

# Less Heavy Systems Engineering; How Much is Appropriate?

by *Gerrit Muller* Buskerud University College

e-mail: `gerrit.muller@embeddedsystems.nl`

`www.gaudisite.nl`

## Abstract

Many companies are aware of opportunities to improve systems development, system integration and complex project execution. Conventional Systems Engineering from the military and aerospace domain, although perceived as useful, also tends to be seen as “heavy” in terms of process and artifacts. In this paper we explore alternative Systems Engineering approaches that are perceived as lighter. We also explore how much Systems Engineering is appropriate.

### 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.

June 20, 2011

status: preliminary

draft

version: 0

# At the Beginning of this Century

Spring 2000, preparing key-note for conference

Let's go for Light Weight Processes

You cannot be serious  
You do not want Light Weight Architecting

Oh yes, absolutely,  
Light Weight Architecting is what  
we need



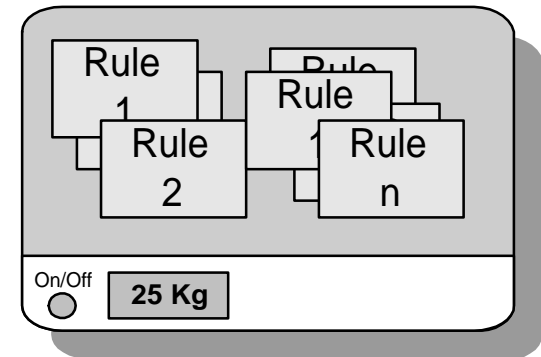
*Process  
Improvement  
Manager*



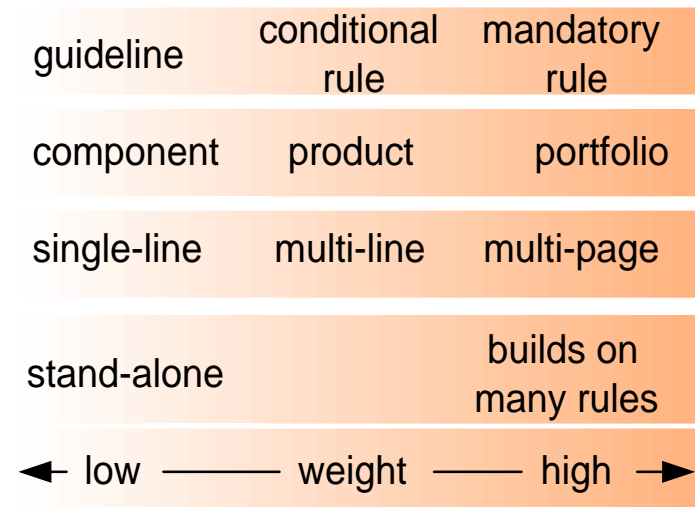
*Architect*

# Architecture Weight

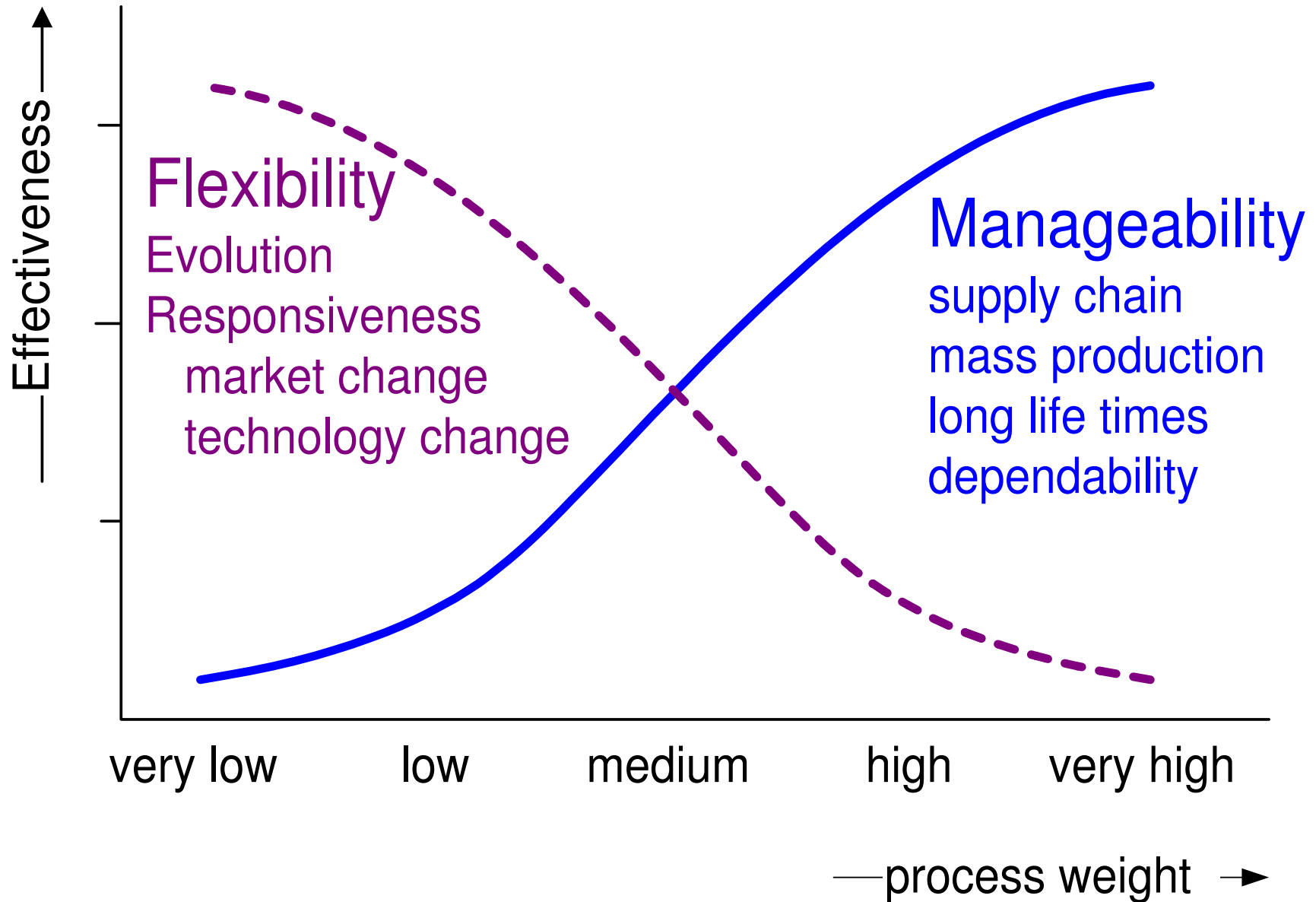
$$\text{weight}(\text{architecture}) = \sum_{\text{all rules}} \text{weight}(\text{rule})$$



$\text{weight}(\text{rule}) = f ($ 
 level of **enforcement** ,  
**scope** (impact) ,  
**size** ,  
 level of **coupling** or  
 number of dependencies  $)$

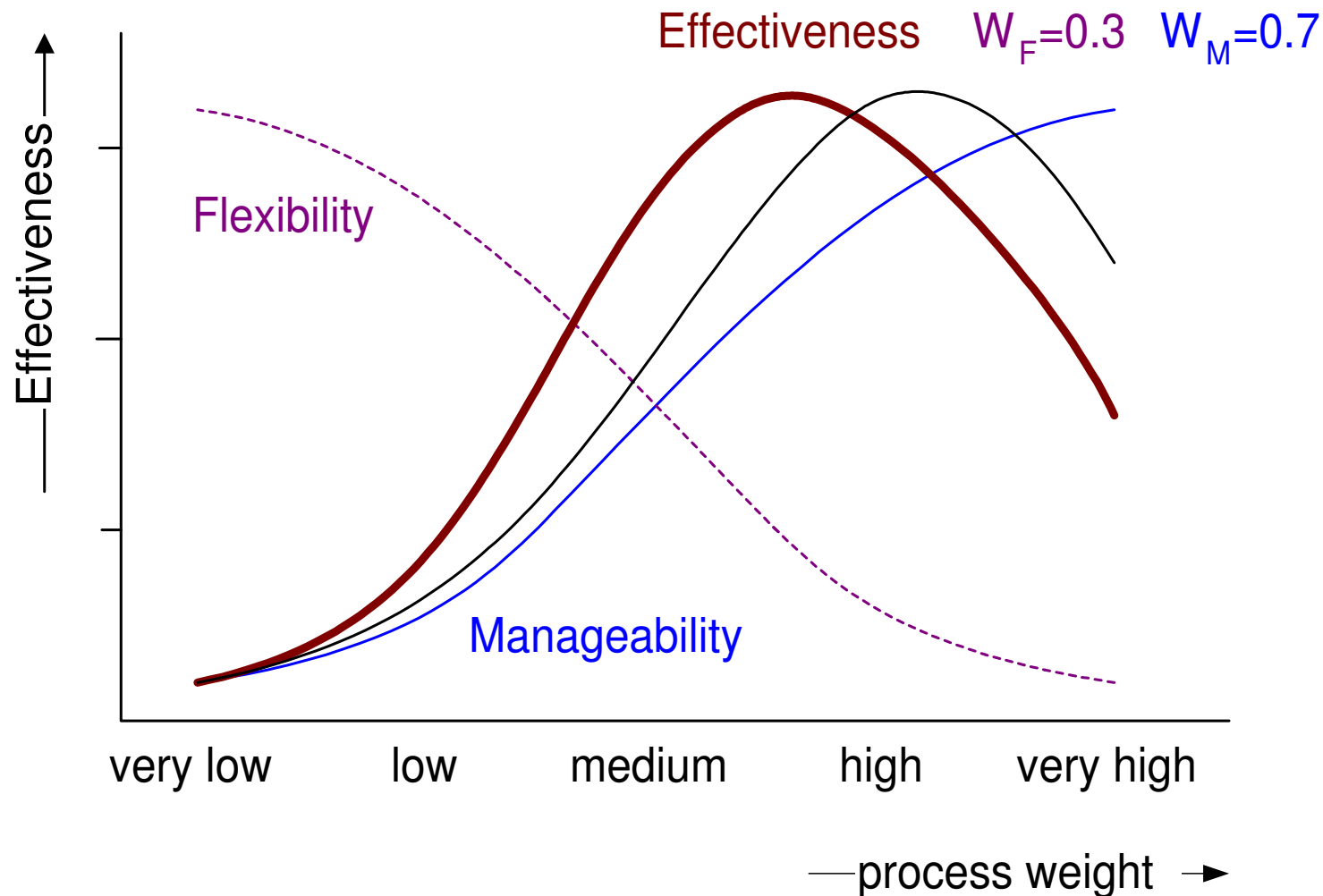


# Effectiveness(Flexibility, Manageability)

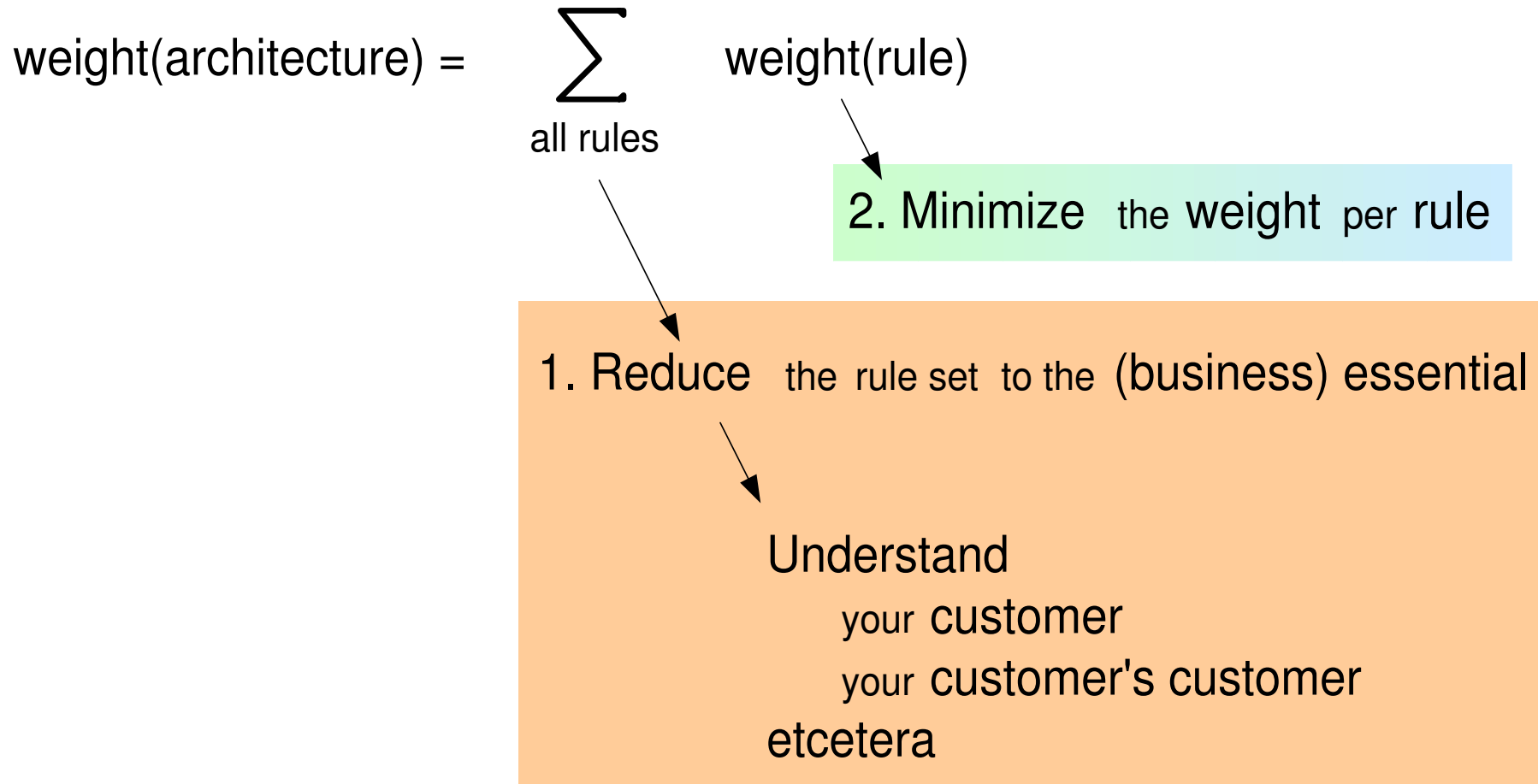


# Effectiveness

$$\text{Effectiveness} = \text{Flexibility}^{W_F} * \text{Manageability}^{W_M}$$



# Light Weight How To



# Minimize Rule Weight

weight(rule)=

f ( level of **enforcement** ,

**scope** (impact) ,

**size** ,

level of **coupling** or  
number of dependencies )

minimize number of mandatory rules

empower, delegate

minimize implementation details  
focus on essential concepts

Apply design principles on architecture

Multi-view architecting

# Simplified Framework

---

## *Effectiveness (Customer Value)*

*Do the right things*

What methods increase (understanding of) Customer Value?

What can you use in your own company to increase

(understanding of) Customer Value?

## *Efficiency (Effort, cost, and time per result)*

*Do things right*

What methods improve the efficiency of the company?

What can you use to improve the efficiency of your company?

# Work Form for KSEE 2011

	<i>Effectiveness (Customer Value)</i> <i>Do the right things</i> What can you use in your own company to increase (understanding of) Customer Value?	<i>Efficiency (Effort, cost, and time per result)</i> <i>Do things right</i> What can you use to improve the efficiency of your company?
<b>Håkan Gustavsson</b> Is it Lean or just common sense?		
<b>Einar Jørgensen</b> Globalising System Engineering and Lean Principles		
<b>Odd Guldsten</b> Complex power systems for offshore oil&gas topside installation		
<b>John Bjarne Bye</b> Lean Transformation		
<b>Jon Wade</b> Systems Engineering: At the Crossroads of Complexity		
<b>Andreas Thorvaldsen</b> Manufacturing Systems Modelling		
<b>Kristian Frøvd</b> Early Validation through the A3 method		
<b>Gerrit Muller</b> Less Heavy Systems Engineering; How Much is Appropriate?		

# Explanation of Work Form

---

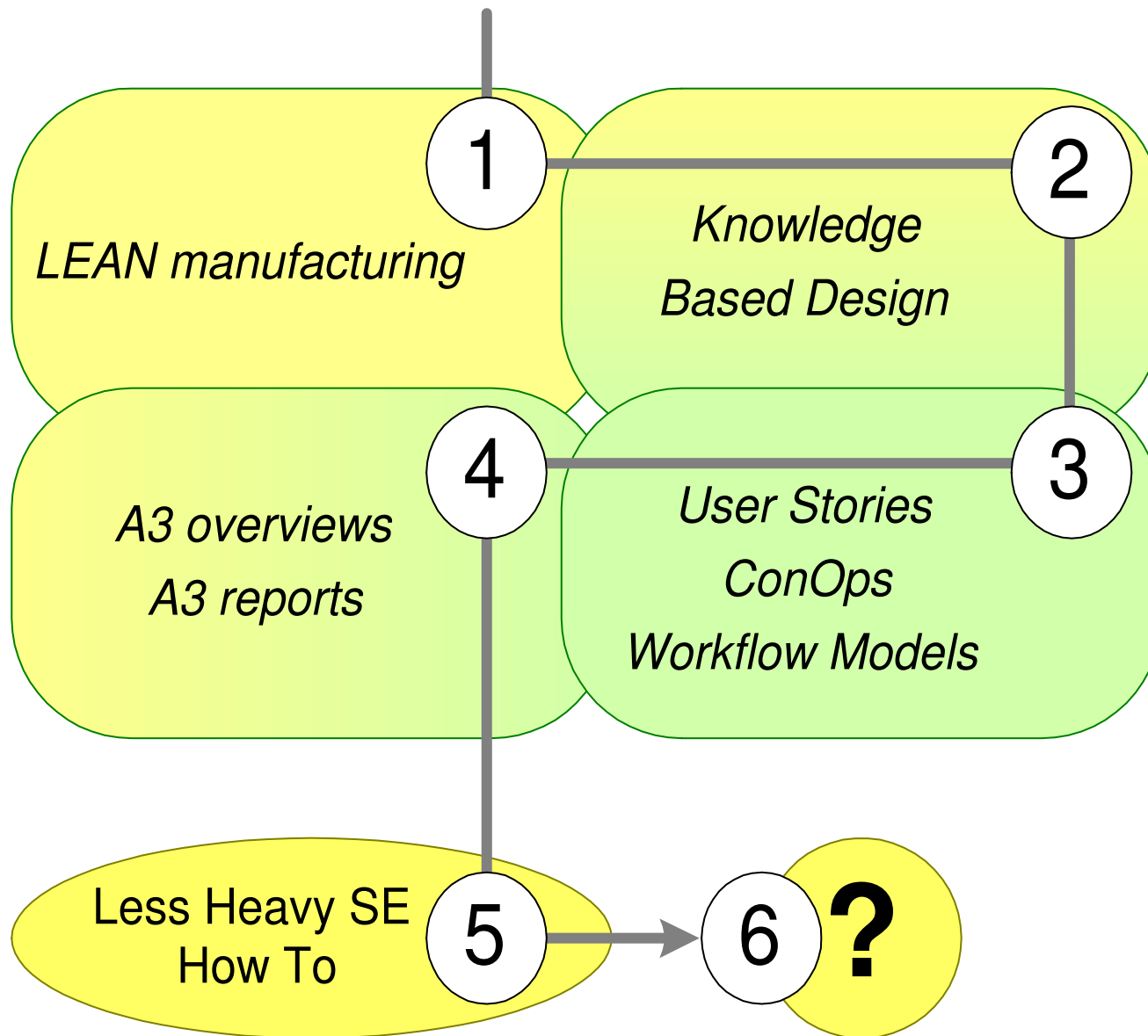
We expect that everyone fills in the form during or at the end of every presentation.

The purpose is to stimulate you to reflect on possible value for your own company.

We recommend to write down specific examples.

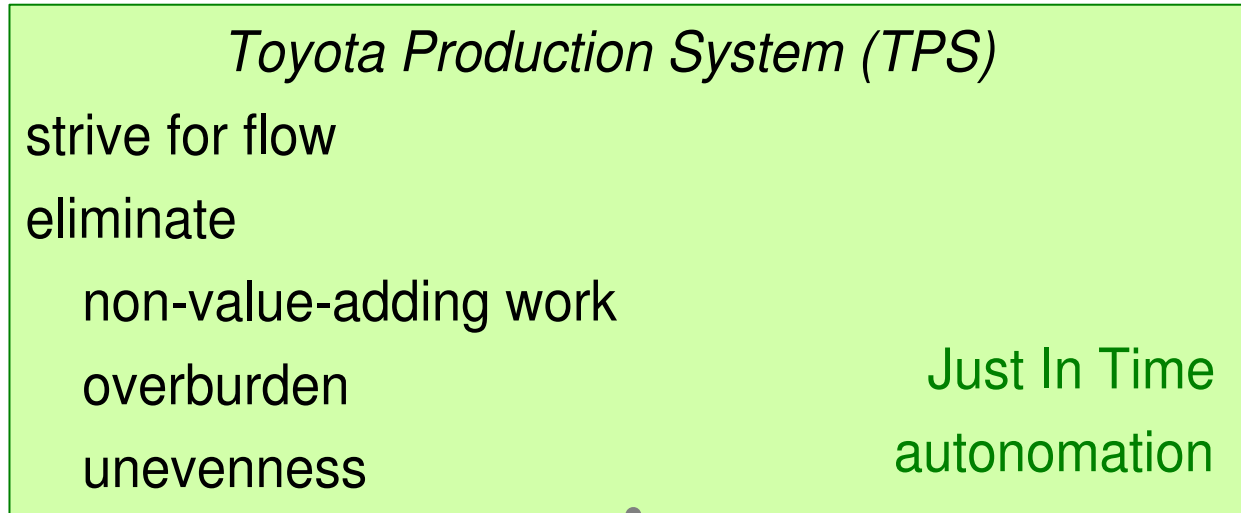
The last presentation will look back at all presentations.

# Time to Harvest! Figure Of Contents™

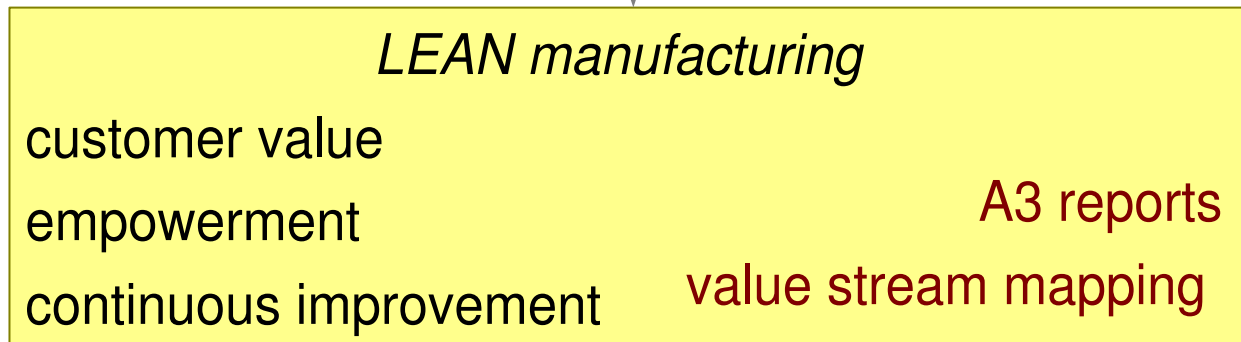


# LEAN Manufacturing

---

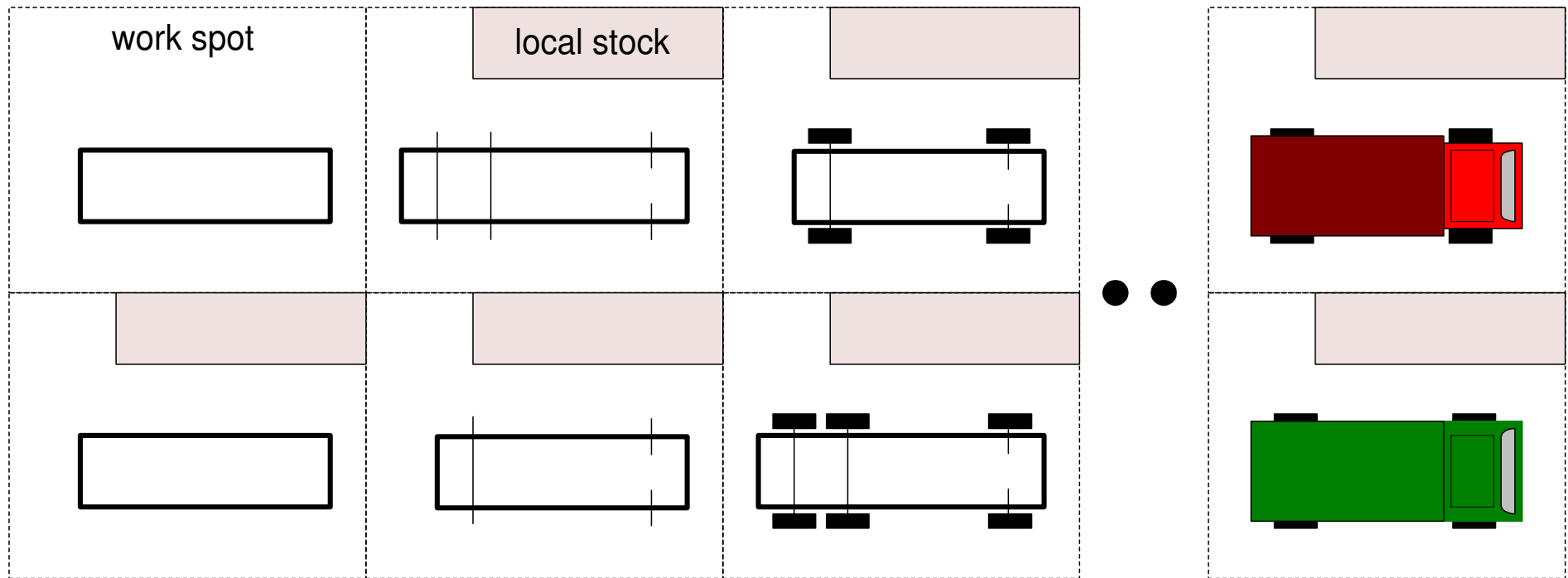


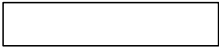
↓ evolved into



result: organic flow manufacturing  
efficient, flexible, short cycle times

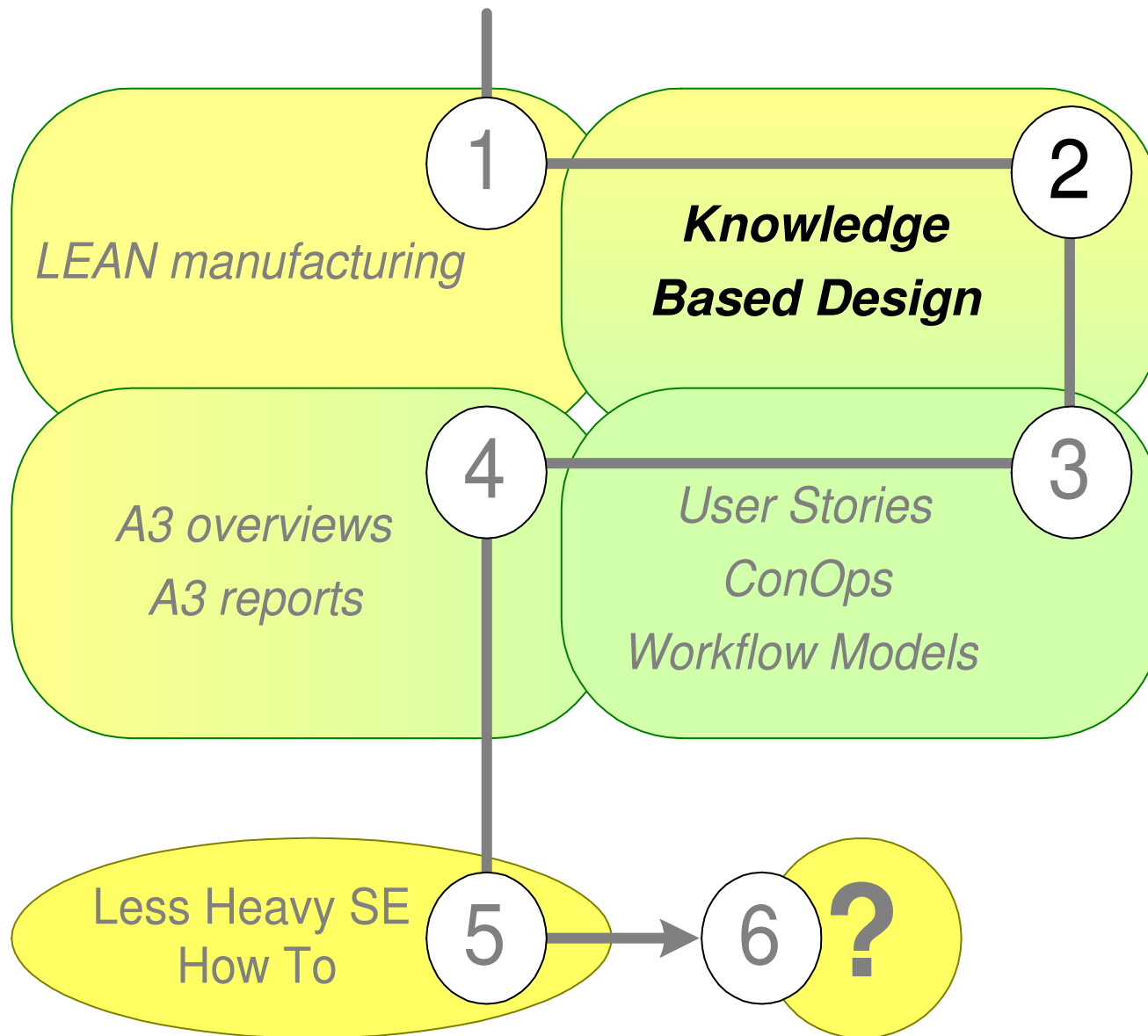
# Example of LEAN Manufacturing in Automotive



  
scheduling  
white board

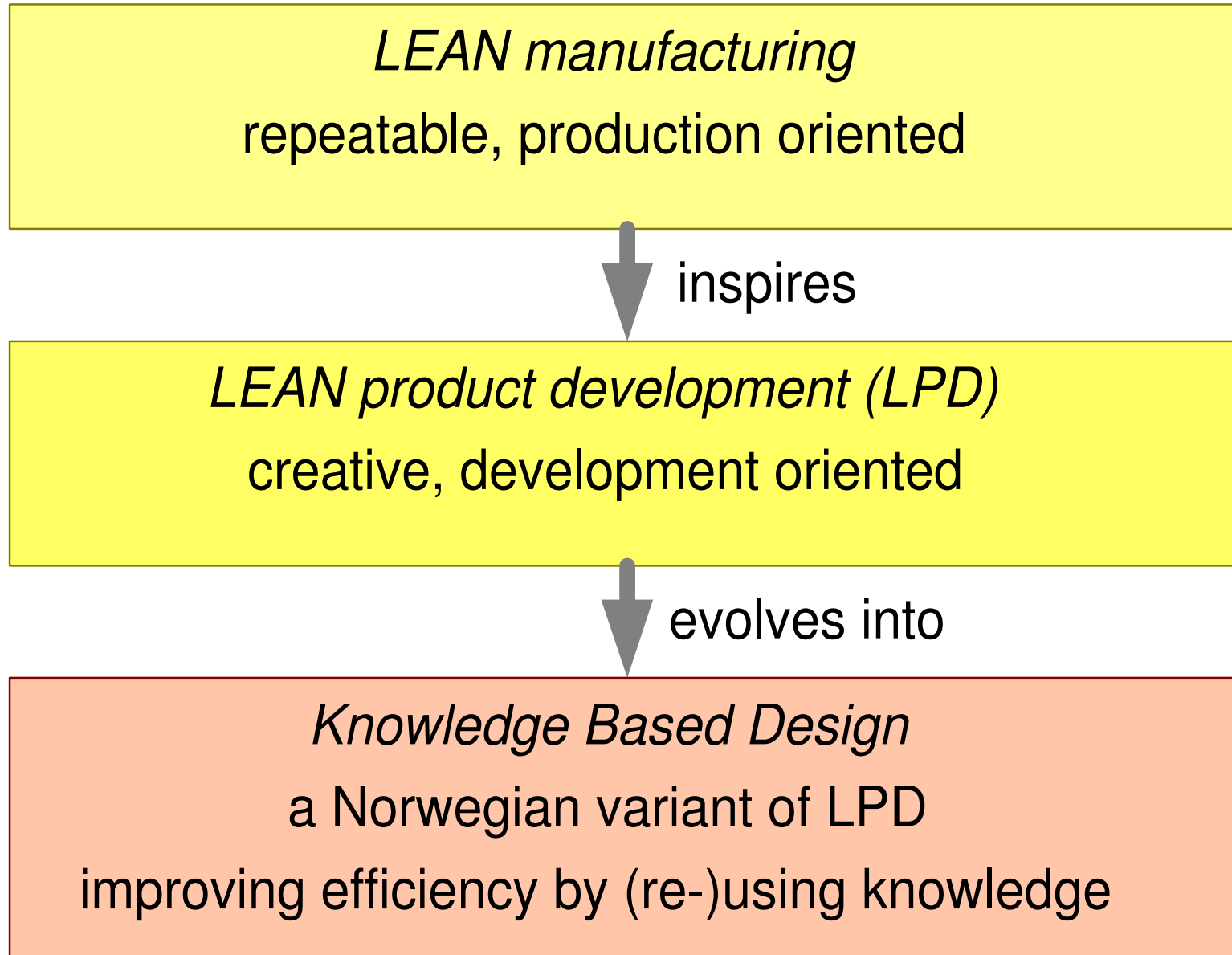
One heart beat  
Every truck is unique  
Local scheduling  
Many practical local solution  
by Continuous Improvement

# Knowledge Based Design

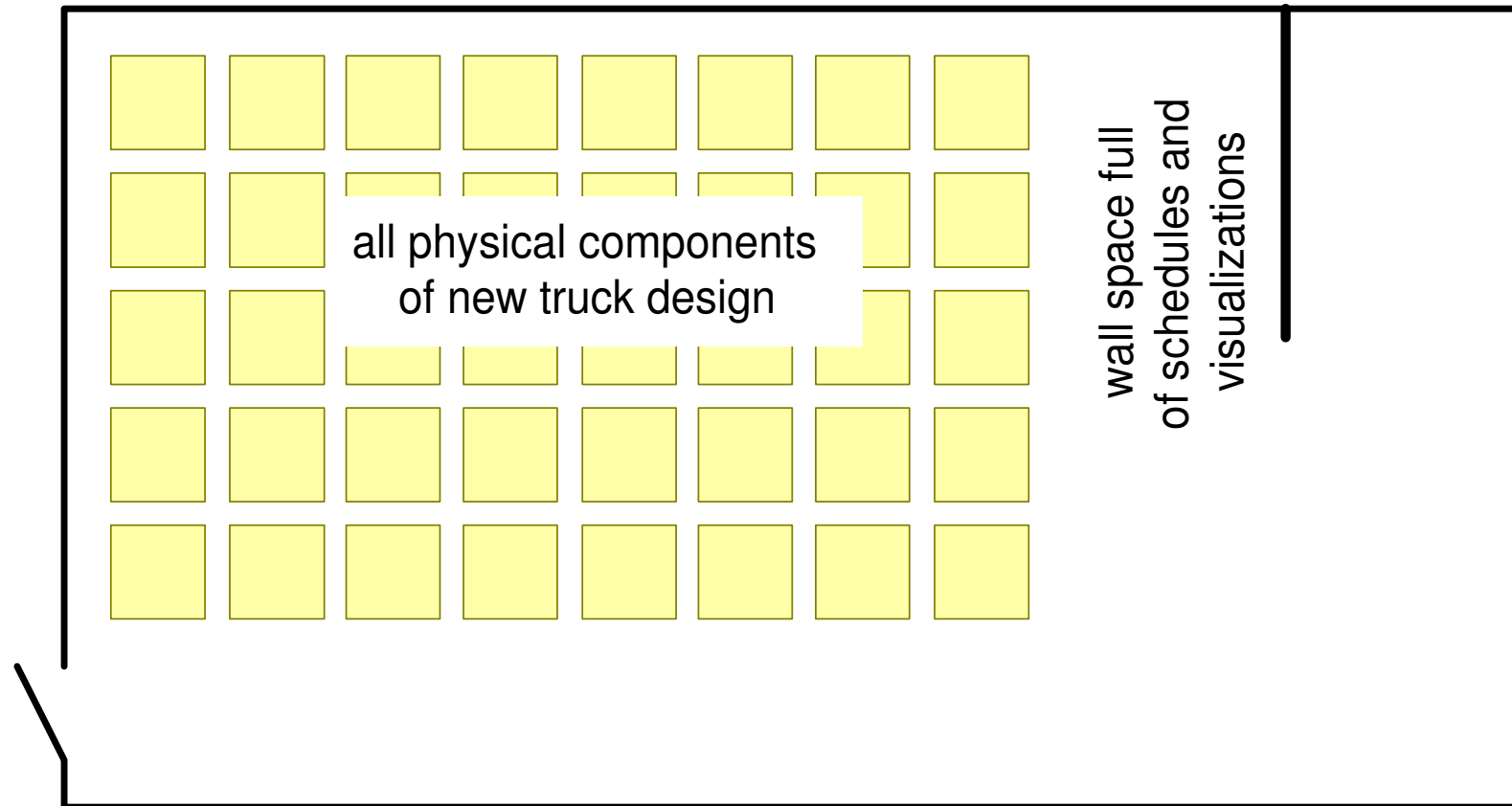


# Knowledge Based Design

---



# Example of LPD in Automotive



development supported by team location

tactile and visual support

developers drive trucks themselves (customer understanding)

# Reflections on Knowledge

---

Knowledge is abstract and intangible.

is data in a computer knowledge?  
are text and figures in a book knowledge?

Value is obtained when knowledge is applied properly.

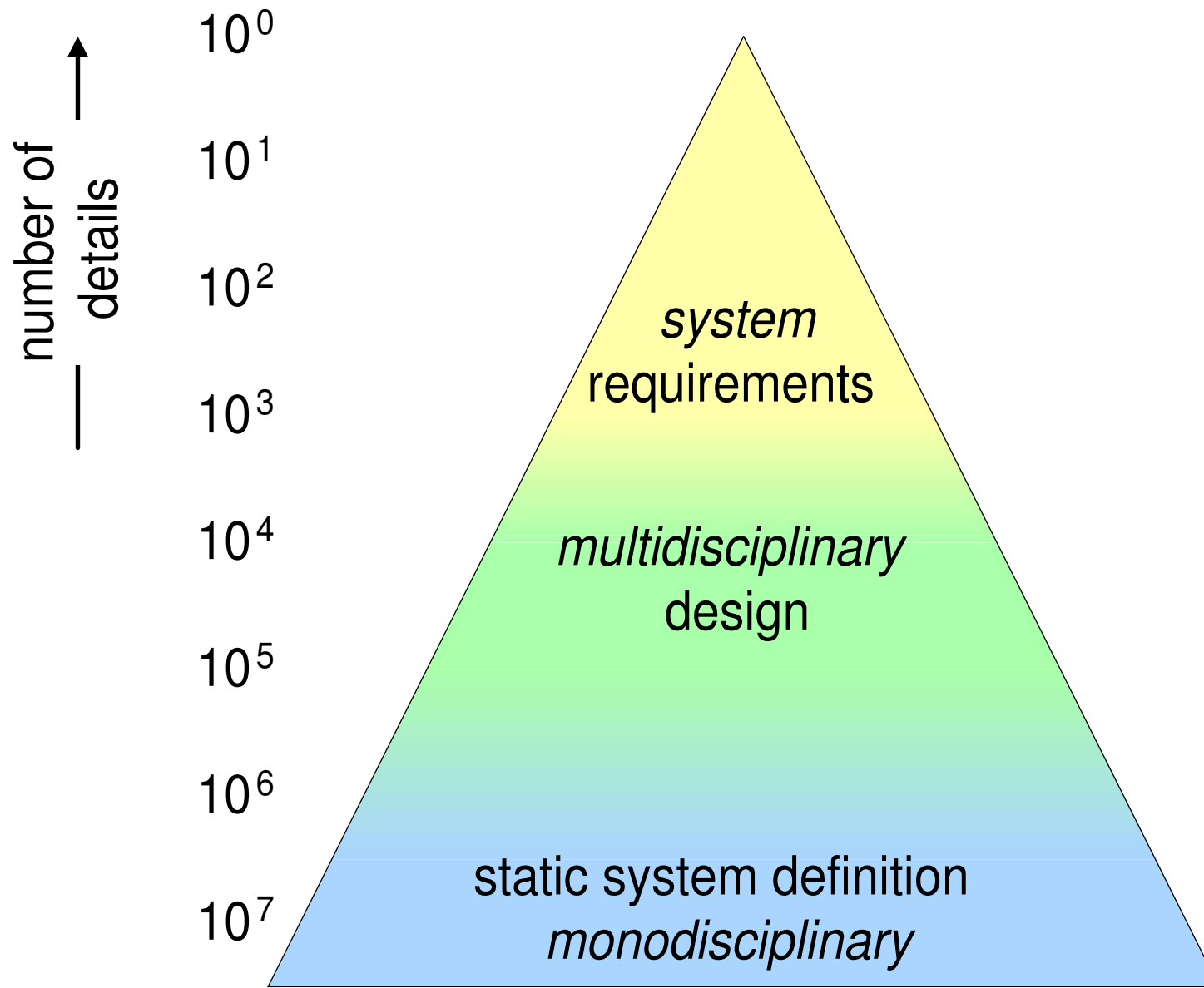
competence = knowledge + skills

Humans need experience to develop skills.

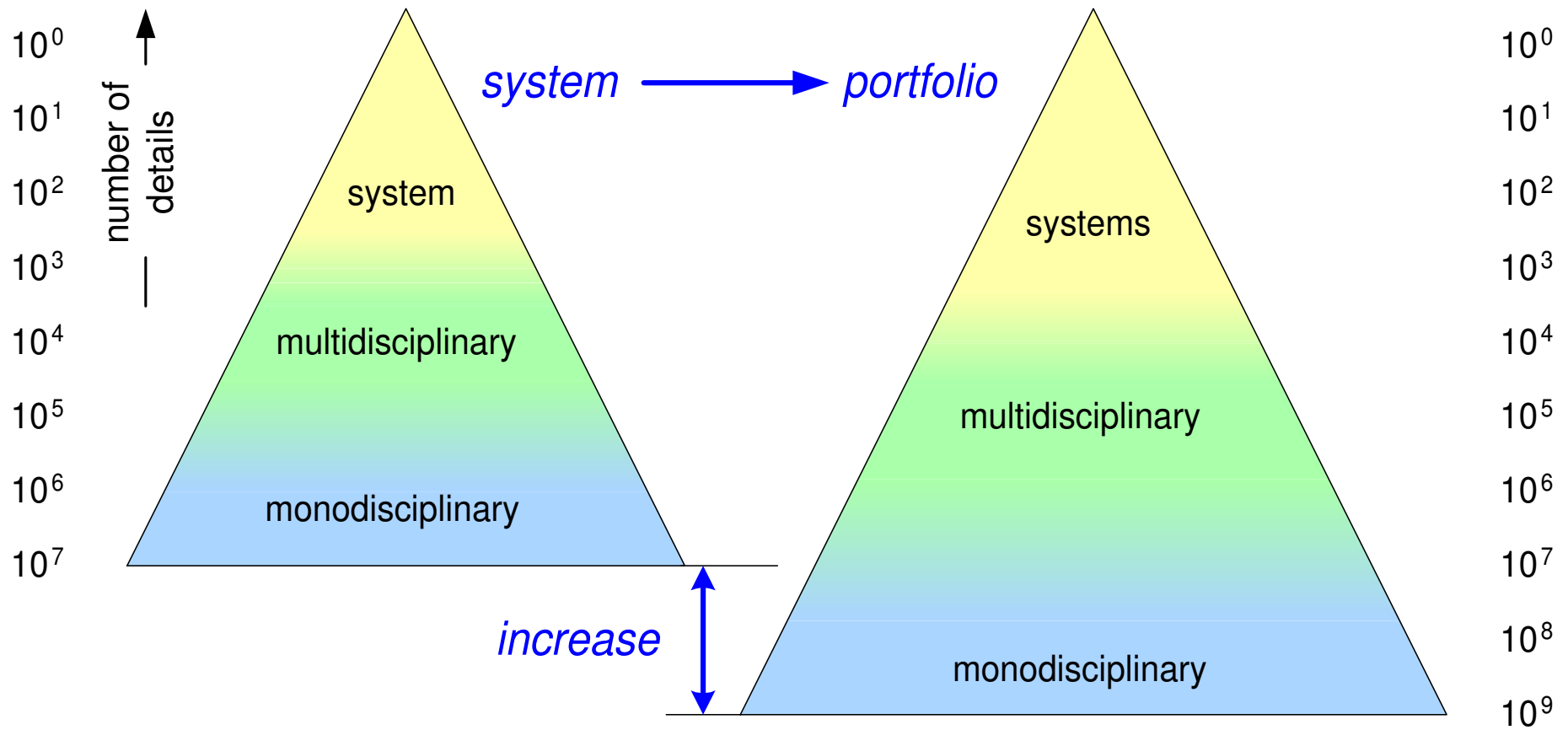
skills are practical, developed by doing

Skills and experience are complementary to knowledge.

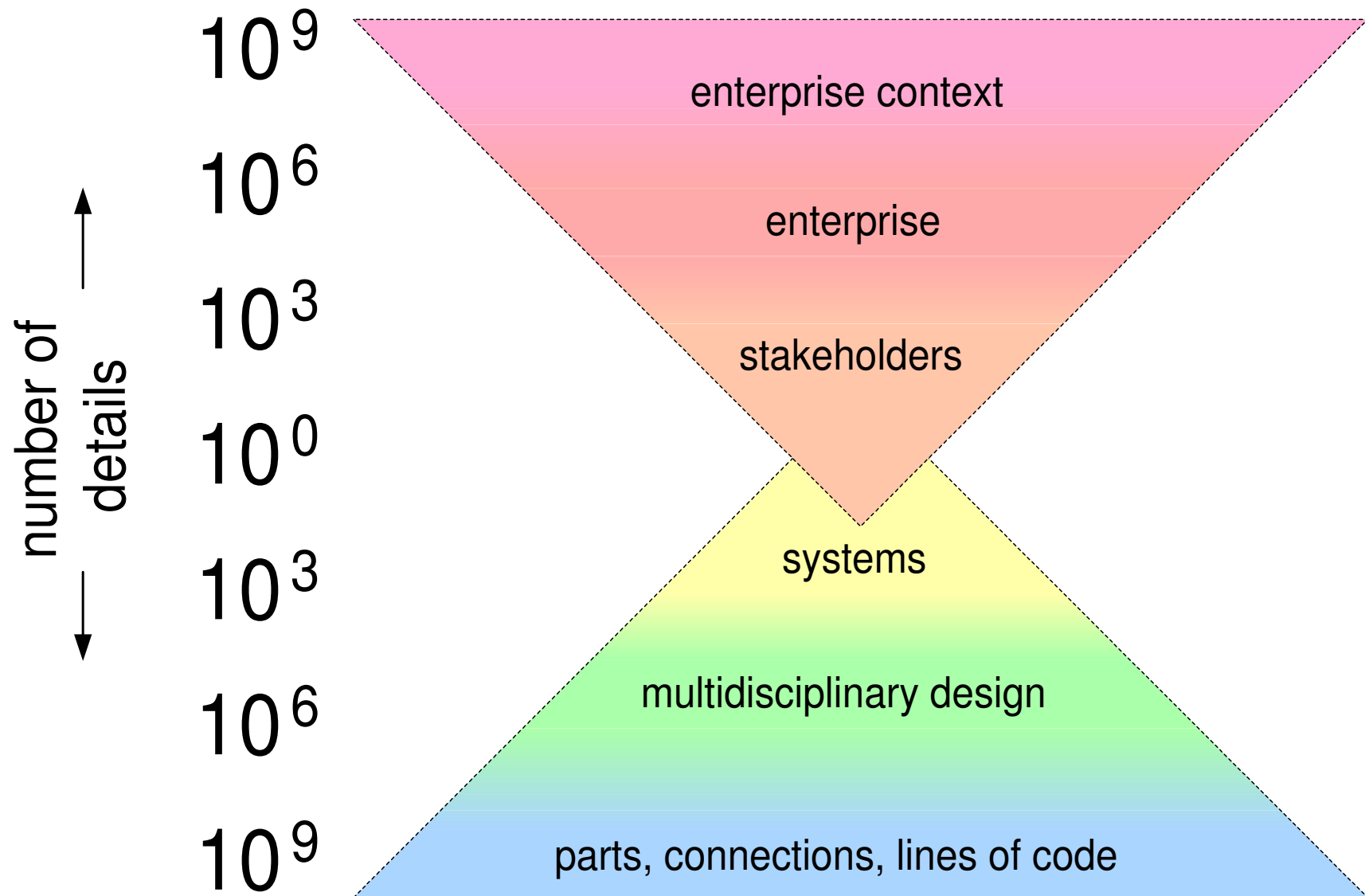
# Level of Abstraction Single System



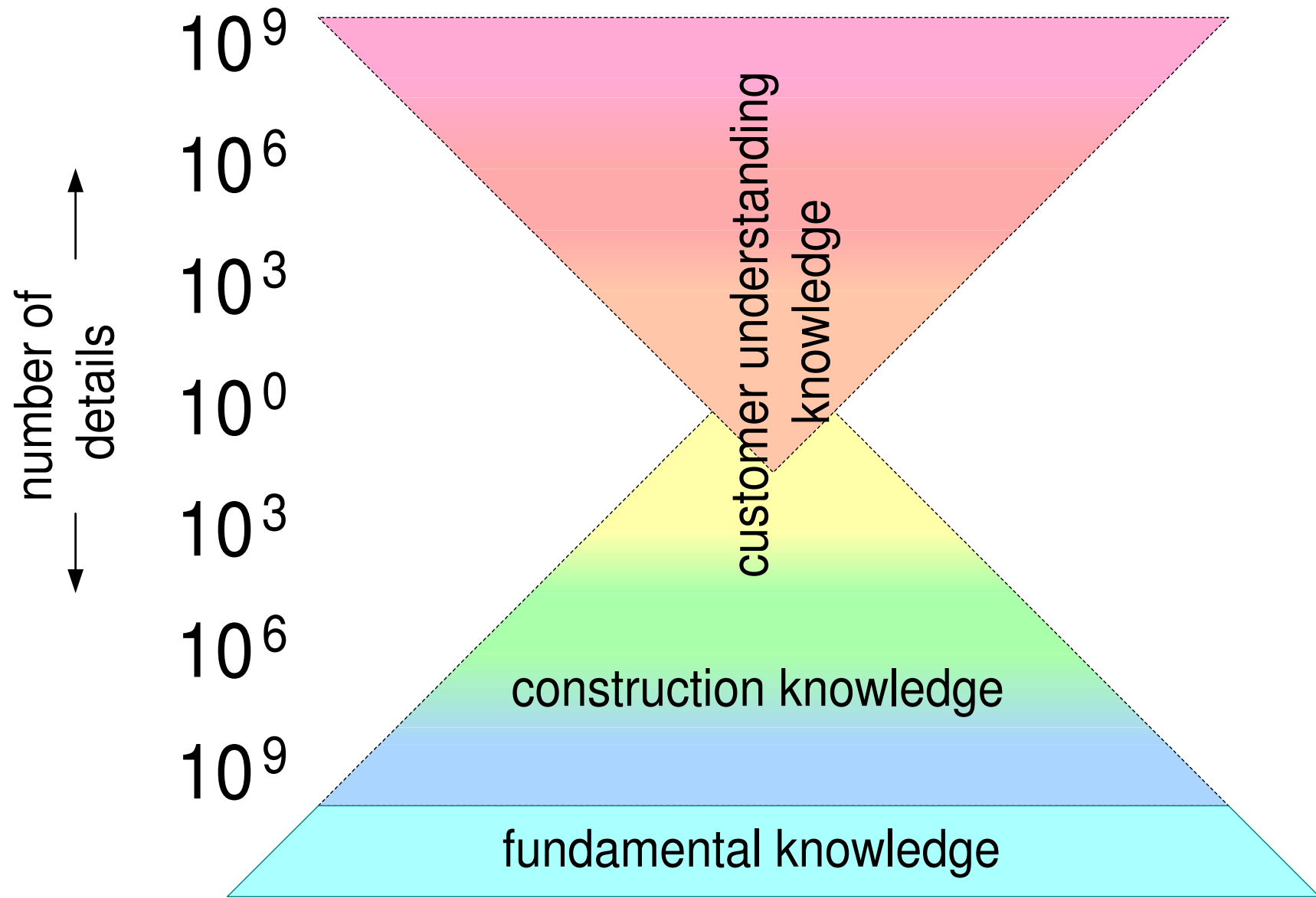
# From system to Product Family or Portfolio



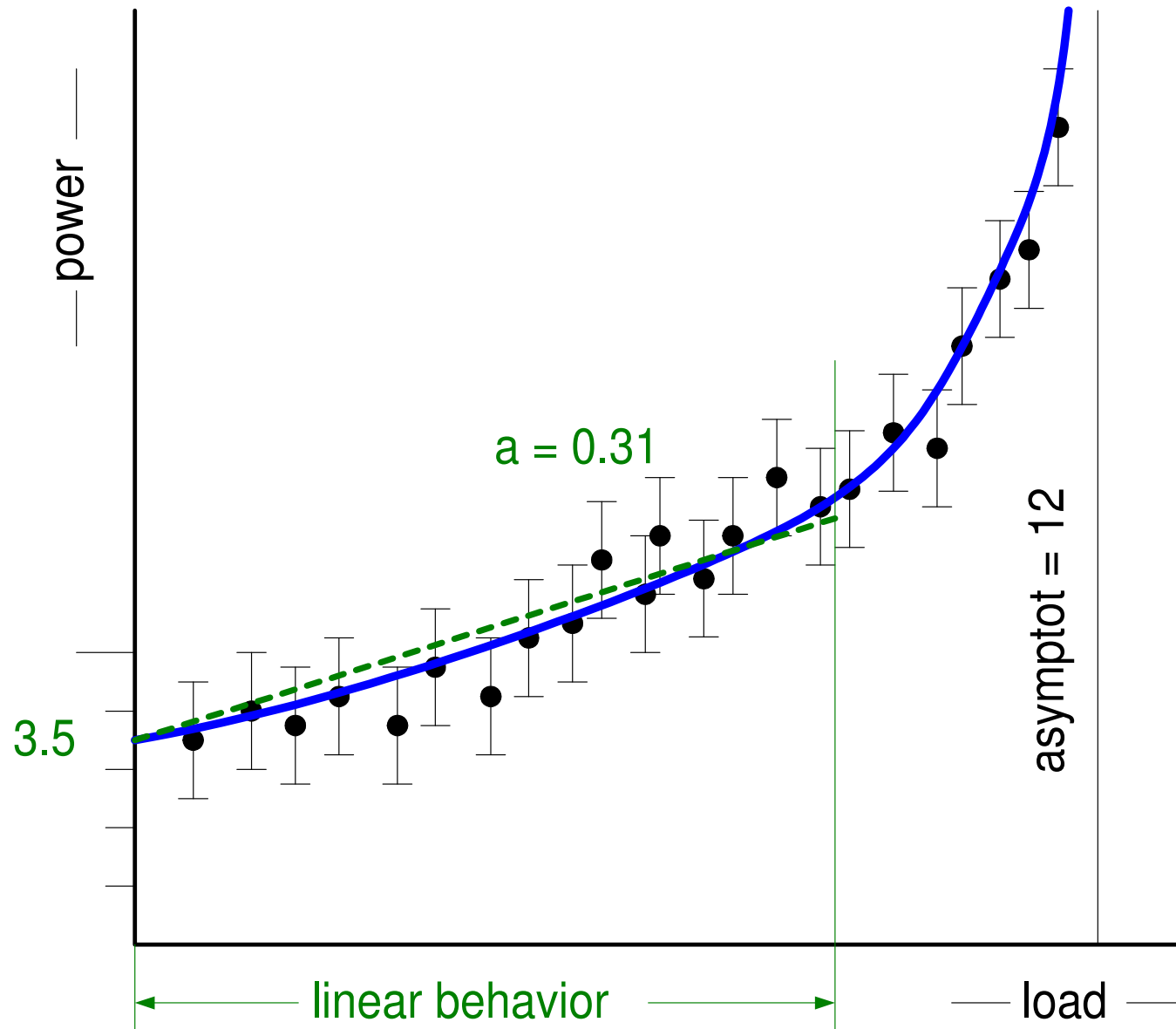
# Product Family in Context



# Knowledge at Multiple Levels



# Example of Fundamental Knowledge

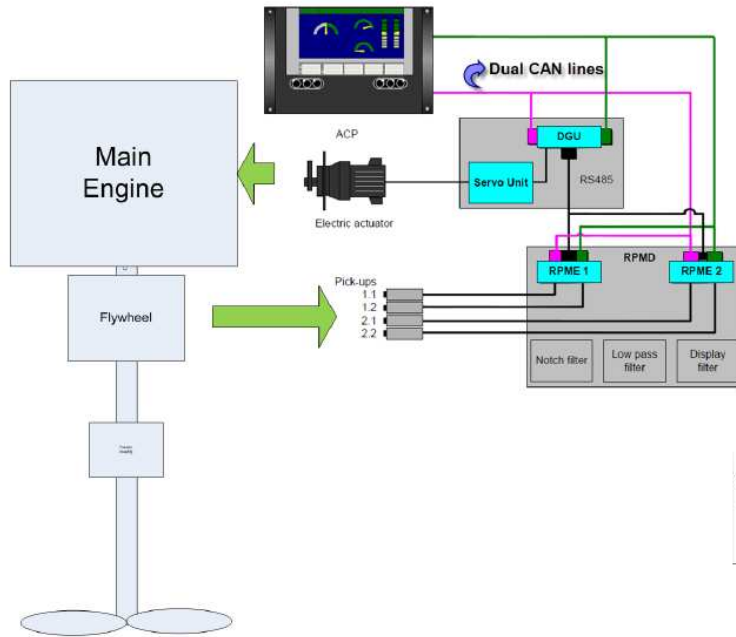


# Example of Construction Knowledge

## Digital Governor System



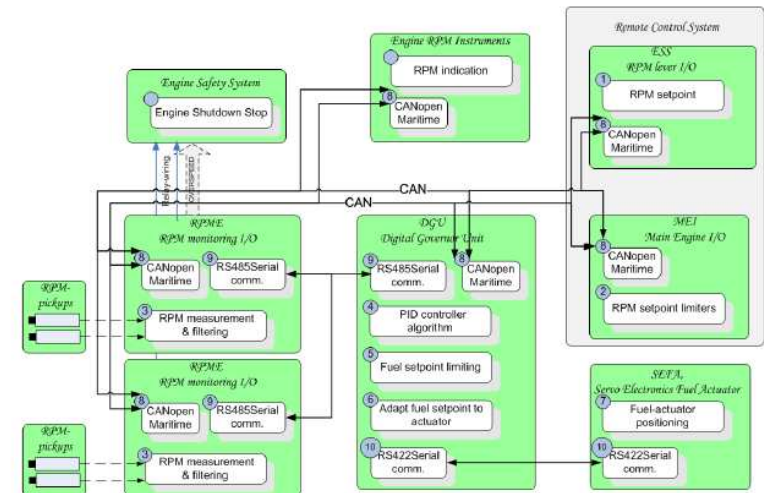
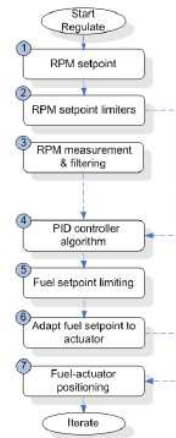
KONGSBERG



## Digital Governor System: Top Architecture Overview "Reverse Architecting"

March 2011

### Physical & Functional Overview / Function Allocation

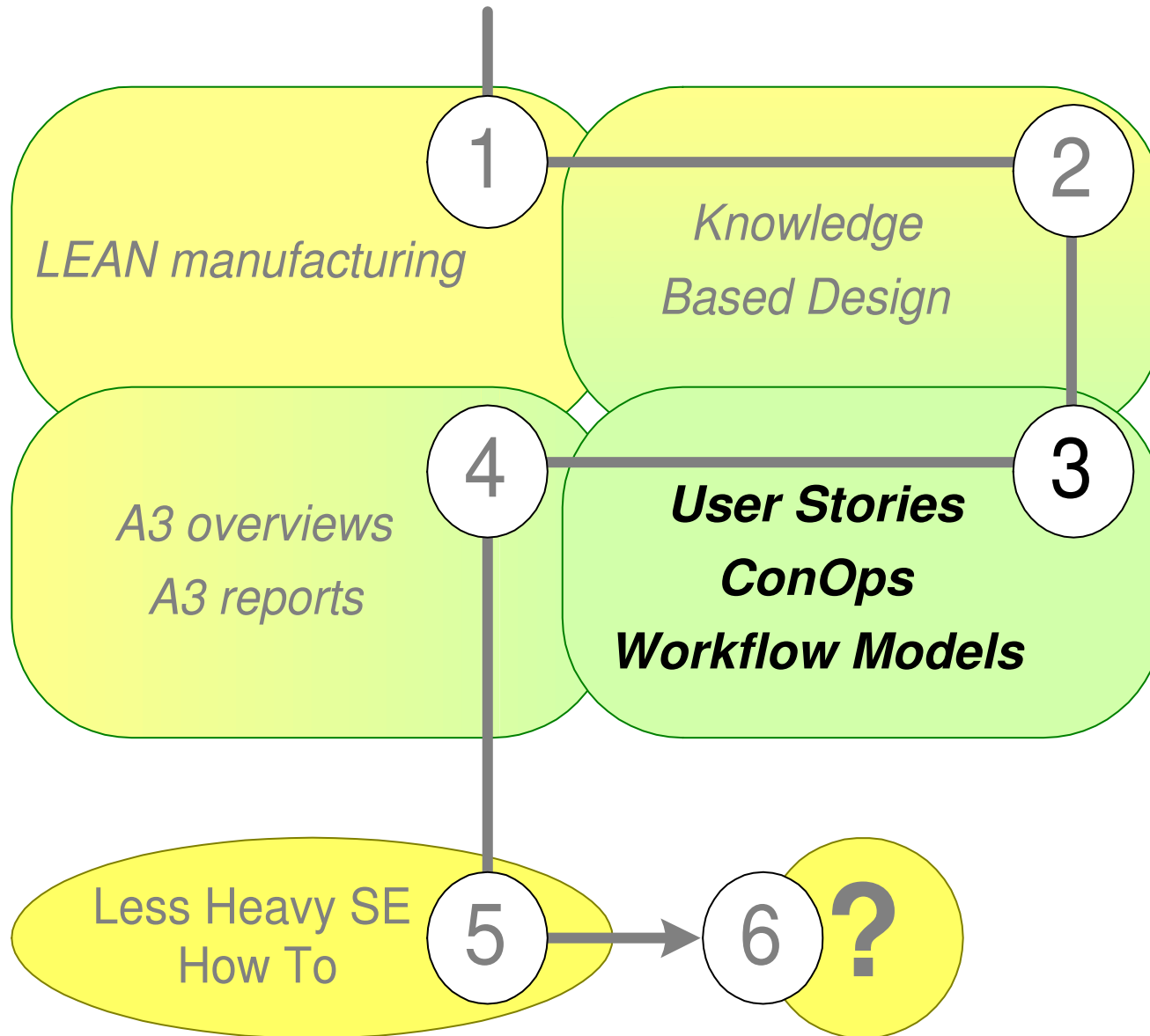


source: Bjørnar Wiulsrød  
SESG presentation  
March 2011

<http://www.gaudisite.nl/>

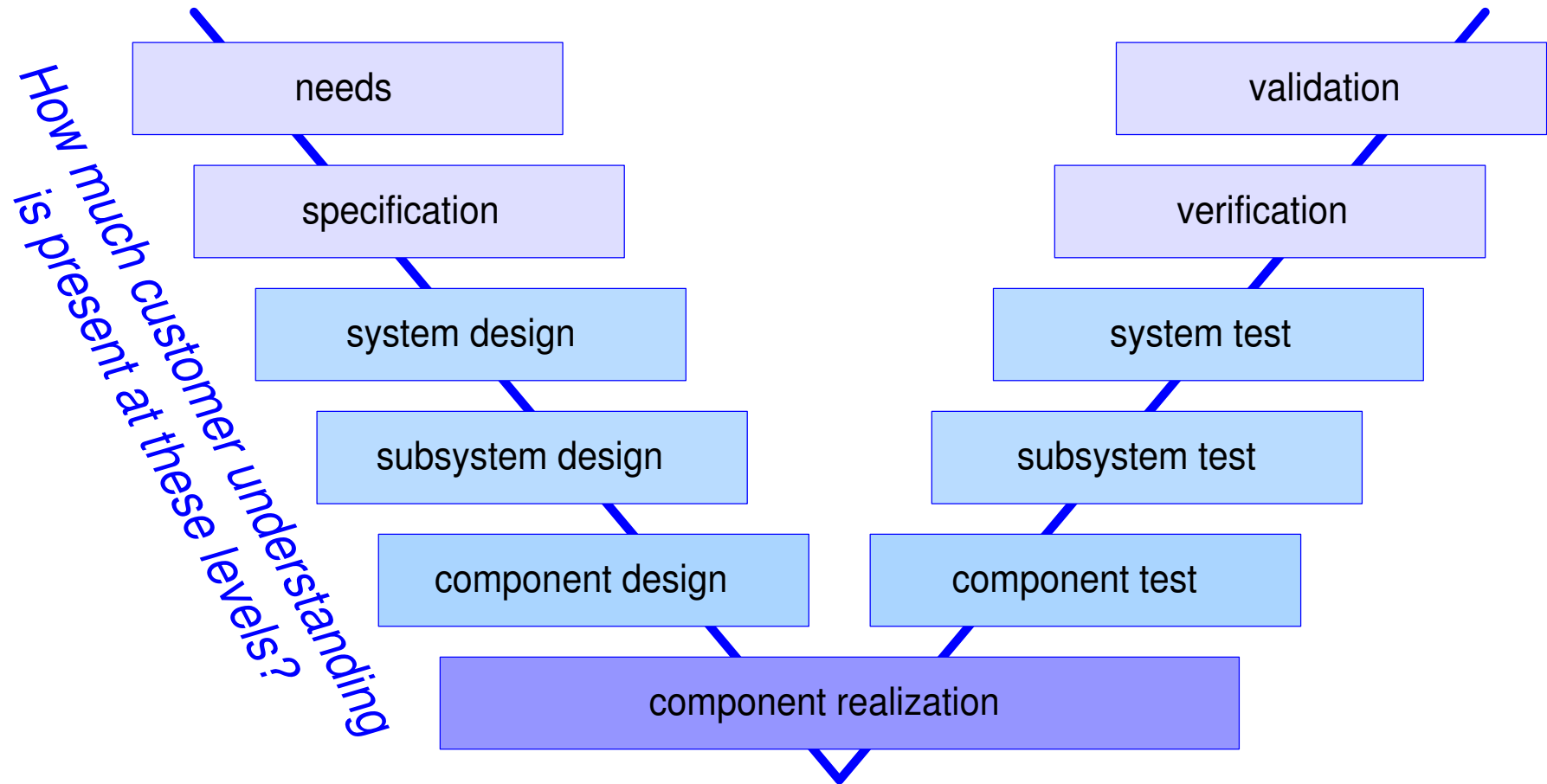
[SESG\\_Wiulsr%EF%BF%BDdA3overviews.pdf](http://www.gaudisite.nl/SESG_Wiulsr%EF%BF%BDdA3overviews.pdf)

# Customer Understanding



# How well do Your Engineers Understand Your Customer?

*In every hand-over and every conversion knowledge is lost*



# Methods to Capture Customer Understanding

---

## *User Stories*

Specific stories to explore specification and design.

Contain social and environmental details to make engineers aware

## *ConOps*

*Concept of Operations*, used in Defense Domain

Factual description of Operational use, a.o. with scenarios

## *Work Flows*

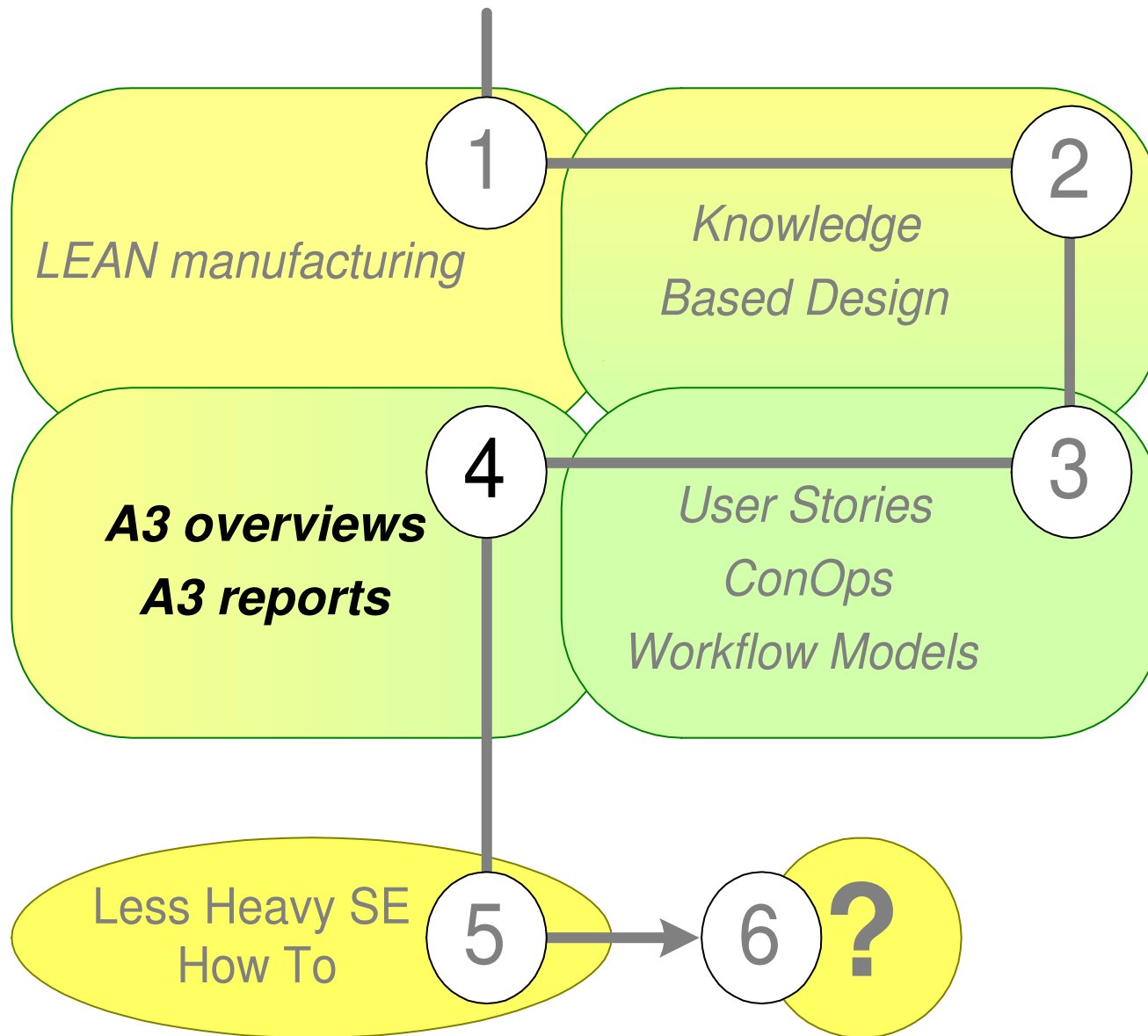
Systematic description of user operations.

Annotated with Where, When, Who, What

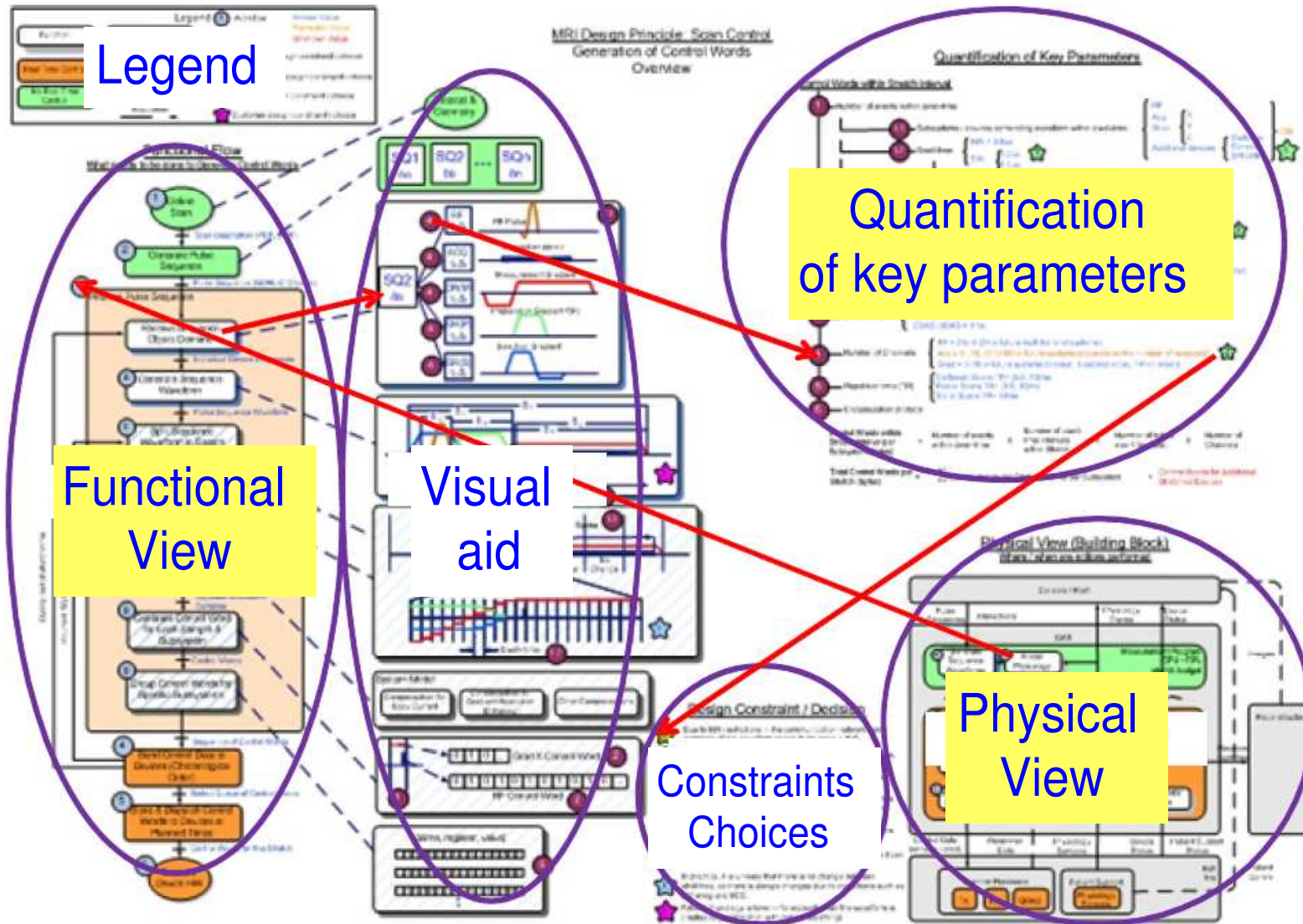
*This is one class of methods, there are many more methods*



# A3 Overviews and A3 Reports



# A3 Overview Fundamentals



A3 Architecture Overviews Focusing architectural knowledge to support evolution of complex systems  
by: Daniel Borches and Maarten Bonnema, INCOSE 2010

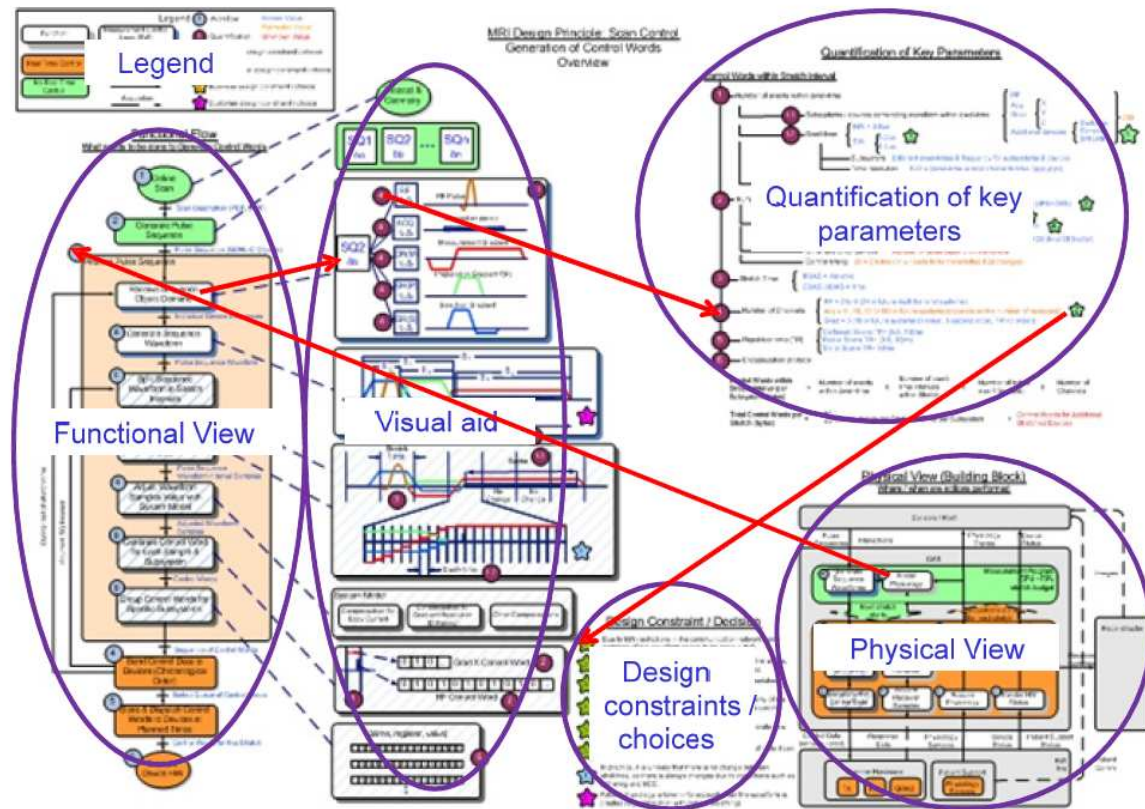
# A3 Overview Fundamentals (2)

multiple related views

quantifications

one topic  
per A3

capture  
"hot" topics



source: PhD thesis Daniel Borches <http://doc.utwente.nl/75284/>

digestible  
(size limitation)

practical  
close to stakeholder experience

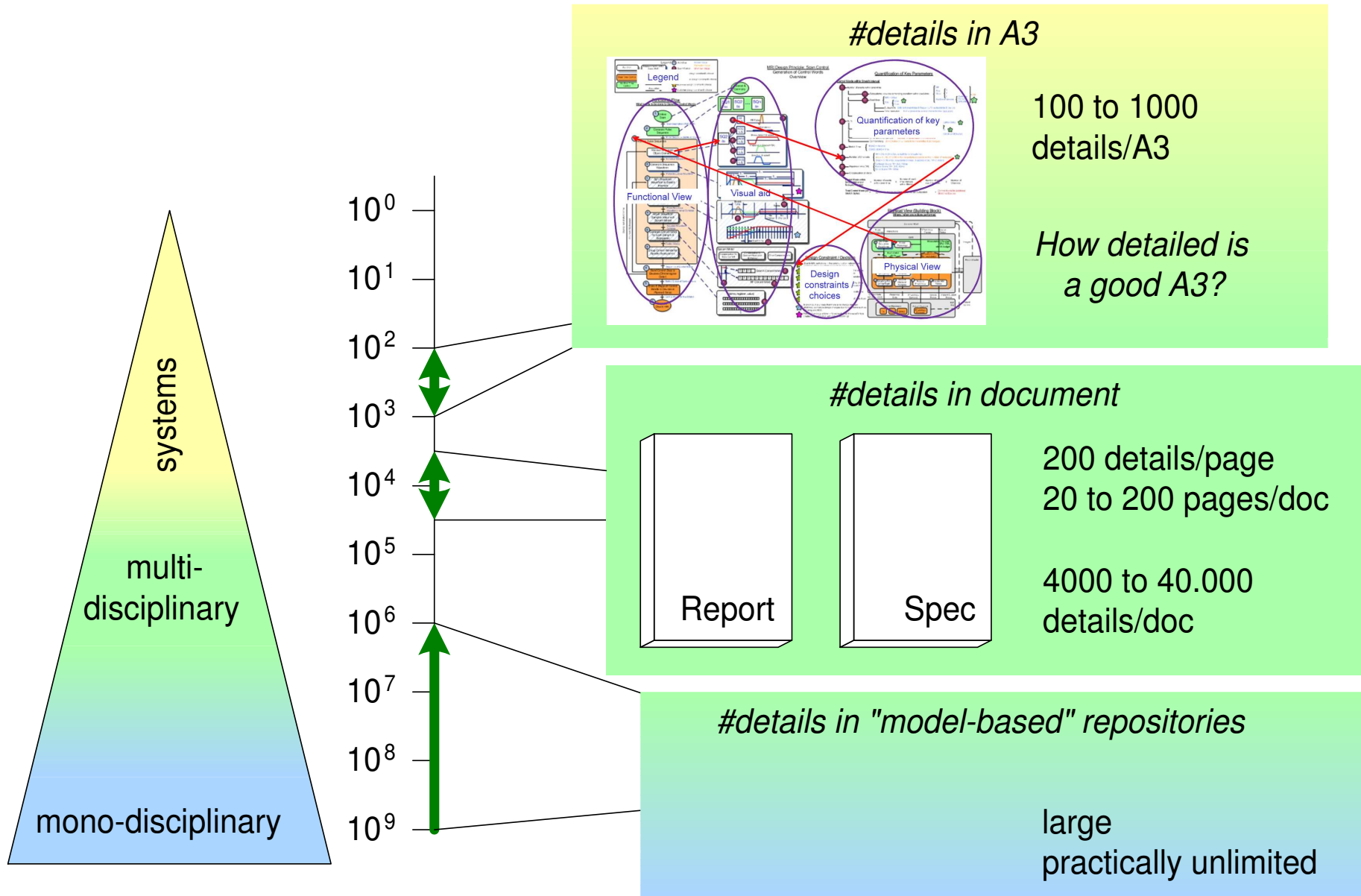
# Evaluation of Conventional Design Spec

## Results of Questionnaire System Design Specification

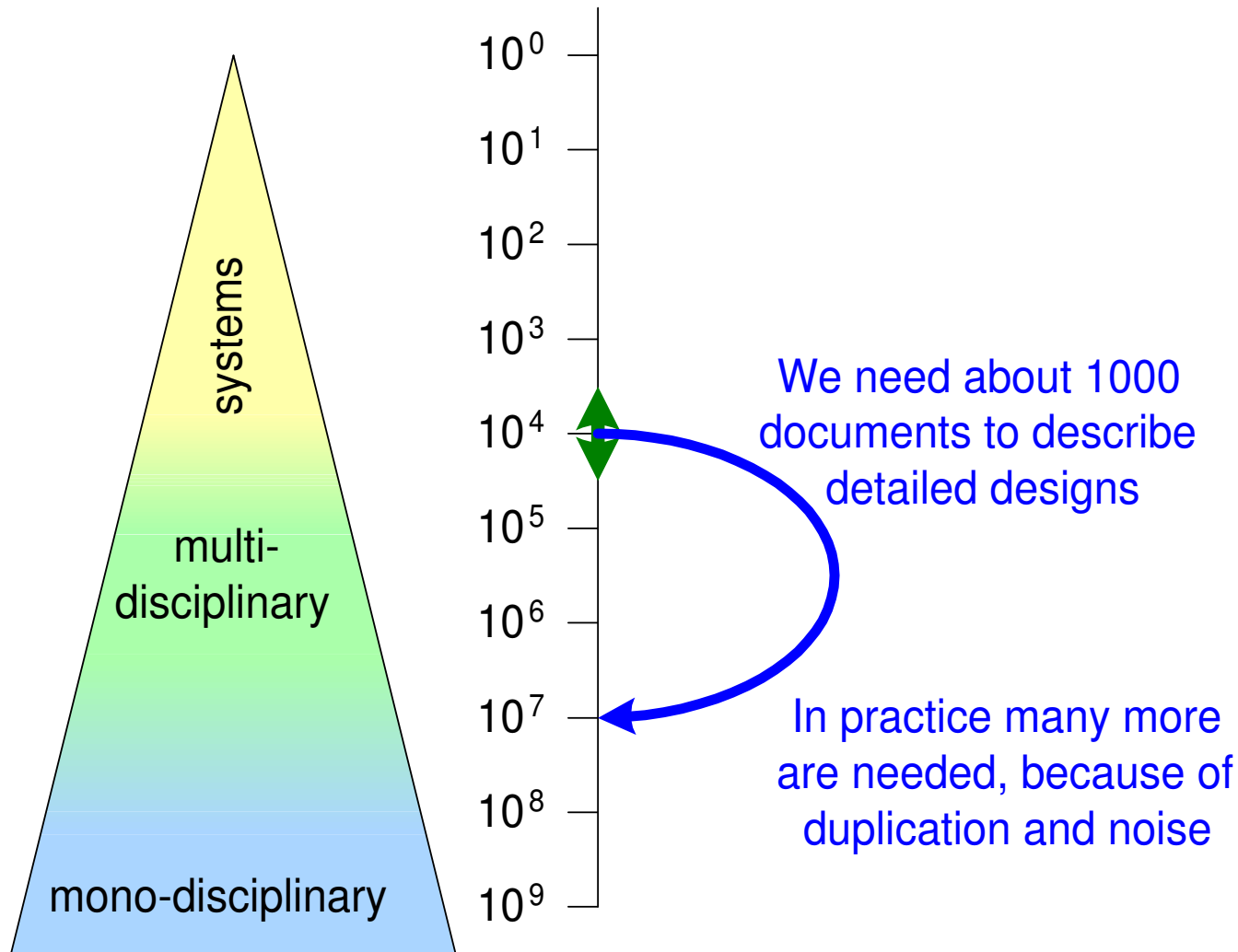
Statement 4: Current SDS document is useful for your work					
General Response		Strongly Agree/Agree per Job Title		Strongly Agree/Agree per Experience	
Strongly Agree	0%	Manager/Leader	50%	<5 Years	75%
Agree	29%	Architect	40%	5 <Years< 10	23%
Disagree	40%	Engineer	30%	10 <Years< 20	22%
Strongly Disagree	14%	Designer	0%	Since MR Proton	22%
Don't Know	17%	Domain Expert	50%	(> 20 Years)	
		Other	0%		
Statement 5: The SDS delivers what you expect from a system specification					
General Response		Strongly Agree/Agree per Job Title		Strongly Agree/Agree per Experience	
Strongly Agree	0%	Manager/Leader	25%	<5 Years	50%
Agree	26%	Architect	20%	5 <Years< 10	31%
Disagree	49%	Engineer	40%	10 <Years< 20	11%
Strongly Disagree	6%	Designer	0%	Since MR Proton	22%
Don't Know	20%	Domain Expert	50%	(> 20 Years)	
		Other	33%		

Source: PhD thesis Daniel Borches <<http://doc.utwente.nl/75284/>>

# Amount of Data per Medium



# Number of Conventional Documents

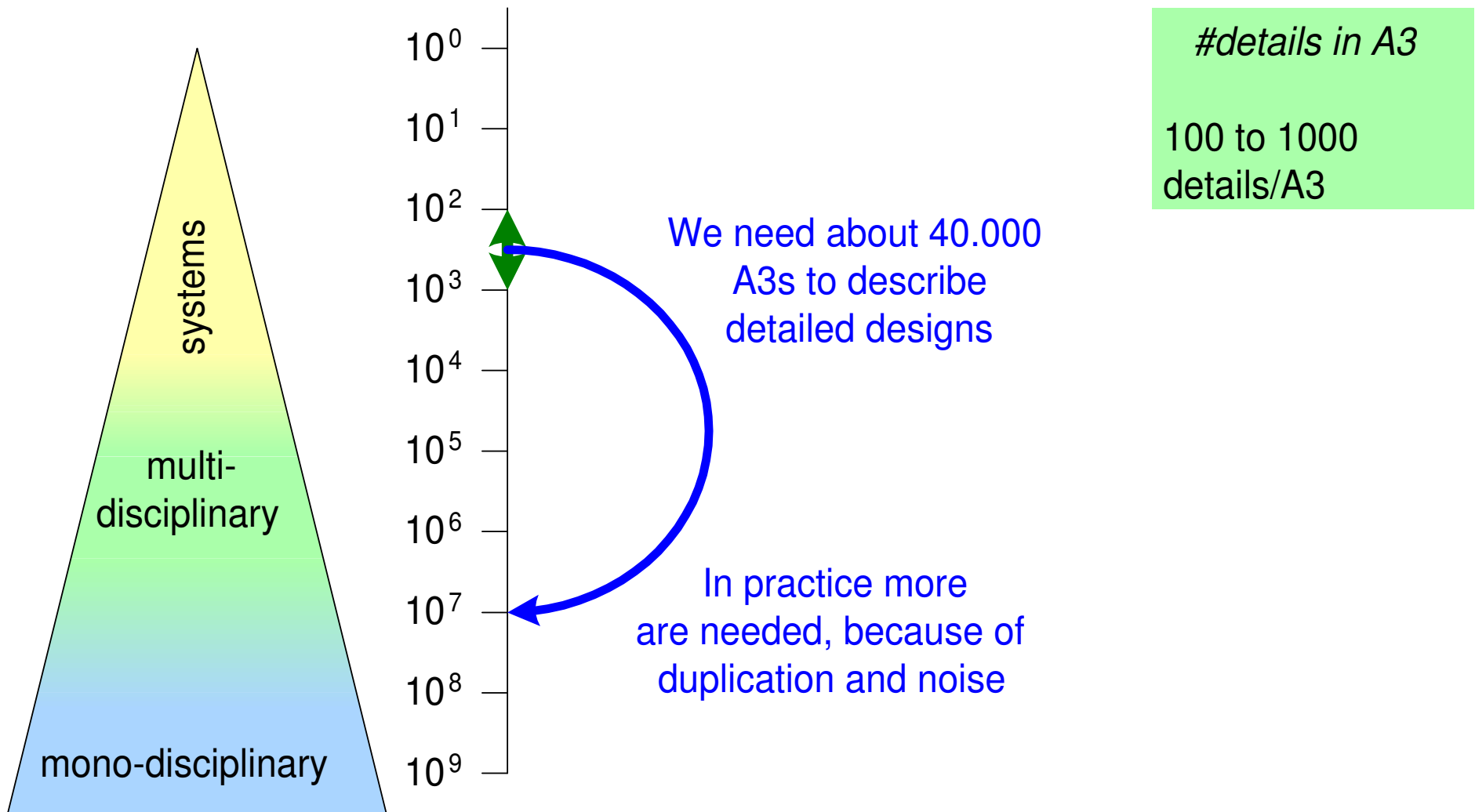


*#details in document*

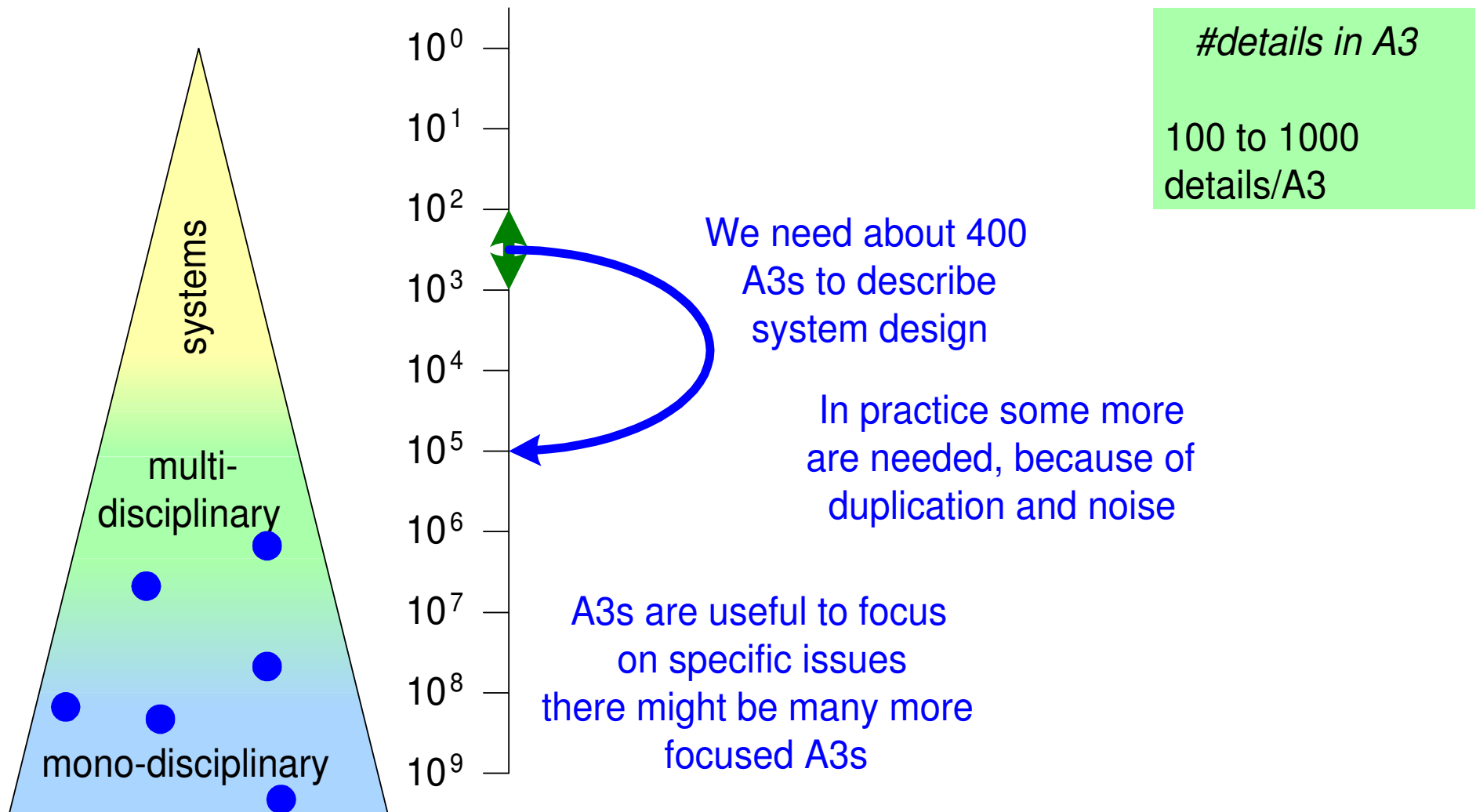
200 details/page  
20 to 200 pages/doc

4000 to 40.000  
details/doc

# What If we Use A3s for all Detailed Designs?



# What If we Use A3s for System Design?



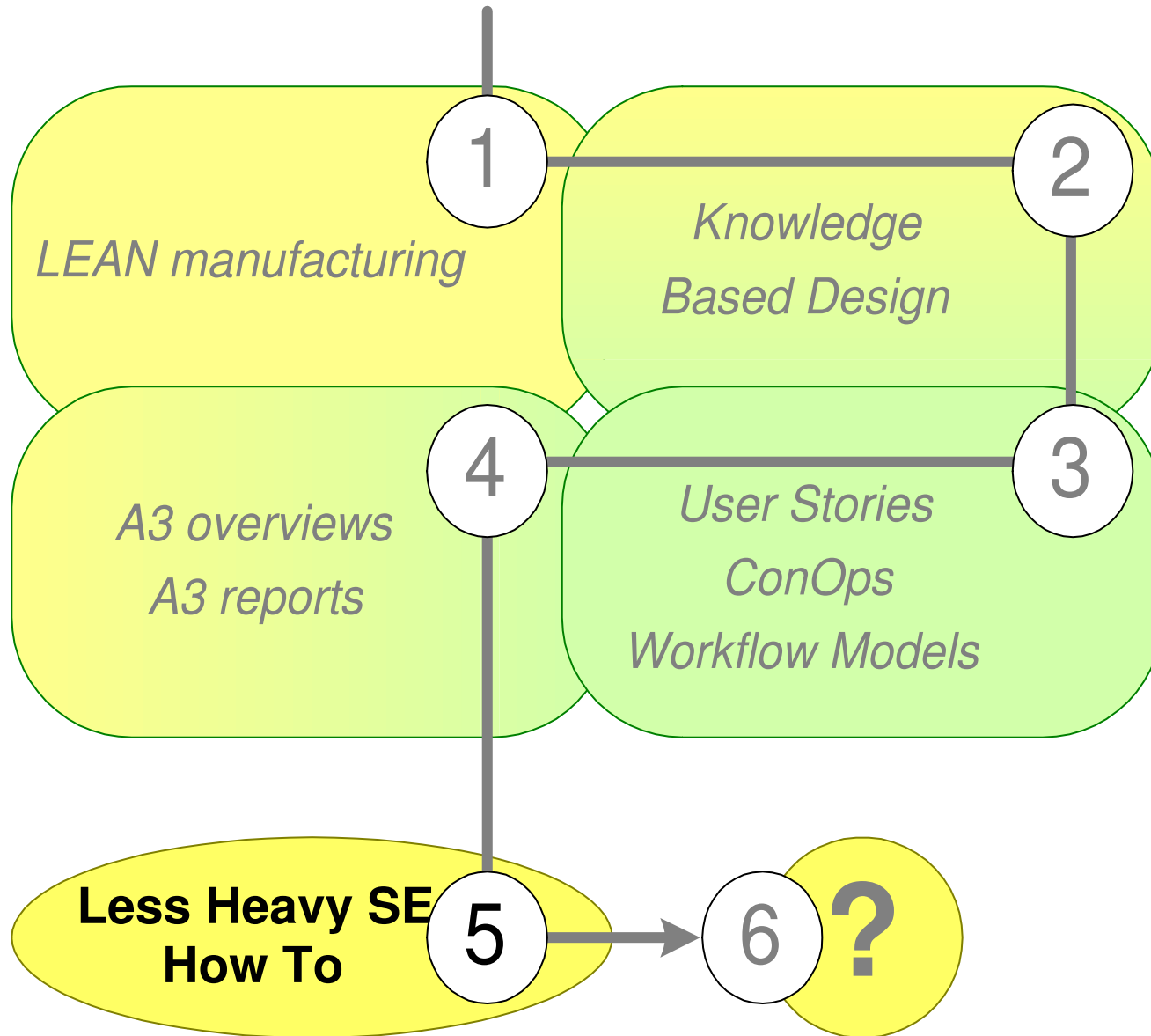
We need documents **and** A3s **and** data bases

We need to design documentation structure

We need conventions for use      naming, meta information,  
structure, storage

A3s fit in broader context

A3s are practical and work well



# Light Weight How To

1. Reduce the rule set to the (business) essential

Understand  
your customer  
your customer's customer  
etcetera

ConOps  
user stories  
work flows

$$\text{weight}(\text{architecture}) = \sum_{\text{all rules}}$$

minimize number of mandatory rules

agile

f ( level of **enforcement** ,

empower, delegate

LEAN

**scope** (impact) ,

minimize implementation details  
focus on essential concepts

A3

**size**,

Apply design principles on architecture  
**and documentation**

systems  
thinking

level of **coupling** or  
number of dependencies )

Multi-view architecting

A3

