

LQ Framework

Executive Overview

Reference **Lq-overview**
Date **13 June 2005**
Version **1.1**

Contents

1	INTRODUCTION	3
2	PRODUCT BACKGROUND	4
3	LQ FRAMEWORK OVERVIEW	5
4	LQ DOCUMENT FEATURES	8
5	LQ SIGNATURE CONTROL FEATURES	9
6	XML DATA HANDLING	13
7	PDF DOCUMENT HANDLING	14
8	TEXT DOCUMENT HANDLING	15
9	TIFF OUTPUT	15
10	FURTHER INFORMATION	15

1 Introduction

- 1.1 The LQ framework is a technology jointly developed by eCom Asia Pacific Pty and Florentis Ltd. It has been designed to allow any new or existing computer system requiring handwritten signatures to be adapted for electronic signing cost-effectively.
- 1.2 This document is intended to provide an executive overview of the system. It includes a review of the background to the framework and a description of its features and capabilities.

2 Product Background

- 2.1 The benefits of signing documents electronically instead of on paper are well established, and ostensibly include lower processing costs, improved security, and much greater flexibility.
- 2.2 In practise organisations are frequently deterred from adopting the technology for a number of reasons:
 - 2.2.1 Very commonly companies have existing computer systems to handle their business processes. Adapting them such that they work with signature-enabled applications is typically difficult and expensive.
 - 2.2.2 Off-the-shelf signature solutions consist of add-ins that work with existing document applications, the most common being Adobe Acrobat, Microsoft Word and Microsoft Excel. The cost of the basic applications is prohibitive unless the organisation already licenses them for other purposes.
 - 2.2.3 Where plug-ins are used they suffer from technology instability, resulting from the constant introduction of product upgrades, which is driven by the need to generate new licensing revenue. This has proven to be very costly and disruptive to existing signature solutions.
- 2.3 The LQ framework has been designed to overcome these problems, allowing organizations of any size to implement stable electronic signing systems cost-effectively, with a minimum of disruption to existing systems.

3 LQ Framework Overview

3.1 Document Format

The most important feature of the LQ framework is the fundamental document technology, namely the use of XHTML Version 1.1. All documents signed using LQ must be in this format. XHTML was chosen because it is a standard that has been developed by the World Wide Web Consortium (W3C) and as a result benefits from the following:

- 3.1.1 The standard is fixed, and therefore not susceptible to instability resulting from proprietary ownership.
- 3.1.2 XHTML documents are platform independent.
- 3.1.3 The technology is free of licensing charges.
- 3.1.4 XHTML documents can be displayed in any web browser.

3.2 Signature Control

To allow XHTML documents to be signed electronically LQ provides a signature component. The control handles the collection and display of signatures, ensuring that the highest quality forensic data is collected to satisfy all legal admissibility requirements. It is responsible for hashing the document at the time of signing, ensuring that the content is not affected by the browser document model.

Note: Although XHTML documents are platform independent, the signature component is designed to work with specific browsers. At present LQ supports Microsoft Internet Explorer V6 and above working on Microsoft Windows 2000 and later.

3.3 Document Creation

LQ has been designed to provide maximum flexibility and ease of use in the way in which documents are created. Of primary importance is the fact that it provides complete independence from the architecture and technology used by the systems used to gather the document data.

- 3.3.1 The most powerful and efficient way of using LQ is to supply the input data in the form of an XML configuration file, and to generate the final document using an XML document template. The advantage of XML is that it consists of simple text files that can be readily created and modified using a basic text editor. LQ templates are very powerful, allowing complete sets of different document types to be specified in a single location. The configuration file is used to specify how a document is to be constructed from the document sections supplied in the template, and the location of the signature areas needed.
- 3.3.2 Many existing systems generate PDF format documents which are currently printed and signed on paper. To avoid the upheaval of modifying such systems

LQ allows PDF documents to be supplied for signing. They are initially converted to XHTML format and a configuration file is used to insert signatures at the required locations.

- 3.3.3 Some regulatory authorities demand that signed documents are stored in PDF format. LQ provides the mechanism needed to convert the signed XHTML document to PDF form for this purpose.

Note: the PDF form of document stores the signature as an image. The document integrity hash remains with the signature in the original XHTML document.

- 3.3.4 An alternative form of input is supported for systems which generate raw text files.
- 3.3.5 A facility is provided for converting signed HTML documents into TIFF images, where appropriate.

3.4 Document Presentation

The LQ system is designed to replace existing document applications, and must therefore provide an equivalent standard of presentation. Unlike typical web documents the framework allows full control over pagination, including the printed output with headers and footers, and graphical images are embedded internally, not as external links.

- 3.4.1 To provide the user controls needed to navigate around the document and to control the signing process LQ uses a viewer page which is displayed in the browser with the actual document embedded.
- 3.4.2 Alternatively the page can be displayed within client (i.e. non web-based) applications when appropriate.

Document Viewer - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Page 1 of 2 [Fast] [Prev] [Next] [Last] [Goto] [Print Preview] [Print Document] [Sign] [Delete] [Done]

Ref No. demo101 Agency Number 1234 Form Version 1.00

**Proposal for Life Assurance
Single Premium / Recurrent Single Premium**

WARNING: STATEMENT PURSUANT TO SECTION 28(5) OF THE INSURANCE ACT (CAP 143), YOU ARE TO DISCLOSE IN THIS PROPOSAL FORM FULLY AND FAITHFULLY, ALL THE FACTS WHICH YOU KNOW OR OUGHT TO KNOW, OTHERWISE YOU MAY RECEIVE NOTHING FROM THE POLICY.

Personal Details of Life to be Assured

Full Name (According to Passport / Birth Certificate)
full-name

Title Gender Married
Mr. male Yes No

Signature of Client

X

Token required: YES NO

*** YOU HAVE COMPLETED THE PERSONAL DETAILS OF LIFE TO BE ASSURED PLEASE PROCEED TO THE NEXT SECTION ***

Done Internet

3.5 Post-signing Processing

When a document has been signed the signature images are displayed in the conventional manner within the signatures areas, but the signature data is held within the component; the document remains unchanged. Therefore before the signed document can be stored the signature data must be inserted into the body. Typically this is done by an LQ component on the server, but in fat-client applications the same component can be used locally.

- 3.5.1 The server-side components can be used for a range of tasks including the automated checking of document integrity without needing it to be displayed in a browser window.

3.6 Platforms

- 3.6.1 LQ is currently supplied in either Java or .Net versions

4 LQ Document Features

- 4.1 All LQ documents must be in XHTML format. The system is capable of generating suitable documents in a number of different ways.
- 4.2 LQ documents have to be self-contained and capable of supplying presentation-quality documents equivalent to the standard of applications such as Adobe Acrobat and Microsoft Word. Conventional web-documents do not fulfil this requirement for a number of reasons:-
 - 4.2.1 HTML pages do not normally have the level of pagination control needed. In most examples long documents are presented as a single scrollable page, sometimes with embedded links to allow rapid transfer to specific location. This level of functionality is unacceptable for LQ applications.
 - 4.2.2 The printed form of HTML documents typically differs significantly from the content displayed in the browser. It is essential that the printed form of the document should also be of presentation quality, and with appropriate headers and footers.
 - 4.2.3 When graphical images are used in web pages they consist of links (URLs) to external data sources. This is unacceptable in legally binding documents which must be self contained.
- 4.3 The LQ framework overcomes these deficiencies whilst adhering to the XHTML standard. In particular:
 - 4.3.1 The user has full control over the document structure, and controls are provided to navigate around it. In web-applications the document is presented within an HTML viewer page which provides the controls used to navigate, sign, print and submit the document. Alternatively a fat-client application can be used which fulfils the same requirements by presenting the document in a browser control.
 - 4.3.2 The printed output is formatted to match the document as it is displayed in the browser, and appropriate headers and footers are supplied.
 - 4.3.3 Any number of graphical images may be embedded within the document as data. These are displayed in the browser and printed in the expected manner, but are actually contained as character data within the document itself.
- 4.4 To ensure integrity of the data LQ documents are hashed on the server before being sent to the browser on the client. Before signing the signature component checks the pre-calculated hash to ensure there has been no data corruption.

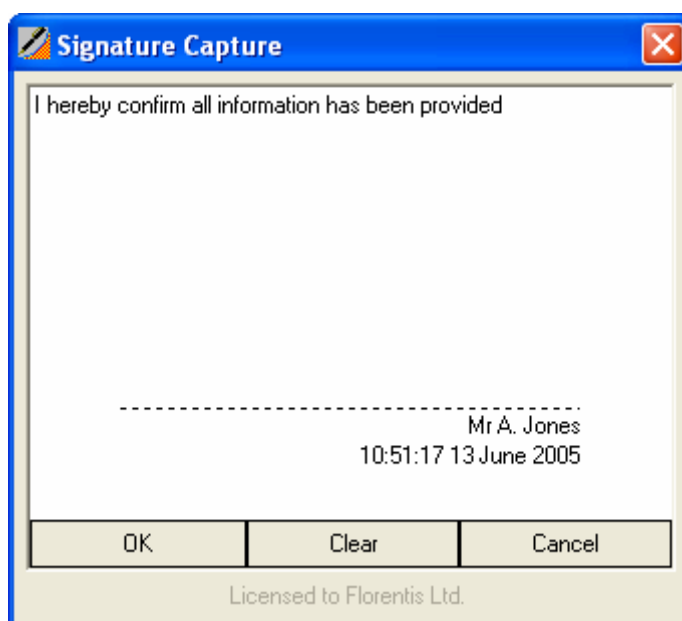
5 LQ Signature Control Features

5.1 The signature control is responsible for binding each handwritten signature to the XHTML document. It fulfils a number of functions including:

- The collection of the pen data and contextual information during signing.
- The hashing of the document.
- Streaming the data to a single entity
- Displaying the signature image in the browser and for printing.

5.2 Signature Data

5.2.1 The signing process is started using the **Sign** button provided by the viewer page or application. The component then calculates the document hash and presents the user with the Signature Capture dialog box:



5.2.2 The capture box confirms the reason for signing the document, the name of the person signing, and the current date and time. The signature can then be collected by signing on any suitable digitizer tablet, or on a Tablet PC. If an inking digitizer is being used (i.e. a pad with an in-built LCD display screen such as the Interlink ePad-Ink) then the contents of the dialog box are shown there as well.

5.2.3 During signing the component collects all available pen data reported by the device. This depends to some extent on the digitizer type and configuration, but should always include the basic pen movement (x, y and t). Some devices also report the pen “pressure” (actually force), orientation, inclination and twist.

5.2.4 At the same time ancillary forensic information is collected, including:

- The type of the digitizer and the version of the device driver used.
- The unique identifier of the digitizer (if available)
- The type and version of the operating system used on the host PC.
- The Network Interface Card address.

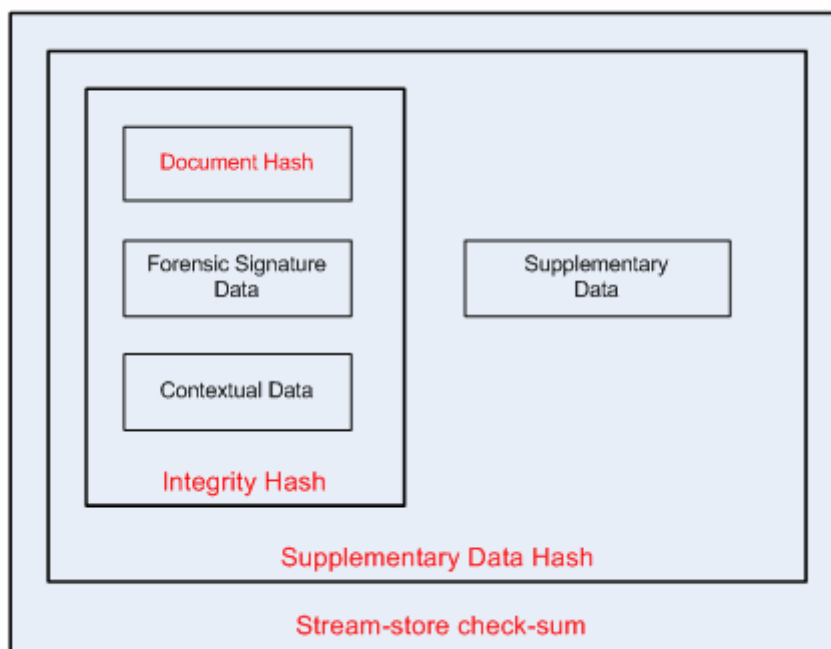
This information is inserted into the signature data envelope.

5.3 Data Structure

5.3.1 The signature data is stored in the form of independent data streams. The advantage of this format is that it preserves forward and backward compatibility of the data. Old applications can read new data with additional data components without a problem, and new applications can also handle old data. This robustness is essential for signed documents that may be stored for many years.

5.3.2 Each data stream is stored in a platform-independent format which ensures that it can be read and handled regardless of the system being used. For example, a signature created on a Palm device can be handled *without modification* on a Windows PC, despite the differences in byte ordering.

5.3.3 The data is structured as illustrated in the figure below:



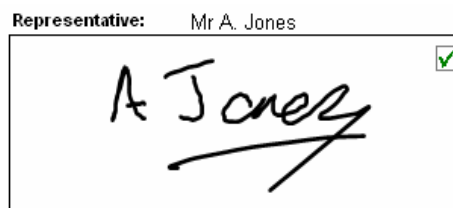
5.3.4 The document hash, signature data and contextual data are protected by an integrity hash.

- 5.3.5 A facility exists to add supplementary data to the signature, and this is protected by a separate hash. Supplementary data can be added and read, but like the rest of the data cannot be modified or deleted.
- 5.3.6 The overall signature data is protected by a stream-store check-sum.
- 5.3.7 The size of the signature data depends on the signature duration and the pad capabilities, but most signatures are less than 2k in binary form.

5.4 Document Hashing

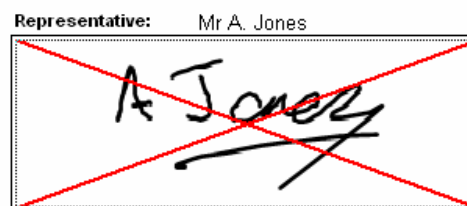
- 5.4.1 Before the document is displayed in the browser the document hash is calculated on the server, and embedded into the document as a comment.
- 5.4.2 At the time of signing the document hash is recalculated in the browser and compared with the one calculated previously. The two hashes must be identical for the signing process to continue.
- 5.4.3 The document hash is inserted into the signature data.
- 5.4.4 Once a signature has been added the hash is re-calculated whenever the document is displayed. If the hash matches then a tick is shown next to the signature to confirm that it has been checked:

The representative confirms that no information has been withheld which may influence the acceptance of the proposal by the company.



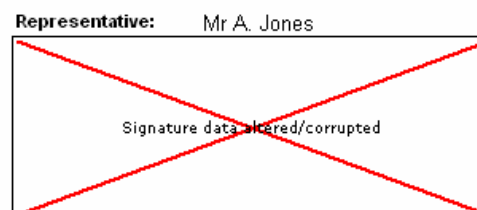
- 5.4.5 Each signature is invalidated (shown crossed out) if a discrepancy is found:

The representative confirms that no information has been withheld which may influence the acceptance of the proposal by the company.



- 5.4.6 If the signature data is modified then it is shown invalidated:

The representative confirms that no information has been withheld which may influence the acceptance of the proposal by the company.



5.4.7 The document hash is calculated from the original document string and is therefore independent of the type of browser being used. (At present only Internet Explorer is supported, but when the component is implemented for other browsers LQ documents will be interchangeable).

5.5 Additional Information

5.5.1 The signature component does not insert the signature data into the document. When the signing process has been completed the signatures are submitted back to the server where the LQ Document component is used to insert them into the document after the hashes have been checked.

5.5.2 Only the signature data is submitted back to the server; the document is not sent because it is already held on the server. This behaviour reduces network traffic, an important consideration on installations with modest connection speeds.

6 XML data handling

- 6.1 The most efficient and preferred method of generating XHTML documents uses the XML2HTML interface. The data collection system supplies the user data in the form of an XML configuration file which specifies how the final XHTML document should be constructed from a template.



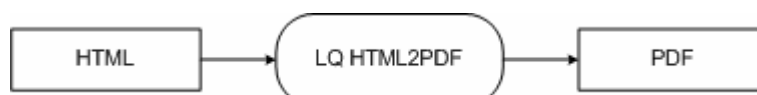
- 6.2 The XML template contains a definition of many different document sections. In any organization there may be hundreds of different document types, but typically many of them contain common sections. The LQ template allows each section to be defined separately within the same template, and then selected parts can be combined to form the full document using the configuration file.
- 6.3 Since XML data consists solely of text information the template can easily be created and maintained by anyone with knowledge of XHTML. It is anticipated that at the majority of sites the template will be developed by the customer, but if the resources are not available then this can be offered as a service.
- 6.4 The configuration file consists of a set of references to document sections found in the template, together with any user-specific data that should be inserted into it. This includes general information that has to be presented as well as the configuration of the signature areas to be supplied, comprising the number of signatures, their location in the document and the name and reason for signing to be displayed.

7 PDF document handling

- 7.1 The LQ framework can accept PDF files as input to the system if XML data is not readily available. In addition the final, signed XHTML document may be converted to PDF form for archiving. The two conversions are completely independent and can be used separately or together.
- 7.2 The LQ PDF2HTML interface is used to convert PDF input documents to XHTML form ready for signing. It is intended for use by systems that already produce PDF documents, and which were originally to be printed for signing on paper. It is emphasised that wherever possible the XML2HTML solution is to be preferred.
- 7.2.1 The advantage of using PDF input is that documents can be signed without requiring expensive Adobe systems, but just using a browser.
- 7.2.2 The conversion of a PDF document uses an XML template which is constant for the type of document being converted, and a user-specific configuration file.



- 7.2.3 The template is used to define the possible location of signature areas for different types of document. Each signature is located using a specified search string which is located within the PDF document, and some offset data. When the document is converted to XHTML form the search string is found and the signature is inserted at the given offset relative to it.
- 7.2.4 The configuration file is used to identify the type of document to use from the template, and the details for each of the signatures needed.
- 7.3 Following the successful signing of the XHTML document and the insertion of the signature data into it, an alternative version may be required in PDF form. This is produced using the LQ HTML2PDF interface.

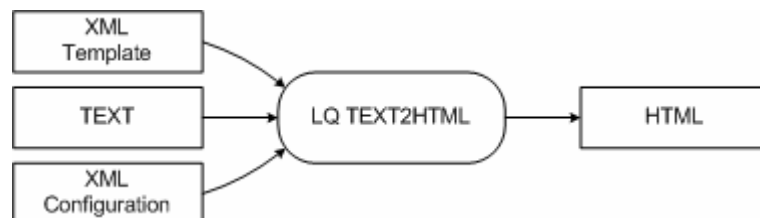


- 7.4 No configuration files are needed to perform the conversion.

- 7.5 The converted (PDF) file cannot be tested for integrity. The document hash remains in the XHTML version which should be retained if this capability is required.

8 Text document handling

- 8.1 An alternative method of input is provided by the TEXT2HTML interface which allows the user data to be supplied in the form of a text file. It is intended for very basic systems which generate text files relying on a fixed-pitch font to preserve layout.
- 8.2 The component works in a similar way to the PDF2HTML component, requiring a template to define the location of signatures in particular types of documents, and a configuration file to select the document type and specify the signature data.



9 TIFF Output

Signed documents can be converted to TIFF images using the HTML2TIFF component. This facility may be required by some regulatory authorities but the resulting image cannot be checked for integrity. When the integrity has to be checked this should be done using the signed HTML file.

10 Further Information

More detailed information is available in the LQ Framework User Manual.