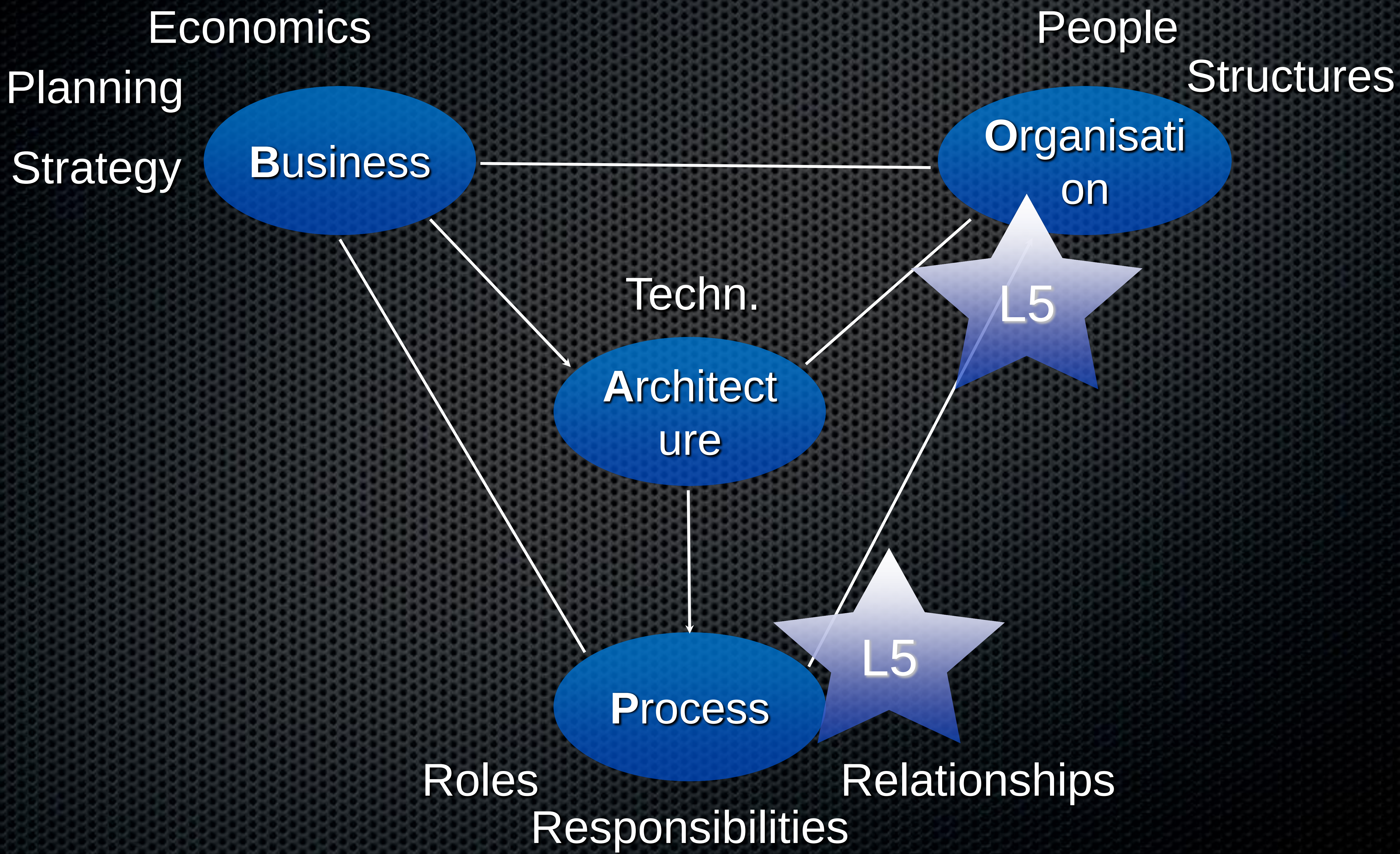
Software Product Line Engineering

L5:Organisations and SPL

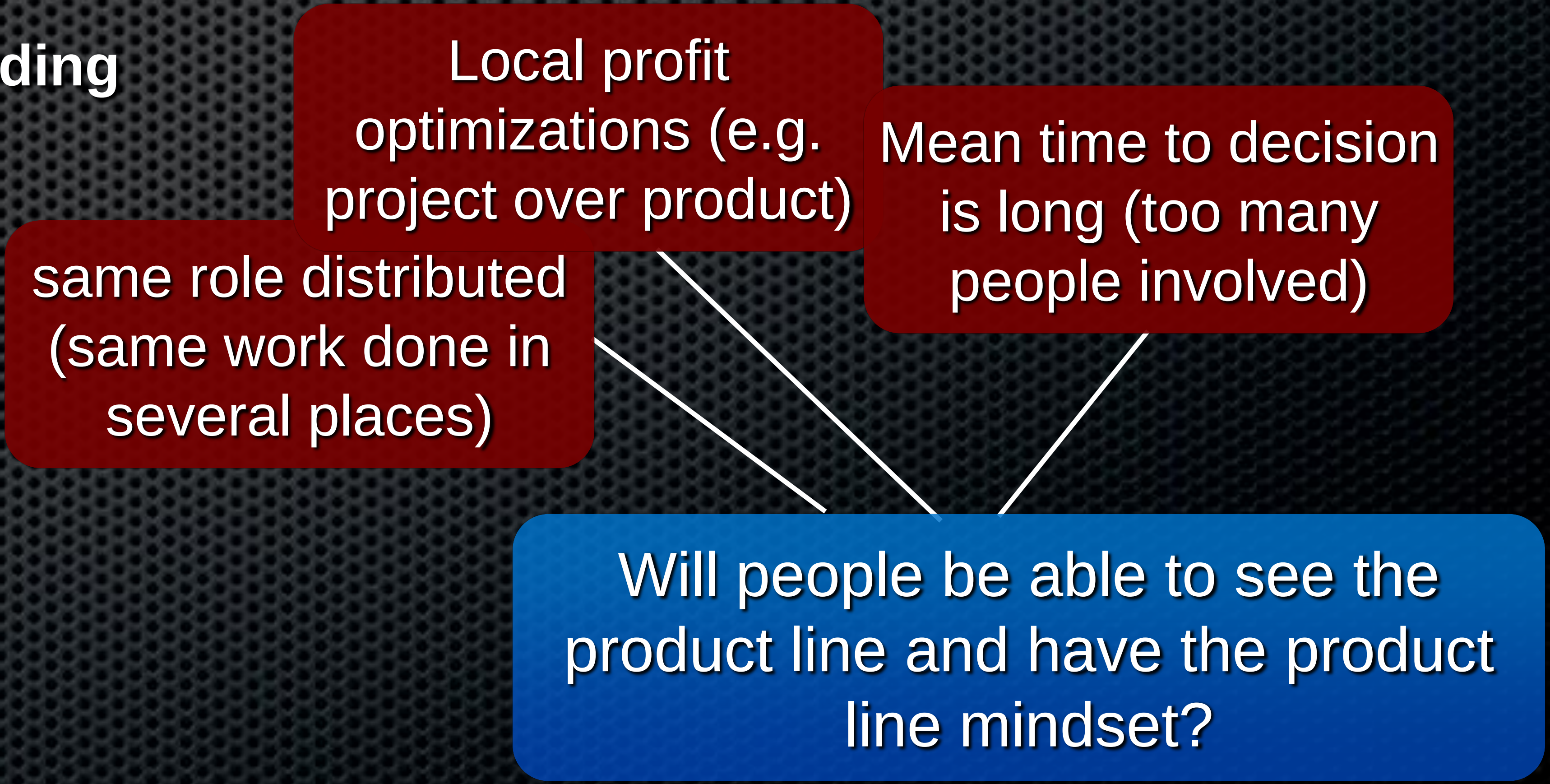
L5:Processes and SPL and Organizations



Organization, roles and responsibilities

why should we bother with this...

- **Mapping of activities (actions) and process and roles to organization is critical as it is central to the successful realization and use of a PL**
  - **Amount of people working together (coherence within unit vs. collaboration btw units)**
  - **Accountability and funding**
  - **Decision hierarchy**



Local profit optimizations (e.g. project over product)

Mean time to decision is long (too many people involved)

same role distributed (same work done in several places)

Will people be able to see the product line and have the product line mindset?

Organization, roles and responsibilities

why should we bother with this (2)...

- **Mapping of activities (actions) and process and roles to organization is critical as it is central to the successful realization and use of a PL**
  - **Organizational SIZE is crucial as it speaks to the impact of the organizational structure and the role and responsibilities division on the product line...**

Small organization has “closeness” and familiarity that can compensate for inadequacies, LARGE organizations DO NOT

“not my job”

Personal mind-set, and motivational structure plays a crucial role if a PL succeeds or not, much more so than having a perfect architecture variability analysis

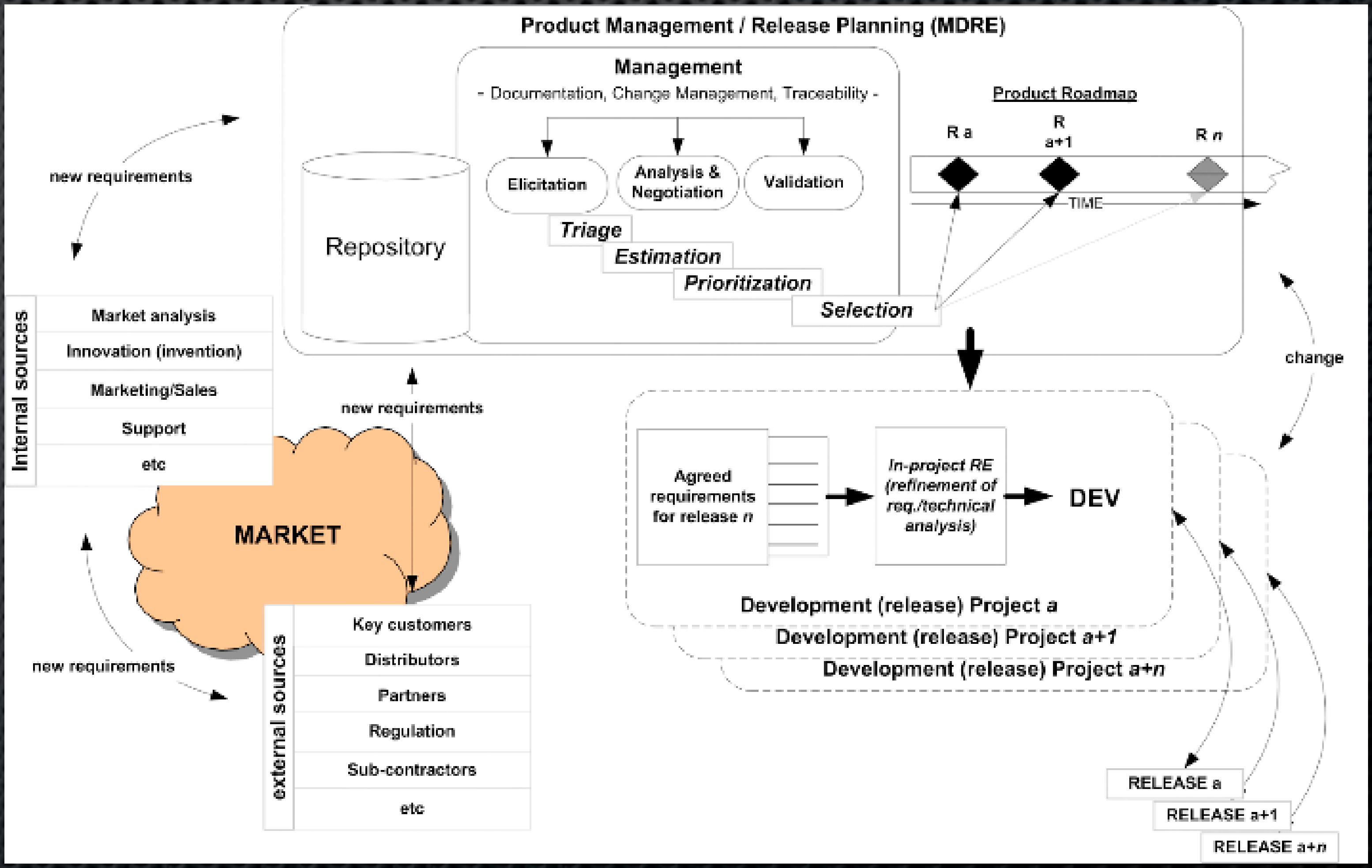
Imbalance in the organization (e.g. domination of application engineering over domain engineering)

What are individual engineers good at (like to do), skill set!
E.g. Domain Eng. (high quality components and maintenance) vs. App. Eng. (build apps fast w. given components)

Roles and responsibilities

- **Product Manager (PM)**

  - Planning and evolution of the complete range of products (present and future) taking features and BUSINESS value into consideration
  - Business value -> Business owner, Features -> marketing and sales
  - Domain requirements engineering -> evolution of the features (commonality and variability)
  - PM initiates application development and coordinates with the application requirements engineer



Roles and responsibilities 2

- **Domain Requirements Engineer**

  - Development and maintenance of the requirements that are relevant for the whole range of products (domain), i.e. the development of common and variable requirements incl. a variability model (in accordance with the roadmaps and plans of the PM)
  - Estimation and feasibility feedback
  - Common and variable req. + variability model -> input to domain architect

- **Domain Architect**

  - Development and maintenance of the reference architecture for the complete set of products
  - Collaborates a lot with the domain requirements engineer
  - The common and variable parts of the arch. are provided to the domain asset manager who performs management on variants and versions
  - Reference architecture -> input to domain developer (includes the selection of reusable domain components and interfaces)
  - The domain architect validates that the designs of the reusable assets fulfill/adhere to the reference arch.
  - To enable configuring, the domain arch. determines what configuration mechanisms should be used to build end products.
  - Domain architect validates application architectures - adherence to domain arch.

  + reference arch -> is used by the application architectures

Roles and responsibilities 3

- **Domain Developer**

  - Development and maintenance of reusable components and interfaces for the complete range of products
  - Development of configuration mechanisms (e.g. through parameters, on model/design level, on CM level (e.g. versions) etc) to support the variance of the systems in the product line

- **Domain Tester**

  - Development and maintenance of reusable test assets for the complete range of products
  - Testing of integrated products, but also integration and system tests on domain assets, and prepare common and variable test assets to be used by the application tester (make sure to plan what has to be tested from a domain perspective in the individual applications)
  - Domain tester -> input to RE (testability etc), -> to PM regarding costs, -> to architect and domain developer as to testability on domain level

- **Domain Asset Manager**

  - Maintaining versions and variants of all domain assets! (everything from requirements to test cases and executables)
  - Traceability and configuration control (-> e.g. versions of individual artifacts to application configurations are kept traceable and under CM control)
  - Large potential of overhead!

Roles and responsibilities 4

- **Application Requirements Engineer**

  - Development and maintenance of the requirements for a single product
  - Use present requirements, if not available create new application specific ones that are validated against the PM
  - Submit suggestions for candidate domain requirements
  - Application RE -> supplies selected requirements application architect and developer, and asset manager gets list for CM purposes

- **Application Architect, Developer, Tester**

  - Specific application
  - Reuse what is possible from the domain level, develop what is needed for the application level
  - Validate against Domain PM and Architect as to adherence to domain assets and architecture
  - Suggest additions (alterations for new variants) to domain level artifacts
  - Early estimation of impact and cost (short and long-term) - not only development but product line impact and cost...

Organizational structures

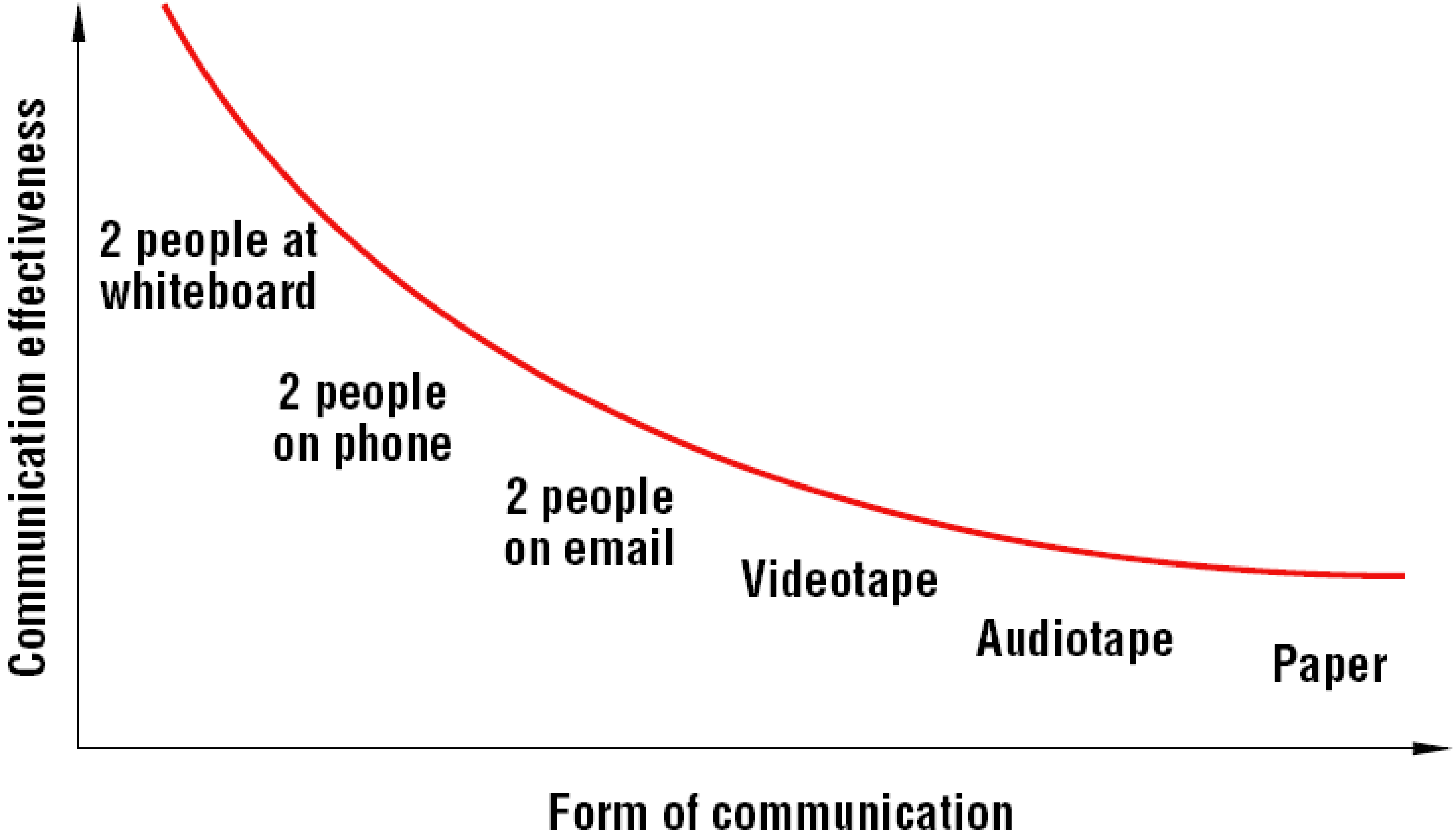
- The way people interact can be captured in communication patterns. The patterns determine what kinds of mechanisms are used for communication and by whom

- Communication patterns are influenced by organizational structure, as it dictates what information needs to be communicated to whom, and who is concerned with what part (functionality wise) and aspect (life cycle perspective)

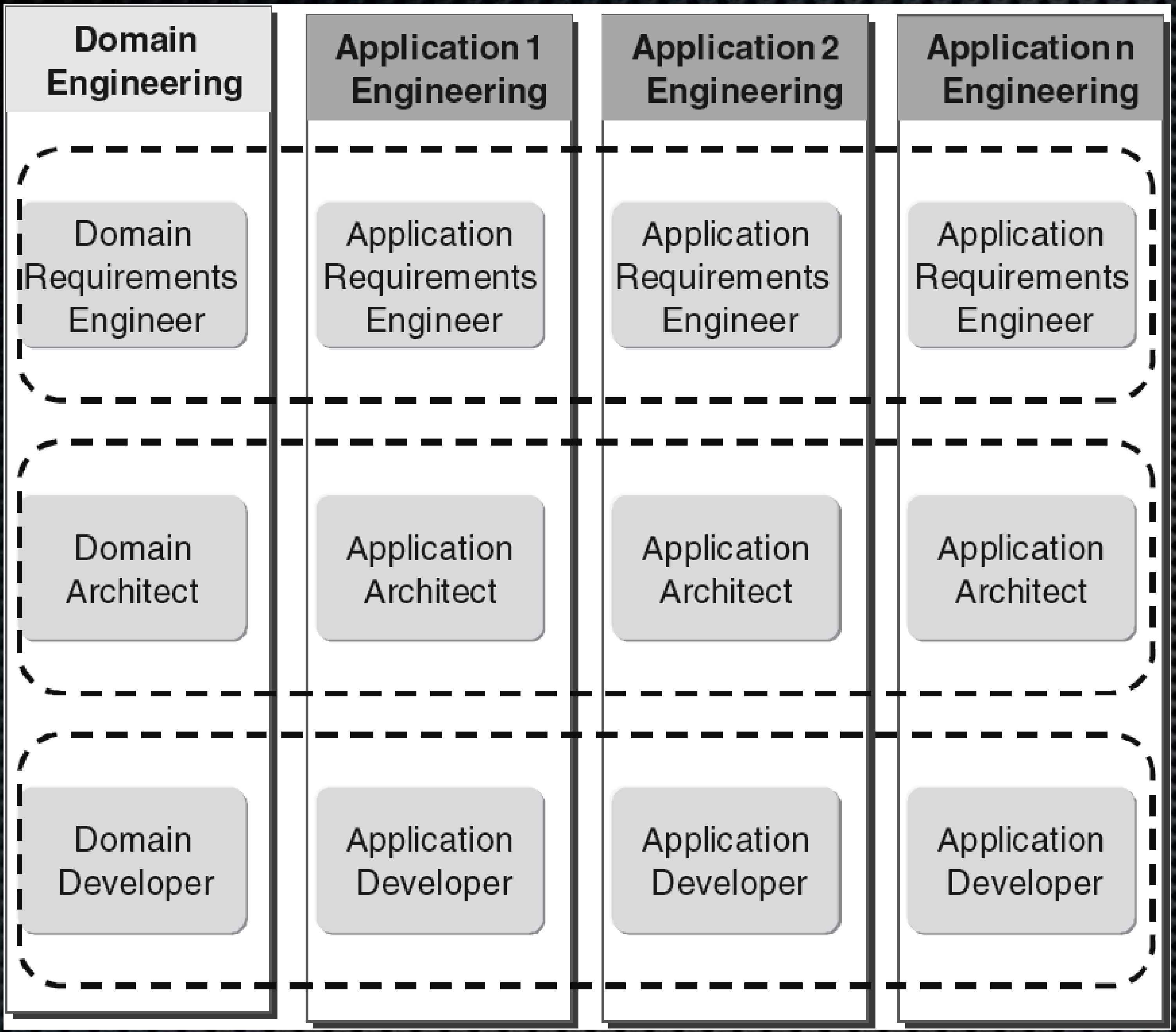
- Organizational structures for PL are linked with roles and responsibilities:
  - Domain and Application engineering - go through a development life-cycle (sequence or in parallel)
  - Interactions btw domain and application engineering are on functional level (requirements, design, realization, test level)
  - Domain asset manager interacts with most engineering roles
  - Product Manager provides input to domain engineering and initiates application engineering

domain and application engineering and their interaction influence organizational structure the most

PM, Asset manager, testing lead to additional structure


Communication effectiveness
2 people at whiteboard
2 people on phone
2 people on email
Videotape
Audiotape
Paper
Form of communication

Product-Oriented Organizations



Domain Engineering	Application 1 Engineering	Application 2 Engineering	Application n Engineering
Domain Requirements Engineer	Application Requirements Engineer	Application Requirements Engineer	Application Requirements Engineer
Domain Architect	Application Architect	Application Architect	Application Architect
Domain Developer	Application Developer	Application Developer	Application Developer

Most common type of organization

Clear division of responsibility and accountability (domain vs application and for each application)

Application units are responsible for obtaining income

Division btw applications can be dependent on both similarity (e.g. one type of applications in same part and/or market targeted)
A key is to have communication heavy parts in the same unit

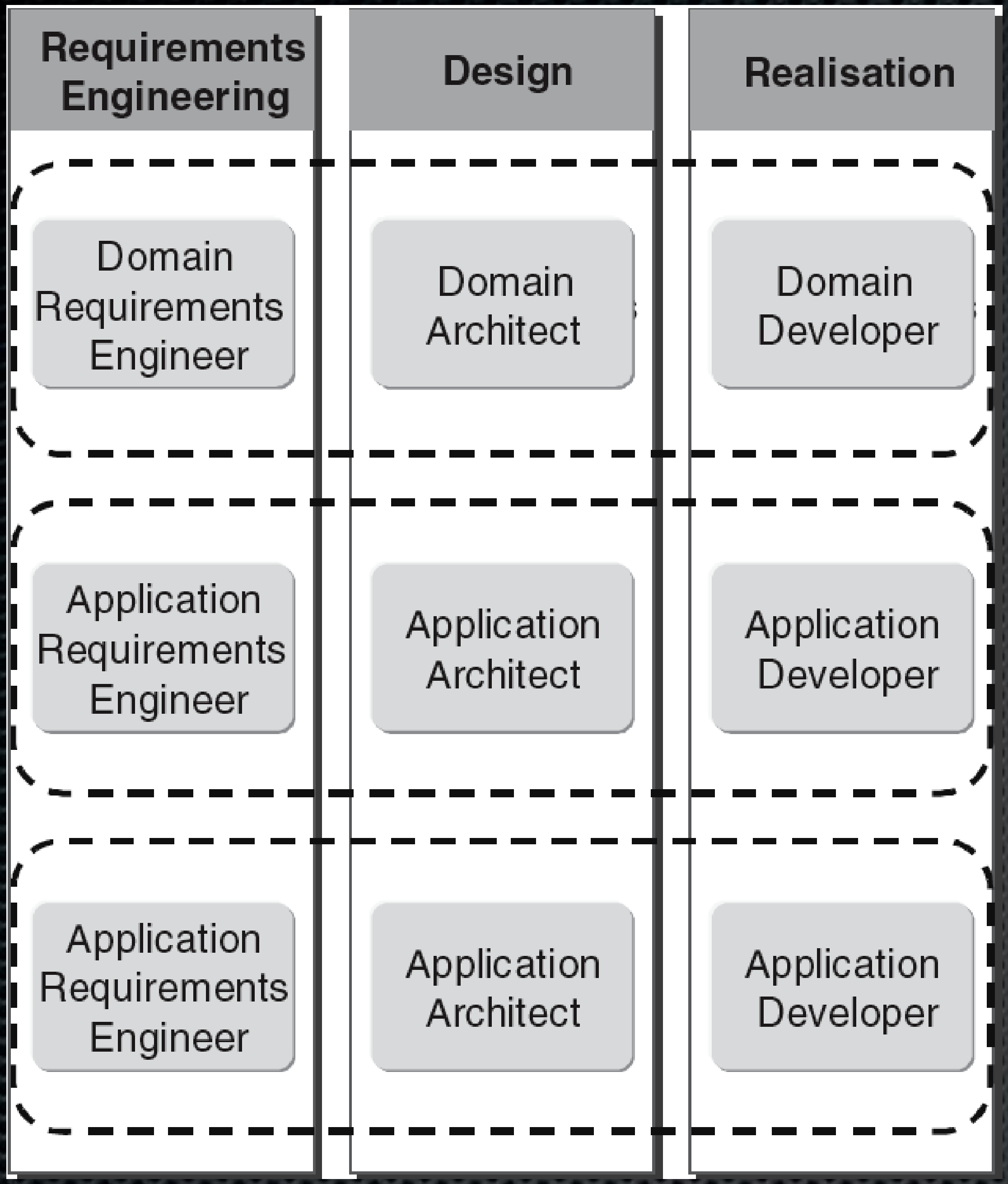
Main challenges:
- Funding the domain unit
- Functional interactions btw developers of different units (also for e.g. architects)

(especially during formation of the PL) app units tempted to go outside the company for the platform

communication btw units considered as overhead (also sometimes as competition)

double development!

Process-Oriented Organizations



Requirements Engineering	Design	Realisation
Domain Requirements Engineer	Domain Architect	Domain Developer
Application Requirements Engineer	Application Architect	Application Developer
Application Requirements Engineer	Application Architect	Application Developer

Functional hierarchy is prime!

Functional interaction is facilitated

Flexible allocation of resources depending on need (btw application but also btw domain and application)

People develop similar functionality for different products:
- Easier to ensure integrity of architecture
- Focus on reusability as it benefits you...

more common in smaller organizations where communication is less of a problem

Main challenges:
- Different phases of engineering are not close
- Domain engineering spread out

communication btw units and planning is necessary

accountability (especially for domain assets is not clear)

Matrix Organizations

	Domain Engineering	Application -1 Engineering	Application -2 Engineering	Application -n Engineering
Requirements Engineering	Domain Requirements Engineer	Application Requirements Engineer	Application Requirements Engineer	Application Requirements Engineer
Design	Domain Architect	Application Architect	Application Architect	Application Architect
Realisation	Domain Developer	Application Developer	Application Developer	Application Developer

Compromise btw product and process focus

Main challenges:
- Scattered focus
- Complex management