Avancier

Avancier Methods (AM) Enterprise Architecture

Analyse and Rationalise Application Portfolio

Avancier

Motivation: the need for app rationalisation

- Solution architects are often driven to meet the immediate requirements of individual business units, using parochial technologies.
- This leads to…
  - tactical stand-alone information systems.
  - a fragmented and disparate application portfolio
  - with duplication and waste of resources.

Do you have this?

- Edited from IT Business Edge 2010

Do you have app portfolio manager?

The mess and the 4 Ds

- An early motivation for EA was simply to “tidy up the mess”.
- Many large enterprises have acquired large portfolios of applications.
- The proliferation of silo applications has led to the 4 Ds:
  - Duplications of data and processes
  - Dis-integrities between data in different systems
  - Delays in completing processes
  - Difficulties with cross-organizational data analysis.
- And to the issues the 4 Ds create.

AM level 2 processes – with an EA perspective

Avancier

Manage

Manage requirements, the process, risks etc

- Manage stakeholders
- Manage requirements
- Manage business case
- Manage readiness & risks

Initiate

Identify requirements and constraints, agree the vision

- Establish capability
- Establish the context
- Scope the endeavour
- Get vision approved

Architect

Develop a target architecture

- Understand the baseline
- Review initiation products
- Clarify NFRs
- Design the target

Plan

Plan migration from baseline to target state

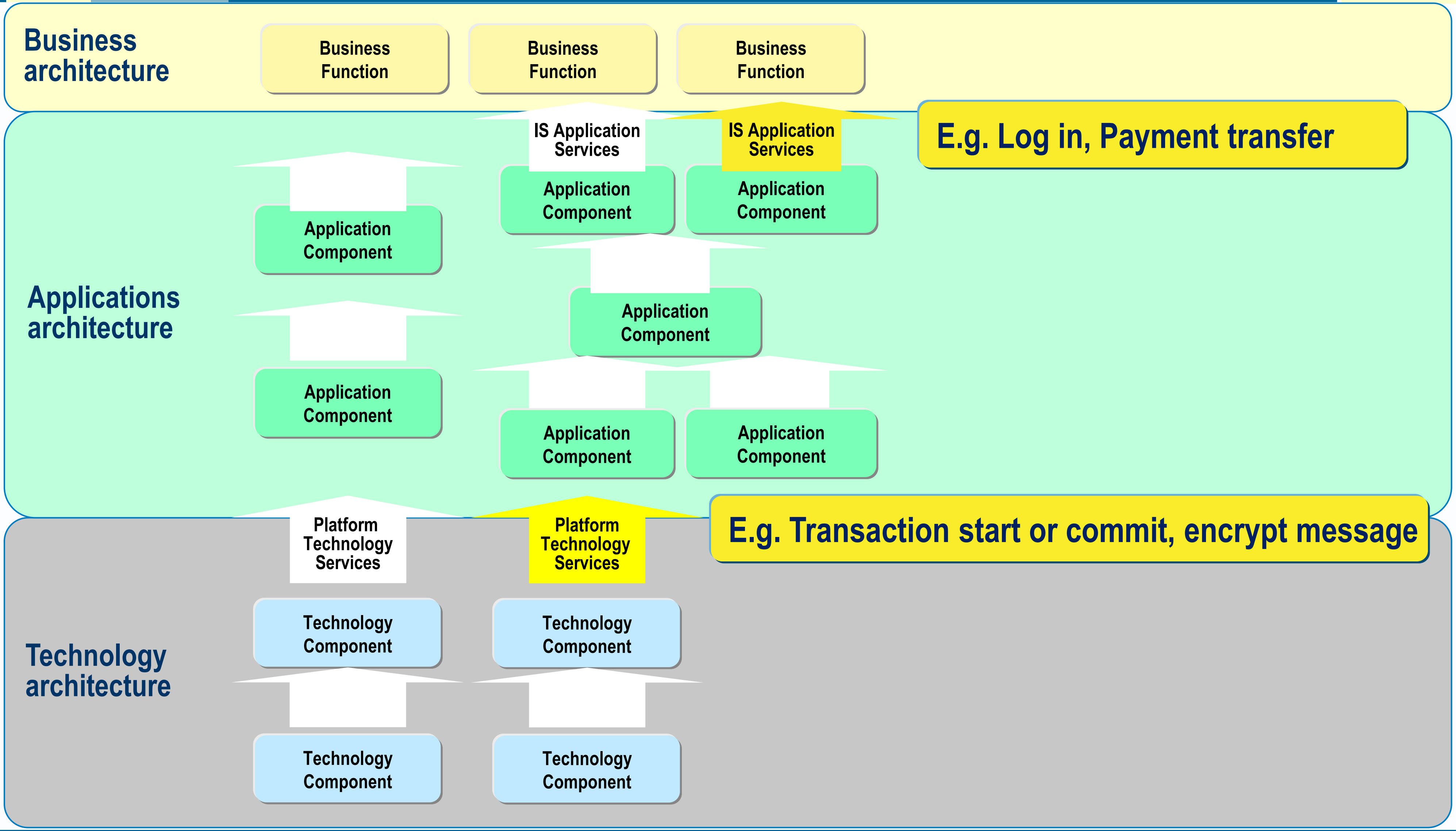
- Select & manage suppliers
- Plot migration path
- Review business case
- Plan delivery

Govern

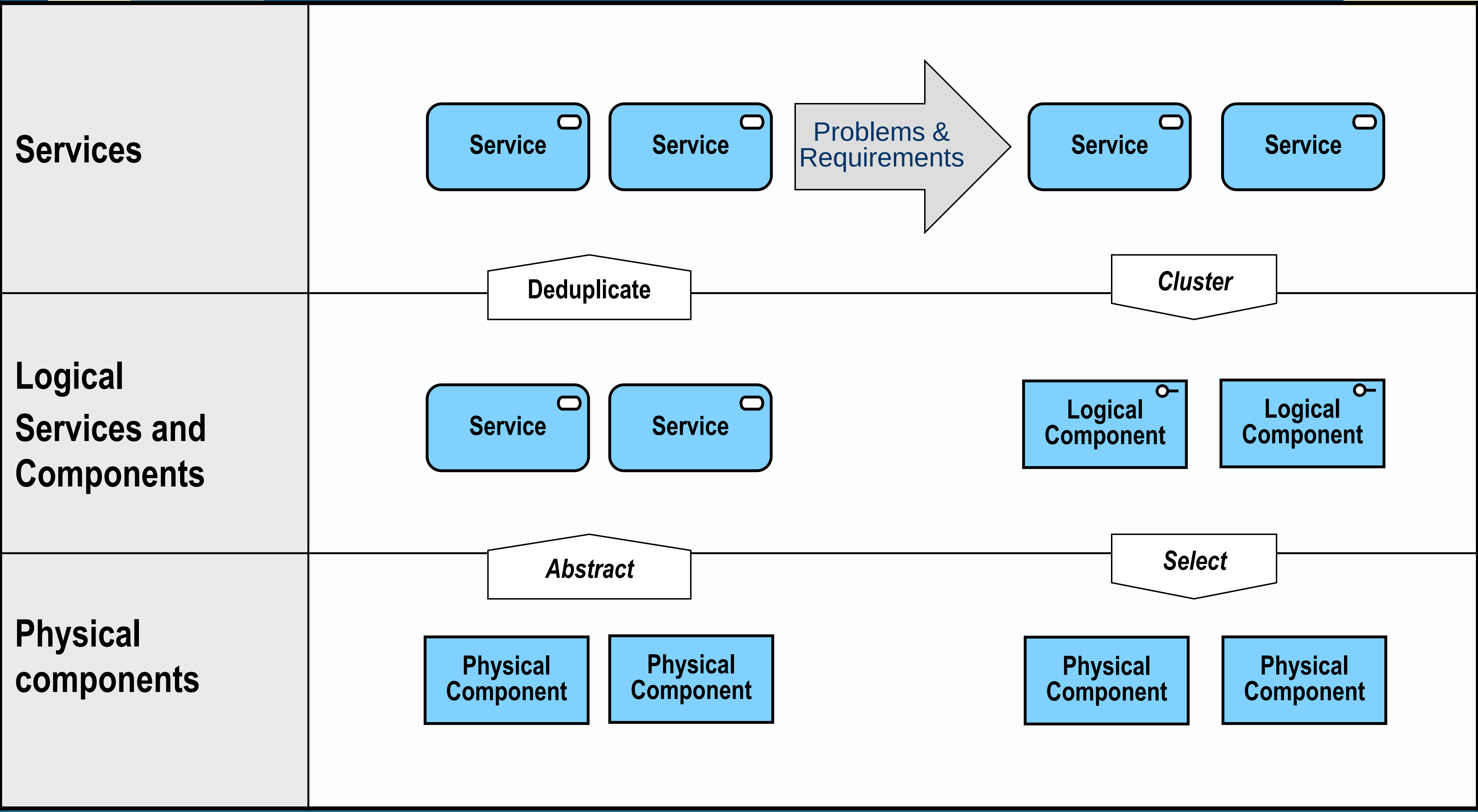
Govern delivery of what has been planned

- Hand over to delivery
- Govern delivery
- Monitor the portfolio(s)
- Respond to oper'l change

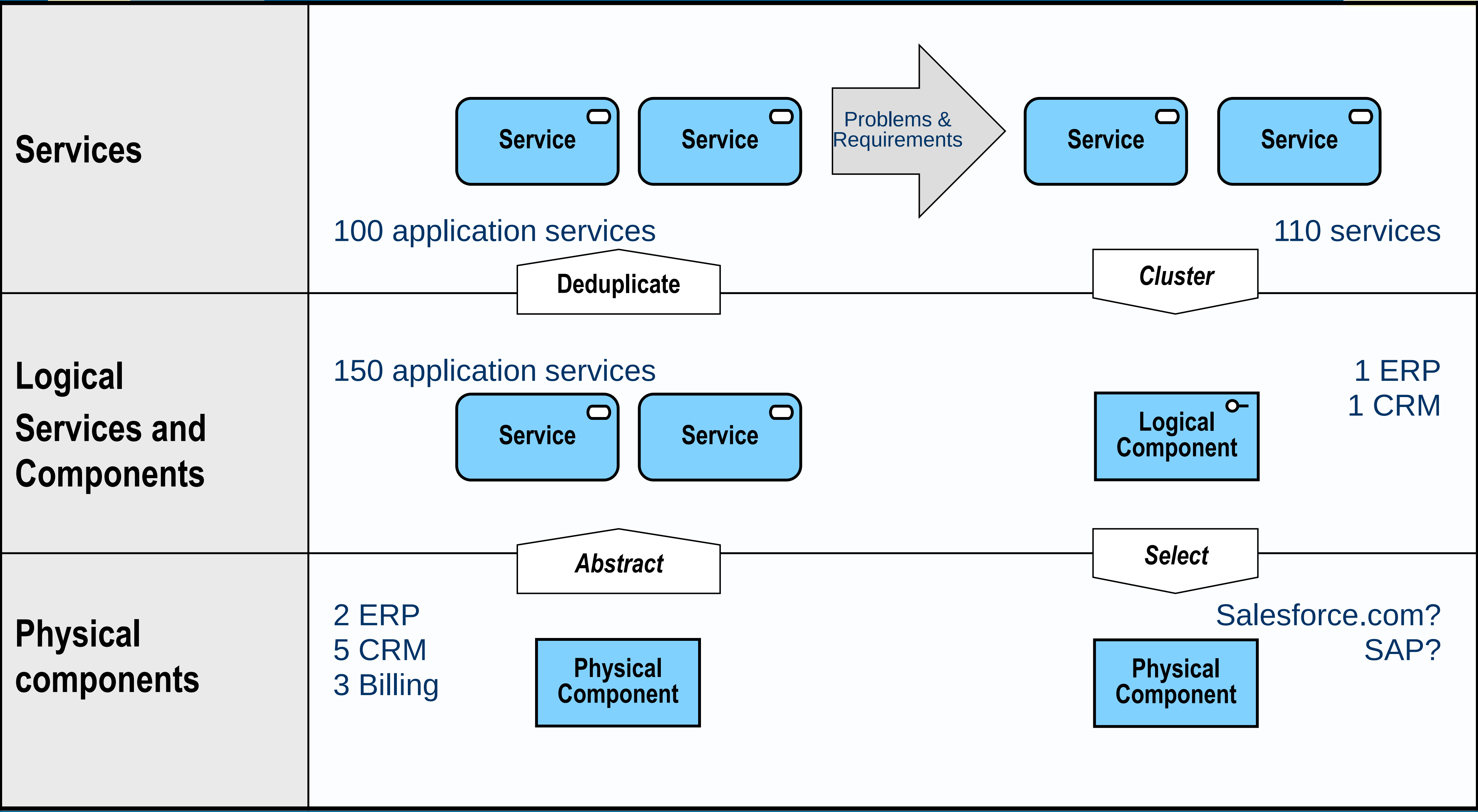
Layers architecture of components and services



Overview of the rationalisation approach



For example


Services
Logical
Services and
Components
Physical
components
Service
Service
Problems & Requirements
Service
Service
100 application services
110 services
Deduplicate
Cluster
150 application services
1 ERP
1 CRM
Service
Service
Logical Component
Abstract
Select
2 ERP
5 CRM
3 Billing
Salesforce.com?
SAP?
Physical Component
Physical Component

Portfolio rationalization

1. Identify the baseline components
2. Understand the baseline components
3. Evaluate baseline components
4. Review the context and motivations
5. Design the target component portfolio
6. Plan baseline-to-target migration
7. Govern delivery of the change

Common COTS applications

- Accounting
- Financial Reporting
- Data Warehousing, Business Intelligence and CPM
- Document Management, Business Process Management
- Content Management
- HR and Payroll
- Project Management

Applications within one ERP package!

- **Accounts Payable**
- **Accounts Receivable**
- Activity Management
- Benefits
- **BI Warehouse**
- **Billing**
- Bills of Material
- Capacity
- **Cash Management**
- **Claim Processing**
- Commission Calculation
- Commissions
- Cost Management
- Costing
- **Customer Contact & Call Center support**
- Engineering
- Fixed Assets
- **General Ledger**
- **Human Resources**
- Inspection of goods
- Inventory
- Manufacturing Flow
- Manufacturing Process
- Manufacturing Projects
- Order Entry
- **Payroll**
- Product Configurator
- **Purchasing**
- Quality Control
- **Rostering**
- Sales & Marketing
- Scheduling
- Service
- Supplier Scheduling
- Supply Chain Planning
- **Time & Attendance**
- Time & Expense
- **Training**
- Workflow Management

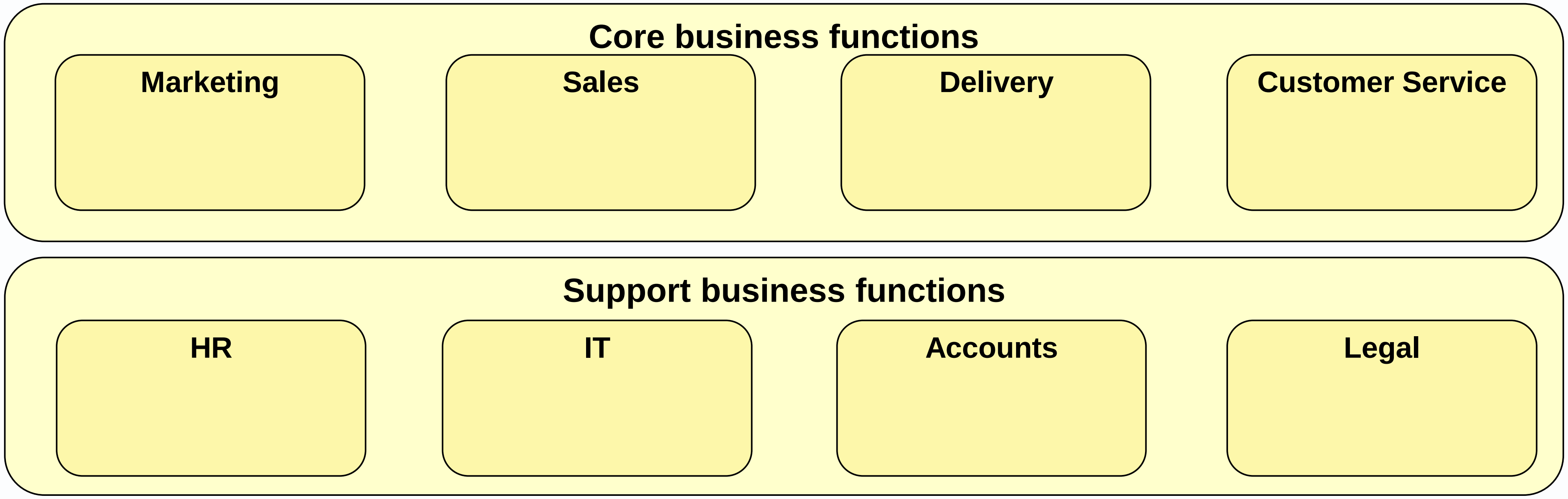
Start an Application Portfolio Catalog

- [an artefact] listing business applications and recording their properties.
- Usually structured under the business function hierarchy.

Application portfolio catalog
Application name
Pseudonym?
Cost to buy or build
Cost to run and maintain per month
Value to the enterprise
Licence/contract expiry dates
Status
Class
Roles (owners, users, maintainers)
Business functions/capabilities supported
Organisation units supported
Applications/components communicated with
Data stores accessed
Networks used
Hardware/software platform technologies
Etc.

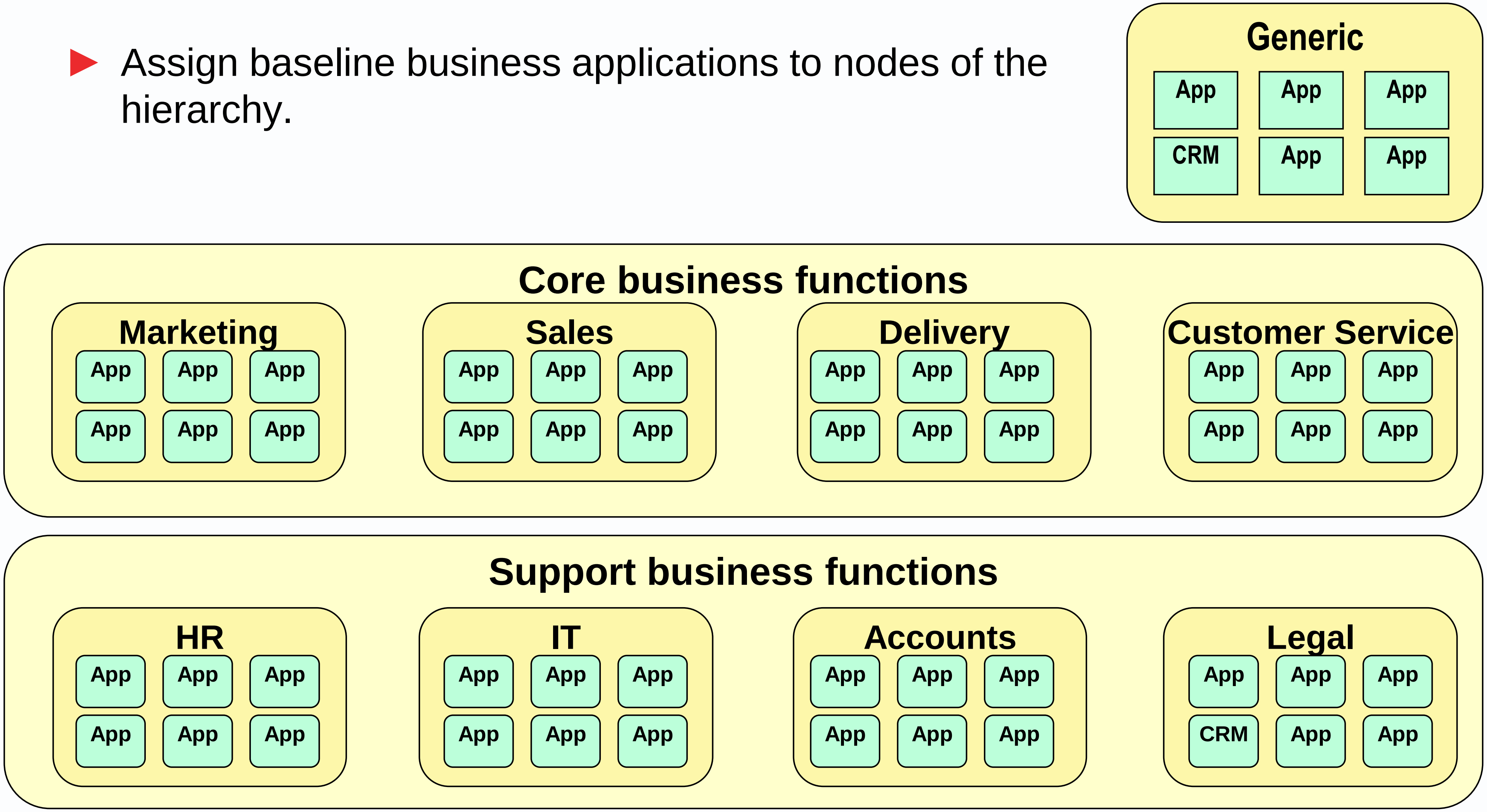
Map applications to function or capabilities

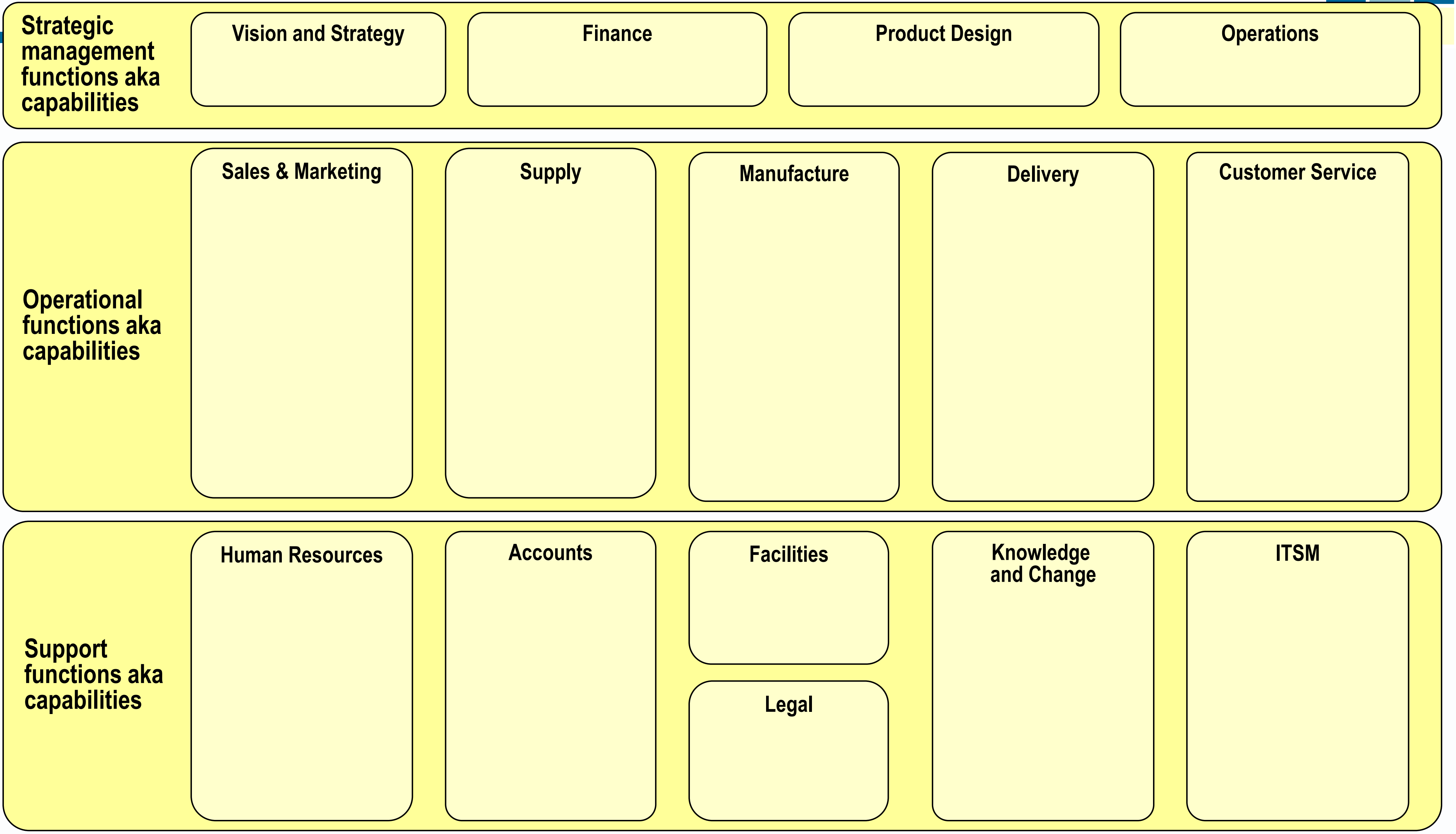
- Identify the part(s) of the organisation/function/capability hierarchy that are supported or enabled by the applications of interest.
- The 2nd or 3rd level of decomposition might be sufficient



Classify the baseline components

- Assign baseline business applications to nodes of the hierarchy.




Strategic management functions aka capabilities
Vision and Strategy
Finance
Product Design
Operations
Operational functions aka capabilities
Sales & Marketing
Supply
Manufacture
Delivery
Customer Service
Support functions aka capabilities
Human Resources
Accounts
Facilities
Legal
Knowledge and Change
ITSM

Strategic management functions aka capabilities

- Vision and Strategy
- Finance
- Product Design
  - Costing
  - Engineering
- Operations
  - Business Intelligence
  - Data Warehouse

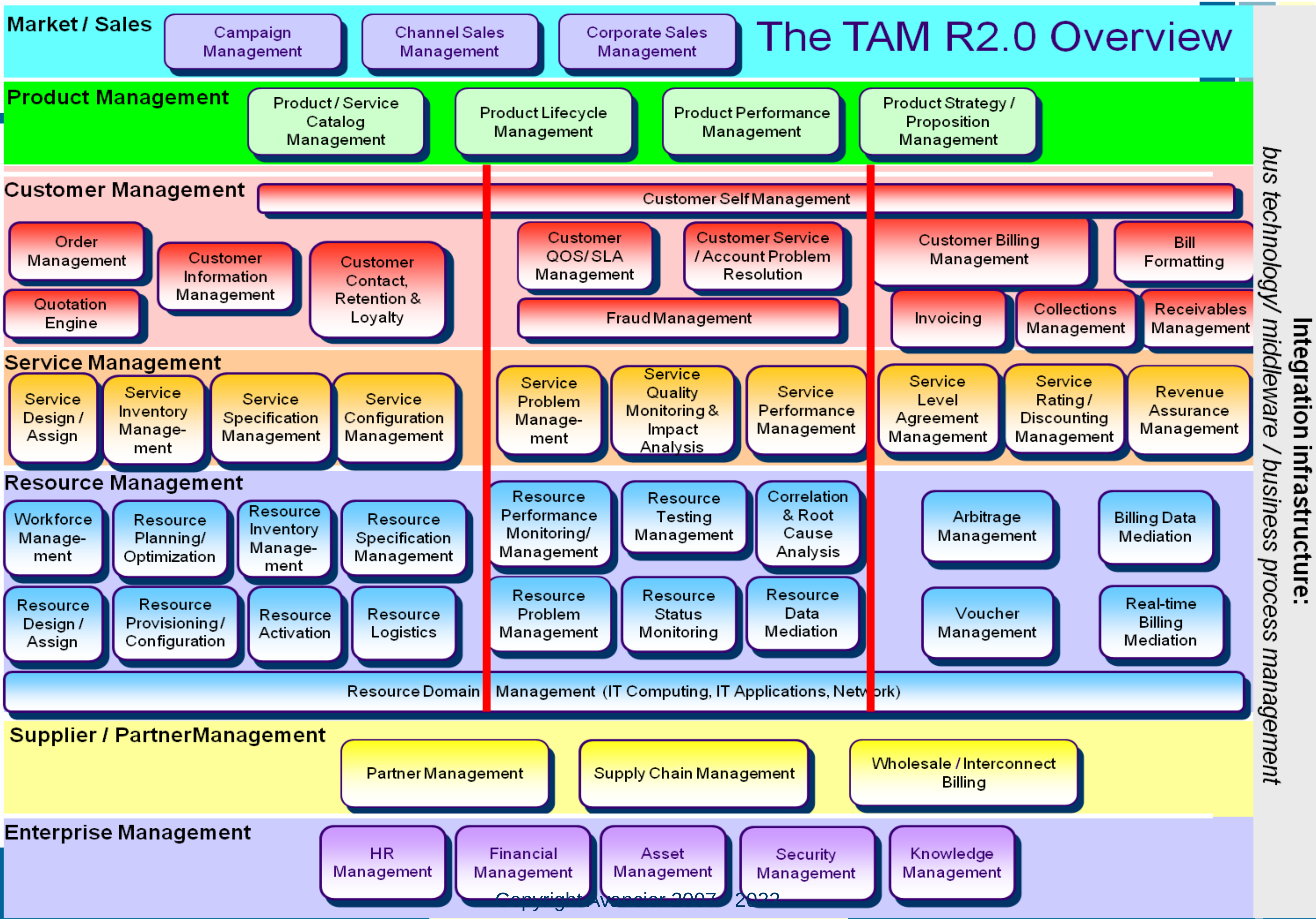
Operational functions aka capabilities

- Sales & Marketing
  - Sales and Marketing
  - Product Configurator
  - Order Entry
  - Pricing
  - Billing
  - Commissions
- Supply
  - Supply Chain Planning
  - Purchasing
  - Supplier Scheduling
  - Inspection of goods
  - Inventory
- Manufacture
  - Manufacturing Projects
  - Manufacturing Process
  - Manufacturing Flow
  - Bills of Material
  - Cost Management
  - Quality Control
- Delivery
  - Scheduling
  - Activity Management
  - Workflow Management
  - Time and Expenses
  - Capacity
- Customer Service
  - Customer Contact
  - Call Center support
  - Service

Support functions aka capabilities

- Human Resources
  - Human Resources
  - Benefits
  - Payroll
  - Rostering
  - Time and Attendance
- Accounts
  - Accounts Receivable
  - Accounts Payable
  - General Ledger
  - Fixed Assets
  - Cash Management
- Facilities
- Legal
- Knowledge and Change
  - Training
  - Project Management
  - Doc Management
- ITSM
  - Identity management
  - IT Service Management
  - Server Management
  - Network Management
  - EAI Middleware

The TAM R2.0 Overview



Market / Sales
Campaign Management
Channel Sales Management
Corporate Sales Management
Product Management
Product / Service Catalog Management
Product Lifecycle Management
Product Performance Management
Product Strategy / Proposition Management
Customer Management
Customer Self Management
Order Management
Customer Information Management
Customer Contact, Retention & Loyalty
Quotation Engine
Customer QOS/ SLA Management
Customer Service / Account Problem Resolution
Fraud Management
Customer Billing Management
Bill Formatting
Invoicing
Collections Management
Receivables Management
Service Management
Service Design / Assign
Service Inventory Manage-ment
Service Specification Management
Service Configuration Management
Service Problem Manage-ment
Service Quality Monitoring & Impact Analysis
Service Performance Management
Service Level Agreement Management
Service Rating / Discounting Management
Revenue Assurance Management
Resource Management
Workforce Manage-ment
Resource Planning/ Optimization
Resource Inventory Manage-ment
Resource Specification Management
Resource Performance Monitoring/ Management
Resource Testing Management
Correlation & Root Cause Analysis
Arbitrage Management
Billing Data Mediation
Resource Design / Assign
Resource Provisioning/ Configuration
Resource Activation
Resource Logistics
Resource Problem Management
Resource Status Monitoring
Resource Data Mediation
Voucher Management
Real-time Billing Mediation
Resource Domain Management (IT Computing, IT Applications, Network)
Supplier / PartnerManagement
Partner Management
Supply Chain Management
Wholesale / Interconnect Billing
Enterprise Management
HR Management
Financial Management
Asset Management
Security Management
Knowledge Management
Integration infrastructure:
bus technology/ middleware / business process management

What if there are too many applications?

- Focus on applications that are
  - mission critical and/or
  - used by many actors.

- Or group related applications into “system families”,
  - map those to functions/capabilities,
  - then document each system family separately.

Note

Many enterprises have some kind of application catalog.

Many don’t.

Portfolio rationalization

1. Identify the baseline components
2. Understand the baseline components
3. Evaluate baseline components
4. Review the context and motivations
5. Design the target component portfolio
6. Plan baseline-to-target migration
7. Govern delivery of the change

Understand the baseline components

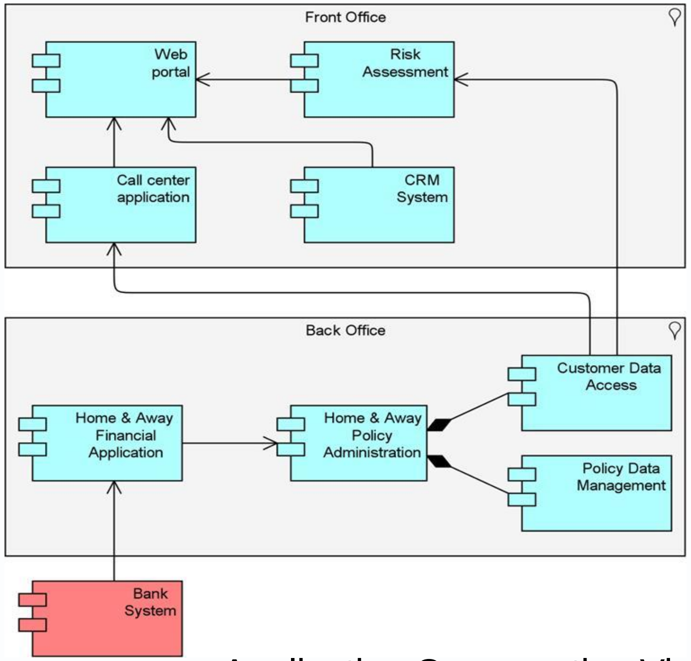
- Catalog the services provided by components.
- List the primary use cases of each application.

Application Communication Diagram

Avancier

- Which apps does an app serve and depend on?
- What service or flows do they exchange?

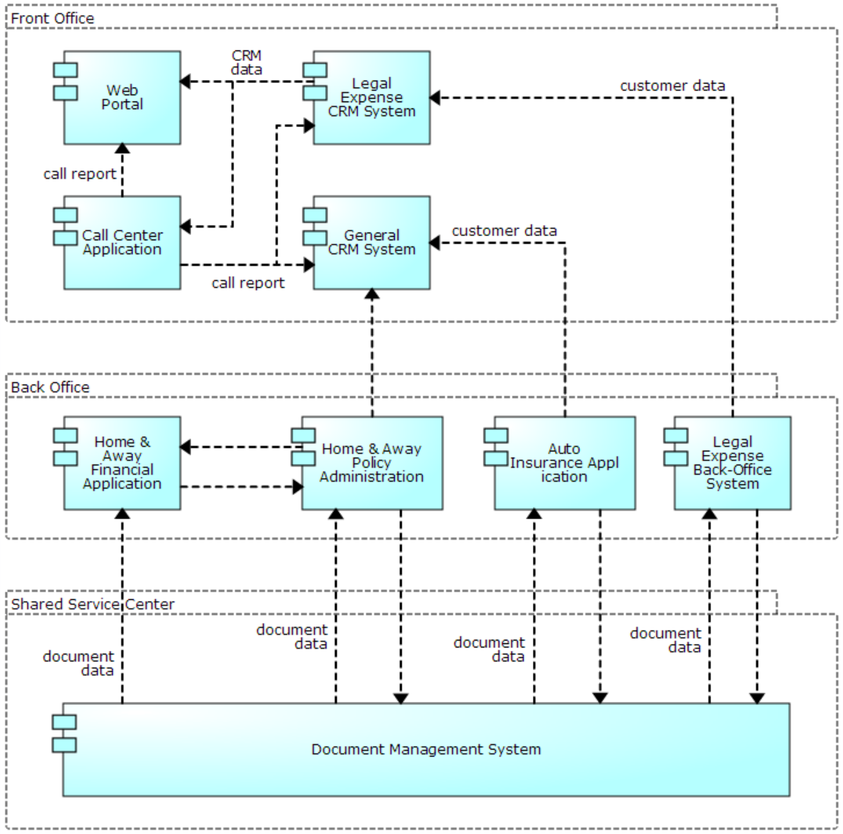
Shows which apps *serve* which apps



Application Co-operation Viewpoint

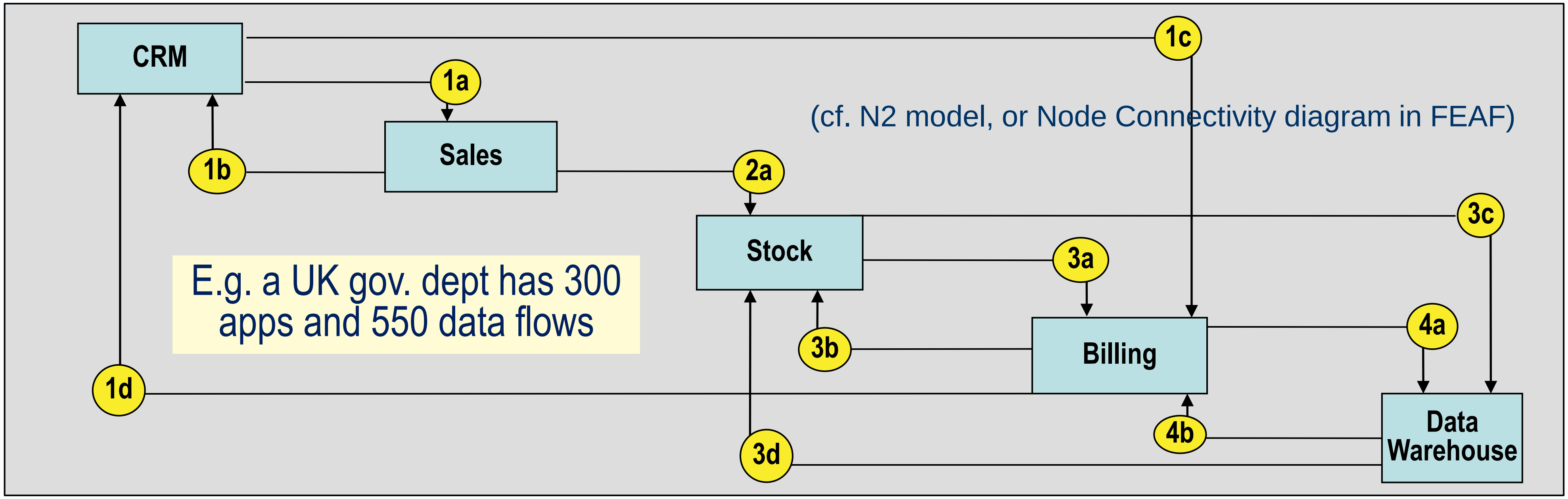
These diagrams use ArchiMate symbols

Shows which apps *send data to* which apps



Applications communication diagram + data flow catalogue

Avancier


CRM
Sales
Stock
Billing
Data Warehouse
1a
1b
1c
1d
2a
3a
3b
3c
3d
4a
4b
(cf. N2 model, or Node Connectivity diagram in FEAF)
E.g. a UK gov. dept has 300 apps and 550 data flows

Data Flow id	Source App	Destination App	Data content	Trigger event
1a	CRM	Sales	Sales order request	New sales order
1b	Sales	CRM	Sales order confirmation	Order created in the Sales system
2a	Sales	Stock	Requisition	Subscribe/Publish timer

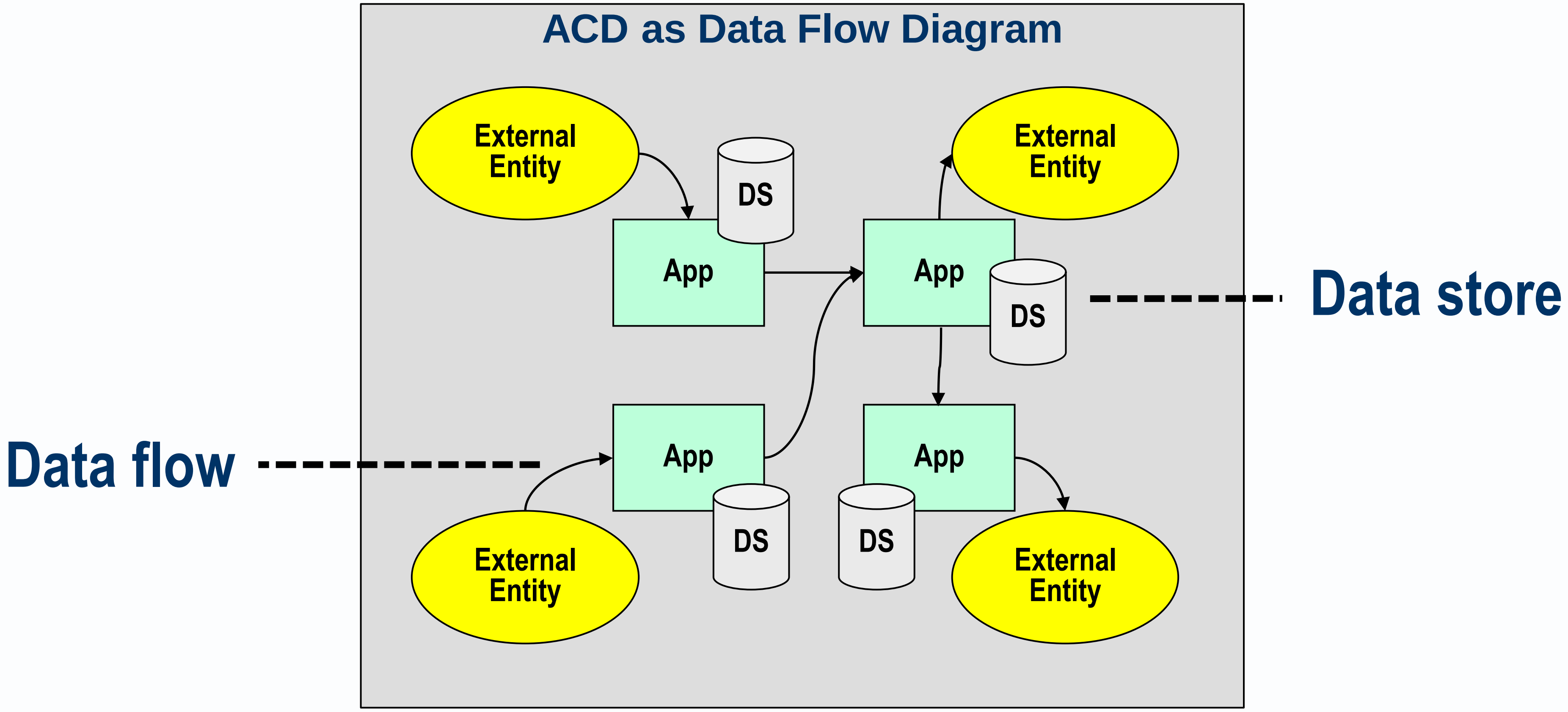
Data Flow (or Message) Catalog

- A flow can be detailed in terms of
  - Flow name, sender(s), receiver(s)
  - Trigger
  - Data structure transported (cross reference to)
  - Data format (Free text, CSV, JSON, XML etc.)
  - Transport mechanism or medium
  - Non-functional qualities such as throughput, CIA etc.

This shows what could be documented rather than what most actually do.
But understanding what is possible in theory is a precursor to deciding what to do in practice.

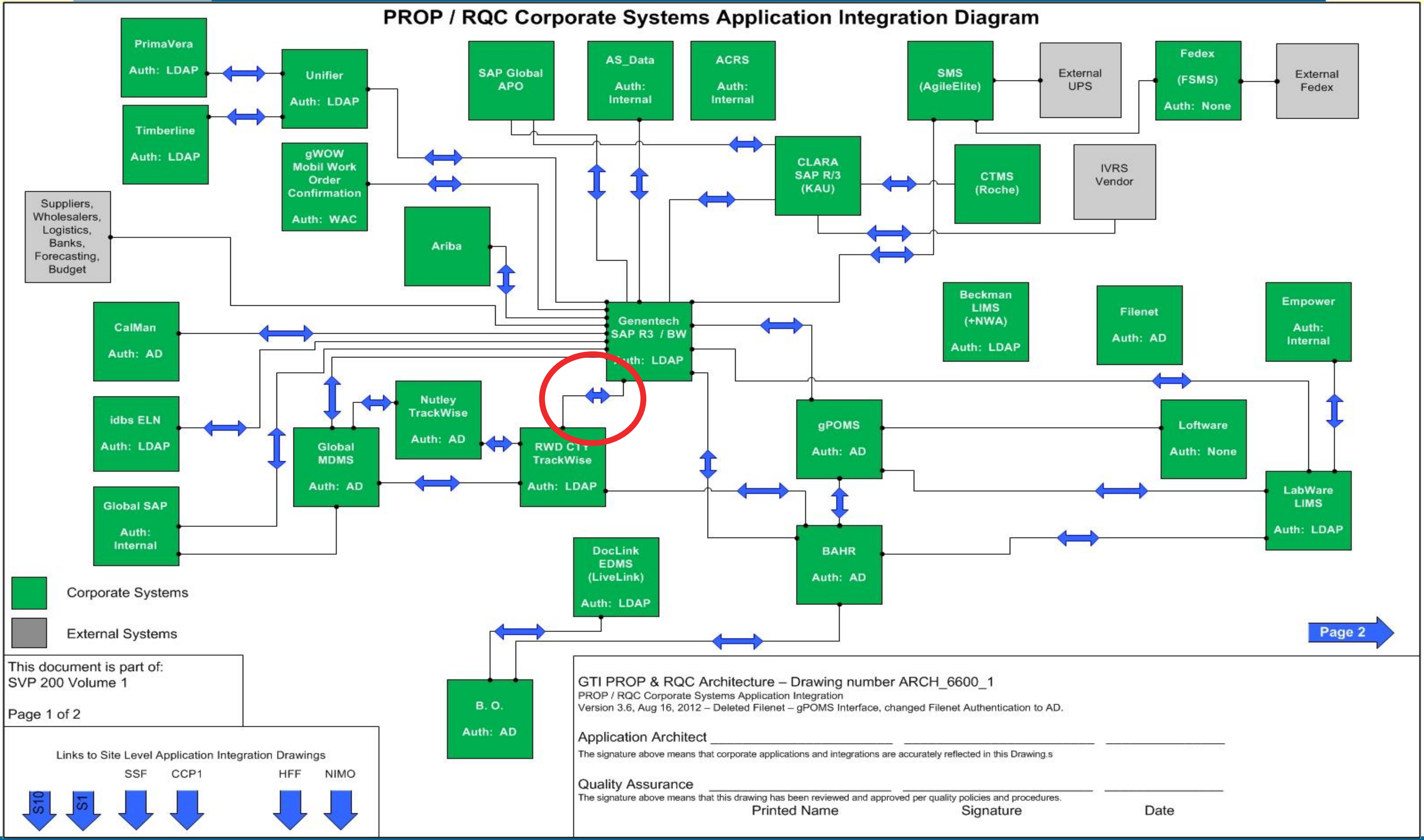
Data flow	Sender	Receiver	Trigger	Content	Transport	Throughput	C	I	A
Enquiry	Customer	Sales	Interest	Unstructured	Email	1000/day	High	Medium	9.00 to 18.00
Order	Customer	Sales	Web site visit	Structured	Web/HTTPS	60/day	High	High	24/7

Map applications to data flows and data stores



Application Communication diagram for one “system family”

Avancier



🟩 Corporate Systems

▨ External Systems

This document is part of:
SVP 200 Volume 1

Page 1 of 2

GTI PROP & RQC Architecture – Drawing number ARCH_6600_1
PROP / RQC Corporate Systems Application Integration
Version 3.6, Aug 16, 2012 – Deleted Filenet – gPOMS Interface, changed Filenet Authentication to AD.

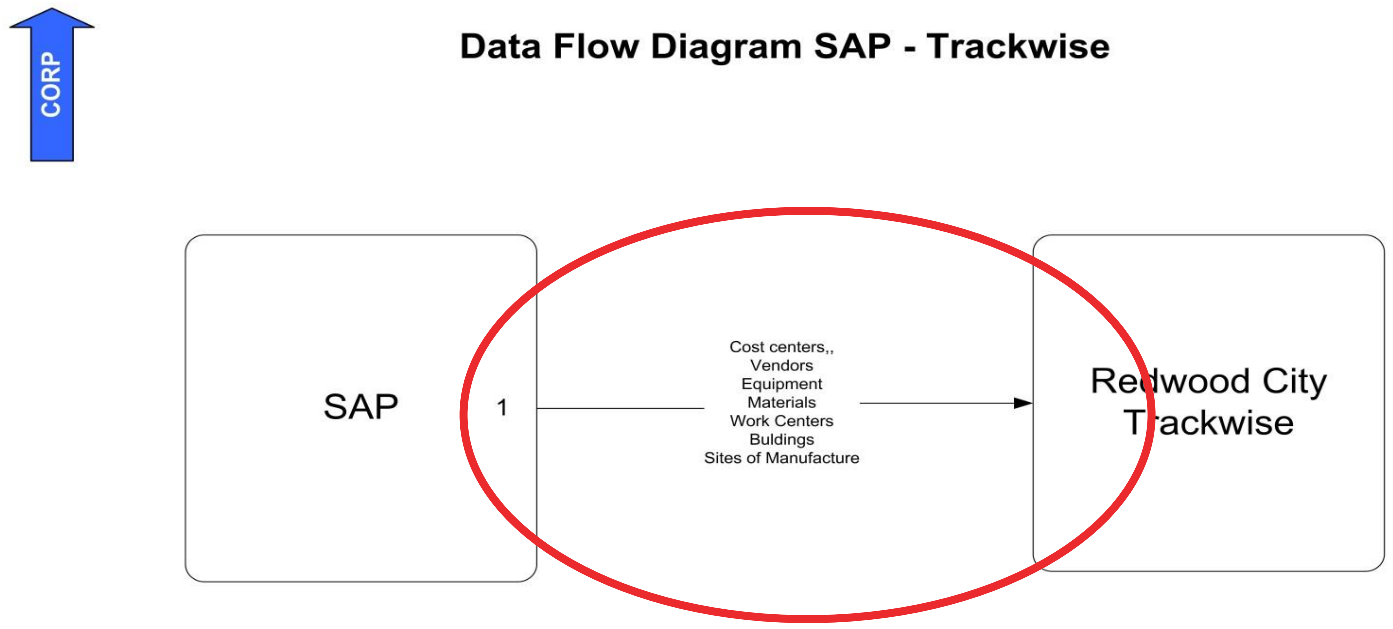
Application Architect ______________ ______________ ______________
The signature above means that corporate applications and integrations are accurately reflected in this Drawing.s

Quality Assurance ______________ ______________ ______________
The signature above means that this drawing has been reviewed and approved per quality policies and procedures.

Printed Name	Signature	Date

Avancier

Data Flow Diagram SAP - Trackwise



GTI PROP IT Architecture – Drawing number ARCH_6639
Data Flow Diagram SAP - Trackwise
Version 3.0 February 25, 2011 – Added border, changed object behavior, standardized dwg. Chanded Trackwise to "Redwood City Trackwise" to distinguish it from the Trackwise instance in Nutley.

SAP Tech Lead ______________ ______________ ______________
The signature above means that data flows are accurately reflected in this drawing.

Trackwise Tech Lead ______________ ______________ ______________
The signature above means that the data flows are accurately reflected in this drawing.

Quality Assurance ______________ ______________ ______________
The signature above means that this drawing has been reviewed and approved per quality policies and procedures.

System Owner ______________ ______________ ______________
The signature above means that the application interactions shown in this drawing are consistent with the validated functionality of this application.

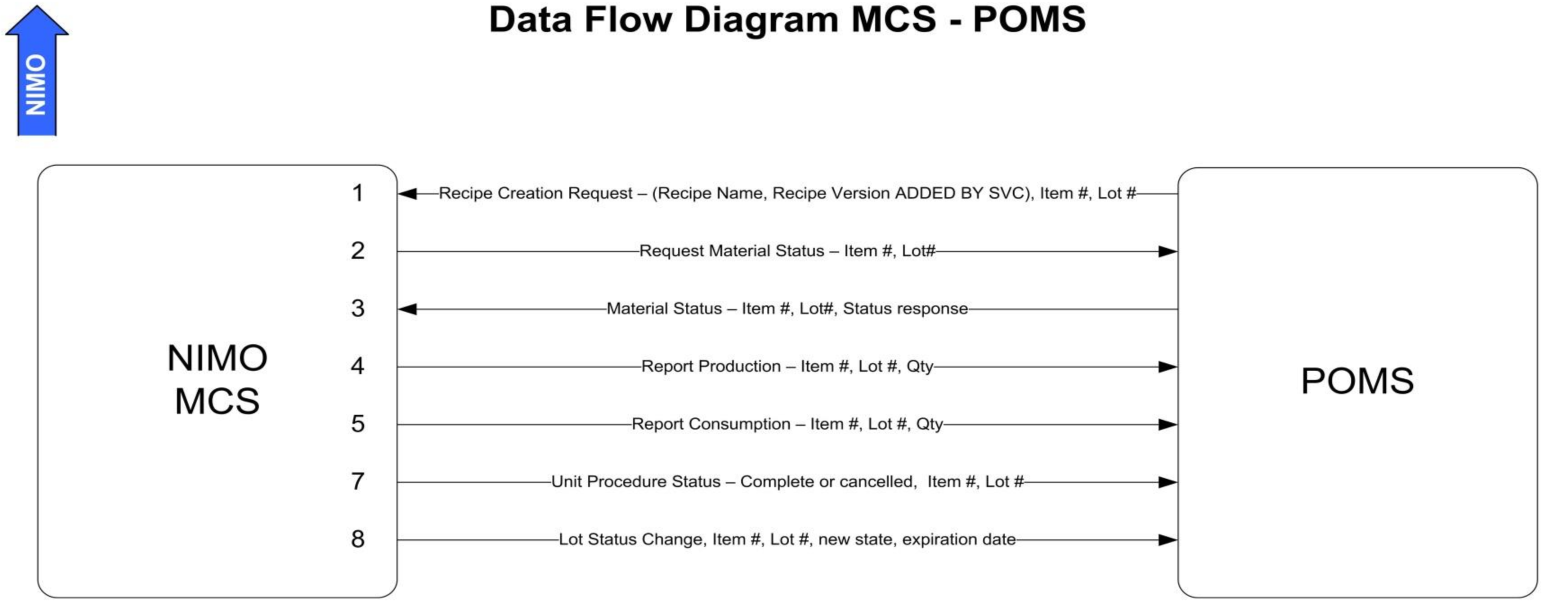
Printed Name Signature Date

Transaction Technology:
1 - Flat file exchange

This document is part of:
SVP 200 Volume 1

Page 1 of 1

Data Flow Diagram MCS - POMS


NIMO
NIMO MCS
POMS
1 Recipe Creation Request – (Recipe Name, Recipe Version ADDED BY SVC), Item #, Lot #
2 Request Material Status – Item #, Lot#
3 Material Status – Item #, Lot#, Status response
4 Report Production – Item #, Lot #, Qty
5 Report Consumption – Item #, Lot #, Qty
7 Unit Procedure Status – Complete or cancelled, Item #, Lot #
8 Lot Status Change, Item #, Lot #, new state, expiration date

GTI PROP Architecture – Drawing number ARCH_6710
DFD Campaign Studio – POMS
Version 3.1 February 28, 2011 – Added border, change object behavior, standardized dwg. Assigned drawing number.

Transaction Technology:
1 – 5 Web Methods Web Service.
6 – Report waste transaction removed from scope – Not yet implemented.
7 – 8 Web Methods Web Service.

	Printed Name	Signature	Date
MCS Tech Lead			
POMS Tech Lead			
Quality Assurance			
System Owner			

MCS Tech Lead: The signature above means that the integrations with MCS are accurately reflected in this drawing.

POMS Tech Lead: The signature above means that the integrations with POMS are accurately reflected in this drawing.

Quality Assurance: The signature above means that this drawing has been reviewed and approved per quality policies and procedures.

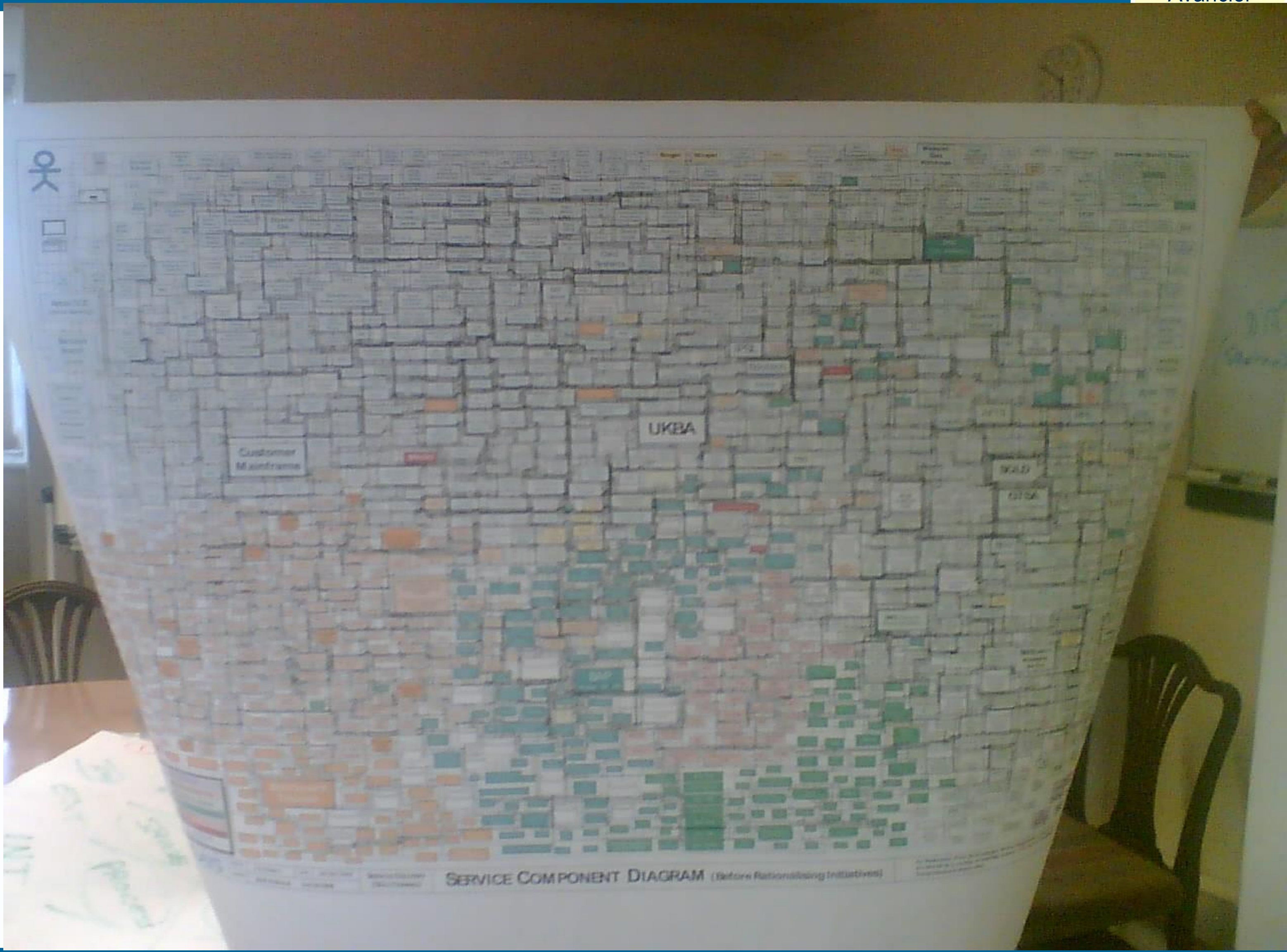
System Owner: The signature above means that the application interactions shown in this drawing are consistent with the validated functionality of this application.

This document is part of:
SVP 200 Volume 1

Page 1 of 1

Another real example

- 1 higher level diagram for system families only?
- N lower level diagrams for apps within a system family?



Map applications to other architectural entities

Avancier

- Role/Application Matrix
- Application/Organization Matrix

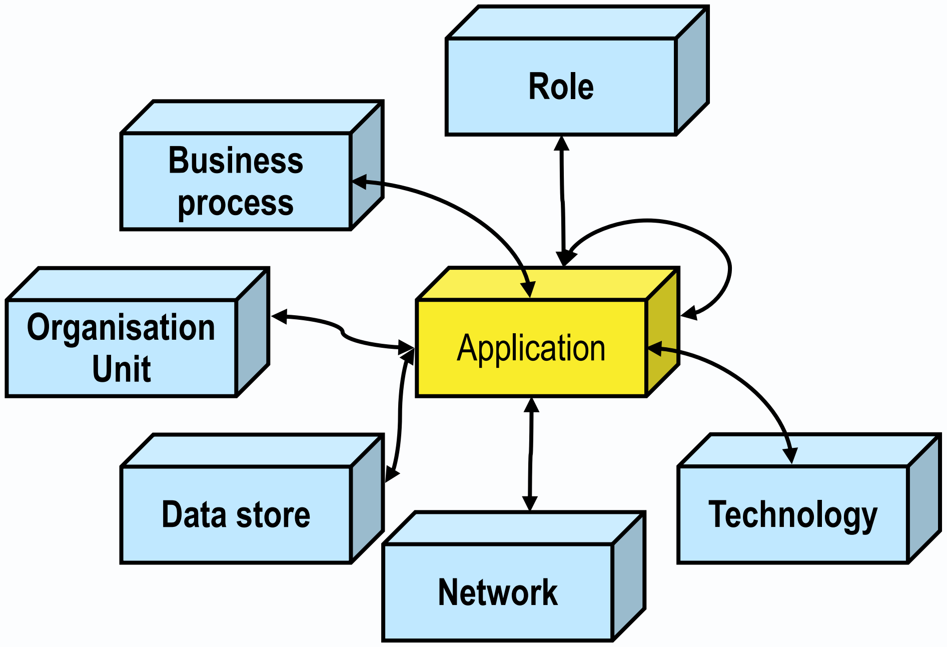
Application / Organisation Unit	CRM	Products	Billing	BI
Marketing	Uses			Uses
Sales	Uses		Uses	
Finance			Uses	
Management				Uses

- How widely or narrowly is a business supported by applications?
- How widely or narrowly is an application used?
- Where might better interoperability or support be needed?
- Where might security threats arise?

Define your meta model

Avancier

- E.g.

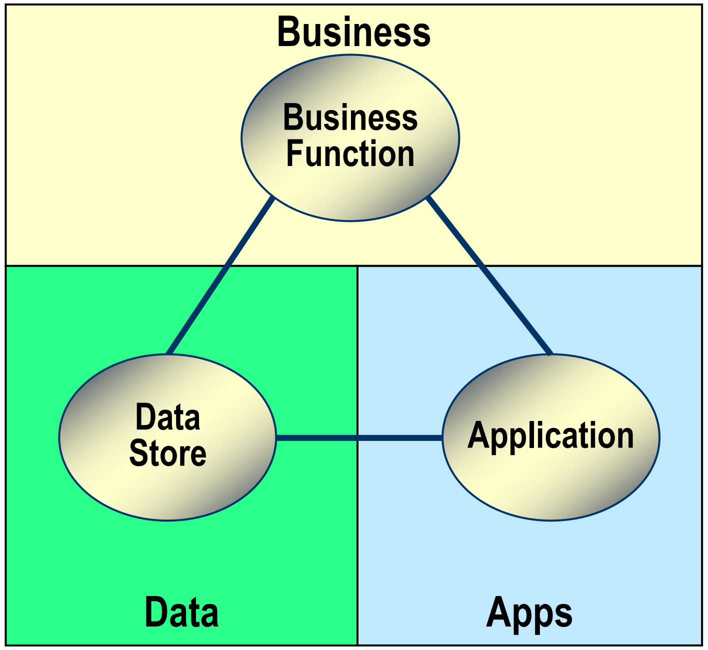


Application portfolio catalog
<u>Application name</u> Primary key
<u>Pseudonym?</u>
Cost to buy or build
Cost to run and maintain per month
Value to the enterprise
Licence/contract expiry dates
Status
Class Foreign keys
Roles (owners, users, maintainers)
Business functions/capabilities supported
Organisation units supported
Applications/components communicated with
Data stores accessed
Networks used
Hardware/software platform technologies
Etc.

Note

It seems few enterprises analyze to components to that level of detail.

In practice, few document more than this



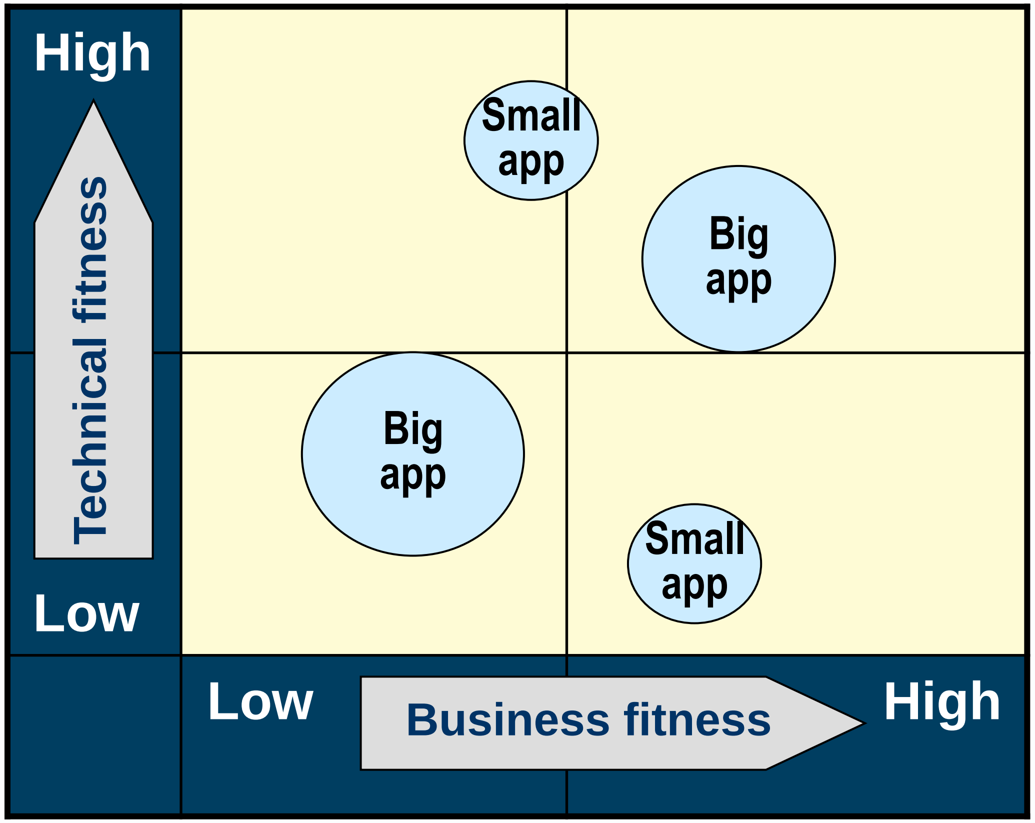
Portfolio rationalization

1. Identify the baseline components
2. Understand the baseline components
3. Evaluate baseline components
4. Review the context and motivations
5. Design the target component portfolio
6. Plan baseline-to-target migration
7. Govern delivery of the change

Evaluate baseline components

- *Business fitness*, considering
  - Usage
  - Benefit
  - Cost per user, per transaction

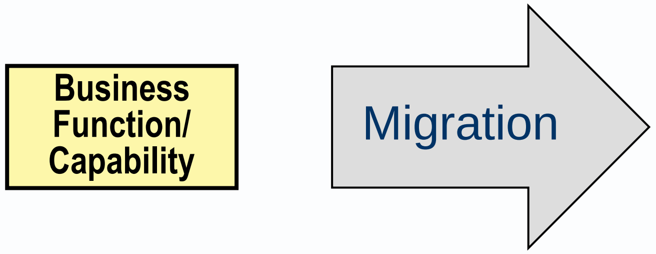
- *Technological fitness*. considering
  - Supportability,
  - Technical debt
  - Compliance to standards
  - Incident/problem frequency



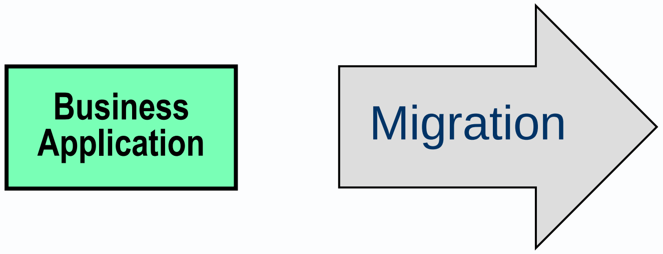
Portfolio rationalization

1. Identify the baseline components
2. Understand the baseline components
3. Evaluate baseline components
4. Review the context and motivations
5. Design the target component portfolio
6. Plan baseline-to-target migration
7. Govern delivery of the change

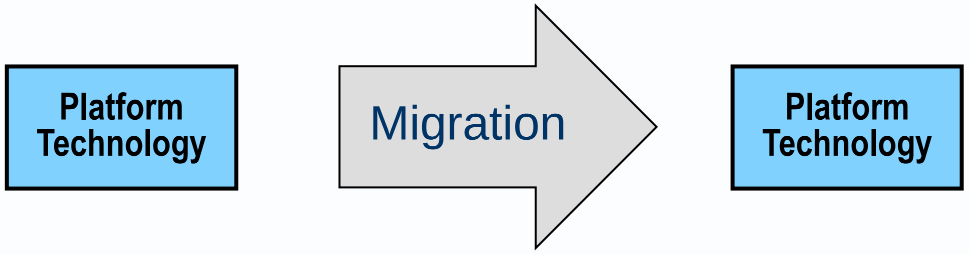
Review the context and motivations



Review any higher-level business change road map and other drivers for application change.



Review any lower-level technology change road map



Note

- Vendors (especially cloud service providers) may dictate update cycles.

Portfolio rationalization

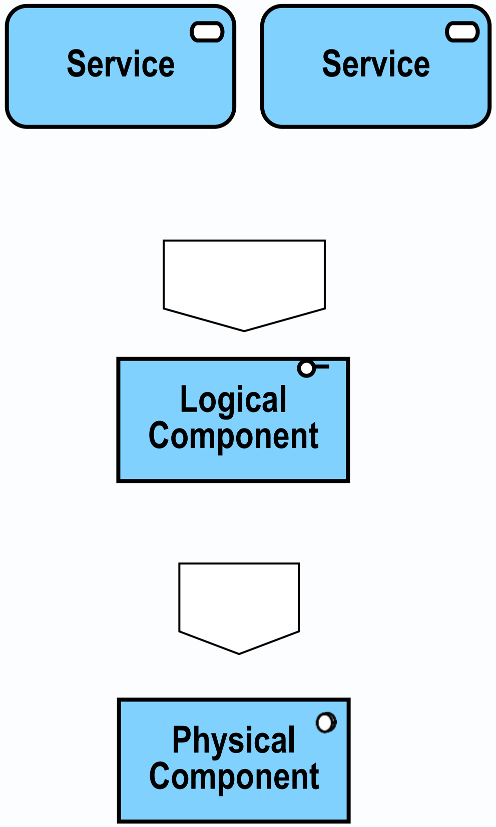
1. Identify the baseline components
2. Understand the baseline components
3. Evaluate baseline components
4. Review the context and motivations
5. Design the target component portfolio
6. Plan baseline-to-target migration
7. Govern delivery of the change

Design the target component portfolio - rationalize

- List and deduplicate services provides
- Refine in the light of the context and motivations.

- Define target application components by clustering cohesive groups of required services,
- Mindful of what is available in the market place by way of generic application components.

- Identify procurable components
- Select and procure components



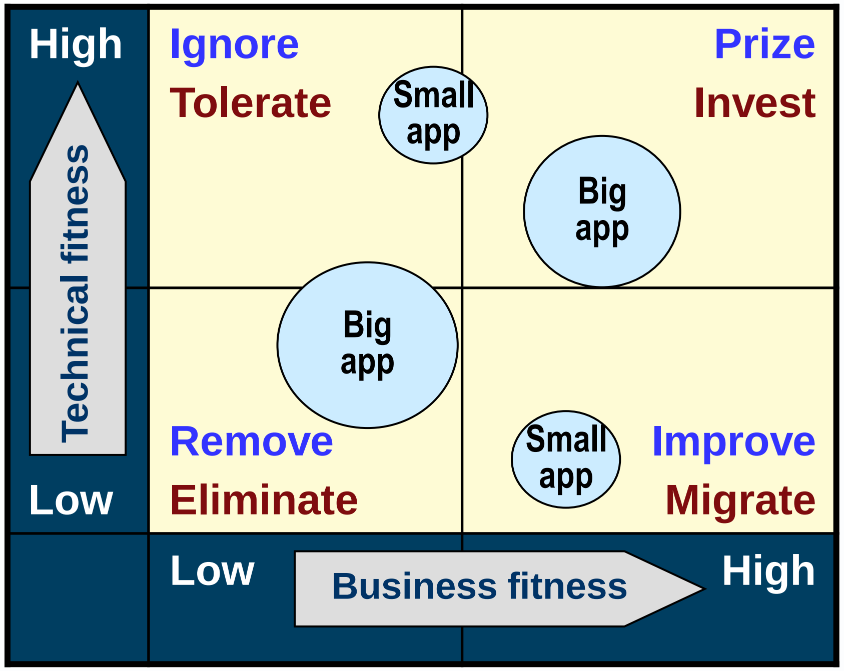
Design the target component portfolio – classify changes

- Define the vision for each component, that is, the end state to be reached after (say) 3 years.

- Classify actions to be taken

RIIP	TIME	MURDeR
Ignore	Tolerate	Monitor (frozen)
Prize	Invest	Update (maintain)
Improve	Invest	Rewrite
Improve	Migrate	Replace
Remove	Eliminate	Delete

- Map actions to applications



Design the target component portfolio - integrate

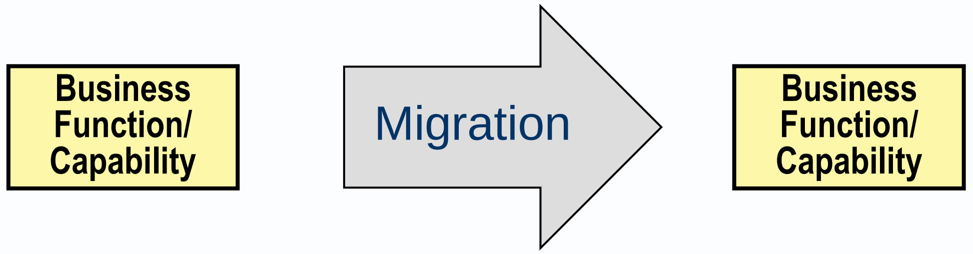
1. Look for application issues
2. Form an enterprise app road map
3. Identify data flows, data stores and applications in scope
4. Select best-fitting Application Integration Pattern
   - Swivel chair integration
   - Lipstick on a pig
   - Nosey neighbour
   - Distributed transaction
   - Run around
   - Data warehouse
   - Database/app consolidation
   - Physical master data
   - Virtual master data
5. Draw application communication diagram (aka DFD)
6. Draw sequence diagrams for key processes
7. Define integration technologies

TBD Later

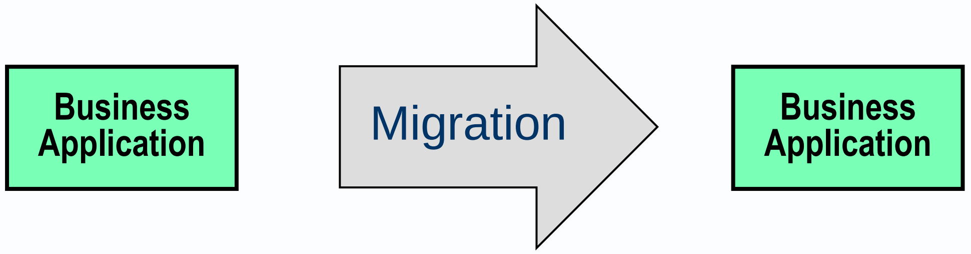
Portfolio rationalization

1. Identify the baseline components
2. Understand the baseline components
3. Evaluate baseline components
4. Review the context and motivations
5. Design the target component portfolio
6. Plan baseline-to-target migration
7. Govern delivery of the change

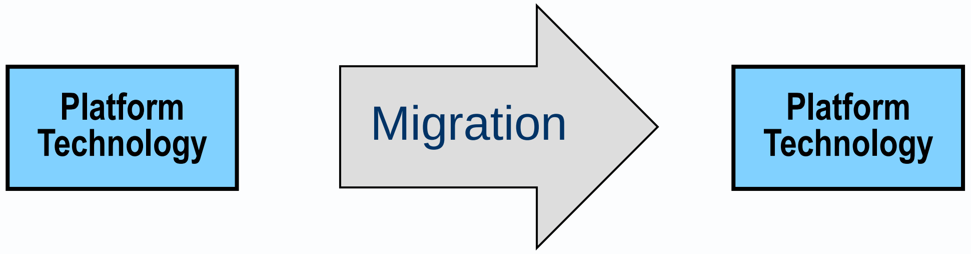
Plan baseline-to-target migration path



Align application changes with business changes.



Align application changes to technology changes.



A road map

Avancier

- A list of things and when we expect to change or replace them

Thing	Year	Year + 1	Year + 2	Year + 3

Define road map

Define a road map for changing components to reach the target

App	Year	Year + 1	Year + 2	Year + 3
ERP 1	Ignore	Ignore	Remove	
ERP 2			Deploy	Improve
CRM 1	Remove			
CRM 2	Deploy	Improve	Prize	Prize
Billing	Prize	Prize	Prize	Prize
DW/BI	Improve	Improve	Improve	Improve
Timesheet	Ignore	Rewrite	Prize	Prize

Portfolio rationalization

1. **Identify the baseline components**
2. **Understand the baseline components**
3. **Evaluate baseline components**
4. **Review the context and motivations**
5. **Design the target component portfolio**
6. **Plan baseline-to-target migration**
7. **Govern delivery of the change**

Govern delivery of the change

Finally (the most difficult step), govern delivery of the changes set out in business, application and technology change road maps.

However you do it

- This is a convoluted process that involves juggling:
  - The requirements of old and new business applications
  - Baseline applications that cannot be changed
  - Overarching principles and strategies
  - Time, cost and resource constraints on change
- Also
  - Generic applications available in the market place
- Where real applications provide services in a different way from your logical requirements, then things get messy