

Corporate Technology

Applying Agile and Scrum Practices at Siemens

Sabine Canditt

Siemens AG, CT SE 3

Otto-Hahn-Ring 6

D 81739 Munich

Tel: +49 89 636 46752







sabine.canditt@siemens.com



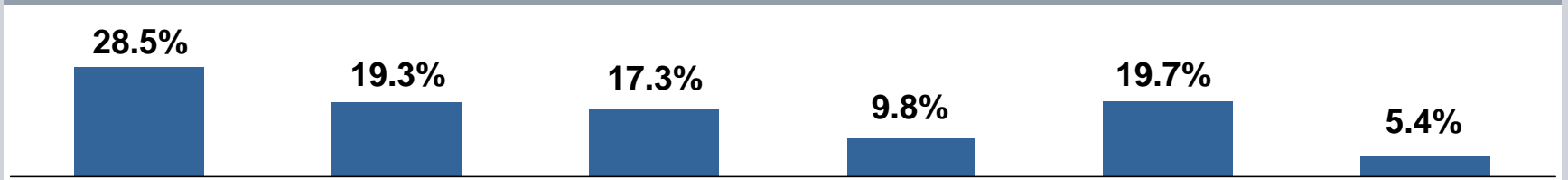
Motivation

- Scrum is a useful, albeit generic framework that does not answer all questions. In large companies like Siemens, special problems have to be considered.
- Introducing Scrum requires many changes; however not everything can / has to be changed – and not all at once.
- Here are some typical problems and pragmatic solution outlines to show how this can be done.
 - They show how „core Scrum“ can be applied to sub-processes.
 - They work under *certain* real life conditions – not everywhere.
 - They may be a step towards more agility – or may be here to stay.

Active in six business areas

Automation and Control	Power	Transportation	Medical	Information and Communications	Lighting
					
Automation and Drives	Power Generation	Transportation Systems	Medical Solutions	Communications ¹⁾	OSRAM
Industrial Solutions and Services	Power Transmission and Distribution	Siemens VDO Automotive AG ³⁾		Siemens IT Solutions and Services ²⁾	
Siemens Building Technologies					

External sales of Operations Groups excluding Other Operations (as of September 30, 2006)

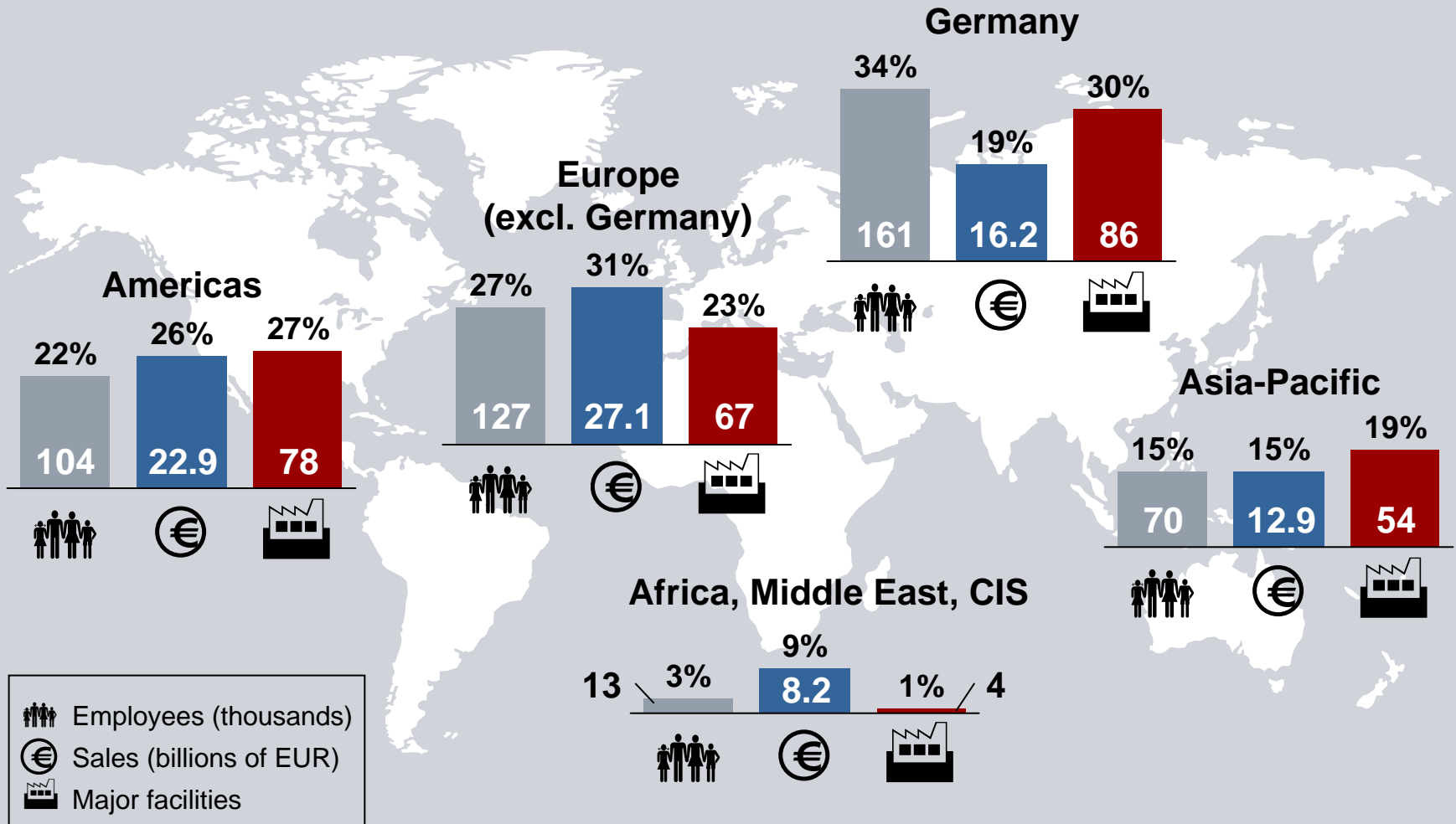


1) Since Oct.1, 2006, Com is no longer a Group.

2) Siemens Business Services (SBS) Group until January 15, 2007

3) Subject to approval by regulatory authorities, the Group will be sold to Continental AG

Global presence – basis for competitiveness

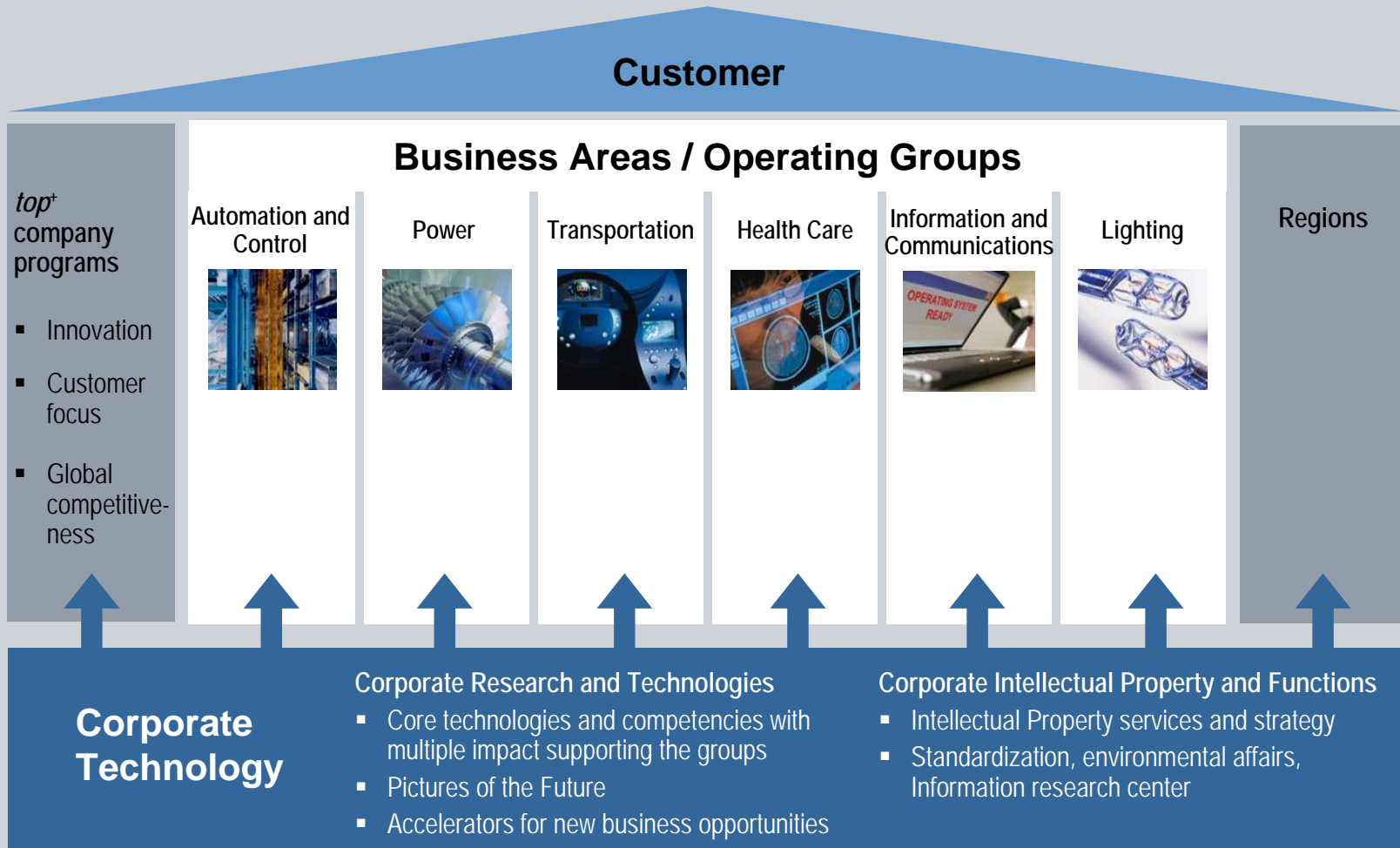


Global presence of R&D: 48,900 R&D employees at more than 150 locations in over 30 countries



Corporate Technology

Core technologies, competencies and services for the Groups



Corporate Research and Technologies

Software & Engineering Technology Division



Optimization of planning, decision and production processes

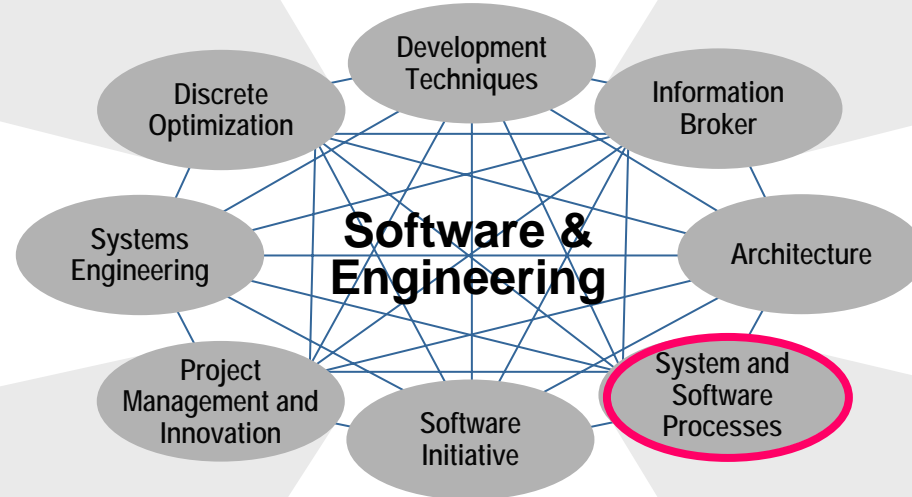


Quality and efficiency in software development



Information brokers and technical liaison managers

Analysis and engineering of complex systems



Software architecture for distributed, mobile and embedded systems



Project management and innovation



Siemens Software Initiative



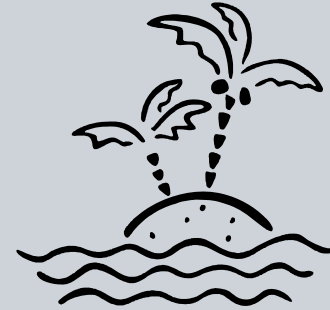
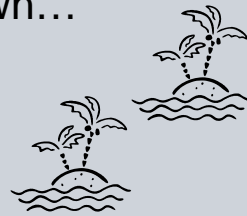
System and software processes



Agile@Siemens

- **Non-uniform, depending on the (sub-)organization**

- Some with 1000s of developers at different sites
- Some with small pilot projects
- Some bottom-up, some top-down...



- **We at CT SE 3...**

- help organizations to establish and agile understanding, goal, and strategy
- coach projects and people
- organize trainings and best practice sharing across business units
 - CSM/CSPO trainings
 - Agile newsletter
 - Internal conferences

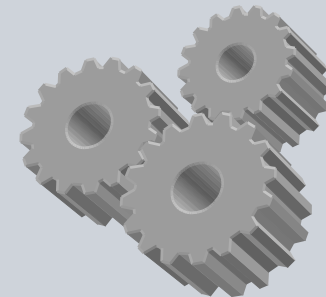
Why is Agile So Difficult in Industrial Environments?

▪ **Characteristics of industrial projects:**

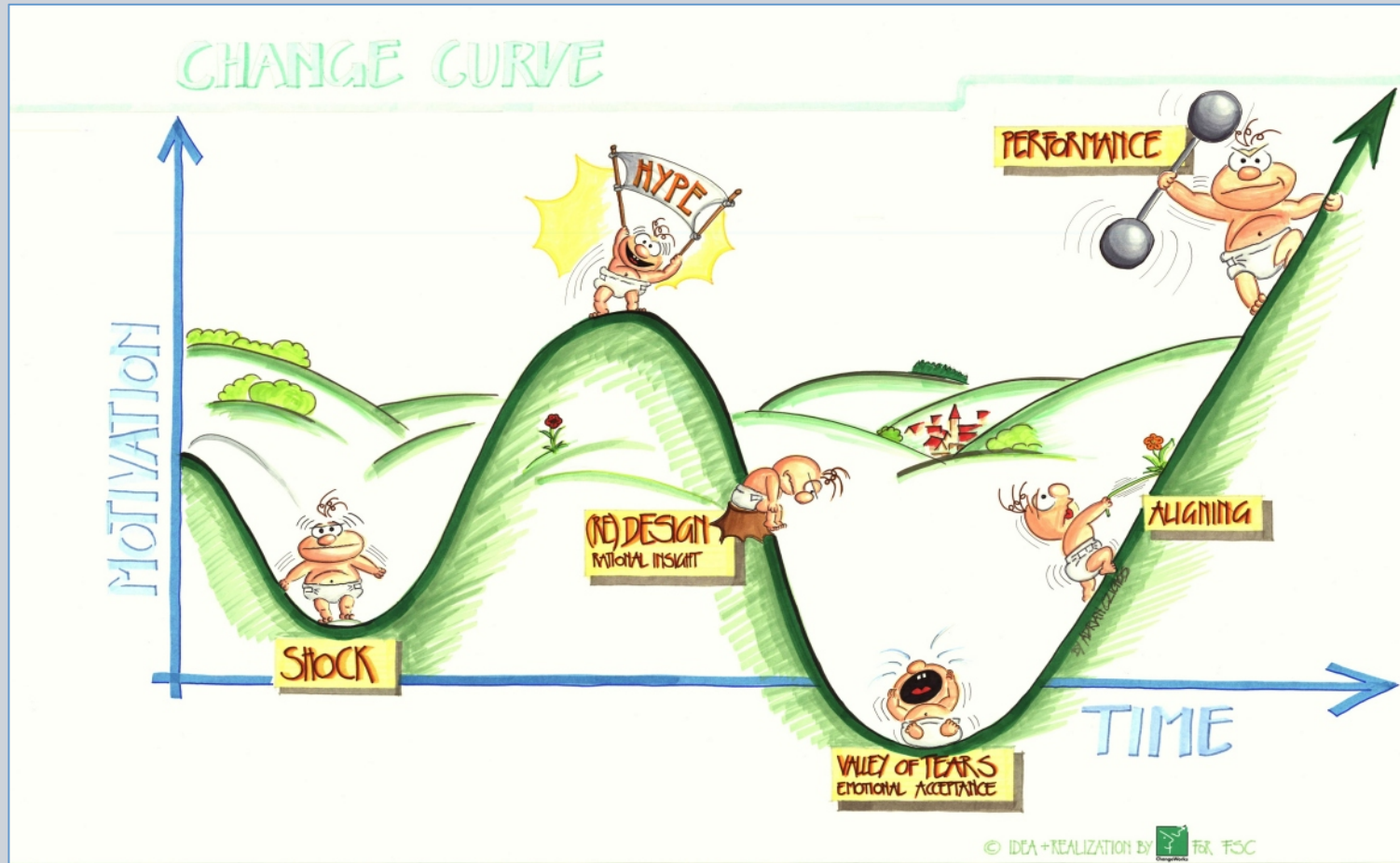
- Complex lifecycles (Portfolio Management, Sales, Marketing, Service...)
- Many dependencies, many stakeholders
- Long-running projects
- System development (incl. hardware, mechanics)
- Big teams, distributed teams
- Outsourcing / offshoring
- Safety/security regulations (e.g. FDA)

▪ **Critical areas**

- Customer involvement
- Organizational culture (e.g. waterfall history)
- Communication (distances, languages, time zones)



The Change Curve



Our Approach: Balancing Traditional and Agile



- ▶ **You should have a process that addresses your **needs, business goals, and environment.****
 - ▶ Organizational culture (e.g. management)
 - ▶ Customers (e.g. contracts)
 - ▶ Products (e.g. embedded)
 - ▶ Projects (e.g. distributed teams)
 - ▶ Tools and Processes (e.g. TDD)
 - ▶ Staff (e.g. expertise)
- ▶ **Agile and established elements can support each other to find an optimized solution.**

Examples

Requirements

Team Structure

Testing

Agile and the Product Lifecycle Process

Requirements

▪ **The problem:**

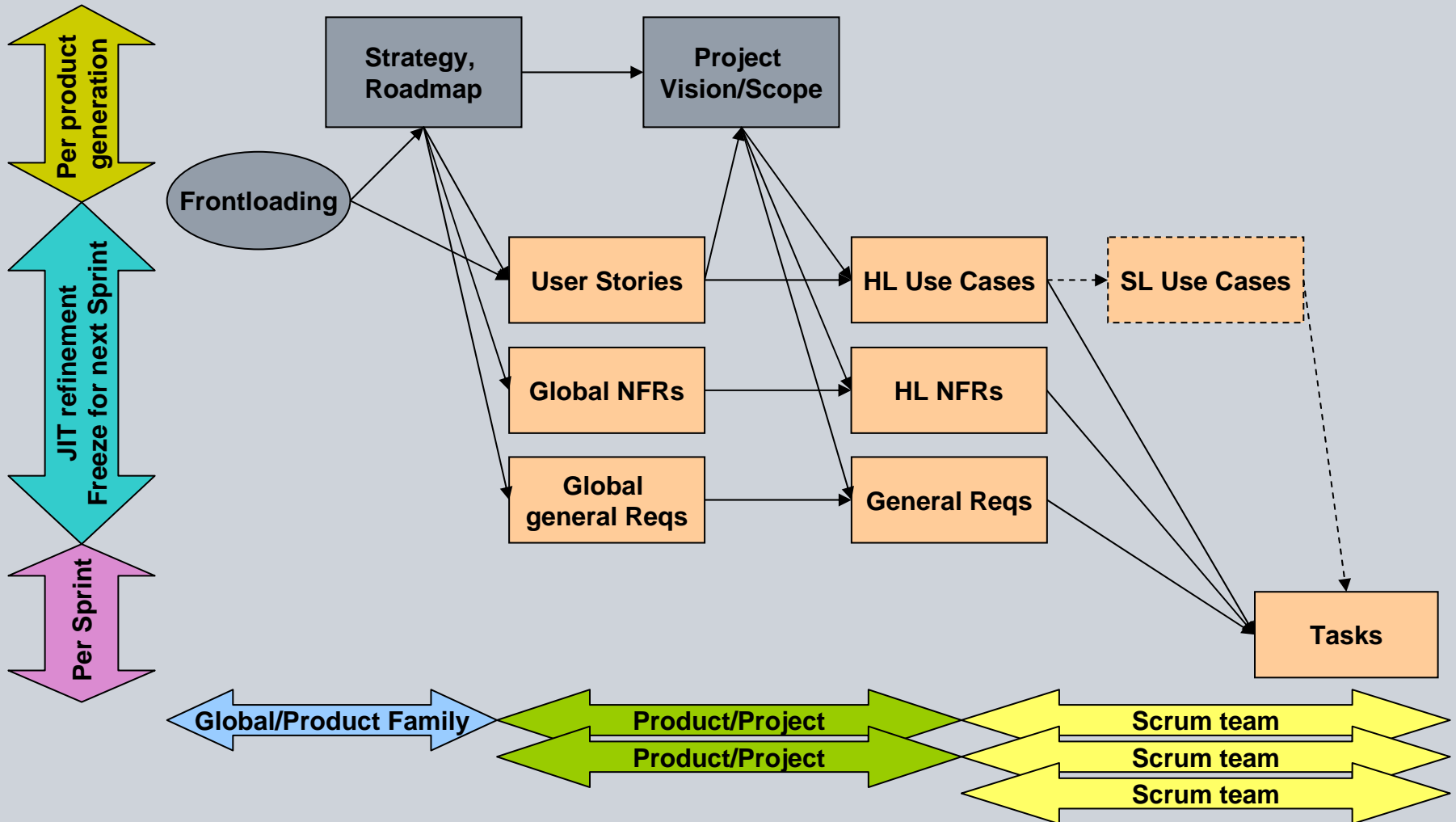
- Commercial requirements are often much too complex to be realized in one Sprint by one team.
- Breaking down requirements to Sprint tasks is time consuming.
- User stories are good for requirements with end user interaction, but how about other types of requirements?

▪ **Solution Outline:**

- The procedure has to be agile enough to capture and integrate new requirements at any time. It has to include not only development, but also “frontloading” (e.g. input from Portfolio Management, Service, lead customers...).
- Requirements should be pre-selected to avoid superfluous analysis activities and flooding the product backlog.
- Different concepts (also traditional) to express requirements can be combined.

Requirements

Example: Product Family Development



Requirements

Example: Product Family / Framework Development

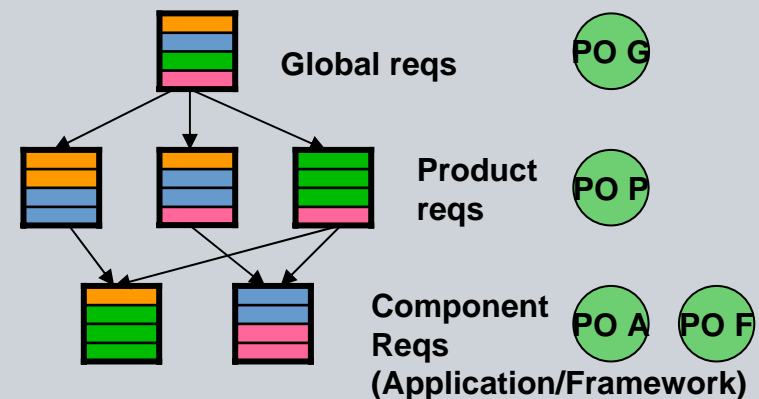
- Framework development aims at reusability of requirements (and solutions)

- Product Owners on different levels:

- Global level
- Product level
- Application level
- Framework level

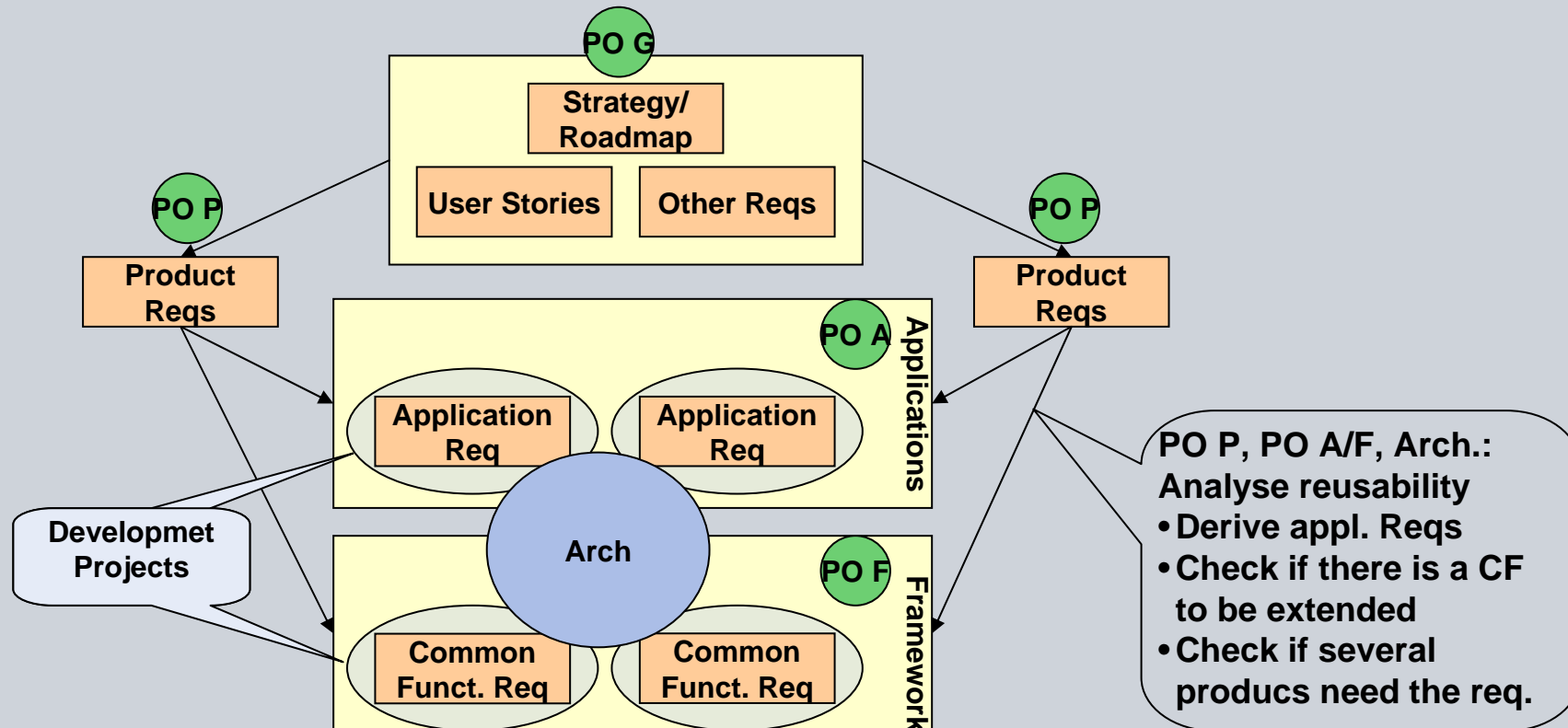
- Prioritization conflicts have to be escalated and decided by the global Product Owner.

- System architects are important to derive framework requirements and keep the architecture consistent.



Requirements

Example: Product Family / Framework Development



Team Structure

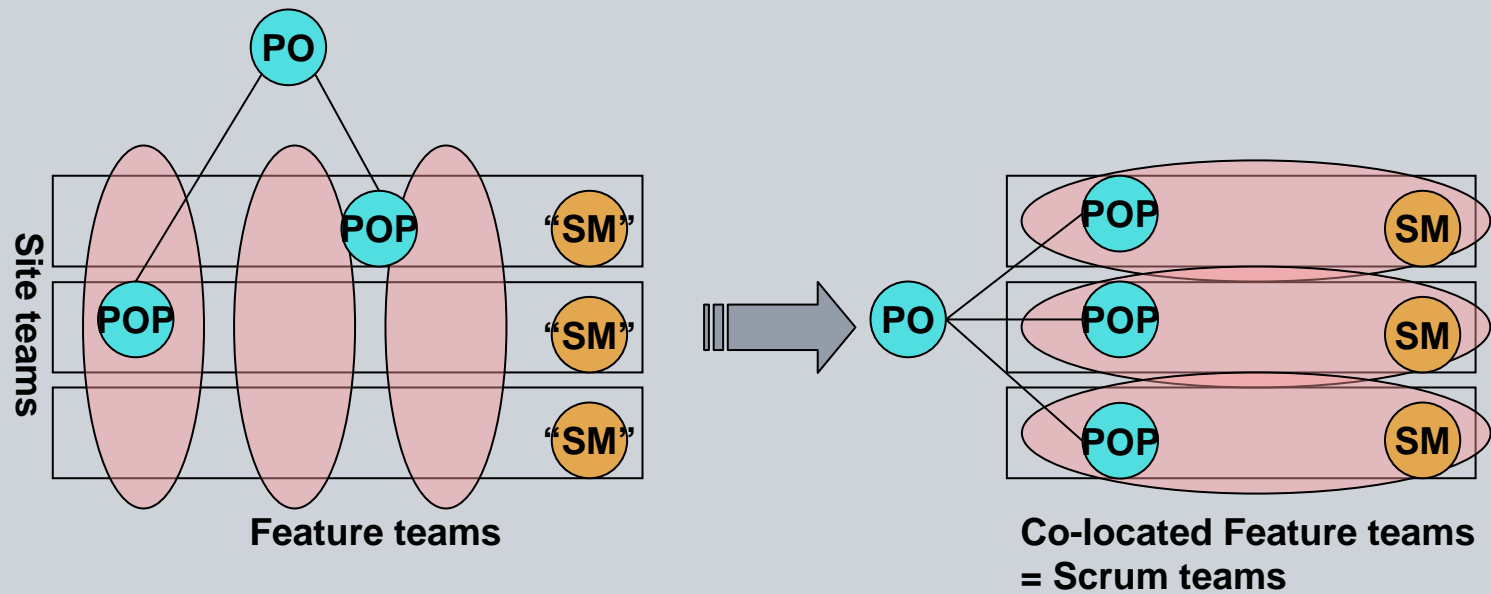
- **The problem:**

- Feature development may need expertise from different sites.
- In distributed feature teams, members hardly know each other and communication is difficult.

- **Solution outline:**

- Depending on size and distribution, there should be PO Proxys for the feature teams who understand what to do and are responsible for the team requirements.
- Scrum Masters for remote team members don't make much sense. People need local Scrum Masters they know and trust.
- Feature teams should meet regularly (requirement workshops, Sprint planning and reviews). This is often considered as too expensive!
- Mid term goal: co-located feature teams. This often requires a new strategy for site-specific core competencies!

Team Structure Example



Where are the Scrum teams???

SM = Scrum Master
PO = Product Owner
POP = Product Owner Proxy

Team Structure

Process Issues and Quality Assurance

- **The problem:**

- For multiple (distributed) Scrum teams, it is hard to establish a „one team“ spirit and common goal.
- Self-organization mainly works locally. It is difficult to inspect and adapt across Scrum team borders.

- **Solution outline:**

- **QA (or Scrum of Scrum Master) to help teams with global quality / process issues**

- Bug tracking (e.g. if reporter and resolver are in different teams)
- Tracking of reviews and test sufficiency (4-eyes-principle)
- Supporting optimization of common activities (e.g. release planning, synchronization of Sprint planning, root cause analysis)
- Providing process guidance (e.g. checklists, templates, definition of common rules and procedures...)
- Providing metrics (e.g. release burndown)
- Risk management

Testing

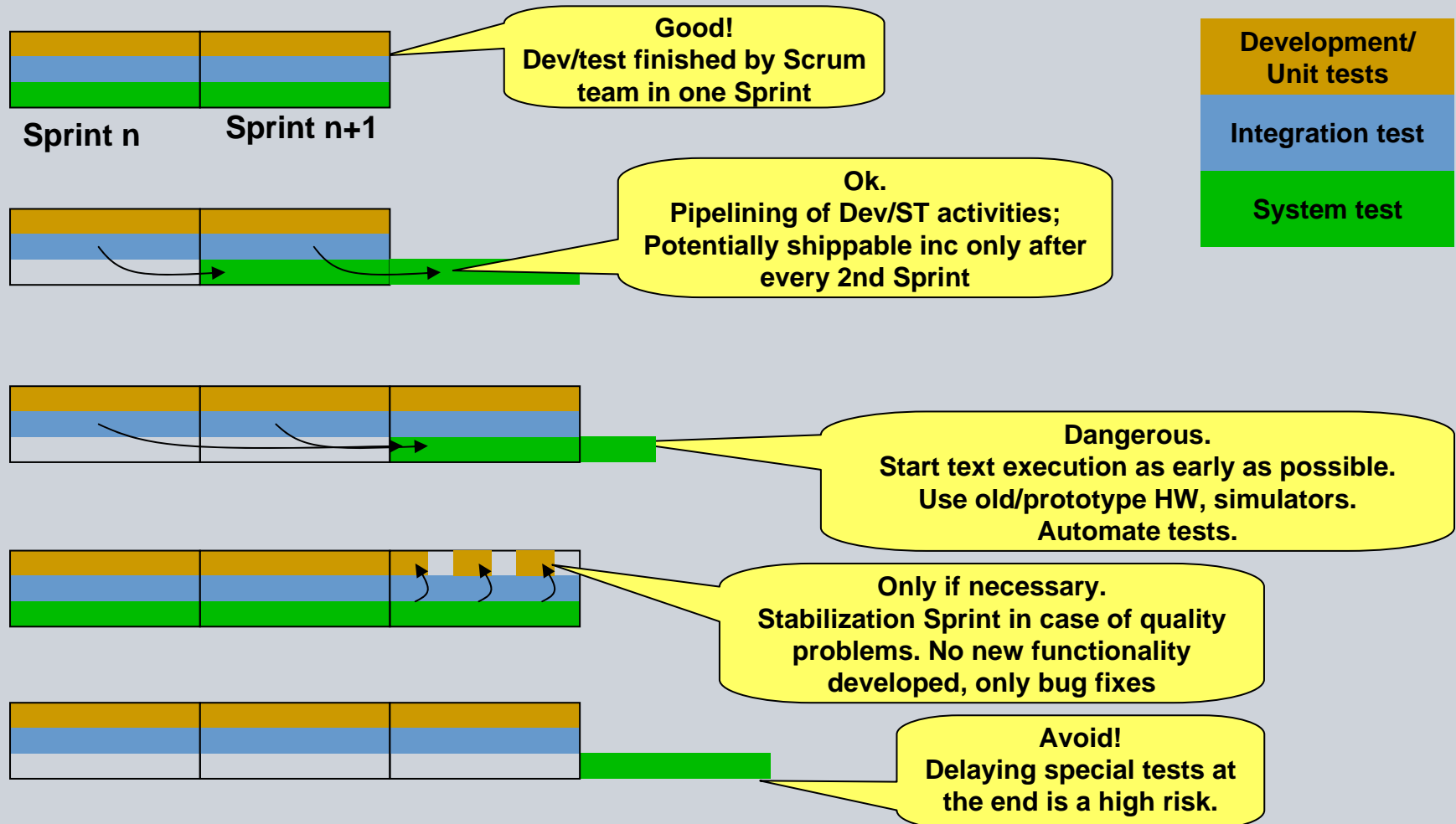
- **The problem:**

- Testing should be completed in each Sprint.
- This is sometimes not possible (yet).
 - Manual regression tests
 - Availability of hardware and non-iterative deliveries
 - Availability of test expertise and equipment

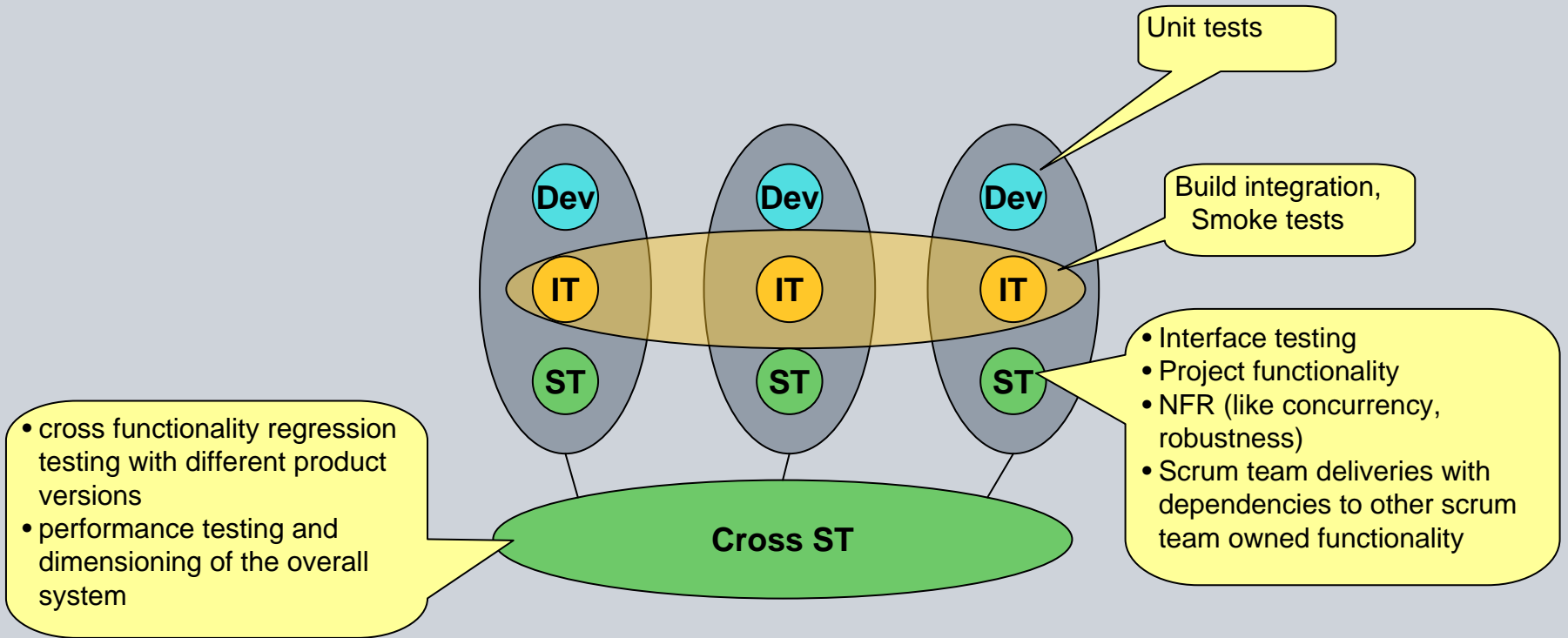
- **Solution outline:**

- Tests should be planned (incl. availability of deliveries and of test hardware).
- It has to be documented clearly which tests have to be executed when (“done” criteria).
- A separate (virtual) team may be useful to cover NFRs and end product tests, test automation and infrastructure, release preparation etc.

Testing



Testing Responsibilities Example



Agile and the Product Lifecycle Process

- **The problem:**

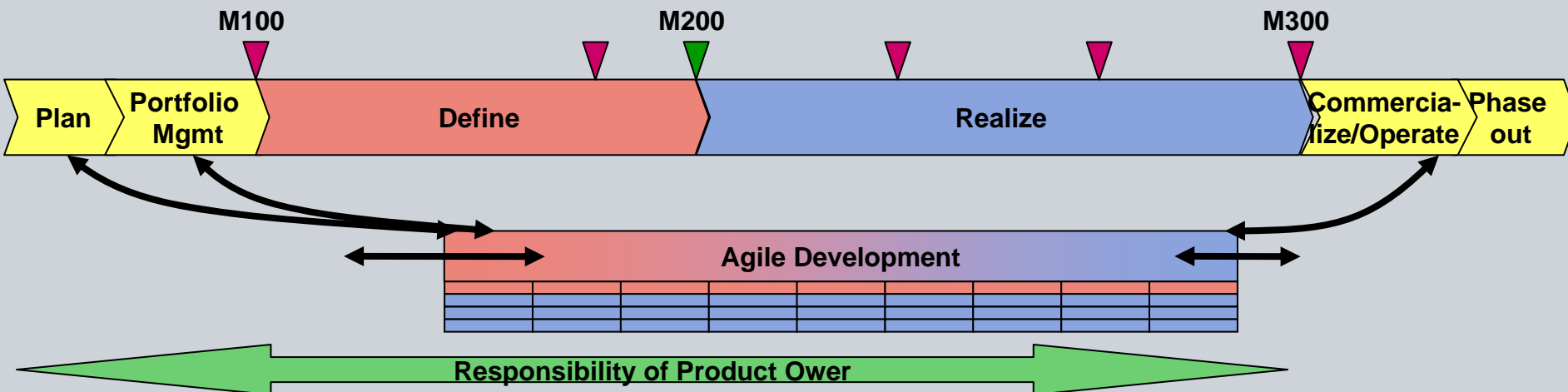
- Usually Agile covers „only“ the phases „Define“ and „Realize“. But there is more than „just“ development.
- In Agile, there is no strict border between these two phases (e.g. no Big Upfront Requirement Freeze, or Big Upfront Project Plan).
- The influence on and interfaces to other disciplines (Portfolio Management, Sales, Service, HW...) have to be adapted and optimized.
 - Development has to be “frontloaded” with new requirements iteratively.
 - Deliveries have to be offered and sold iteratively.



Agile and the Product Lifecycle Process

- **Solution outline:**

- Adapt M200 (e.g. “n% MUST requirements”)
- Avoid responsibility shift from Define to Realize (“throwing requirements over the fence”).
- Product Owner continuously involves all disciplines.



Agile and the Product Lifecycle Process

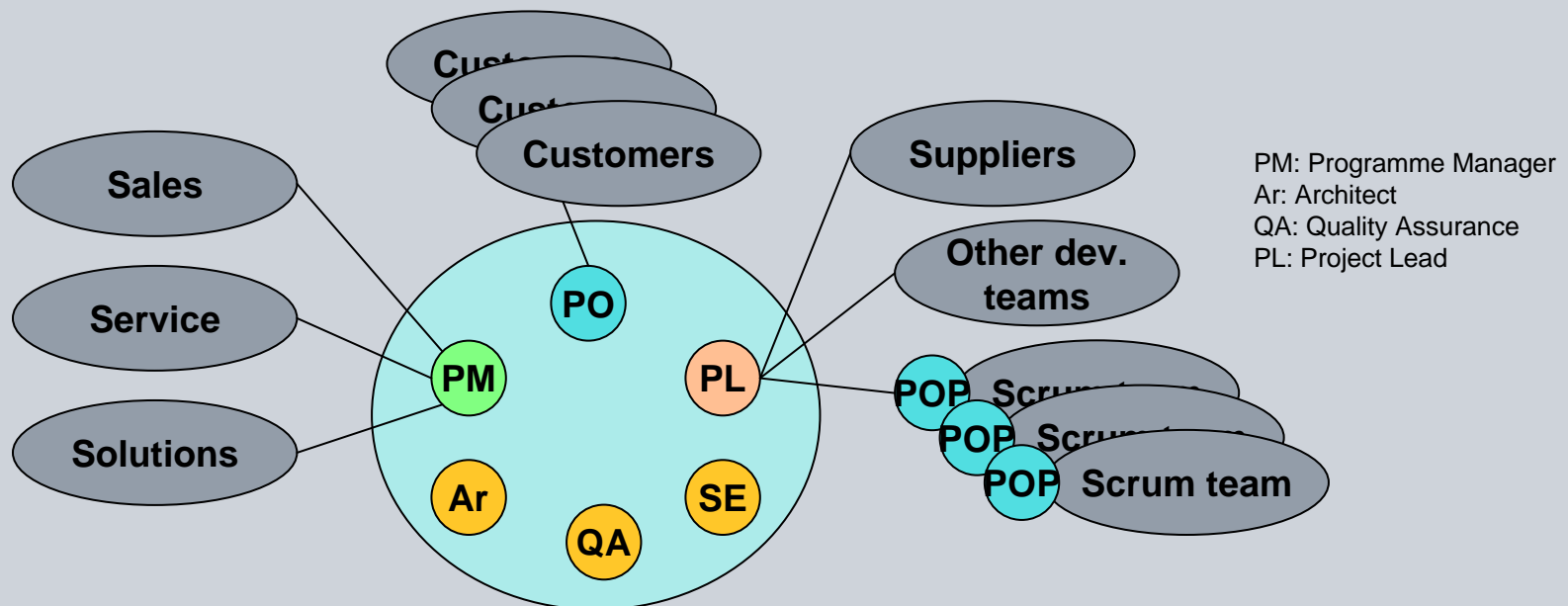
Co-laboration of PO and „traditional“ roles.

- **The problem:**

- Product Owners have a lot to do and tend to become unavailable.

- **Solution outline:**

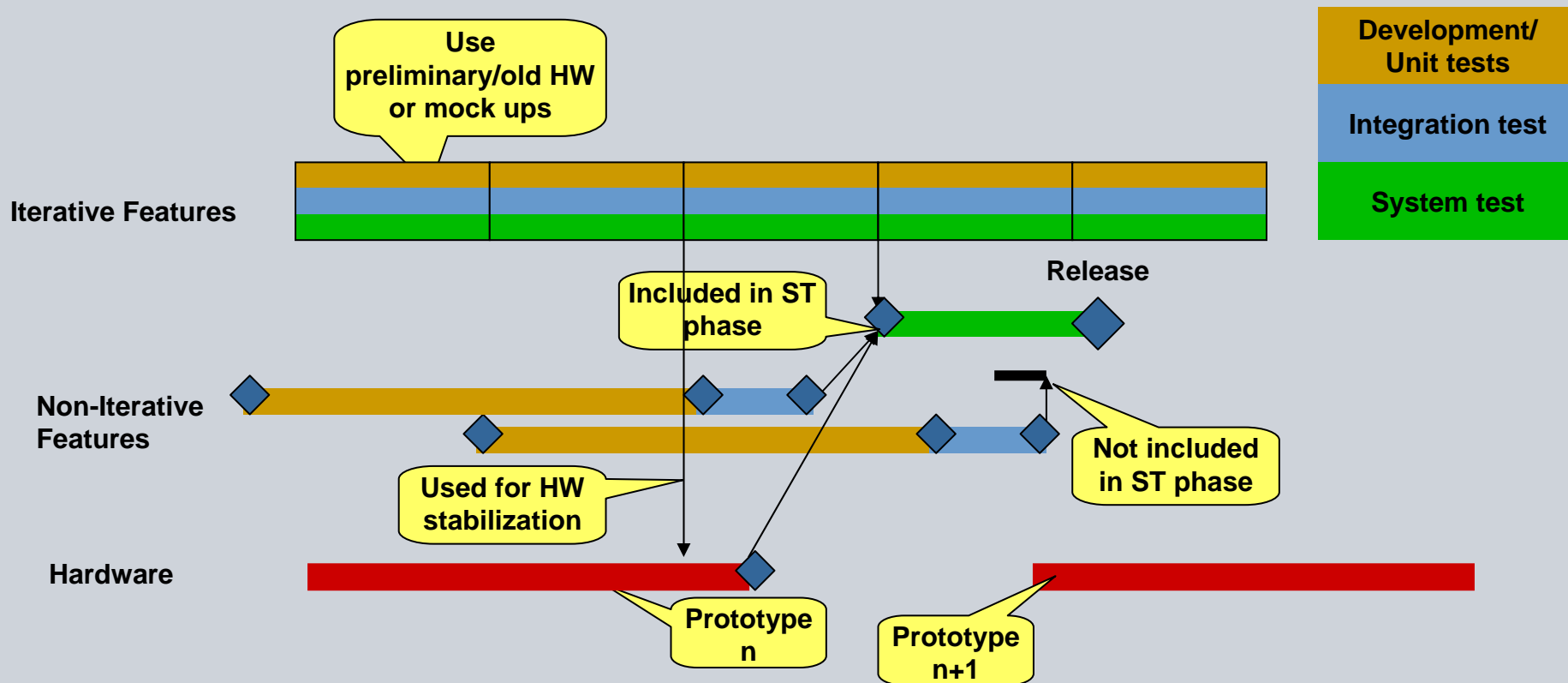
- The PO has the overall responsibility.
- The “traditional” roles support the PO and work together as a team.



Agile and the Product Lifecycle Process

Example: Integration with Non-Iterative Deliveries

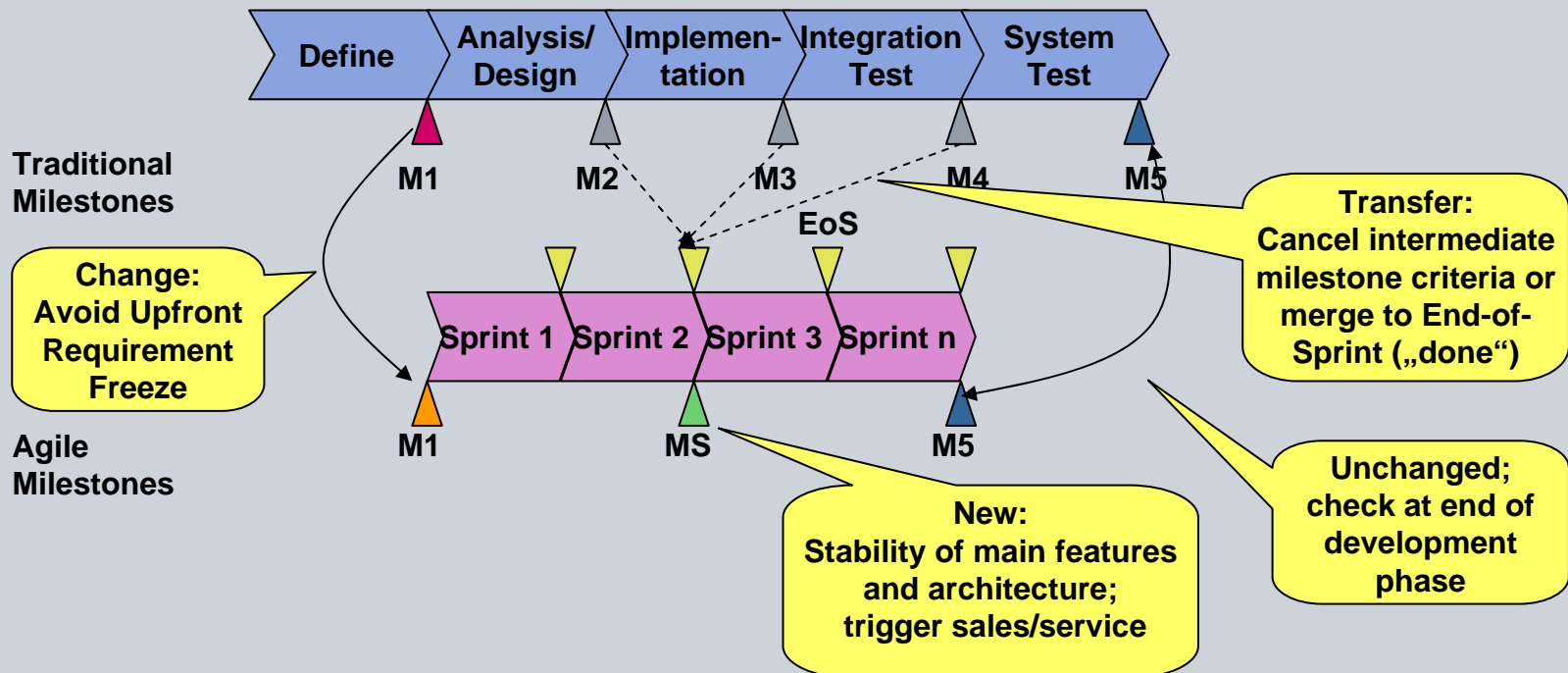
Common System test phase with clear entry criteria (milestones)



Agile and the Product Lifecycle Process

Defining Agile Milestones

- Agile Milestones can be derived from traditional ones (as suitable for the organization).
- End of Sprint “done” criteria have to cover required deliveries for product lifecycle management. Most of these deliveries are created iteratively



Agile Documentation

- **The problem:**

- Agile aims to reduce waste and focuses on generating business value.
- Documentation often does not create (direct) business value.
- An agile process requires has different (less) documentation needs.

- **Solution outline:**

- Don't say: "We are agile, we won't write documents any more"!
- Identify what information is necessary and useful, for whom and when (e.g. to support communication in distributed teams).
- Consider information needs beyond the project (and team) scope.
- Create the documents iteratively and in parallel to the software to keep them consistent.
- Include documentation in backlogs and plan accordingly.
- Use documentation to complement communication, not to replace it!

Agile Documentation

- **Examples for needed and useful documentation:**

- Everything required by (external) processes and regulations (e.g. FDA, ISO...)
- User stories and other forms of requirements (e.g. Use Cases)
- Architecture concepts and high level architecture (e.g. UML)
- Code inline documentation
- Documented test scripts (also for acceptance tests)

- **Sometimes not needed:**

- Extensive design specification (replaced by inline documentation and preliminary sketches)
- Test case specification (replaced by documented test scripts)

Conclusion

- **Going agile does not mean forgetting about everything that we have learned in the past.**
- **However, if you honestly want to live agility, don't make lousy compromises.**
 - A single bottleneck system test at the end of the project is not agile.
 - Renaming Project Leaders to Scrum Masters does not create a self-organizing team.

Preserve the agile principles and values.