1 Introduction

This is the program specification for development of release-1 of EasiCompare. Release-1 focuses on implementing the EasiCompare Lite functionality as development timescales preclude the delivery of the full EasiCompare functionality within project timescales.

EasiCompare Lite functionality is specified in the Customer Requirements specification document for EasiCompare [Reference-1].

2 User Interface

2.1 EasiCompare is invoked from the command line and accepts a set of arguments form the command line. The arguments consist of:

(a) Mode - An operating mode flag (optional)
(b) File1 – The path to the first file to compare.
(c) File2 – The path to the second file to compare.

2.2 The Mode parameter takes the form of a slash '/' followed by the mode name. The modes supported are:

(a) /VERBOSE – Full reporting of errors and outcomes.
(b) /SILENT – No reporting of errors or outcomes to the screen.

If not mode is specified then the default mode operates. Mode parameter values can be shortened by dropping characters from the end and so /V, /VR, /VERB etc are equivalent to /VERBOSE and /S and other shortenings of /SILENT are equivalent to /SILENT. Matching of the mode parameter is not case sensitive.

If an invalid Mode parameter value is present then the system will report a mode parameter error.

2.3 The files to be compared are specified as two parameters following the optional mode parameter. These two parameters are file system paths that are used to locate the files to compare. The system will operate with:

(a) Relative paths.
(b) Absolute paths.

Any valid windows path format is supported including network based paths that identify the server on which the file resides. Paths with spaces in are specified using quotes. The path can specify files on the local disk and files on remote storage, including shared folders.

2.4 The system performs basic parameter validation when the programme is invoked. This validates that that the sequence and form (syntax) of parameters are correct. If errors are found then they are reported and operation halts, if none are found then file location and comparison takes place. Errors detected and reported at this stage are:

- Invalid mode specified.
- Incorrect number of file paths provided.
- File1 or File2 parameters are not valid path formats.

The error report(s) include the parameter value that has a problem together with the nature of the problem. The errors format is:

ERROR:	Description of the type of error detected. Details of any variable data triggering the error.

Errors reported here are:

ERROR:	Incorrect number of parameters supplied. Number of parameters supplied is : X

ERROR:	Invalid mode supplied. Mode supplied is /ABCDEF

ERROR:	Invalid path supplied. Path supplied is [][][(0::C\

Note : Error messages wordings are indicative rather than definitive, the exact wording will be determined during development and this document may not be updated.

3 Comparison Operation

3.1

Following basic parameter validation the system operates as follows:

- Check that File1 identifies a file that is valid for use in file contents comparison.
- If File1 does not identify a suitable file then report an error including (a) the nature of the problem detected and (b) relative and absolute path information to identify the file that is causing the problem.
- Check that File2 identifies a file that is valid for use in file contents comparison.
- If File2 does not identify a suitable file then report an error including (a) the nature of the problem detected and (b) relative and absolute path information to identify the file that is causing the problem.
- If either File1 or File2 have problems then abort the sequence.
- Compare the contents of File1 and the contents of File2. If they are different then the program reports the difference, or if they are identical then it report them as identical.

3.1.1

Problems with files reported by EasiCompare are:

- A file does not exist.
- A file cannot be opened.
- The file to be compared is not a normal file suitable for comparison. Special cases that the program can identify are to be individually reported to identify why the final is not 'normal', if this can not be determined then a simple 'not a normal file' error report is given. Specific cases of abnormal files identified as the time of specification for which dedicated error messages are required are:
  - The specified file is a directory.

3.1.2

If there are no problems with the files identified by File1 and File2 parameters then the program will report the outcome if it is operating in /VERBOSE mode. The report contains:

- A message indicating that the comparison is being done.
- The relative and absolute path information for each of the files.
- The result of the comparison – IDEN|TICAL or DIFFERENT.

3.2

Output Messages

Error messages generated during this stage are:

ERROR:	File does not exit Path supplied is : FileOne.txt Absolute path supplied is : C:\TEMP\FileOne.txt

ERROR:	File cannot be opened Path supplied is : FileOne.txt Absolute path supplied is : C:\TEMP\FileOne.txt

ERROR:	File is not a normal file. Path supplied is : FileOne.txt Absolute path supplied is : C:\TEMP\FileOne.txt

ERROR:	File is a directory. Path supplied is : FileOne.txt Absolute path supplied is : C:\TEMP\FileOne.txt

Note : Error messages wordings are indicative rather than definitive, the exact wording will be determined during development and this document may not be updated.

If a successful comparison is performed then the output message is of the form:

```
Path supplied is : FileOne.txt
Absolute path supplied is : C:\TEMP\FileOne.txt
Path supplied is : FileTwo.txt
Absolute path supplied is : C:\TEMP\FileTwo.txt
The files are different.
```

Or

```
Path supplied is : FileOne.txt
Absolute path supplied is : C:\TEMP\FileOne.txt
```

	Path supplied is : FileTwo.txt Absolute path supplied is : C:\TEMP\FileTwo.txt The files are identical. Note : Output message wordings are indicative rather than definitive, the exact wording will be determined during development and this document may not be updated.
4.	**Implementation Notes**
4.1	The implementation needs to deal with the user specifying a path that points to a shortcut rather than to a file. In this case it is not the shortcut that gets compared with the other file but the content of the file located via the shortcut. Output messages should be amended to convey this information.
4.2	Consideration should be given to the way the component operates when used from scripts where it could be repeatedly invoked to perform comparisons of large numbers of files. The design should ensure that there are not overheads that mean that under these circumstances EasiCompare is not seen as 'too slow' to be used and is replaced by alternative comparison operations.
A	**References**
R1	EasiCimpare Requirements – February 2009 Issue 0.2 DRAFT