Requirement Management Tool rmtoo

Introduction

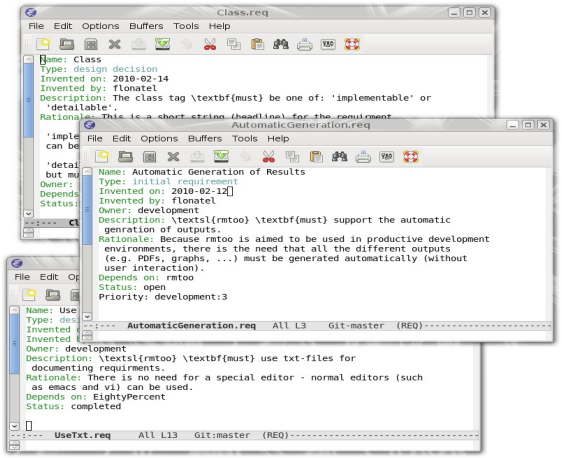
by flonatel GmbH & Co. KG

Content

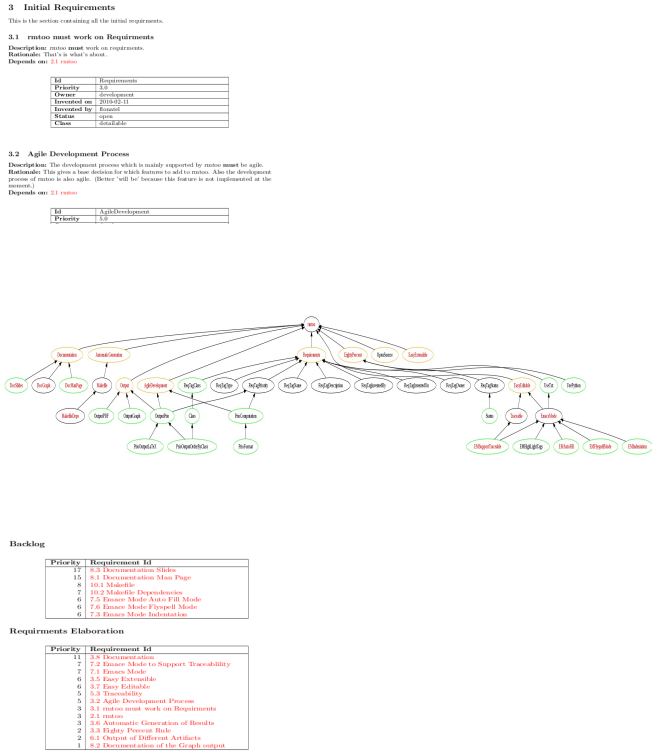
© 2010 by flonatel

1.Introduction

Overview



rmtoo



Input Data Command line tool Output Artifacts

© 2010 by flonatel

Basic Facts (1/2)

- *rmtoo* is a minimalistic non-interactive requirements management tool
- *rmtoo* works on data stored in the file system (plain text files)
- *rmtoo* is a command line tool which reads in files and creates output
- *rmtoo* supports different output formats and artifacs

© 2010 by flonatel

Basic Facts (2/2)

- *rmtoo* data files can be handled by standard *nix commands (emacs, vi, grep, awk, streplace, sed, ...)
- *rmtoo* runs (mostly) on the same hardware and operating system where the development takes place – no need for a dedicated machine
- *rmtoo* baselineing, backup and restore can be done by a revision control system

© 2010 by flonatel

rmtoo is not

- *rmtoo* has no GUI
- *rmtoo* comes with no database
- *rmtoo* has no integrated editor
- *rmtoo* does not provide an UML editor
- *rmtoo* does not provide any import possibility (e. g. from a spreadsheet or a word processing document)

© 2010 by flonatel

License

- *rmtoo* is Open Source
- *rmtoo* is free
- *rmtoo* is licensed under GPLv3
- Commercial support is available

© 2010 by flonatel

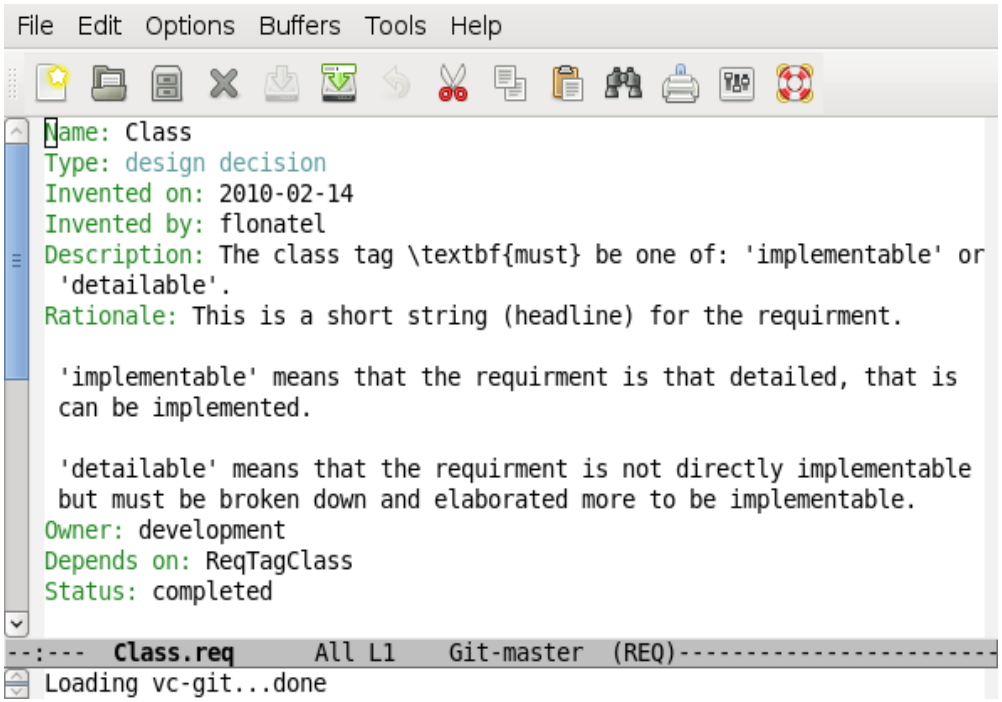
2.Input Data Files

Data Files

- Input files are standard plain text files
- Each requirement is basically a list of key-value pairs
- Most used keys for requirement management are supported
- Files can be handled by most *nix commands (sed, streplace, awk, grep, ...)
- Revision control can be done by revision system (git, mercurial, subversion, ...)

© 2010 by flonatel

Data File Example



- Simple *key: value* notation
- Space in first column: extend value
- Keys are fixed (predefined)
- Values are checked when possible
- Editable with standard text editor

© 2010 by flonatel

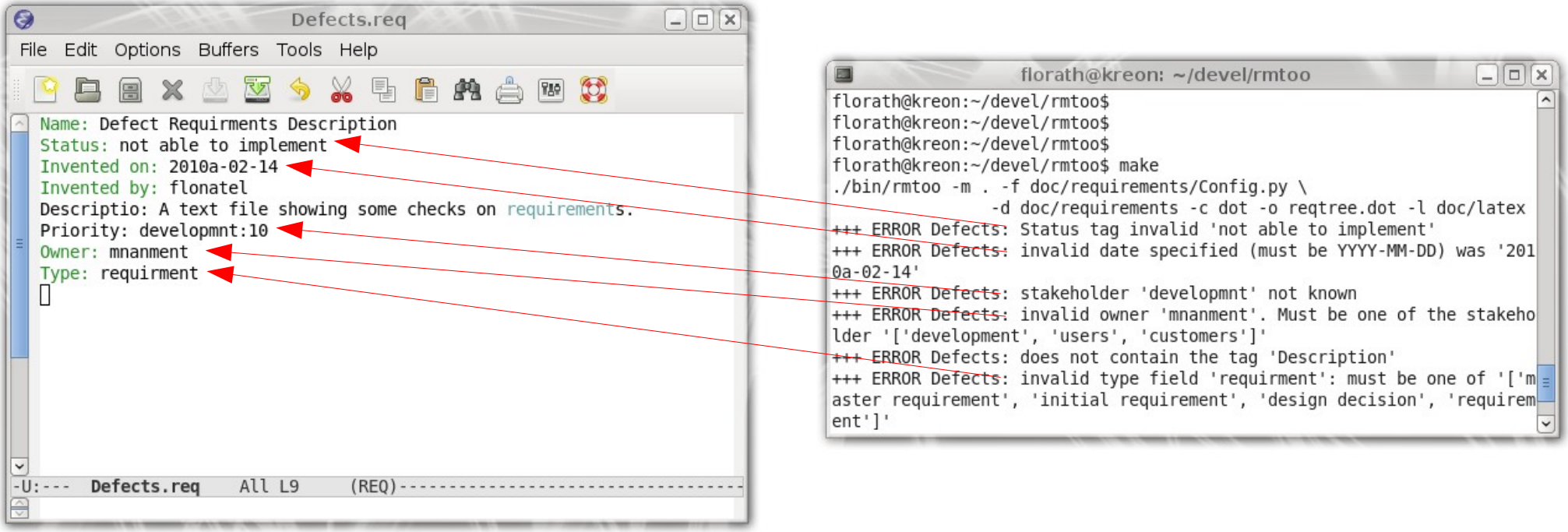
3. Checks on Requirements

Checks on Requirements

- Because there is no built-in editor, consistency checks must be done
- Checks include:
  - Syntax / Format checks e.g. for date fields
  - Type checks: some fields are allowed to contain only a limited set of (key-)words
  - Typo checks for e.g. stakeholders
  - Dependency checks

© 2010 by flonatel

Example: Checks



© 2010 by flonatel

4. Output Artifacts

Requirements Document – Requirements Dependency Graph – Project Backlog – Project Elaboration List

Output: Requirements Document

- *rmtoo* can create a requirements document containing all requirements
- Output intermediate format of requirements is LaTeX using hyperref
- Resulting documents can be e.g. PDF and HTML
- Links in table of contents and dependencies available in PDF and HTML
- Arbitrary text can be added

© 2010 by flonatel

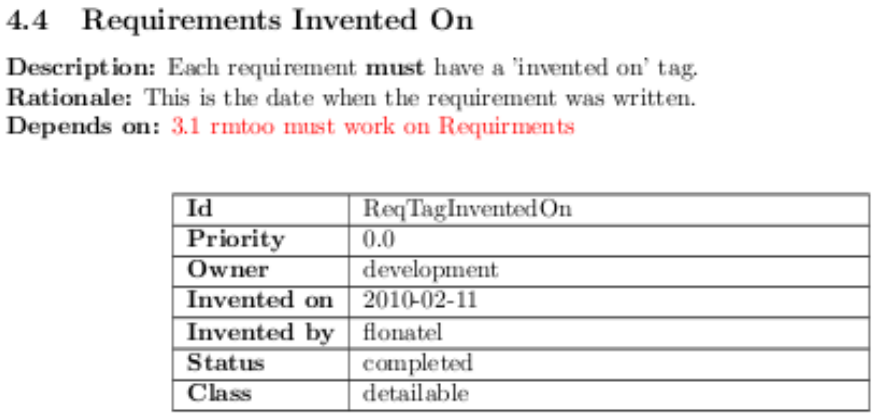
Output: Table of Contents

Contents

- Each requirement fits in it's own subsection
- Hyperlinks for fast navigation

© 2010 by flonatel

Output: Requirement



4.4 Requirements Invented On

Description: Each requirement **must** have a 'invented on' tag.
Rationale: This is the date when the requirement was written.
Depends on: 3.1 rmtoo must work on Requirments

Id	ReqTagInventedOn
Priority	0.0
Owner	development
Invented on	2010-02-11
Invented by	flonatel
Status	completed
Class	detailable

- Each requirement fits in it's own subsection
- All key-values are available
- Hyperlinks to dependencies for fast navigation

© 2010 by flonatel

4.Output Artifacts

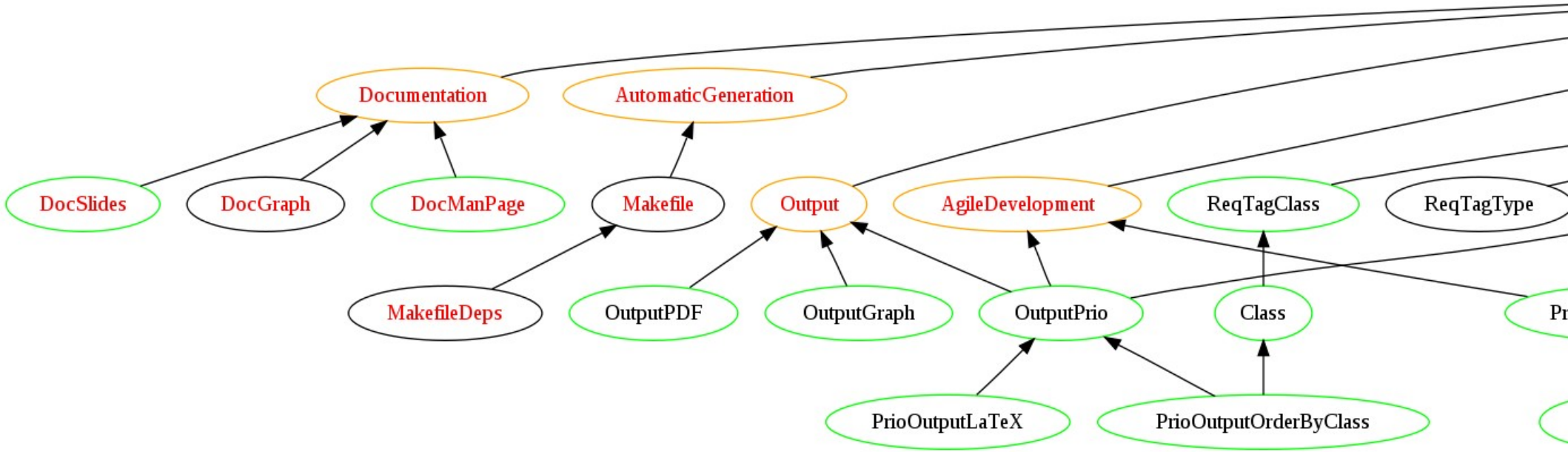
Requirements Document – Requirements Dependency Graph – Project Backlog – Project Elaboration List

Output: Dependency Graph

- *rmtoo* can create a requirement dependency graph
- Simple to visualize dependencies of requirements
- Colorized status information. Example: red font means open, black font completed

© 2010 by flonatel

Output: Dependency Graph Example (Part)



© 2010 by flonatel

4.Output Artifacts

Requirements Document – Requirements Dependency Graph – Project Backlog – Project Elaboration List

Output: Project Backlog

- *rmtoo* can create the project backlog as used in SCRUM
- Project backlog contains all elaborated requirements (which means they can be implemented)
- Project backlog is the ToDo list for the developers
- Requirements are sorted by priority

© 2010 by flonatel

Output: Project Backlog Example

Backlog

Priority	Requirement Id
17	8.3 Documentation Slides
15	8.1 Documentation Man Page
8	10.1 Makefile
7	10.2 Makefile Dependencies
6	7.5 Emace Mode Auto Fill Mode
6	7.6 Emace Mode Flyspell Mode
6	7.3 Emacs Mode Indentation

- Prioritized list of requirements
- Hyperlinks for fast navigation
- Embedded in the PDF / HTML document

© 2010 by flonatel

4.Output Artifacts

Requirements Document – Requirements Dependency Graph – Project Backlog – Project Elaboration List

Output: Project Elaboration List

- *rmtoo* can create a list of all requirements that must be further elaborated
- Elaboration List is the ToDo list for the SCRUM master
- Requirements are sorted by priority

© 2010 by flonatel

Output: Project Elaboration List Example

Requirments Elaboration

Priority	Requirement Id
11	3.8 Documentation
7	7.2 Emace Mode to Support Traceablility
7	7.1 Emacs Mode
6	3.5 Easy Extensible
6	3.7 Easy Editable
5	5.3 Traceability
5	3.2 Agile Development Process
3	3.1 rmtoo must work on Requirments
3	2.1 rmtoo
3	3.6 Automatic Generation of Results
2	3.3 Eighty Percent Rule
2	6.1 Output of Different Artifacts
1	8.2 Documentation of the Graph output

- Prioritized list of requirements
- Hyperlinks for fast navigation
- Embedded in the PDF / HTML document

© 2010 by flonatel

5. Future

Future / Plans

- Some features are missing but planned for the next releases
  - Traceability
  - Better support writing requirements in Emacs mode
- Community, User and Customer driven

© 2010 by flonatel

Thank you!

© 2010 by flonatel

Copyright

This document is distributed under the creative commons license 'Attribution-Noncommercial-No Derivative Works 3.0 Germany'

© 2010 flonatel GmbH & Co. KG