



# Requirement Management Tool rmtoo

## Introduction

by flonatel GmbH & Co. KG

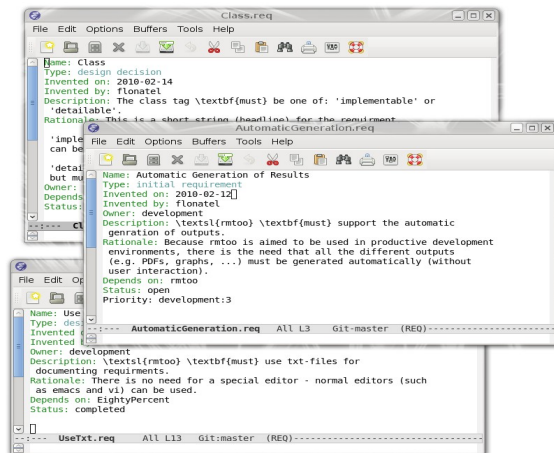
# Content

---

- 1.Introduction
- 2.Input Data Files
- 3.Checks on Requirements
- 4.Output Artifacts
- 5.Future

# 1.Introduction

# Overview



rmtoo



## 3 Initial Requirements

This is the section containing all the initial requirements.

### 3.1 rmtoo must work on Requirements

Description: rmtoo must work on requirements.

Rationale: That's what's about.

Depends on: 2.1 rmtoo

Id	Requirement
1	Requirements
2	Requirements
3	Requirements
4	Requirements
5	Requirements
6	Requirements
7	Requirements
8	Requirements
9	Requirements
10	Requirements

### 3.2 Agile Development Process

Description: The development process which is mainly supported by rmtoo must be agile.

Rationale: This gives a hint for which features to add to rmtoo. Also the development process of rmtoo is also agile. (Rmtoo will be because this feature is not implemented in the process.)

Depends on: 2.1 rmtoo

Id	Requirement
1	Agile Development
2	Agile Development



## Backlog

Priority	Requirement Id
11	2.1 Documentation after Release
10	2.1 Documentation after Release
9	10.1 Makefile
8	10.1 Makefile
7	10.1 Makefile
6	7.5 Feature: Make Auto Fill Mode
5	7.5 Feature: Make Auto Fill Mode
4	7.5 Feature: Make Auto Fill Mode
3	7.5 Feature: Make Auto Fill Mode
2	7.5 Feature: Make Auto Fill Mode
1	7.5 Feature: Make Auto Fill Mode

## Requirements Elaboration

Priority	Requirement Id
11	2.1 Documentation after Release
10	2.1 Documentation after Release
9	2.1 Documentation after Release
8	2.1 Documentation after Release
7	2.1 Documentation after Release
6	2.1 Documentation after Release
5	2.1 Documentation after Release
4	2.1 Documentation after Release
3	2.1 Documentation after Release
2	2.1 Documentation after Release
1	2.1 Documentation after Release

Input Data

Command line tool

Output Artifacts

# 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 the files and creates output
- *rmtoo* supports different output formats and artifacts

# 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 be a revision control system

# 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)

# License

---

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

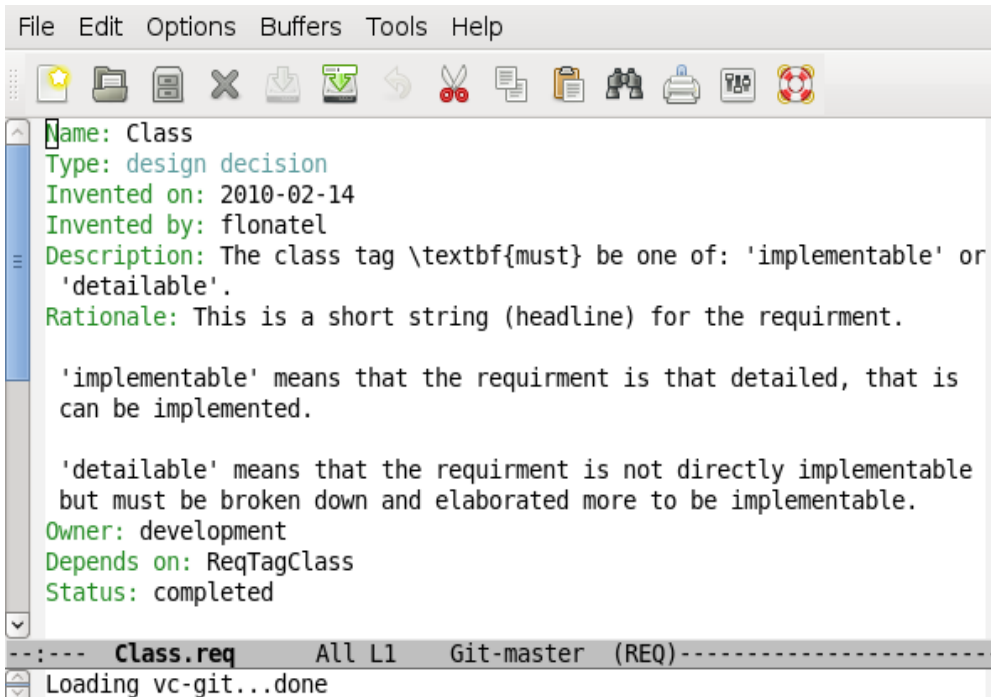


## 2.Input Data Files

# 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, ...)

# Data File Example



```
File Edit Options Buffers Tools Help
[Icons]
Name: Class
Type: design decision
Invented on: 2010-02-14
Invented by: flonatel
Description: The class tag \textbf{must} be one of: 'implementable' or
'detailable'.
Rationale: This is a short string (headline) for the requirement.

'implementable' means that the requirement is that detailed, that is
can be implemented.

'detailable' means that the requirement is not directly implementable
but must be broken down and elaborated more to be implementable.
Owner: development
Depends on: ReqTagClass
Status: completed
--:--- Class.req All L1 Git-master (REQ)-----
Loading vc-git...done
```

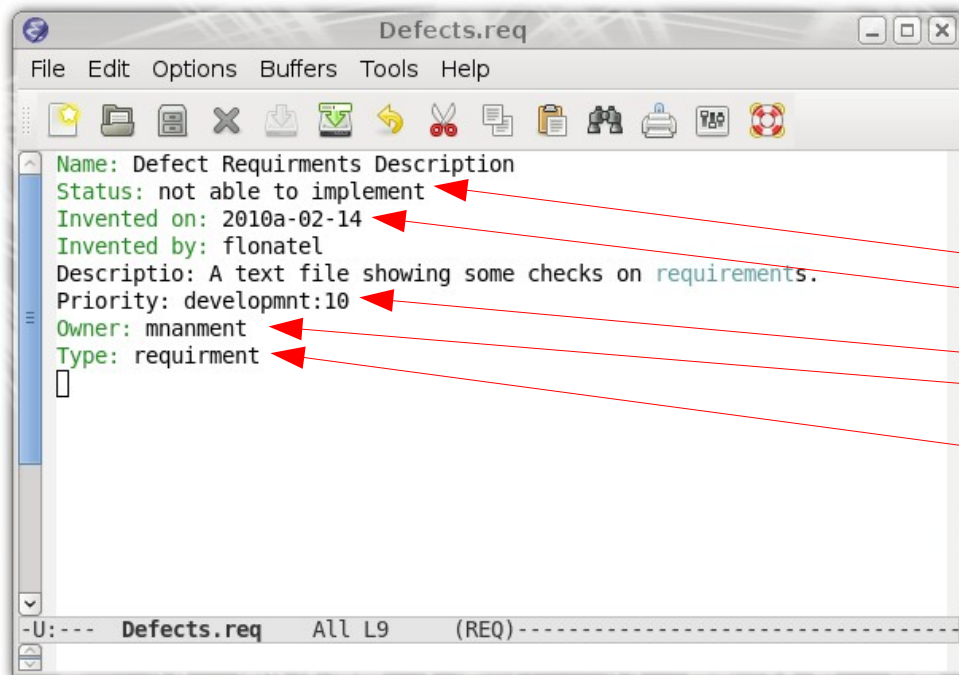
- Simple *key: value* notation
- Space in first column: extend value
- Keys are fixed (predefined)
- Values are checked when possible
- Editable with standard text editor

### 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

# Example: Checks



## 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
- Arbitrarily text can be added



# Output: Table of Contents

## Contents

<b>1 Status</b>	<b>5</b>
1.1 Backlog	5
1.2 Requirments Elaboration	5
<b>2 What's all about</b>	<b>5</b>
2.1 rntoo	5
<b>3 Initial Requirements</b>	<b>6</b>
3.1 rntoo must work on Requirments	6
3.2 Agile Development Process	6
3.3 Eighty Percent Rule	6
3.4 Open Source rntoo	7
3.5 Easy Extensible	7
3.6 Automatic Generation of Results	8
3.7 Easy Editable	8
3.8 Documentation	8
<b>4 Requirements Tags</b>	<b>9</b>
4.1 Requirements Name	9
4.2 Requirements Type	9
4.3 Requirements Invented By	9
4.4 Requirements Invented On	10
4.5 Requirements Description	10
4.6 Requirements Owner	10
4.7 Requirements Status	11
4.8 Status	11
4.9 Requirement Priority	12
4.10 Priority Format	12
4.11 Requirments Class	12
4.12 Class	13
<b>5 Implementation Decisions</b>	<b>13</b>
5.1 Use Txt	13
5.2 Use Python	13
5.3 Traceability	14

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

# 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 rntoo must work on Requirments](#)

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

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

## 4. Output Artifacts

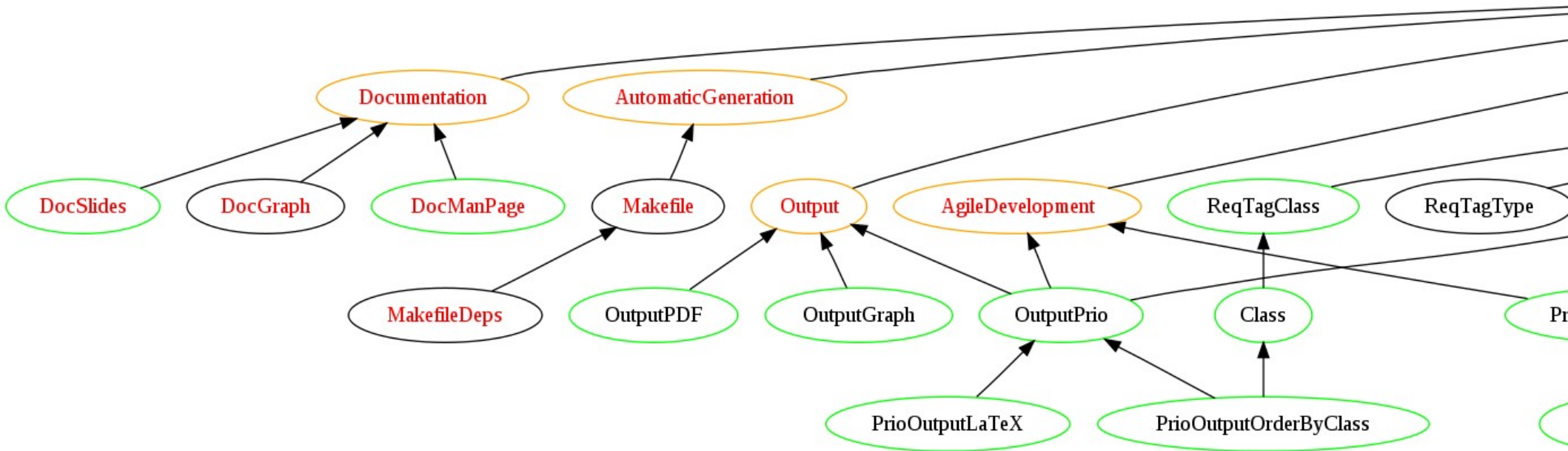
Requirements Document – Requirements  
Dependency Graph – Project Backlog – Project  
Elaboration List

# Output: Dependency Graph

---

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

# Output: Dependency Graph Example (Part)



## 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 are all elaborated requirements (which means they can be implemented)
- Project backlog is the ToDo list for the developers
- Requirements are sorted by priority

# Output: Project Backlog Example

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

Backlog

Priority	Requirement Id
17	<a href="#">8.3 Documentation Slides</a>
15	<a href="#">8.1 Documentation Man Page</a>
8	<a href="#">10.1 Makefile</a>
7	<a href="#">10.2 Makefile Dependencies</a>
6	<a href="#">7.5 Emace Mode Auto Fill Mode</a>
6	<a href="#">7.6 Emace Mode Flyspell Mode</a>
6	<a href="#">7.3 Emacs Mode Indentation</a>



## 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

# Output: Project Elaboration List Example

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

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 rntoo must work on Requirments
3	2.1 rntoo
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

## 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

# Thank you!



# Copyright

---

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

© 2010 flonatel GmbH & Co. KG