Module Platform and Evolvability; Process and People

by *Gerrit Muller* HSN-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module provides processes and insights in people, processes and organization issues for evolvable platforms.

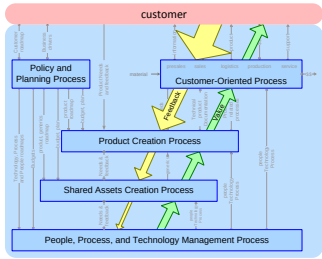
September 9, 2018
status: planned
version: 0

Product Families and Generic Aspects

by *Gerrit Muller* University of South-Eastern Norway-NISE
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.



September 9, 2018
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

Improved reliability

Improved predictability

→ maturing realization

Easier diversity management — modularity

Increases uniformity

Employees only have to understand one base system

→ less learning

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	good
longer time to market	reduced time to market
high investments	reduced investment
lots of maintenance	reduced (shared) maintenance cost
poor quality	improved quality
poor reliability	improved reliability
diversity is opposed	easier diversity management
lot of know how required	understanding of one base system
predictable too late	improved predictability
dependability	larger purchasing power
knowledge dilution	means to consolidate knowledge
lack of market focus	increase added value
interference	enables parallel developments
but integration required	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

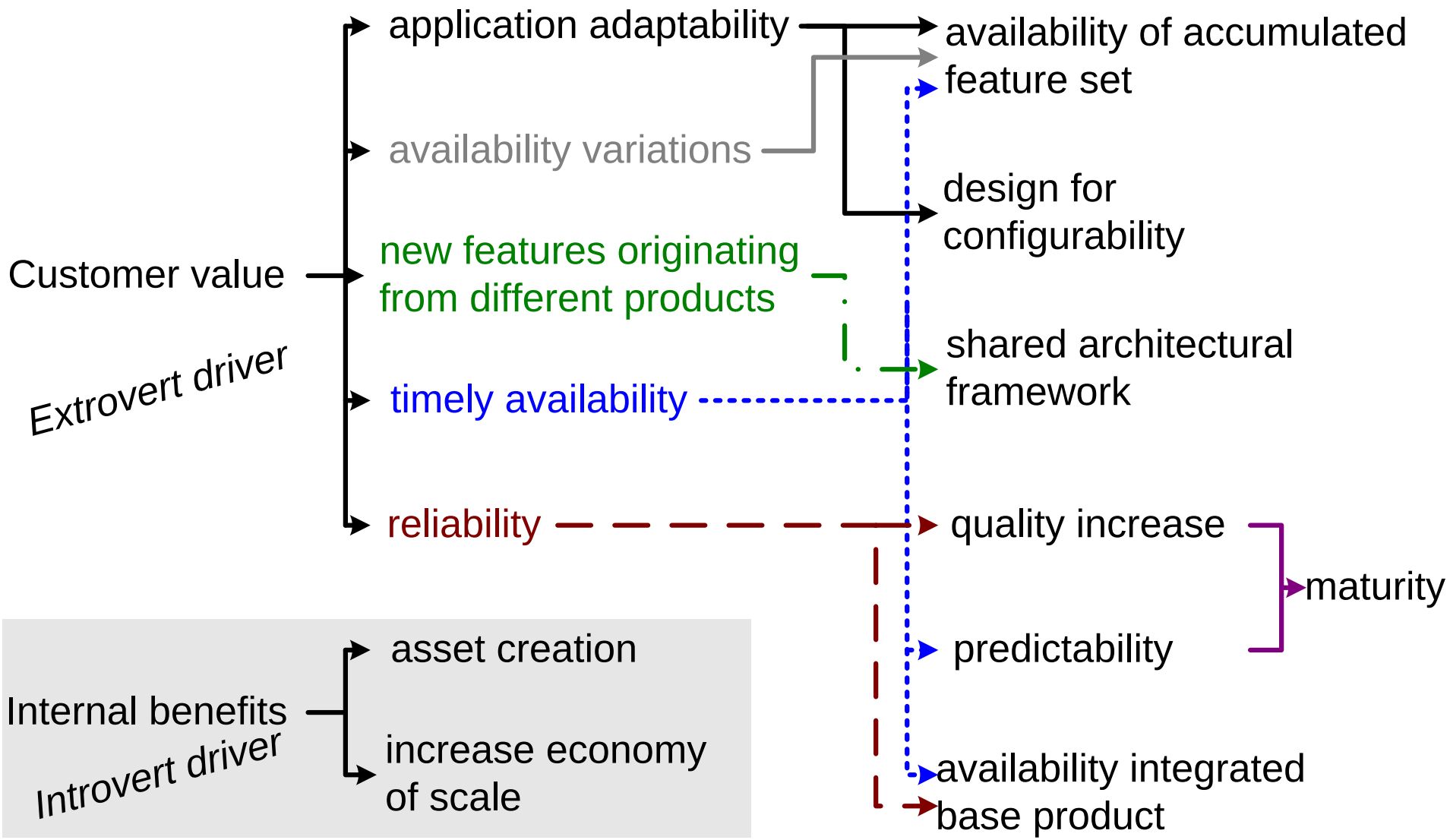
TV: LCD screens
cath lab: image based acquisition control

software maintenance, configurations, integration, release

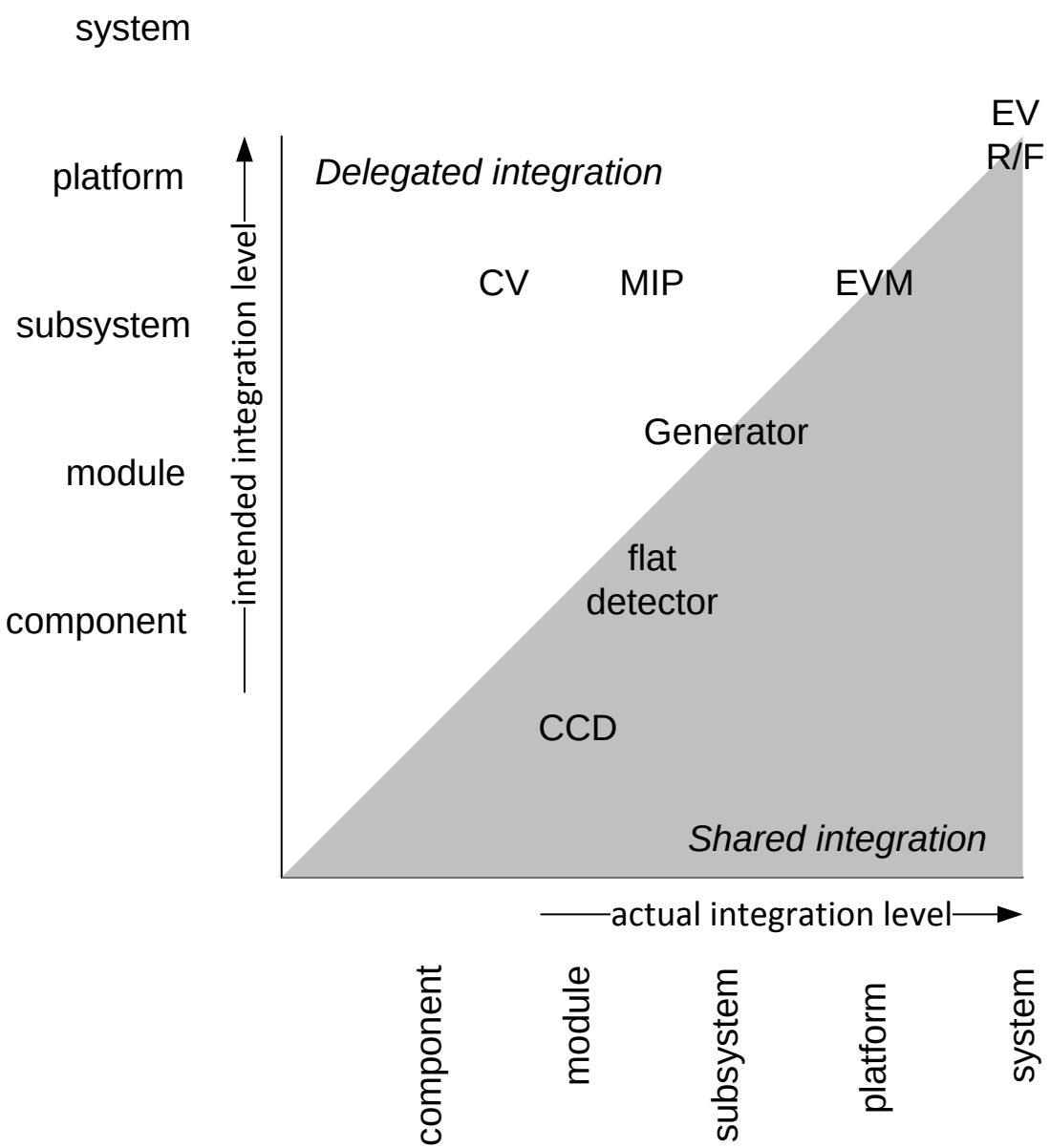
MRI: integration and test
wafersteppers: number of configurations

how to innovate?

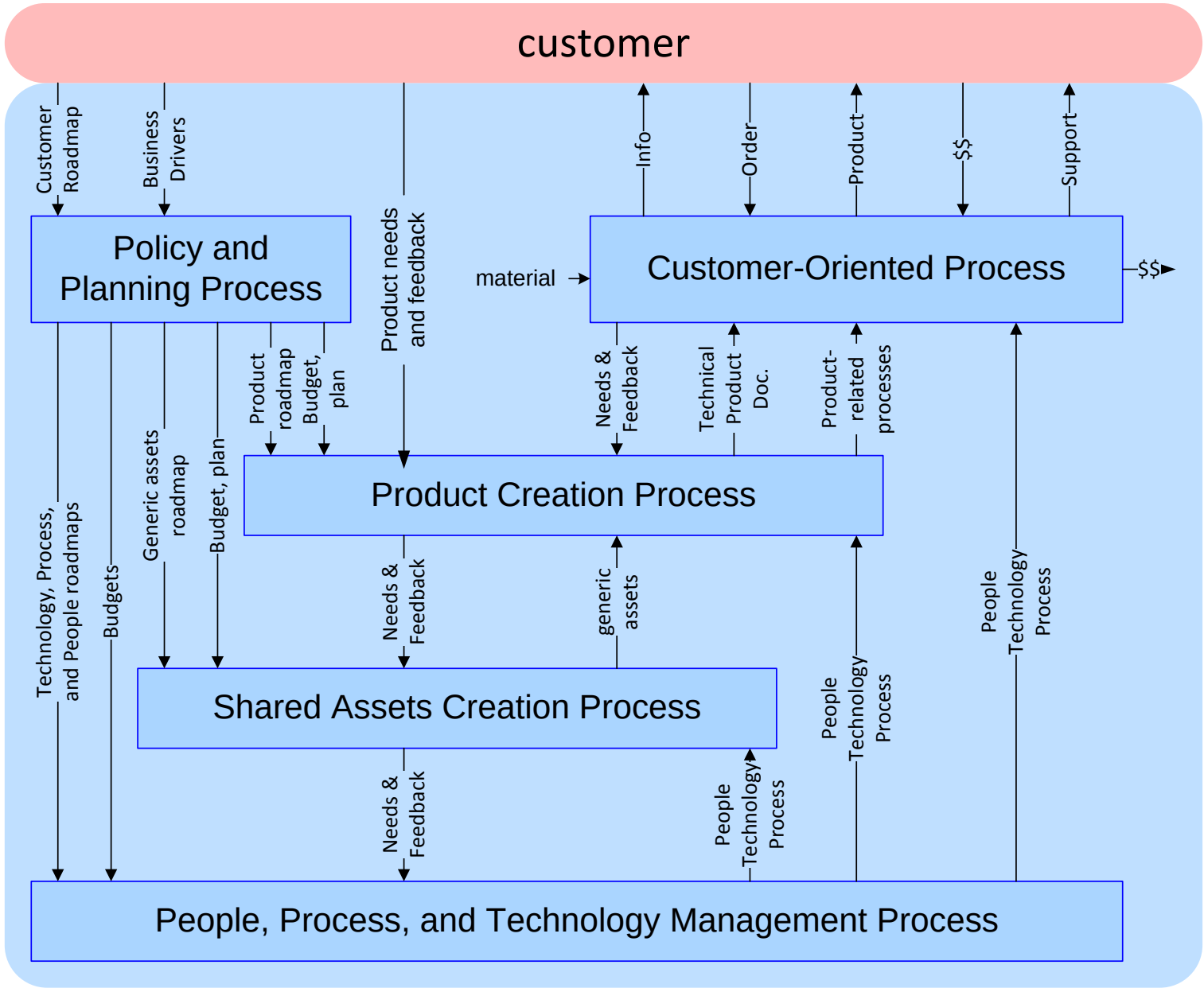
Drivers for Generic Developments



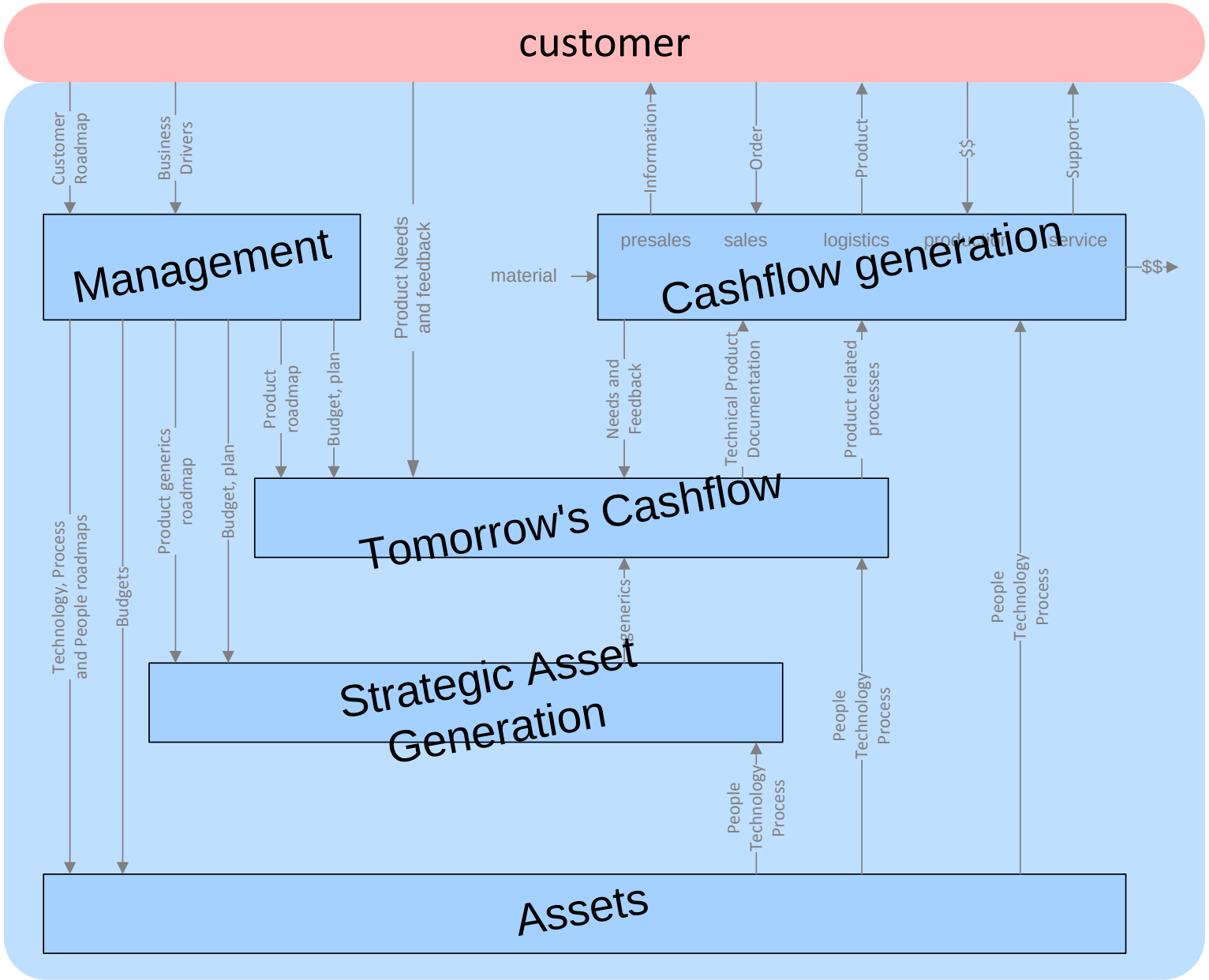
Granularity of generic developments shown in 2 dimensions



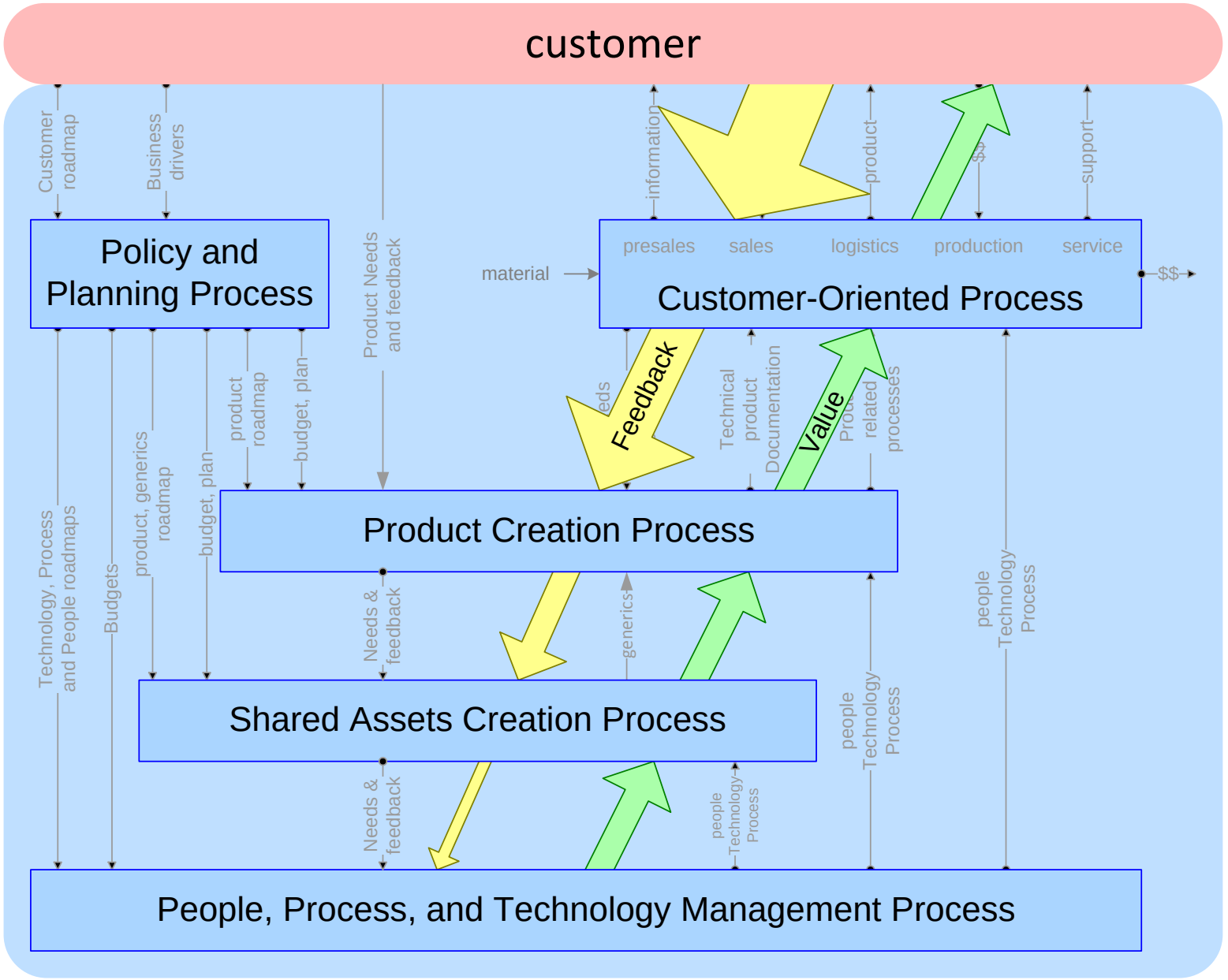
Modified Process Decomposition



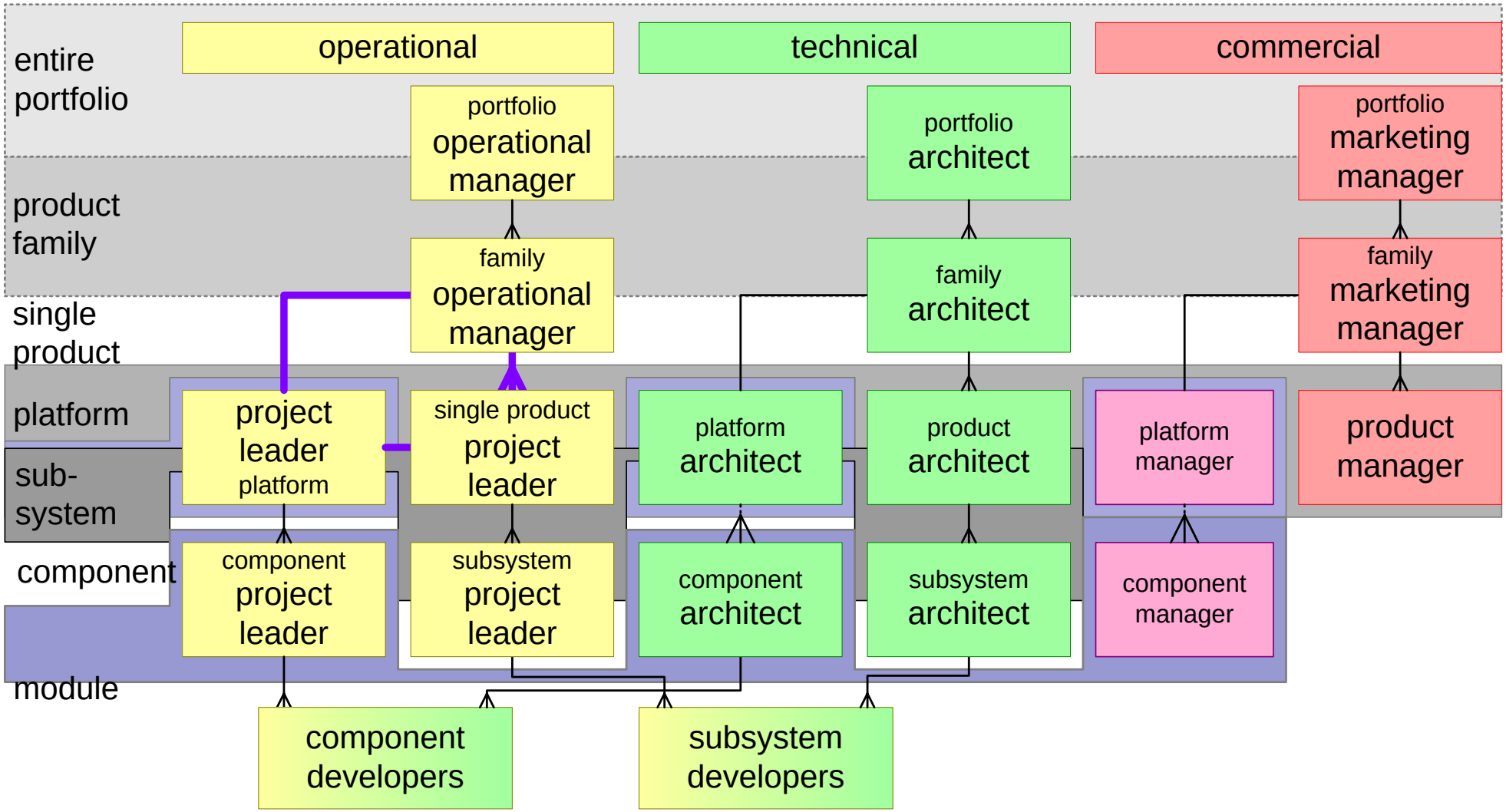
Financial Viewpoint on Process Decomposition



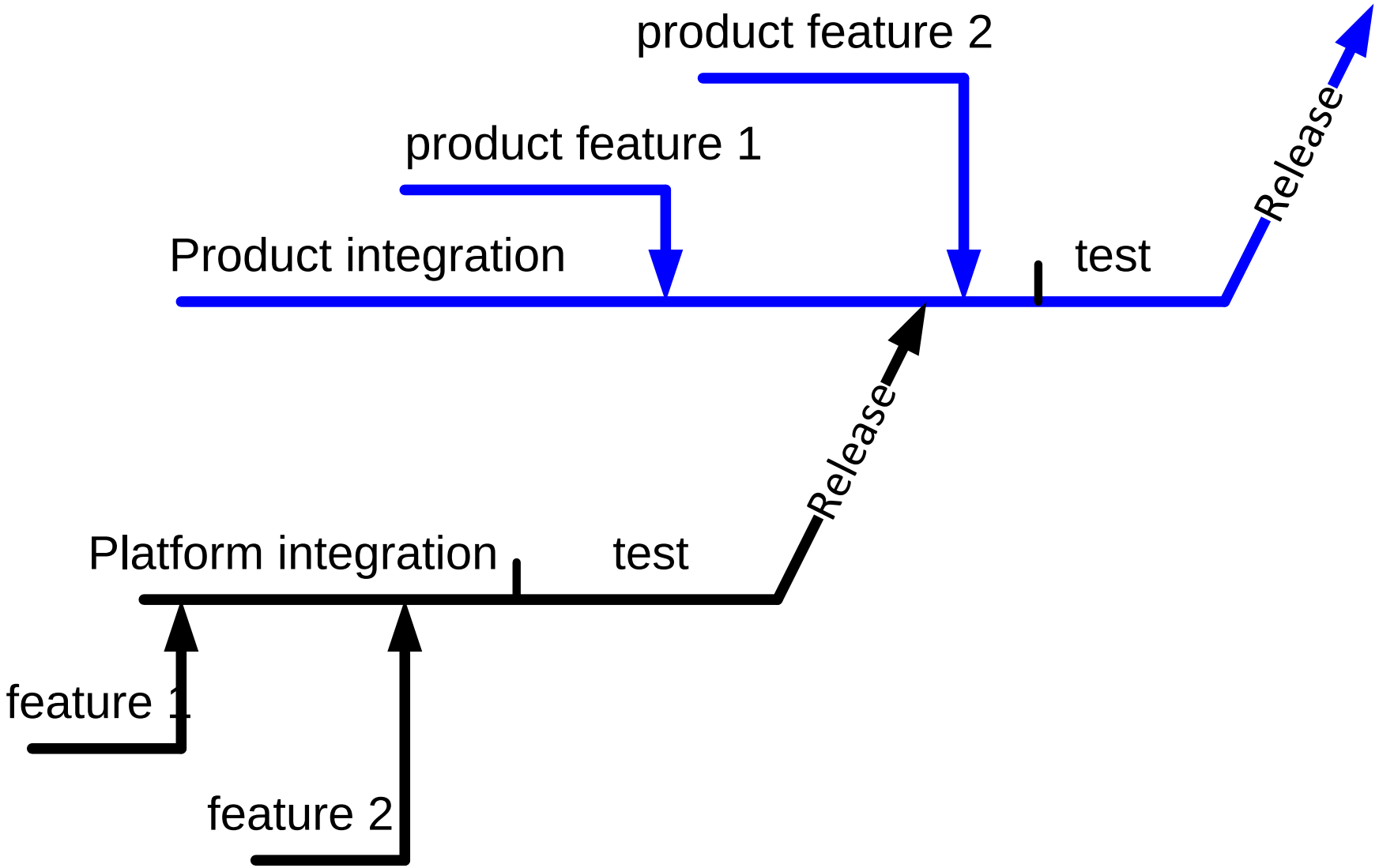
Value and Feedback Flow



Modified Operational Organization PCP



Propagation Delay Platform Feature to Market



Sources of Failure in Generic Developments

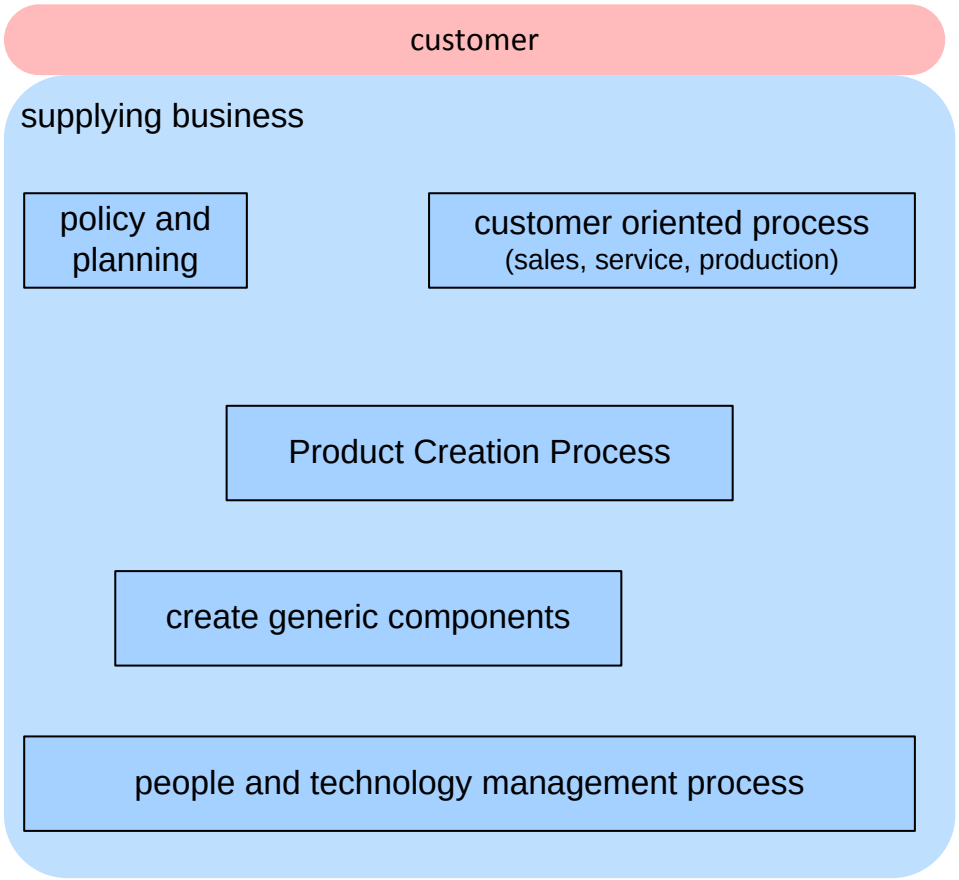
Technical

- 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



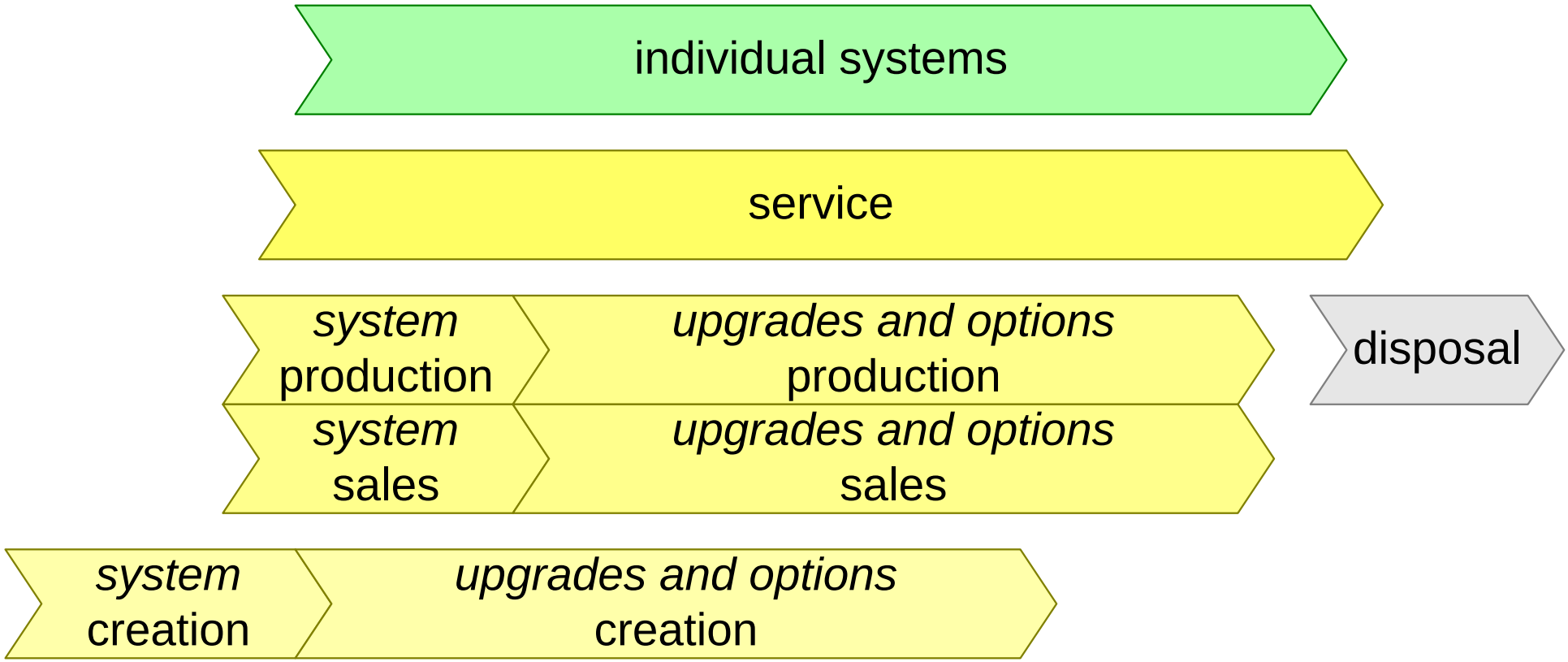
lead customer — direct feedback, too specific?

carrier product — product feedback, product specific?

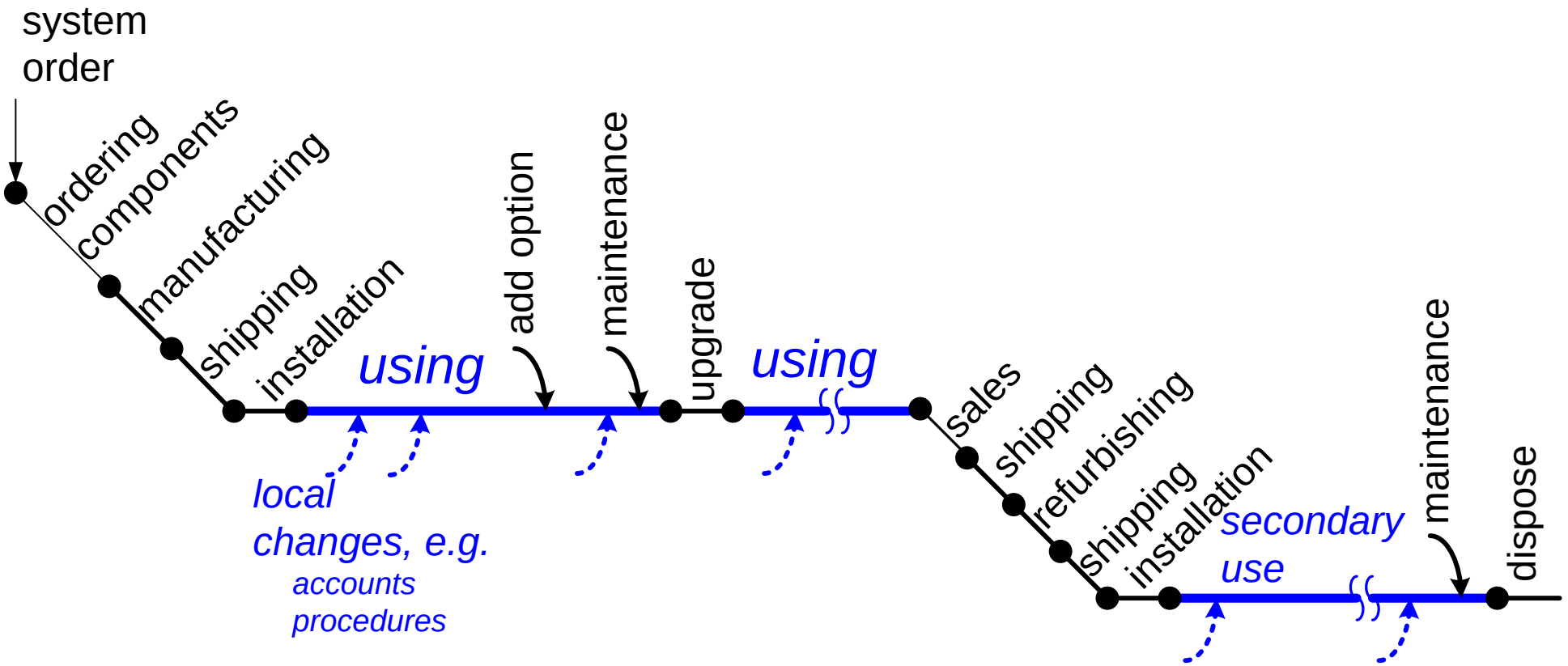
platform — feedback problem, too generic

technology push — no feedback

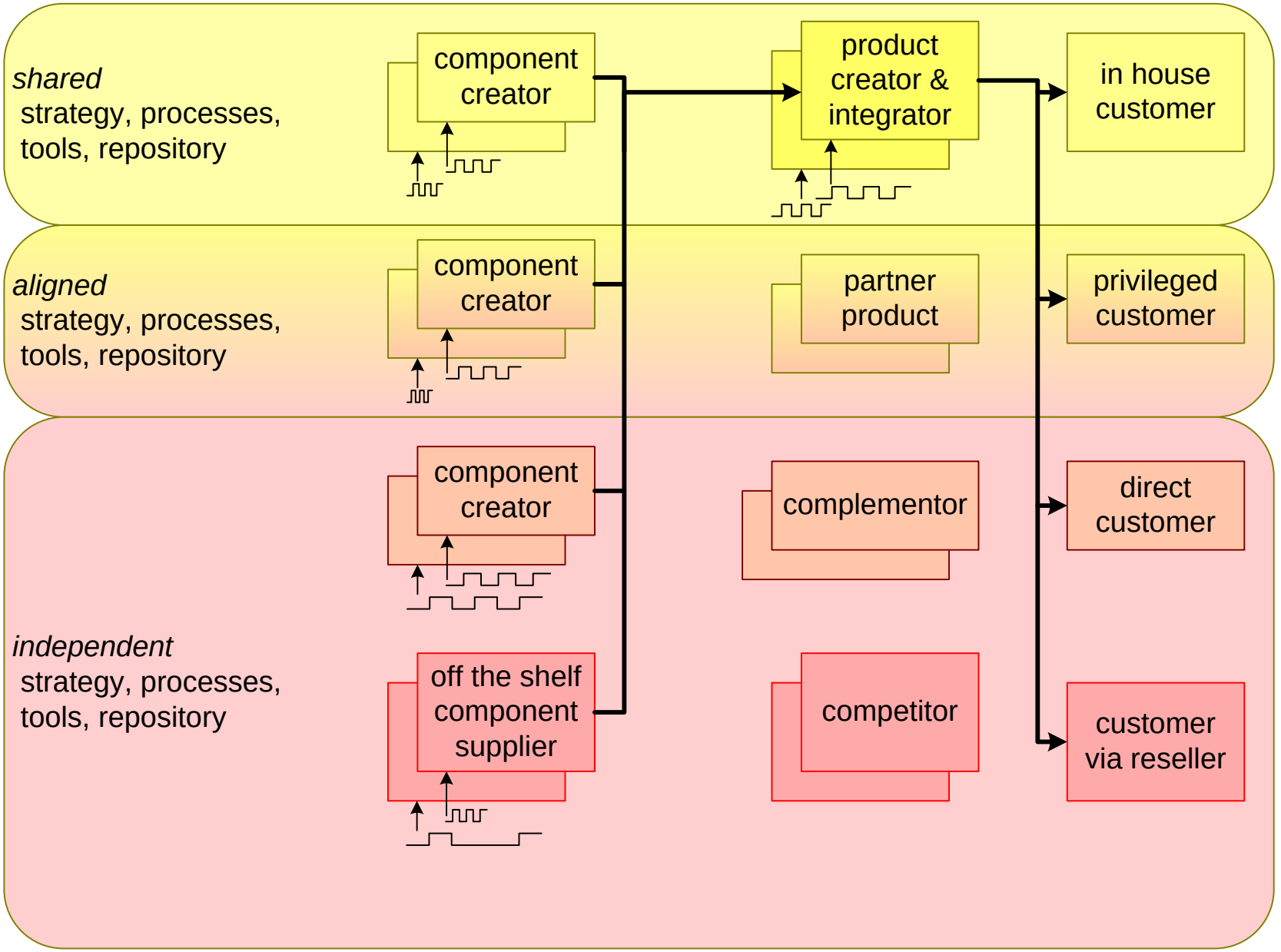
Product Related Life Cycles



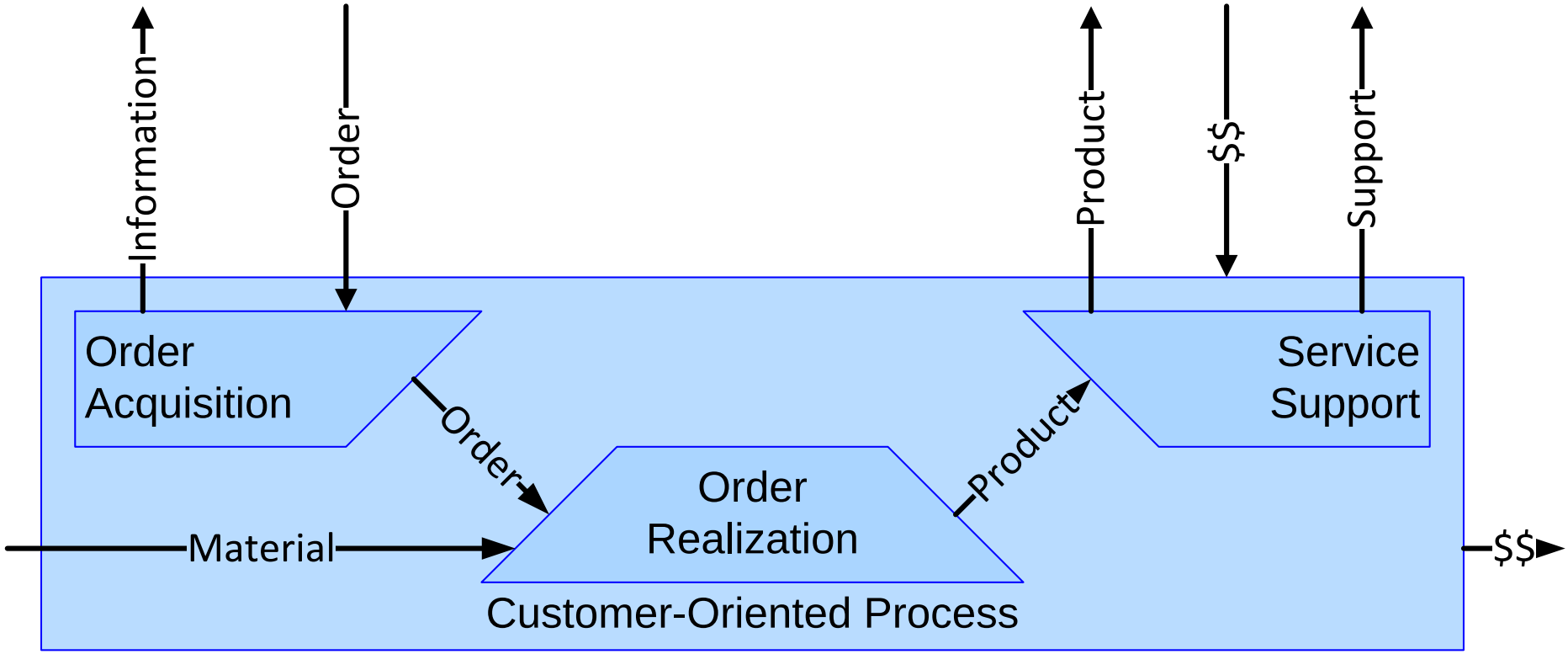
System Life Cycle



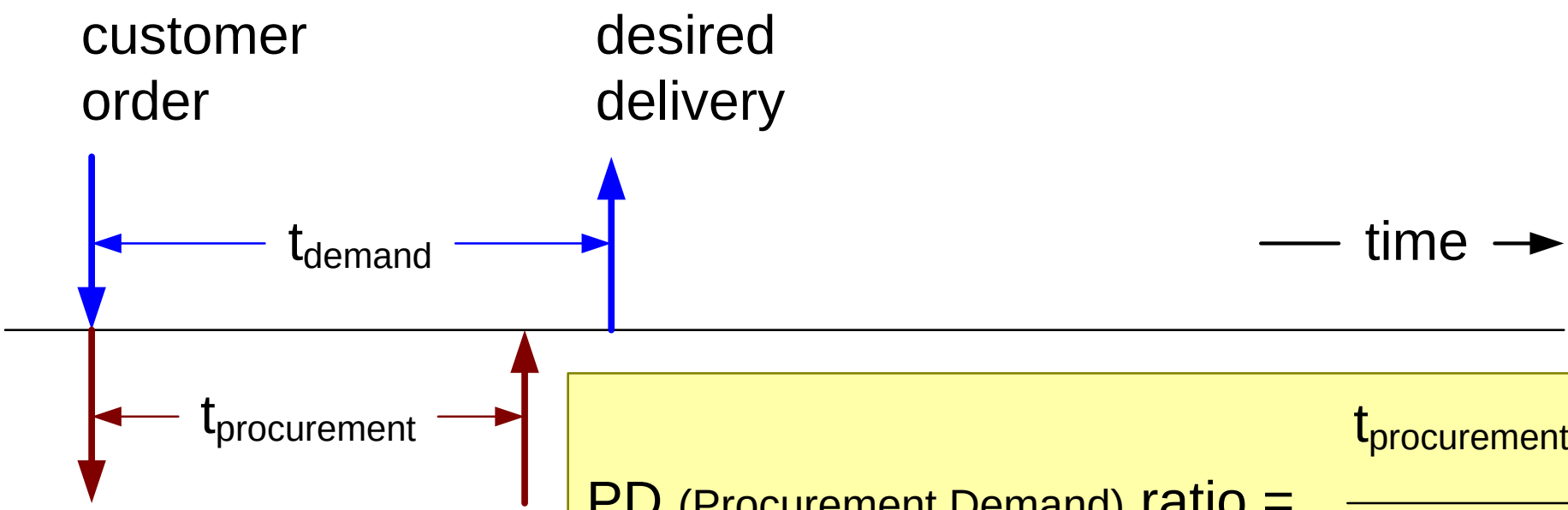
Creation Chain



Customer Oriented Process



Impact of Procurement Duration



PD (Procurement Demand) ratio = $\frac{t_{procurement}}{t_{demand}}$

if PD ratio<1 then
    build on order
else
    forecast based procurement

less robust