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

\[ \sum \text{all Forces} \quad \text{induces} \quad \text{Reaction} \quad \text{counteract} \quad \text{Support} \quad = \quad \text{Net change} \]
Template as Support for Process

principle — *drives* — process — *elaborated* in — procedure — *supported* by — tool

formalism

abstract — specific and executable — template
Types of Templates

- **Header**
  - **Body**
  - **Footer**
  - **layout only**

- **Title**
  - **Author**
  - **Abstract**

- **Title, Date**
  - **Page, Author**

**recommended template type**

- **Title, Date**
  - **Body**
  - **Page, Author**

- **Title, Date**
  - **3 Design**
  - **Page, Author**

- **Title, Date**
  - **17 Interfaces**
  - **Page, Author**

**prescribing contents**

**meta information**

**Template How To**

Gerrit Muller

version: 1.6
March 6, 2013
THTtypes
<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>
</tbody>
</table>

- Use templates for meta-information.
- Use checklists for structure and contents.
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

version: 1.6
March 6, 2013
THTdevelopmentSpiral
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