Requirements Decomposition Analysis

Allen P. Nikora
Jet Propulsion Laboratory,
California Institute of Technology
Pasadena, CA
Allen.P.Nikora@jpl.nasa.gov

The work described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology. This work is sponsored by the National Aeronautics and Space Administration's Office of Safety and Mission Assurance under the NASA Software Program led by the NASA Software IV&V Facility. This activity is managed locally at JPL through the Assurance Technology Program Office (ATPO).
Requirements Decomposition Analysis

Task Description

- **Problem Statement:** Requirements play a pivotal role in planning, selection, development, testing and operation of NASA's missions. Starting from mission objectives, requirements are successively decomposed. The correctness of this decomposition is critical, yet V&V of this crucial step is limited to manual inspection and pointwise testing, which are cumbersome and fallible (e.g., Mars Polar Lander).

- **Task:** Rigorous lightweight analysis methods for requirements decomposition have been developed by the software engineering research community, and have shown promise in successful application to critical systems (e.g., rail transportation). This proposal will study their application to the V&V of spacecraft software requirements, to ascertain if, when and how they are suitable for use by NASA.
Requirements Decomposition Analysis

Goals and Objectives

- **Goal**: study the applicability to NASA spacecraft requirements of rigorous analysis methods for requirements decomposition that have been developed by the software engineering research community.

- **Objectives**:
  1. Manually apply decomposition analysis methods applied to spacecraft requirements.
  2. Based on the results of these application studies, emerge with recommendations for the application of these methods, identify needed extensions to those methods, and indicate the opportunities for their support (e.g., via checklists, procedures and/or tool support).
  3. Develop the most promising support approaches identified by the first phase to make them suitable for application to NASA's spacecraft requirements.
Requirements Decomposition Analysis

Highlights

- Examined ST-6 Autonomous Sciencecraft Experiment requirements (approx. 9 pages)
  - Got a feel for the potential complexity of analyzing the decomposition of resource requirements, while working with a relatively small set of requirements (approximately 9 pages of technical detail)
- Focused on the Mars Reconnaissance Orbiter requirements (approx. 1,370 in all)
  - Developed a means to use the project-provided traceability information to extract all the requirements that are related to a requirement of interest
    - WHY: Convenience & comprehension – extracts just those requirements connected, directly or indirectly, assembling the results into a (web-browser-viewable) table. The result is easier to study than following individual links within the large set of requirements, and is more focused than the "graphic mode" that the requirements tool DOORS provides. In the event of the need to make a change to a requirement, this capability has potential utility, by finding and reporting all the requirements related (directly or indirectly) to that requirement.
    - HOW: This is in the form of an automatic script, which takes as input the user's identification of the requirement of interest, and returns the requirements linked to that (both "parents" of that requirement, and "children" of that requirement), the requirements linked to those requirements, etc.
  - Developed a means independent of the project traceability information to extract relevant requirements.
    - WHY: avoids reliance on the potentially incomplete or incorrect the traceability information within the existing documentation, thus giving a means in independently assure the correctness of requirements decomposition.
    - HOW: text-based search for keywords (e.g., search for the word "mass" and the string "kg" – for kilograms), and regular-expression textual searches for more refined patterns (e.g., a digit, then a space, then the letters "kg")
Trace- and string-based means to query a set of requirements have been developed. The result of such a query is a set of requirements. We have also developed capabilities to compare such result sets. Given two or more queries each of which yields a set of requirements, the capabilities developed allow the calculation of the intersection, difference and union among the several returned sets.

The simplest example, of two results sets, is shown below. The results are placed into one of three categories: occurs in only the results returned by the first query; occurs in only the results returned by the second query; occurs in both sets of results (see diagram below). More generally, for N queries, results are distributed among 2N – 1 categories.

Query 1: yields this set of requirements
Example: requirements that trace to a mass-related requirement
Result: mass-related requirements that do NOT involve the string Kg

Query 2: yields this set of requirements
Example: requirements that involve the string Kg
Result: requirements that do involve the string Kg but are not related to mass requirements

Result: requirements that do involve the string Kg AND are related to mass requirements

As before, for easy of viewing the results are presented in HTML tables that provide hyperlinks to the requirements themselves.

8 June, 2004  NASA Assurance Technology Symposium
Requirements Decomposition Analysis
Highlights (cont’d)

- Tools Developed
  - TraceRequirements.pl
    - Traces one or more requirements through a set of documents
    - For each specified requirement, parents and children are found, as well as siblings and all other possible relatives
  - FindPatterns.pl
    - A specified set of documents is searched for one or more patterns
    - Patterns can be specified as regular expressions
  - CompareResults.pl
    - Two or more trace and/or search results are compared to identify requirements common to the results
Requirements Decomposition Analysis
Tools Overview

- How TraceRequirements.pl differs from parent and child tracing:
Requirements Decomposition Analysis

Tools Overview (cont’d)

- How CompareResults.pl compares results produced by TraceRequirements.pl and/or FindPatterns.pl
  - Example: A, B, and C are results produced by TraceRequirements.pl and FindPatterns.pl

8 June, 2004

NASA Assurance Technology Symposium
Requirements Decomposition Analysis

Running the Tools – Preliminaries

- The tools operate on HTML files exported from DOORS documents. Before attempting to use the tools:
  - Export all of the requirements documents you’ll be working on from DOORS to HTML documents.
  - Make sure that you export them in tabular form and include the links.
- The tools are executed from the DOS command prompt.
- Before running the tools, change to the directory in which the exported DOORS files reside.
  - These aren’t quite production tools yet, and may not work properly if you don’t change directories.
Exporting DOORS modules as HTML files
Help Screen – TraceRequirements.pl

This script constructs a requirements lattice by identifying all of the parents and children of a given requirement, all of their parents and children, and so on until no more parents and children can be identified.

Usage: TraceRequirements.pl -i <Filename> [options]

where <Filename> is the name of file listing the requirements to be traced. Each row of <Filename> contains the pair <HTML_FILENAME>, <Requirement ID>. <HTML_FILENAME> and <Requirement ID> are separated by comma characters. <HTML_FILENAME> and <Requirement ID> must not contain comma characters. <HTML_FILENAME> can be a fully-qualified filename, or can refer to a file in the current directory.

NOTE: This script assumes that all other files to which the requirements in <HTML_FILENAME> are linked are in the same directory as <HTML_FILENAME>.

<HTML_FILENAME> identifies a DOORS document, containing the requirement <Requirement ID>, that has been exported to HTML. <Filename> may identify several requirements from the same document, but may not identify requirements from different documents. When exporting the DOORS document to HTML, all of the modules related to each requirement must be exported.

Output is written to an HTML file having the following format:

Column 1 - requirements file name - the name of the html file containing the requirement being traced.
Column 2 - requirement ID - the numerical ID of the requirement being traced.
Column 3 - the inbound links to the requirement - each link is a pair of the form <requirements file name, requirement ID>.
Column 4 - the outbound links from the requirement - the format is the same as that for inbound links.
Column 5 - requirement type - a requirement can be a root (and there can be more than one root in a trace lattice), a terminal, or neither.
The originally requested requirement for tracing (identified in <Filename>), is indicated by the value "Requirement being traced".
Column 6 - the text of the requirement.

Command line options are as follows:

-O <Output File Name> - specifies the name of the output file. The extension ".html" is appended to the end of specified name. An existing file with the same name will be overwritten.
-q - Selecting this option will suppress printing of the column containing the text of the requirement - see description of Column 6 above.
Input file identified by "-i" command line flag lists the requirements that are to be traced.

Example input file:

C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 273
C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 274
C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 275
C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 276
C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 277
C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 278
C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 279
C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 877
C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 878
C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML
Tables\MRO_Exhibit_I_-__Spacecraft_Requirements_D-20381.htm, 1127
Command line: TraceRequirements.pl -i PowerRequirementsFile.txt -o TestPowerTrace.html

8 June, 2004

NASA Assurance Technology Symposium
Help Screen – FindPatterns.pl

This script finds occurrences of specified patterns in HTML documents exported from DOORS and writes the search results to an HTML file that contains the text of each requirement matching any of the specified patterns.

Usage: FindPatterns.pl -d <Document List> -p <Search Pattern(s)> [options]

where <Document List> is a list of command line arguments identifying the HTML documents that are to be searched for patterns specified by the user. Each argument in <Document List> contains the fully qualified name of an HTML file exported from DOORS in table format. <Search Pattern(s)> is one or more regular expressions, separated by white space, for which a search will be made in the text of the documents identified by <Document List>.

Example: FindPattern.pl -p <Sample Text> -d Requirements1.html Req_Doc2.html

will search for digits and the text "Sample Text" in the documents "Requirements1.html" and "Req_Doc2.html".

Output is written as HTML having the following format:

Column 1 - requirements file name - the name of the HTML file matching one or more of the search patterns.
Column 2 - requirement ID - the numerical ID of the requirement containing one or more of the patterns.
Column 3 - the inbound links to the requirement - each link is a pair of the form <requirements file name>, requirement ID.
Column 4 - the outbound links from the requirement - the format is the same as that for inbound links.
Column 5 - identifies the pattern(s) matched in the requirements text, and how many times each pattern was matched.
Column 6 - the text of the requirement containing one or more of the search patterns.

Unless the name of an output file is specified, the output is written to the standard output, STDOUT.

Command line options are as follows:
-d <Document Name File> - specifies the name of a text file containing a list of the HTML documents that will be searched for patterns. Users have the option of specifying document file names in line with the -d option instead of or in addition to this option. This option, if no document names are specified, the user will be prompted to either specify the names in-line or specify the name of a text file containing the documents to be searched.

-p <Search Pattern File> - specifies the name of a text file containing a list of patterns for which the specified documents will be searched. Users have the option of specifying patterns in line with the -p option instead of or in addition to this option. However, if no patterns are specified, the user will be prompted to either specify the patterns in-line or provide the name of a text file containing the patterns for which to search the documents.

-o <Output File Name> - specifies the name of the output file. The extension "html" is appended to the end of specified name. An existing file with the same name will be overwritten with no warning.

-q - Selecting this option will suppress printing of the column containing the text of the requirement - see description of Column 6 above.
Input file identified by "-df" command line flag lists the documents that are to be searched for patterns. An example is shown below:

```
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables_MRO_Exhibit_I_-_Spacecraft_Requirements_D-20381.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables_MRO_DAV_V_Design_Principles_Rev_1_copy_to_MRO_02-26-02.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables_MRO_Exhibit_IV_-_Mission_Ops_Rqmts_D-20519.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables_MRO_L1_Level_1_Requirements_-_Appendix_to_Program_Plan_D-22204.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables_MRO_L2.5_Project_System_Requirements_D-22212.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables_MRO_L2_Mission_Assurance_Requirements_D-20327.htm
```

Input file identified by "-pf" command line flag identifies the patterns for which the documents are to be searched. An example is shown below:

```
(ls)+[t]ime
(ls)+[s]oon
(ls)+[l]ate
(ls)+[e]arly
(ls)+[e]arly
(ls)+[b]efore
(ls)+[a]fter
(ls)+[s]econd
(ld)+(ls)*[m]*[M]*[s]
(ls)+[m]icrose
(ls)+[h]our
(ls)+[m]inute
(ls)+[d]ay
(ls)+[m]onth
(ls)+[y]ear
```
Requirements Decomposition Analysis
Output Files from FindPatterns.pl

Command line: FindPatterns.pl --df DocumentNames_3.txt --p Kg kg --o KilogramSearchFrame.html

8 June, 2004

NASA Assurance Technology Symposium
Help Screen – CompareResults.pl

This script compares two or more files obtained with either of the scripts "TraceReqs.pl" or "FindPatterns.pl". The output is an HTML file identifying the requirements common to each of the possible subsets of the specified result files. For example, if 3 result files were specified, there would be 7 combinations of result files that would be analyzed (2^3-1). See the help screens for "TraceReqs.pl" and "FindPatterns.pl" for detailed information on their use.

Usage:

```
CompareResults.pl -i <ResultFile 1> <ResultFile 2> [<ResultFile 3> ...]
[-c|-e|-u] [-if <ResultFileNamesFile>] [-o <OutputFileName>] [-u]
```

where:

- `<ResultFile X>` is the name of an HTML file produced with the script "FindPatterns.pl" or "TraceReqs.pl".
- `<ResultFileNamesFile>` is an optional file containing the names of result files. The `-if` option can be used instead of or in addition to the `-i` option to avoid entering a large number of result file names on the command line.

There are three choices for specifying how the results are to be compared:

- `-c`: In each subset, look for a result, but do not check to make sure that the result is not contained in results files that are not part of the subset. For example, if there are four results files A, B, C, and D, and a search is being done to find results common to B and C, this option will not check for the nonexistence of the results in A and D.
  
  THIS IS THE DEFAULT CHOICE FOR COMPARING RESULTS.

- `-e`: In each subset, look for a result, and check to make sure that the result is not contained in results files that are not part of the subset. For example, if there are four results files A, B, C, and D, and a search is being done to find results common to B and C, this option will check for the nonexistence of the results in A and D.

- `-u`: Do not compare results, but report the union of all the result files.

- `<OutputFileName>` is an optional name for an HTML file that will contain the result of running this script. The extension "_.html" is added to the name of the file. If no filename is supplied, the results are written to the standard output.

- `-v` denotes the verbose option. If this option is selected, progress messages will be printed on the standard output if the `-o` option has also been selected. If this option has not been selected, or if the `-o` option has not been selected, no progress messages will be displayed.
Requirements Decomposition Analysis
Input Files to CompareResults.pl

- Input file identified by "-if" command line flag identifies the results that are to be compared. An example is shown below:

  TestMassTrace.html
  KilogramSearchFrame.html
  MassSearchFrame.html

- TestMassTrace is the output of the "TraceRequirements.pl" script for a set of requirements related to mass.
- KilogramSearchFrame.html is the output of the "FindPatterns.pl" script when searching all documents for either "Kg" or "kg".
- KilogramSearchFrame.html is the output of the "FindPatterns.pl" script when searching all documents for either "Mass" or "mass".

8 June, 2004
NASA Assurance Technology Symposium
Command line: CompareResults.pl -i ResultFileNames4.txt -o TestOutput13 -e

8 June, 2004

NASA Assurance Technology Symposium
Requirements Decomposition Analysis

Allen P. Nikora
Jet Propulsion Laboratory,
California Institute of Technology
Pasadena, CA
Allen.P.Nikora@jpl.nasa.gov

The work described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology. This work is sponsored by the National Aeronautics and Space Administration’s Office of Safety and Mission Assurance under the NASA Software Program led by the NASA Software IV&V Facility. This activity is managed locally at JPL through the Assurance Technology Program Office (ATPO).
Requirements Decomposition Analysis

Task Description

- **Problem Statement:** Requirements play a pivotal role in planning, selection, development, testing and operation of NASA's missions. Starting from mission objectives, requirements are successively decomposed. The correctness of this decomposition is critical, yet V&V of this crucial step is limited to manual inspection and pointwise testing, which are cumbersome and fallible (e.g., Mars Polar Lander).

- **Task:** Rigorous lightweight analysis methods for requirements decomposition have been developed by the software engineering research community, and have shown promise in successful application to critical systems (e.g., rail transportation). This proposal will study their application to the V&V of spacecraft software requirements, to ascertain if, when and how they are suitable for use by NASA.
Requirements Decomposition Analysis

Goals and Objectives

- **Goal:** study the applicability to NASA spacecraft requirements of rigorous analysis methods for requirements decomposition that have been developed by the software engineering research community.

- **Objectives:**
  1. Manually apply decomposition analysis methods applied to spacecraft requirements.
  2. Based on the results of these application studies, emerge with recommendations for the application of these methods, identify needed extensions to those methods, and indicate the opportunities for their support (e.g., via checklists, procedures and/or tool support).
  3. Develop the most promising support approaches identified by the first phase to make them suitable for application to NASA's spacecraft requirements.
Requirements Decomposition Analysis

Highlights

- Examined ST-6 Autonomous Sciencecraft Experiment requirements (approx. 9 pages)
  - Got a feel for the potential complexity of analyzing the decomposition of resource requirements, while working with a relatively small set of requirements (approximately 9 pages of technical detail)
- Focused on the Mars Reconnaissance Orbiter requirements (approx. 1,370 in all)
  - Developed a means to use the project-provided traceability information to extract all the requirements that are related to a requirement of interest
    - WHY: Convenience & comprehension – extracts just those requirements connected, directly or indirectly, assembling the results into a (web-browser-viewable) table. The result is easier to study than following individual links within the large set of requirements, and is more focused than the "graphic mode" that the requirements tool DOORS provides. In the event of the need to make a change to a requirement, this capability has potential utility, by finding and reporting all the requirements related (directly or indirectly) to that requirement.
    - HOW: This is in the form of an automatic script, which takes as input the user's identification of the requirement of interest, and returns the requirements linked to that (both "parents" of that requirement, and "children" of that requirement), the requirements linked to those requirements, etc.
  - Developed a means independent of the project traceability information to extract relevant requirements.
    - WHY: avoids reliance on the potentially incomplete or incorrect the traceability information within the existing documentation, thus giving a means in independently assure the correctness of requirements decomposition.
    - HOW: text-based search for keywords (e.g., search for the word "mass" and the string "kg" – for kilograms), and regular-expression textual searches for more refined patterns (e.g., a digit, then a space, then the letters "kg")
Requirements Decomposition Analysis

Highlights (cont’d)

- Trace- and string-based means to query a set of requirements have been developed. The result of such a query is a set of requirements. We have also developed capabilities to compare such result sets. Given two or more queries each of which yields a set of requirements, the capabilities developed allow the calculation of the intersection, difference and union among the several returned sets.

- The simplest example, of two results sets, is shown below. The results are placed into one of three categories: occurs in only the results returned by the first query; occurs in only the results returned by the second query; occurs in both sets of results (see diagram below). More generally, for N queries, results are distributed among \(2N - 1\) categories.

Query 1: yields this set of requirements

Example: requirements that trace to a mass-related requirement

Result: mass-related requirements that do NOT involve the string Kg

Query 2: yields this set of requirements

Example: requirements that involve the string Kg

Result: requirements that do involve the string Kg but are not related to mass requirements

Result: requirements that do involve the string Kg AND are related to mass requirements

As before, for easy of viewing the results are presented in HTML tables that provide hyperlinks to the requirements themselves.

8 June, 2004

NASA Assurance Technology Symposium
Requirements Decomposition Analysis

Highlights (cont’d)

- **Tools Developed**
  - **TraceRequirements.pl**
    - Traces one or more requirements through a set of documents
    - For each specified requirement, parents and children are found, as well as siblings and all other possible relatives
  - **FindPatterns.pl**
    - A specified set of documents is searched for one or more patterns
    - Patterns can be specified as regular expressions
  - **CompareResults.pl**
    - Two or more trace and/or search results are compared to identify requirements common to the results
Requirements Decomposition Analysis
Tools Overview

- How TraceRequirements.pl differs from parent and child tracing:
Requirements Decomposition Analysis
Tools Overview (cont’d)

- How CompareResults.pl compares results produced by TraceRequirements.pl and/or FindPatterns.pl
- Example: A, B, and C are results produced by TraceRequirements.pl and FindPatterns.pl

8 June, 2004
NASA Assurance Technology Symposium
Requirements Decomposition Analysis
Running the Tools – Preliminaries

- The tools operate on HTML files exported from DOORS documents. Before attempting to use the tools:
  - Export all of the requirements documents you’ll be working on from DOORS to HTML documents.
  - Make sure that you export them in tabular form and include the links.
- The tools are executed from the DOS command prompt.
- Before running the tools, change to the directory in which the exported DOORS files reside.
  - These aren’t quite production tools yet, and may not work properly if you don’t change directories.
### 3 Project Requirements

#### 3.1 Mission Lifetime

44 The orbiter shall be designed to operate within specification for 5.4 years after launch.

---

**Exporting DOORS modules as HTML files**

8 June, 2004

NASA Assurance Technology Symposium
Help Screen – TraceRequirements.pl

This script constructs a requirements lattice by identifying all of the parents and children of a given requirement, all of their parents and children, and so on until no more parents and children can be identified.

Usage: TraceRequirements.pl -i <Filename> [options]

where <Filename> is the name of file listing the requirements to be traced. Each row of <Filename> contains the pair <HTML_FILENAME>, <Requirement ID>. <HTML_FILENAME> and <Requirement ID> are separated by comma characters. <HTML_FILENAME> and <Requirement ID> must not contain comma characters. <HTML_FILENAME> can be a fully-qualified filename, or can refer to a file in the current directory.

NOTE: This script assumes that all other files to which the requirements in <HTML_FILENAME> are linked are in the same directory as <HTML_FILENAME>.

<HTML_FILENAME> identifies a DOORS document, containing the requirement <Requirement ID>, that has been exported to HTML. <Filename> may identify several requirements from the same document, but may not identify requirements from different documents. When exporting the DOORS document to HTML, all of the modules related to each requirement must be exported.

Output is written to an HTML file having the following format:

Column 1 - requirements file name - the name of the html file containing the requirement being traced.
Column 2 - requirement ID - the numerical ID of the requirement being traced.
Column 3 - the inbound links to the requirement - each link is a pair of the form <requirements file name, requirement ID>.
Column 4 - the outbound links from the requirement - the format is the same as that for inbound links.
Column 5 - requirement type - a requirement can be a root (and there can be more than one root in a trace lattice), a terminal, or neither.

The originally requested requirement for tracing <identified in <Filename>>, is indicated by the value "Requirement being traced".

Column 6 - the text of the requirement.

Command line options are as follows:

-o <Output File Name> - specifies the name of the output file. The extension ".html" is appended to the end of specified name. An existing file with the same name will be overwitten.

-q - selecting this option will suppress printing of the column containing the text of the requirement - see description of Column 6 above.
Requirements Decomposition Analysis

Input Files for TraceRequirements.pl

- Input file identified by "-i" command line flag lists the requirements that are to be traced.

- Example input file:

  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 273
  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 274
  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 275
  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 276
  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 277
  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 278
  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 279
  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 877
  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 878
  C:\Documents and Settings\anikora\Desktop\Complete MRO Requirements HTML Tables\_MRO_Exhibit_I__Spacecraft_Requirements_D-20381.htm, 1127
Requirements Decomposition Analysis

Output Files from TraceRequirements.pl

Command line: TraceRequirements.pl –i PowerRequirementsFile.txt –o TestPowerTrace.html

8 June, 2004

NASA Assurance Technology Symposium
Help Screen – FindPatterns.pl

This script finds occurrences of specified patterns in HTML documents exported from DOORS and writes the search results to an HTML file that contains the text of each requirement matching any of the specified patterns.

Usage: FindPatterns.pl -d <Document List> -p <Search Patterns> [optional]

where <Document List> is a list of command line arguments identifying the HTML documents that are to be searched for patterns specified by the user. Each argument in <Document List> contains the fully qualified name of an HTML file exported from DOORS in table format. <Search Patterns> is one or more regular expressions, separated by white space, for which a search will be made in the text of the documents identified by <Document List>.

Example: FindPatterns.pl -p "\d SampleText -d Requirements_1.html Req.Doc_2.htm" will search for digits and the text "Sample text" in the documents "Requirements_1.html" and "Req.Doc_2.htm".

Output is written as HTML having the following format:

Column 1 - requirements file name - the name of the HTML file matching one or more of the search patterns.
Column 2 - requirement ID - the numerical ID of the requirement containing one or more of the patterns.
Column 3 - the inbound links to the requirement - each link is a pair of the form <requirements file name, requirement ID>.
Column 4 - the outbound links from the requirement - the format is the same as that for inbound links.
Column 5 - identifies the pattern(s) matched in the requirements text, and how many times each pattern was matched.
Column 6 - the text of the requirement containing one or more of the search patterns.

Unless the name of an output file is specified, the output is written to the standard output, STDOUT.

Command line options are as follows:

-df <Document Name File> - specifies the name of a text file containing a list of the HTML documents that will be searched for patterns. Users have the option of specifying document file names inline with the -d option instead of, or in addition to this option. This option is used in place of specifying document file names on the command line. If no document names are specified, the user will be prompted to either specify the names in-line or specify the name of a text file containing the documents to be searched.

-pf <Search Pattern File> - specifies the name of a text file containing a list of patterns for which the specified documents will be searched. Users have the option of specifying patterns inline with the -p option instead of, or in addition to this option. However, if no patterns are specified, the user will be prompted to either specify the patterns in-line or provide the name of a text file containing the patterns for which to search the documents.

-o <Output File Name> - specifies the name of the output file. The extension "html" is appended to the end of specified name. If existing file with the same name will be overwritten with no warning.

-q - Selecting this option will suppress printing of the column containing the text of the requirement - see description of Column 6 above.
Input file identified by "-df" command line flag lists the documents that are to be searched for patterns. An example is shown below:

C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables\MRO_Exhibit_1._Spacecraft_Requirements_D-20381.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables\MRO_DAV_V_Design_Principles____Rev_1_copy_to_MRO_02-26-02.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables\MRO_Exhibit_IV._Mission_Ops_Rqmts_D-20519.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables\MRO_L1_Level_1_Requirements__Appendix_to_Program_Plan_D-22204.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables\MRO_L2_5_Project_System_Requirements_D-22212.htm
C:\Documents and Settings\anikoral\Desktop\Complete MRO Requirements HTML Tables\MRO_L2_Mission_Assurance_Requirements_D-20327.htm

Input file identified by "-pf" command line flag identifies the patterns for which the documents are to be searched. An example is shown below:

(ls)+[t]ime
(ls)+[s]oon
(ls)+[l]ate
(ls)+[e]arly
(ls)+[e]arlie
(ls)+[b]efore
(ls)+[a]fter
(ls)+[s]econd
(id)+(ls)*[m]*[s]crose
(ls)+[h]our
(ls)+[m]inute
(ls)+[d]ay
(ls)+[m]onth
(ls)+[y]ear
Requirements Decomposition Analysis

Output Files from FindPatterns.pl


8 June, 2004

NASA Assurance Technology Symposium
Help Screen – CompareResults.pl

This script compares two or more files obtained with either of the scripts "TraceReqs.pl" or "FindPatterns.pl". The output is an html file identifying the requirements common to each of the possible subsets of the specified result files. For example, if 3 result files were specified, there would be 7 combinations of result files that would be analyzed (2^3-1). See the help screens for "TraceReqs.pl" and "FindPatterns.pl" for detailed information on their use.

Usage:

```
CompareResults.pl [-o <OutputFileName>][-u] [-c][-e][-u] [-if <ResultFileNamesFile>] [-i <ResultFile 1> <ResultFile 2> [<ResultFile 3> ...]]
```

where:

- `<ResultFile X>` is the name of an html file produced with the script "FindPatterns.pl" or "TraceReqs.pl".
- `<ResultFileNamesFile>` is an optional file containing the names of result files. The -if option can be used instead of or in addition to the -i option to avoid entering a large number of result file names on the command line.

There are three choices for specifying how the results are to be compared:

- `-c`: In each subset, look for a result, but do not check to make sure that the result is not contained in results files that are not part of the subset. For example, if there are four results files A, B, C, and D, and a search is being done to find results common to B and C, this option will not check for the nonexistence of the results in A and D. **THIS IS THE DEFAULT CHOICE FOR COMPARING RESULTS.**

- `-e`: In each subset, look for a result, and check to make sure that the result is not contained in results files that are not part of the subset. For example, if there are four results files A, B, C, and D, and a search is being done to find results common to B and C, this option will check for the nonexistence of the results in A and D.

- `-u`: Do not compare results, but report the union of all the result files.

- `<OutputFileName>` is an optional name for an html file that will contain the result of running this script. The extension ".html" is added to the name of the file. If no filename is supplied, the results are written to the standard output.

- `-u` denotes the verbose option. If this option is selected, progress messages will be printed on the standard output if the -o option has also been selected. If this option has not been selected, or if the -o option has not been selected, no progress messages will be displayed.
Requirements Decomposition Analysis
Input Files to CompareResults.pl

- Input file identified by "-if" command line flag identifies the results that are to be compared. An example is shown below:

  TestMassTrace.html
  KilogramSearchFrame.html
  MassSearchFrame.html

- TestMassTrace is the output of the "TraceRequirements.pl" script for a set of requirements related to mass.
- KilogramSearchFrame.html is the output of the "FindPatterns.pl" script when searching all documents for either "Kg" or "kg".
- KilogramSearchFrame.html is the output of the "FindPatterns.pl" script when searching all documents for either "Mass" or "mass".
### Result Files Compared

#### Subset 1.1.1

**Included In:**
- 1. Test/Macro/Frame.html
- 2. Filename/Macro/Frame.html
- 3. Macro/Frame.html

**Not Included In:**
- 1. Test/Macro/Frame.html
- 2. Filename/Macro/Frame.html
- 3. Macro/Frame.html

#### Substructures

**Incl.**
- MRG 'Ecliptic' - Sensor (Amendment D-20311) 100

**Excl.**
- MRG 'Ecliptic' - Sensor (Amendment D-20311) 100

---

### Output Files from CompareResults.pl

**Command line:**
`CompareResults.pl -if ResultFilenames4.txt -o TestOutput13 -e`

---

**Date:** 8 June, 2004

**Conference:** NASA Assurance Technology Symposium