Abstract

Most products fit in a larger family of products. The members of such a product family share a lot of functionality and features. It is attractive to share implementations, designs et cetera between those members to increase the efficiency of the entire company.

In practice many difficulties pop up when product developments become coupled, due to the partial developments which are shared. This article discusses the advantages and disadvantages of a family approach based on shared developments and provides some methods to increase the chance on success.
Typical Examples of Generic Developments

- Platform
- Common components
- Standard design
- Framework
- Family architecture
- Generic aspects, functions, or features
- Reuse
- Products (in project environment)
### Claimed Advantages of Generic Developments

<table>
<thead>
<tr>
<th>Advantage</th>
<th>Implication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduced time to market</td>
<td>building on shared components</td>
</tr>
<tr>
<td>Reduced cost per function</td>
<td>build every function only once</td>
</tr>
<tr>
<td>Improved quality</td>
<td>maturing realization</td>
</tr>
<tr>
<td>Improved reliability</td>
<td></td>
</tr>
<tr>
<td>Improved predictability</td>
<td></td>
</tr>
<tr>
<td>Easier diversity management</td>
<td>modularity</td>
</tr>
<tr>
<td>Increases uniformity</td>
<td>less learning</td>
</tr>
<tr>
<td>Employees only have to understand one base system</td>
<td></td>
</tr>
<tr>
<td>Larger purchasing power</td>
<td>economy of scale</td>
</tr>
<tr>
<td>Means to consolidate knowledge</td>
<td></td>
</tr>
<tr>
<td>Increase added value</td>
<td>not reinventing existing functionality</td>
</tr>
<tr>
<td>Enables parallel developments of multiple products</td>
<td></td>
</tr>
<tr>
<td>“Free” feature propagation</td>
<td>product-to-product or project-to-project</td>
</tr>
</tbody>
</table>
### Experiences with reuse, from counterproductive to effective

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduced time to market</td>
<td>Longer time to market</td>
</tr>
<tr>
<td>Reduced investment</td>
<td>High investments</td>
</tr>
<tr>
<td>Reduced (shared) maintenance cost</td>
<td>Lots of maintenance</td>
</tr>
<tr>
<td>Improved quality</td>
<td>Poor quality</td>
</tr>
<tr>
<td>Improved reliability</td>
<td>Poor reliability</td>
</tr>
<tr>
<td>Easier diversity management</td>
<td>Diversity is opposed</td>
</tr>
<tr>
<td>Understanding of one base system</td>
<td>Lot of know how required</td>
</tr>
<tr>
<td>Improved predictability</td>
<td>Predictable too late</td>
</tr>
<tr>
<td>Larger purchasing power</td>
<td>Dependability</td>
</tr>
<tr>
<td>Means to consolidate knowledge</td>
<td>Knowledge dilution</td>
</tr>
<tr>
<td>Increase added value</td>
<td>Lack of market focus</td>
</tr>
<tr>
<td>Enables parallel developments</td>
<td>Interference</td>
</tr>
<tr>
<td>Free feature propagation</td>
<td>But integration required</td>
</tr>
</tbody>
</table>

Product Families and Generic Aspects

Gerrit Muller

version: 2.3
June 21, 2020
SWReXperiences
### Successful examples of reuse

<table>
<thead>
<tr>
<th>Homogeneous Domain</th>
<th>cath lab</th>
<th>MRI</th>
<th>television</th>
<th>waferstepper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Dominated</td>
<td>car</td>
<td>airplane</td>
<td>shaver</td>
<td>television</td>
</tr>
<tr>
<td>Limited Scope</td>
<td>audio codec</td>
<td>compression library</td>
<td>streaming library</td>
<td></td>
</tr>
</tbody>
</table>
Limits of successful reuse

- struggle with integration/convergence with other domains
  - TV: digital networks and media
  - cath lab: US imaging, MRI

- poor/slow response on paradigm shifts
  - TV: LCD screens
  - cath lab: image based acquisition control

- software maintenance, configurations, integration, release
  - MRI: integration and test
  - wafersteppers: number of configurations

how to innovate?
Drivers for Generic Developments

Customer value
- application adaptability
- availability variations
- new features originating from different products
- timely availability
- reliability
- asset creation
- increase economy of scale

Internal benefits
- availability of accumulated feature set
- design for configurability
- shared architectural framework
- quality increase
- predictability
- availability integrated base product
- maturity

Extrovert driver
Introvert driver
Granularity of generic developments shown in 2 dimensions

Product Families and Generic Aspects
8   Gerrit Muller
version: 2.3
June 21, 2020
GDgranularity
Modified Process Decomposition

Product Families and Generic Aspects

version: 2.3  
June 21, 2020

Gerrit Muller
Financial Viewpoint on Process Decomposition

Management

Cashflow generation

Tomorrow’s Cashflow

Strategic Asset Generation

Assets

Customer

Sales

Logistics

Production

Service

Presales

Generics

People

Technology

Process

Information

Order

Product

Support

Needs and Feedback

Material

Product roadmap

Budget, plan

Product needs and feedback

Customer Roadmap

Business Drivers

Product roadmap

Budget, plan

Product needs and feedback

Customer

Sales

Logistics

Production

Service

Presales

Generics

People

Technology

Process

Information

Order

Product

Support

Needs and Feedback

Material

Product roadmap

Budget, plan

Product needs and feedback

Product Families and Generic Aspects

version: 2.3
June 21, 2020
GDprocessDecompositionFamilyByValue
Value and Feedback Flow

- Policy and Planning Process
- Customer-Oriented Process
- Product Creation Process
- Shared Assets Creation Process
- People, Process, and Technology Management Process

Needs and feedback flow through the processes, impacting product development and customer satisfaction.

- Customer feedback influences product roadmap and document generation.
- Business drivers guide budget and plan related to product creation.
- Technology, process, and people roadmaps support the creation of shared assets.
- Value and feedback flow through the sales, logistics, production, and service departments.

---

Product Families and Generic Aspects

version: 2.3
June 21, 2020
GDprocessDecompositionFamilyPlusFlow
## Product Families and Generic Aspects

**Modified Operational Organization PCP**

<table>
<thead>
<tr>
<th>Operational</th>
<th>Technical</th>
<th>Commercial</th>
</tr>
</thead>
<tbody>
<tr>
<td>portfolio operational manager</td>
<td>portfolio architect</td>
<td>portfolio marketing manager</td>
</tr>
<tr>
<td>family operational manager</td>
<td>family architect</td>
<td>family marketing manager</td>
</tr>
<tr>
<td>project leader</td>
<td>single product project leader</td>
<td></td>
</tr>
<tr>
<td>single product project leader</td>
<td>platform architect</td>
<td>product manager</td>
</tr>
<tr>
<td>platform</td>
<td>product architect</td>
<td></td>
</tr>
<tr>
<td>component project leader</td>
<td>subsystem architect</td>
<td>component manager</td>
</tr>
<tr>
<td>subsystem project leader</td>
<td></td>
<td></td>
</tr>
<tr>
<td>component architect</td>
<td>subsystem architect</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>component developers</td>
<td>subsystem developers</td>
<td></td>
</tr>
</tbody>
</table>
## Sources of Failure in Generic Developments

### Technical
- Too generic
- Innovation stops (stable interfaces)
- Vulnerability

### Process/People/Organization
- Forced cooperation
- Time platform feature to market
- Unrealistic expectations
- Distance platform developer to customer
- No marketing ownership
- Bureaucratic process (no flexibility)
- New employees, knowledge dilution
- Underestimation of platform support
- Overstretching of product scope
- Nonmanagement, organizational scope increase
- Underestimation of integration
- Component/platform determines business policy
- Subcritical investment