Abstract

This module addresses supporting processes, for instance documentation, templates, and reviewing.
Granularity of Documentation

by Gerrit Muller University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

The design of documentation is discussed, with emphasis on the requirements, the need for decomposition, the measures needed to maintain overview and criteria for granularity.
Requirements for the Entire Documentation Structure

- Accessibility for the readers
- Low threshold for the readers
- Low threshold for the authors
- Completeness
- Consistency
- Maintainability
- Scalability
- Evolvability
- Process to ensure the quality of the information
Requirements from Reader Point of View

Convenient
  viewing
  printing
  searching
easy
  fast
Requirements per Document

- High cohesion (within the unit)
- Low coupling (outside of the unit)
- Accessibility for the readers
- Low threshold for the reader
- Low threshold for the author
- Manageable steps to create, review, and change
- Clear responsibilities
- Clear position and relation with the context
- Well-defined status of the information
- Timely availability
Accessibility Requirements

Ease of reading, “juiciness”
High signal-to-noise ratio: information should not be hidden in a sea of words.

Understandability
Reachability in different ways, e.g., by hierarchical or full search
Reachability in a limited number of steps
Responsibility Requirements

- single author
- limited amount of reviewers
well defined documentation structure

overview specifications at higher aggregation levels

recursive application of structure and overview

delegation of review process
The Stakeholders of a Single Document

- **Project leader**: is responsible for time, budget, result
- **architect or editor**: is responsible for technical
- **context**: interacts with...
- **others**: interacts with

**Granularity of Documentation**

- **author**: writes
- **specification**: describes
- **implementation**: realizes
- **producer**: realizes
- **consumer**: uses
- **artifact**: relation
- **stakeholder**: relation

*version: 1.2 September 9, 2018*
Decomposition of Large Documents

compound document

overview

document structure

document

document

document

document

document

version: 1.2
September 9, 2018
DGcompoundDocument
Documentation Tree by Recursive Decomposition
Payload: the Ratio between Content and Overhead

Granularity of Documentation
12 Gerrit Muller

version: 1.2
September 9, 2018
DGpayload
Abstract

LEAN product development is in the process and means area pragmatic. Low tech tools, such as paper, pen and magnets, with very direct interaction are used. For communication the use of single A3-size documents is promoted, because this is a manageable amount of information.
Characteristics of LEAN

A holistic, systems approach to product development including people, processes, and technology.

Multi-disciplinary from the early start, with a drive to be fact based.

Customer understanding as the starting point.

Continuous improvement and learning as cultural value.

Small distance between engineers and real systems, including manufacturing, sales and service and the system of interest.
Example of A3 Architecture Overview

A3 architecture overview of the Metal Printer (all numbers have been removed for competitive sensitivity)

Document meta-information

key performance parameters

Customer key-drivers and Key Performance Parameters

1. Close doors
2. Align
3. Move to proximity
4. Process
5. Move substrate unloading position
6. Open doors

$ t_{\text{print}} = t_{\text{prepare}} + t_{\text{align}} + t_{\text{chamber}}(\text{thickness}) + t_{\text{finalize}}$

$ t_{\text{prepare}} = t_{\text{close doors}} + t_{\text{move to proximity}}$

$ t_{\text{finalize}} = t_{\text{move to unload}} + t_{\text{open doors}}$

note: original diagram was annotated with actual performance figures for confidentiality reasons these numbers have been removed

LEAN and A3 Approach to Supporting Processes

version: 0.1
September 9, 2018
LEANoverviewA3

Gerrit Muller
multiple related views

quantifications

one topic per A3

capture "hot" topics

digestable (size limitation)

practical close to stakeholder experience

source: PhD thesis Daniel Borches http://doc.utwente.nl/75284/
Light Weight Review Process

by Gerrit Muller  
University of South-Eastern Norway-NISE

e-mail: gaudisite@gmail.com  
www.gaudisite.nl

Abstract

A light weight review process is described that can be used for documents made during product creation. This review process is focused on improving the contents of specifications as early as possible. The process is light weight to increase the likelihood that it is performed *de facto* instead of *pro forma*.
Product Life Cycle and Change Management

**micro specification control board**
- project team present
- specification = communications means
- very dynamic, many changes
- light weight review process

**maintenance control board**
- no project team any more
- documentation = organizational memory
- changes only to cope with logistics or safety problems
Light Weight Specification Review Process

the author is responsible for contents and organization of the flow (consults and review)

draft

final review = final check contents

consultation & review

- wide group of people, with an active concern or an expected contribution;
- many iterations
- multiple media:
  + meetings,
  + on paper
  + informal et cetera

concept

change request

authorization = check process

authorized

specification specific Change Control Board
4 peoples/roles:
1 producer
1 consumer
1 context
1 independent

criteria for reviewers:
+ know how
+ critical
+ sufficient time

by "lowest" operational manager:
project leader, subsystem PL, ...

version: 0
September 9, 2018
LWRstateDiagram
Abstract

The introduction of a new process (way of working) is quite often implemented by supplying ready-to-go tools and templates. This implementation mainly serves the purpose of a smooth introduction of the new process. Unfortunately the benefits of templates are often cancelled by unforeseen side-effects, such as unintended application, inflexibility, and so on. This intermezzo gives hints to avoid the Template Trap, so that templates can be used more effectively to support introduction of new processes.
Rationale for Templates

- Low threshold to apply a (new) process (1)
- Low effort to apply a (new) process (2)
- No need to know low level implementation details (3)
- Means to consolidate and reuse experiences (4)
Bogus Arguments for Templates

- Obtain a uniform look (5)
- Force the application of a (new) process (6)
- Control the way a new process is applied (7)
Forces of Change: Action = - Reaction

\[ \sum \text{all Forces} \]

\[ \text{induces} \]

\[ \text{counteract} \]

\[ \text{Reaction} \]

\[ \text{Net change} \]
Template as Support for Process

principle \( \rightarrow \) process \( \xrightarrow{drives} \) elaborated \( \xrightarrow{in} \) procedure \( \xrightarrow{is \ supported \ by} \) tool template

abstract \( \leftarrow \) specific and executable
Types of Templates

- Header
- Body
- Footer

**recommended template type**

- Header
- Body
- Footer

layout only

- Title, Date
- Body
- Page, Author

meta information

- Title
- Author
- Abstract
- 1 Introduction
- 2 Scope
- 3 Design
- 17 Interfaces

prescribing contents
<table>
<thead>
<tr>
<th>template type</th>
<th>context knowhow</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>layout only</td>
<td>no</td>
<td>low</td>
</tr>
<tr>
<td>meta information</td>
<td>process</td>
<td>high</td>
</tr>
<tr>
<td>prescribing content</td>
<td>process and domain</td>
<td>constraining</td>
</tr>
<tr>
<td>● Use templates for meta-information.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Use checklists for structure and contents.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Template Development

Templates are an optimization of the Copy Paste Modify pattern:

- Look for a similar problem
- Copy its implementation
- Modify the copy to fulfil the new requirements
Spiral model: Use before Re-use

Extract template → Implement document

Evaluate → Use
Mandatory per page:

- Author
- Title
- Status
- Version
- Date of last update
- Unique Identification
- Business Unit
- Page number
Mandatory per document:

- Distribution (Notification) list
- Reviewers and commentators
- Document scope (Product family, Product, Subsystem, Module as far as applicable)
- Change history
Recommended Practice:
- Short statement on frontpage stating what is expected from the addressed recipients, for example:
  - Please send comments before February 29, this document will be reviewed on that date
  - This document is authorized, changes are only applied via a change request
- See Granularity of Documentation [?] for guidelines for modularization and contents
Template Pitfalls

- Author follows template instead of considering the purpose of the document.
- Template is too complex.
- There is an unmanageable number of variants.
- Mandatory use of templates results in:
  - no innovation of templates (= no learning)
  - no common sense in deployment
  - strong dependency on templates

**Recommendation:**
- Enforce the procedure *(what)*
- Provide the template *(how)* as supporting means.
• Templates support (new) processes

• Use templates for layout and meta information support

• Do not use templates for documents structure or contents

• Stimulate evolution of templates, keep them alive

• Keep templates simple

• Standardize on what (process or procedure), not on how (tool and template)

• Provide (mandatory) guidelines and recommended practices

• Provide templates as a supportive choice, don’t force people to use templates
Abstract

In this document we will discuss the full integration flow. We will discuss the goal of integration, the relation between integration and testing, what is integration and how to integrate, an approach to integration, scheduling and dealing with disruptive events, roles and responsibilities, configuration management aspects, and typical order of integration problems occurring in real life.
Typical Concurrent Product Creation Process

-1. strategy
0. feasibility
1. definition
2. system design
3. engineering
4. integration & test
5. field monitoring
6. product operational life cycle

- policy
- requirements
- design
- integrate
- test

- requirements and specification
- design
- integrate
- test
Zooming in on Integration and Tests

0. feasibility
1. definition
2. system design
3. engineering
4. integration & test
5. field monitoring
6. product operational life cycle
Integration Takes Place in a Bottom-up Fashion

- Component
- Subsystem
- System function
- Product
- Context

integrate
alpha

test
Transition from Previous System to New System

2 partial systems for SW testing

2 existing base systems

new base systems

adopt existing base SW

new application

SW dev system

test and refine application

existing base system

integrate and refine application

SW for new HW subsystem

test SW for new HW subsystem

new HW subsystem

test HW subsystem

integrate subsystem

existing base system

test HW subsystem

integrate subsystem

adopt existing base SW

new base system

test new base system

integrate HW system

integrate HW system

integrated system

new subsystem integration

application integration

existing

new

time
Alternatives to Integrate a Subsystem Early in the Project

- Physical reality
- Physical simulated
- Complex virtual
- Simple

Spectrum

Virtual environment

Simulated subsystems

Stubs

To-be-integrated subsystem

Physical environment

(Modeled) existing subsystems

(Prototype) new subsystems
## Stepwise Integration Approach

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Determine most critical system performance parameters.</td>
</tr>
<tr>
<td>2</td>
<td>Identify subsystems and functions involved in these parameters.</td>
</tr>
<tr>
<td>3</td>
<td>Work towards integration configurations along these chains of subsystems and functions.</td>
</tr>
<tr>
<td>4</td>
<td>Show system performance parameter as early as possible; start with showing &quot;typical&quot; system performance.</td>
</tr>
<tr>
<td>5</td>
<td>Show &quot;worst-case&quot; and &quot;boundary&quot; system performance.</td>
</tr>
<tr>
<td>6</td>
<td>Rework manual integration tests in steps into automated regression tests.</td>
</tr>
<tr>
<td>7</td>
<td>Monitor regression results with human-driven analysis.</td>
</tr>
<tr>
<td>8</td>
<td>Integrate the chains: show system performance of different parameters simultaneously on the same system.</td>
</tr>
</tbody>
</table>
Roles and Responsibilities During the Integration Process

- **project leader**
  - organization
  - resources
  - schedule
  - budget

- **systems architect/engineer/integrator**
  - system requirements
  - design inputs
  - test specification
  - schedule rationale
  - troubleshooting
  - participate in test

- **system tester**
  - test
  - troubleshooting
  - report

- **logistics and administrative support**
  - configuration
  - orders
  - administration

- **engineers**
  - design
  - component test
  - troubleshooting
  - participate in test

- **machine owner**
  - maintain test model
  - support test
Configuration Management Entities

supplier
- Customer Oriented Process
  - components
  - test models
  - TPD

company
- goods flow
  - content of pipeline
  - technical product documentation (TPD)
  - specifications
  - test models

customer
- product lifecycle
  - requirements
  - tender

System Integration How-To
version: 0.2
September 9, 2018
Typical Order of Integration Problems

1. The (sub)system does not build.
2. The (sub)system does not function.
3. Interface errors.
4. The (sub)system is too slow.
5. Problems with the main performance parameter, such as image quality.
6. The (sub)system is not reliable.
Make a design for the documentation structure of the case, take into account a.o.:

- target audience per documentation module
- lifecycle
- author
- size (budget)

Present (max 1 flip) the proposed documentation structure and the rationale.
Documentation

Requirements Entire Documentation

Accessibility for the readers
Low threshold for the readers
Low threshold for the authors
Completeness
Consistency
Maintainability
Scalability
Evolvability
Process to ensure the quality of the information

Requirements per Document

High cohesion (within the unit)
Low coupling (outside of the unit)
Accessibility for the readers
Low threshold for the reader
Low threshold for the author
Manageable steps to create, review, and change
Clear responsibilities
Clear position and relation with the context
Well-defined status of the information
Timely availability

Decompose Large Documents

Recursive Decomposition

Summary Module Supporting Processes
version: 0.2
September 9, 2018
Maximize Payload

front page

- title
- identification
- author
- distribution
- status
- review

- history
- changes

- diagrams
- tables

meta information
max 2 pages

contents
2-18 pages

1. aap
2. noot
3. mies
lists
and ca 50% text

Light Weight Review

- consultation
  review

draft

final review
= final check contents

concept

authorization
= check process

intentionally left blank

the author is responsible for contents and organization of the flow (consults and review)

summary module supporting processes

version: 0.2
September 9, 2018
Systems Integration

Integration Starts at Feasibility

Alternatives for Early Integration

Propagation of Configuration Issues

Summary Module Supporting Processes

version: 0.2
September 9, 2018