SIEMENS
Ingenuity for life

PLM and Innovation Excellence

Learning Campus

Your partner for Business Learning

Siemens Core Learning Program

Test Architect in the Development Process

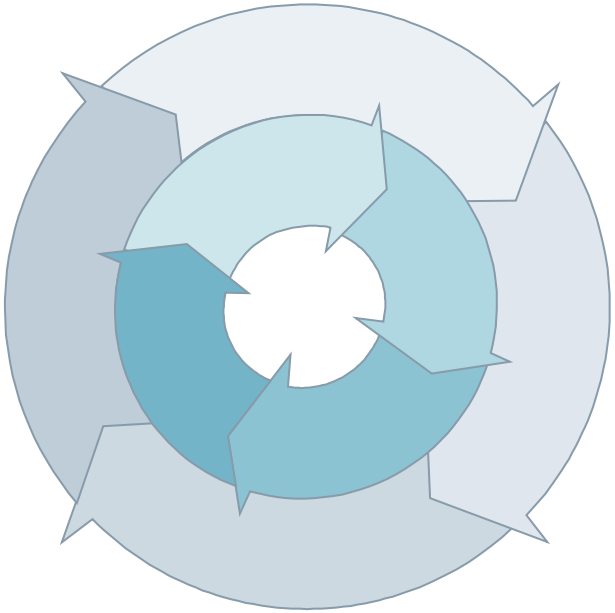
Author:
Rüdiger Kreuter, CT

Habit improves performance

Repetitio mater studiorum est!

Repetition is the mother of all learning!”

[ancient quote]



Learning objectives

- Understand the Test Architect's responsibility and interfaces in the Siemens Processes for Excellence (SIPEX) context
- Understand a system development framework and a Test Architect's responsibility within it
- Understand iterative-incremental development
- Understand interworking of a 'classic' system development with an embedded "Agile" Software development

Agenda

Core Elements of PLM

Important Lifecycle Models

Agile Software Development

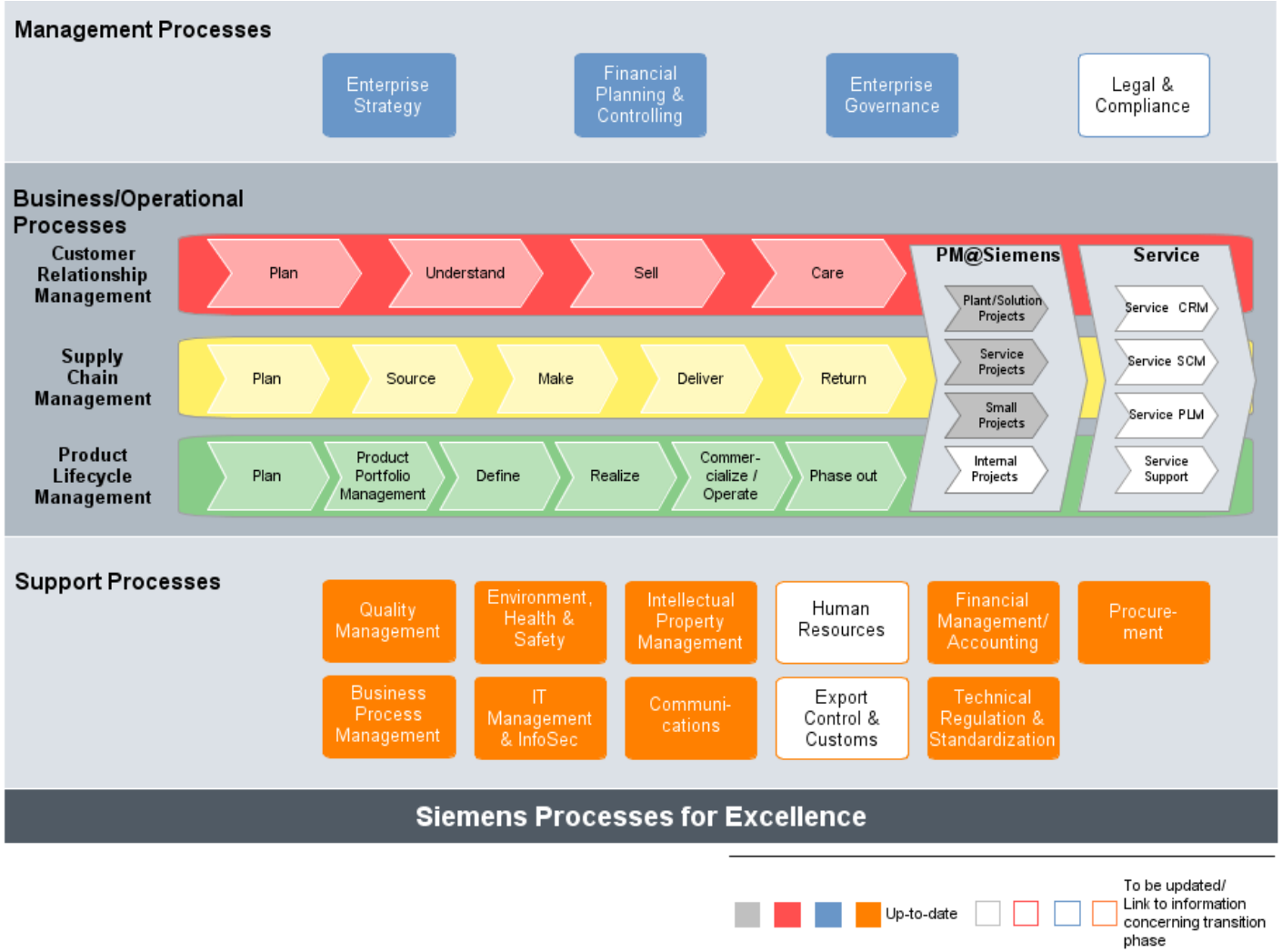
Classic system development + Agile SW development

Lean Frameworks

Summary

“Siemens Processes for Excellence” (SIPEX)

former “Siemens Reference Process House” (RPH)



Siemens Processes for Excellence (SIPEX)
Main ideas of SIPEX

SIEMENS
Ingenuity for life

- Cross-functional benchmarking and best practice sharing
- High level process harmonization across Siemens AG; e.g.
  - business processes: CRM, PLM, SCM,
  - Milestones: 10, 20, …, 100, 200, …
- Common wording and understanding for important key terms
- Define minimum scope for process content

CRM	Customer Relationship Management
PLM	Product Lifecycle Management
SCM	Supply Chain Management

Why processes anyway?

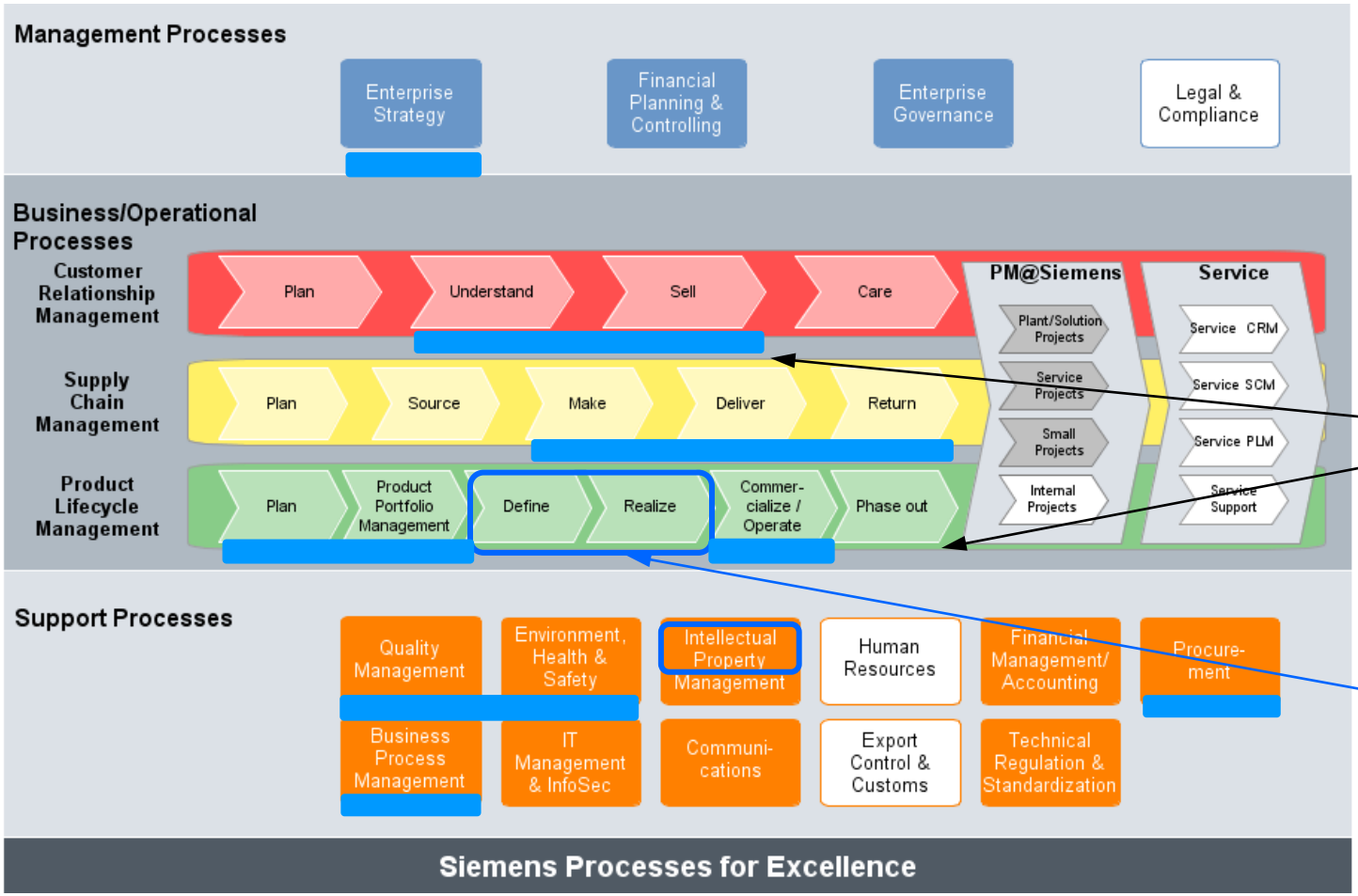
Increase quality
Minimize non-conformance cost
Enhance predictability

- best practice application
- prevent / eliminate failures early
- repeatability of successful systematic

Increase performance
Eliminate waste

- best practice off the shelf
- trained work flows (process as habit)
- learn from experience

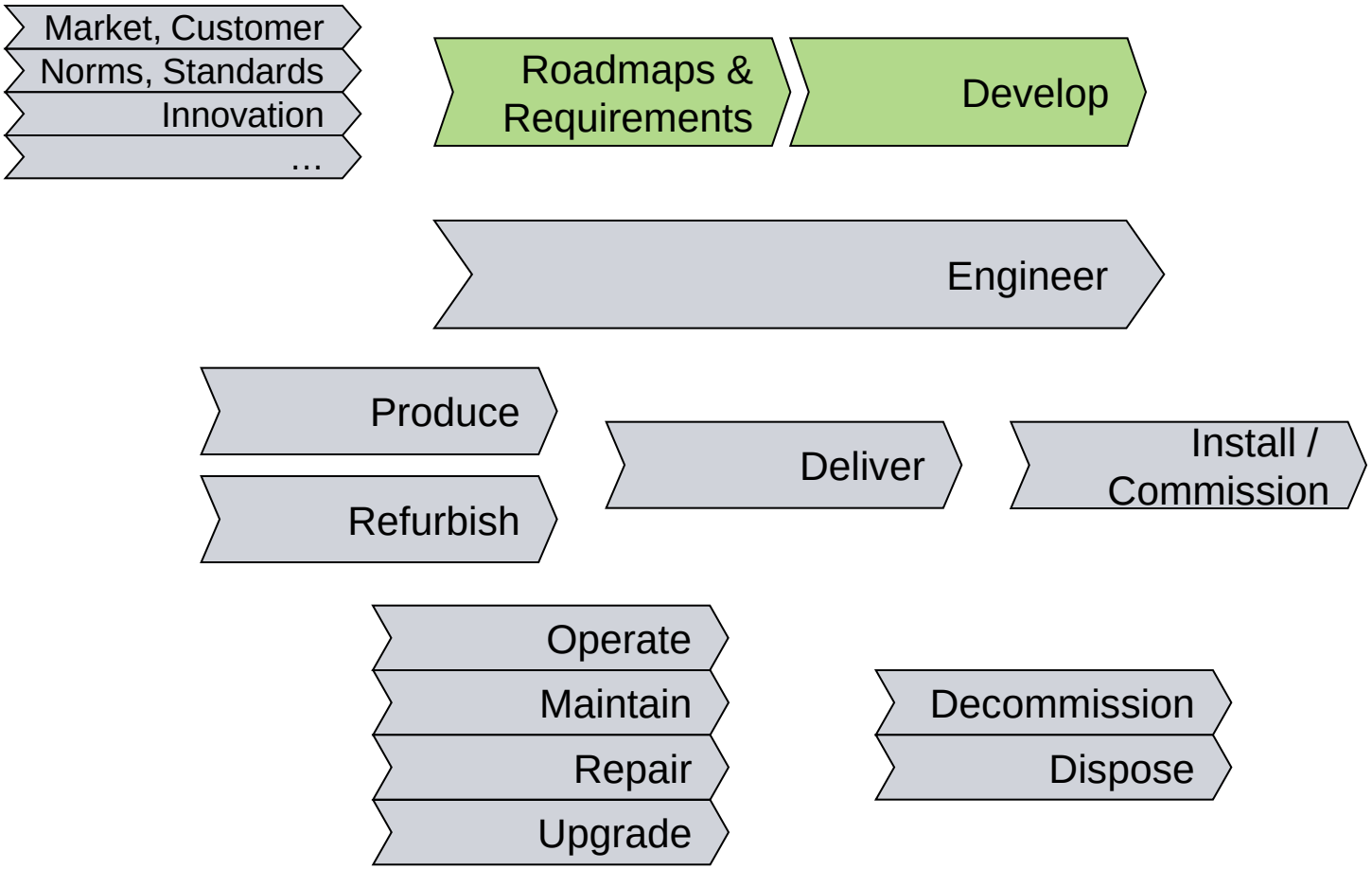
Test Architect's scope in SIPEX



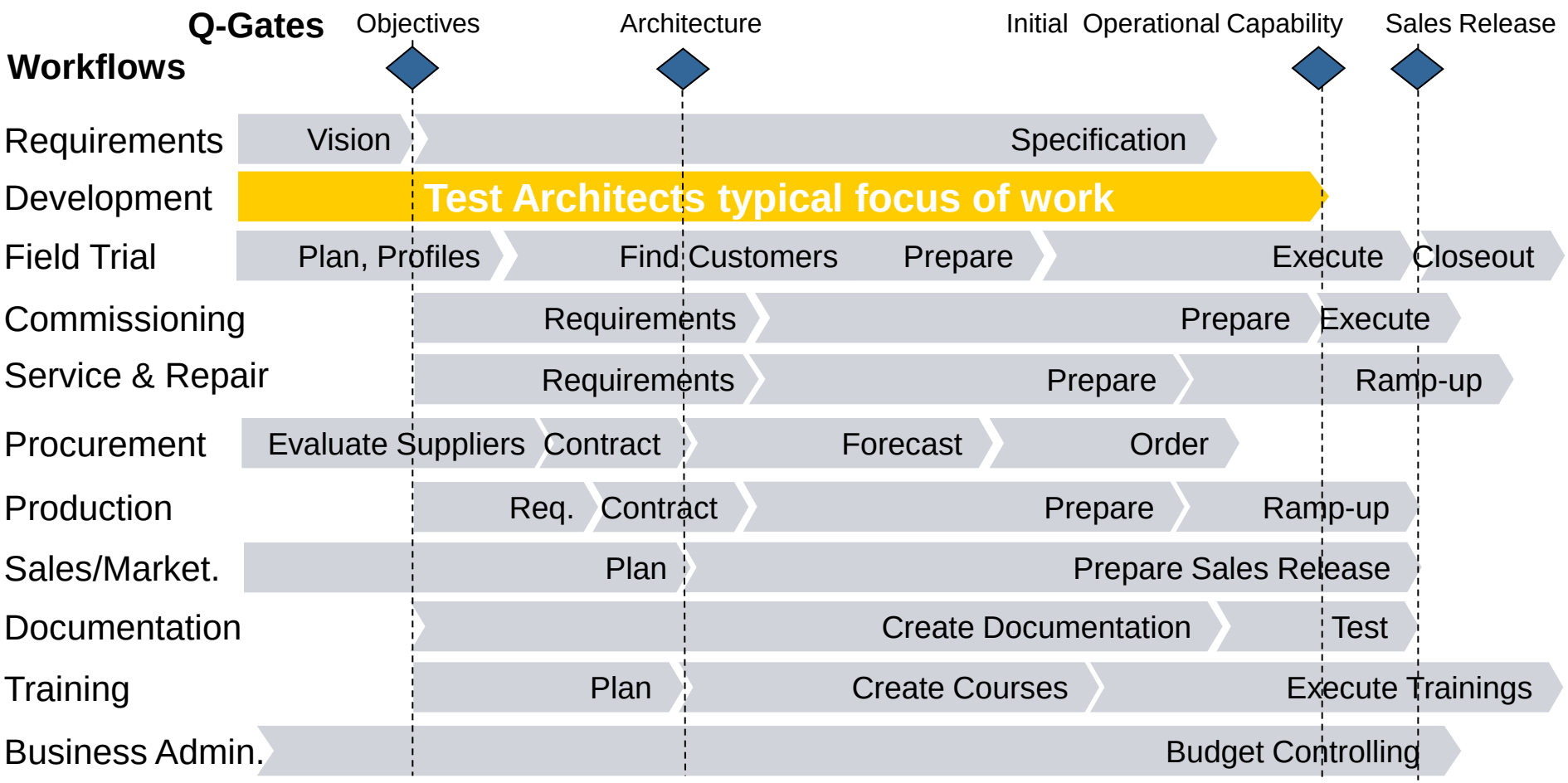
Interfaces, Support

Main focus of responsibility

Major lifecycle elements that influence a system architecture



Parallelism of Processes



Agenda

Core Elements of PLM

Important Lifecycle Models

Agile Software Development

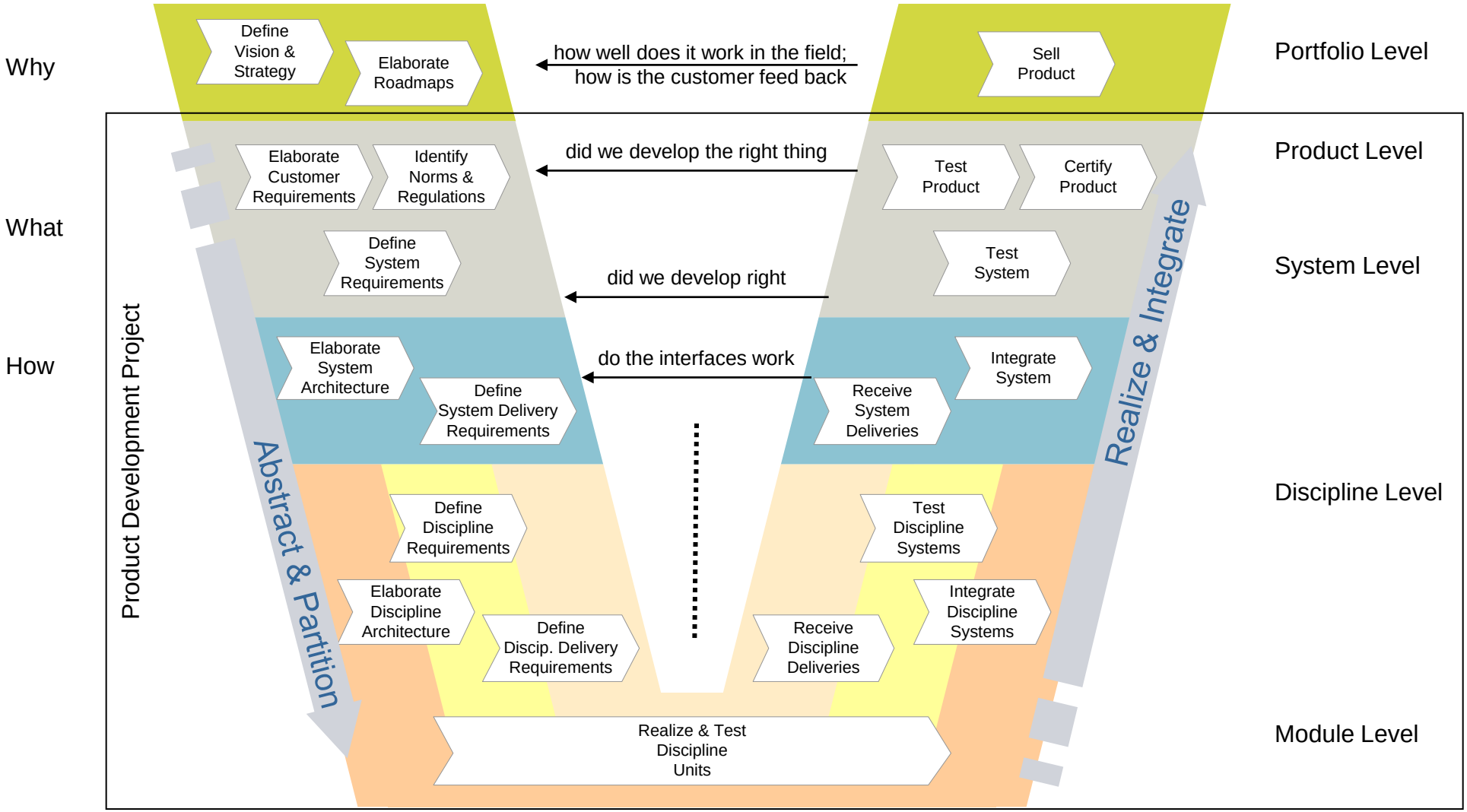
Classic system development + Agile SW development

Lean Frameworks

Summary

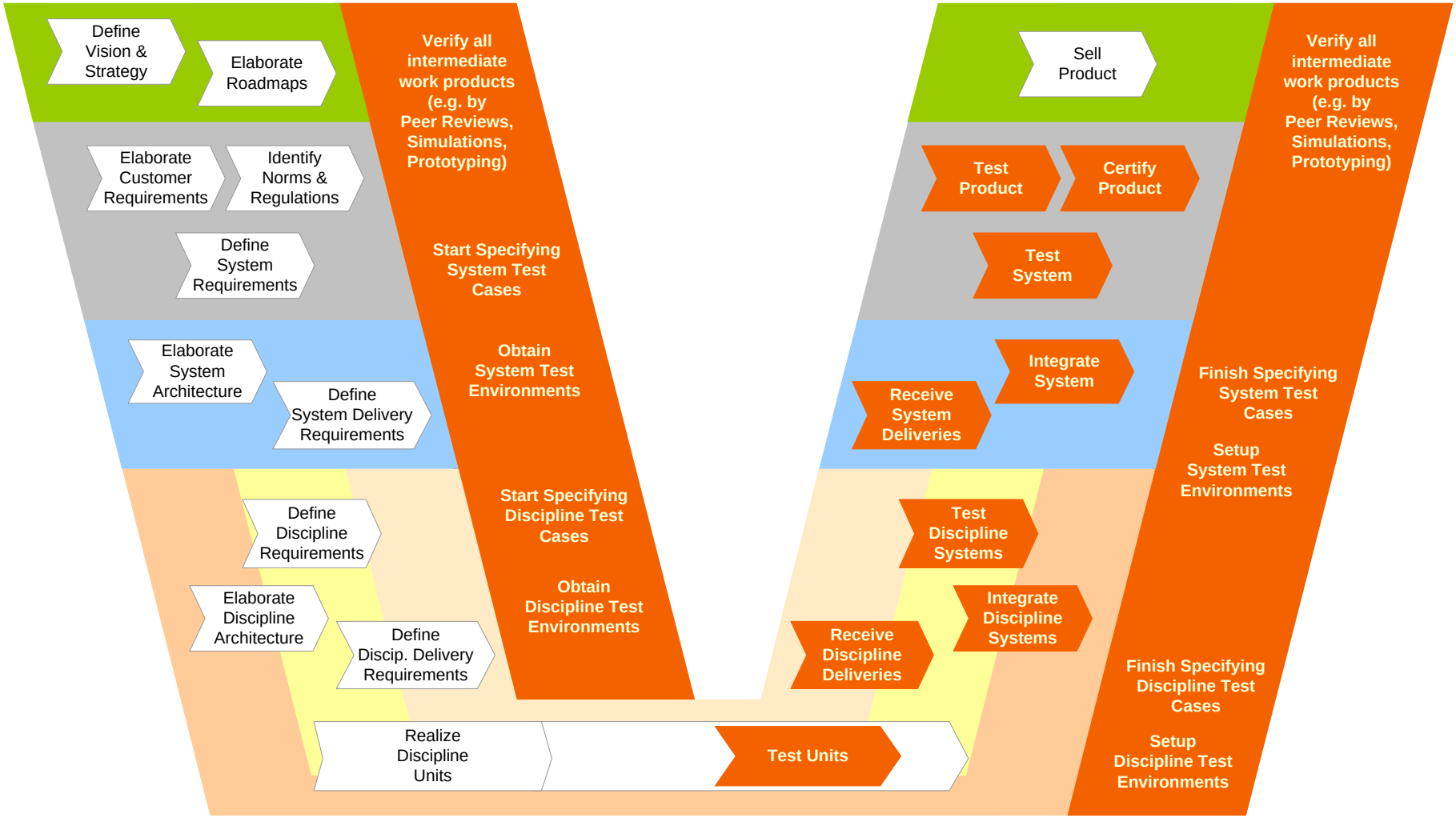
System Development Process Model basic V-Model

SIEMENS
Ingenuity for life

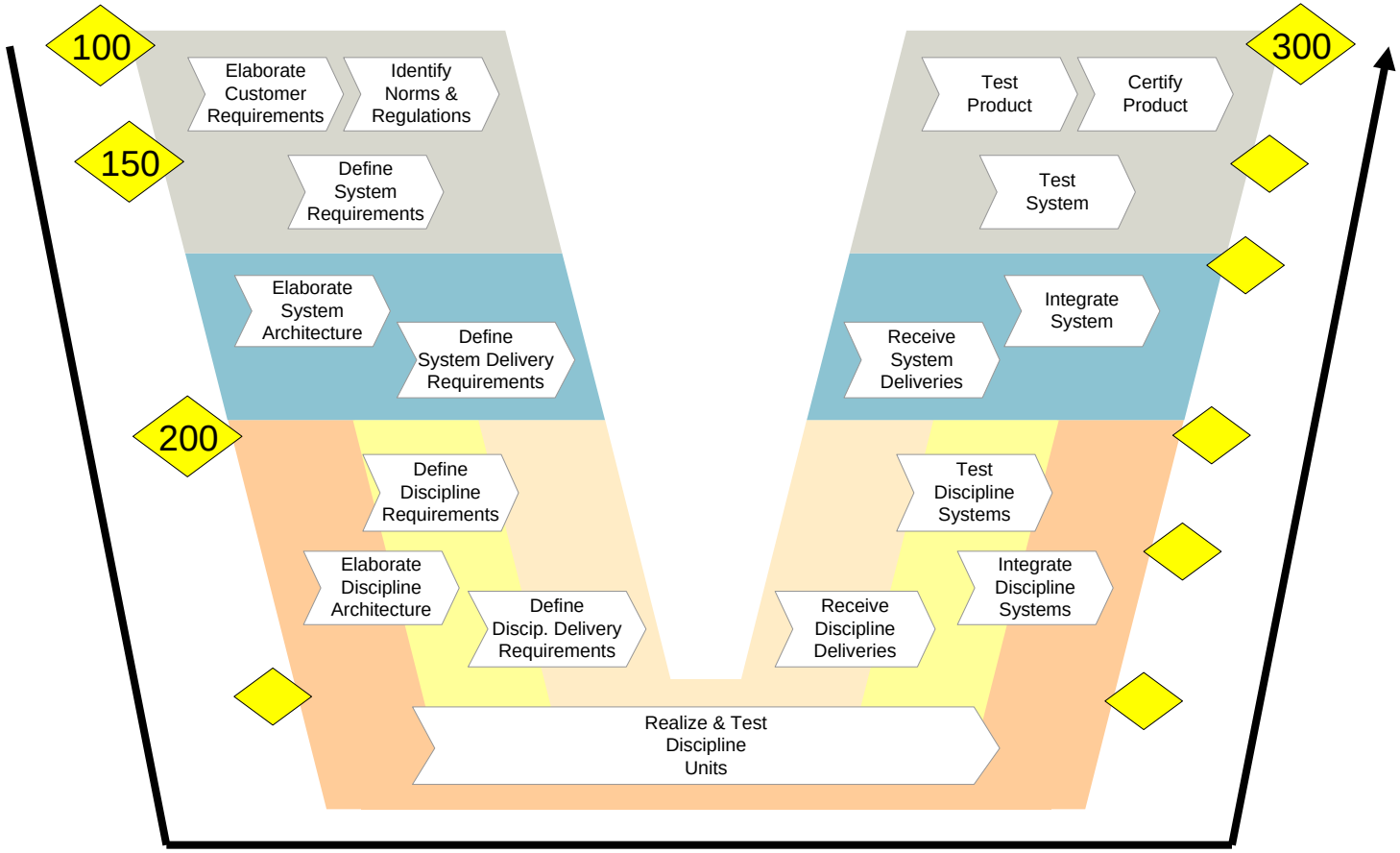


System Development Process Model
W-Model for Test & Quality

SIEMENS
Ingenuity for life



System Development Process Model Waterfall

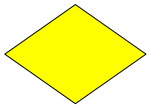


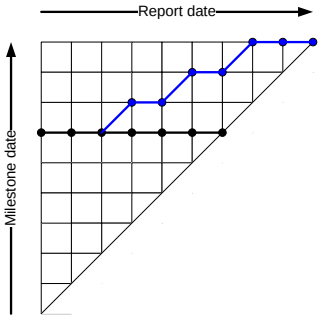
Milestone

System Development Process Model
Waterfall basics

SIEMENS
Ingenuity for life

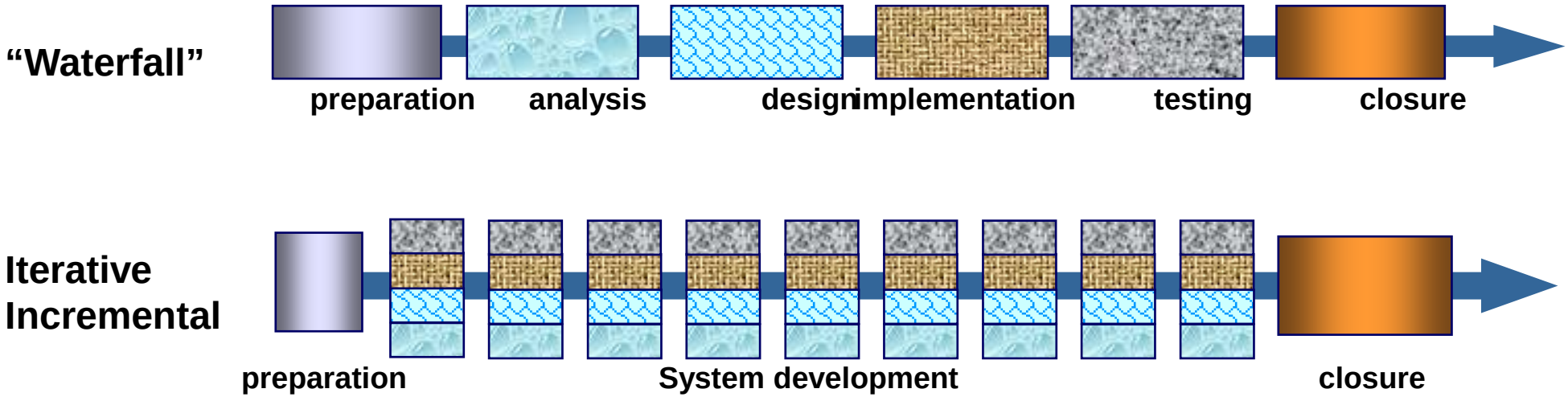
- Inherits all basic V-Model characteristics
- In addition
  - Milestones divide the process in phases
  - All (important) results of a phase shall be available in sufficient quality, before the next phase is started
  - There is only one delivery at the end of the 'V' (= "big bang integration")
  - Project status monitoring is usually done using "Milestone Trend Analysis" or (more modern) "Earned Value Analysis" and other reportings
- Works good, if requirements are (relatively) stable and the technical concept is well understood
- Does not work good in new or rapid changing environments





System Development Process Model from Waterfall to Iterative-Incremental

SIEMENS
Ingenuity for life

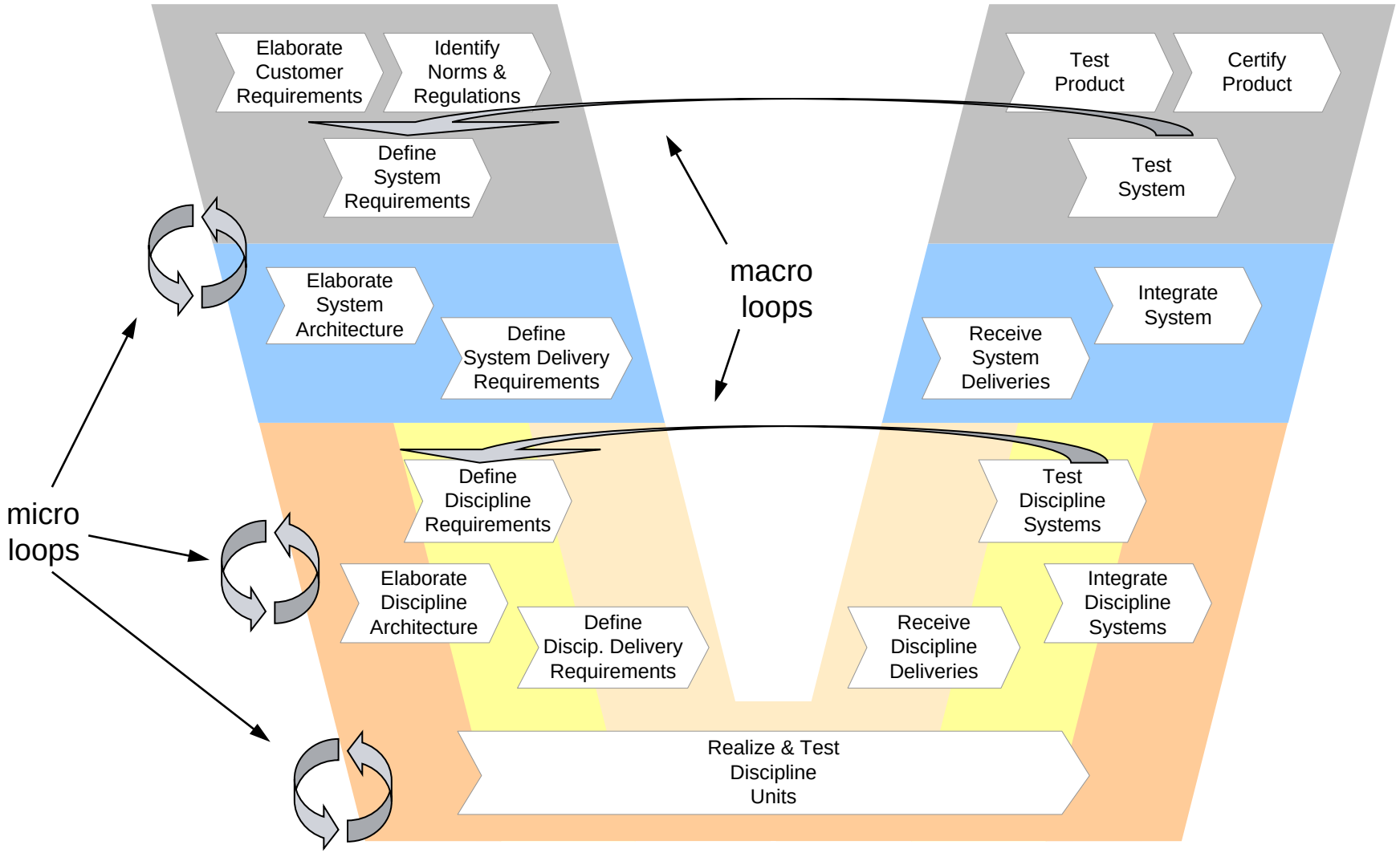


- Evolutionary model
- Each iteration includes more or less requirements analysis, design, implementation, integration, and testing within a period of time (in SW: 1-4 weeks),
- An increment delivers a running and testable system release providing new functionality

Iterative* means *re-do:
A rework scheduling strategy which helps to improve the (quality of the) product

***Incremental* means *add on to*:**
A staging and scheduling strategy which helps to improve the process and feature set by avoiding a big-bang integration

System Development Process Model
Iterative-Incremental

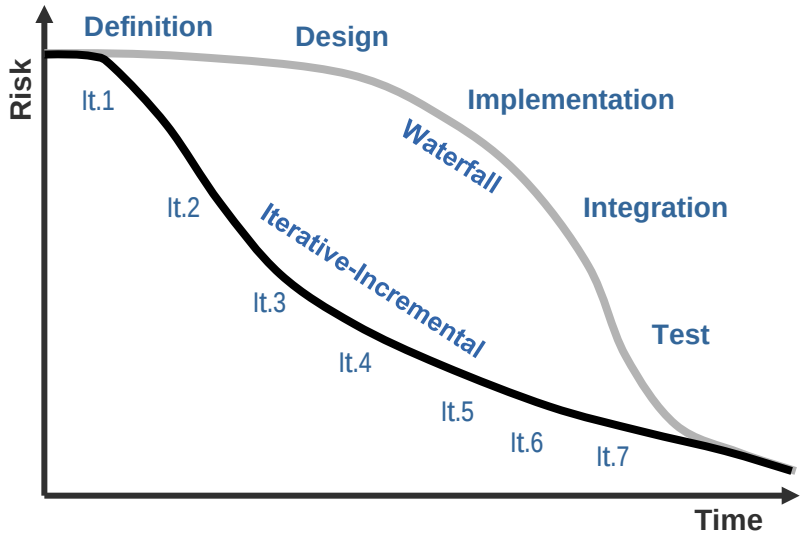


Iterative-Incremental System Development
… helps master complexity

SIEMENS
Ingenuity for life

Each iteration realizes

- a functional increment; including associated qualities
- in product quality



With each iteration the number of slices or the breadth and stability of existing slices grows …

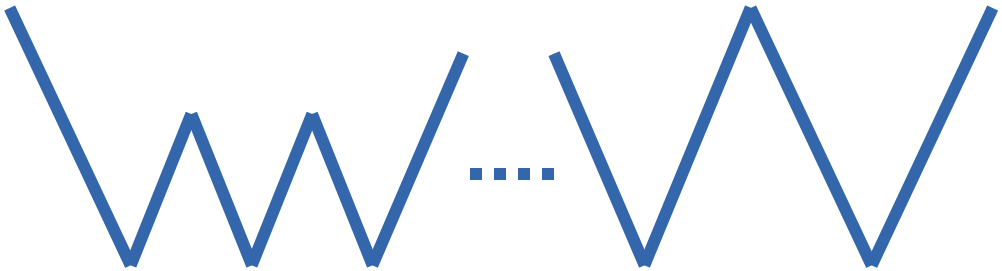
… and after the last iteration, the base-line architecture should be completely defined, realized, and tested

System Development Process Model Tailoring

SIEMENS
Ingenuity for life

Each development project needs a tailoring and adaptation of the organization's (standard) development process to the project's needs !

- **eliminate** tasks if not needed
  (e.g. a discipline development path
  or supplier management)
- **add** additional tasks if needed
  (e.g. if requested by customer,
  or for additional micro cycles)
- **define needed iterations**, depending on
  - release planning
  - necessary prototypes



Intermezzo:
The role of prototyping in Iterative-Incremental development

SIEMENS
Ingenuity for life

Prototyping in incremental development is an important quality instrument

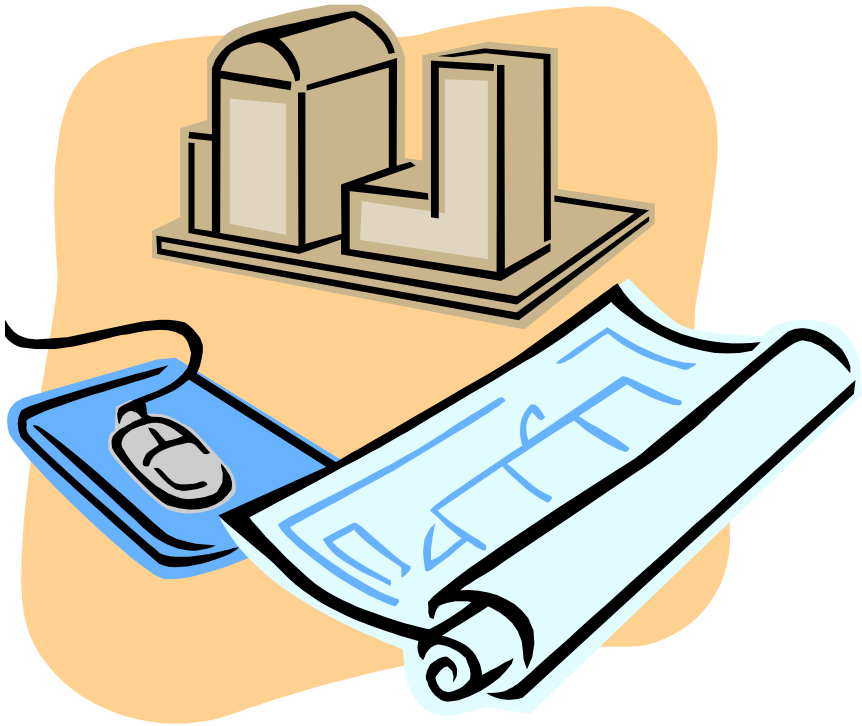
- Each iteration result is a "prototype" with product quality, i.e. an increment
- These increments can be delivered to customers at any time

Rapid prototyping

- Increments are not about "rapid prototyping"
- Challenge: Management often expects keeping quick-and-dirty prototypes

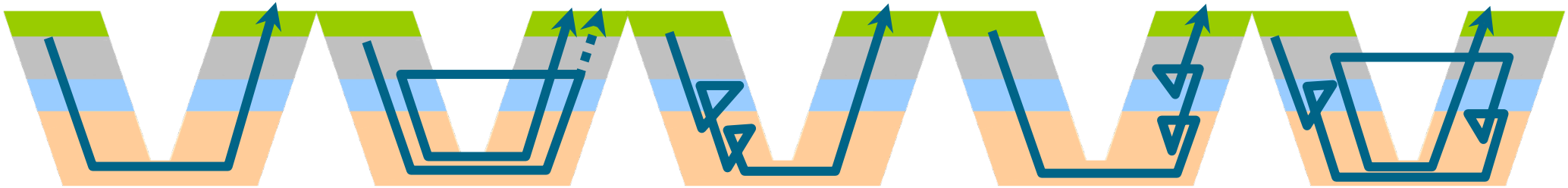
Implications

- Use "rapid prototyping" only for feasibility (throw-away) prototypes



Iterative development models
Benefits and Drawbacks

SIEMENS
Ingenuity for life

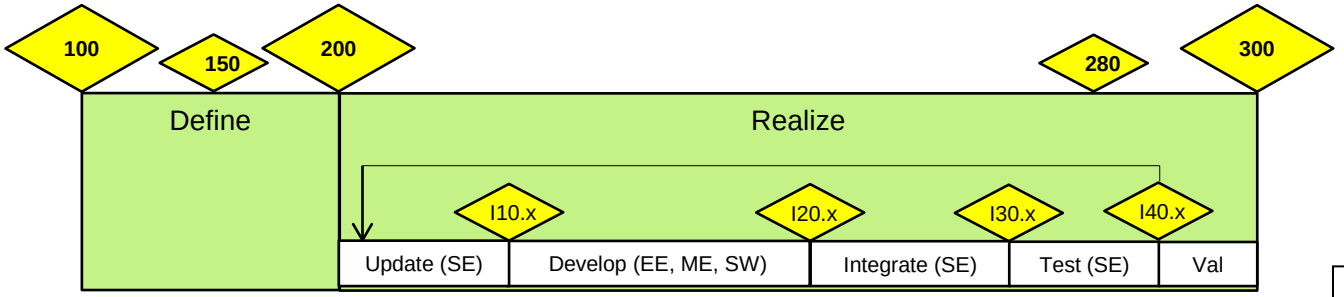


	Waterfall, Big Bang	Iterative incremental Learning Cycles	Virtual Prototyping	Incremental Integration	Hybrid Integration
+	+ „shortest path“	+ macro level incremental learning + early feedback and validation via Real Prototypes (RPT)	+ incremental learning during concept phase + early feedback and validation via Model Prototype (MPT)	+ failure encapsulation and incremental learning during integration	+ risk reduction and incremental learning via MPT and RPT + early feedback and validation in all stages
-	- high risk to fail real needs	- „long path“ learning cycle - time & cost impact	- time & cost for MPT - risk to fail real needs	- additional integration & test effort - risk to fail real needs	- time & cost for MPT/RPT - learning cycles cost & time
useful if…	solution: known concept: proven requirements: stable experience: available	solution: known concept: (non-)proven requirements: stable experience: missing	solution: unknown concept: non-proven requirements: (un)clear experience: available	solution: known concept: non-proven requirements: stable experience: missing	solution: unknown concept: non-proven requirements: (un)clear experience: missing

Iterative-Incremental System Development
… not all disciplines may deliver in each iteration

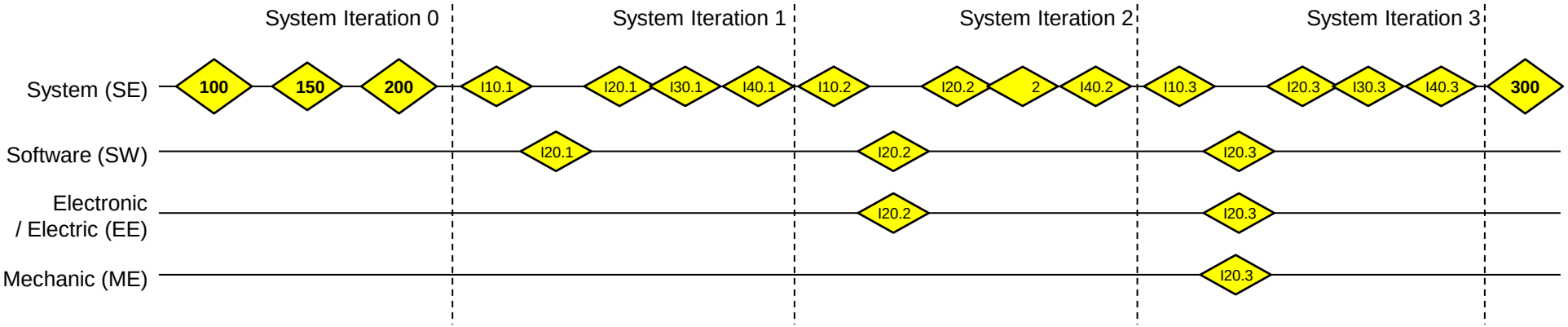
SIEMENS
Ingenuity for life

Process



EE	Electronic / Electric
ME	Mechanic
SE	System
SW	Software
Val	Validation / Certification

Instantiation of the process:
Deliveries to integration (iteration planning)



Iterative-Incremental system development

Examples for Incremental system or hardware development

- early simulations or prototypes that evolve to products
- rapid prototyping for early tests
- starting with a first increment that provides some testable but not all functionality; add functionality in each iteration
- start with old HW version and new SW in first iteration steps
- use HW / SW in the loop simulations (HIL /SIL)
- from FPGA prototypes to ASIC
- samples from pre-production to production quality
- from reference design to engineered solution
- …

ASIC	= Application Specific Integrated Circuit
FPGA	= Field Programmable Gate Array
HIL	= Hardware in the loop
SIL	= Software in the loop

Agenda

Core Elements of PLM

Important Lifecycle Models

Agile Software Development

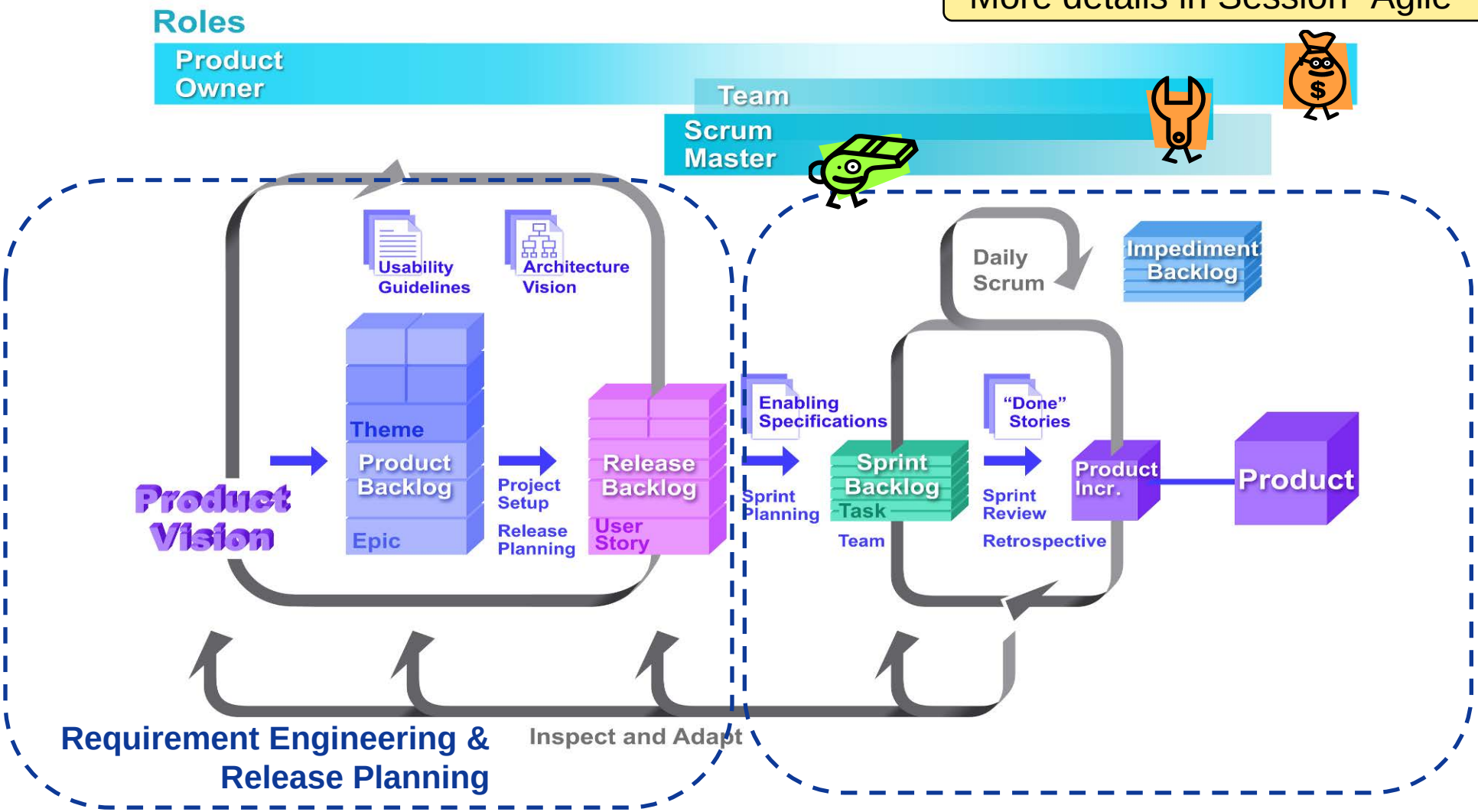
Classic system development + Agile SW development

Lean Frameworks

Summary

Scrum Overview

More details in Session “Agile”



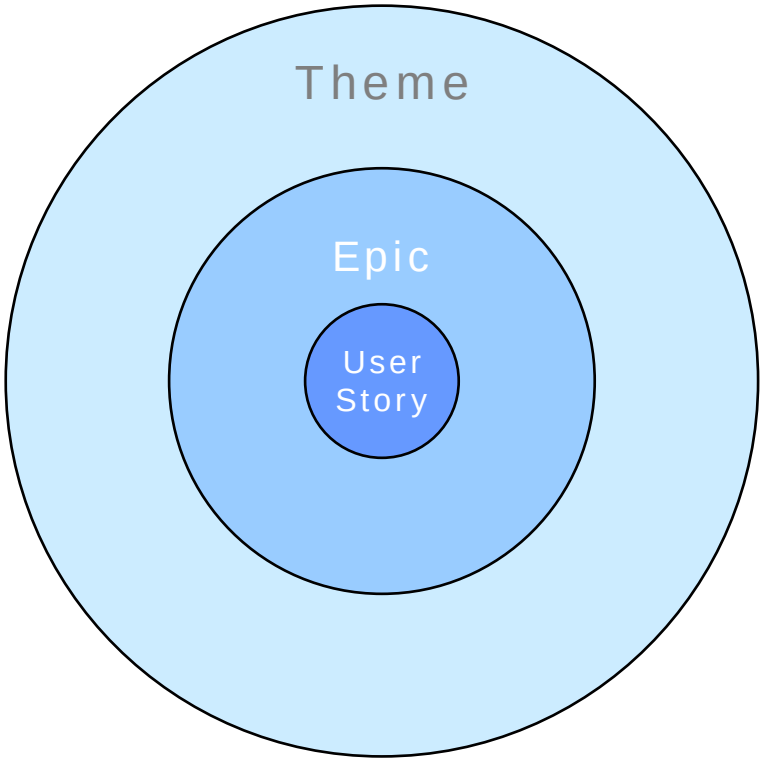
Requirements in an agile environment
Theme – Epic – User Story

SIEMENS
Ingenuity for life

More details in Session “Agile”

User story hierarchy terminology and types

- **Theme** – a top level objective
- **Epic** – A group of related user stories (Features or Requirements) that needs to be broken down to fit in a sprint
- **User Story** - a story that is small enough to fit into a sprint (does not have children)
- **Spike** - user stories that are unknown and cannot be estimated (need investigation)



Agenda

Core Elements of PLM

Important Lifecycle Models

Agile Software Development

Classic system development + Agile SW development

Lean Frameworks

Summary

Integrating "classic" system development with "Agile" software development 1

SIEMENS
Ingenuity for life

Requirements management

- User Stories are a key element of requirements engineering in an agile environment; User Stories can be used at system level too, for describing functional requirements
- RE Tool could support "backlog" and "classic" behavior at the same time

Agile welcomes change

- Set of requirements may not be complete at development start in an agile environment; but need to be clarified enough to allow a sufficiently mature system architecture
- Requirements are adjusted according to customer feedback after each delivery; therefore the system architecture should have sufficient flexibility
- Refactoring of code (to cleanup SW architecture) is part of strategy; therefore delay decisions regarding System / HW architecture as far as possible

Integrating "classic" system development with "Agile" software development 2

SIEMENS
Ingenuity for life

Project Management

- Different (understanding of) planning and reporting methods;
  but iteration planning for SW can be integrated in milestone planning at system level
- Burn down charts of agile SW projects fit better with Earned Value Analysis
  instead Milestone Trend Analysis
- Time boxing is hard to realize for HW / System development;
  this needs a careful integration planning with synchronization driven by HW deliveries
- SCRUM teams need to include experts from disciplines besides pure development,
  e.g. documentation or production ramp up

Integration strategy

- SCRUM requires that teams choose the requirements for the next iteration themselves;
  in a system environment it must be ensured that functionalities that are distributed across several products (or components), each developed by a separate team, are synchronized

Agenda

Core Elements of PLM

Important Lifecycle Models

Agile Software Development

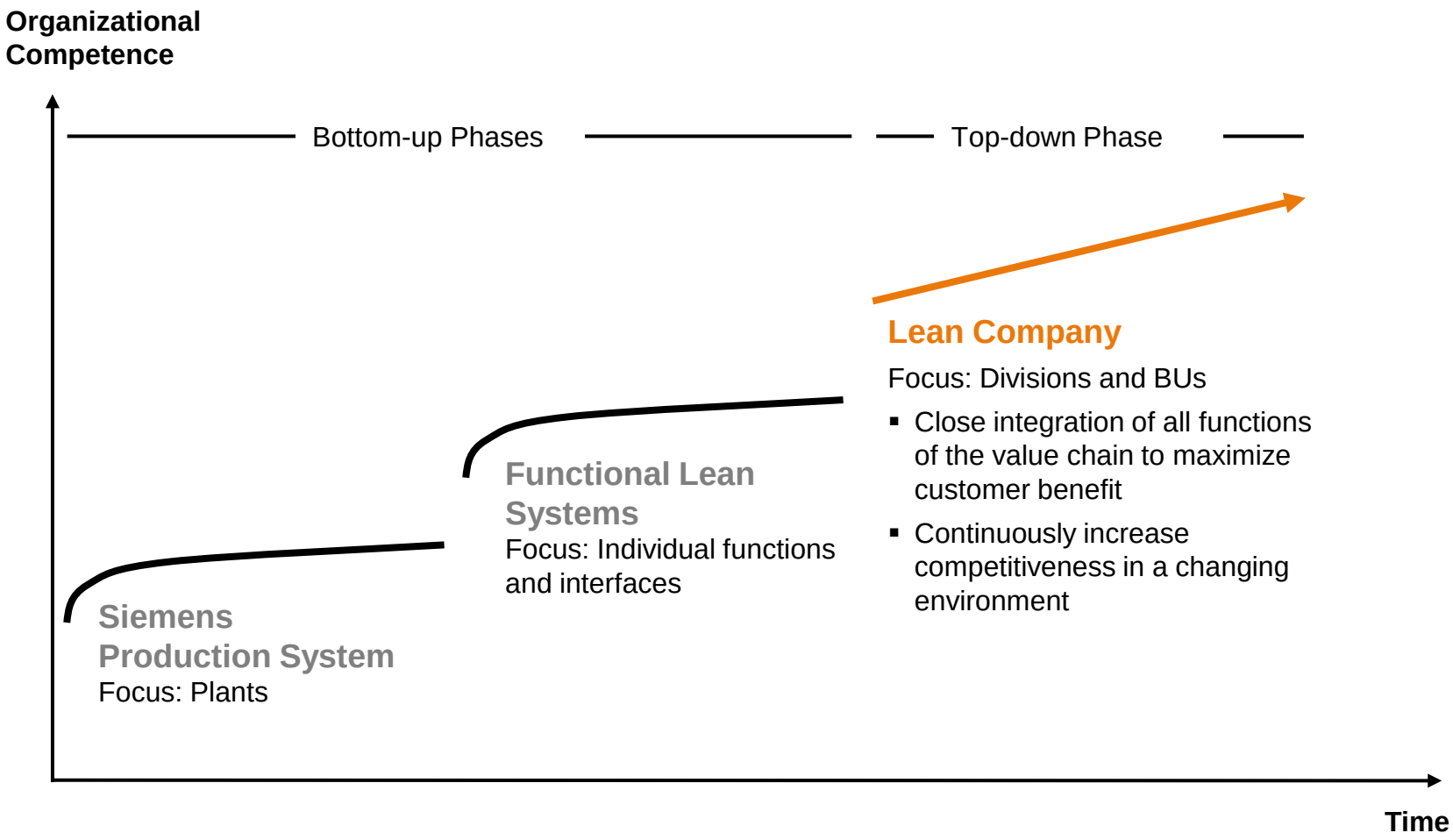
Classic system development + Agile SW development

Lean Frameworks

Summary

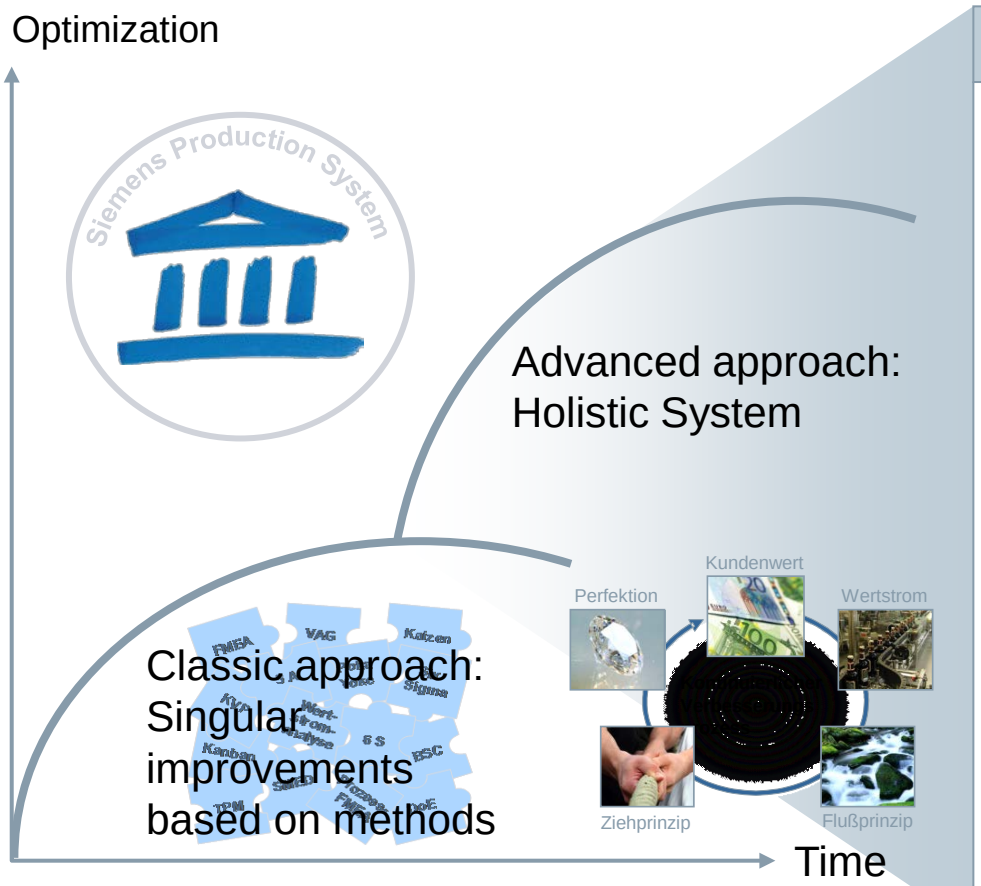
LEAN@SIEMENS
Various phases

SIEMENS
Ingenuity for life



Siemens Production System (Focus on Plants)
Structured Way to drive Lean Thinking

SIEMENS
Ingenuity for life



Holistic Production System

- A modern approach needs a holistic view across functional departments.
- A cross-functional approach along the value stream generates new perspectives for optimization
- Strong connection of principles and methods to the employees.
- Sustainability needs a sustainable culture change.

https://intranet.w1.siemens.com/cms/operations-development/en/lean/sps/Pages/default.aspx

Analog to SPS, we will establish a Siemens Development System in PLM to become much faster and flexible

SIEMENS
Ingenuity for life

Set of common, business-proven principles and methodologies

8 + 14

Lean/Agile Principles
- Focus on customer value and eliminate waste
- Optimize the entire value stream
- Establish pull and flow
- Deliver fast and create knowledge
- Go and see
- Respect people
- Improve continuously
- Quality from the beginning

Lean/Agile Methods
CEO driven Hoshin Kanri
- Ranked Backlog
- Time boxed Development
- Kanban – WIP Limits, Def. of Done
- Retrospectives
- Self-organizing/cross functional teams
- Visual management
- Daily stand-up meetings
- Gemba walk
- Value stream mapping
- Scrum for Software Development
- Structured problem solving (e.g. A3, PDCA)
- Value/Effort matrix
- Minimal Viable/Marketable Product (MVP/MMP)

One SDS manager per BU

~ 40 till FY17

- **Clear mandate** based on full management commitment
- **Dedicated time and budget** for continuous improvement activities and knowledge exchange
- **Act on behalf of BU** in lean communities and networks

Lean leadership

~ 300 till FY19

PLM heads: CTOs, R&D, ENG and PPM

PLM and Innovation Excellence Board as role model

- **Culture Lab -** 1 day
- **Go and see** with 1-2 visits p.a. to best-in class BUs
- **Lean for Leaders**: 2+2 day coaching & training

Functional experts

~ 1000 till FY21

Strengthen **expert programs based on BU needs**

- → Continue with lean experts
- → Adapt R&D project manager development to SDS
- → Integrate SDS in system / SW architects programs

The PLM and Innovation Board decided on September 9, 2016 to implement the Siemens Development System (SDS)

SDS = Siemens Development System, p.a. = per annum, SPS = Siemens Production System

8 Lean / Agile Principles and 14 related methods in the Siemens Development System (SDS)

SIEMENS
Ingenuity for life

Lean/Agile Principles

- Focus on customer value and eliminate waste
- Optimize the entire value stream
- Establish pull and flow
- Deliver fast and create knowledge
- Go and see
- Respect people
- Improve continuously
- Quality from the beginning

Lean/Agile Methods

CEO driven Hoshin Kanri

- Ranked Backlog
- Time boxed Development
- Kanban – WIP Limits, Def. of Done
- Retrospectives
- Self-organizing/cross functional teams
- Visual management
- Daily stand-up meetings
- Gemba walk
- Value stream mapping
- Scrum for Software Development
- Structured problem solving (e.g. A3, PDCA)
- Value/Effort matrix
- Minimal Viable/Marketable Product (MVP/MMP)

Principles of the Siemens Development System

Focus on customer value and eliminate waste	We know our customers and strongly focus on their benefit. We eliminate all types of non-value adding activities.
Establish pull and flow	We organize work and empower people to create fast, flexible and continuous flow – we limit work demand to capacity and maximize throughput.
Go and see	We go to the place of action and see what really happens in order to accelerate communication and decision making. We ask for impediments and offer support.
Improve continuously	We keep on improving. Lean is a journey, not a final state.
Optimize the entire value stream	We foster effective, trustful and networked collaboration between all involved global and local functions along the entire value stream.
Deliver fast and create knowledge	We actively request customer feedback in fast iterations. We encourage systematic learning and continuous knowledge exchange.
Respect people	Our teams thrive on pride, commitment, trust and applause. We value colleagues from other functions and their competence and contributions. Our managers respect their employees and provide support.
Build quality in from the beginning	We build in quality from the start. We regard problems as treasures. We detect them immediately at the point of origin and sustainably solve the root cause.

Methods of the Siemens Development System (1/2)

CEO-driven Hoshin Kanri	A strategic planning methodology, applied on BU / Segment level and used to align goals between all functions.
Time-boxed development	Strive for iterative, incremental interdisciplinary development or commissioning to gain early feedback, ideally in fixed time frames.
Retrospectives	Systematic approach to regularly reflect collaboration and team work in short cycles by identifying good practices and possible improvements.
Visual management	Key information for daily work is visualized to get transparency as basis for open and trustful communication across all hierarchical levels in order to improve efficiency and overall performance.
Ranked backlog	Ordering the work packages as guidance for daily work – what needs to be done before the next step – as early alignment and commitment of all functions involved.
Kanban – WIP limits, definition of done	Balancing of work in progress (WIP) with the available capacity, transparent communication of status and clear definition of completion criteria to improve lead time and efficiency.
Self-organizing / cross-functional teams	The team includes all functions necessary to get the work done and decides HOW it's going to do its work within the given business constraints and without departmental thinking (silos), but rather focused on product, solution, etc.

Methods of the Siemens Development System (2/2)

Daily stand-up meetings	Daily / regular meetings on all hierarchical levels to communicate: • What did I accomplish? • What do I plan to achieve until the next stand-up? • What obstacles are impeding my progress?
Gemba walk	Go and See by management replacing reports and indirect information capturing.
Scrum for software development	A time-boxed iterative and incremental approach using ranked backlogs and visual management.
Value / Effort matrix	Evaluation and prioritization of work packages based on their value and implementation effort, e.g. to create a ranked backlog.
Value stream mapping	Method to analyze and optimize the process based on the value stream.
Structured problem solving (e.g. A3, PDCA)	Structured approaches to guide collaborative in-depth problem solving by identifying root causes, and to continuously improve on all levels.
Minimal viable / marketable product (MVP / MMP)	Develop a system in smallest usable chunks to gain fast (customer) feedback.

Agenda

Core Elements of PLM

Important Lifecycle Models

Agile Software Development

Classic system development + Agile SW development

Lean Frameworks

Summary

Comparing Process Models

SIEMENS
Ingenuity for life

Some basic questions …

- Are requirements known and stable?
- Is the best solution known, or do we have a given architecture from history?
- Is the solution known to be working?
- Do we expect a "first time right implementation" ?

… will control the coarse grain tailoring

- waterfall versus
- iterative incremental models versus
- early phase loops (e.g. "Haberfellner Modell" = Pre-Studie + Main Study + …)
- agile …

A departing thought

No matter what the problem is,
it's always a people problem.

[Jerry Weinberg]

Further readings

Use the SSA Wiki :
https://wiki.ct.siemens.de/x/fReTBQ

and check the “Reading recommendations”:
https://wiki.ct.siemens.de/x/-pRgBg

- Architect's Resources:
  - Competence related content
  - Technology related content
  - Design Essays
  - Collection of How-To articles
  - Tools and Templates
  - Reading recommendations
  - Job Profiles for architects
  - External Trainings
  - ... more resources