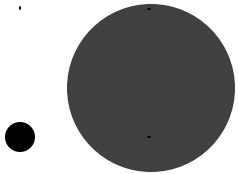
Hochschule Konstanz
Fakultät Informatik

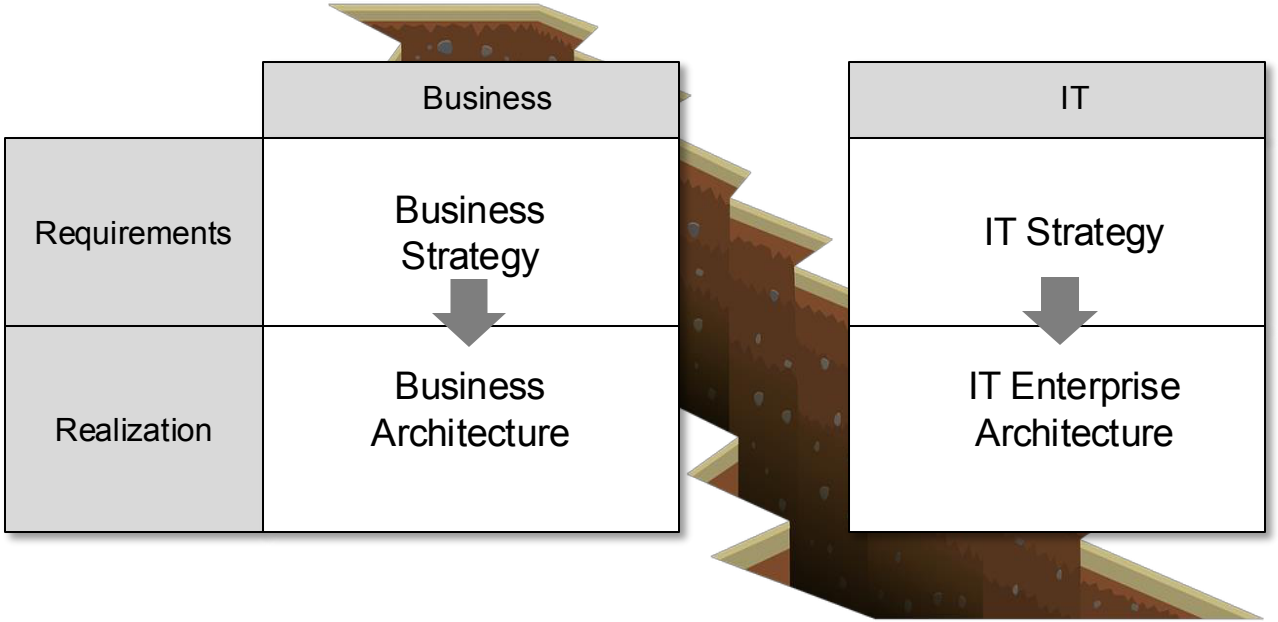
IT Management
Chapter 3

- Enterprise Architecture Management Patterns

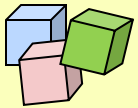


Business-IT-Alignment with Capabilities

The Business-IT-Gap



What are Business Capabilities?



A business capability is an abstract statement of **what** work is being done in a certain area, such as „pay employees“.

- Implementations of a capability represent how that work is done in terms of people, process and technology.
- How it gets done changes often in most companies, but what is getting done is comparatively far more stable
- Capabilities can be implemented with technology (IT) or they can be implemented as manual tasks without any IT support.

Example:

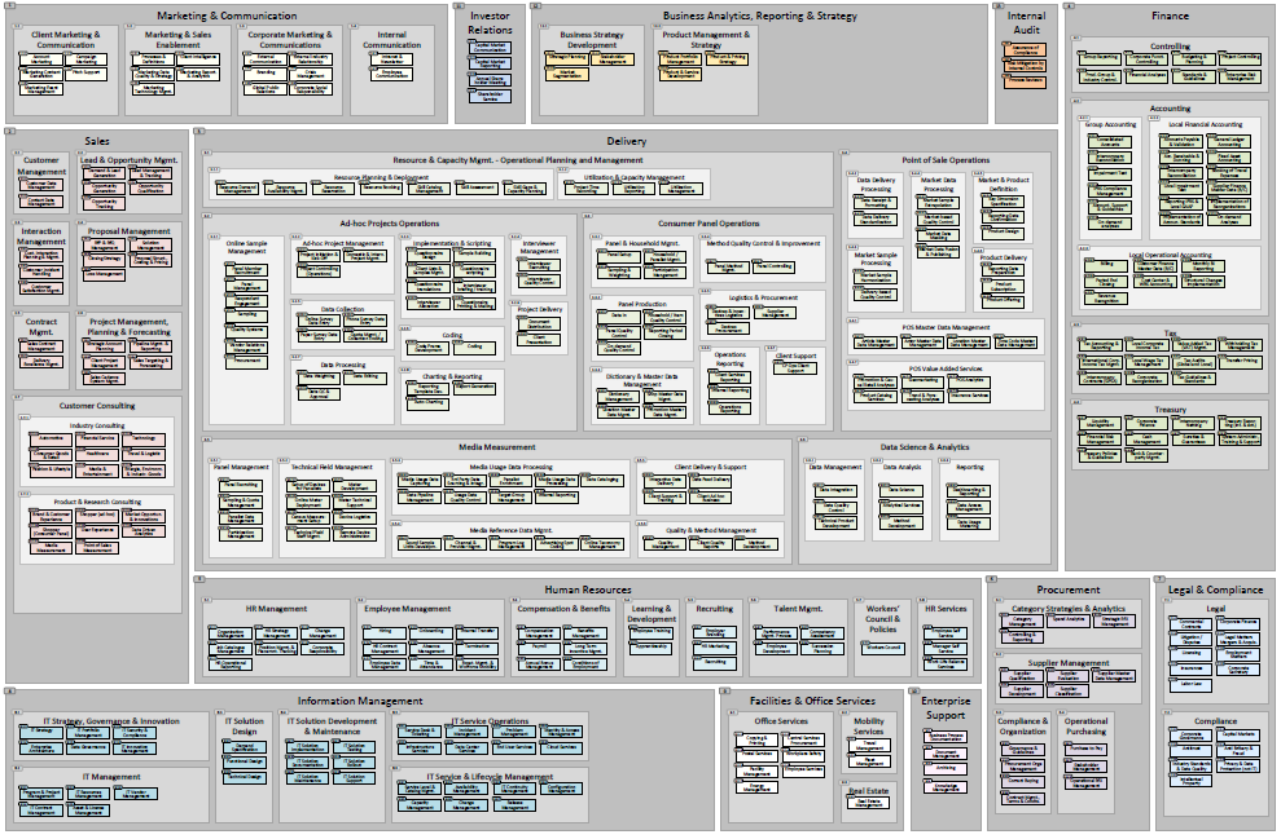
Capability (“what?”) — **Pay employee**

Organization (“who?”) — HR

Process (“how?”) — Payroll process

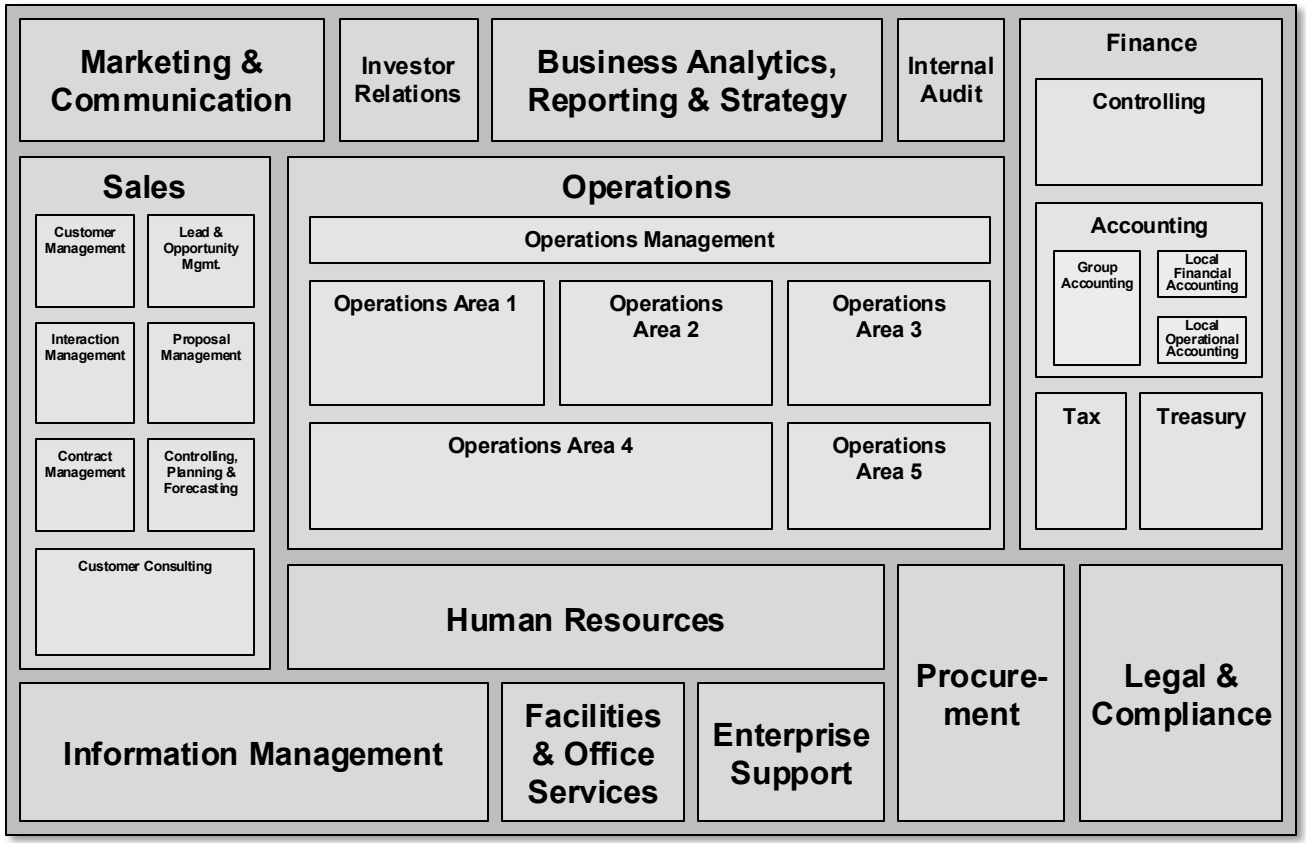
Technology (“with what?”) — HR system

Capability Map Example (*) (Levels 1 to 4)



(*) Real-life example, therefore blurred intentionally

Capability Map Example



Design Guidelines for Capabilities

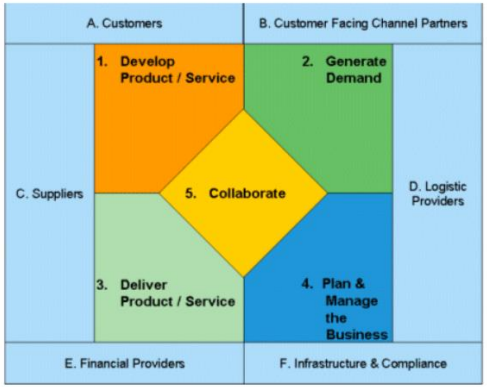
1. **No overlap**: Capabilities must be mutually exclusive
2. **Complete**: All business functions of the organization must be covered by capabilities

"MECE" = mutually exclusive and collectively exhaustive

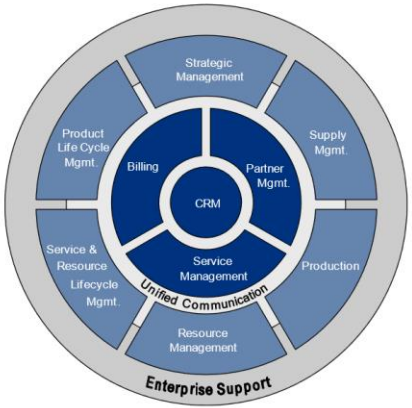
3. **Stability**: Capabilities should be stable over a long period of time
4. **Independence**: Capabilities must be independent from organizational structures, processes, persons and technology
5. **Hierarchy**: Capabilities should be structured hierarchically (with approx. 7-10 capabilities on the highest level)
6. **Broad vs. deep:** Capabilities must cover as much of the organization as possible without going into too much detail → The focus is on broadness and not on depth
7. **Include stakeholders**: Capabilities must be aligned with all relevant stakeholders and as many stakeholders as possible should agree on the developed set of capabilities
8. **Clear responsibility**: Capabilities should be managed by a single organizational unit or a single person within the organization. Avoid „diversity of variants"!
9. **Common language**: Capability names must be aligned with the language of the organization. I.e., capability names must use established terminology.

How to get a good Capability Map

1. If a capability map is good or bad depends on the questions an organization would like to answer with the map
2. Capability map must be designed **individually for each organization**
3. **Capability map templates** for specific sectors are a good starting point (→ reference models)



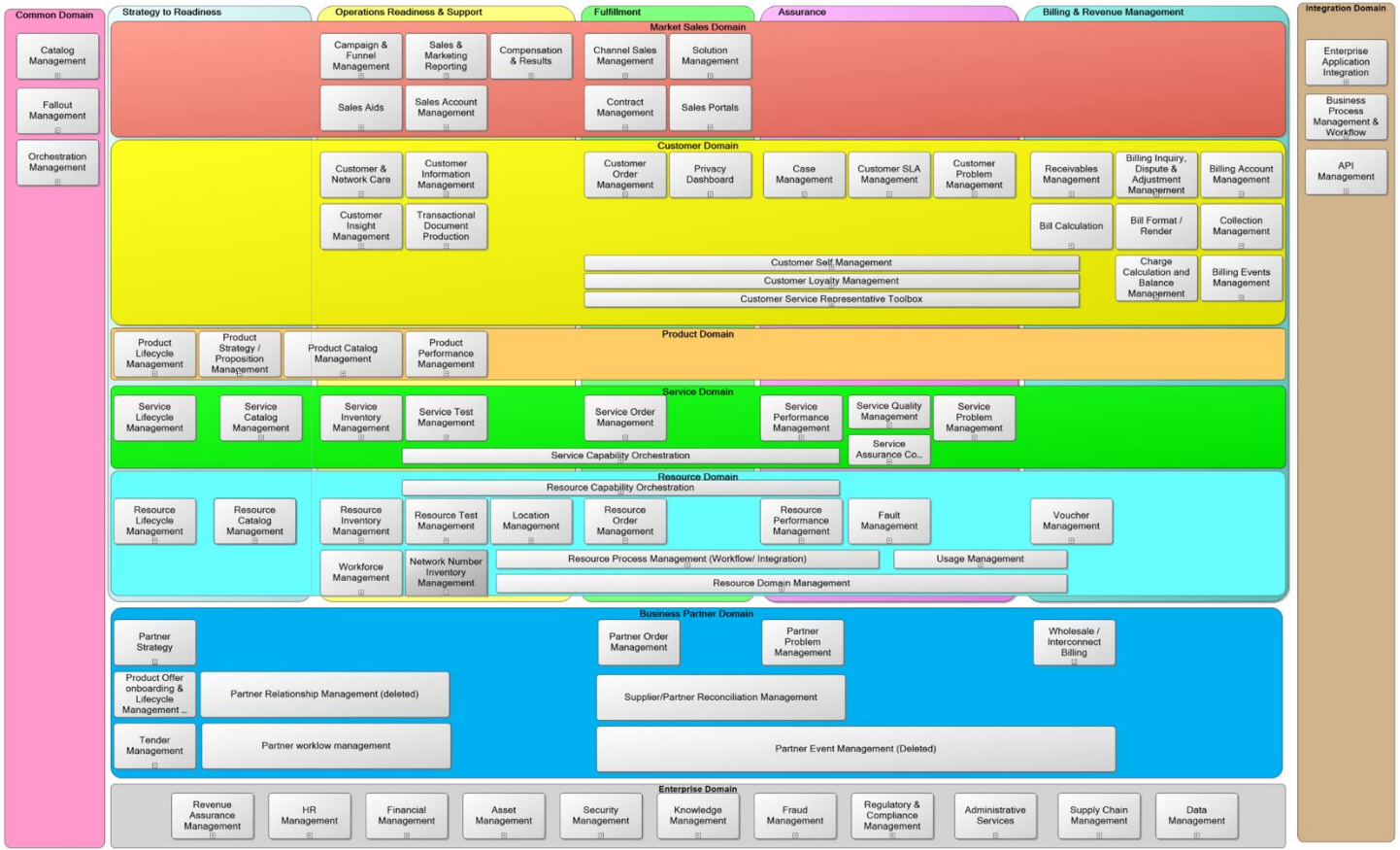
Generic capability map from Microsoft
(Source: „IT-Unternehmensarchitektur“, W. Keller, 2017)



High-level capability map of Deutsche Telekom AG (Source: https://docplayer.org/2163106-Fallstudie-deutsche-telekom-ag-einheitliche-datenarchitektur-als-grundlage-fuer-unternehmensweites-datenqualitaetsmanagement.html)

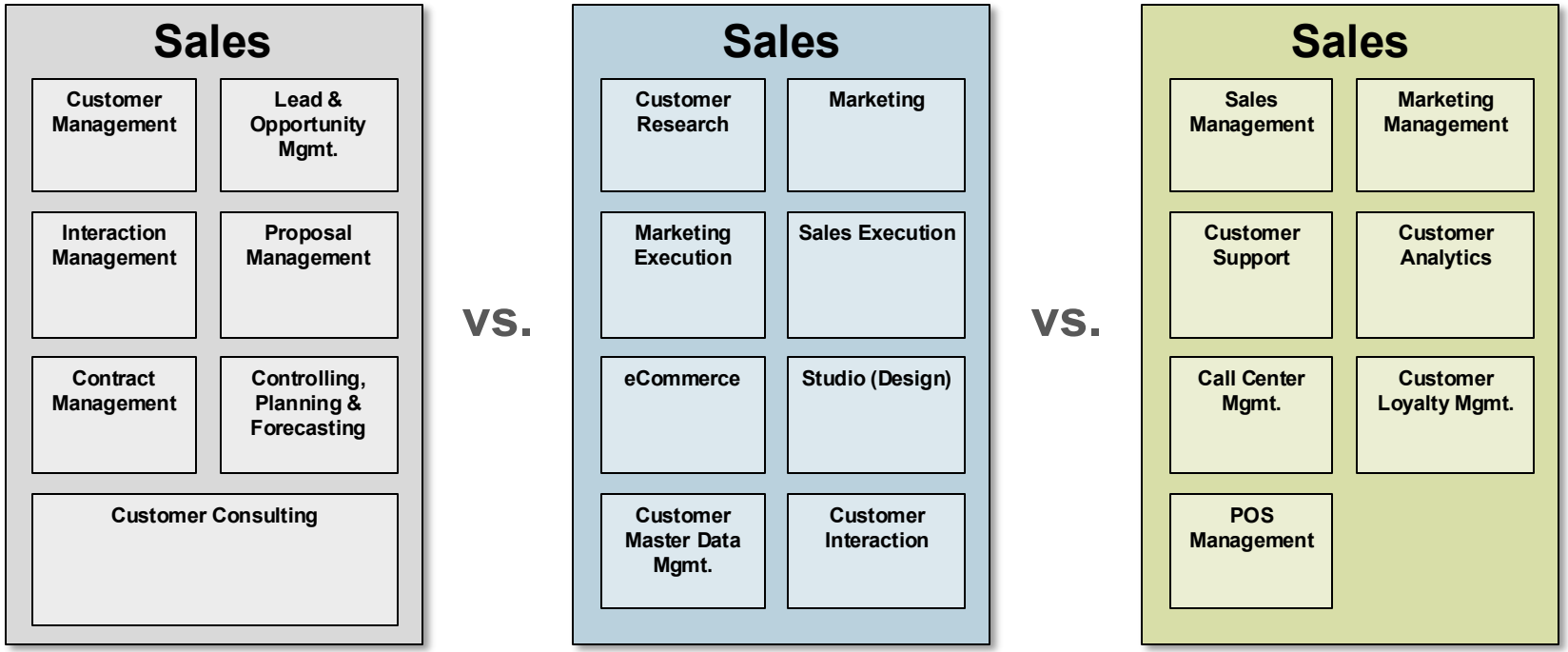
4. Even generic capability map templates are available
5. Consulting companies usually have more detailed templates available (but they keep them a secret)
6. Layout of a capability map is an individual decision (most organizations use layouts with hierarchical boxes)
7. Different organizations within the same sector (e.g., banking) may have ended up with totally different capability maps depending on their priorities, strategy, language, …

Reference Model Example: Telecom Operation Model (eTOM)

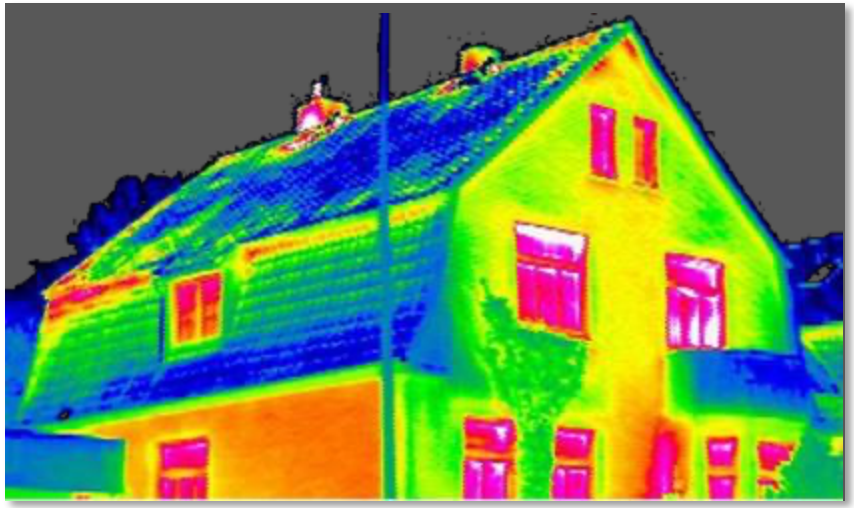


Source: http://www.casewise.tmforum.org/evolve/statics/tmfmodel/#/cwtype=index&cwview=index_diagrams_tam_start&lang=en (2022-10-30)

Capability Maps are unique for each Organization
Example: 3 different Views on the „Sales“ Business Function



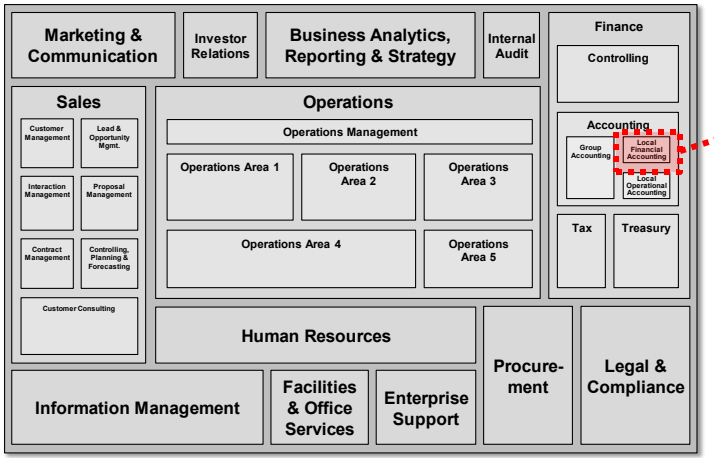
Termographic Inspection of a House: „Heat Map“



- „Heat Map“ shows areas with critical isolation of a house
- It helps homeowners with investment decisions for improving energy efficiency of the house

Source: https://carelabz.com/wp-content/uploads/2017/11/Thermographic-testing-thermographic-inspection.jpg

Define Capability Scores (Example)



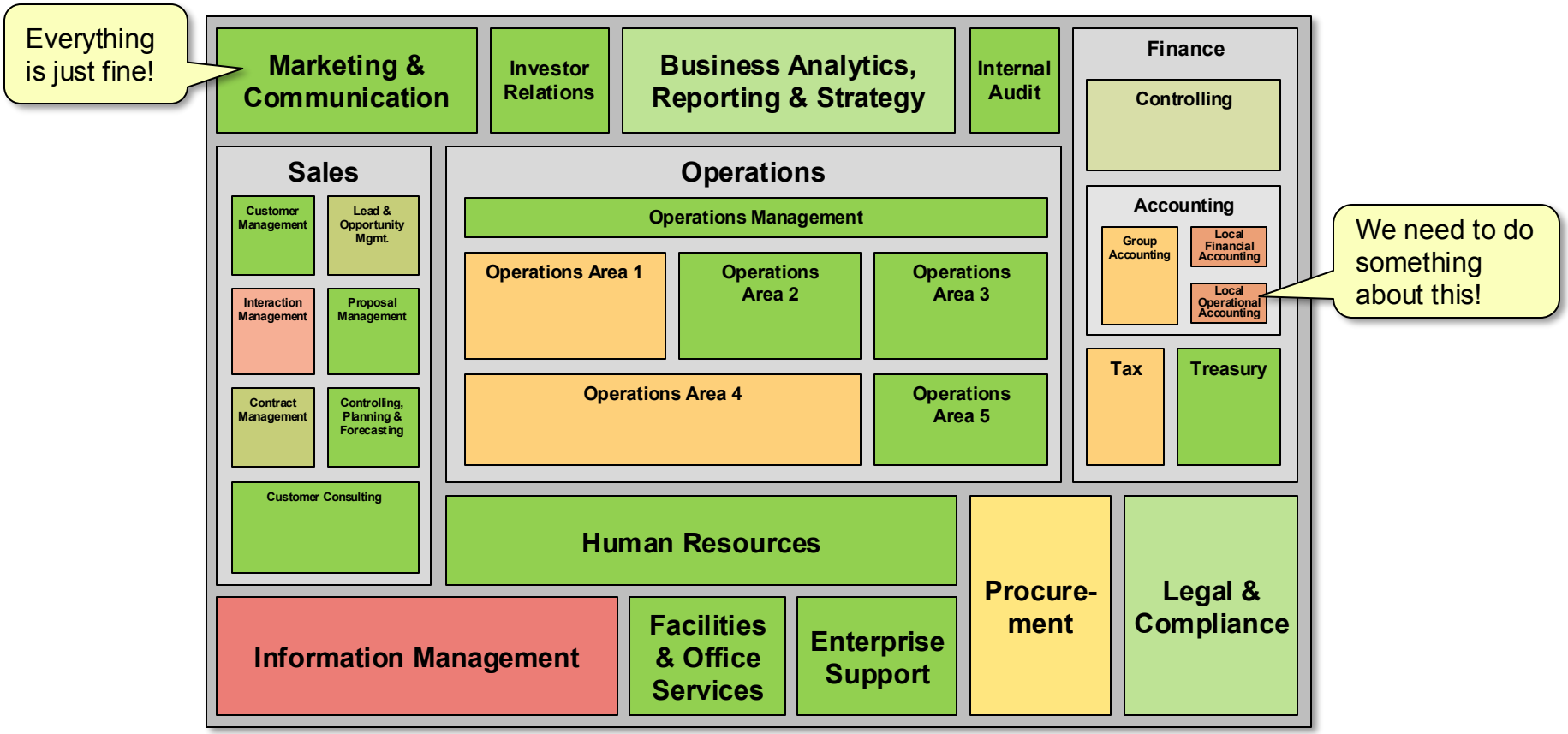
"Local Financial Accounting"

Criteria	Value
Strategic value	low
Quality of (IT) implementation	high
Quality of business processes	high
Compliance status	medium
Cost situation	bad
(other…)	…

Color on map:

Please note: There are no rules for how to calculate capability scores. Each organization can define their scoring method and criteria individually.

„Heat Map“ Example

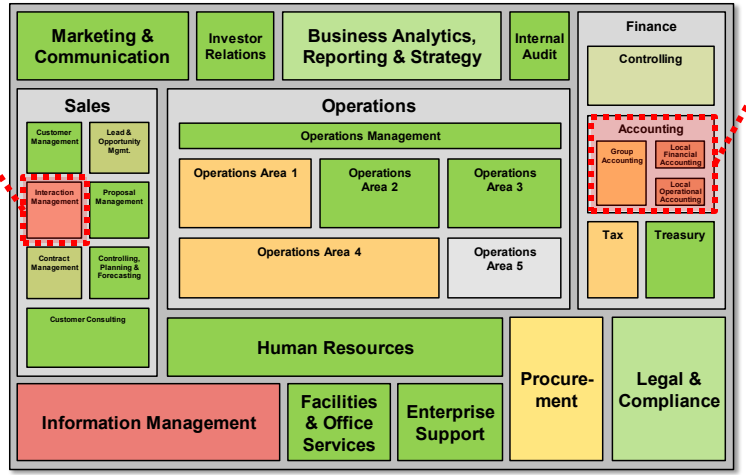


Using „Heat Maps“ to improve Business-IT-Alignment (Example)

„Interaction Management“

- Managing customer interactions has high strategic importance
- Current IT support and process quality is poor
- Cost situation is bad (very low budget available)

IT underinvestment:
Improve IT support, e.g., implement state-of-the-art CRM software

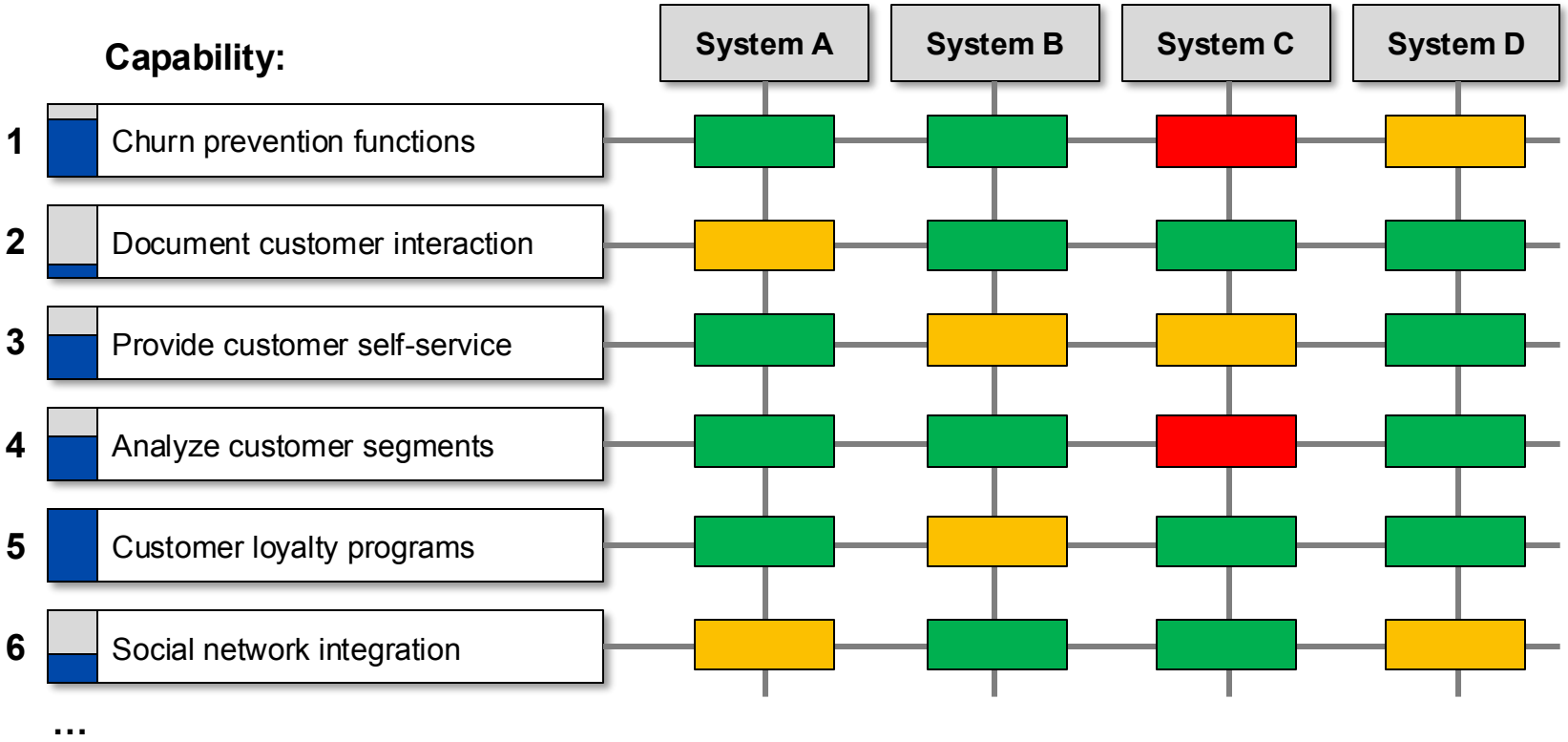


„Accounting“

- Low strategic importance
- Very good processes and excellent IT implementation
- Bad cost situation (very high license costs)

IT overinvestment:
Replace very expensive accounting software with cheaper system or even outsource function as a service

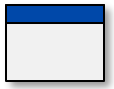
Using Capabilities to compare IT Systems



Application Portfolio Assessment

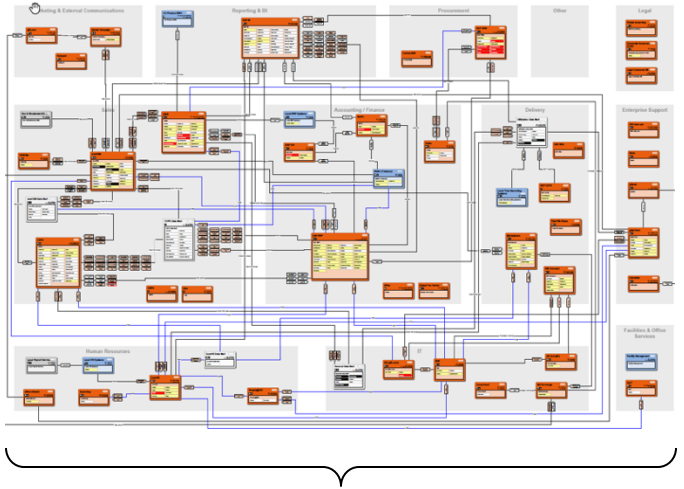
Management of Application Portfolio

Portfolio: In finance, a portfolio is a collection of investments held by an investment company, hedge fund, financial institution or individual.



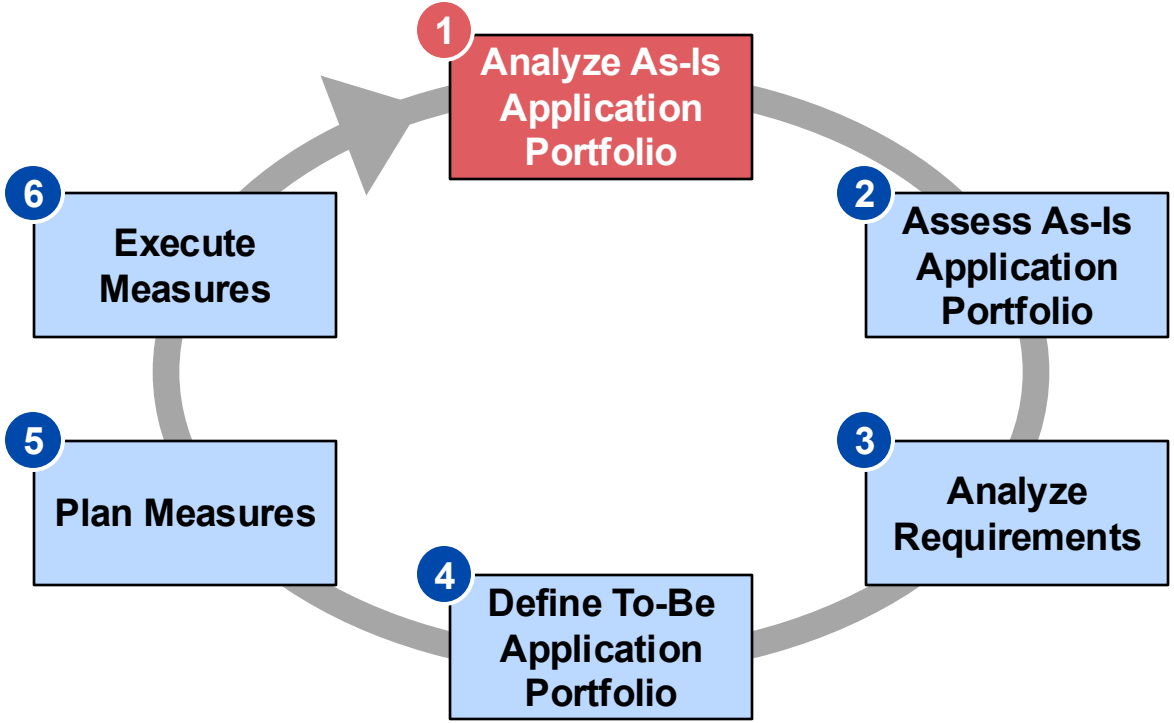
Application: A set of versioned program modules, data structures, tables (including the corresponding documentation) implementing a defined functionality.

Application Portfolio: Applications are inventoried and documented in the application portfolio by an organization.

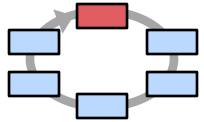


Application landscape with >50 applications

Management of Application Portfolio

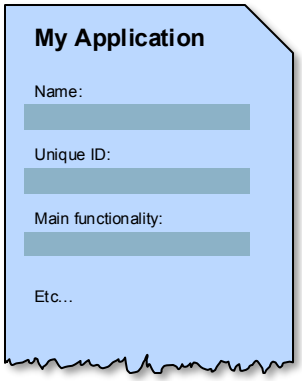


Analyze As-Is Application Portfolio



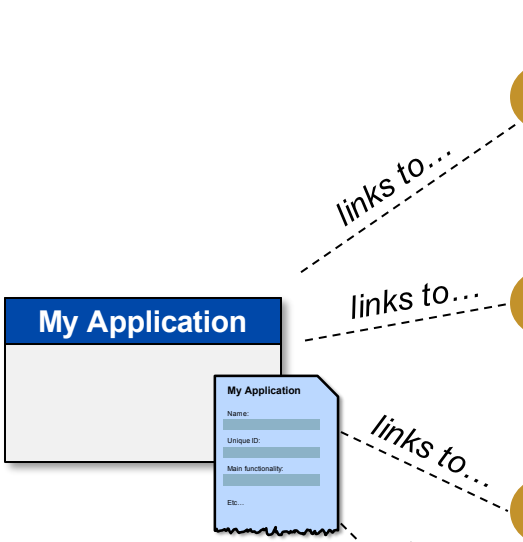
Why should you create and manage an application portfolio?

Create Application Profiles:
Examples for what can be specified for each application…

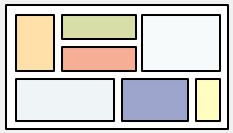


- **Name** of application
- …

Describing Applications on different Architecture Levels

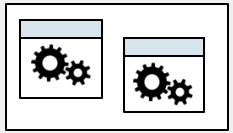


A



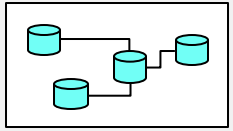
Business Architecture *specifies the business functions, processes and organizational structures required to meet the business strategy.*

B



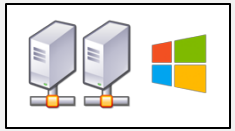
Application Architecture *describes the IT applications, the platforms they are built with and the design principles that are used to develop an application*

C



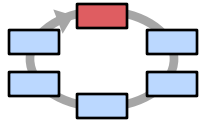
Data Architecture *describes the structure of an organization's logical and physical data assets and data management resources.*

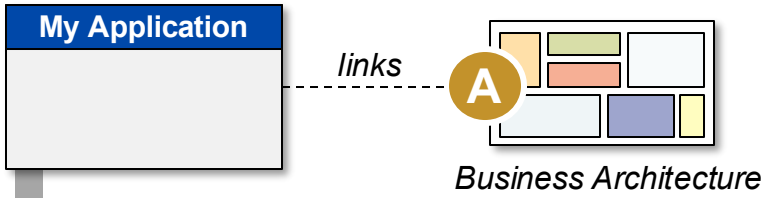
D



Infrastructure Architecture *describes the physical and virtual assets required to run applications and to exchange data between applications*

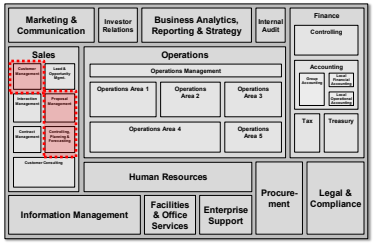
Application on Business Architecture Level





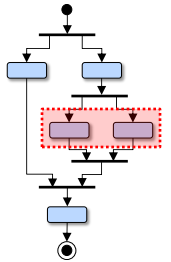
Business Capabilities

- Main business capabilities that are supported by the application and that are actually used in the application
- Example: Application X supports capability „*Manage customer loyalty program*“, but this functionality is not used by the organization

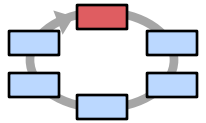


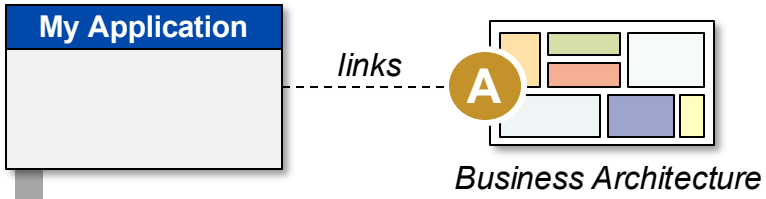
Business Processes

- Business processes supported by the application
- Example: Application supports „Order-to-Cash“ process



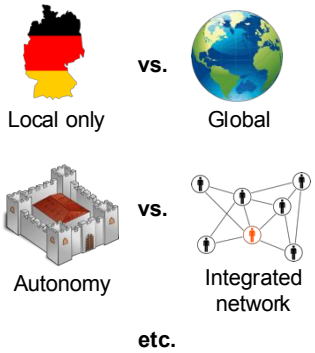
Application on Business Architecture Level





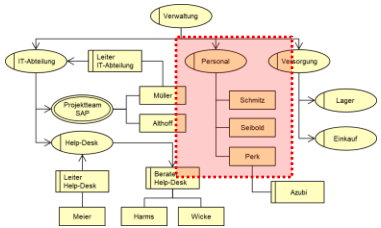
Strategic Goals

- Elements from business strategy and/or IT strategy supported by application
- Example: Application supports IT strategy element „global applications for all legal entities“

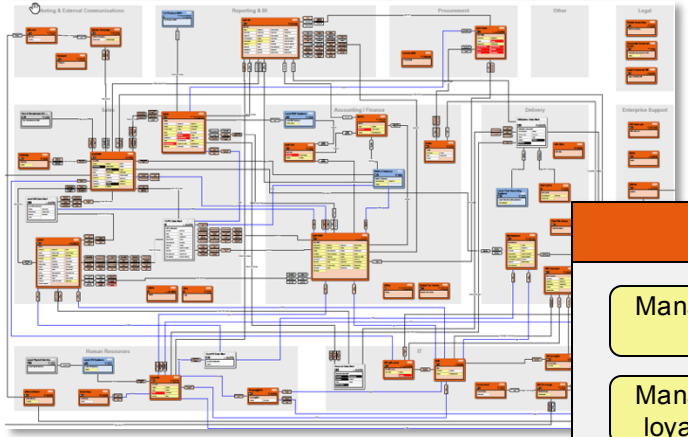


Organization

- Organizational units that use the application
- Exampe: App is used by HR department in Germany and UK



Business Architecture: Map Business Capabilities to Applications

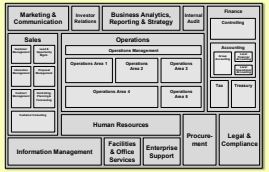


CRM-System

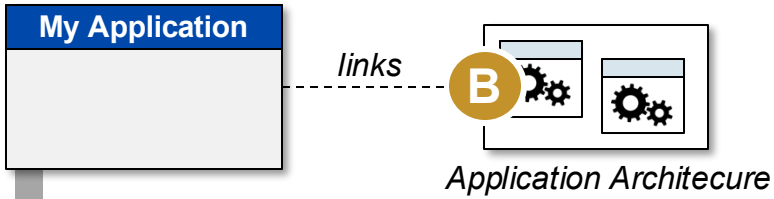
- Manage customer data
- Manage sales offers
- Manage customer loyalty programs
- Manage business opportunities
- Analyze customer segments
- Manage contracts

...

Capabilities from business capability map:



Application on Application Architecture Level



Interfaces

- Main interfaces of the application and data that is exchanged with each interface
- Description of each interface on business level and/or technical level

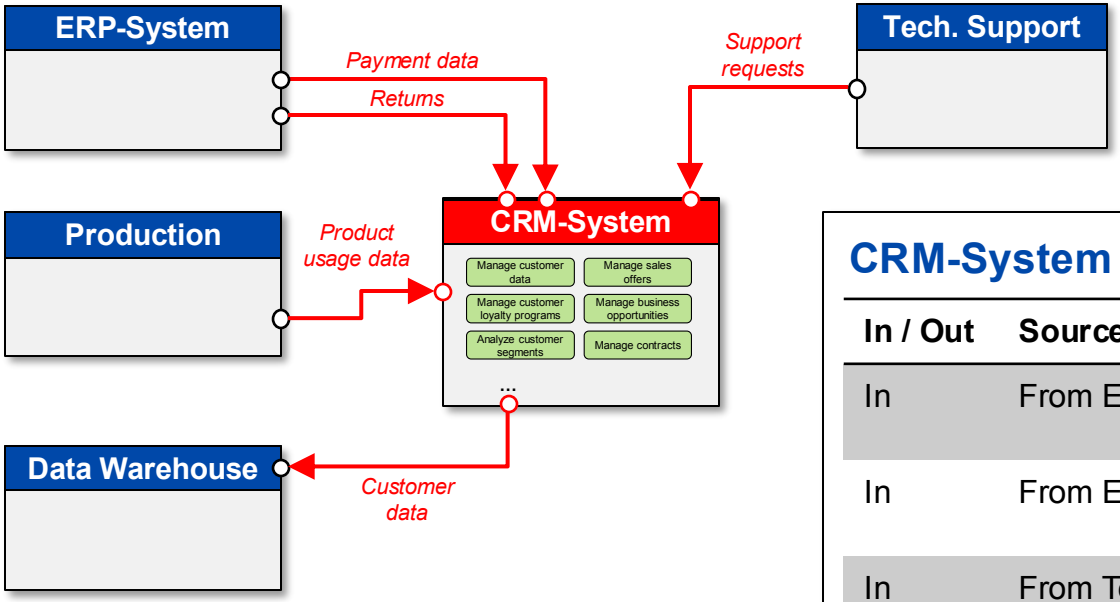
Design Principles

- Details about how is the application built
- Example: commercial application from a specific vendor, Software-as-a-Service (SaaS), in-house development, SOA with list of provided services, …

IT platforms

- Platforms the application is running on
- Example: .NET, Java, Oracle database, MySQL, Windows, MacOS, mobile devices vs. desktop PC, …

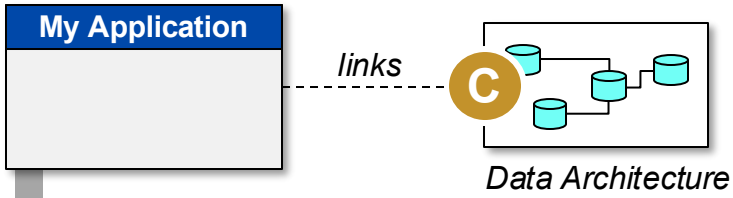
Application Architecture: Documenting Application Interfaces



CRM-System Interfaces:

In / Out	Source / Target	Data	Frequency
In	From ERP	Payment data	1x per day
In	From ERP	Customer returns	Daily, 2am
In	From Tech. Support	Support requests	Continuously
Out	To Data Warehouse	Customer data	Daily, 3-4am
…			

Application on Data Architecture Level



Business Objects

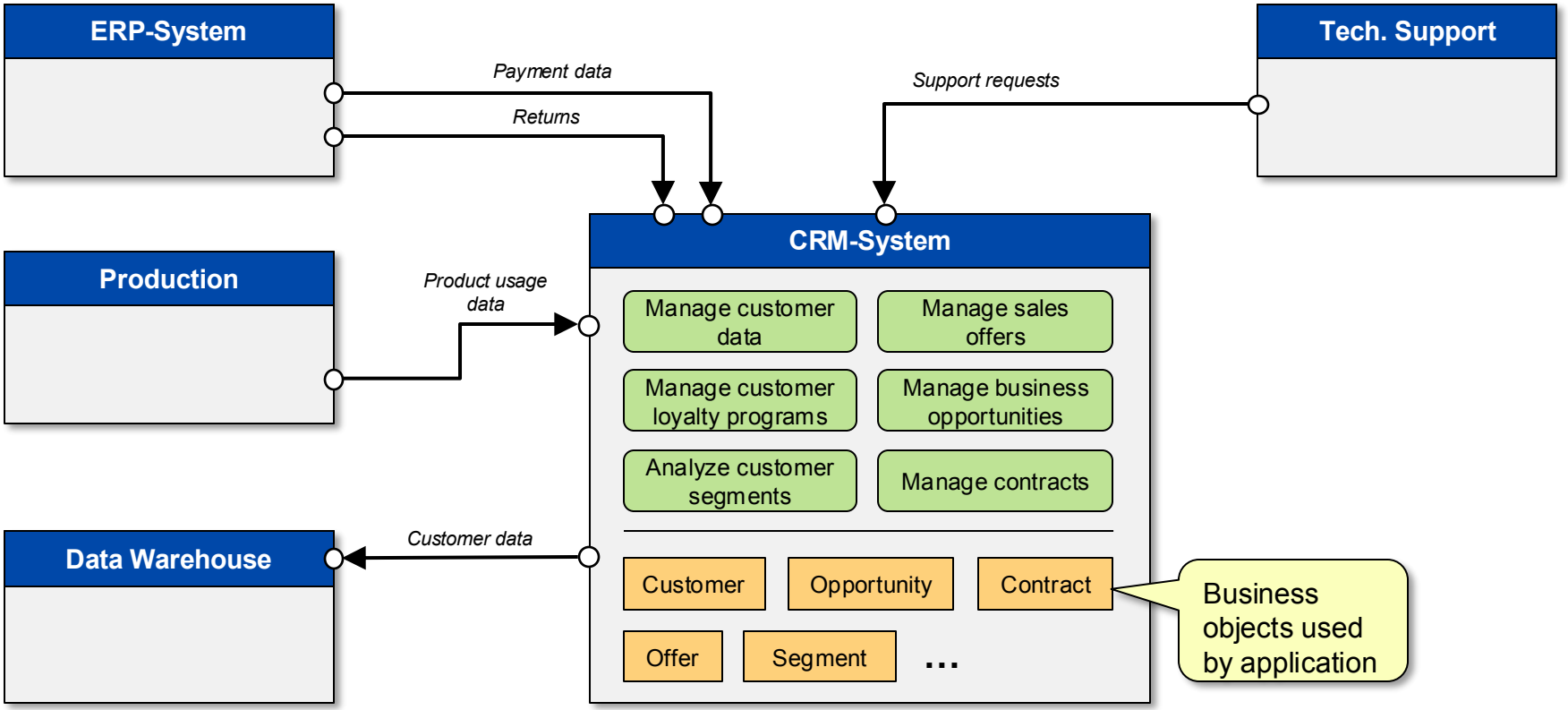
- Business Objects that are managed and/or used by the application
- CRUD for each relevant business object
- **Example:** Application X creates, updates and reads customers, but cannot delete customers
- **Example:** Application X reads product data (→ requires information about where product data comes from because it is not created in the application)

C: Create
R: Read
U: Update
D: Delete

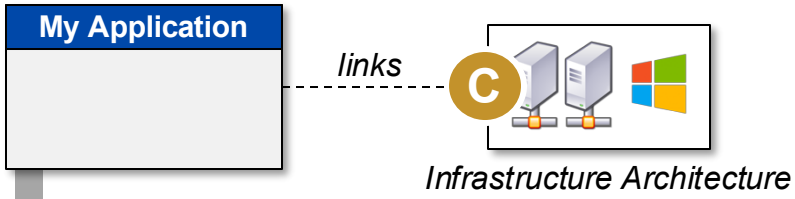
Data Security

- Describe which person, organization and/or other application can access which data

Data Architecture: Document major Business Objects per Application



Application on Infrastructure Architecture Level



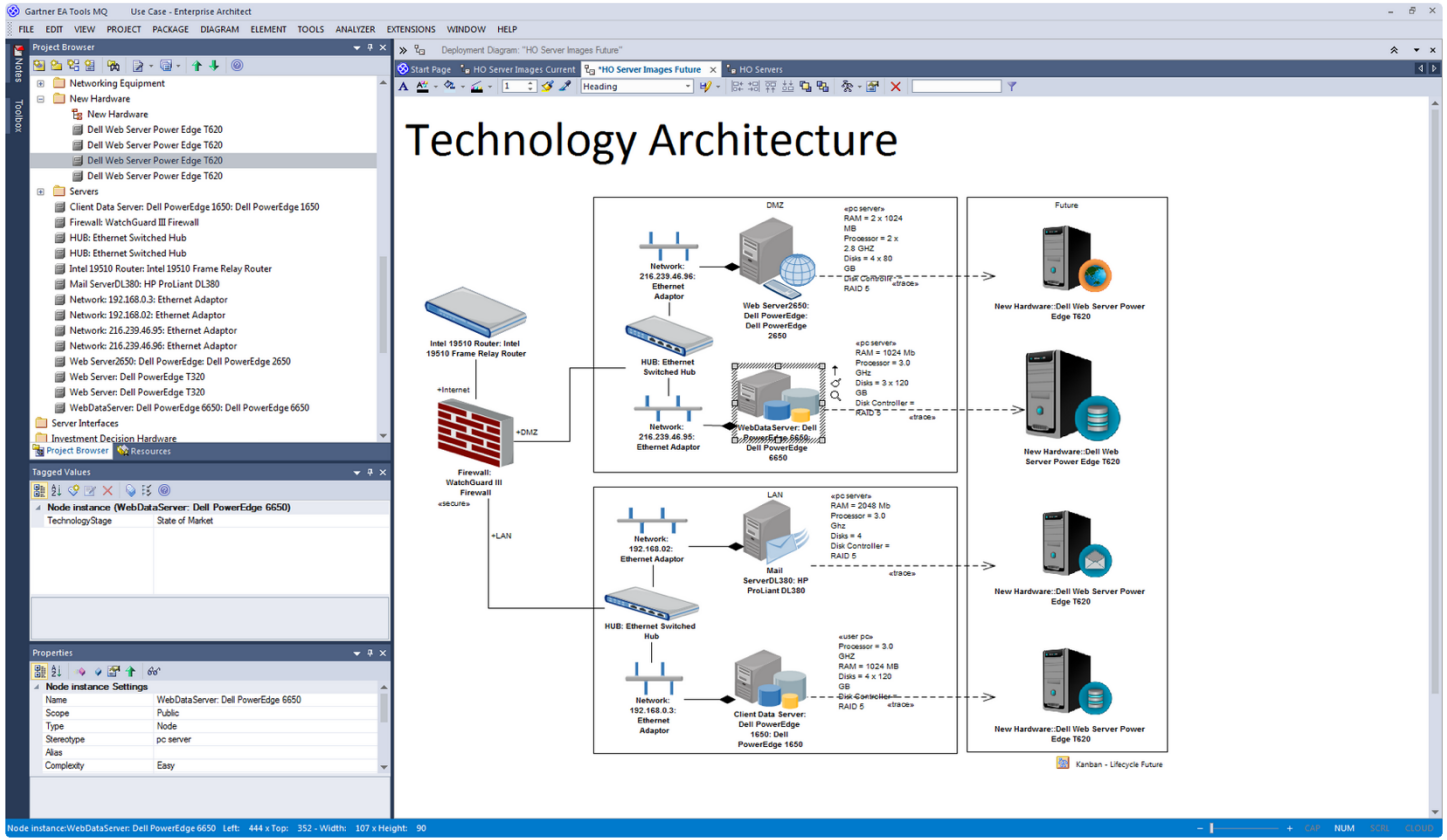
Environment & Locations

- Details about the technical environment and the location the application runs in
- Example: Application X is deployed on a Windows Server in Frankfurt. It uses an Oracle database hosted on a Linux server in Berlin.

Devices

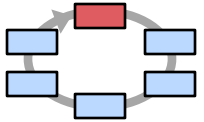
- Devices used by the application and their technical details
- E.g., server with RAM, CPU, disc capacity, network connection, …

Application on Infrastructure Architecture Level



Source: https://sparxsystems.com/domains/enterprise-architecture/enterprise-architecture.html

Best Practices for Application Portfolio Management



1. Begin as-is application portfolio with **small number of attributes** per application
2. **Decentralize** creation and maintenance of application portfolio
3. Assign person to each application who is responsible for keeping the application record up to date („**business owner**“)
4. **Excel or Sharepoint** list is a good starting point for creating an application portfolio
5. Use **professional EAM tools** once the application portfolio grows and more people are involved in maintaining the portfolio
6. Establish **incentive system** for maintaining the application portfolio (or constraints for neglecting portfolio management, e.g., budget constraints)
7. Ensure that **support from management** is available for maintaining an application portfolio
8. When appropriate use **lists with predefined values** for selected attributes, e.g., list of business capabilities, departments, employees, subsidiaries, countries, etc.
9. Track date of last update for each application