows the user to create their own Libraries, Courses and Modules.
2. A Standard edition which allows user administration rights for setting users and passmarks on modules.
3. A Player edition, which is usually purchased by users who only want a limited number of modules, especially if they have attended a training course at KELTON®. They are then able to review the module at a later date and complete the module to obtain the training certificate.

Modules within libraries are licensed. It is therefore necessary that users with a Developer license are given permission to use their eligible modules.

8 Software Design - Data Transfer between Interfaces

8.1 Introduction

This chapter reviews the requirements for data transfer between different programs in the n-tier [3] architecture structure (see sections 7.3.4 and 7.4.4) and ultimately how the data is presented in different interfaces.

8.2 Data Store

This layer deals with the storage of data. Data can be stored in a variety of methods using tools such as Structured Query Language (SQL) [49] Server, MS Access, MS Excel, text, XML [19] and many more.

To choose how data is stored, the following is considered: -

- Security
- Storage requirements
- Complexity
- Possible redundancy
- How the data is to be presented
- What the data is to be used for
- Validation requirements

8.2.1 Software Databases

For storage of large quantities of data it is normal to use a database. This means data can be controlled and normalised reducing duplication and

minimising the size of the database. Data can be accessed easily through the use of queries. The data is organised in tables with primary and foreign keys used as necessary in a conventional relational model.

When using a database, access to the data by the program needs to be considered. This is done by including a com object in the program such as Active X Data Objects (ADO) [38] or a Data Access Object (DAO) [39].

8.2.2 Alternatives to Database Software

Data can be stored in any format, e.g., image, animation, text, RTF, Excel, XML [19], etc. The data may even be generated through queries from a large database or by software written to generate files. By reducing the streams of data, software works faster by only passing data requested to the user.

8.2.3 Web Based Data Transfer Considerations

When delivering data through a web application, it is important to remember that the end user will not have direct access to a database. The speed of access and numbers of requests made to the server at any one time must also be considered. For the purpose of this project, only files containing the minimum data required were considered for passing to a client in the web versions. The files created in FLOCALC[®] are generated by the Core program and the files for KITS[®] are generated by the backend server. The data is created following an HTTP [29] request from the client by returning a file through a response object (see section 5.6.1, Figure 5.5).

8.3 XML

8.3.1 XML Data

XML [19] is a simplified subset of Standard Generalized Markup Language (SGML) [40]. XML is a W3C standard used extensively by the software industry. It is also a tree based data structure stored in a text format. Of the data types discussed previously, XML is the predominant choice for transferring data to the client of a web application.

Some of the key benefits of XML which made it an appropriate choice for this project are as follows:

- It can be read and processed easily by computers.
- It can contain any data type from multimedia data, such as images, to active components, such as Active X [27], and has no fixed set of tags, therefore, new tags can be created as required.
- Presentation is separated from the data which allows flexibility and means a client's corporate "look and feel" can be created without the need to alter the data. The data can be transformed using XSLT [5] to present it in a more readable format such as HTML [24] and validated using an XSD [7] file.

There are also disadvantages when using XML which make it less appropriate when handling some data within this project detailed as follows: -

- XML is not as compact as binary data, therefore is slower to transport and would be an inappropriate technology for transporting large amounts of data within this project.
- As XML is text based, it can be tampered with easily. This makes storing data in this format less secure.

An example XML Document is shown in Figure 8.1.

```
<?xml version="1.0" ?>
<Unit unitType="length">
  <Input type="meter" value="1"/>
  <Output type="centimetres" value="100" />
</Unit>
```

Figure 8.1 XML document

The Unit, Input and Output are all elements. The Unit element contains the attribute “unitType”. The Input and Output elements contain the attributes “type” and “value”. Elements are used to group data, attributes are used to hold values for data.

8.3.2 DOMDocument versus SAX

The data is stored using tags to identify data elements and attributes. Data can be parsed using a DOMDocument [6] object or a Simple Application Program Interface (API) [41] for XML (SAX) [42].

SAX is preferred when working with large XML documents as it starts parsing at the beginning of a document and raises events as it encounters various elements. Writing code using SAX is slightly more complicated as it is mostly

a set of interfaces rather than objects, which requires a set of classes to be developed to implement the interfaces. SAX is more memory efficient and only loads sections of the document as they are required.

By contrast, the DOMDocument object loads the entire document, which works well when working with small documents.

This project uses XML for transferring small amounts of information to the client, therefore the DOMDocument was sufficient for use in this project.

8.3.3 Schema

A schema document is required for validating XML using predefined rules. Two types of schemas used for validating XML are examined, DTD [31, 43] and XSD [7] [44].

An example of an XSD (Figure 8.2) and DTD (Figure 8.2) to validate an XML document (Figure 8.4) containing licensing details is illustrated.

```

<?xml version="1.0" encoding="iso-8859"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <!--Define the element in the XML document as License and
  Required -->
  <xsd:element name="License" use="required"/>
  <xsd:complexType>
    <xsd:sequence>
      <!--Define the attributes and the datatypes in the element
      License -- >
      <xsd:element name="serialNo" type="xsd:string"
      use="required"/>
      <xsd:element name="operator" type="xsd:string"
      use="required"/>
      <xsd:element name="site" type="xsd:string"
      use="required"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>

```

Figure 8.2 XSD document

```

<?xml version="1.0" ?>
<!DOCTYPE License [
  <!--Define the element in the XML document as License -- >
  <!ELEMENT License>
  <!--Define the attributes required in the License element-- >
  <!--CDATA represents character data and #REQUIRED is used to
  force a value to be entered -->
  <!ATTLIST License serialNo CDATA #REQUIRD>
  <!ATTLIST License operator CDATA #REQUIRED>
  <!ATTLIST License site CDATA #REQUIRED>
]>

```

Figure 8.3 DTD Document


```
<?xml version="1.0" ?>  
<License serialNo="F/60/00D/DCE/1" operator="Dawn Wood"  
site="Developer Commemorative Edition"/>
```

Figure 8.4 XML License details

The DTD schema describes the content and structure of an XML [19] document. XSD's have capabilities which include datatypes used to restrict the content that can occur within an attribute value such as integer. It also has more powerful element occurrence constraints such as minimum and maximum occurrences.

This project has used XSD [7] for validating data due to its increased capabilities over DTD as stated above. XSD documents have been used in FLOCALC[®] to validate imported calculation and workbook data and in KITS[®] to validate training records.

8.3.4 Transform

A transform can be implemented using either XSL or XSLT [5]. The latter works as a rule based programming language using templates to generate transformed documents. This is used for displaying XML in a user friendly way. In the case of the web based programs, the XML can be transformed into DHTML [16] using an XSLT. XSL can also transform XML into other formats, such as PDF, but this was not required for this project.

Figure 8.5 is an XSL document used to transform an XML document containing licensing details into HTML.

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output omit-xml-declaration="yes" method="xml"
encoding="iso-8859-1" />
  <!--Show the HTML document within the KELTON® HTML frame-->
  <xsl:include href="frame.xslt"/>
  <!--Get the element License -->
  <xsl:template match="License">
    <!--Create an HTML Header, paragraph and table-->
    <h1>FLOCALC<sup>&#0174;</sup> V3.0 FLOCALC</h1>
    <p><h3>Licence Information</h3>
      <table border="0">
        <!--Display the values for the attributes operator, site
and serialNo -->
        <tr><td>Operator</td><td>:</td><td><xsl:value-of
select="@operator" /></td></tr>
        <tr><td>Site</td><td>:</td><td><xsl:value-of
select="@site" /></td></tr>
        <tr><td>Serial No.</td><td>:</td><td><xsl:value-of
select="@serialNo" /></td></tr>
      </table>
    </p>
  </xsl:template>
</xsl:stylesheet>
```

Figure 8.5 XSL document

8.3.5 Off The Shelf XML Software

Creating transforms and schemas can be very complex when developing code using traditional methods such as Notepad.

XML [19] documents are generated by VB code at run-time. They are validated against an XSD [7] document and transformed by running code that creates an HTML [24] document using an XSLT [5]. For development purposes it was necessary to investigate how to create the schemas and transforms so that they would work correctly at run time.

XSLT's and XSD's are XML documents and can be broken down into smaller documents. This project used code to process XML documents with XSLT transformations and XSD schemas. This is done using the DOMDocument [6] component and can be run in the Visual Basic's Integrated Development Environment (IDE) [45] to check that the transform is working correctly and that inputted XML documents are valid.

There are problems debugging XSLT using the Visual Basic IDE as errors in the XSLT are not adequately picked up such as tags not being ended (Note: All XML documents such as XSLT must be well formed, with tags ended correctly). Errors in the XSLT are ignored and a blank HTML page is returned.

XSD's proved to be simpler to create and there was better error handling when using the DOMDocument object for the schemas as opposed to the

XSL(T) documents which provided no such error handling capabilities. In addition only documents imported to the project such as calculation workbooks need to be validated whereas all XML documents created need to be presented.

Attention turned to software that would ease creating and debugging XSLT's more efficiently. There are software packages available for creating transforms but it is important to know what the document will need to be able to do such as validating user input with the use of JavaScript [17], submitting user requests, the format that the transform should produce, e.g., HTML or DHTML [16], and the level of customisation required before purchasing and using the software for the development of the transforms.

Following research into the requirements of the web based applications, it was decided that software should be able to transform XML into DHTML. There was also a need to be able to manually change the rules in the transform for increased control. Debugging tools have to be available that can step through the code, view node values and get appropriate error messages if a failure occurs. After research, MarrowSoft Xselerator [46] software was purchased and used extensively.

8.3.6 XML within the Project

XML is used extensively as a means to pass information to the client in the intranet versions of the software by being passed to the FES dll [12] for display from the web server.

The XML file was generated by the web server as required, as illustrated in Figure 8.6.

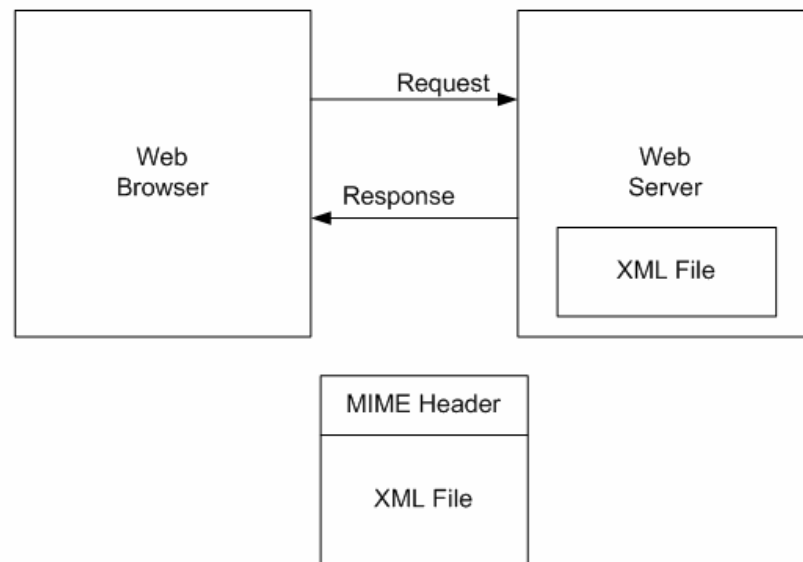


Figure 8.6 Transfer of XML using HTTP

The MIME [30] header content type used is “text/xml”.

There is also the ability to import and export calculation/ workbook information so that the data can be transferred from one format to another. After investigation of the different data types, it was decided to use the XML format for saving the files required.

KITS[®] also uses the XML format for displaying training records. This is appropriate as the file sizes are relatively small and used for display purposes only of a single user’s details. The training records are stored in an encrypted data file format to prevent tampering.

8.4 Additional Data Types Used

8.4.1 KITS[®] Data File

Data on users, training records, pass marks, usage and permissions need to be stored securely. From previous work done at KELTON[®] it was decided to use the Microsoft Joint Engine Technology (JET) 4 database engine [47]. JET was chosen because of ease of use, experience within KELTON[®], and the ability to encrypt/ protect data to prevent tampering.

8.4.2 KITS[®] Library

A file to store KITS[®] library information such as courses, modules and frames, etc was required. A frame is a page within a training module and requires the largest amount of storage. A frame incorporate files such as RTF and images used in the training modules. A module is made up of frames. A course is made up of modules and frames.

The KITS[®] library is too large and complex to be efficiently stored in an XML [19] format (see section 8.3.1). Also users must have appropriate licenses to access modules. Consequently, it was necessary to investigate other formats for storing the KITS[®] library data.

The KITS[®] library is stored as an Object Linking and Embedding (OLE) [48] structured storage file. This is a single file that acts like a file system and stores frame data in streams. This file type is not a relational database but record sets are stored. The reason for choosing this file type is that data is queried by searching through single tables rather than against multiple

tables. This means the added overhead of a database engine is not required for using SQL [49]. This makes the file smaller and faster to run. The file is protected against tampering and access to unlicensed data through encryption.

8.4.3 KCCL

KCCL does not require a data store as the data is generated through a Visual Basic dll in the form of calculation objects. The calculation objects are encapsulated within FLOCALC[®] calculation objects. There is however a licence file which enables the creation of these objects as required according to the current user licensing.

9 Software Design - Object Model Requirements

9.1 Introduction

This chapter discusses the object models developed and used by the business logic Core programs for FLOCALC[®] and KITS[®].

It also deals with ways that ensure data is passed correctly to the data layer and considers public properties, communications between individual components and handling of object pointers.

9.2 n-Tier Architecture

Software has been developed using an n-tier [3] architecture to ensure reusability of code. The WIN32 interfaces and the BES code both require access to the Core program, which holds the business logic of the application. (See section 7.3.4 for FLOCALC[®] and section 7.4.4 for KITS[®].)

The programs developed using the n-tier software architecture can be stored on different machines, see Figure 9.1. This means that using the Distributed Component Object Model (DCOM) [50], the UI and BES programs can communicate with the Core from other machines. Updates to the data in the Core program raise events of changes which the UI and BES programs detect so that each user sees changes immediately.

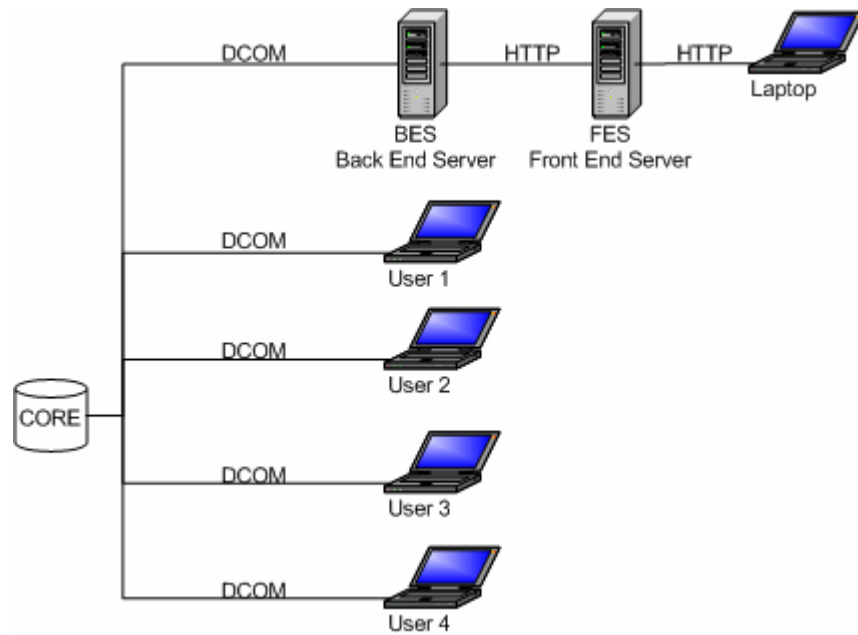


Figure 9.1 Server Communication with Core Data

9.3 Object Model

An object model was created to encapsulate the business logic of each application. The model specifies the objects required, their functions and collections. The model also defines the uses of interface objects.

The objects have been identified using Unified Modelling Language (UML) [51] to create object diagrams as these highlight errors in design before the code is started. The diagrams were also useful when working with the object model in the WIN32 programs and the BES as they show how the objects work together without the need to look up the objects in VB or by reference to the original code.

9.3.1 UML Diagram

Figure 9.2 illustrates how to read a UML diagram with reference to class objects used in the business logic.

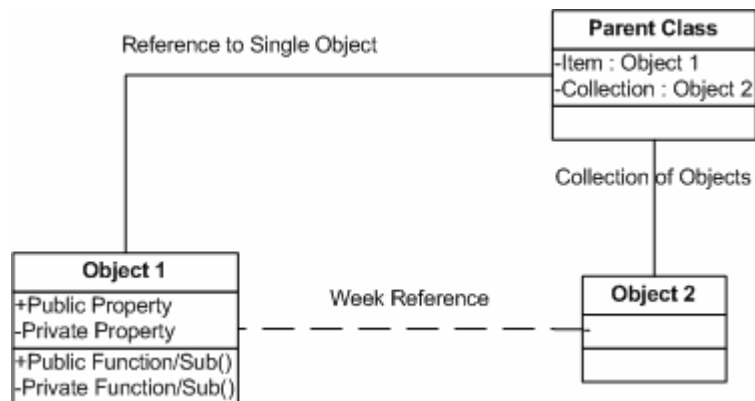


Figure 9.2 Basic UML Diagram

The parent class has a reference to two types of object. A single object is stored as a single item of type Object 1 (see Figure 6.3 for sample code). There is also a collection of objects stored of type Object 2 (see Figure 6.4 for sample code).

9.3.2 Top Level Application Object Model

The object model shown in Figure 9.3 was created to describe the top level objects in the Core application. This describes the main objects used in the business logic:-

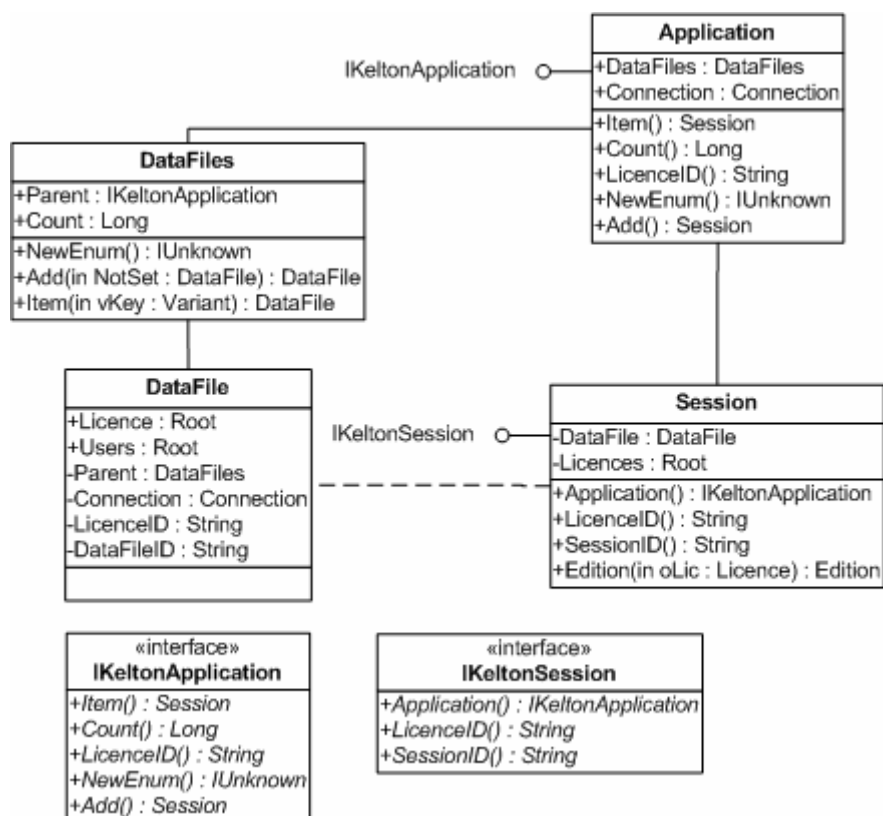


Figure 9.3 Top Level Object Model

The object model shows the relationship between the various objects required to run a KELTON[®] software application using KELTON[®] session and application interface objects.

The application object implements the application interface for storing licence and session objects.

It contains a collection of Session objects. The standalone version usually only requires one session but the web version can have multiple users simultaneously accessing the application, hence, there is a requirement for multiple sessions to be available. However it is sometimes possible to have multiple users in the standalone version using the same data from the Core

in an application such as KITS[®], where a training session is set up and multiple users log on to run training modules (see Figure 9.1).

The DataFiles object contains a collection of DataFile objects which store application specific data such as licence information and can also be used to store data such as user ID's and application records.

9.4 Core

The Core program contains the objects needed in the top level object model and provides the business logic required. The object model is utilised by the main application.

Two Core programs were created within this project, FLOCALC[®] and KITS[®] as described below.

9.4.1 FLOCALC[®]

The main objects need to be defined. An illustration of the objects required for a FLOCALC[®] workbook is shown in Figure 9.4.

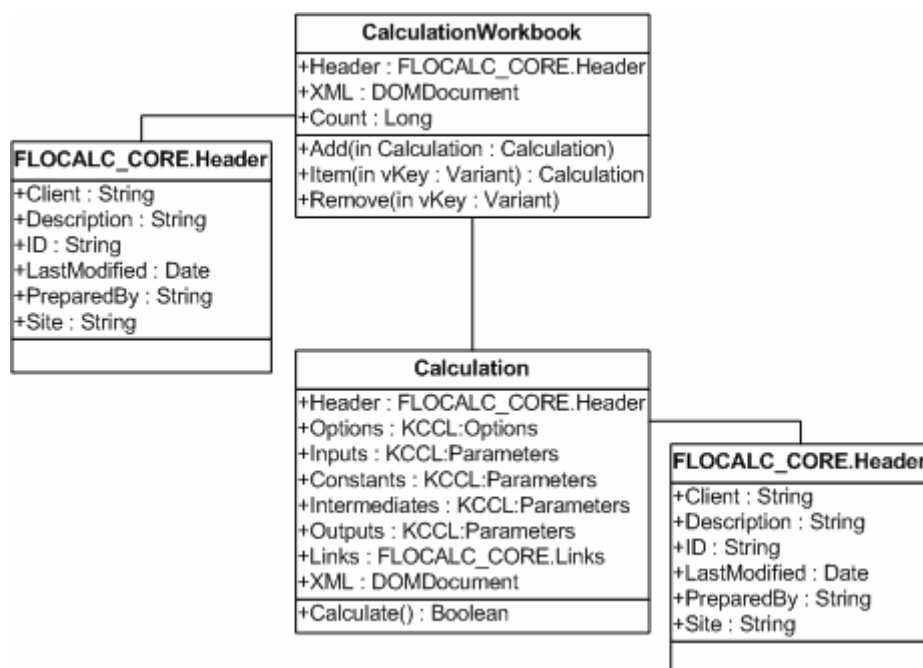


Figure 9.4 FLOCALC Calculation Workbook Object Model

The Calculation Workbook object stores information such as Client, Description and Site. There is also a collection of calculation objects within a workbook object. An XML [19] document is generated on request for exporting workbook data.

The Calculation object stores Client, Description and Site information, etc. A Header object is created which the Calculation and Workbook objects encapsulate to store header information.

Both the Calculation Workbook and Calculation objects return an XML document on request by generating XML using a DOMDocument [6] object. The XML is created by generating XML elements and attributes from the data stored within the object such as collections and properties.

An XML document is created for exporting calculation data and returned to the workbook XML property as needed. These are created in accordance with the XSD [7] instructions.

XML is read in by the DOMDocument object and used to create a calculation or workbook object. This is carried out through the session object. The XML document is validated against the XSD and then the appropriate object is created if valid.

A Calculation Workbook is made up of a number of calculations. This is handled by creating a collection of zero to many Calculation objects within the Workbook object.

The Calculation object stores KCCL calculation data which has parameters for Options, Inputs, Constants, Intermediates and Outputs.

9.4.2 KITS[®]

A brief outline of the library object is shown in Figure 9.5.

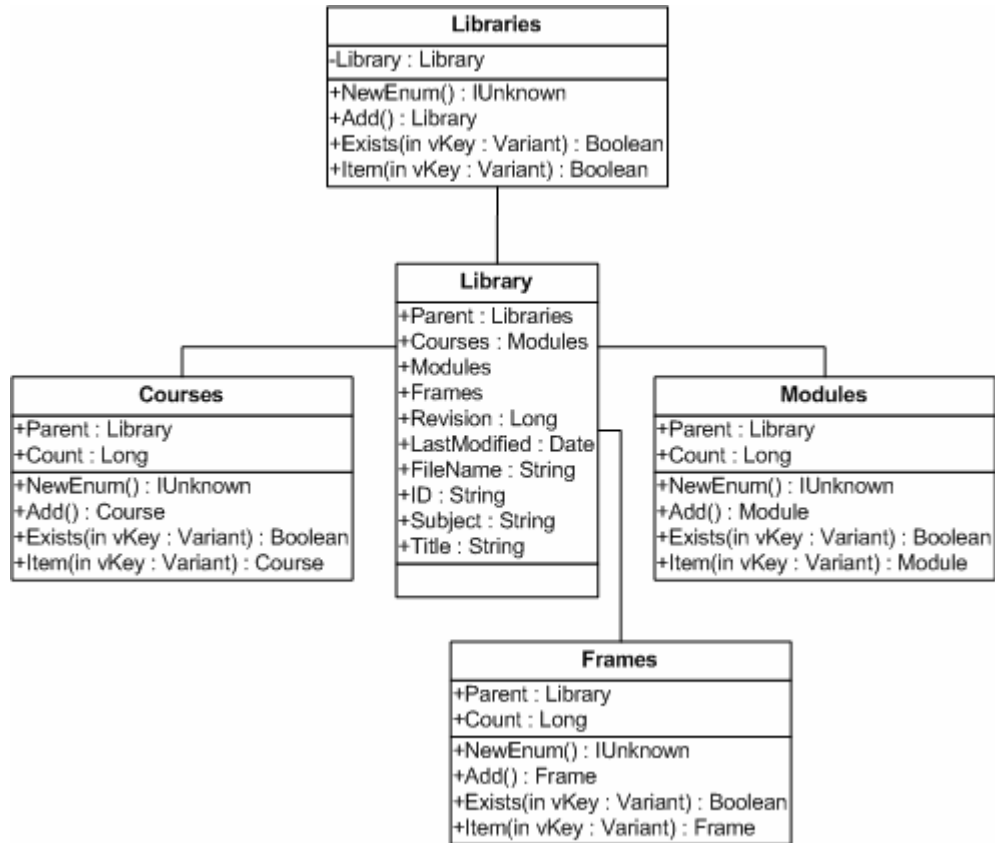


Figure 9.5 KITS® Libraries Object Model

The Library object includes a Courses object which is used to store a collection of course objects. The Library object also includes a Modules object and a Frames object to store collections of Module and Frame objects, respectively.

It is possible to have multiple KITS® library objects open and this is handled by the Libraries parent object. Each Library therefore needs to be uniquely identified and this is done with an ID.

The parent object is stored as a weak reference and a strong reference is created when required. Creating a strong reference from a weak reference is

not normally available in VB 6 but has been made available using advanced techniques [52]. The reason for using weak and strong references when pointing to the parent object is to prevent memory leakage as objects are deleted, e.g., deleting the Frames object from the Library object without having to first remove the strong reference to the parent.

10 Conclusions and Future Work

10.1 Introduction

This chapter details the main findings and conclusions from the project and discusses potential future work.

10.2 Market Survey

KELTON[®] gained an online marketing tool which can be used for gathering customer feedback on services and software products.

From the survey carried out in this project, KELTON[®] obtained a valuable insight into current market requirements. Analysis of the responses showed that industry had a preference for intranet based software. Prior to this, KELTON[®] had only considered expanding from standalone to internet based software. This demonstrates the benefit of obtaining customer feedback before large projects are undertaken.

Additional comments about products were received through the survey which KELTON[®] was able to act upon. This gave KELTON[®] more information on individual customers.

10.3 Web Based Applications

Based on the findings of the market survey it was decided to develop FLOCALC[®] and KITS[®] as web-based (intranet) applications.

From the development work carried out, the following advantages and disadvantages of web based software were identified: -

Advantages:

- The software does not need to be installed onto the user's PC, therefore bypassing Common Operating Environment (COE) issues which have become frequent in the industry.
- Users can have access anywhere provided they are connected to the network and have an internet browser.
- Licensing the software has meant that companies are purchasing floating rather than fixed licenses which has meant reduced outlay, making the package more attractive.

Disadvantages:

- The web version needs to make requests through servers and is therefore marginally slower than a standalone version.
- As the web applications are created using a thin interface, there is reduced "local" functionality.
- Users must have scripting turned on to increase speed and reduce the need to access the server due to the level of functionality created on the client side.

10.3.1 Benefits from Web Application Development

During the project, backgrounds and tools were standardised to create a corporate “look and feel” and are available for use in all KELTON[®] online applications. These were developed using XML [19].

Common functions were also created including login and administration functions.

10.3.2 i.FLOCALC[®]

KELTON[®] previously gave away free calculations on a standalone version of FLOCALC[®] which needed to be installed on a user’s PC. These calculations can now be provided by KELTON[®] on the web using i.FLOCALC[®], bypassing the need for users to install FLOCALC[®].

i.FLOCALC[®] was successfully developed and launched in November 2005 as part of this project and is now being marketed by KELTON[®].

10.3.3 i.KITS[®]

From the work carried out, it was concluded that i.KITS[®] has the most value to KELTON[®] as a web package out of all their applications. This is because users generally only require limited access to courses and modules while undergoing training and therefore do not need to store large course libraries or install software for viewing the information on their computers.

The design of i.KITS[®] as well as a large part of the development was completed during this project. The final completion and launch of i.KITS[®] is currently on hold due to other projects taking priority within KELTON[®].

10.4 Standalone Applications

In spite of the findings of the market survey it was decided to redevelop the standalone versions of FLOCALC[®] and KITS[®].

10.4.1 Benefits from Standalone Application Development

During the project, standardised interfaces were created for common software functions. These were developed as user controls using KELTON[®] components and made available for all KELTON[®] standalone applications. This included a Unit Converter created for FLOCALC[®] and available for other standalone applications such as KAMS[®].

It was concluded an n-tier [3] software architecture would be appropriate for developing the software to make it available for deployment by multiple methods, namely standalone and web. This meant that common code was shared efficiently between different applications and testing was made easier.

10.4.2 FLOCALC[®]

As the standalone interfaces for FLOCALC[®] needed to be available independently and also in Excel, it was concluded that the common standalone interfaces should be developed in a win32 program. The

standalone and Excel program could then make use of the common functionality and interfaces created in the win32 program.

FLOCALC[®] has been successfully redeveloped with additional functionality over previous versions such as export/import of calculations to enable portability of calculations between software.

The new version was launched by KELTON[®] in November 2005.

10.4.3 KITS[®]

KITS[®] has been successfully redeveloped and now allows data to be stored in multiple libraries, making it easier to handle data. Other improvements involved removing restrictions on naming data files stored in the libraries. Both these changes have reduced the likelihood of storing duplicate training materials. Also training material only needs to be updated once.

The preliminary research, design and a large part of the redevelopment for KITS[®] was carried out as part of this project. The redevelopment work was completed by KELTON[®] and the software is currently in the final phase of testing. The redeveloped KITS[®] is expected to be launched during 2007.

10.5 Other Key Findings

10.5.1 Common Code

From the work performed, the n-tier software architecture worked well and reduced the need to rewrite common code. However, there were also

drawbacks in that changes to common code can be made by developers which may affect other applications.

An example of this problem occurred when using the common KELTON[®] MDI [53] form in FLOCALC[®]. The MDI common code was updated to handle logging into KELTON[®] applications and this is called at start-up. FLOCALC[®] does not however require a user to log in so the common code had to be adjusted to accommodate this. The code should have been tested to ensure it still worked on all the KELTON[®] applications using the MDI. These kinds of changes can be missed if the application is not correctly tested whenever a new deployment of the software is created.

10.5.2 AJAX

During the latter stages of the project, use of AJAX [28] for loading small XML [19] documents to update existing web pages was investigated. It was found to increase speed when updating pages which would otherwise require to be either re-rendered or created with larger XML documents.

10.5.3 Testing

FLOCALC[®] and i.FLOCALC[®] were tested to ensure the software worked as expected. The testing was carried out by KELTON[®] engineers and signed off by the responsible senior flow engineer before the software was marketed. Throughout the testing process, feedback was encouraged to help improve the software.

The first phase of testing was to make sure all the functions worked correctly and the software performed as expected. Using the software functional specification (SFS) document, a test procedure was created with tables for each interface and test to be performed. A sample of the test procedure is shown in Table 10.1.

Table 10.1 FLOCALC® Test Procedure

<u>FLOCALC® V3.0 Web Edition Testing Procedure – General</u>			
<u>Functionality</u>			
-			
<p>Please save the document in the FLOCALC project Test Results\</p> <p>Name it as FC3W General Testing with your initials i.e. FC3W</p> <p>General Testing AB</p> <p><u>Test Carried out by:</u></p> <p><u>Calculation List Screen (Main Menu)</u></p>			
Test To Perform	Result Pass/Fail	Date	Comments
Open a calculation	Pass	01/11/2004	Opened F001 to F010 successfully. F011 was locked.
Calculation groups available such as AGA, common and All	Pass	01/11/2004	Found all calculation groups and checked that the AGA group had all their calculations listed.
Create new calculation set	Pass	01/11/2001	Opened an empty calculation set on a new popup page.

Each stage of testing needed to have all items passed before the next phase of testing could be started. If an item failed, the software was corrected and the testing restarted.

After the first phase of testing was passed, FLOCALC[®] was deployed onto KELTON[®] engineers' PCs and laptops for them to use in the work place and test against the old version of FLOCALC[®]. This was carried out to determine if the new version of FLOCALC[®] performed in a real engineering environment. The KCCL calculations were also tested and signed off by the responsible senior engineer.

Only when all the testing was completed and signed off by the responsible senior engineer was the software marketed to industry.

The training software (KITS[®] and i.KITS[®]) was not completed and hence not tested during this project (see section 10.8.2 for details on work completed).

10.6 Benefits to Industry - FLOCALC[®] and i.FLOCALC[®]

From the user feedback received the following benefits have been found: -

- No other flow calculation software has as comprehensive a calculation list.
- The help file provides traceability and transparency for the calculations.
- The interface is easy to use.
- Calculations can be shared and are easy to use.

- The application is now been used by so many people that KELTON® believe it to be the industry standard.
- FLOCALC® is flexible and new / updated calculations can be added quickly. For example, the DTI [10] issued a directive to industry to use a new calculation and it was included in FLOCALC® within one month.

KELTON® also benefited from the new software as it aided the development of their UNCERTAINTY PLUS™ [8] program. The program utilises KCCL which had been made more accessible through FLOCALC®.

Prior to the web development, KELTON® sold calculation software to work on standalone PCs. They have now included i.FLOCALC® on their web site to provide free calculations to their registered users. This has resulted in users being able to access the calculations from anywhere with internet access without the need to deploy software onto their hard drive.

From the new software, KELTON® anticipates a 10% gain in new markets and 20% increase in sales.

10.7 Future Work

10.7.1 FLOCALC®

From the project work and attending a presentation on MathCAD it was observed that flow metering engineers use MathCAD heavily and would be interested in using FLOCALC® through MathCAD. It is recommended that

this is investigated to check the feasibility and likely value to the industry of having a MathCAD Add-In within FLOCALC[®].

Engineering values are commonly expressed in scientific notation which was not possible in HTML [24]. It is therefore recommended that displaying numeric values using a scientific display in HTML is researched.

10.7.2 KITS[®]

Currently the main user of KITS[®] is KELTON[®] through in-house training courses run for industry. The developer edition could prove beneficial not only to the flow metering industry but to any organisation which has regular training courses.

It could be used as an alternative to PowerPoint as not only are the training materials presented in a professional manner but materials can be stored and easily reused in other training modules. There is also the ability to create questions which would give the trainer an opportunity to test the trainees understanding of the subject and produce certificates on completion of training courses and modules.

It is recommended that the potential for KITS[®] to be used directly by third parties be explored further.

10.7.3 KAMS[®]

Following the market survey and discussions within KELTON[®], it was decided that development of a web-based version of KAMS[®] was not a priority for this project.

Discussions have indicated that a management system of i.KAMS[®] used to generate reports could have potential and it is recommended that this be explored further. However, full functionality would be unlikely due to the sensitivity of sites where KAMS[®] is used offshore.

10.7.4 KIMS[®]

From the work performed it was decided not to progress the development of a web-based KIMS[®] package as it is most commonly used in remote locations where no network connections are available due to the proximity to sensitive instrumentation.

However, due to the level of industry interest found in the market survey, it is recommended an investigation be carried out to check the feasibility of developing a web based version of KIMS in the future.

10.7.5 AJAX

AJAX [28] was found to provide benefits when developing i.KITS[®] in terms of speeding up changes made to DHTML pages. It is considered similar benefits would be realised if AJAX was used in i.FLOCALC[®] especially in updating values when units are changed.

10.8 Summary

10.8.1 Flow Calculation Software

In this project, FLOCALC[®] was successfully redeveloped and launched to run on a standalone PC, with an option of running in Excel. Since the launch, the new version of FLOCALC[®] has attracted many new users.

i.FLOCALC[®] was also successfully developed and launched as a web application. This has resulted in an increased product range for KELTON[®] which has increased their market share in flow metering calculation software.

The new software makes use of KELTON's common calculation library (KCCL) for performing all the calculations, avoiding duplication and enhancing traceability.

The main challenge faced with the development was securing engineering time for testing to ensure FLOCALC[®] and i.FLOCALC[®] performed correctly and that the KCCL calculations were giving correct answers. Following regular discussions of this issue at project meetings, testing resources were eventually allocated to ensure the software was completed and launched before the end of the project.

10.8.2 Training Software

Training software was partly developed in this project to work on an intranet as i.KITS[®]. KITS[®], the standalone version (all three editions), was partly

redeveloped to use enhanced data types for training modules to give a more up-to-date feel. It was also redesigned to handle data more effectively.

KITS[®] and i.KITS[®] were not fully completed during the project due to changing priorities within KELTON[®].

There was work to be done in the Core program to allow i.KITS[®] to run multiple sessions and the data file object was a work in progress.

The WIN32 program contains the interfaces required for running on a standalone PC. The prototype for the player edition was completed. The standard and developer editions were still in progress.

The BES and FES design and code had most of the administration functions written to allow setting of users and training pass marks. There was also the ability to view training courses and modules. Work was still being done for transporting data such as graphics and text from the BES to the FES. Saving/loading training records had still to be started.

Appendix A – Market Survey Report



Client : **KELTON®**

Project Title : **KELTON® KTP**

Document Title : **Analysis of Market Survey for Web Based Flow Metering Software**

Document Ref. : **K1040K-001**

REV	ISSUE DATE	DESCRIPTION	PREP. BY	APP BY	APP . BY
0		Document Development	DW		
1.0	24/3/04	Update to Contents from Comments by CG	DW		
1.1	25/3/04	Update with Comments by CG and change Graphs in Section 7.0	DW		
1.2	26/3/04	Update with comments by GT	DW		
2.0	1/4/04	Approved	DW	CG	GT

KELTON® KTP**ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE*****CONTENTS***

1.0	Introduction.....	3
2.0	Conclusion/Recommendation	4
2.1	Conclusions	4
2.2	Recommendations	5
3.0	Mail Shot	6
4.0	Exsisting Users.....	6
5.0	Key to Market Survey	7
6.0	Market Survey Responses.....	8
6.1	Q1 Which Tools Do You Have or Would Like to Have?.....	8
6.2	Q2 Is there a need for Fully Functional Web Based Software?	10
6.3	Q3 Is there a need for Management Functions Only Web Based Software?.....	10
6.4	Q4 How Would the Software Be Delivered?	12
7.0	Deployment of software	13
7.1	Internet	13
7.2	Intranet	14
7.3	Local.....	14
7.4	Unknown	15
8.0	Details of deployment analysis	16
8.1	Cumulative Positive Responses	16
8.2	Positive Responses per Question.....	17
8.3	Correct Responses for Inconsistent Answers.....	18
Appendix A	Survey Response Numbers	19
Appendix B	Market Survey Questionnaire	20
Appendix C	Comments Recieved	22

KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING
SOFTWARE

1.0 INTRODUCTION

A Market Survey has been conducted to find client requirements of web based flow metering software. This will be used as part of the decision process for the update of Kelton's current software. This has been done as part of the (Knowledge Transfer Partnerships) Program 4243.

The purpose of this report is to present the analysis of the survey which has run from the 1st March 2004 until the 22nd March 2004.

The TCS (Teaching Companies Scheme) Program Grant Application and Proposal form, now known as KTP shows the original requirements for the Web Based software.

The market survey program has generated a set of files, containing client requirements for web based flow metering software.

This document describes the results of the market survey, with 49 responses.

KELTON® KTP ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE

2.0 CONCLUSION/RECOMMENDATION

2.1 Conclusions

The TCS Grant Application and Proposal form, now known as KTP shows a proposal for web based software.

- The application specified that FLOCALC®, K-TRAC®, KAMS® and KITS® are to be changed to web based software
- FLOCALC® and K-TRAC® being created in Phase 1 of the project
- KAMS® and KITS® in phase 2 of the project.

From the results there is an indication that intranet based software would be preferred over internet and local. The software that has had the biggest response has been FLOCALC® and KIMS® which was not initially considered, K-TRAC® and KITS® also had a reasonable response.

- In section 6.0, FLOCALC® and KIMS® software packages have been chosen the largest amount of times, 82% of clients replied that they have or would like to have FLOCALC® followed by 51% for KIMS®.
- For the fully functional packages, 43% replied FLOCALC® and 35% replied KIMS®.
- For the reduced management function software, 29% of clients replied FLOCALC® and KIMS® if the fully functional package was unavailable.
- KIMS® and FLOCALC® are the most popular software sold by KELTON®, see section 4.0 for more details.
- The intranet was chosen by 53% of clients asked followed by 24% for the local option, see section 6.4.
- FLOCALC® has the most positive responses for internet based software, section 7.1.
- The intranet graph in section 7.2, shows that clients would be interested in management functions for KIMS®, FLOCALC® followed by K-LOG® and KAMS®.
- The internet option in section 7.1 showed more interest in FLOCALC® and K-TRAC® management functions compared with the other software.

This is from a mail shot response of 3% completing the market survey.

KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING
SOFTWARE

2.2 Recommendations

It is recommended that FLOCALC® and KITS® are developed with FLOCALC® developed as a local standalone, excel and web application using KCCL.

KELTON® KTP ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE

3.0 MAIL SHOT

The market survey was added as a news item on the KELTON® web site on 1st March 2004. A mail shot directing Kelton's clients to the news item was sent out on this date also.

The details for the number of mail shot as of 24th March are as follows: -

Total Sent (Clients)	1729
Total Delivery Receipts	72
Total Reported Errors	130
Total Out Of Office Replies	40
Total Received	1599
Total Survey Responses	49

The market survey has had 1599 emails that have not errored, therefore there has been a 3% response rate.

4.0 EXSISTING USERS

Listed below is the number of software packages currently owned/used by Kelton's clients. The most popular software sold being KIMS® (36%) and FLOCALC® (33%).

Software	Number sold	% Sold	Company Contacts	Actual Customers	Customer % of Clients
K-LOG®	2	1%		2	0%
K-TRAC®	29	8%		29	2%
KIMS®	126	36%	37	76	4%
KAMS®	16	5%	15	16	1%
FLOCALC®	117	33%	1615	1615	93%
KITS®	58	17%	195	58	3%
K-VIEW®	3	1%		3	0%
Total®	351				

KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING
SOFTWARE

5.0 KEY TO MARKET SURVEY

The graphs use the values from the market survey as follows: -

Graph	Question
Q1	Which of the following software tools do you have or would like to have?
Q2	Do you or your company have a business need for a fully functional intranet, extranet or internet based version of these tools?
Q3	Do you or your company have a business need for management functions via an intranet, extranet or internet, i.e. reporting and data analysis?
Q4	Where would you envisage the software residing?

Question 1 to 3

Graph	Question
KIMS®	Instrument Management & Flow Computer Validation
K-TRAC®	Turbine Meter Performance Monitoring
KITS®	Interactive Training
K-LOG®	Electronic Logbooks
FLOCALC®	Engineering Flow Calculations
KAMS®	Audit Management System
NONE	None of the above

Question 4

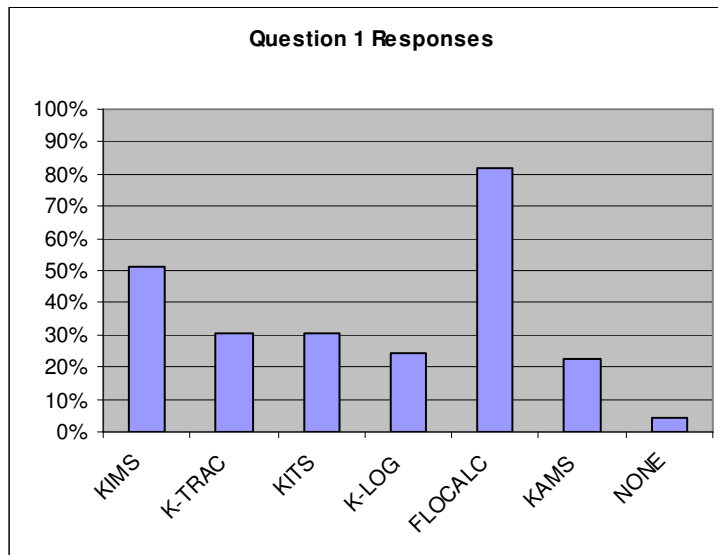
Graph	Question
Internet	KELTON® hosting the software on its servers, accessible by anyone with the right authorisation from anywhere with an internet connection, including all management and updating of the system
Intranet	Your company hosting the software on its servers, accessible by anyone with the right authorisation and a connection to that specific server.
Local	Your local machine to allow you to run offline with a backend hosted at a KELTON® or your company server
Unknown	Don't know

KELTON® KTP**ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE****6.0 MARKET SURVEY RESPONSES**

This section shows the results from the market survey for web based flow metering software. The following tables show the number of positive responses and percentages out of a total of 49.

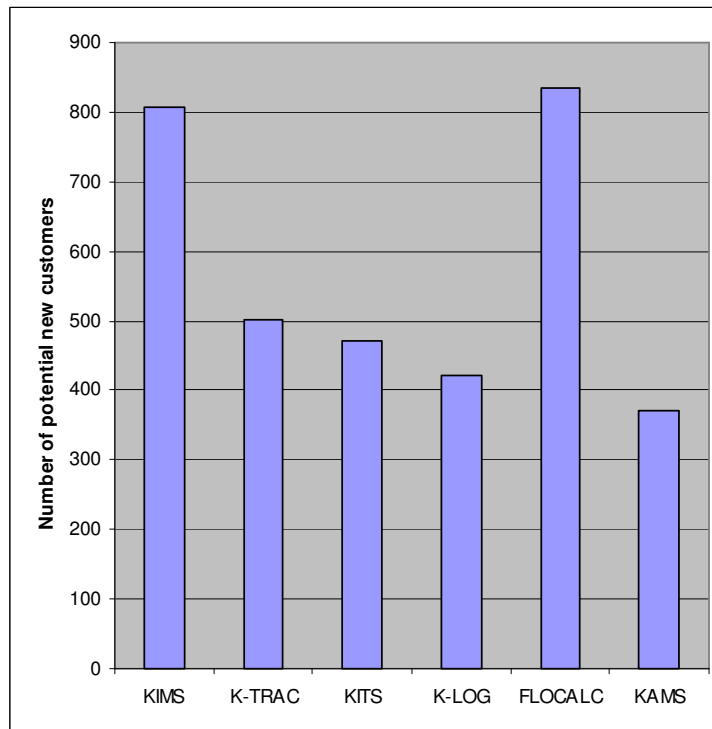
6.1 Q1 Which Tools Do You Have or Would Like to Have?

	KIMS®	K-TRAC®	KITS®	K-LOG®	FLOCALC®	KAMS®	NONE
Positive Responses	25	15	15	12	40	11	2
% of survey	51%	31%	31%	24%	82%	22%	4%
Actual Customers	4%	2%	3%	0%	33%	1%	
Potential new customers	806	500	471	421	835	372	



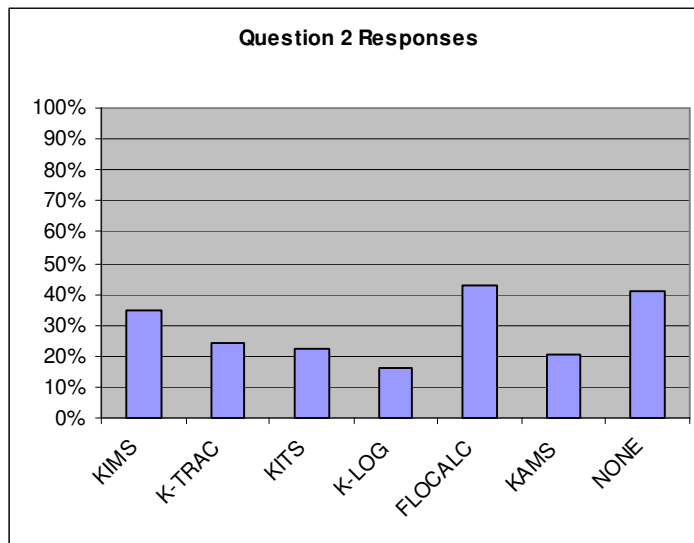
KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING
SOFTWARE

The following shows the number of possible new clients by comparing the number of existing clients with the number of people responding that they have or would like to have the software.



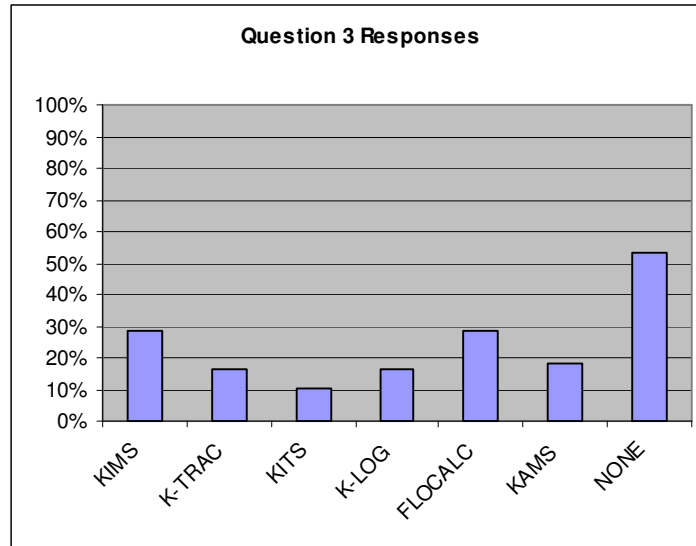
KELTON® KTP**ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE****6.2 Q2 Is there a need for Fully Functional Web Based Software?**

KIMS®	K-TRAC®	KITS®	K-LOG®	FLOCALC®	KAMS®	NONE
17	12	11	8	21	10	20
35%	24%	22%	16%	43%	20%	41%

**6.3 Q3 Is there a need for Management Functions Only Web Based Software?**

KIMS®	K-TRAC®	KITS®	K-LOG®	FLOCALC®	KAMS®	NONE
14	8	5	8	14	9	26
29%	16%	10%	16%	29%	18%	53%

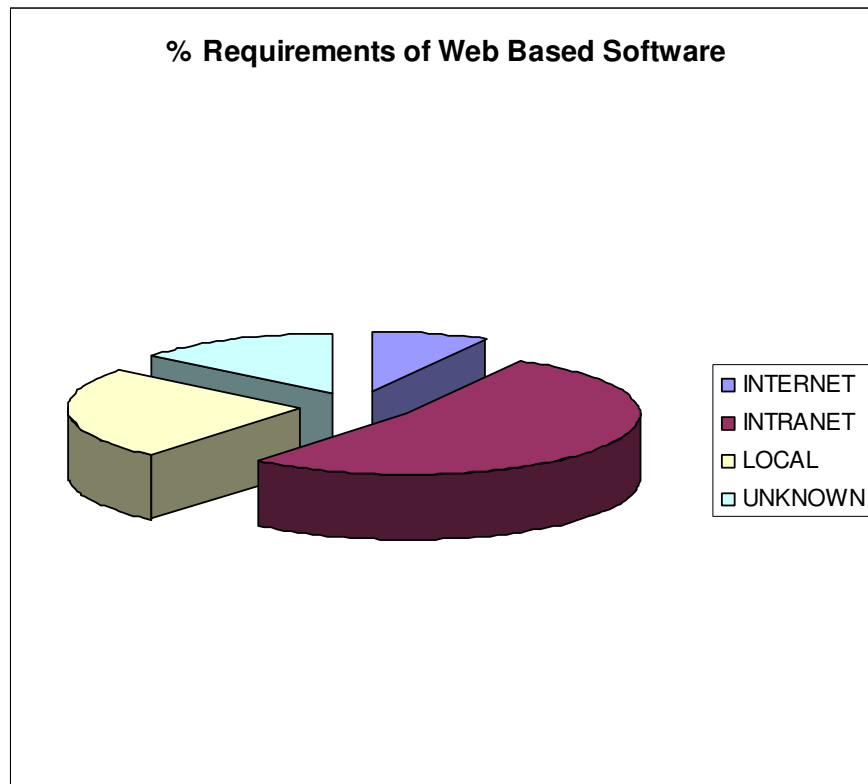
KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE



KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING
SOFTWARE

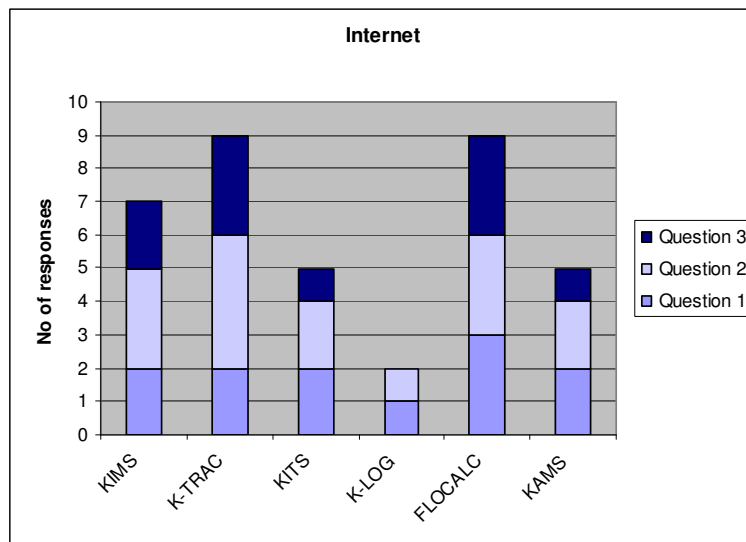
6.4 Q4 How Would the Software Be Delivered?

INTERNET	INTRANET	LOCAL	UNKNOWN
4	26	12	7
8%	53%	24%	14%



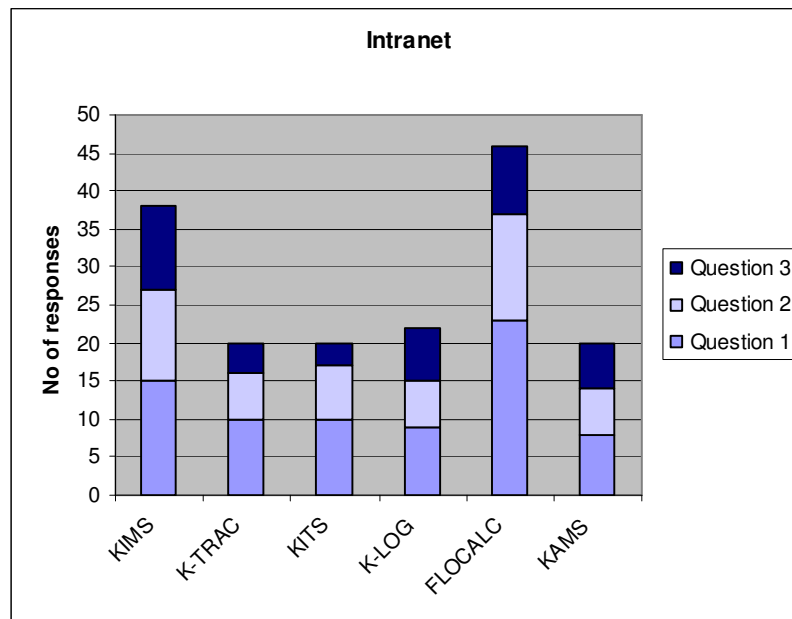
KELTON® KTP**ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE****7.0 DEPLOYMENT OF SOFTWARE**

The following are charts of the different methods of deployment of metering software. The x-axis shows the software types and the y-axis shows the number of positive responses. These charts are the total number of positive responses broken down to question (Q1 to 3), see the market survey for more details of the questions - Appendix B Market Survey Questionnaire.

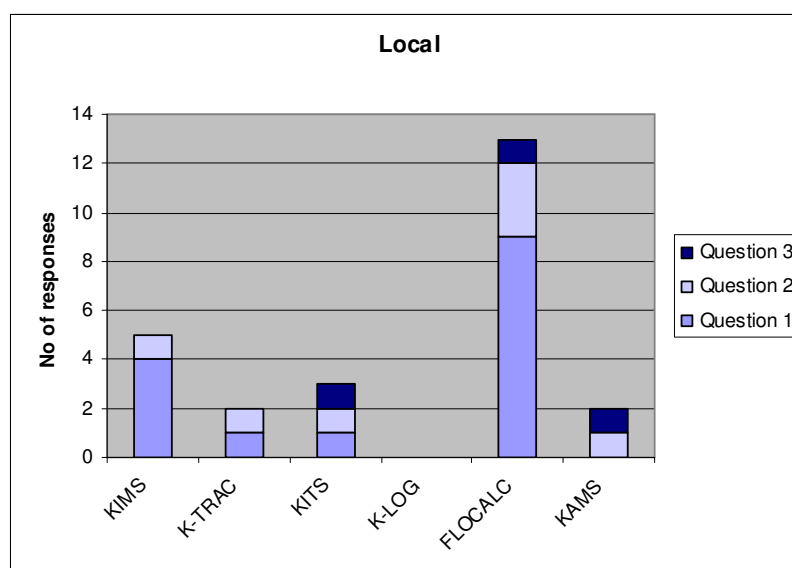
7.1 Internet

KELTON® KTP ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE

7.2 Intranet

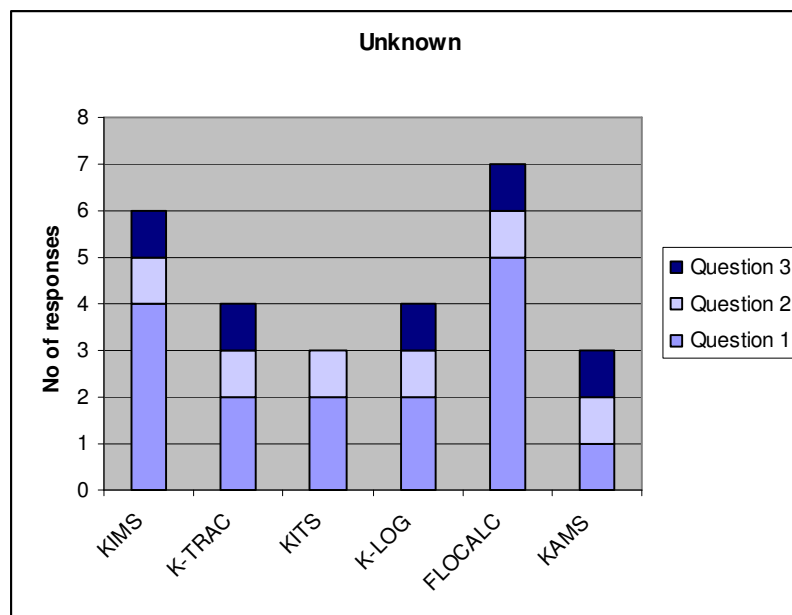


7.3 Local



KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING
SOFTWARE

7.4 Unknown

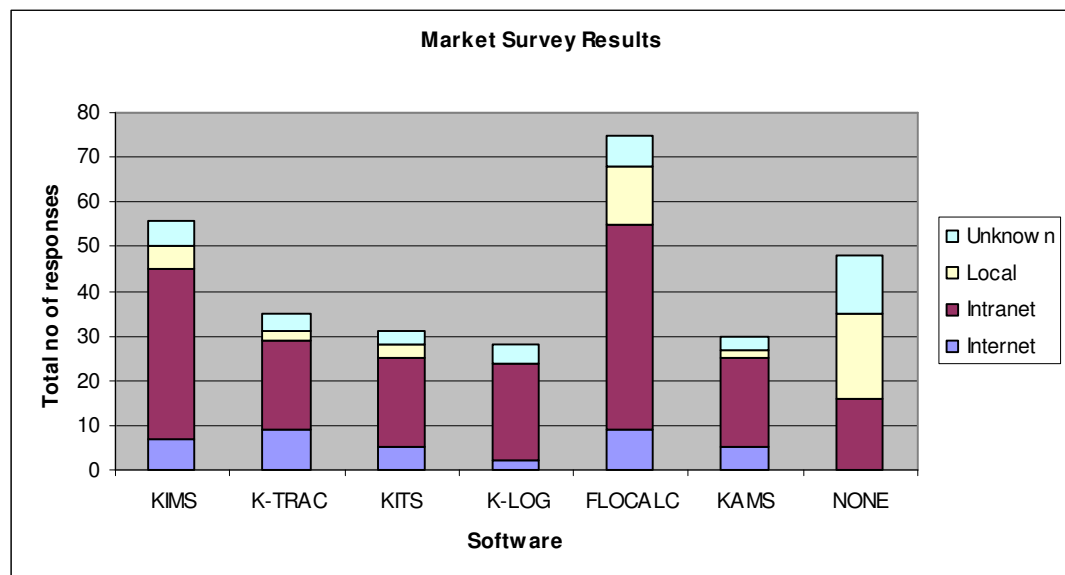


KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING
SOFTWARE

8.0 DETAILS OF DEPLOYMENT ANALYSIS

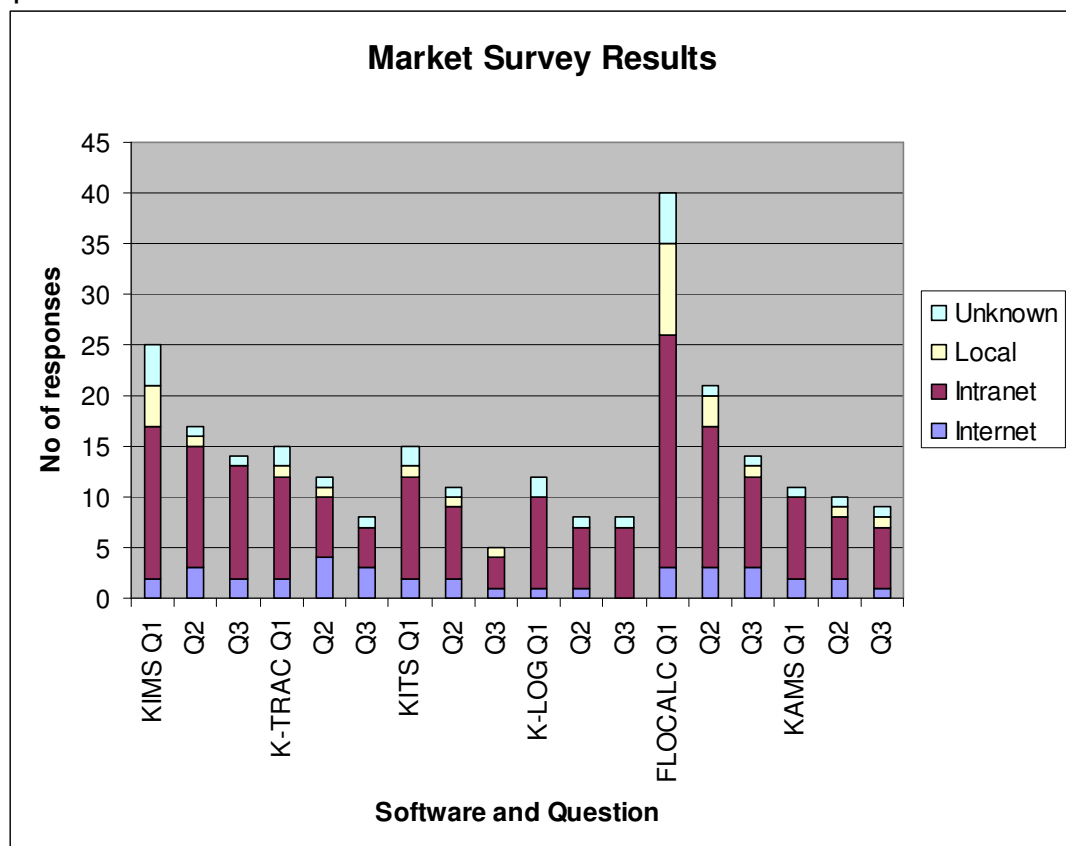
8.1 Cumulative Positive Responses

This graph shows the total number of positive responses for each software package.



KELTON® KTP**ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE****8.2 Positive Responses per Question**

The following graph shows the number of positive responses for each software package and broken down to each question (Q1 to 3) see the market survey questionnaire Appendix B Market Survey Questionnaire for question details.

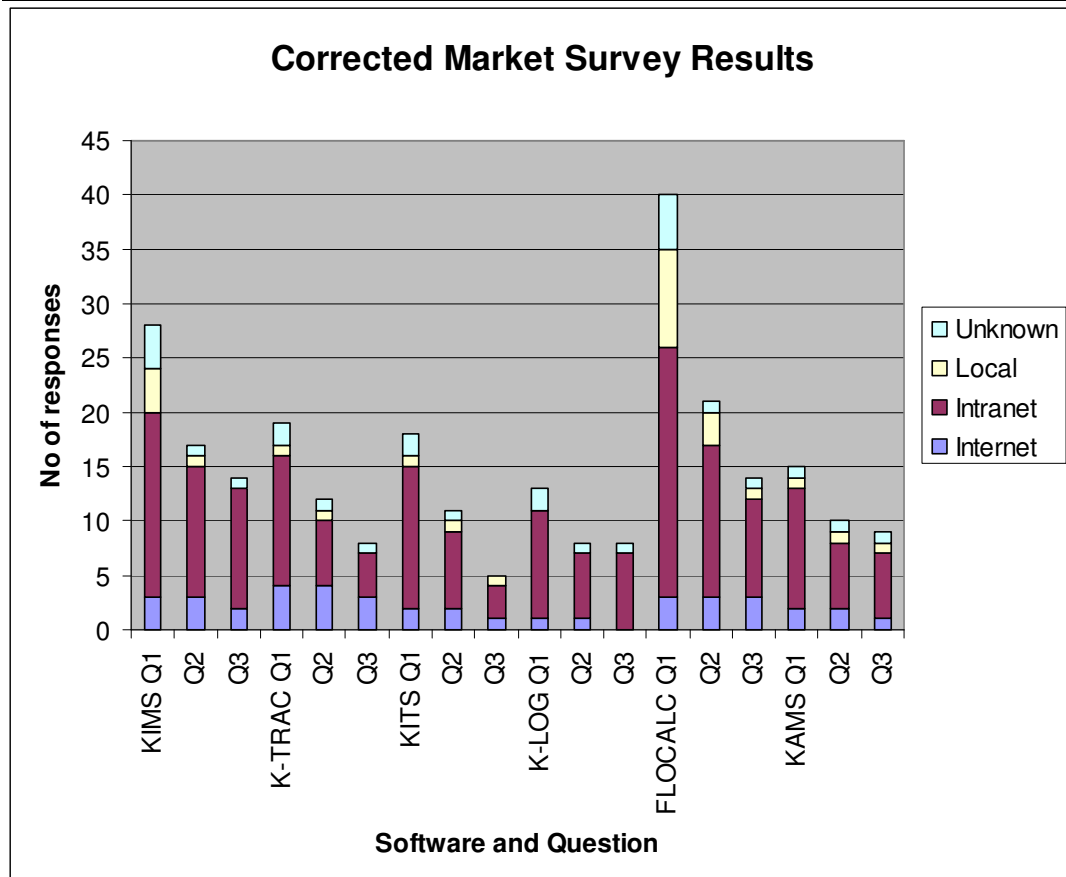


KELTON® KTP**ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE****8.3 Correct Responses for Inconsistent Answers**

The responses were reviewed and some were found to have positive responses for question 2 and 3 but not for question 1, in other words the client wanted fully functional software or management functions but did not want or use the actual software.

The missing responses from question 2 and 3 have been added to question 1 and are listed below. The graph has been recreated to show the updated values for question 1. Question 1 has had the following values added: -

Requirements	KIMS®	K-TRAC®	KITS®	K-LOG®	FLOCALC®	KAMS®
Internet	1	2				
Intranet	2	2	3	1		3
Local						1



KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING
SOFTWARE

APPENDIX A SURVEY RESPONSE NUMBERS

The following table shows the number of positive responses of deployment for each software question.

Q ID	KIMS®	K-TRAC®	KITS®	K-LOG®	FLOCALC®	KAMS®	NONE	Response
1	2	2	2	1	3	2	0	Internet
2	3	4	2	1	3	2	0	Internet
3	2	3	1	0	3	1	0	Internet
1	15	10	10	9	23	8	0	Intranet
2	12	6	7	6	14	6	6	Intranet
3	11	4	3	7	9	6	10	Intranet
1	4	1	1	0	9	0	1	Local
2	1	1	1	0	3	1	8	Local
3	0	0	1	0	1	1	10	Local
1	4	2	2	2	5	1	1	Unknown
2	1	1	1	1	1	1	6	Unknown
3	1	1	0	1	1	1	6	Unknown

KELTON® KTP
ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE

APPENDIX B MARKET SURVEY QUESTIONNAIRE

Help us to help you Manage your Metering Requirements Better.

Good Metering Management requires both the right skills and the right tools.

KELTON® are conducting a survey to find the best way to deliver metering software.

We value your opinion and would appreciate if you could spend a few minutes of your time by answering the following questions: -

Please enter your e-mail address:

- * Q1 Which of the following software tools do you have or would like to have?**

<input type="checkbox"/>	Instrument Management & Flow Computer Validation	More Info
<input type="checkbox"/>	Turbine Meter Performance Monitoring	More Info
<input type="checkbox"/>	Interactive Training	More Info
<input type="checkbox"/>	Electronic Logbooks	More Info
<input type="checkbox"/>	Engineering Flow Calculations	More Info
<input type="checkbox"/>	Audit Management System	More Info
<input type="checkbox"/>	None of the above	

KELTON® KTP**ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE**

- * Q2 Do you or your company have a business need for a fully functional intranet, extranet or internet based version of these tools?**

<input type="checkbox"/>	Instrument Management & Flow Computer Validation	Complete calibrations using a web browser
<input type="checkbox"/>	Turbine Meter Performance Monitoring	Enter and validate probes using a web browser
<input type="checkbox"/>	Interactive Training	Complete competencies and training online
<input type="checkbox"/>	Electronic Logbooks	Log events using a web browser
<input type="checkbox"/>	Engineering Flow Calculations	Setup and complete calculations online
<input type="checkbox"/>	Audit Management System	Complete audits using a web browser
<input type="checkbox"/>	None of the above	

- * Q3 Do you or your company have a business need for management functions via an intranet, extranet or internet, i.e. reporting and data analysis?**

<input type="checkbox"/>	Instrument Management & Flow Computer Validation
<input type="checkbox"/>	Turbine Meter Performance Monitoring
<input type="checkbox"/>	Interactive Training
<input type="checkbox"/>	Electronic Logbooks
<input type="checkbox"/>	Engineering Flow Calculations
<input type="checkbox"/>	Audit Management System
<input type="checkbox"/>	None of the above

- * Q4 Where would you envisage the software residing?**

<input type="checkbox"/>	KELTON® hosting the software on its servers, accessible by anyone with the right authorisation from anywhere with an internet connection, including all management and updating of the system
<input type="checkbox"/>	Your company hosting the software on its servers, accessible by anyone with the right authorisation and a connection to that specific server.
<input type="checkbox"/>	Your local machine to allow you to run offline with a backend hosted at a KELTON® or your company server
<input type="checkbox"/>	Don't know

- Q5 If you have any additional comments or queries please enter them below**

KELTON® KTP

ANALYSIS OF MARKET SURVEY FOR WEB BASED FLOW METERING SOFTWARE

APPENDIX C COMMENTS RECIEVED

The comments received from the survey are show in the table below:

<p>1. We are a pharmaceutical company and operate computer systems validated to GMP (good manufacturing practice) standards, this requires that applications run on mainstream/Intranet connected machines are strictly limited to those for mainstream business. Engineering is not mainstream so engineering programmes are usually run on stand alone machines, I have an old version of Flowcalc which I run on such a machine.</p> <p>2. I would be willing to pay good money for a good tank calculation programme which covers both horizontal and vertical cylindrical tanks with various end dishings. I have several manufacturers freeware offerings but all are wanting to some extent.</p>
<p>1. Q2 and Q3 are confusing. We don't have the features via internet but we found out the hard way that we need it now!</p> <p>2. I am impressed with your "Flowclac". Thanks for the free CD.</p> <p>3. I think you should also provide the "uncertainty" software FOC. It will assist customers in the design. Good publicity will help Kelton. !!</p> <p>4. Spare part management software will be very useful.</p> <p>5. You should have an office in KL.</p> <p>6. You should convince Petronas to include KELTON packages in gas sales agreement or PO.</p> <p>7. I think a software package on system availability should be prepared for metering system specifically. Many systems can be bypassed or forced or overridden to get a plant running - but NOT the metering system. Hence system design (in terms or spare runs, spare parts etc.) for max availability is necessary.</p>
<p>As a commissioning engineer I travel to locations, where there may not be facilities to access, and therefore assist with my working tasks. To have various up-to-date software tools available on-line would be of benefit to me.</p>
<p>Connected to and working from our instrument and gas analysis database.</p>
<p>Have bought all the software I require at this time, but always keen to hear of your developments with the packages and any endorsements from the major players.</p> <p>Cheers and keep up the good work!</p> <p>Jim</p>
<p>I am in the business of Fiscal Measurement (i.e. Custody Transfer) integrated systems. So far, neither myself nor my customers see an impending need for web-based instrument management / flow computer validation systems.</p>
<p>KELTON are more than aware of the issues regarding software due to the company policy on globalisation (GID)</p>
<p>Potential for Monitoring of trends and records verification on predicted trending could be developed to provide first line audit on daily meter readings. Access would be necessary to customer server, but could provide back-up if customer firewall issues were agreed.</p>
<p>Problems interfacing with Shell 'GID' mean most of the software struggles on our systems.</p>
<p>Urgently need validation software for ISO 5167:2003.</p>
<p>We don't feel the need for an internet/intranet based tool.</p> <p>However we might be interested in a fee-for-service solution where KELTON is hosting the software and access is made through the internet (first option of Q4).</p>
<p>What is the possibility to get the interactive training course in Norwegian? What is the content in a training course?</p>

Appendix B – Module List using AJAX

Main transform document generates DHTML for displaying the module groups and modules in a library, initially displays the first level down only.

ModulesTransform.xslt

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output omit-xml-declaration="yes" method="xml"
encoding="iso-8859-1" />
  <xsl:param name="submitUrl"/>
  <xsl:param name="XML"/>
  <xsl:include href="frame.xslt"/>
  <xsl:include href="ModuleGroupItems.xslt" />
  <xsl:template match="Library">
<style>
  .fakelink { cursor: hand }
  .fakelink_Hover { cursor: hand; color: red; text-decoration:
underline }
</style>
<script src="koutline.js" type="text/javascript"></script>
<!-- main javascript used in the document -->
<script language="javascript">
  <xsl:comment>
    //Open the help file
    function pOpenHelp(HelpFileID,lHelpContextID) {
      window.open("<xsl:value-of
select=\"$submitUrl\"/>?RequestOption=Help" +
        "&HelpFileID=" + HelpFileID + "&HelpContextID=" +
        lHelpContextID,"kits_help","location=no,menubar=no,toolbar=
no,resizable=yes,scrollbars=yes");
      frm1.submit();
      return true;
    }
    //Play module
    function pPlayModule(LibraryID,ModuleID) {
      window.open("<xsl:value-of
select=\"$submitUrl\"/>?RequestOption=PlayModule" +
        "&LibraryID=" + LibraryID + "&ModuleID="+
        ModuleID,"","location=no,menubar=no,toolbar=no,resizable=ye
s,scrollbars=yes");
      frm1.submit();
      return true;
    }
    //Show the module groups -modules and module groups at the
next level down
    function pExpandModuleGroup(LibraryID, ModuleGroupID) {
      window.open("<xsl:value-of
select=\"$submitUrl\"/>?RequestOption=ExpandModuleGroup" +
        "&LibraryID=" + LibraryID + "&ModuleGroupID="+
        ModuleGroupID,"","location=no,menubar=no,toolbar=no,resizab
le=yes,scrollbars=yes");
      frm1.submit();
```

```

        return true;
    }
    //Call to get an xml file for the list
    function pGetModuleGroupItems(LibraryID, ModuleGroupID) {
        var url = "<xsl:value-of
        select=\"$submitUrl\"/>?RequestOption=ExpandModuleGroup" +
        "&LibraryID=" + LibraryID + "&ModuleGroupID="+
        ModuleGroupID;
        req = false;
        if (window.XMLHttpRequest) {
            try {
                req = new XMLHttpRequest();
                req.onreadystatechange = processReqChange;
                req.open("GET", url, true);
                req.send(null);
            } catch(e) {
                req = false;
            }
        } else if (window.ActiveXObject) { //IE
            try {
                req = new ActiveXObject("Msxml2.XMLHTTP");
                req.onreadystatechange = processReqChange;
                req.open("GET", url, true);
                req.send();
            } catch (e) {
                try {
                    req = new ActiveXObject("Microsoft.XMLHTTP");
                    req.onreadystatechange = processReqChange;
                    req.open("GET", url, true);
                    req.send();
                } catch(e) {
                    req = false
                }
            }
        }
        if (req) {
        } else {
            // TODO need to request the page to be reloaded
            alert('Cannot create an XMLHttpRequest Instance');
            return false;
        }
    }
    // Transform xml document with the xsl transform
    function fnTransform(xslIsland,oXml,divOutput){
        var xslDoc=new
        ActiveXObject("MSXML2.FreeThreadedDOMDocument.4.0");
        var xslTemplate=new
        ActiveXObject("MSXML2.XSLTemplate.4.0");
        xslDoc.load(xslIsland.XMLDocument);
        xslTemplate.stylesheet=xslDoc;
        var xslProc=xslTemplate.createProcessor();
        xslProc.input=oXml;
    }

```



```

    if (arguments.length > 3 & & (arguments.length-3)
    % 2 == 0) {
        for (var i=0; i<Math.floor((arguments.length-
        3)/2); i++) {
            paramName=arguments[2*i+3];
            paramValue=arguments[2*i+4];
            xslProc.addParameter(paramName,paramValue);
        }
    }
    xslProc.transform();
    divOutput.innerHTML="";
    divOutput.insertAdjacentHTML("beforeEnd",xslProc.output);
}
//Transform xml using the ModuleList XSL
function processReqChange() {
    if (req.readyState == 4) {
        if (req.status == 200) {
            var oXML = req.responseXML;
            var id = 'mg_' +
            oXML.documentElement.getAttribute('strip');
            var target = document.all(id);
            fnTransform(modulelistxsl,oXML,target);
            KinitFakeLinks(target);
            KdynToggleIcon(target.parentElement,'colicon');
            target.style.display='';
        } else {
            alert("There is a problem retrieving the XML data:\n" +
            req.statusText);
        }
    }
}
//Details for licensing a module
function pAlertUser() {
    alert("You do not have a licence for this module.\r\nPlease
    contact KELTON&#0174; on +44 1224 630000 to have this
    module unlocked.");
}
//Load images used for toolbar
function FP_preloadImgs() {
    var d=document,a=arguments; if(!d.FP_imgs) d.FP_imgs=new
    Array();
    for(var i=0; i<a.length; i++) {
        d.FP_imgs[i]=new Image;
        d.FP_imgs[i].src=a[i];
    }
}
//Change image on toolbar when mouse moved or selected
function FP_swapImg() {
    var doc=document,args=arguments,elm,n; doc.$imgSwaps=new
    Array();
    for(n=2; n<args.length;n+=2) {
        elm=FP_getObjectByID(args[n]);

```

```

        if(elm) {
            doc.$imgSwaps[doc.$imgSwaps.length]=elm;
            elm.$src=elm.src; elm.src=args[n+1];
        }
    }
}
function FP_getObjectByID(id,o) {
    var c,el,els,f,m,n;
    if(!o)o=document;
    if(o.getElementById) el=o.getElementById(id);
    else if(o.layers) c=o.layers;
    else if(o.all) el=o.all[id];
    if(el) return el;
    if(o.id==id || o.name==id) return o;
    if(o.childNodes) c=o.childNodes;
    if(c)
        for(n=0; n<c.length; n++) {
            el=FP_getObjectByID(id,c[n]);
            if(el) return el;
        }
    f=o.forms;
    if(f)
        for(n=0; n<f.length; n++) {
            els=f[n].elements;
            for(m=0; m<els.length; m++){
                el=FP_getObjectByID(id,els[n]);
                if(el) return el;
            }
        }
    return null;
}
//
</xsl:comment>
</script>
<xml id="modulelistxsl" src="modulelist.xslt"></xml>
<h1>KITS<sup>&#0174;</sup> V3.0 Module List</h1>
<form id="frm1" method="post">
    <input type="hidden" name="ListXML">
        <xsl:attribute name="value">
            <xsl:value-of select="$XML"/>
        </xsl:attribute>
    </input>
<div onclick="KdynOutline()" language="javascript1.2">
    <ul id="ModuleList" Kdynamicoutline="true"
        Kinitcollapsed="false">
        <!--Use the ModuleGroupItems xslt to display all the module
        groups and modules in the library -->
        <xsl:call-template name="ModuleGroupItems">
            <xsl:with-param name="LIBRARYID" select="@id"/>
        </xsl:call-template>
    </ul>
</div>

```

```

<script language="javascript">
  <xsl:comment>
    KinitOutline();
  </xsl:comment>
</script>
</form>
<br />
</xsl:template>
<!-- get the number of items in a module group -->
<xsl:template match="ModuleGroup" mode="CountOptions">
  <xsl:for-each select=".">
    <xsl:if test="position() = last()">
      <xsl:value-of select="position()" />
    </xsl:if>
  </xsl:for-each>
</xsl:template>
<!--get the sign off date and user name of the module -->
<xsl:template match="SignOff">
  <xsl:value-of select="@date" />
  <xsl:value-of select="@userName" />
</xsl:template>
<!--instructions for displaying a button in the toolbar -->
<xsl:template name="toolbarbutton">
  <xsl:param name="id" />
  <xsl:param name="tooltip" />
  <xsl:param name="onclick" />
  <xsl:param name="width" />
  <img border="0" height="26">
    <xsl:attribute name="width"><xsl:value-of select="$width" />
  </xsl:attribute>
    <xsl:attribute name="id">img<xsl:value-of select="$id" />
  </xsl:attribute>
    <xsl:attribute name="src">images/Toolbar/Normal/<xsl:value-of
select="$id" />.gif</xsl:attribute>
    <xsl:attribute name="alt"><xsl:value-of select="$tooltip" />
  </xsl:attribute>
    <xsl:attribute name="onmouseover">
      FP_swapImg(0,0,/*id*/'<xsl:value-of select="$id"
/>',/*url*/'images/Toolbar/Selected/<xsl:value-of
select="$id" />.gif')
    </xsl:attribute>
    <xsl:attribute name="onmouseout">
      FP_swapImg(0,0,/*id*/'<xsl:value-of select="$id"
/>',/*url*/'images/Toolbar/Normal/<xsl:value-of select="$id"
/>.gif')
    </xsl:attribute>
    <xsl:attribute name="onclick"><xsl:value-of select="$onclick"
/></xsl:attribute>
  </img>
</xsl:template>
</xsl:stylesheet>

```

Generates DHTML to list modules and module groups within a library or module group.

ModuleGroupItems.xslt

```
<?xml version="1.0"?>
  <xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template name="ModuleGroupItems">
      <xsl:param name="LIBRARYID"/>
      <xsl:apply-templates select="ModuleGroup|Module"
        mode="ModuleGroupItem">
        <xsl:sort select="@sequence" data-type="number"/>
        <xsl:with-param name="LIBRARYID" select="@id"/>
      </xsl:apply-templates>
    </xsl:template>
    <!--Instructions for displaying a Module Group on the list,
    creates javascript to expand group when selected-->
    <xsl:template match="ModuleGroup" mode="ModuleGroupItem">
      <xsl:param name="LIBRARYID"/>
      <xsl:param name="PARENTID" select="''"/>
      <xsl:param name="PARENTNAME" select="''"/>
      <li style="list-style-img:url(images/ModuleGroup.gif)">
        <!-- uses fnInitPopulateIndex -->
        <xsl:attribute name="KDynExpand">
          <xsl:text>javascript:pGetModuleGroupItems('</xsl:text>
          <xsl:value-of select="@libraryID"/>
          <xsl:text>','</xsl:text>
          <xsl:value-of select="@id"/>
          <xsl:text>');</xsl:text>
        </xsl:attribute>
        <span class="fakelink">
          <xsl:value-of select="@name"/>
          <xsl:text>-</xsl:text>
          <xsl:value-of select="@description"/>
          
        </span>
        <ul style="display: none">
          <xsl:attribute name="id">mg_<xsl:value-of
            select="translate(@id, '{}-',',')"/>
          </xsl:attribute>
          &#160;
        </ul>
      </li>
    </xsl:template>
    <!--Instructions for displaying a Module on the list, creates
    javascript to play module when selected if licensed-->
    <xsl:template match="Module" mode="ModuleGroupItem">
      <xsl:param name="PARENTID"/>
      <xsl:param name="LIBRARYID"/>
      <xsl:param name="PARENTNAME"/>
      <li>
        <xsl:choose>
          <xsl:when test="@licenced='-1'">
```

```

        <xsl:attribute name="style">list-style-
img:url(images/Module.gif)</xsl:attribute>
</xsl:when>
<xsl:otherwise>
    <xsl:attribute name="style">list-style-
img:url(images/locked.gif)</xsl:attribute>
</xsl:otherwise>
</xsl:choose>
<span class="fakelink">
    <xsl:choose>
        <xsl:when test="@licenced='-1'">
            <xsl:attribute name="onclick">
                <xsl:text>javascript:pPlayModule('</xsl:text>
                <xsl:value-of select="@libraryID"/>
                <xsl:text>', '</xsl:text>
                <xsl:value-of select="@id"/>
                <xsl:text>');</xsl:text>
            </xsl:attribute>
        </xsl:when>
        <xsl:otherwise>
            <xsl:attribute
                name="onclick">javascript:pAlertUser();</xsl:attribute>
        </xsl:otherwise>
    </xsl:choose>
    <xsl:value-of select="@name"/>
    <xsl:text>-</xsl:text>
    <xsl:value-of select="@description"/>
</span>
</li>
</xsl:template>

</xsl:stylesheet>

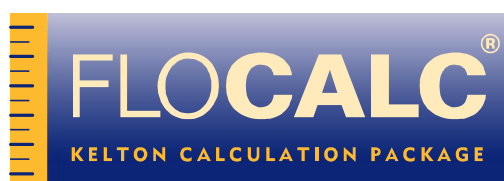
```

Reference ModuleGroupItems xslt for module group items so that the user can continue to drill down a group, used by the main xsl

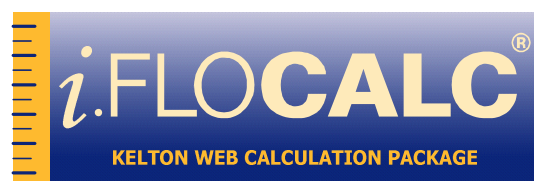
ModuleList.xslt

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output omit-xml-declaration="yes" method="xml"
encoding="iso-8859-1" />
  <xsl:include
href="Default.asp?RequestOption=ModuleGroupItems.xslt" />
  <xsl:template match="ModuleGroup">
    <xsl:call-template name="ModuleGroupItems">
      <xsl:with-param name="LIBRARYID" select="@LibraryID"/>
    </xsl:call-template>
  </xsl:template>
</xsl:stylesheet>
```

Appendix C – Calculation List

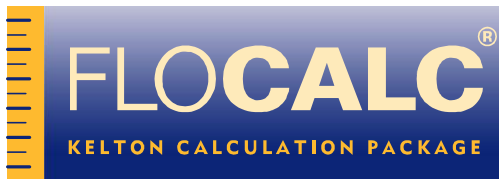


&

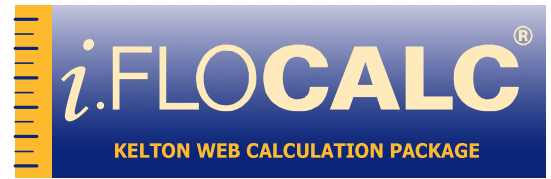


CALCULATION DETAILS

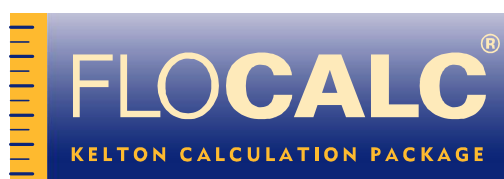
SOURCE	TITLE/DESCRIPTION	CALC REF.
ISO 6976:1983	ISO 6976:1983 Calorific Value and Relative Density This calculates calorific values, density, relative density and Wobbe index from a gas composition. Results are calculated for the composition treated as both a real and an ideal gas, inferior (net) and superior (gross) calorific value and Wobbe index are displayed in each case.	F001
ISO 6976:1989	ISO 6976:1989 Calorific Value and Relative Density This calculates calorific values, density, relative density and Wobbe index from a gas composition. Results are calculated for the composition treated as both a real and an ideal gas, inferior (net) and superior (gross) calorific value and Wobbe index are displayed in each case.	F002
ISO 6976:1995 BS 7589	ISO 6976:1995 Calorific Value and Relative Density This calculates calorific values, density, relative density and Wobbe index from a gas composition. Results are calculated for the composition treated as both a real and an ideal gas, inferior (net) and superior (gross) calorific value and Wobbe index are displayed in each case. This version of the calculation included both the definitive and alternative methods of calculating the calorific value on a volumetric basis.	F003
ISO 6976: 1995 GPA 2145:2000	ISO 6976/GPA 2145:2000 Calorific Value, Relative Density This calculates calorific values, density, relative density and Wobbe index from a gas composition. Results are calculated for the composition treated as both a real and an ideal gas, inferior (net) and superior (gross) calorific value and Wobbe index are displayed in each case. This version of the standard uses the gas properties given in the GPA 2145:2000 tables.	F036
ISO 5167:1991	ISO 5167:1991 Orifice Flow Rate Calculation This follows the process outlined in the standard to calculate flow rate through an orifice meter. Density and temperature can be entered at up or downstream conditions to mimic the calculations performed by a flow computer and the calculation can iterate to solve for flow, differential pressure or orifice bore size. This version of the standard uses the Stoltz equation to calculate the discharge coefficient.	F004
ISO 5167:1991	ISO 5167:1991 Venturi Flow Rate Calculation This follows the process outlined in the standard to calculate flow rate through a Venturi tube or nozzle. The calculation can iterate to solve for flow, differential pressure or Venturi throat size.	F005
ISO 5167: 1991 ISO 5167: 2003	ISO 5167 Gas System Uncertainty This calculation estimates the uncertainty in flow rate through an orifice plate meter based on the 'practical working formula' outlined in ISO 5167 (Versions 1991 to 2003)	F010
ISO 5167-1:1991 & 2003	Orifice Plate Validation This calculation is used to check the condition and geometry of an orifice plate meets the criteria laid out in the standard. The user can either enter measurements taken directly from an Orifice Plate or independently validate an orifice plate certificate produced and issued by a calibration laboratory.	F026



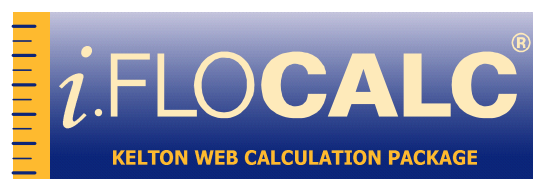
&



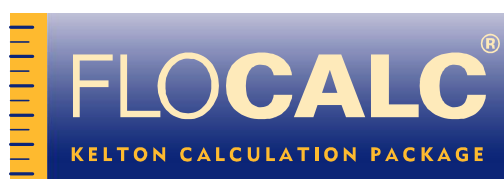
SOURCE	TITLE/DESCRIPTION	CALC REF.
ISO 5167-1: 1991 Amendment 1:1998	ISO 5167-1: Amendment 1:1998 Orifice Meter Flow Rate Calculation This follows the process outlined in the standard to calculate mass flow rate through an orifice meter. Density and temperature can be entered at up or downstream conditions to mimic the calculations performed by a flow computer and the calculation can iterate to solve for flow, differential pressure or orifice bore size. This version of the standard uses the Reader-Harris/Gallagher equation to calculate the discharge coefficient.	F027
ISO 5167 McCrometer 24509-54	ISO 5167: McCrometer V-Cone This is essentially an ISO 5167 flow rate calculation modified by McCrometer for the geometry and characteristics of their V-Cone meters. The calculation has options to use either the 2000 or 2005 version on the McCrometer calculation the latter of which contains a revised method of determining expansibility. To utilise calibration data the option is included to enter a characterisation curve showing the change in discharge coefficient with Reynold's number.	F030
ISO 5167 Murdock	ISO 5167: Wet Gas Venturi (Murdock) This calculation is based on the ISO 5167 standard to calculate mass flow rate through a Venturi tube or nozzle extended to include the Dickenson/Jamieson variant of the Murdock correction. The wet gas (saturated) flow rate is calculated along with the flow rate for each phase of the fluid.	F032
ISO 5167 De Leeuw	ISO 5167: Wet Gas Venturi (Chisholm/De Leeuw) This calculation is based on the ISO 5167 standard to calculate mass flow rate through a Venturi tube or nozzle extended to include the Chisholm De Leeuw wet gas correction. The wet gas (saturated) flow rate is calculated along with the flow rate for each phase of the fluid.	F033
ISO 5167: 2003	ISO 5167: 2003 Orifice Meter Flow Rate Calculation This follows the process outlined in the standard to calculate flow rate through an orifice meter. Density and temperature can be entered at up or downstream conditions to mimic the calculations performed by a flow computer and the calculation can iterate to solve for flow, differential pressure or orifice bore size.	F044
ISO 5167: 2003	ISO 5167: 2003 Venturi Meter Flow Rate Calculation This follows the process outlined in the standard to calculate flow rate through a Venturi tube or nozzle. The calculation can iterate to solve for flow, differential pressure or orifice bore size.	F045
ISO 3171:1999	ISO 3171:1999 Estimating Water-in-Oil Dispersion (Annex A) This calculation is used to indicate whether the dispersion of water in oil is likely to be adequate for sampling.	F031
AGA 3:1992	AGA 3:1992 Orifice Meter Flow Rate Calculation This follows the process outlined in the American Gas Association standard to calculate flow rate through an orifice meter. The calculation includes three calculation methods, Part 1 imperial, Part 1 metric and Part 3.	F019
AGA 8:1985	AGA 8:1985 Gas Density Computation The compressibility and density of a gas are calculated from its composition, temperature and pressure in accordance with the 'Detail Characterisation' method outlined in this standard. Results are displayed for both standard (user configurable) temperature and pressure and operating temperature and pressure.	F013



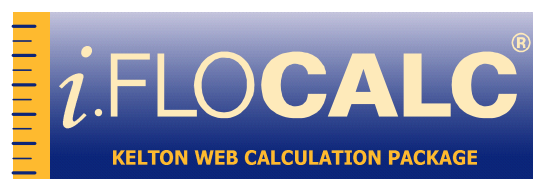
&



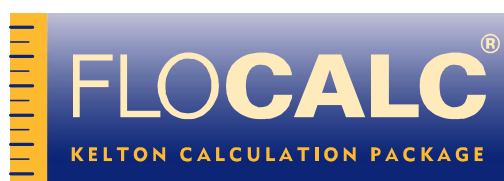
SOURCE	TITLE/DESCRIPTION	CALC REF.
AGA 8:1994 ISO 12213-1:1997	AGA 8:1994/ISO 12213-1:1997 Gas Density Computation The compressibility and density of a gas are calculated from its composition, temperature and pressure in accordance with the 'Detail Characterisation' method outlined in this standard. Results are displayed for both standard (user configurable) temperature and pressure and operating temperature and pressure. This 1994 printing of the Second Edition 1992 achieves computational consistency with GPA 2172-94 and AGA 3 1992.	F014
AGA 8: 1994 (SGERG)	AGA 8 Density Computation Gross Method (SGERG) This calculates density and compressibility using the SGERG equation of state (Gross Characterisation Method). The calculation can be executed using a choice of input options such as from relative density or molecular weight and for a choice of reference conditions.	F024
AGA 10: 2003 AGA 8: 1994 API MPMS 14.2 ISO 12213 Part 2	AGA 10 Velocity of Sound Calculation This calculation estimates the velocity of sound of a gas at line conditions based on the composition, pressure and temperature using the formulae presented in the American Gas Association Report.	F046
ASTM D3588:1991	ASTM D3588:1991 Calorific Value and Relative Density This calculates calorific values, density and relative density from a gas composition. Results are calculated for the composition treated as both a real and an ideal gas, net and gross calorific value are displayed in each case. The option is also included to perform a dry to wet conversion using factors for the gross and net CV.	F021
IP Paper 2 IP 200 API Std 2540 ASTM D1250 ANSI/ASTM D1250	IP Paper 2 Density Referral This calculation is used to 'convert' density values between standard conditions and operating conditions by applying a correction for the change in temperature (C_{ti}) and pressure (C_{pi}). C_{pi} is calculated using the methods outlined in IP Paper 2 and C_{ti} using the API equations from which the appropriate product group can be selected. The option is given to either perform the calculation following the rounding/truncation algorithms outlined in the standard or to use full precision.	F022
IP Paper 2 ASTM D1250 IP 200 Petroleum Measurement Tables for Light Hydrocarbon Liquids ASTM- IP-API	IP Paper 2: Density Referral (Light Hydrocarbon Liquids) This calculation is used to 'convert' density values between standard conditions and operating conditions by applying a correction for the change in temperature (C_{ti}) and pressure (C_{pi}). C_{pi} is calculated using the methods outlined in IP Paper 2 and C_{ti} using the petroleum measurement tabled for light hydrocarbons (Table 53/54). The option is given to either perform the calculation following the rounding/truncation algorithms outlined in the standard or to use full precision.	F029
IP Petroleum Measurement Manual Part VII Section 2 Gas Laws	PTZ Gas Density Calculations This calculation is used to determine the density of a non-ideal gas at a given temperature and pressure from known values of pressure temperature and compressibility or molecular weight. Options include solving for either line density, standard density or relative density.	F006



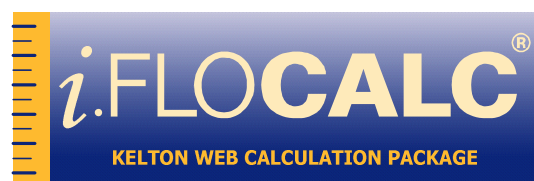
&



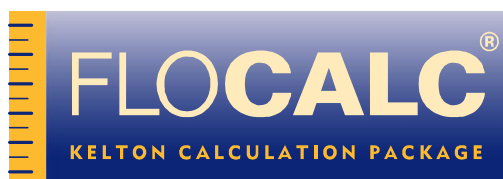
SOURCE	TITLE/DESCRIPTION	CALC REF.
IP Manual Part X API MPMS 11.2.1M API MPMS 11.2.2M GPA 8286-86 (M) IP 200 API Std 2540 ASTM D1250 ANSI/ASTM D1250 Petroleum Measurement Tables for Light Hydrocarbon Liquids ASTM- IP-API	K-Factor Computation This calculates the K-Factor for a meter which has been 'proved' using either a pipe prover or a master meter. Corrections are applied to compensate for changes in the geometry of the 'prover' (pipe or master meter) and changes in the volume of the liquid caused by temperature and pressure. Where applicable these corrections may be calculated using a choice of industry and international standards.	F011
IP Petroleum Measurement Manual Part VII Section 2	Upstream Density Calculation This calculation corrects density from downstream to upstream conditions for an orifice meter. The option is included to calculate the density exponent from the isentropic exponent.	F037
API MPMS 11.2.1M API MPMS 11.2.2M GPA 8286-86 (M) IP 200 API Std 2540 ASTM D1250 ANSI/ASTM D1250	API Density Referral This calculation is used to 'convert' density values between standard conditions and operating conditions by applying a correction for the change in temperature (C_{ti}) and pressure (C_{pi}). C_{pi} is calculated using the methods outlined in the petroleum measurement standards and C_{ti} using the API equations from which the appropriate product group can be selected. The option is given to either perform the calculation following the rounding/truncation algorithms outlined in the standard or to use full precision.	F023
API MPMS 11.2.1M API MPMS 11.2.2M GPA 8286-86 (M) Petroleum Measurement Tables for Light Hydrocarbon Liquids ASTM- IP-API	API Density Referral (Light Hydrocarbon Liquids) This calculation is used to 'convert' density values between standard conditions and operating conditions by applying a correction for the change in temperature (C_{ti}) and pressure (C_{pi}). C_{pi} is calculated using the methods outlined in the petroleum measurement standards and C_{ti} using the petroleum measurement tabled for light hydrocarbons (Table 53/54). The option is given to either perform the calculation following the rounding/truncation algorithms outlined in the standard or to use full precision.	F028
API MPMS 14 API 2530 AGA 3 ANSI/API 2530-1985 GPA 8185-85	API Volumetric Flow Rate of Natural Gas at Standard Conditions (Factors Approach Method) 1992 This calculates the volumetric flow rate of natural gas at standard conditions using the 'Factors Approach' method outlined in the Manual of Petroleum Measurement Standards Chapter 14 Section 3. Appendix 3-B	F034
API Technical Data Book	Viscosity/Isentropic Exponent This calculates the viscosity and isentropic exponent from the gas composition temperature and pressure following methods outlined in the American Petroleum Institute Technical Data Book. Other useful parameters returned by this calculation include the specific heat ratio which can be used in gas densitometer calculations.	F020



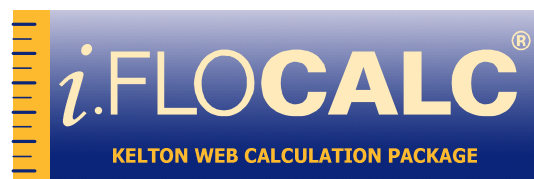
&



SOURCE	TITLE/DESCRIPTION	CALC REF.
API MPMS 14.5 GPA 2172	Calculation of Gross Heating Value, Specific Gravity and Compressibility This uses the procedure for calculating heating value, specific gravity and compressibility factor from the compositional analysis of a natural gas mixture.	F038
Solartron Technical Manuals	Solartron Density/Velocity of Sound This calculates the density from a time period, applying a correction for the velocity of sound of the measured gas. The velocity of sound is calculated from the composition using the method detailed in the Solartron 7915 flow computer manual.	F007
Solartron Technical Manuals	Gas Densitometer Calculations - Solartron Solartron densitometers work on the principle that the natural frequency of the transducers vibrating element is affected by the density of the fluid in which it is submerged. This calculation calculates the density from the measured frequency and densitometer constants obtained from calibration. Options are given to apply corrections for temperature and the velocity of sound, a further option is included to correct density from down to upstream conditions.	F008
Solartron Technical Manuals	Liquid Densitometer Calculations - Solartron Solartron densitometers work on the principle that the natural frequency of the transducers vibrating element is affected by the density of the fluid surrounding it. This calculation calculates the density from the measured frequency and densitometer constants obtained from calibration. Options are given to apply corrections to compensate for the temperature and pressure of the fluid.	F009
Solartron Technical Manuals	Gas Relative Density Calculation - Solartron This calculates the relative density from the Solartron RD transducer constants and the measured time period. The transducer constants can be calculated by entering the known relative densities of two calibration gases along with their corresponding measured time periods.	F016
Solartron Technical Manuals	Solartron RD Analyser Appendix A Calculation The 'Solartron Appendix A calibration considerations' calculated using this form reduce the effect of systematic errors associated with the density sensor, and also the non-ideal behaviour of gasses.	F017
Instromet Technical Papers	Ultrasonic Meter Calculations - Instromet This calculates the volume flow rate, applying corrections for the elastic distortion of the ultrasonic meter spool due pressure and thermal expansion. Options are also given to apply a linearity correction to include data obtained by calibration and convert the calculated volume flow rate to mass and standard volume.	F039
Sarasota/Peek Technical Manuals	Density Computation - Sarasota/Peek Sarasota/Peek densitometers work on the principle that the natural frequency of the transducers vibrating element is affected by the density of the fluid in which it is submerged. This calculation calculates the density from the measured frequency and densitometer constants obtained from calibration. Options are given to apply corrections for temperature and pressure. An option to calculate the corrected time period for use during an air-check is also included.	F040
R Norman, M S Rawat and P Jepson 1983 P Jepson and R Chipchase 1975	Orifice Plate Buckling An orifice plate, when exposed to differential pressure, will always experience a degree of elastic deformation, in certain cases the elastic deformation can be augmented by plastic (permanent) deformation. This calculates the differential pressure that would cause the plastic distortion of a simply supported orifice plate. In addition to this, flow measurement errors caused by the deformation of the orifice plate are estimated.	F015




&



SOURCE	TITLE/DESCRIPTION	CALC REF.
S. Lewis and G. Peggs The Pressure Balance 1992	Pressure Calculation - Absolute and Gauge This calculation is used to determine the pressure generated by deadweight testers, pressure indicators and gauges. Pressure can either be calculated from first principals using mass and piston area or simply applying corrections to the nominal applied pressure. The calculation can also be reversed to calculate the mass required to generate a required pressure. Absolute pressure can be calculated for either using a deadweight tester in absolute mode or combining gauge pressure with barometric pressure.	F018
IGF 1967 Wollard 1979 IAC 1980/WGS 1984	Local Gravity Calculation The local value of gravitation acceleration for a geographical location can be estimated from the latitude and height above sea-level. This calculation provides a choice of three accepted formulae for determining this value. In addition to this the option is given to calculate for an offshore or an onshore location which takes applies an additional correction for the density of the rock base.	F025
M. Hay and D. Simpson NPL Report CMAM 41: 1999	Pressure Calculation - High-line DP This calculation is used to determine the pressure generated by differential deadweight testers, pressure indicators and gauges. Differential pressure can either be calculated from first principals using mass and piston area (or Kn) or correcting the nominal applied pressure. The calculation can also be reversed to calculate the mass required to generate a required differential pressure	F041
R. S. Davis Metrologia 1992	Density of Moist Air This calculates the density of moist air from density pressure and relative humidity using the process outlined by R. S. Davis in metrologia 1992.	F043
GRI-91/0184	GRI-91/0184 Velocity of Sound Computation This calculation estimates the velocity of sound of a gas at line conditions based on the composition, pressure and temperature using the formulae presented in the formulae outlined in the Gas Research Institute technical reference document.	F035
BS EN 60751 BS 1904	PRT Calculation This calculation is used to either calculate the temperature from a resistance value or vice versa. The option is given to select either BS EN 60751 or the BS 1904 which it superseded.	F042

Third Party Calculations

SOURCE	TITLE/DESCRIPTION	CALC REF.
 EffecTech Specialists in Gas Measurement Peng-Robinson Redlich-Kwong-Soave	EffecTech: Hydroacarbon Dew point Calculation This calculates the phase equilibrium dew temperature, the vapour-liquid equilibrium and the circondentherm from a composition at a given pressure. Components can be chosen from eight different data sources and calculations performed using a choice of either the Peng-Robinson or the Redlich-Kwong-Soave equation of state method.	F047

Appendix D – KELTON[®] License Agreement

Licence Agreement**1 Definitions**

KELTON®	-	Kelton Engineering Ltd, incorporated under Companies Act 1985, Registered in Scotland Number 129648 and having its Registered Office at The MacKenzie Building, 168 Skene Street, Aberdeen AB10 1PE
FLOCALC®	-	Flow Measurement Calculation Software
USER	-	The Licensee

2 Grant of Licence

KELTON® grants you the following rights provided that you comply with all terms and conditions of this EULA:

2.1 Installation and use.

You may:

- a) install and use a copy of the Software on one personal computer or other device;
and
- b) install an additional copy of the Software on a second, portable device for the exclusive use of the primary user of the first copy of the Software.

2.2 Alternative Rights for Storage/Network Use.

As an alternative to Section 2.1(a), you may install a copy of the Software on a network storage device, such as a server computer, and allow one access device, such as a personal computer, to access and use that Licenced copy of the Software over a private network. You must obtain a Licence to the Software for each additional device that accesses and uses the Software installed on the network storage device, except as permitted by Section 2.4 of this EULA.

2.3 Licence Grant for Remote Desktop.

You may use remote access technologies, such as the Remote Desktop features in Microsoft Windows or NetMeeting, to access and use your Licenced copy of the Software, provided that only the primary user of the device hosting the remote desktop session accesses and uses the Software with a remote access device. These remote desktop rights do not permit you to use the Software on both the device hosting the remote desktop session and the access device at the same time.

2.4 Licence Grant for Remote Assistance.

You may permit any device to access and use your Licenced copy of the Software for the sole purpose of providing you with technical support and maintenance services.

3 Restrictions

As a user you may not:

- Copy, with the exception of one copy for "back-up" purposes only, the software. Any such copy will be in all respects subject to the terms and conditions of this Licence Agreement.
- Loan, rent, lease or transfer to another user.
- De-compile or disassemble the Software save as provided for as of right under Section 50B of Copyright Designs and Patents Act 1988.
- Use FLOCALC® on other than equipment suitable for its operation.
- Seek to repair, adjust alter or modify FLOCALC® in any way.
- Use FLOCALC® for a purpose other than for the purpose for which it is designed.
- Remove or alter any copyright or other proprietary notice on FLOCALC®.

Breach of the foregoing will entitle KELTON® to immediately terminate this Licence, the Licensee will indemnify KELTON® in respect of all claims, losses, expenses and damages thereby incurred and the Licensee shall return FLOCALC® and all copies made (authorised or unauthorised) without refund.

4 Warranty

KELTON® warrants that:

- The Software will perform substantially in accordance with the specifications.
- That the materials within the package are not defective.

The warranty period is ninety (90) days from the date of despatch of the package from KELTON® and complaints must be received by KELTON® in writing within that period failing which no claim under warranty will be competent.

5 Limitation of Liability

KELTON® warrants to the User only that the software will function substantially in accordance with the specification, always provided that the software has been used and maintained strictly in accordance with Kelton's instructions, and the User has complied with the terms of this Licence Agreement in all respects. Save as herein provided, all representations, warranties, conditions or terms whether expressed or implied and whether statutory or otherwise are hereby expressly excluded. Under no circumstances shall KELTON® be liable to the User or third parties for loss of profit or direct or indirect loss or damage whether special, consequential or otherwise and howsoever arising including, but not limited to, loss of profit or loss or damage arising from breakdown or failure of the software supplied by KELTON®, or negligence of KELTON®, and in no event shall the total liability of KELTON® (howsoever arising) exceed the amount paid by the User for the Module covered by this Agreement.

6 Force Majeure

Neither party to this Agreement shall be liable for failure to perform, or for delay in performing, its obligations hereunder if such failure or delay shall be due to Acts of God, war, riot, civil commotion, weather, labour disputes, failure of sub-contractors or any other cause beyond the reasonable control of the party concerned and whether or not a similar nature to the foregoing. Performance shall be suspended during said period and if such period shall continue for more than 4 weeks then either party shall be entitled to terminate this Agreement with no claims being due to or by either party in respect thereof.

7 Non-Assignment

The User is not permitted to assign the benefits and obligations of this Agreement to a third party without the prior and written consent of KELTON®.

8 Law

This Agreement shall be construed and operated in accordance with Scots Law.

9 Support

KELTON® will attempt to answer specific customer support requests. This service is offered on a "best endeavours" basis and KELTON® may not be able to resolve every support request, and in the absence of contract to the contrary will have no liability for failure to respond or for error or negligence in responding.

Ongoing customer support is available following the 90 day warranty period by separate contract.

Kelton Engineering Ltd
The MacKenzie Building
168 Skene Street
Aberdeen, AB10 1PE
Scotland
t: +44(0)1224 630000 f: +44(0)1224 630004
e: flocalc@kelton.co.uk <<mailto:flocalc@kelton.co.uk>>

Bibliography

Harold Elliott Rusty, Means W. Scott *XML in a nutshell*. 2nd ed., USA: O'Reilly; 2002.

Sybex Inc., *ASP, ADO and XML Complete*. 1st ed., USA: SYBEX; 2001

Curland, Matthew J, *Advanced Visual Basic 6, Power Techniques For Every Day Programs*. 1st ed., USA: Addison Wesley Professional; 2000.

MSDN Library, *Microsoft Developer Network*. USA: Microsoft Corp.; 2001 October

The Internet

References

- 1 Tonner Gilbert M, Metering Management – the Future, *Measurement + Control*, 2002 July; 35. pp. 168-172.

- 2 Standards Glossary *The ISO Standards Glossary*. [homepage on the Internet]. c.2003-2005 [updated 2005 Nov 26; cited 2005 Nov 29]. Available from:
<http://www.standardsglossary.com/isob.htm>

- 3 Wikipedia, The Free Encyclopaedia *Multitier architecture*. [homepage on the Internet]. c.2004-2005 [updated 2005 Oct 28; cited 2005 Nov 28]. Available from: http://en.wikipedia.org/wiki/Multitier_architecture

- 4 Webopedia *CSS definition*. [homepage on the Internet]. Jupitermedia Corp.; c.2002-2005 [updated 2005 Nov 26; cited 2005 Nov 28]. Available from:
<http://isp.webopedia.com/TERM/C/CSS.html>

- 5 Webopedia *XSL definition*. [homepage on the Internet]. Jupitermedia Corp.; c.2002-2005 [updated 2005 Nov 22; cited 2005 Nov 28]. Available from:
<http://isp.webopedia.com/TERM/X/XSL.html>

- 6 <http://www.theScrams.com> *TheScrams XML Document Object Model (DOM) Tutorial*. [homepage on the Internet]. c.2001-2005 [updated 2005 Nov 25;

cited 2005 Nov 29]. Available from:

<http://www.thescarms.com/XML/DOMTutorial.asp>

- 7 <http://searchwebservices.techtarget.com/> *What is XSD?, Definition.*
[homepage on the Internet]. c.2002-2005 [updated 2005 Oct 13; cited 2005 Nov 29]. Available from:
http://searchwebservices.techtarget.com/sDefinition/0.,sid26_gci831325,00.html

- 8 KELTON® *Overview of our Products.* [homepage on the Internet]. Aberdeen; c.2003-2005 [updated 2004 May 10; cited 2005 Nov 29]. Available from:
<http://www.kelton.co.uk/products/overview.asp>

- 9 TUV NEL *Welcome to PPDS.co.uk.* [homepage on the Internet]. Glasgow; c.2000-2005 [updated 2005 Oct 28; cited 2005 Nov 29]. Available from:
<http://www.ppds.co.uk>

- 10 Department of Trade and Industry (DTI) - Oil and Gas Division *DTI Guidance Notes For Petroleum Measurement Under The Petroleum (Production) Regulations.* [serial on the Internet] 2002 March [cited 2005 Nov 29]; 1(16). Available from:
<http://www.og.dti.gov.uk/upstream/measurement/MeasGuidelines.pdf?nourl=www.dti.gov.uk/publications/pdflink/&pubpdfload=02%2F1441>

- 11 MSDN, Microsoft Developer Network *CAPICOM Reference*. [homepage on the Internet]. USA: Microsoft Corp.; c.2005-2006 [updated 2005 Apr 20; cited 2006 Apr 24]. Available from:
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/seccrypto/security/capicom_reference.asp
- 12 Wikipedia *Dynamic-link Library*. [homepage on the Internet]. C.2005-2007 [updated 2006 Dec 30, cited 2007 Jul 16]. Available from
http://en.wikipedia.org/wiki/Dynamic-link_library
- 13 Wikipedia *Component Object Model*. [homepage on the Internet]. c.2004-2006 [updated 2005 Dec 25; cited 2006 Apr 24]. Available from:
http://en.wikipedia.org/wiki/Component_Object_Model
- 14 Information Commissioners Office *Rules on email marketing, What do they mean for Individuals?*, [homepage on the Internet]. United Kingdom: Uk Government; c.2006-2007 [updated 2007 Mar 09; cited 2007 Jul 1]. Available from:
http://www.ico.gov.uk/upload/documents/library/privacy_and_electronic/introductory/rules~1.pdf

- 15 Information Commissioners Office *The 8 principals in place by the Data Protection Act 1988*. [homepage on the Internet]. United Kingdom: Uk Government; c.2003-2005 [updated 2005 Nov 29; cited 2005 Nov 29].
Available from:
<http://www.informationcommissioner.gov.uk/eventual.aspx?id=302>

- 16 The Free Dictionary *Dynamic HTML*. [homepage on the Internet]. Farlex Inc.; [cited 2005 Nov 29]. Available from: <http://computing-dictionary.thefreedictionary.com/Dynamic%20HTML>

- 17 <http://www.google.co.uk/> *Definition of JavaScript on the web*. [homepage on the Internet]. [cited 2005 Nov 29]. Available from:
<http://www.google.co.uk/search?hl=en&lr=&oi=defmore&defl=en&q=define:JavaScript>

- 18 <http://www.google.co.uk/> *Definition of HTML Form on the web*. [homepage on the Internet]. [cited 2005 Nov 29]. Available from:
<http://www.google.co.uk/search?hl=en&lr=&oi=defmore&defl=en&q=define:HTML+FORM>

- 19 <http://www.wisegeek.com> *What is XML or Extensible Markup Language*. [homepage on the Internet]. Conjecture Corp.; c.2005 [updated 2005 Nov 10;

cited 2005 Nov 28]. Available from: www.wisegeek.com/what-is-xml-or-extensible-markup-language.htm

- 20 The free Dictionary *ASP*. [homepage on the Internet]. Farlex Inc.; [cited 2005 Nov 29]. Available from: <http://computing-dictionary.thefreedictionary.com/ASP>

- 21 Sun Developer Network (SDN) *Code Samples and Apps, Applets*. [homepage on the Internet] Sun Microsystems Inc., c.1998-2005 [updated 2005 Nov 26; cited 2005 Nov 30]. Available from: <http://java.sun.com/applets/>

- 22 Qusay H. Mahmoud *Developing Web Applications with JavaServer Faces*:. [serial on the Internet] 2003 [cited 2005 Jul 1]. Available from: http://java.sun.com/developer/technicalArticles/GUI/JavaServer_Faces/

- 23 Selena Sol *Introduction to the Web Application Development Environment (Tools) 1999 May*. [homepage on the Internet]. Jupitermedia Corp.; c.1999-2005 [updated 2005 Aug 1; cited 2005 Aug 1]. Available from: <http://www.wdvl.com/Authoring/Tools/Tutorial/>

- 24 Webopedia *HTML Definition*. [homepage on the Internet]. Jupitermedia Corp.; c.2001-2005 [updated 2005 Nov 24; cited 2005 Nov 30]. Available from: <http://www.webopedia.com/TERM/H/HTML.html>

- 25 SearchQualitySoftware.com *XHTML*. [homepage on the Internet]. TechTarget; c.2000-2006 [cited 2006 Jun 22]. Available from:
http://searchoracle.techtarget.com/gDefinition/0,294236,sid41_gci213550,00.html
- 26 WebAsyst *Glossary of Terms used in this site -Plug-in*. [homepage on the Internet]. c.2004-2006 [updated 2006 Apr 23; cited 2006 May 01]. Available from: <http://www.webasyst.net/glossary.htm>
- 27 MSDN, Internet Developer Explorer Centre *Introduction to Active X controls*. [homepage on the Internet]. Microsoft Corp.; c.2001-2006 [updated 2006 Feb 06; cited 2006 Apr 14]. Available from:
<http://msdn.microsoft.com/library/default.asp?url=/workshop/components/activex/intro.asp>
- 28 Wikipedia, the free encyclopaedia *AJAX (Programming)*. [homepage on the Internet]. c.2005-2006 [updated 2006 Oct 05; cited 2006 Nov 23]. Available from: <http://en.wikipedia.org/wiki/AJAX>
- 29 Wikipedia, the free encyclopaedia *Hyper Text Transfer Protocol*. [homepage on the Internet]. c.2003-2006 [updated 2006 Mar 22; cited 2006 Nov 24]. Available from: <http://en.wikipedia.org/wiki/HTTP>

- 30 Webopedia *MIME*. [homepage on the Internet]. Jupitermedia Corp.; c.2000-2006 [updated 2006 Apr 23; cited 2006 May 01]. Available from:
<http://www.webopedia.com/TERM/M/MIME.html>
- 31 <http://searchwebservices.techtarget.com/> *Document Type Definition*. [homepage on the Internet]. c.2002-2006 [updated 2006 Jan 04; cited 2006 Jan 22]. Available from:
http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci213918,00.html
- 32 Microsoft Corp. *Internet Information Service*. [homepage on the Internet]. c.2003-2006 [updated 2006 Jan 17; cited 2006 Jan 23]. Available from:
<http://www.microsoft.com/WindowsServer2003/iis/default.msp>
- 33 The Apache Software Foundation *Apache HTTP Server Project*. [homepage on the Internet]. c.2000-2006 [updated 2006 Jan 19; cited 2006 Jan 23]. Available from: <http://httpd.apache.org/>
- 34 MSDN Magazine, The Microsoft Journal for Developers *Com: Handle Late bound Events with Visual Basic using an ALT Bridge*. [homepage on the Internet]. Microsoft Corp.; c.2003-2006 [updated 2006 Jan 29; cited 2006 Apr 14]. Available from:
<http://msdn.microsoft.com/msdnmag/issues/01/03/connpoints/>

- 35 Morcillo Eduardo A. *Namespace Edanmo Visual Basic 6*. [homepage on the Internet]. c.2005-2006 [updated 2006 Mar 07; cited 2006 Apr 21]. Available from: <http://www.mvps.org/emorcillo/en/code/vb6/index.shtml>
- 36 MSDN, Visual Basic Developer Centre *Common Dialog Control for Visual Basic 6.0 Users*. [homepage on the Internet]. Microsoft Corp.; c.2007 [updated 2007 Feb 07; cited 2007 Feb 14]. Available from: [http://msdn2.microsoft.com/en-us/library/256tssz7\(VS.80\).aspx](http://msdn2.microsoft.com/en-us/library/256tssz7(VS.80).aspx)
- 37 Sun Microsystems Inc. *Designing Enterprise Applications with the J2EE Platform Second Edition, 11.1.1 Model View-Controller Architecture*. [homepage on the Internet]. c.2002-2006 [updated 2006 Feb 04; cited 2006 Apr 14]. Available from: http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/app-arch/app-arch2.html
- 38 Wikipedia, The Free Encyclopedia *Active X Data Objects*. [homepage on the Internet]. c.2005-2006 [updated 2005 Dec 15; cited 2006 Apr 14]. Available from: http://en.wikipedia.org/wiki/Active_X_Data_Objects

- 39 Wikipedia, The Free Encyclopedia *Data Access Objects*. [homepage on the Internet]. c.2005-2006 [updated 2005 Dec 15; cited 2006 Apr 14]. Available from: http://en.wikipedia.org/wiki/Data_Access_Objects
- 40 Connolly Dan-Editor of the [HTML 2.0 Specification](#) *Overview of SGML Resources*. [homepage on the Internet]. W3C; c.1997-2006 [updated 2006 Apr 05; cited 2006 Apr 17]. Available from: <http://www.w3.org/MarkUp/SGML/>
- 41 Webopedia *What is API?* [homepage on the Internet]. Jupitermedia Corp.; c.2000-2006 [updated 2006 Apr 24; cited 2006 Apr 24]. Available from: <http://www.webopedia.com/TERM/A/API.html>
- 42 Megginson David *About SAX, Official Website for SAX*. [homepage on the Internet]. Megginson Technologies Ltd; c.2001-2006 [updated 2006 Apr 11; cited 2006 Apr 16]. Available from: <http://www.saxproject.org/>
- 43 XMLFiles.com *XML DTD – An Introduction to XML Document Type Definitions*. [homepage on the Internet]. Jupitermedia Corp.; c.2003-2006 [updated 2006 Feb 09; cited 2006 Apr 16]. Available from: <http://www.xmlfiles.com/dtd/>
- 44 Nyffenegger Rene *XML Schema vs DTD*. [homepage on the Internet]. c.2004-2006 [updated 2006 Apr 26; cited 2006 May 29]. Available from: http://www.adp-gmbh.ch/xml/schema_vs_dtd.html

- 45 Wikipedia, The Free Encyclopaedia *Integrated Development Environment*.
[homepage on the Internet]. c.2004-2006 [updated 2006 Feb 06; cited 2006
Apr 16]. Available from:
http://en.wikipedia.org/wiki/Integrated_development_environment
- 46 TopXML.com *MarrowSoft Xselerator XSL Editor and Debugger*. [homepage on
the Internet]. c.2001-2006 [updated 2006 Apr 23; cited 2006 Apr 24]. Available
from: <http://www.topxml.com/xselerator/>
- 47 Wikipedia, The Free Encyclopaedia *Microsoft JET Database Engine*.
[homepage on the Internet]. c.2005-2006 [updated 2006 Feb 21; cited 2006
Apr 16]. Available from:
http://en.wikipedia.org/wiki/Microsoft_Jet_Database_Engine
- 48 LearnThat.com. *OLE Definition*. [homepage on the Internet]. c.2005-2006
[updated 2006 Apr 06; cited 2006 Apr 16]. Available from:
<http://www.definethat.com/define/81.htm>
- 49 Refsnes Data *Introduction to SQL*. [homepage on the Internet]. W3C.; c.2001-
2006 [updated 2006 Apr 14; cited 2006 Apr 16]. Available from:
http://www.w3schools.com/sql/sql_intro.asp

- 50 Webopedia *What is DCOM?* [homepage on the Internet]. Jupitermedia Corp.; c.2001-2006 [updated 2006 Apr 12; cited 2006 Apr 21]. Available from: <http://www.webopedia.com/TERM/D/DCOM.html>
- 51 DevX.com *UML for the Software Developer, Part 1:Building Classes*. [homepage on the Internet]. Jupitermedia Corp.; c.2004-2006 [updated 2005 Jun 03; cited 2006 Apr 16]. Available from: <http://www.devx.com/enterprise/Article/22515>
- 52 Curland, Matthew J *Advanced Visual Basic 6, Power Techniques For Every Day Programs*. 1st ed., USA: Addison Wesley Professional; 2000.
- 53 Wikipedia, The Free Encyclopaedia *Multiple Document Interface*. [homepage on the Internet]. c.2004-2006 [updated 2005 Dec 14; cited 2006 Apr 16]. Available from: http://en.wikipedia.org/wiki/Multiple_document_interface