

LiteRM Illustrated

Copyright © 2011 by ClearSpecs Enterprises.

LiteRM[®], the game changer, is a light-weight requirements manager that

- stores a rich assortment of requirements information, including states and links
- provides the core capabilities found in heavy-weight RMs
- is easy to learn, tailor, use, and change

The following figures illustrate aspects of LiteRM[®]:

Figure 1. A Framework for Requirements Information

Figure 2. Grove of Assorted Fruit Trees

Figure 3. Top of the Requirements Tree

Figure 4. Types of Quality Requirements

Figure 5. Facet Templates

Figure 6. Template for a Function Requirement

Figure 7. Profiler Report

Figure 8. Partial Indexer Report

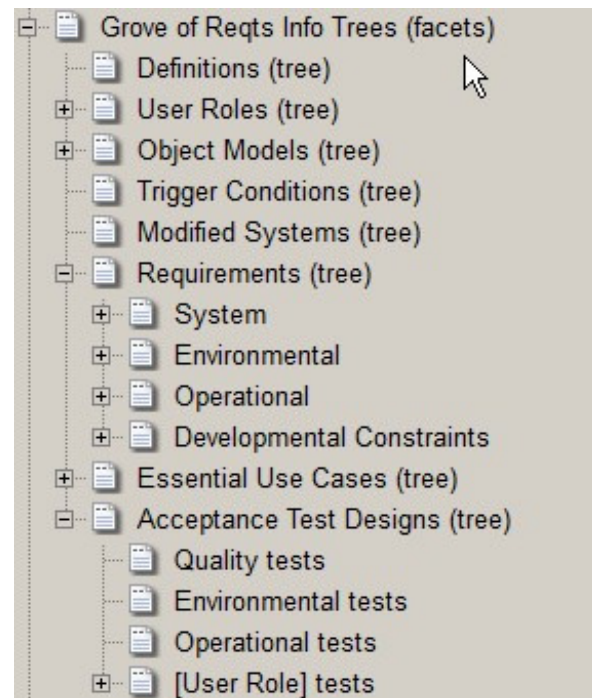
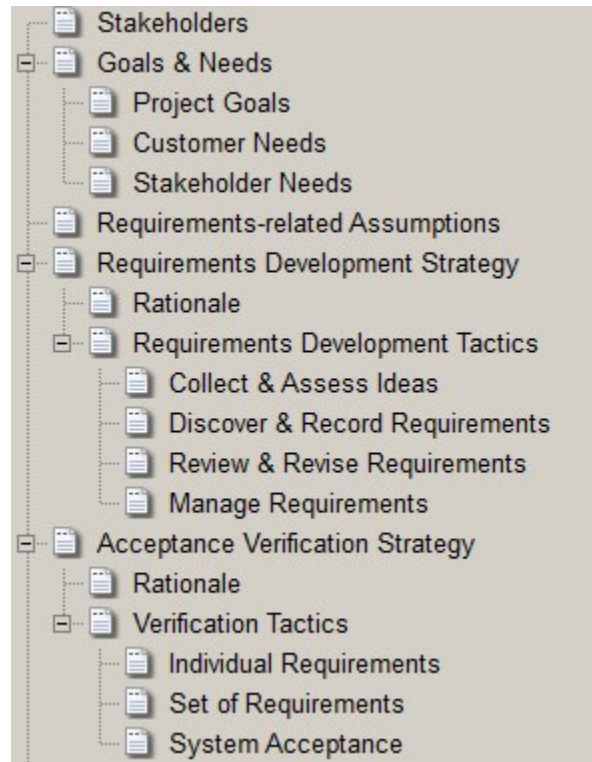
Figure 9. Sample Requirements Information List

Figure 10. Sample questions that LiteRM can easily answer

Figure 11. LiteRM Architecture Diagram

Figure 12. RM Feature Comparison Table

Figure 1. A Framework for Requirements Information



Imagine the types of requirements information (facets) in this framework, along with the requirements, as a grove of assorted fruit trees.

Figure 2. Grove of Assorted Fruit Trees



Further, imagine that a fruit in one tree e.g. a use case, is linked to (a subtree or) fruit e.g. acceptance test, in another tree. This framework organizes the individual facet types and links individual facets to associated facets of other types (i.e. in other trees).

Figure 3. Requirements Framework

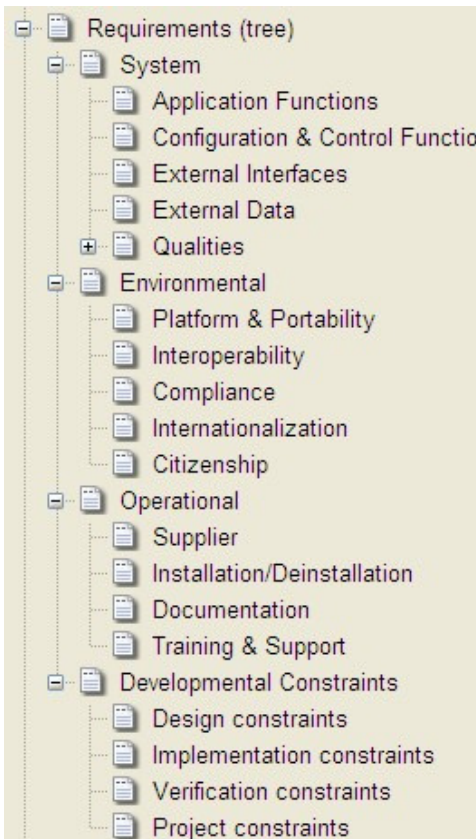


Figure 4. Quality Requirements

Qualities
Accessibility
Appeal
Availability
Reliability
Stability & Robustness
Recovery & Backup
Compliance
Ease
Ease of installation
Ease of understanding
Ease of tailoring
Ease of use
Ease of operation
Flexibility
Interoperability
Portability
Scalability
Sustainability
Performance
Capacity
Efficiency
Response Time
Throughput
Privacy
Safety
Security
Transparency
Auditability
Verifiability

Figure 5. Facet Templates

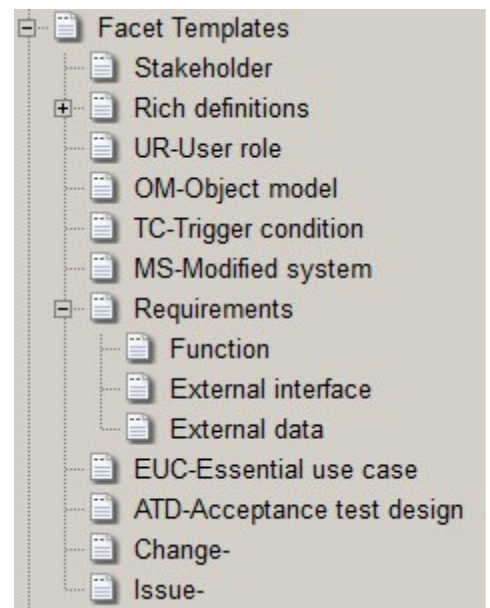


Figure 6. Template for a Function Requirement

Details

Basic statement templates

Unconditional i.e., always necessary
<system id> shall <system response(s)>

Event-driven (keyword *When*)
When *<trigger event(s)>*, *<system id> shall <system response(s)>*

State-driven (keyword *While*)
While *<in state(s)>*, *<system id> shall <system response(s)>*

Configuration-driven (keyword *Where*)
Where *<configuration(s)>*, *<system id> shall <system response(s)>*

Compound (at least 2 keywords) statement template

[Where < configuration(s)> and] [[While] <in state(s)> [and]] [[When] <trigger event(s)>],
<system id> shall <system response(s) - visible at the system boundary>

e.g., Where OnStar is installed and connected to the network and a crash is detected,
OnStar shall transmit crash site coordinates and inform occupants that help is on the way

Derived from “Easy Approach to Requirements Syntax (EARS)”, Alistair Mavin

Limits & Constraints (on both functionality and usage)

Assumptions/Rationale

Implications/Expectations

Risk Factors

- a. Developer Understanding states = < Sup, Lim, Deep >
- b. Likelihood of serious defects = [high, medium, low, NR]
- c. Threat level = [high, medium, low] -- (threat of valid reqts to design, schedule, or budget)

Other Attributes

- a. Sources/Parents:
- b. Value (to org) = [essential, necessary, desirable]
- c. Cost (to impl) = [high, medium, low]
- d. Priority (to impl) = [high, medium, low]
- e. Version =
- f. Iteration:
- g. Allocation:

STATES

- a. Requirement states are < @Unverified, Verified, Impl, Inactive >

ASSOCIATIONS

- a. Definitions: &
- b. Object Models: &OM-
- c. Trigger Conditions: &TC-
- d. Modified Systems: &MS-
- e. Depends-on Requirements: &

Notes

Figure 7. Profiler Report



Profile of file.txt on 2011-09-29

Requirement Types

Req't Type	Essentials	Necessarys	Desirables	Unverifieds	Verifieds	Implementeds
Functionals	15	3	1	19	0	0
External Interfaces	0	0	0	0	0	0
External Data	0	0	0	0	0	0
Qualities	9	3	0	12	0	0
Environmentals	0	0	0	0	0	0
Operational	0	1	0	1	0	0
Constraints	0	0	0	0	0	0
Totals	24	7	1	32	0	0

Risk Factors

Risk Type	Essentials	Necessarys	Desirables	Supples
Superficial Dev Know	5	2	1	0
Limited Dev Know	17	4	0	6
Seriously Defective Reqts	0	0	0	
High Threat Level Reqts	0	0	0	
Reqts Issues	0	0	0	
Reqts Conflicts	0			

Supplemental Facets

Definitions	User Roles	Object Models	Trigger Conditions	Modified Systems	Essential Use Cases	Acceptance Test Designs
6	5	6	2	0	1	1

Figure 8. Partial Indexer Report

Link Index for KI Website Reqts Info V7.txt on 2011-07-23

Facet: OM-Deal

Link to: OM-Vendor

Link to: OM-Purchase

Link to: UR-Vendor

Link from: Deal conditions

Link from: EUC11-Customer buys a deal

Link from: F11 Auto Start/End Deal

Link from: F21 Deliver certificates & receipts

Link from: F22 Displays for Customer

Link from: F23 Displays for Vendor

Link from: F25 Email vendor, charity, & customer

Link from: F28 Reports for Vendor

Link from: F34 CRUD Deal

Link from: F93 Stop/Restart Deal

Link from: OM-Purchase

Link from: OM-Vendor

Link from: TC-Start/End Deal Time

Attributes with Link Values

Allocation: Deal object

Link from: F34 CRUD Deal

Assignment: Bob

Link from: Ease of Learning & Use

Assignment: Jane

Link from: Capacity

Link from: Efficiency

Link from: Privacy

Link from: Response Time

Link from: Security

Link from: Throughput

Sources/Parents: Bob

Link from: F23 Displays for Vendor

Link from: F28 Reports for Vendor

Figure 9. Sample Requirements Information List

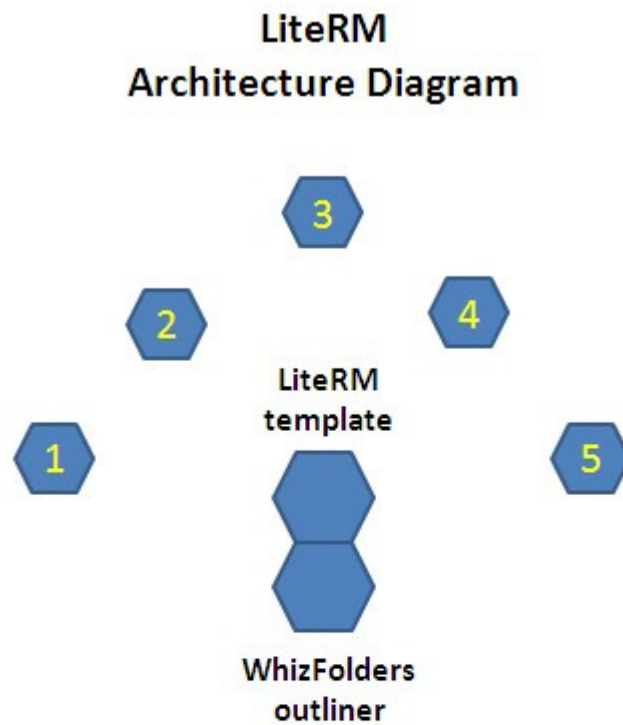
- Definitions (tree)
 - Deal conditions
 - deal-has-ended
 - deal-has-expired
 - deal-has-soldout
 - deal-has-started
 - deal-is-active
 - request
 - Request conditions
- User Roles (tree)
 - UR-Admin
 - UR-Charity
 - UR-Customer
 - UR-Guest
 - UR-Vendor
- Object Models (tree)
 - Person
 - OM-Charity
 - OM-Customer
 - OM-Vendor
 - Place
 - Thing
 - OM-Deal
 - Event
 - OM-Donation
 - OM-Purchase
- Trigger Conditions (tree)
 - TC-Start/End Deal Time
 - TC-Start/End Request Time
- Requirements (tree)
 - System
 - Application Functions
 - F1 Auto Start/End
 - F11 Auto Start/End Deal
 - F12 Auto Start/End Request
 - F2 Communication
 - F21 Deliver certificates & receipts
 - F22 Displays for Customer
 - F23 Displays for Vendor
 - F24 Displays for Admin
 - F25 Email vendor, charity, & customer
 - F26 Reports for Admin
 - F27 Reports for Charity
 - F28 Reports for Vendor
 - F3 CRUD for Object Models
 - F31 CRUD Charity
 - F32 CRUD Customer
 - F33 CRUD Vendor
 - F34 CRUD Deal
 - F35 CRUD Donation
 - F36 CRUD Purchase

The searching and reporting capabilities of LiteRM enable you to answer the following questions.

Figure 10. Sample questions that LiteRM can easily answer

- What are the details of each requirement?
- Which requirements are essential?
- Which requirements were provided by a specific stakeholder?
- Which requirements are associated with a specific user role?
- Which requirements are derived from a specific requirement?
- Which requirements are dangerous and why?
- Which requirements are poorly understood by the developers?
- Which requirements have been implemented?
- Which requirements and acceptance tests are associated with this use case?
- How many high-priority requirements' issues are open?

Figure 11. LiteRM Architecture Diagram



1. Profiler reporting module
2. Indexer reporting module
3. User Guides for WhizFolders, LiteRM, & Reporting modules
4. Search Term library
5. Paste Selections utility file

Figure 12. RM Feature Comparison Table

RM Feature Comparison Table

	"Raw" Platforms			RM systems	
	Word Processor	Spreadsheet Processor	Outliner / Personal Info Mgr	LiteRM	Medium – Heavyweight RM
Reusable platform	✓	✓	✓	✓	
Easy to learn	✓	✓	✓	✓	
Easy to use	✓	✓	✓	✓	
Outlines supported	✓		✓	✓	✓
Boolean search			✓	✓	✓
(Outline) entry selection		✓	✓	✓	✓
Core RM capabilities				✓	✓
Precise definitions				✓	
Comprehensive risk assessment with tracking of developer understanding				✓	
Diagram generation					✓
Additional ALM functions (test generation & mgmt)					✓
Change control					✓