

Module Product Families and Generic Developments

by *Gerrit Muller* HBV-NISE

e-mail: `gaudisite@gmail.com`

`www.gaudisite.nl`

Abstract

This module addresses product families and generic developments.

Distribution

This article or presentation is written as part of the Gaudí project. The Gaudí project philosophy is to improve by obtaining frequent feedback. Frequent feedback is pursued by an open creation process. This document is published as intermediate or nearly mature version to get feedback. Further distribution is allowed as long as the document remains complete and unchanged.

October 20, 2017

status: preliminary

draft

version: 1.3

Product Families and Generic Aspects

by *Gerrit Muller* Buskerud University College

e-mail: gaudisite@gmail.com

www.gaudisite.nl

Abstract

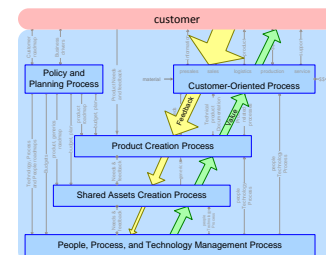
Most products fit in a larger family of products. The members of such a product family share a lot of functionality and features. It is attractive to share implementations, designs et cetera between those members to increase the efficiency of the entire company.

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.

Distribution

This article or presentation is written as part of the Gaudí project. The Gaudí project philosophy is to improve by obtaining frequent feedback. Frequent feedback is pursued by an open creation process. This document is published as intermediate or nearly mature version to get feedback. Further distribution is allowed as long as the document remains complete and unchanged.

October 20, 2017
status: concept
version: 2.3



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

Reduced time to market	building on shared components
Reduced cost per function	build every function only once
Improved quality	maturing realization
Improved reliability	
Improved predictability	
Easier diversity management	modularity
Increases uniformity	less learning
Employees only have to understand one base system	
Larger purchasing power	economy of scale
Means to consolidate knowledge	
Increase added value	not reinventing existing functionality
Enables parallel developments of multiple products	
“Free” feature propagation	product-to-product or project-to-project

Experiences with reuse, from counterproductive to effective

bad

longer time to market
high investments
lots of maintenance
poor quality
poor reliability
diversity is opposed
lot of know how required
predictable too late
dependability
knowledge dilution
lack of market focus
interference
but integration required

good

reduced time to market
reduced investment
reduced (shared) maintenance cost
improved quality
improved reliability
easier diversity management
understanding of one base system
improved predictability
larger purchasing power
means to consolidate knowledge
increase added value
enables parallel developments
free feature propagation

Successful examples of reuse

homogeneous domain

cath lab
MRI
television
waferstepper

hardware dominated

car
airplane
shaver
television

limited scope

audio codec
compression library
streaming library

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

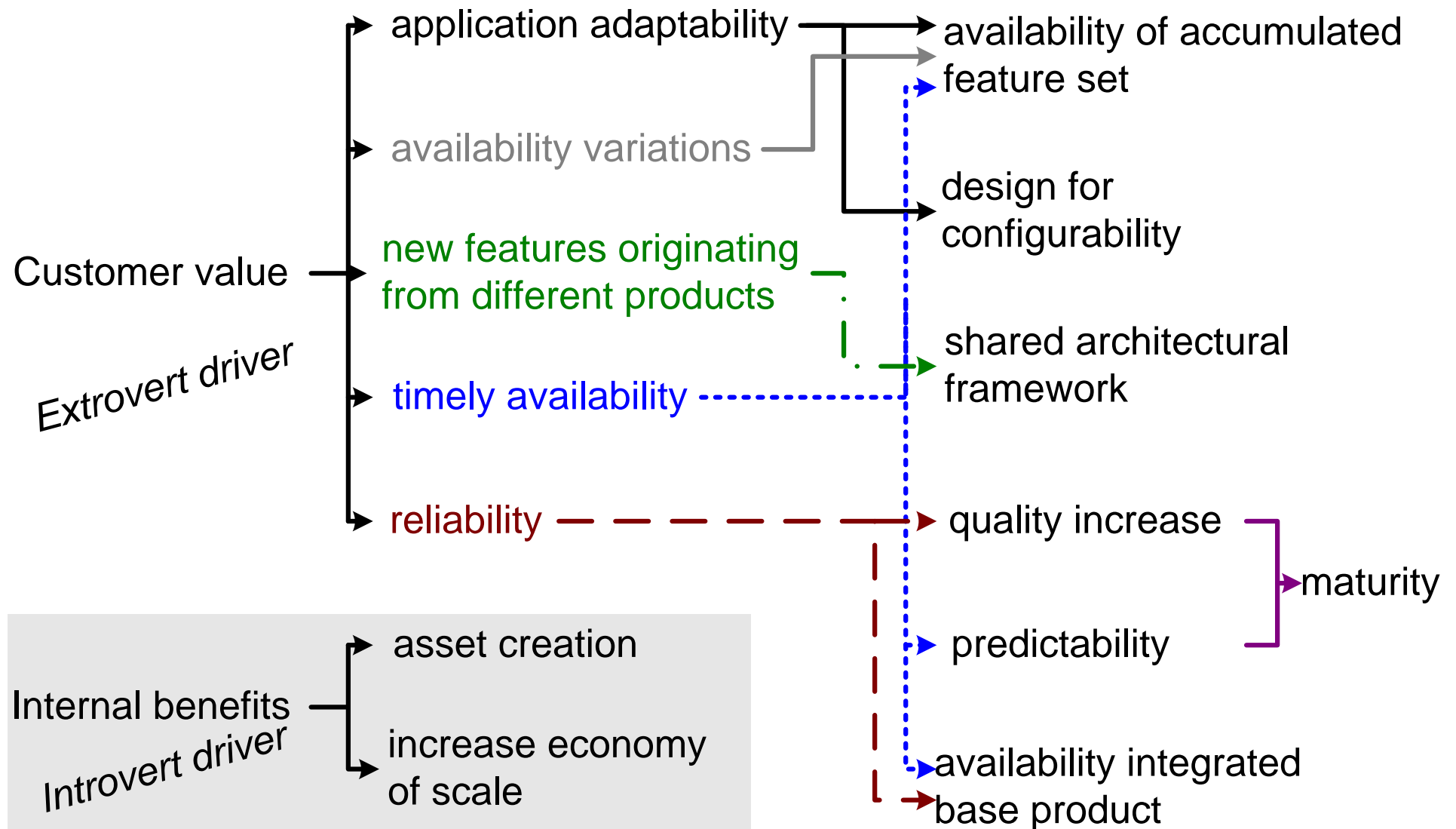
TV: LCD screens
cath lab: image based acquisition control

software maintenance, configurations, integration, release

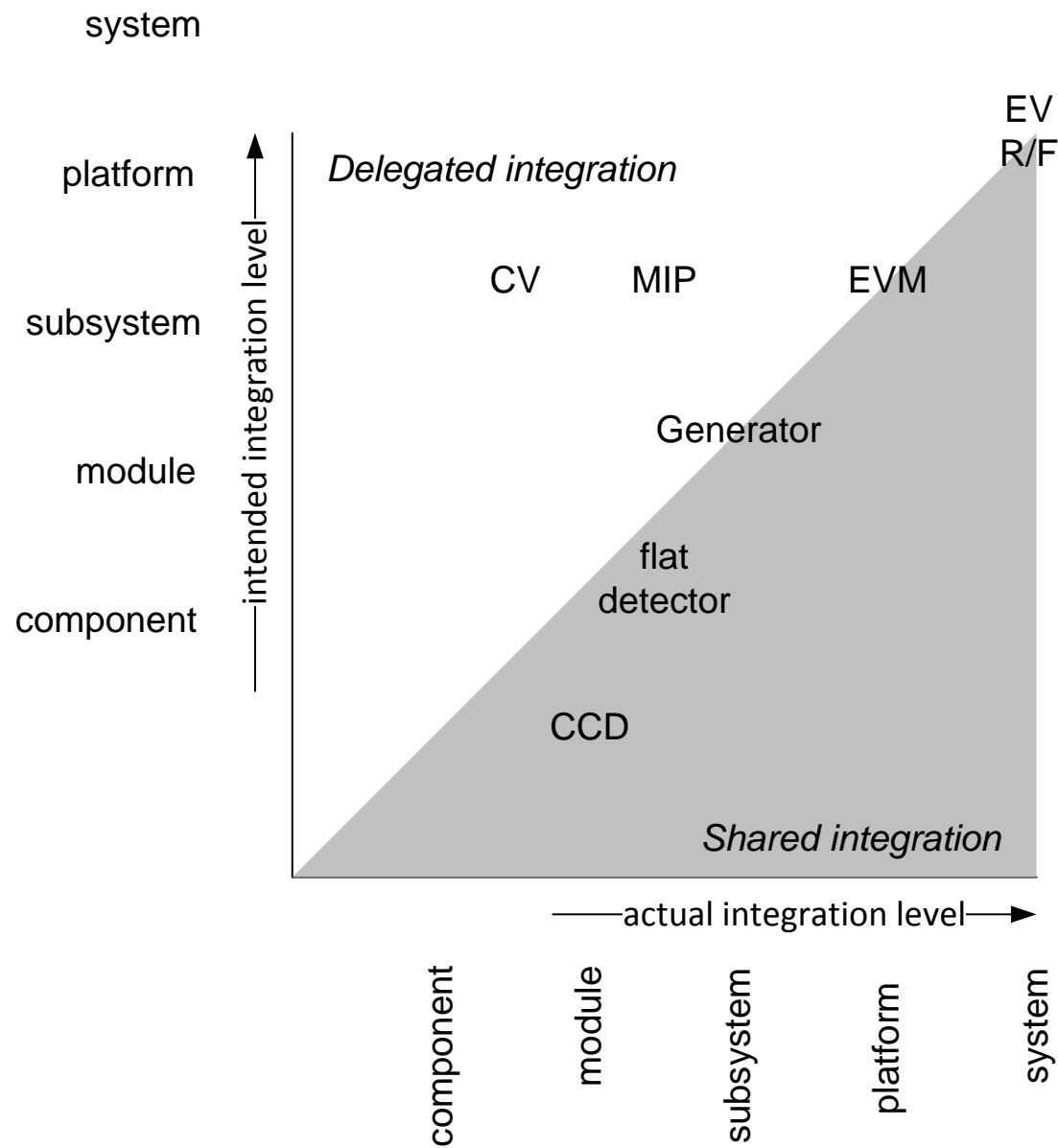
MRI: integration and test
wafersteppers: number of configurations

how to innovate?

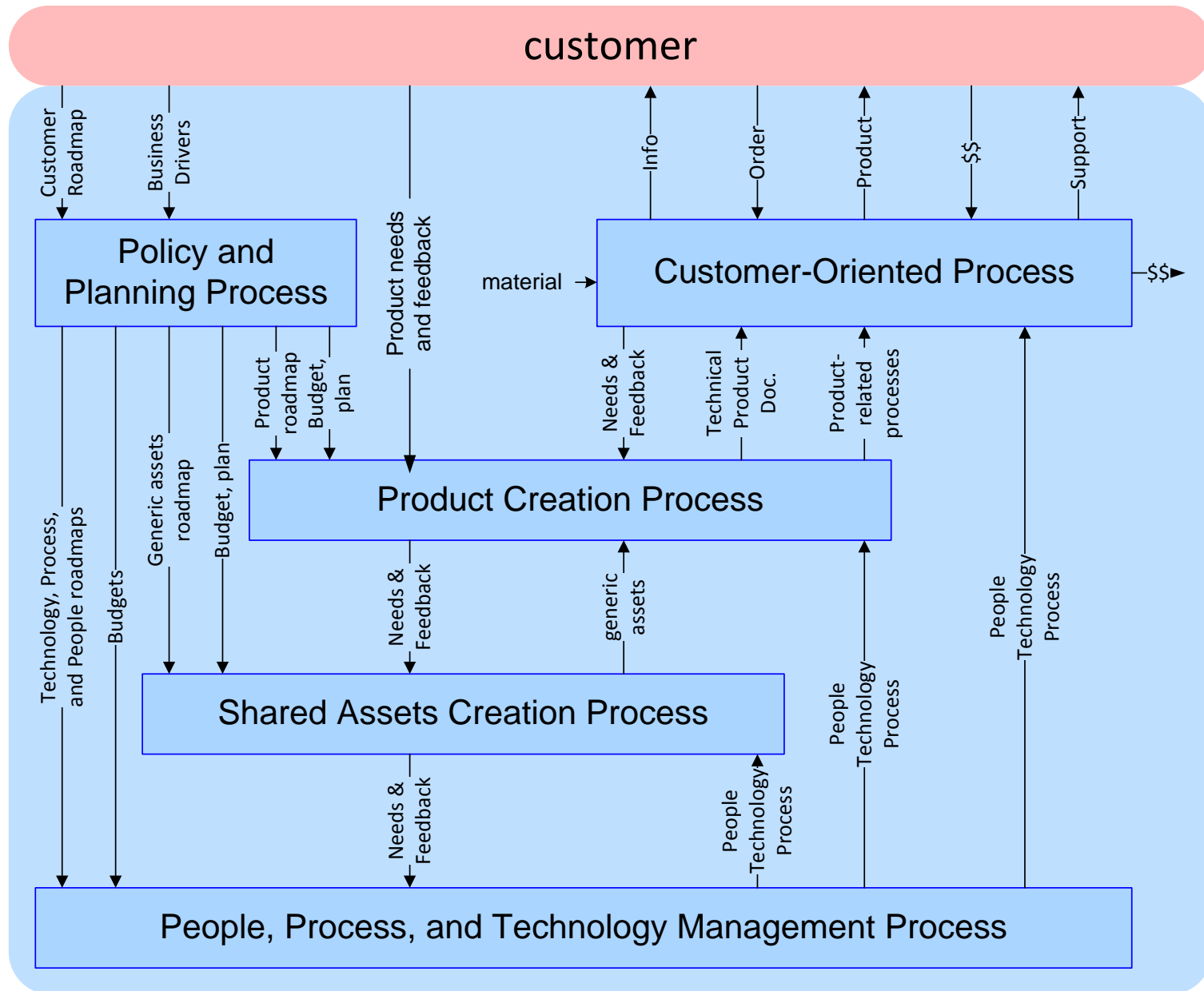
Drivers for Generic Developments



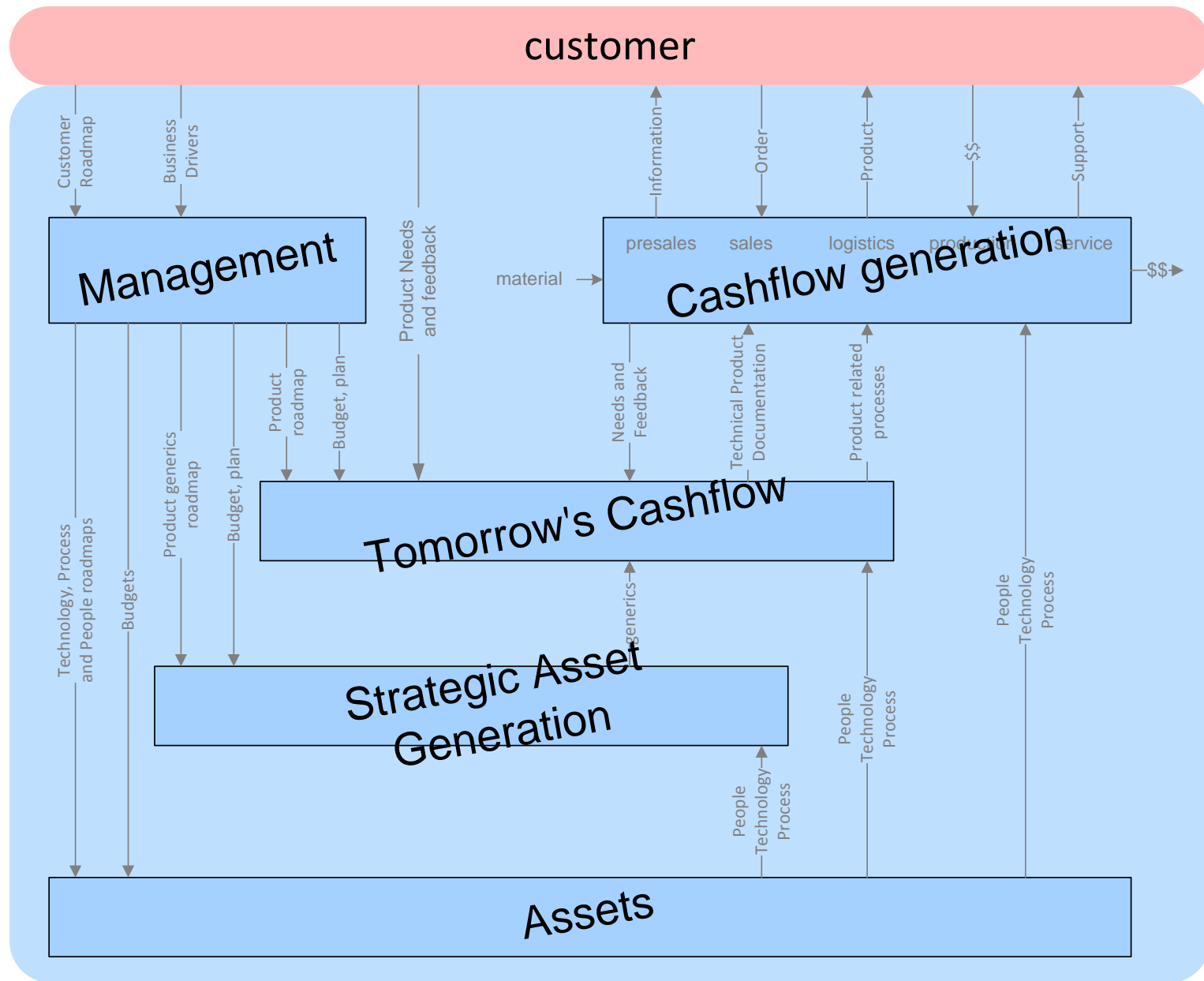
Granularity of generic developments shown in 2 dimensions



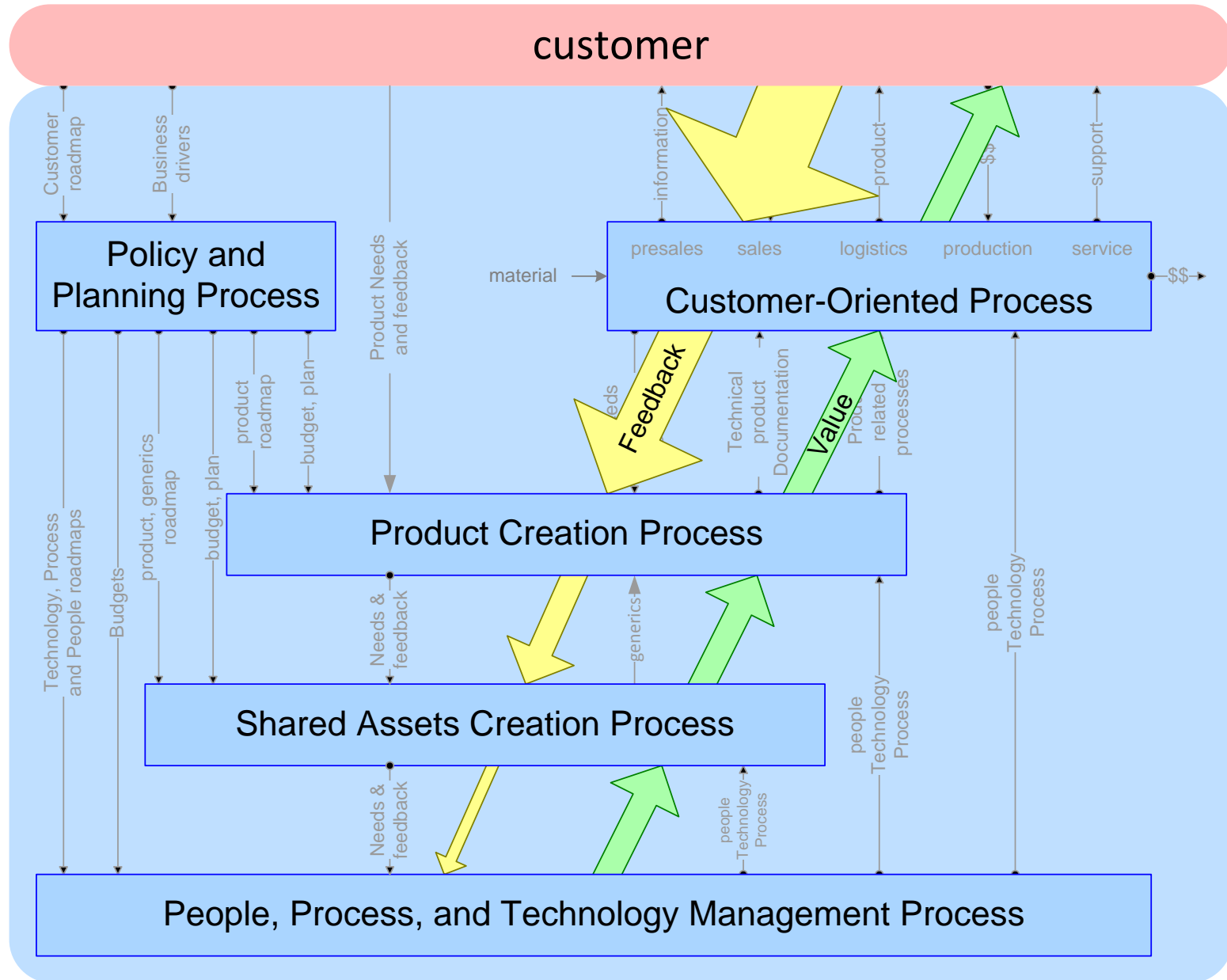
Modified Process Decomposition



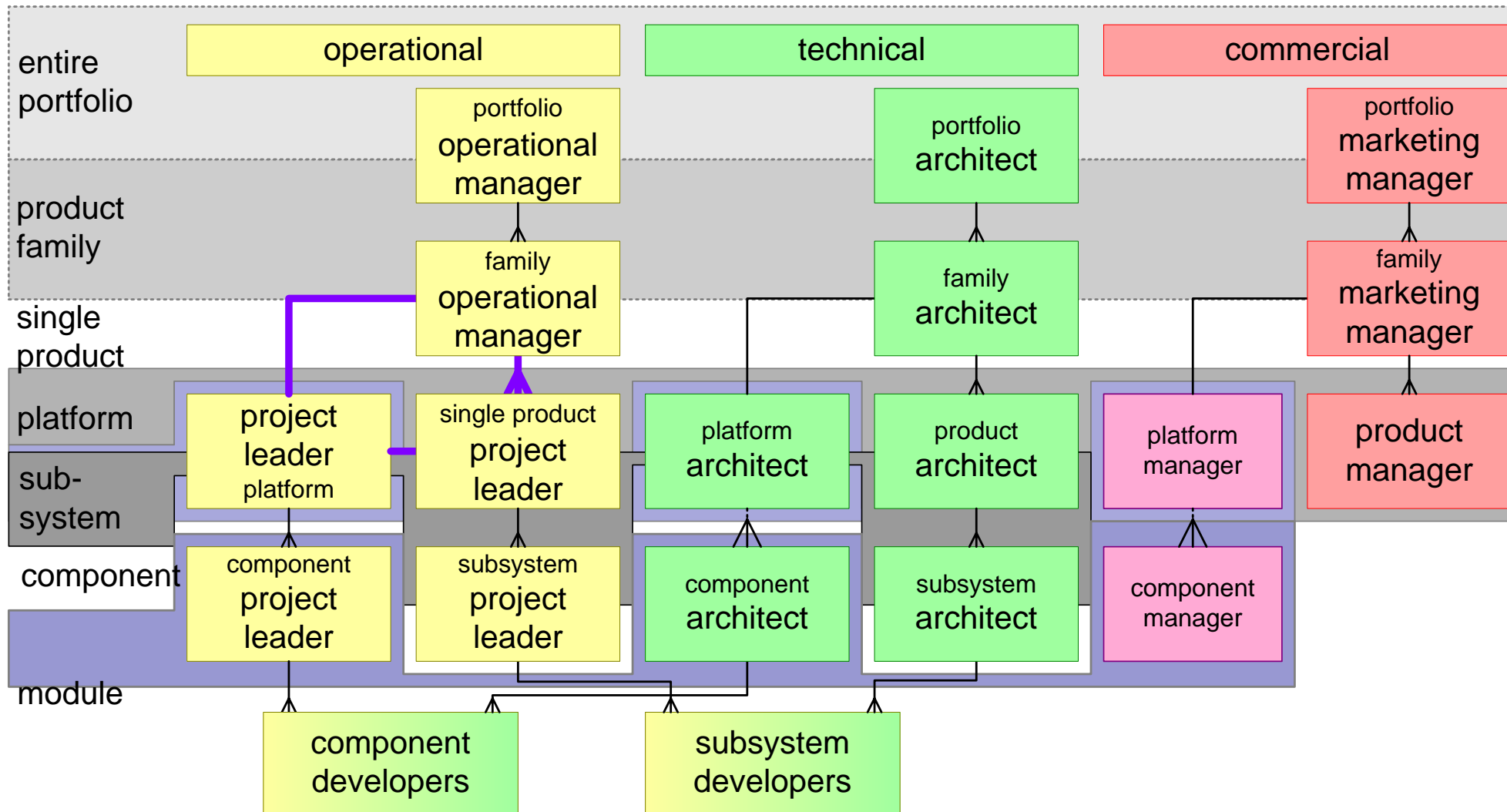
Financial Viewpoint on Process Decomposition



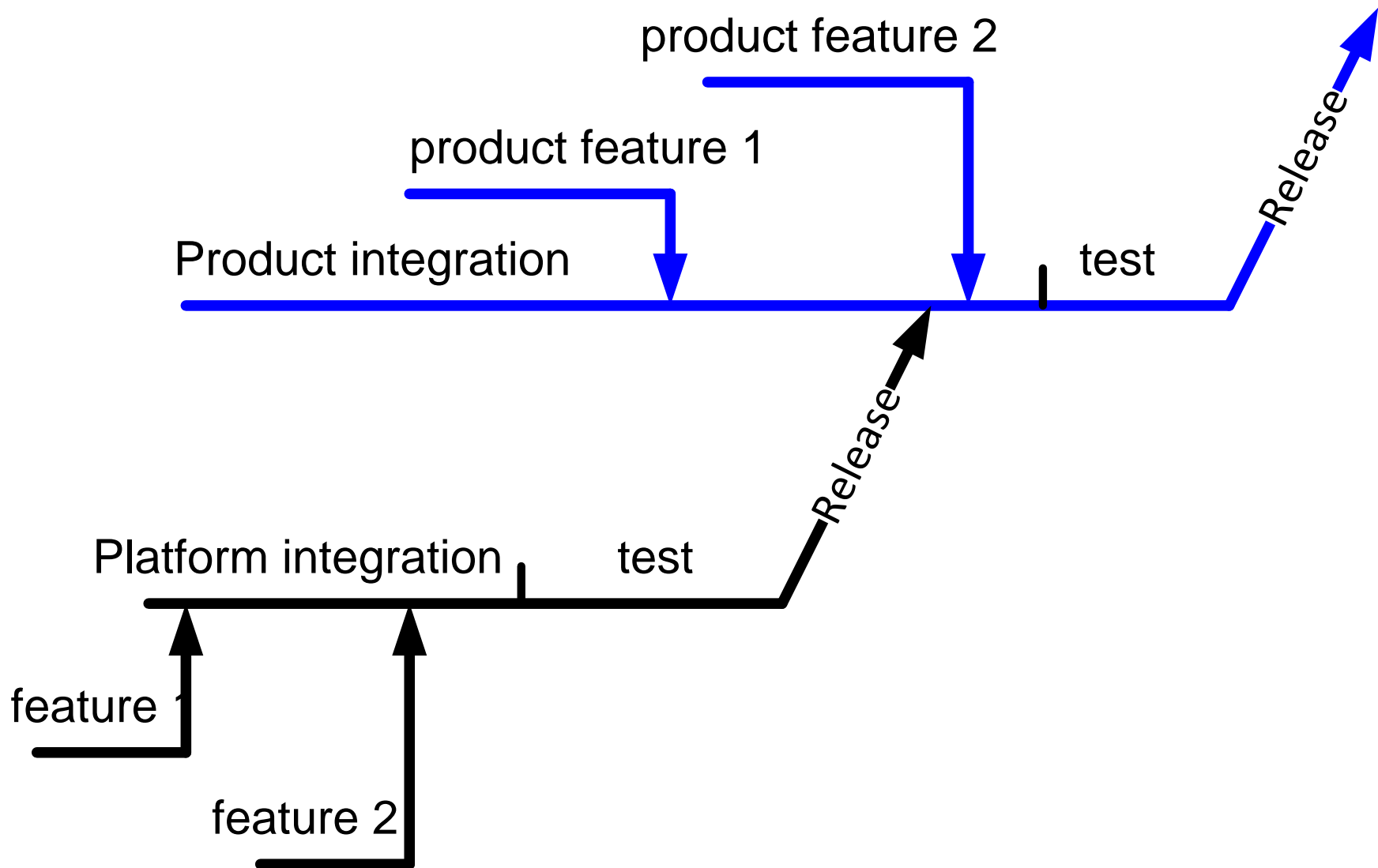
Value and Feedback Flow



Modified Operational Organization PCP



Propagation Delay Platform Feature to Market



Sources of Failure in Generic Developments

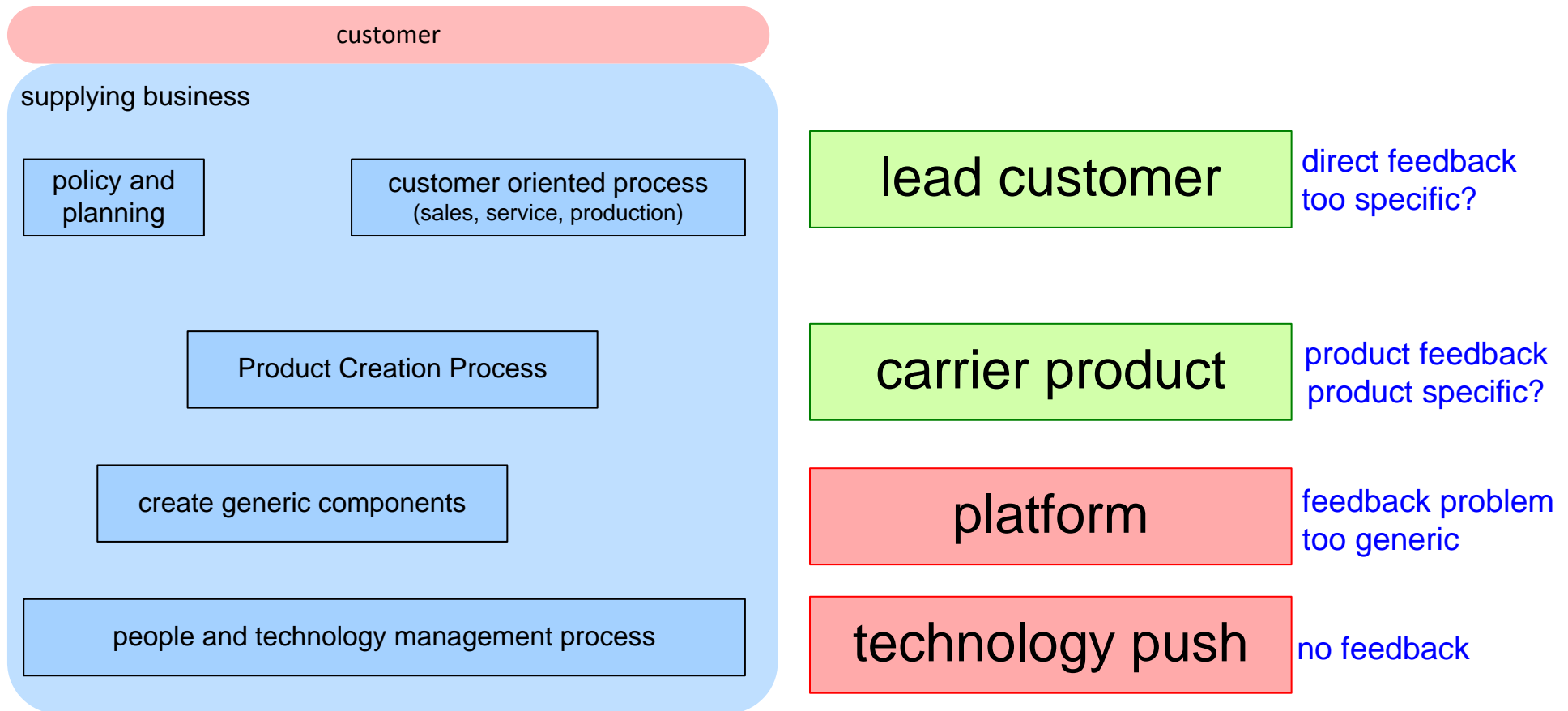
Technical

- 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

Models for Generic Development



Exercise Generic Developments

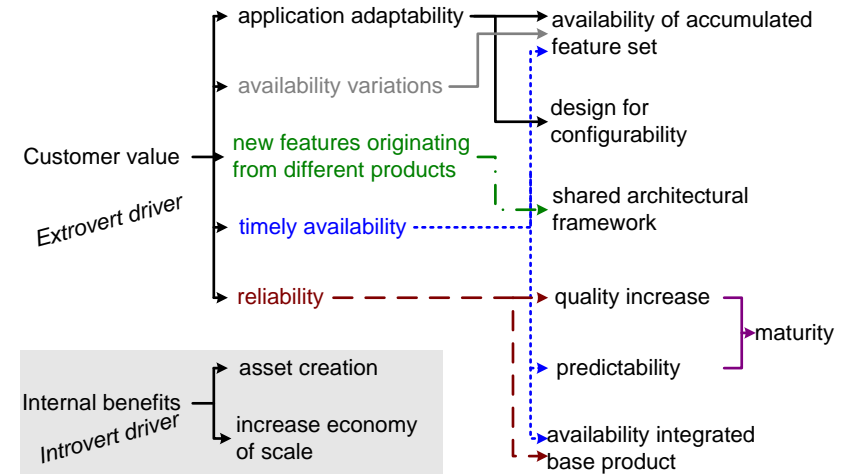
What are the top 3 benefits for your product family or generic development?
What are the top 3 disadvantages?

Harvesting Synergy

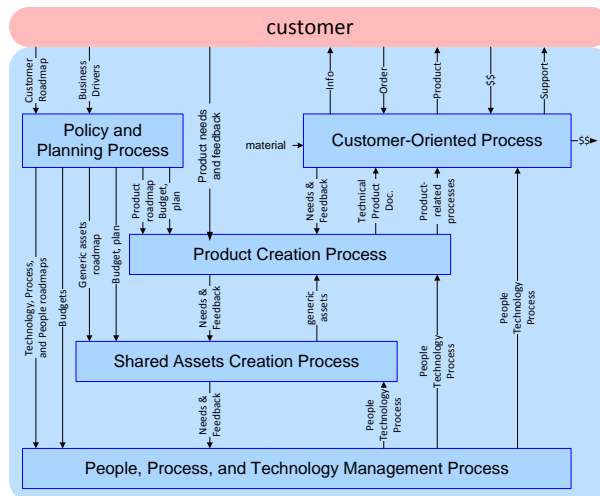
Contradicting Experiences

<p>bad</p> <ul style="list-style-type: none"> longer time to market high investments lots of maintenance poor quality poor reliability diversity is opposed lot of know how required predictable too late dependability knowledge dilution lack of market focus interference but integration required 	<p>good</p> <ul style="list-style-type: none"> reduced time to market reduced investment reduced (shared) maintenance cost improved quality improved reliability easier diversity management improved predictability larger purchasing power means to consolidate knowledge increase added value enables parallel developments free feature propagation
---	--

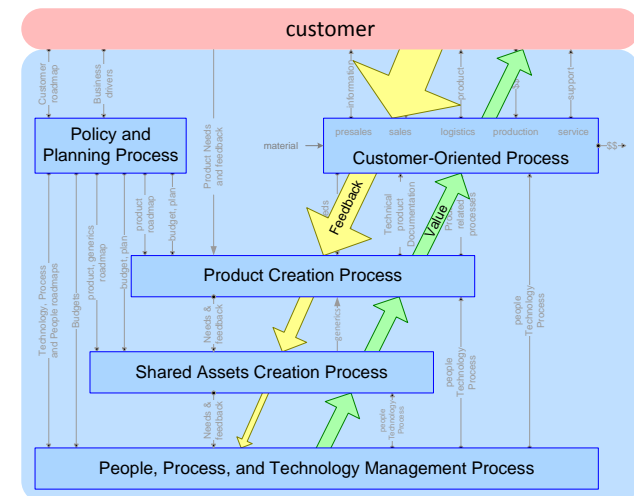
Drivers



Shared Asset Creation Process

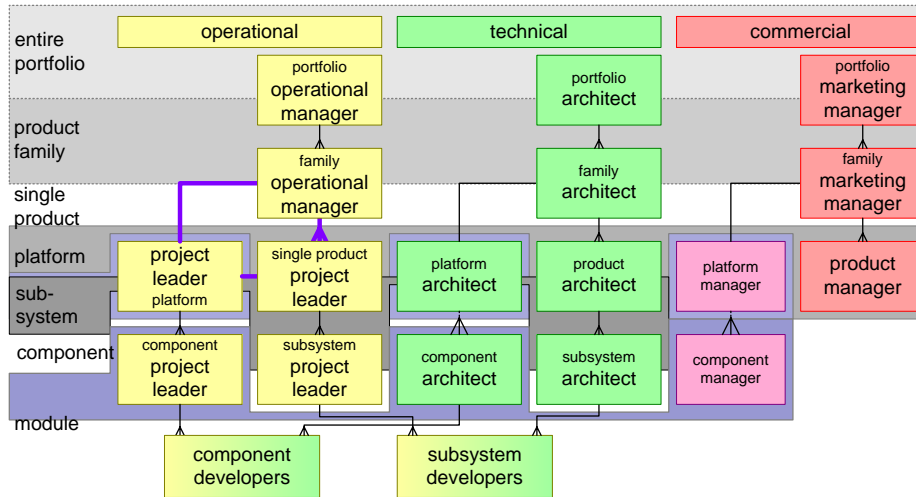


Longer Chains

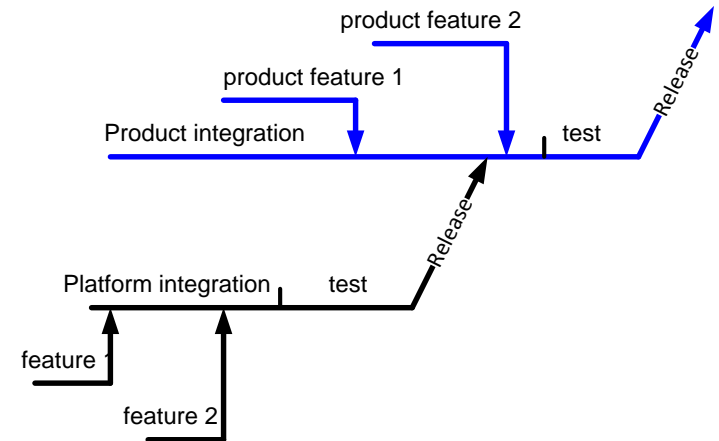


Some Architecting Means

Organizational Complexity



Delay to Market



Pitfalls

Technical	Process/People/Organization
<ul style="list-style-type: none"> • Too generic • Innovation stops (stable interfaces) • Vulnerability 	<ul style="list-style-type: none"> • 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

Successful and Failing Models

