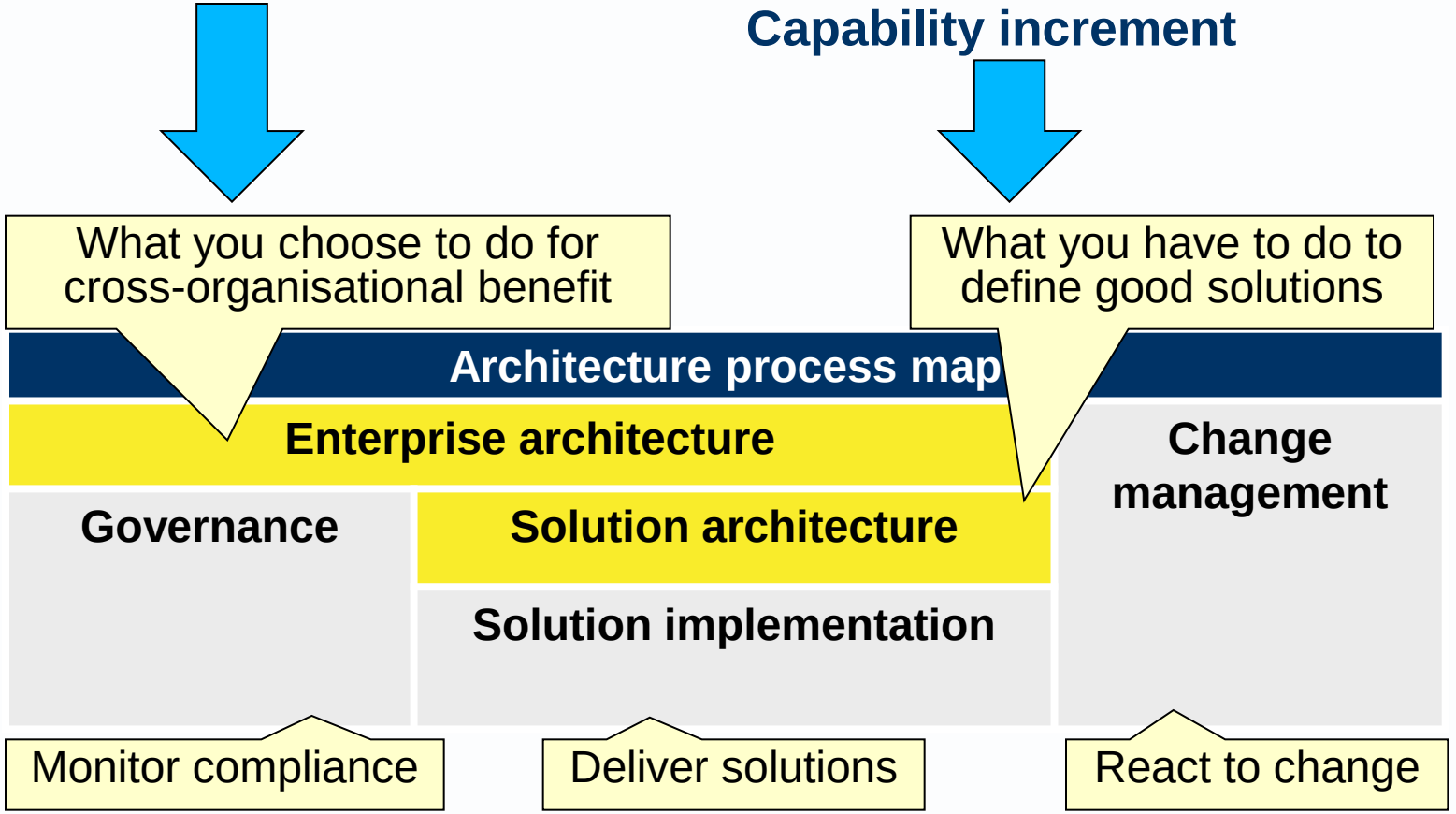
Avancier Reference Model

Architecture and Architects (ESA 1)

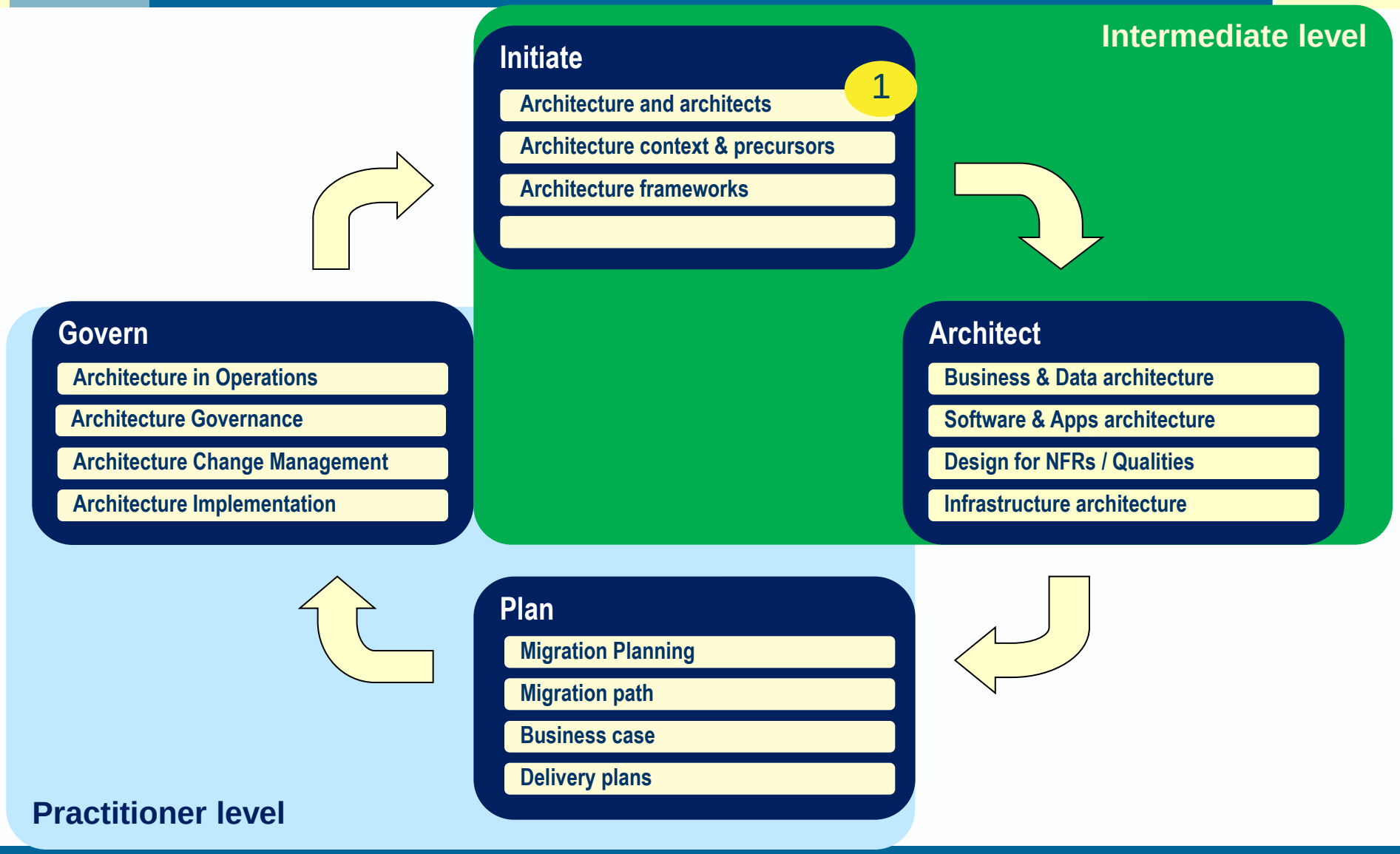
Positioning enterprise and solution architecture

CBP: Capability-based planning



Mapping this session to an architecting process

Avancier



Architecture and architects

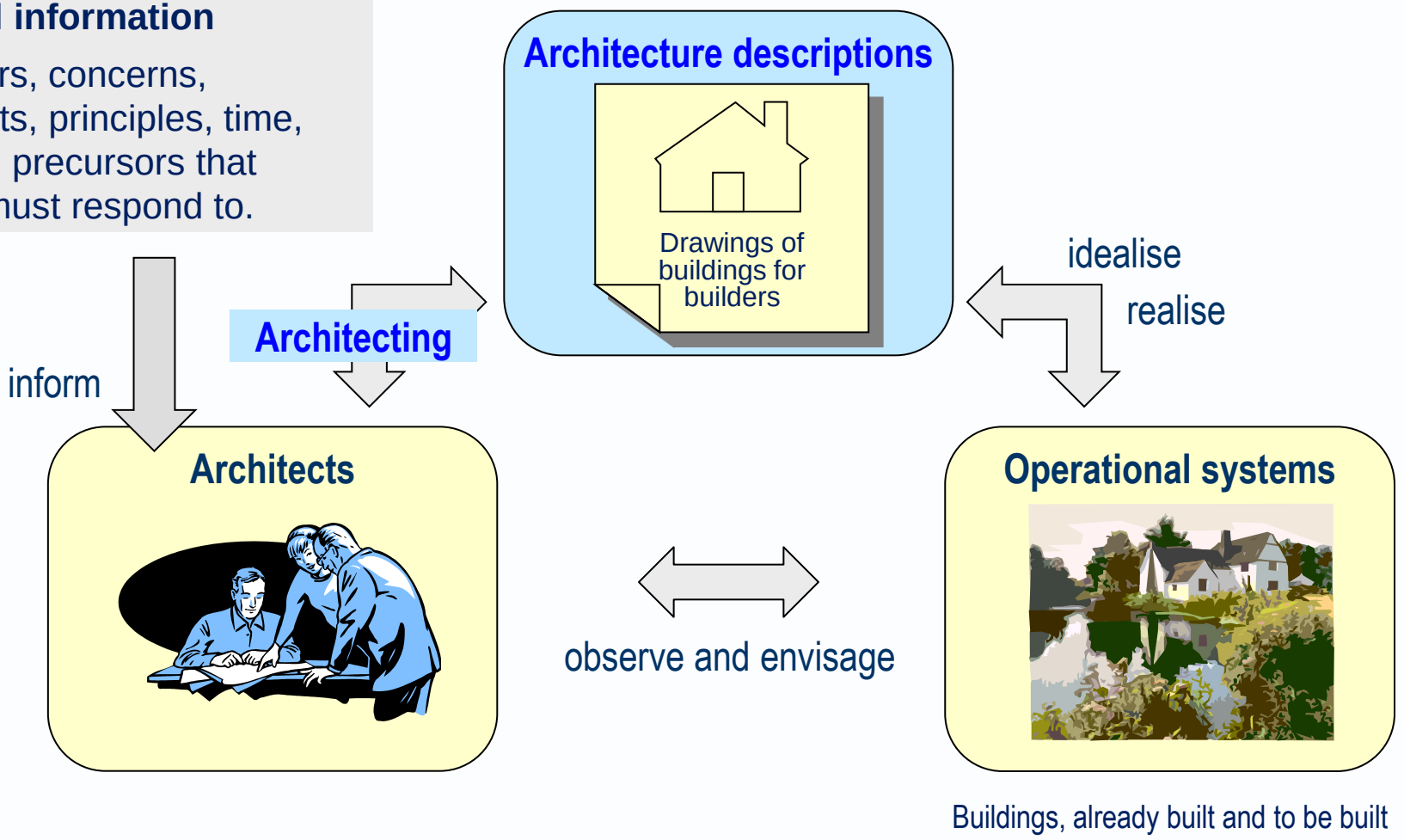
Avancier

Architecture and architects
Basic concepts
Architecture layers as views
Architecture domains as views
Architecture responsibility level
Architect roles, goals and skills

Architects do architecting and produce descriptions

Contextual information

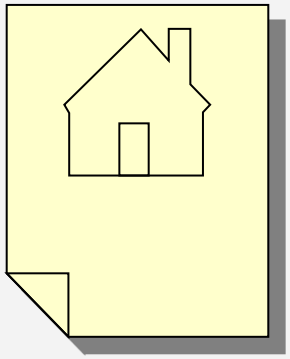
Stakeholders, concerns, requirements, principles, time, cost etc. All precursors that architects must respond to.



Architects produce descriptions

Avancier

Architecture description



[a work product] that describes a system, presented in views to address the concerns of stakeholders about that system.

It may map elements of the system to elements of the wider environment or context.

It may inform planning of the work to implement the system.

One architectural system description can define one or more system instances, or operational systems.

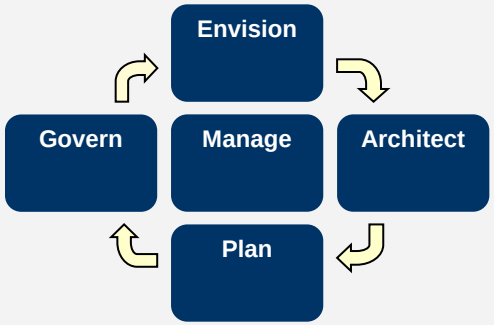
Reference model term

Reference model definition

Architects do architecting

Avancier

Architecting



[a work process] that creates an architecture description and a plan for constructing a system.

It involves analysis of the context, engaging with stakeholders, making decisions, analysing and choosing between options, defining system elements, recording them in an architecture description and ensuring agreement of what is described.

It creates a target architecture description to meet some requirements, under some constraints.

It often involves planning the move from the baseline state to the target state, and governing that change to ensure conformance of implemented systems to architecture descriptions.

Architecture and architects

Avancier

Architecture and architects
Systems and Architecture
Architecture layers as views
Architecture domains as views
Architecture responsibility level
Architect roles, goals and skills

Encapsulation of components (aka building blocks)

Avancier

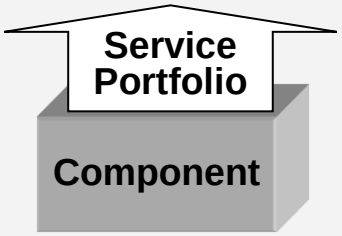
Encapsulation

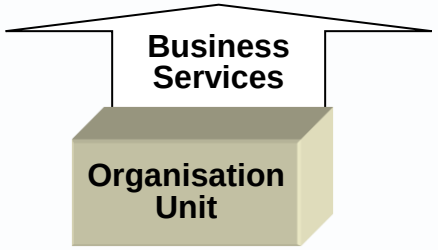
[a technique] for defining a system component or building block by its input/output interface, by the discrete events it can process and services it can offer.

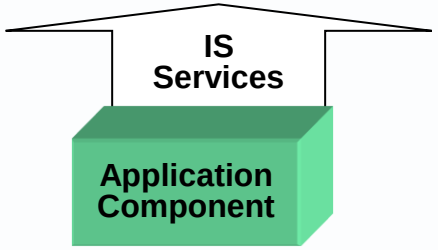
It hides inner workings or processes from outsiders.

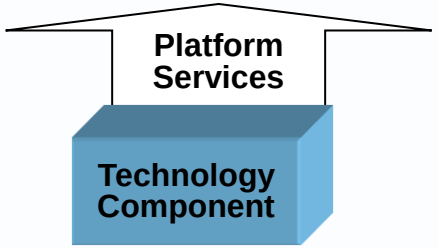
It hides internal resources (notably data) from outsiders, so the only way to access those resources is through the interface of the component.

In the table below, the architecture layers are considered to be encapsulated.



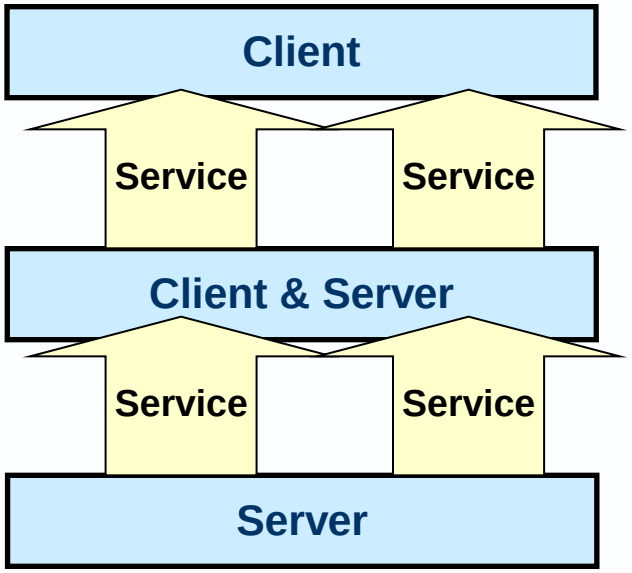






What do we mean by client and server?

Client component	[a component] that requests one or more services from a server component. In software, the request is usually called an invocation
Server component	[a component] that can provide one or more services in response to requests from a client. Cluster analysis can be used to group services that are related (e.g. by access to the same data) into one server component.



1.2 Architecture layers as views

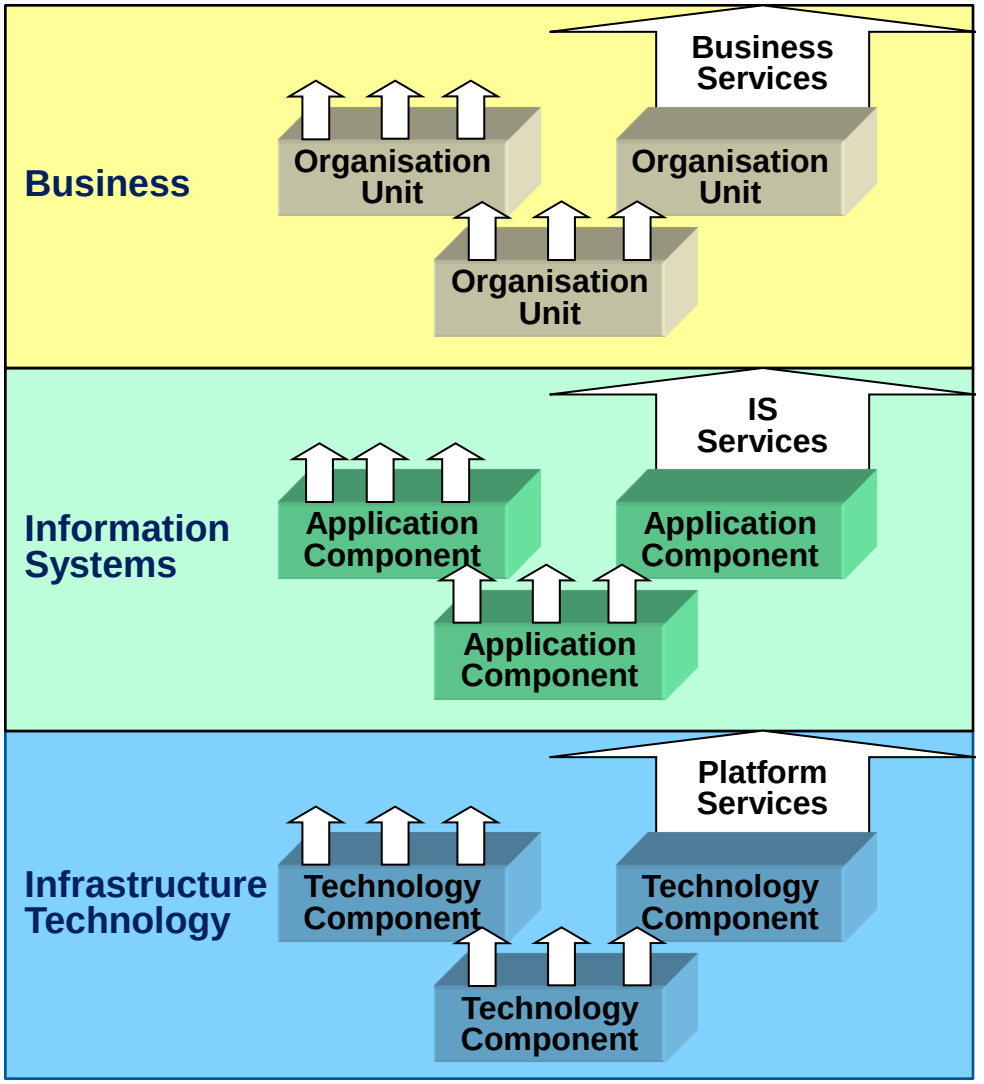
Avancier

Layered architecture

[a structure] in which components in one layer provide services to components in the layer above.

For example, layers in a network stack, or the business, application and infrastructure domains.

This is a useful mental model for enterprise and solution architects.



Architecture and architects

Avancier

Architecture and architects
Basic concepts
Architecture layers as views
Architecture domains as views
Architecture responsibility level
Architect roles, goals and skills

.

The four primary architecture domains

Avancier

- Established in the PRISM report (1986) and in
- "EA Planning" (Stephen Spewak, 1993)
- Now the basis countless of EA frameworks, including TOGAF

	Passive Structure	*Required Behaviour*	*Logical Structure*	*Physical Structure*
Business		Business Service Business Process	Function Role	Org Unit Actor
Data / Information	Data Entity	Data Flow	Log Data Model	Data Store
Applications		IS Service	Application Interface	Application
Infrastructure Technology		Platform Service	Platform Interface	Platform Application

Read the 4 reference model definitions

Architecture domains as views

Avancier

Business architecture

[a view] that identifies and relates business elements:

- desired business outcomes (visions, goals, objectives)
- business products and services
- business processes (scenarios, value streams) needed to maintain business resources and deliver services
- business resources, roles and actors needed to perform processes.

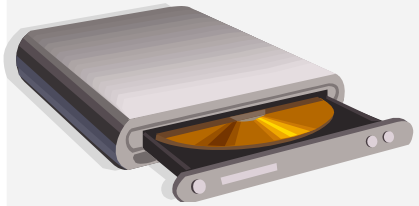
These elements may be mapped to goals and locations, to business data and applications.

EA is concerned with standardisation and integration of business roles and processes.

Architecture domains as views

Avancier

Data architecture



[a view] that identifies and relates data elements:

- identifies data stores and flows created and used by business activities
- describes data stores and the data structures they contain
- describes data flows and the data structures they contain
- describes data qualities: data types, confidentiality, integrity and availability

These elements may be mapped to business activities and to applications.

EA is concerned with standardisation and integration of data between systems.

Architecture domains as views

Avancier

Applications architecture

[a view] that identifies and relates application elements:

- identifies business applications and their uses to support business activities.
- describes coarse-grained applications rather than fine-grained software components.
- identifies application use cases
- identifies input/output data flows and inter-application relationships

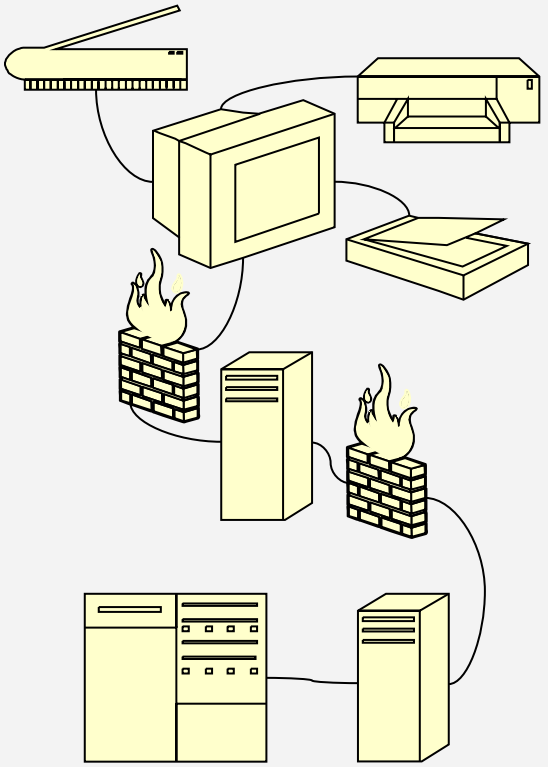
These elements may be mapped to business activities and to platform applications.

EA is concerned with standardisation, integration and life cycles of business applications.

Architecture domains as views

Avancier

Infrastructure architecture



[a view] that identifies and relates platform elements

- identifies platform applications and the services they offer to business applications
- is concerned with the deployment of business applications to platform applications
- defines the client and server nodes that platform applications are deployed on
- defines the protocols and networks by which nodes are connected.

These elements may be mapped to business applications, data stores and data flows.

EA is concerned with standardisation, integration and life cycles of platform applications.

Architecture and architects

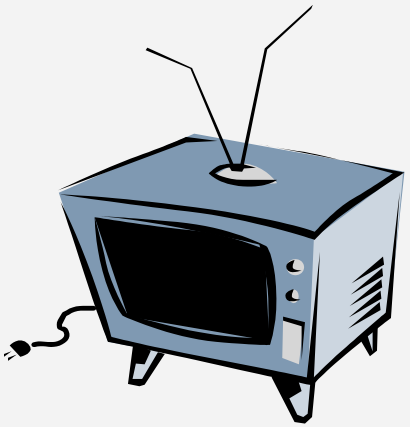
Avancier

Architecture and architects
Basic concepts
Architecture layers as views
Architecture domains as views
System/capability elements
Architecture responsibility level
Architect roles, goals and skills

What is to be architected? A system or capability

Avancier

System



a bounded collection of interrelated elements.

An activity system is

- a network of actors/components playing
- roles in processes to produce desired effects by
- maintaining system state and/or
- transforming inputs into outputs.

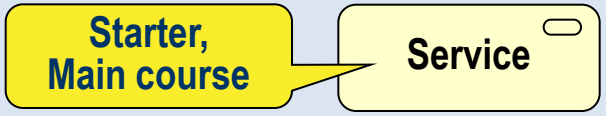
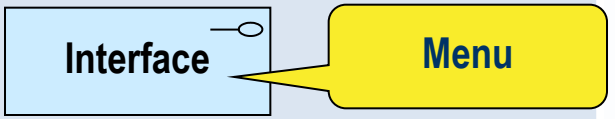
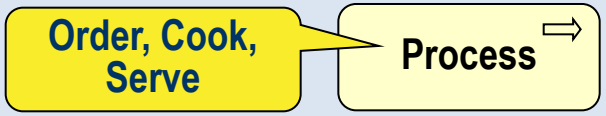
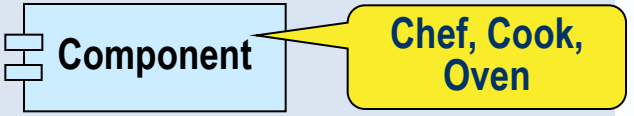
It can be encapsulated behind an interface.

It is open, meaning it interacts with entities and events in its environment.

Structural and behavioural views

Avancier

Structural views	show what a system or capability **is made of**; they show actors and components and how they are connected in structures.
Behavioural views	show what a system or capability **does**; they show services and processes that (repeatedly) run from start to end.

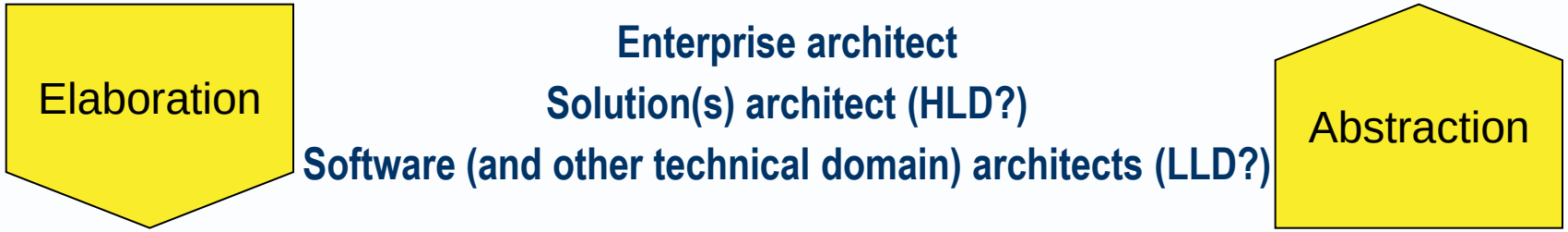
	Behavioural view	Structural view
External view	**Events/Services** trigger or encapsulate the processing of a discrete event or service request 	**Interfaces** present services for access by clients 
Internal view	 **Processes** comprise one or more activities that respond to an event or meet a service request.	 **Actors/Components** are subsystems that performs processes

Structural and behavioural views

- Skip pages 6 and 7 in the manual
- We'll return to them in the second pass

Architecture and architects

Architecture and architects
Basic concepts
Architecture layers as views
Architecture domains as views
Architecture responsibility level
Architect roles, goals and skills



Architect Responsibility Level

Avancier

Architect	a person who designs, plans and oversees the construction of systems. Responsible for responding to requests for work, gathering contextual information, architecting systems and superintending lower level design or construction.

For your amusement only

Architect Responsibility Level

Avancier

Architect role	[a work role] that may be classified by architecture level and/or domain. Higher level roles abstract from detail and operate more widely, with a broader scope. Higher level roles may *govern* lower roles to some extent, but might not directly manage them

The architects' working space				
Architecture facet / domain Architecture level	Business Architecture	Data Architecture	Applications Architecture	Technology Architecture
Enterprise Architecture				
Solution Architecture				
Software Architecture & other Technical Specialisms				

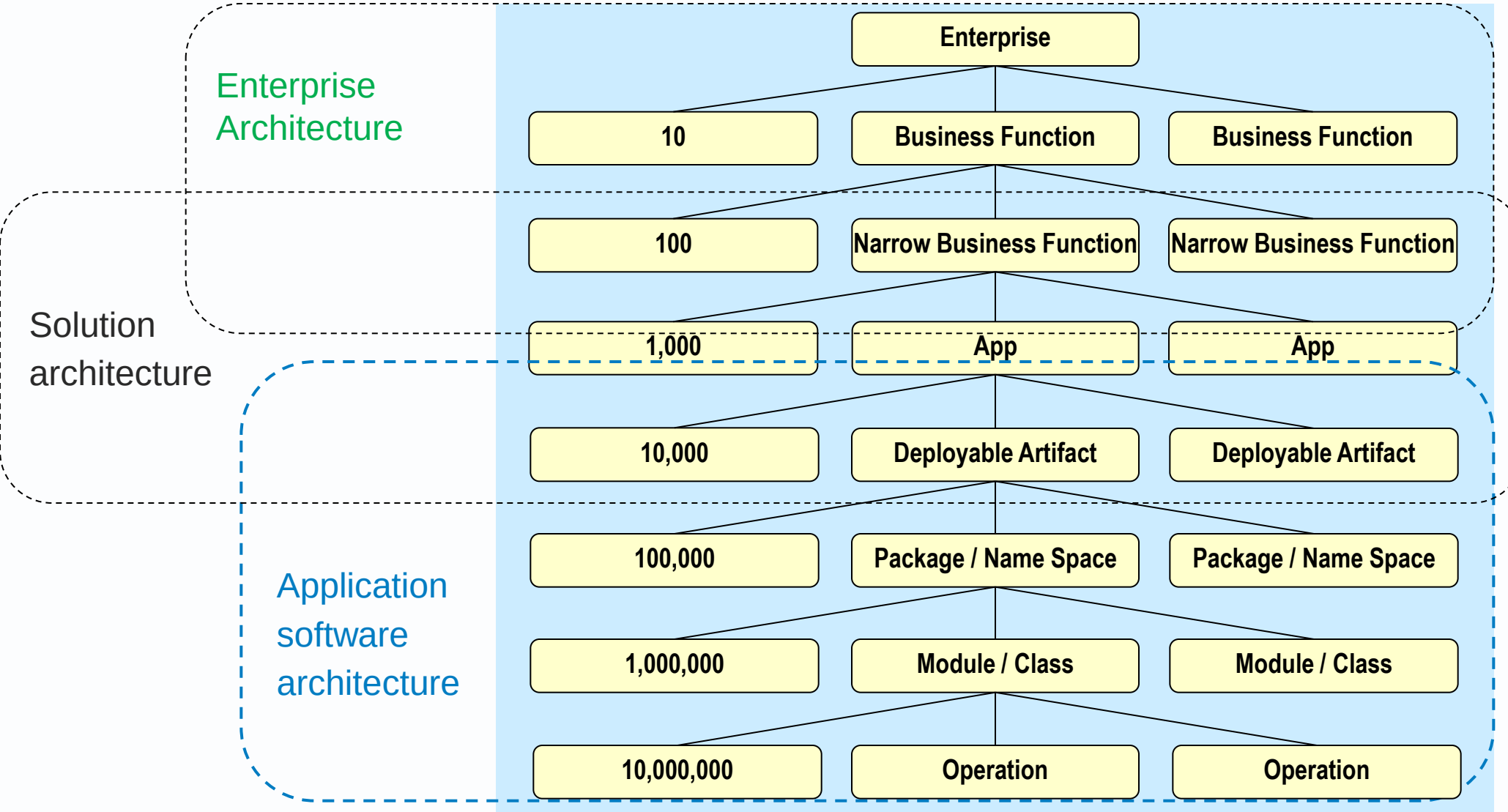
The power and the politics vary widely. An architect may work in roles in more than one domain and at more than one level.

Levels of architecture granularity

Architect level	Abstraction	Scope	Time	Target
Enterprise Architect	High-level	Wide (Enterprise)	Far distant (3 years)	Soft target
Solution Architect	Mid-level	Moderate	Medium time-frame	Flexible target
Software Architect	Low-level	Narrow (A few user stories)	Short time-frame (30 days)	Hard target

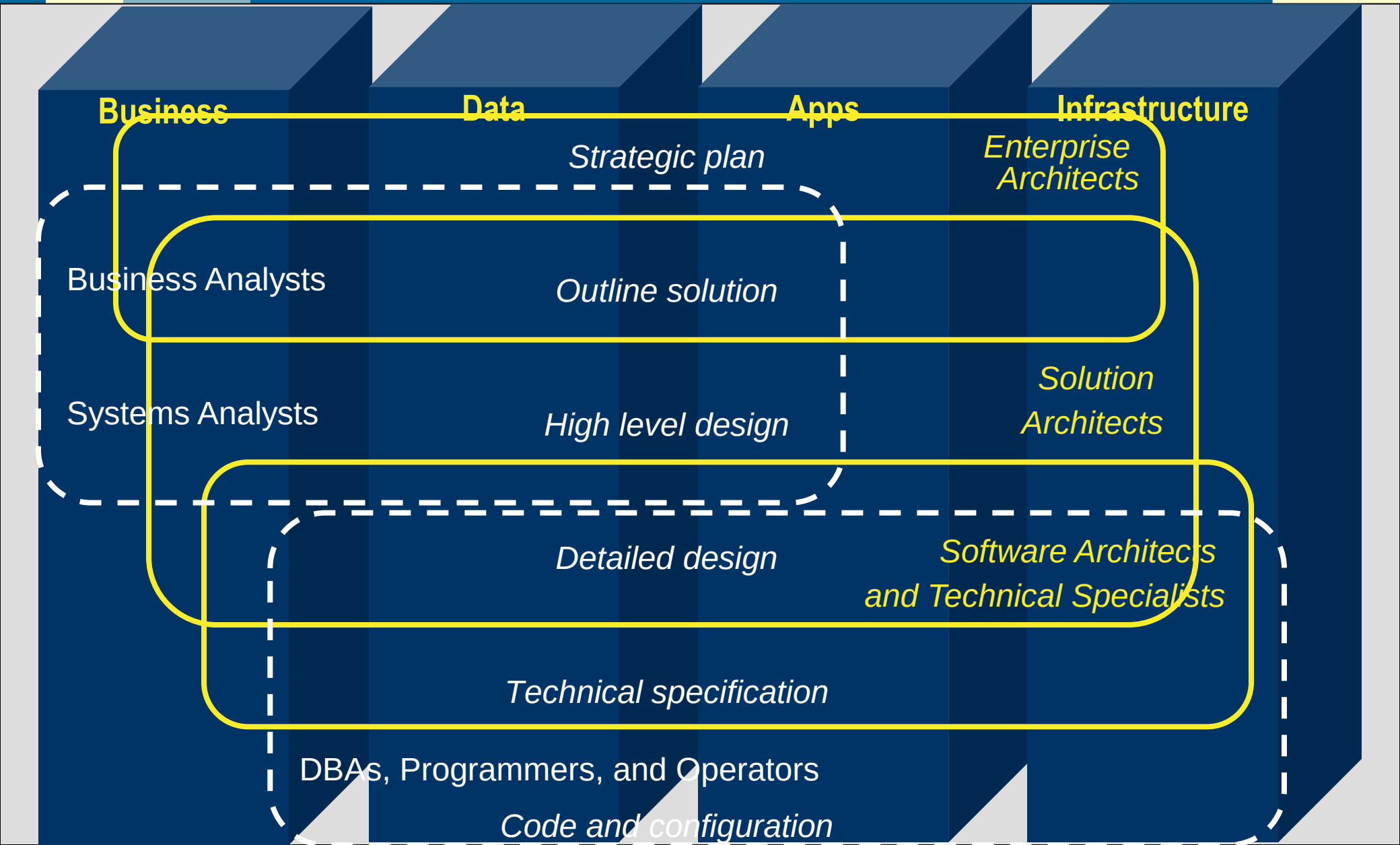
Naïve mapping of responsibility level to enterprise system level

Avancier



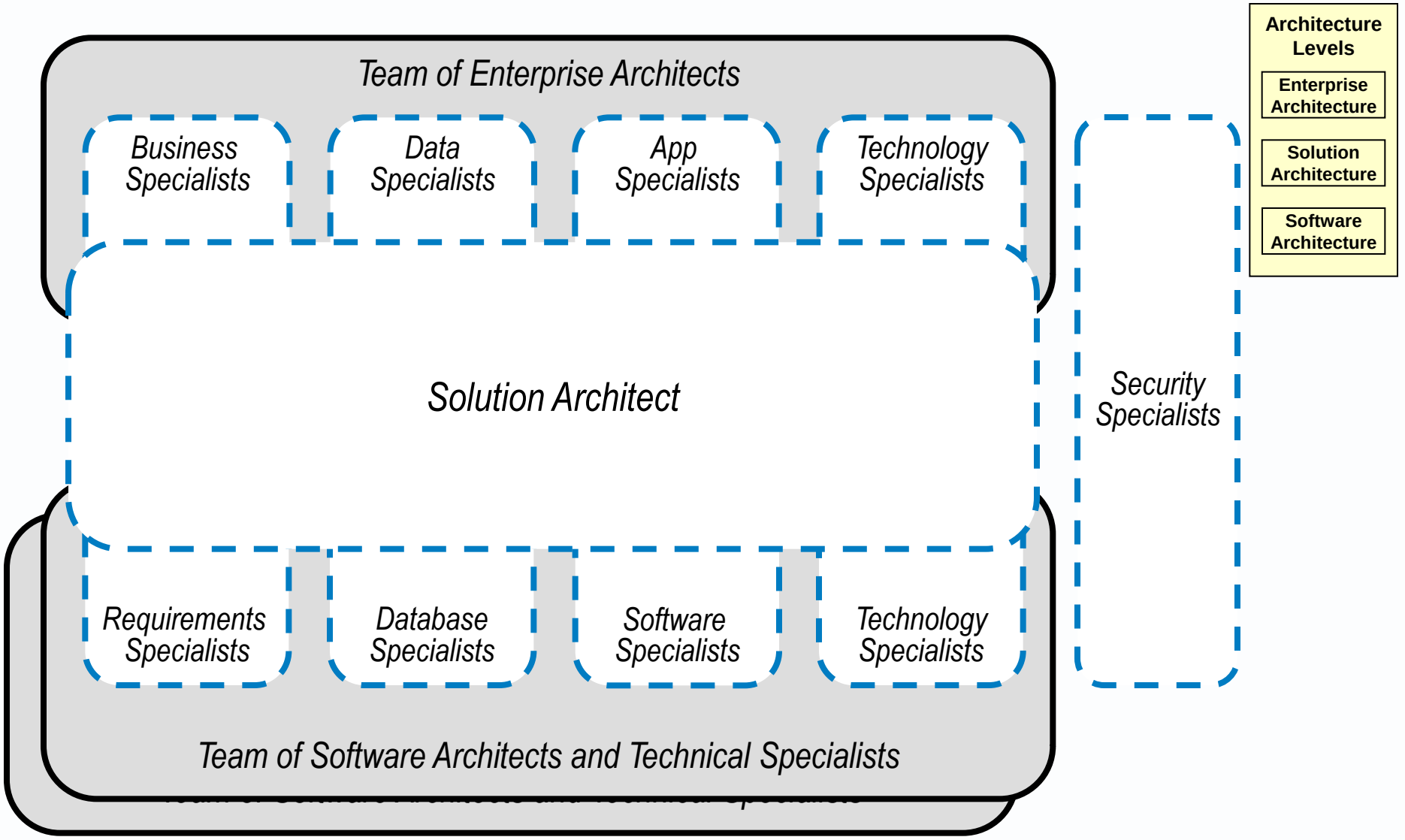
Mapping architecture domains, levels and roles to a traditional process with milestones and specification levels

Avancier



1.4 Architecture responsibility level

Avancier



Architecture and architects – end of pass 1

- SHOW RELEVANT MOCK EXAM QUESTIONS

Generic architecture views/domains/layers

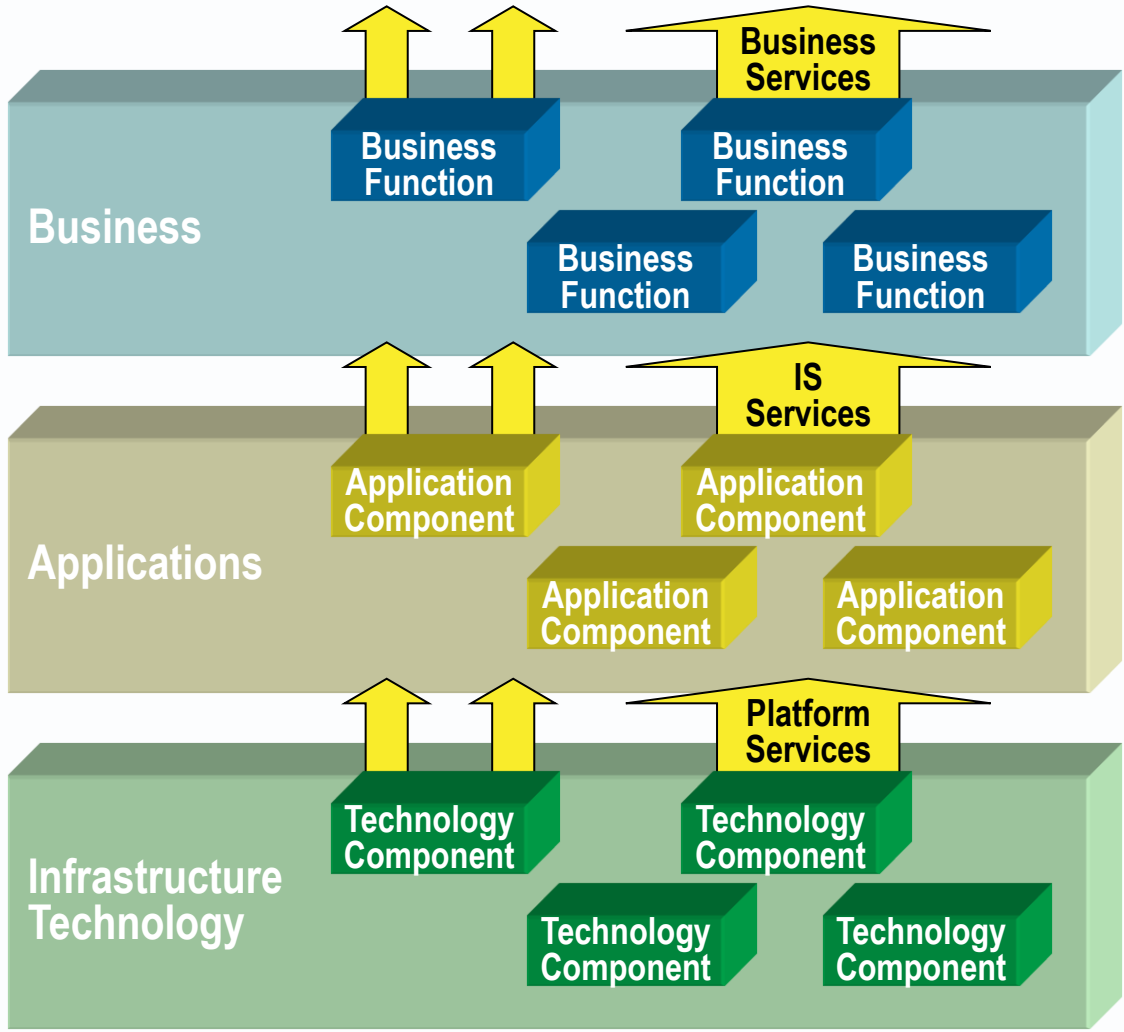
Avancier

TOGAF

- divides human and computer activity systems into views, domains or layers, in which components offer services to the layer above

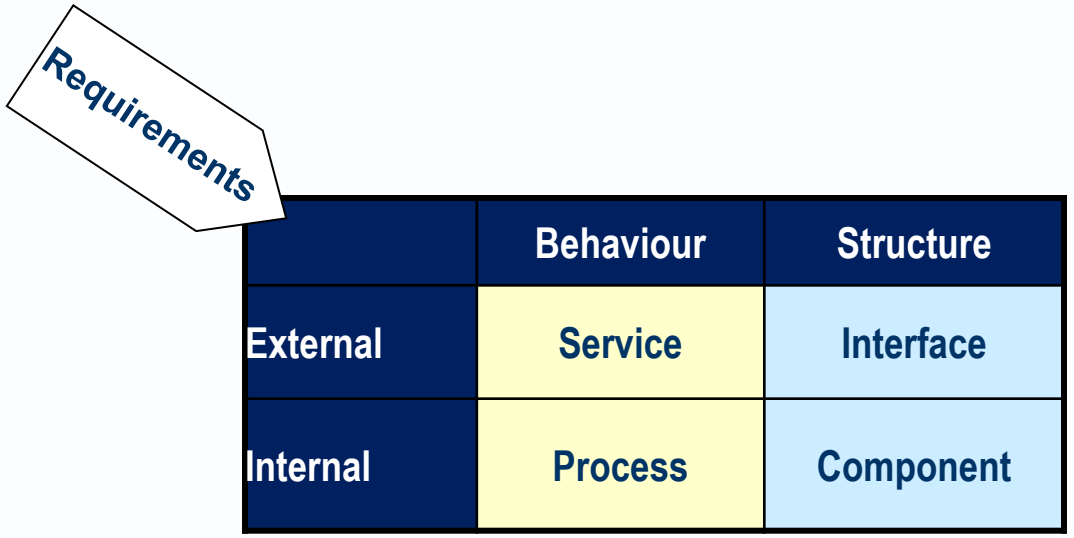
ArchiMate

- Is concerned with “information-intensive organisations” where business processes are supported by systems that capture and provide business data.



Generic system elements

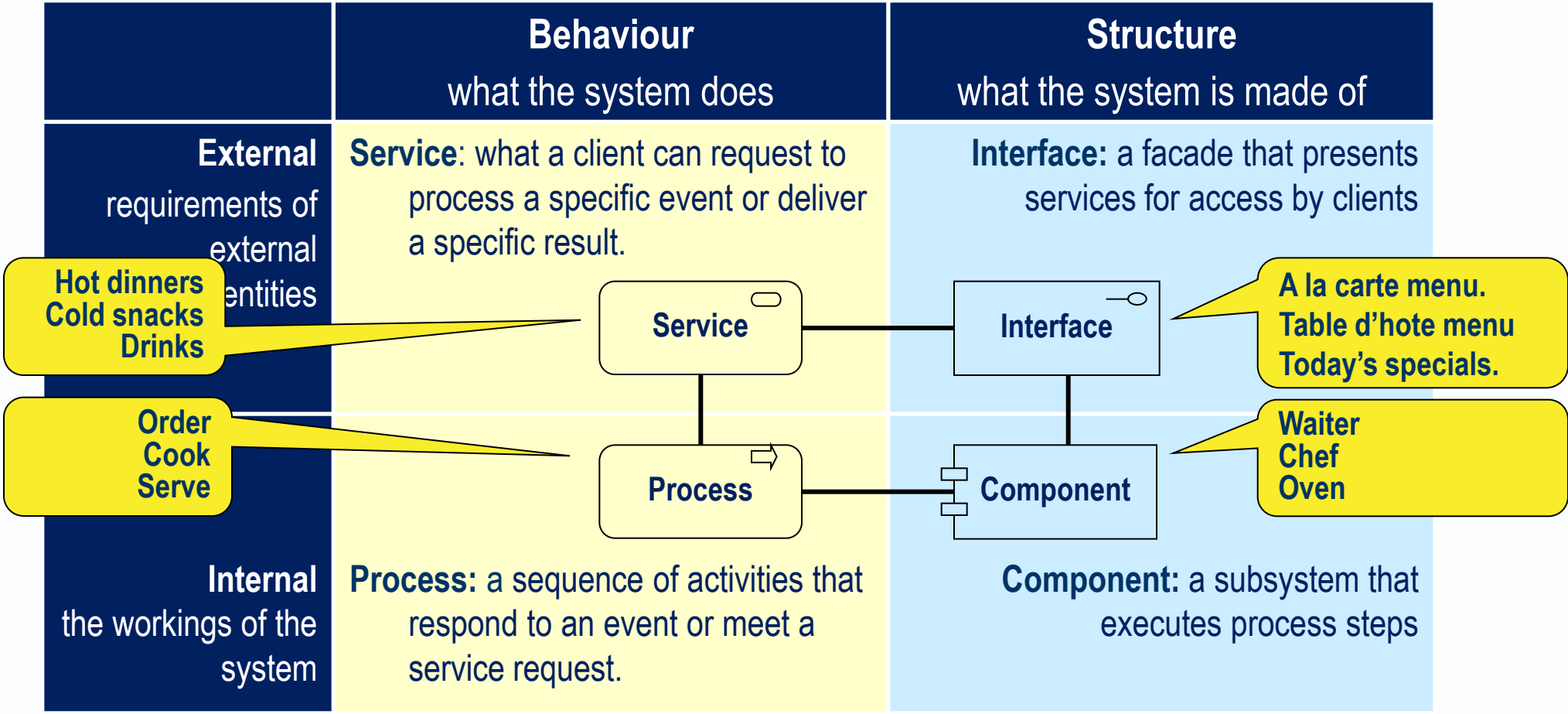
- The ArchiMate standard employs a conceptual framework of generic system elements, which we present in the simplified form below.



- One architect's whole system is another's component in a wider system, so this model can be applied at any level of granularity.
- (There is a fifth kind of system element: an object: an item or structure that is used, moved or made by processes.)

Mapping the conceptual framework to a restaurant

Avancier



STRUCTURAL elements – addressable in space

Avancier

System element	Structural element	Active structural element	Component
			Interface
		Passive structural element	Object
			Location
	Behavioural element	Service	
		Process	

Structural view

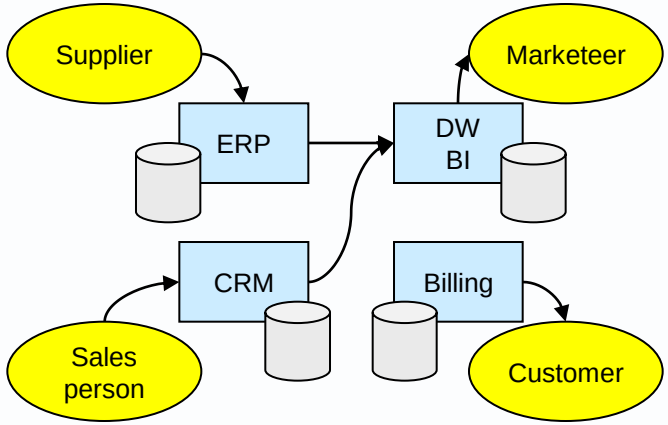
Structure

What a system is made of. A configuration of items.

A collection of inter-related components or services.

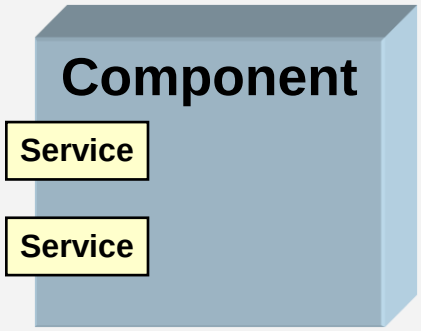
Possible structures include

- hierarchies (items arranged in a cascade of one-to-many relationships) and
- networks (items connected by many-to-many relationships).



What is a component?

Component



A subsystem that

- is encapsulated behind an interface
- can be replaced by any other with the same interface
- is related to other subsystems by requesting or delivering services.

A component can have several interfaces.

- Business component (Function, Actor, Role, Organisation Unit)
- Application component
- Technology component

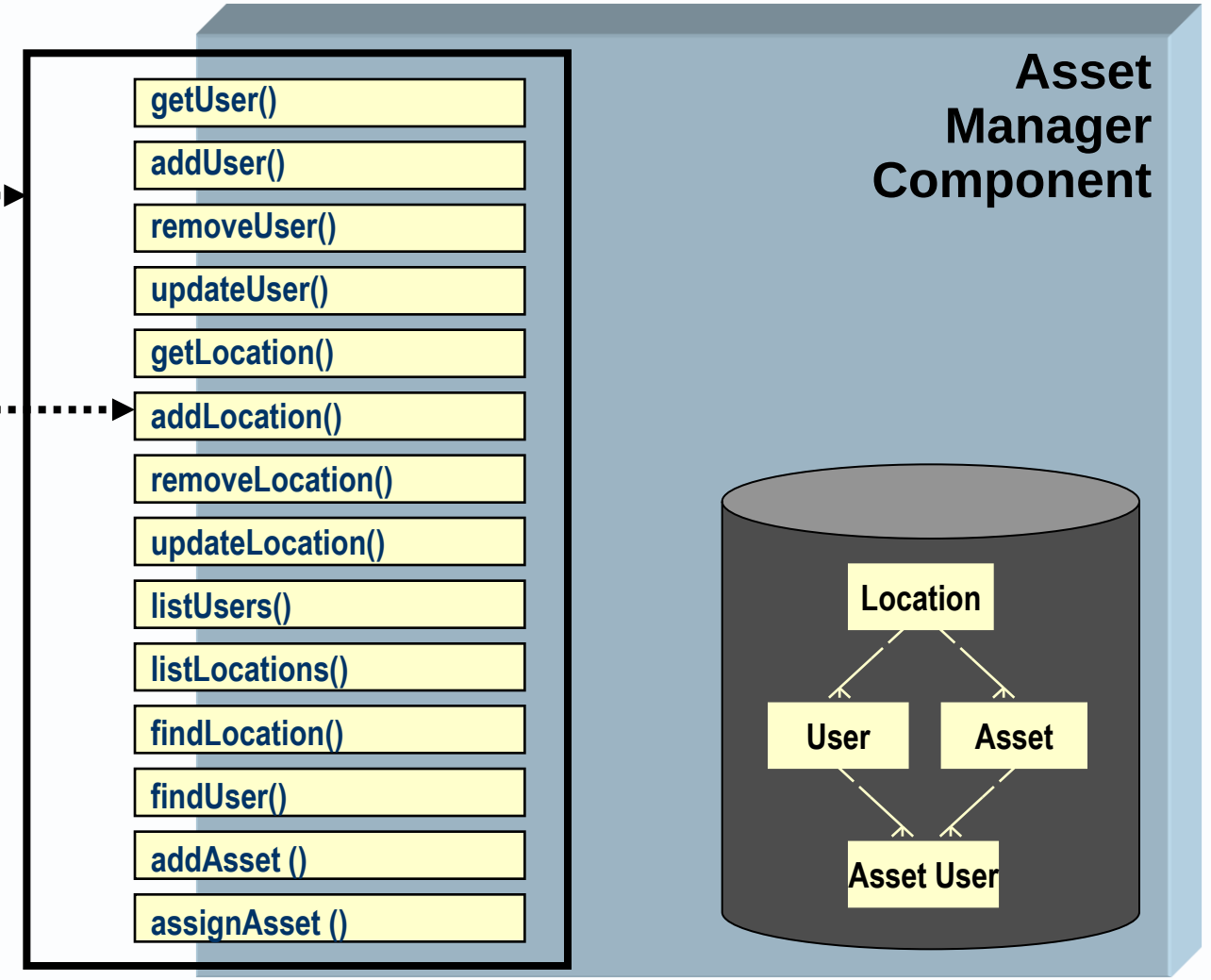
A role/component definition template

Avancier

Role/component	Name
External interface	Listing services a client can request of an actor/ component
Internal processes	Listing activities the actor/component should be able perform (each documentable as a process).
Non-functionals	The place, space, power, skills or other resources an actor/component needs to perform the role defined above.

What is an interface?

- Interface
- Service

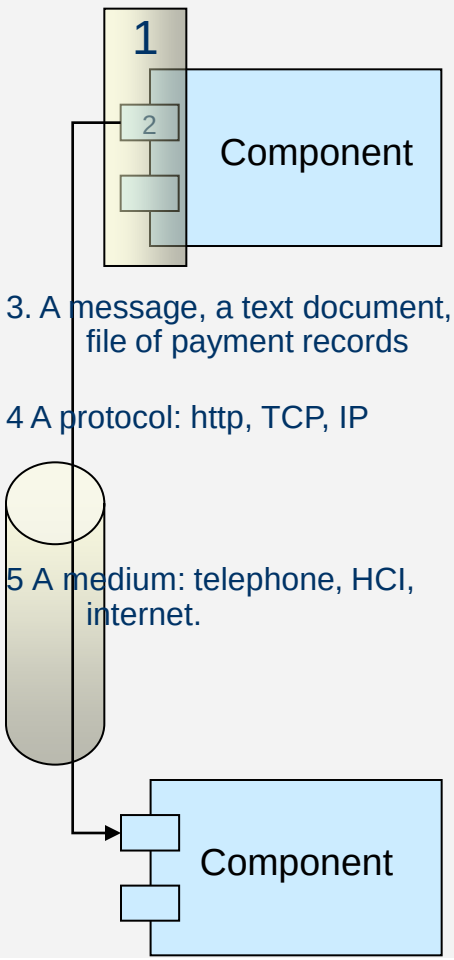


File Transfer Protocol (FTP) – an interface – listing services

Avancier

FTP	An interface implemented by a platform component whose role is to copy files to and from computers. The services below are expressed as in the common FTP utility program on a UNIX computer.
Service name	**Summary description of service contract**
?	to request help or information about the FTP commands
ascii	set the mode of file transfer to ASCII
bye	exit the FTP environment (same as quit)
cd	change directory on the server computer
close	terminate a connection with another computer
delete	delete (remove) a file in the current remote directory (same as rm in UNIX)
get ABC DEF	copies file ABC in the current remote directory to a file named DEF in your current local directory.
get ABC	copies file ABC in the current remote directory to a file with the same name, in your current local directory.
help	request a list of all available FTP commands
mget	copy multiple files from the server computer to the client computer; you are prompted for a y/n answer before transferring each file
mput	copy multiple files from the client computer to the server computer; you are prompted for a y/n answer before transferring each file
open	open a connection with another computer
put	to copy one file from the client computer to the server computer
quit	exit the FTP environment (same as bye)
rmdir	to remove (delete) a directory in the current remote directory

Which do *you* mean by interface?



The means of connecting to a system, process or component. This reference model favours the first of five different ways the term is used

1: A list of services, offered by one or more components.

2: The signature (name, inputs and outputs) of one service.

3: A data flow between sender and receiver components

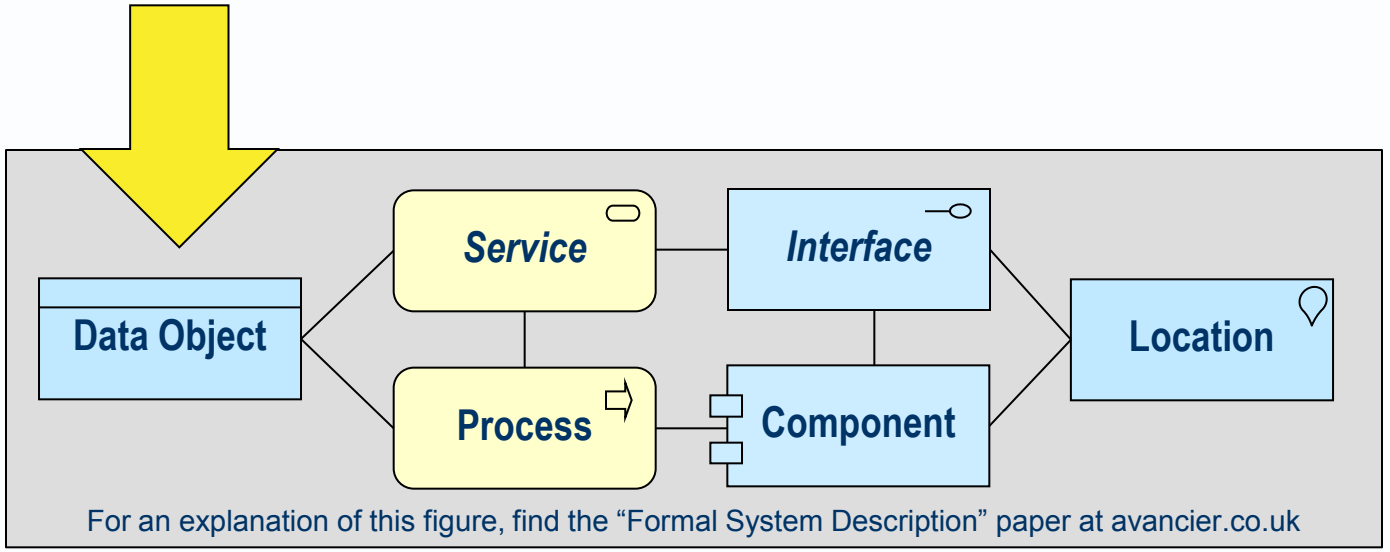
4: The protocols used to exchange data between components.

5: The channel via which data flows are passed

Object

- An item or structure that is used, moved or made.

- E.g. a data item, data structure, or any kind of structural component.



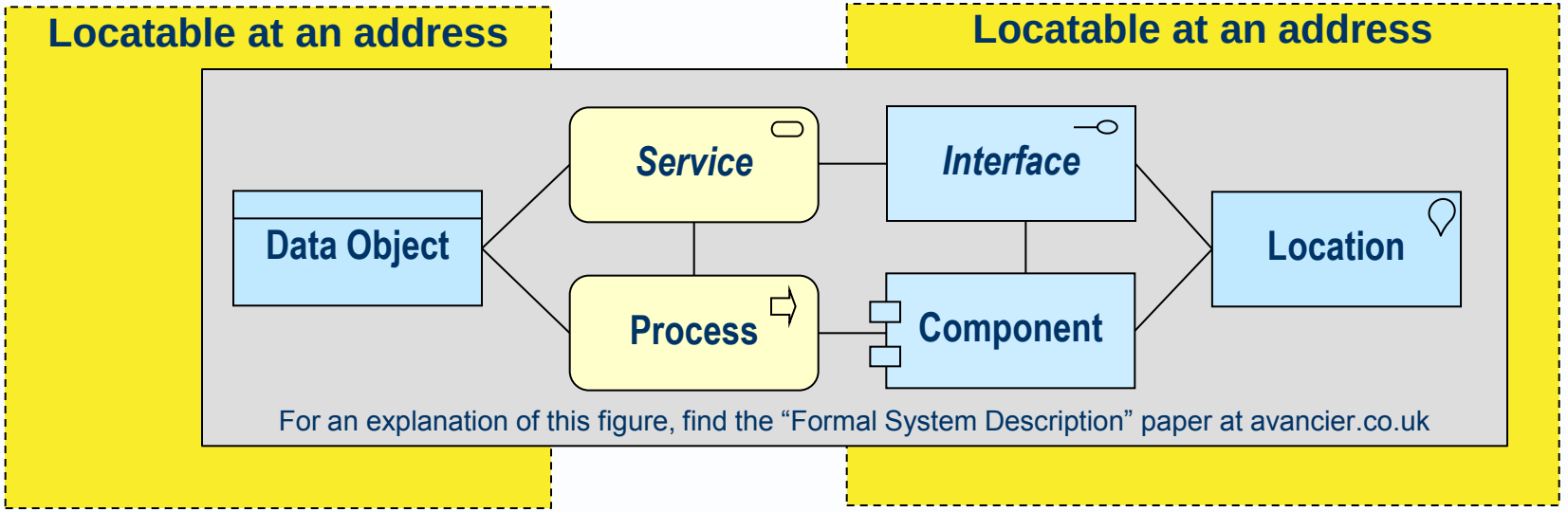
Foundation concepts

Location

A place where business is done, or where computers, applications or their users are.

An identifiable point in space or geography.

An architectural entity that appears in artifacts such as technology deployment diagrams.



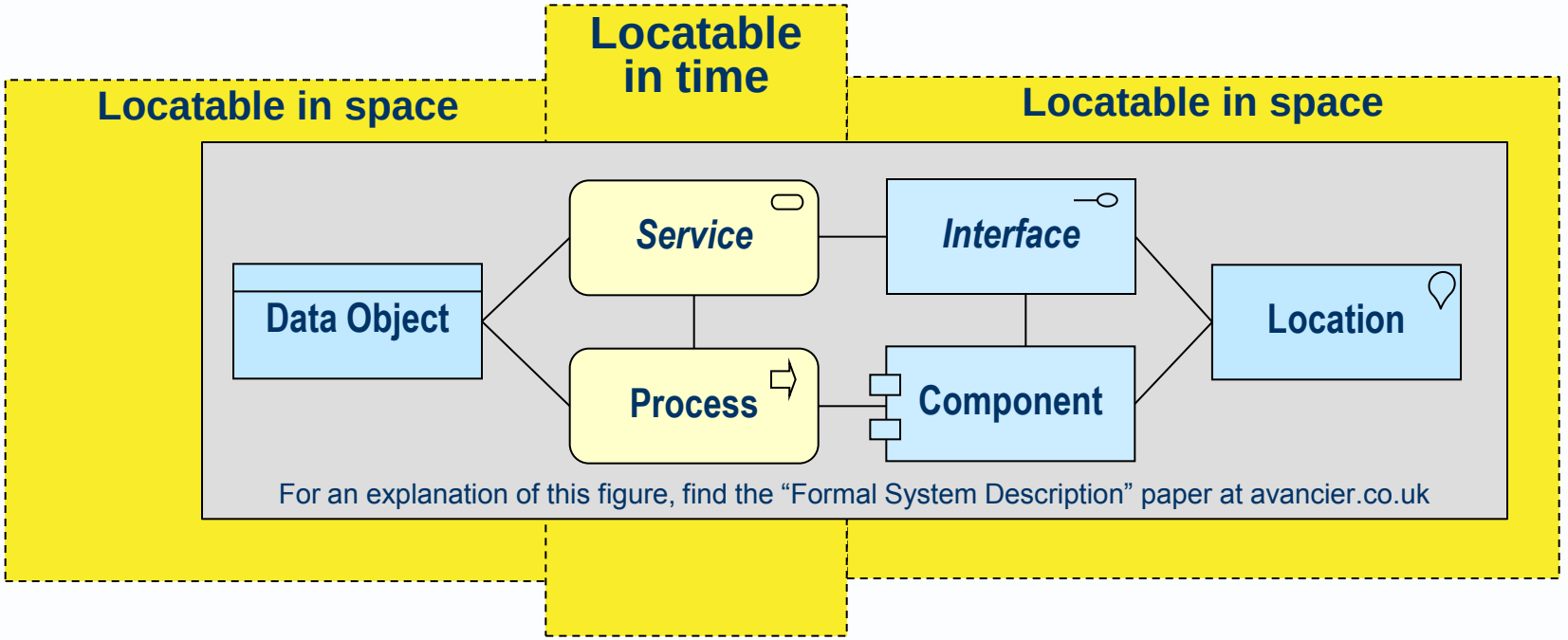
BEHAVIOURAL elements – run from start to end

Avancier

System element	Structural element	Active structural element	Component
			Interface
		Passive structural element	Object
			Location
	Behavioural element	Service	
		Process	

Time Period	A slot in a schedule. An architectural entity that appears in artifacts such as application availability tables.

- Services and processes run from start to end in a time period

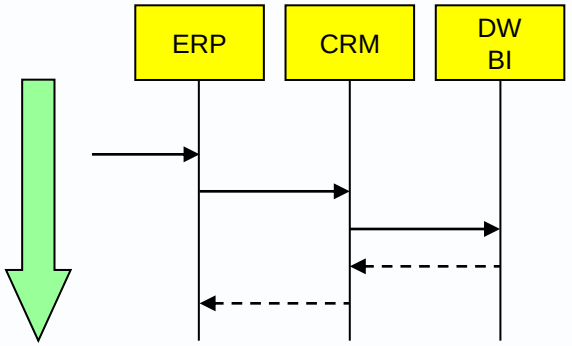


Behavioural elements

Behaviour	What a system does. What external entities observe a system as doing. The processes executed or supported by a system.

- Service
  - Business service
  - IS Service (Use Case or Automated IS Service)
  - Platform Service

- Process
  - Business process
  - Application process (Use Case)



Service *a discretely invokable operation*

- FTP examples include
  - Open, Close
  - Put, Get. Delete
- TOGAF examples include:
  - check customer credit
  - provide weather data
  - consolidate drilling reports
- ArchiMate examples include
  - policy creation
  - premium payment
  - claim registration

- A service encapsulates processes needed to deliver the desired result, and is describable using a service contract.

What is a service contract?

Avancier

Service contract

[an artefact] a template for defining the properties of a service, which encapsulates a process. It is divisible into three main parts.

- The **signature:** the service name, inputs and outputs.
- The **rules:** the preconditions and post conditions of the service.
- The **non-functional** characteristics: including performance and commercial conditions.

Service Contract	Business Service	999
Signature	Name	Cut hair
	Input	Hair length
	Output	Shorter hair (see also post conditions)
Functional rules or semantics	Preconditions	Barbershop open and barber ready
	Post conditions	Money received. Resource wear.
Non-functional requirements	Response time	45 minutes
	Throughput	6 per hour per shop
	Availability	90% waiting times less than 20 minutes from 09.00 to 17.00

Infrastructure service contract example

Avancier

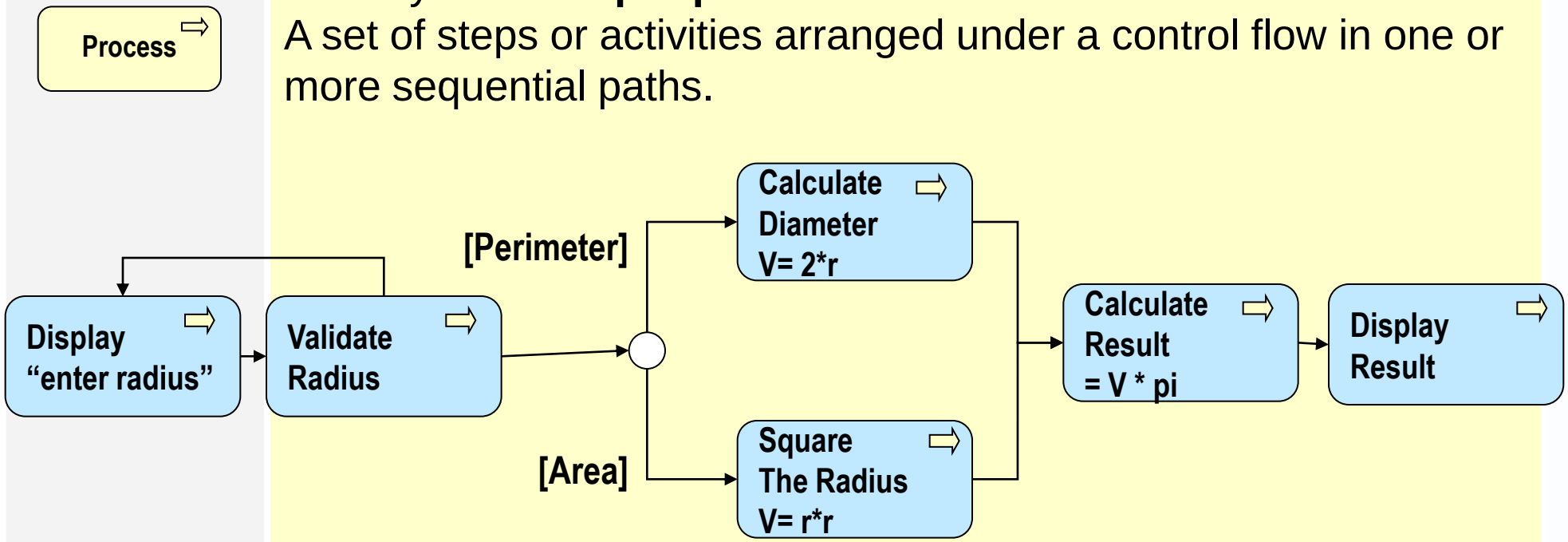
Service contract for FTP “get” operation		
Signature	Name	get
	Inputs	Remote file name Local file name
	Outputs or results	Reply = OK or Fail (see post conditions)
Functional rules or semantics	Preconditions - the state of the system in which the event is allowed	Remote computer can be reached. Remote file exists in the current remote directory.
	Post conditions - the state of the system after the event is complete	Remote file copied to (or on top of) local file current local directory.
Non-functionals	Response time	Time out = 30 seconds
	Throughput	20 per minute
	Availability	99.99%
	Integrity	100% perfect file copy
	Scalability	Up to 100 per minute
	Security	No encryption
	Serviceability	
	Etc. Other non-functionals, dependencies and commercials.	

What is a process?

Avancier

Process

1: A procedure, started by an **event**, which terminates with the delivery of an **output product or service.**
A set of steps or activities arranged under a control flow in one or more sequential paths.



OR 2: An encapsulated system or subsystem - as in the "active process" on a computer. But this RM favours definition 1.

Process

- There are human processes and computer processes
- There are hybrid human-computer processes (sometimes called workflows or use cases.)

- A process is
  - described
    - using an artefact like a flow chart, use case definition or interaction diagram
  - made implementable
    - by defining actions at the level understandable by actors/components.
  - deployed
    - by publishing it where actors/components can read it
  - performed
    - when actor/computers read and follow the process.

Use case definition = process flow inside a service contract

Avancier

Signature	Name
	Inputs
	Outputs or results
Trigger event	The event that causes the process to start
Functional rules or semantics	Preconditions - the state of the system in which the process is allowed
	Post conditions - the state of the system after the process is complete
Process flow	
Non-functionals	Response time
	Throughput
	Availability (derived from reliability and recoverability)
	Integrity
	Scalability
	Security
	Serviceability
	Etc. Other non-functionals, dependencies and commercials.

Use case = service contract *encapsulating a* process

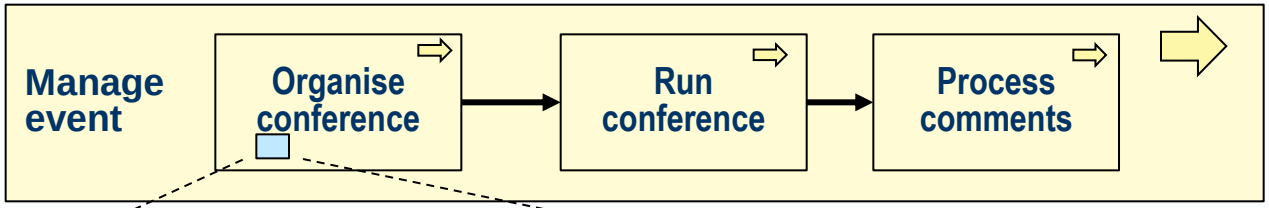
Avancier

Signature	Name
	Inputs
	Outputs or results
Trigger event	The event that causes the process to start
Functional rules or semantics	Preconditions - the state of the system in which the process is allowed
	Post conditions - the state of the system after the process is complete
Process flow	The flow of the process may be defined in narrative, as a main path and alternative paths, or else in a diagram • A conventional process flow chart, or • Activity diagram (UML), or • Sequence diagram (UML), or • Business process diagram (ArchiMate), or • Application behaviour diagram (ArchiMate).
Non-functionals	Response time
	Throughput
	Availability (derived from reliability and recoverability)
	Integrity
	Scalability
	Security
	Serviceability
	Etc. Other non-functionals, dependencies and commercials.

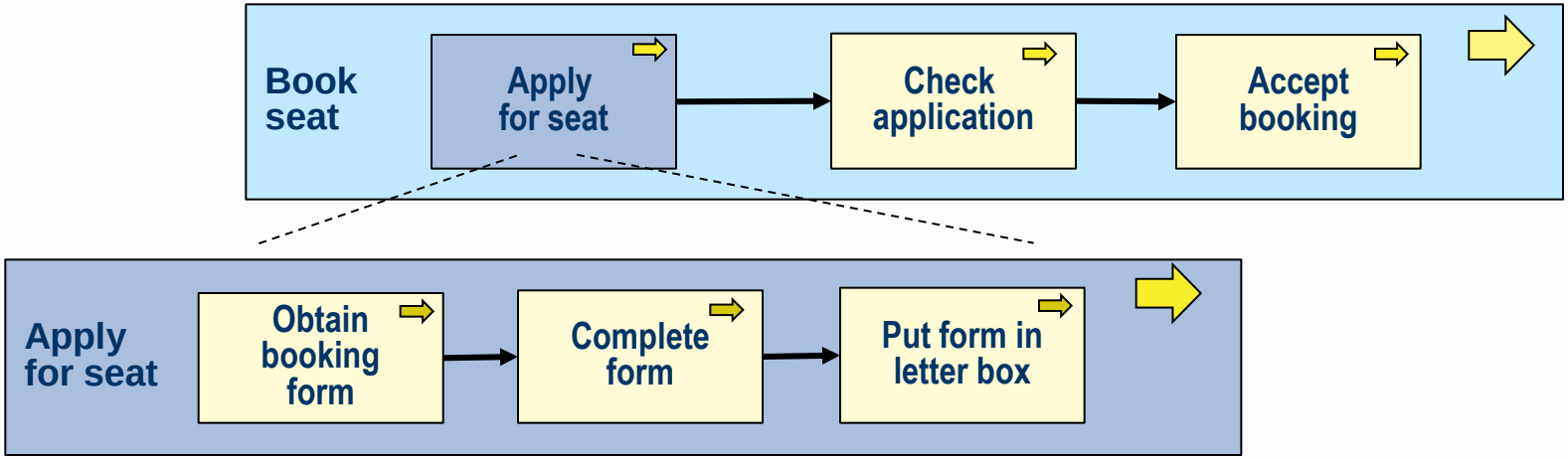
Service/process decomposition

Avancier

- A process if coarse-grained processes/steps/activities



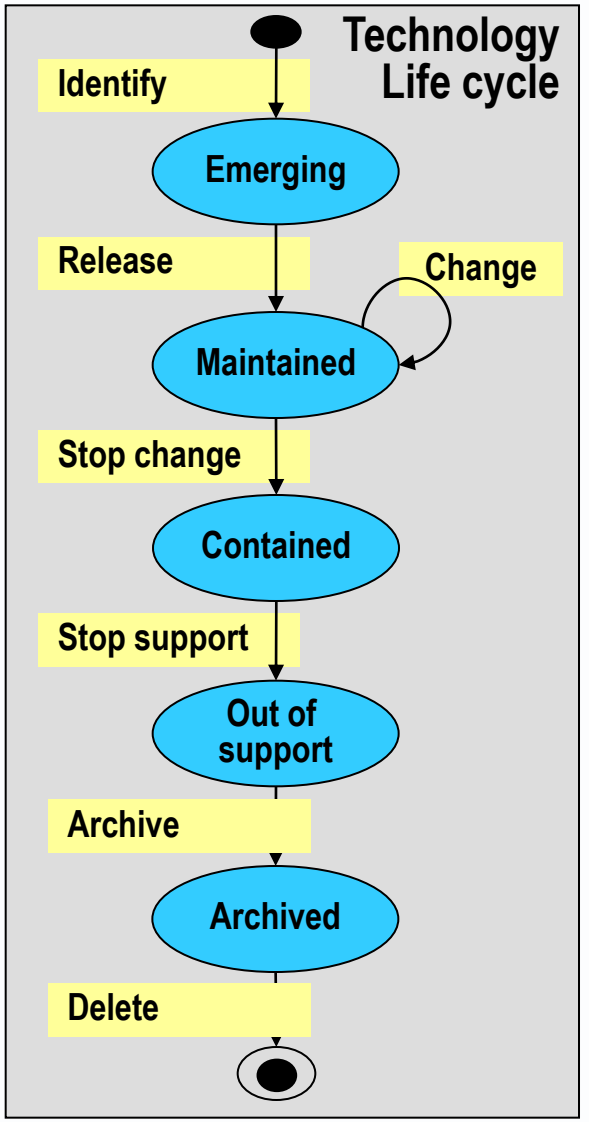
- Can be decomposed into finer-grained processes/steps/activities



Another kind of process

Avancier

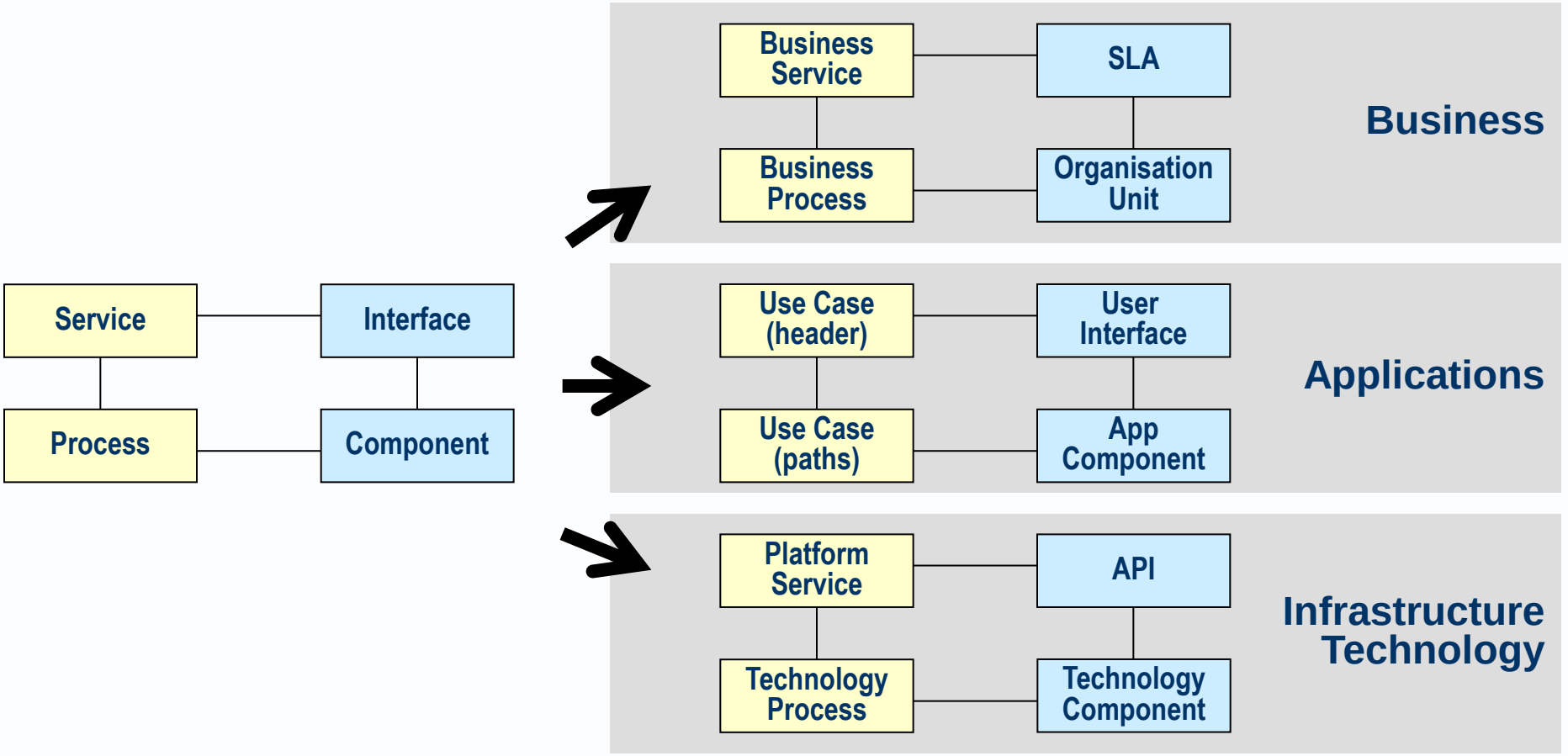
Life cycle	A process from birth to death [of a thing] E.g. the life of: ▶ a system from conception through deployment and use to removal ▶ a project from conception through development to delivery ▶ an entity [See Data Lifecycle.]



- The life cycle can be defined in terms of:
  - states
  - events that trigger state transitions
  - actions that are performed on a state transition
  - actions that are performable while in a state.

Mapping the conceptual framework to architecture domains

- The generic system elements can be specialised in each enterprise architecture domain/layer as shown below.



1. Architecture and architects – end of pass 2

- SHOW RELEVANT MOCK EXAM QUESTIONS

Architecture and architects

Architecture and architects
Basic concepts
Architecture layers as views
Architecture domains as views
Architecture responsibility level
Architect roles, goals and skills
Other architecture domains

The architects' working space

Avancier

- EA tends to be more abstract in every possible way,
  - More generic - a higher level of generalisation
  - More conceptual - a higher level of idealisation
  - More coarse-grained - a higher level of granularity

The architects' working space				
Architecture facet Architecture level	Business Architecture	Data Architecture	Applications Architecture	Technology Architecture
Enterprise Architecture				
Solution Architecture	Abstraction	Abstraction	Abstraction	Abstraction
Software Architecture & Technical Specialisms				

Levels of architecture granularity

Avancier

Enterprise architecture	[a responsibility level] that is **strategic: the highest, widest, longest term** kind of architecture; treats a whole enterprise as a system. ❖aims to optimise an enterprise's many systems to remove redundancy, to standardise and integrate systems ❖is responsible for addressing cross-organisational concerns and goals ❖understands the enterprise's estate and maintains enterprise architecture collateral ❖sets out cross-organisational and/or strategic road maps and migration programmes ❖guides other architects on cross-organisational standardisation and integration opportunities.
Solution(s) architecture	[a responsibility level] that is **relatively tactical: addresses specific problems and requirements**, related to selected business processes and applications. ❖aims to ensure the quality of a particular solution delivery, in compliance with overarching goals, principles and standards where possible. ❖describe solutions and govern their delivery, usually at a project level ❖understands all domain/views well enough to work with others: e.g. a general solution architect may work in partnership with a lead business analyst and/or software architect ❖ensures the architecture of a system is sufficiently detailed for work to be planned and detailed design and building to proceed ❖shapes projects and then govern others according to agreed principles and plans ❖is responsible for delivery quality: focuses on critical success factors, especially non-functional qualities.

Avancier

Enterprise architect goals

The list of goals below has been agreed as sufficient for examination purposes.

1. Improved alignment of business and IT
2. Improved IT cost-effectiveness
3. Business agility
4. Technical agility.
5. Long term planning: enablement of strategically beneficial IS/IT work.
6. [Portability:] Vendor and technology independence
7. [Standardisation:] De-duplication of applications and technologies
8. [Integration:] Interoperation of applications and technologies
9. Simpler systems and systems management.
10. Improved procurement.

Avancier

Solution architect goals

The list of goals below has been agreed as sufficient for examination purposes.

The solution architect supports the goals of an enterprise architect, but focuses more on:

1. Timeliness of IS/IT project **deliverables**
2. Cost of IS/IT project **deliverables**
3. Quality of IS/IT project **deliverables**
4. Solution-level risk identification and mitigation
5. Application integration and data integrity
6. Conformance of solution to non-functional and audit requirements
7. Conformance of solution to principles, standards, legislation.
8. Effective interaction between managers and technicians.
9. Governance of detailed design to architecture principles and standards.

Architects

Avancier

Architect knowledge and skills

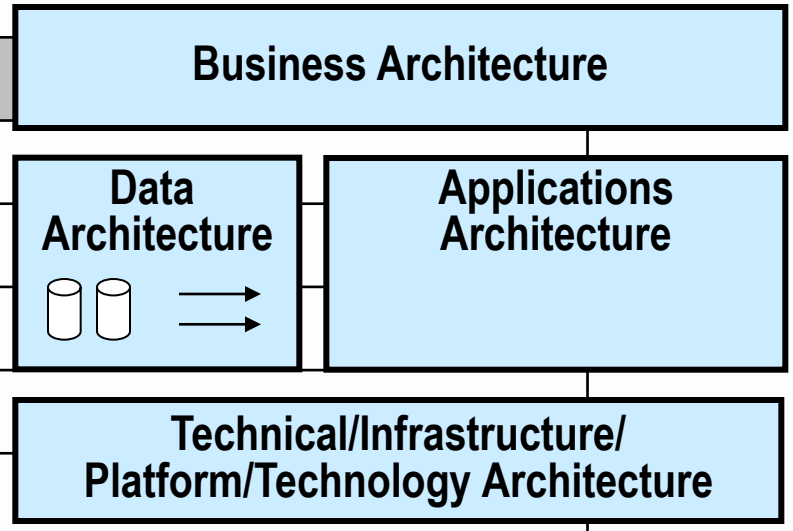
The list below has been agreed as sufficient for examination purposes.

1. Holistic understanding of business and technical goals.
2. Holistic understanding of business and technical environment
3. Broad technical knowledge – including current trends.
4. Broad methodology knowledge
5. Analysis of requirements and problems
6. Innovation.
7. Leadership.
8. Stakeholder management.
9. Communication, political and soft skills.
10. Awareness of project management and commercial risks and issues.

1. Architecture and architects

Avancier

- 1. Architecture and architects
- Basic concepts
- Architecture layers as views
- Architecture domains as views >>>>
- Architecture responsibility level
- Architect roles, goals and skills
- Other architecture domains

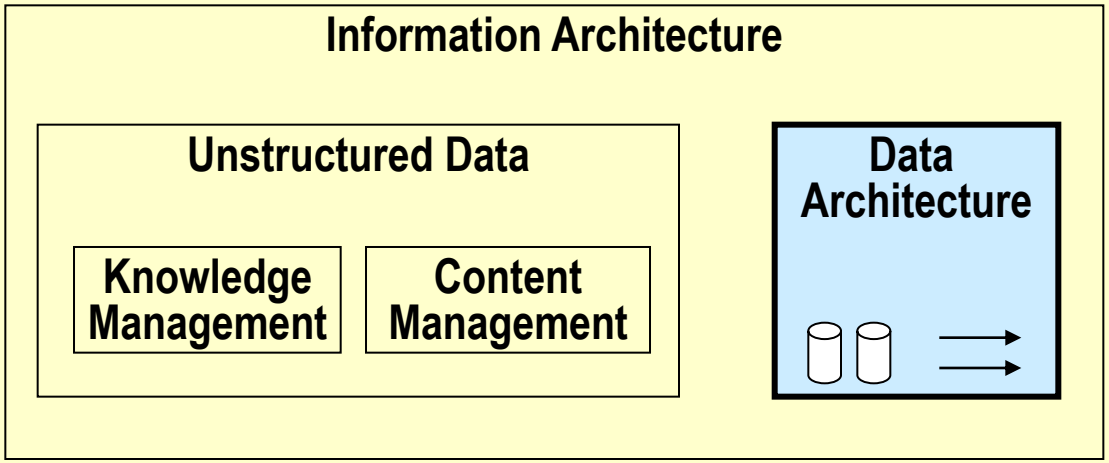


Other architecture domains

Avancier

Information architecture

The broad domain that includes structured data architecture, content management and knowledge management.



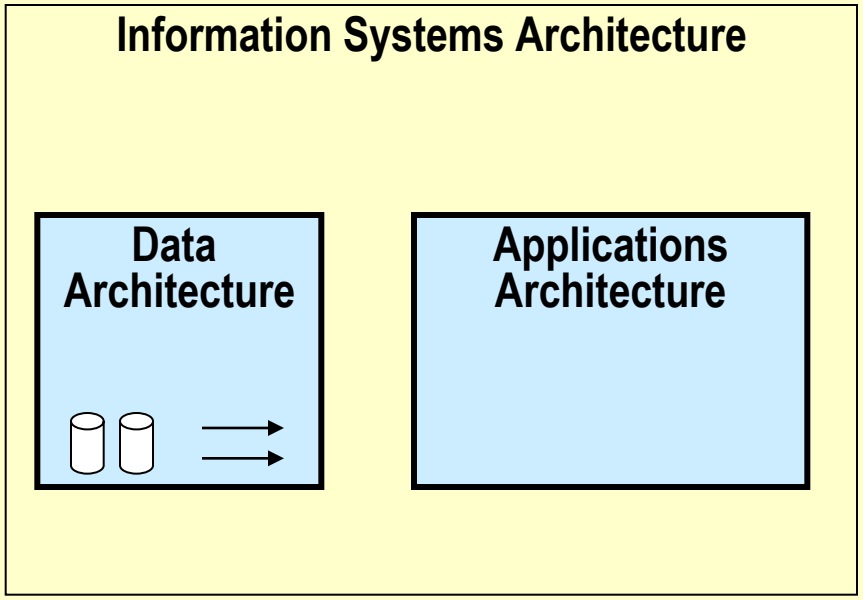
(This reference model focuses on structured data.)

Other architecture domains

Avancier

Information systems architecture

Often used to mean the combination of applications architecture and data architecture.

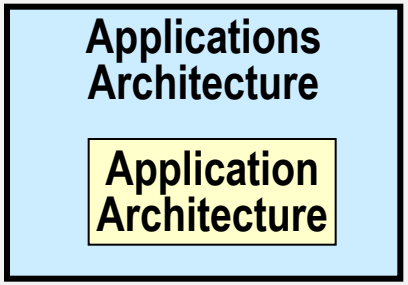


It depends on but does not include the technology platform.

Other architecture domains

Avancier

Software (aka application) architecture



The internal structure, the modularisation of software, within an application.

This is software architecture at the lowest level of granularity (as discussed for example in “Patterns of enterprise application architecture” by Martin Fowler).

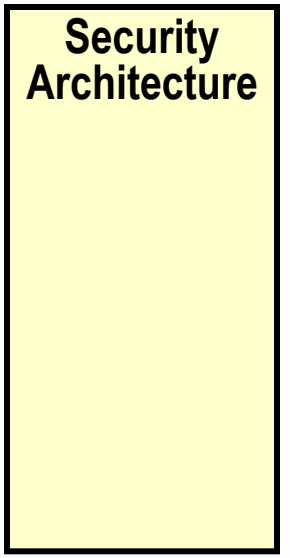
It is usually below the level of modularity that enterprise and solution architects define. However, there is no rigid dividing line.

Other architecture domains

Avancier

Security architecture	The various design features designed to protect a system from unauthorised access. Not a cohesive architecture on its own so much as features of the other architecture domains – business, data, applications and infrastructure.

- All domains may be vulnerable to breaches of security and therefore may require design to prevent those breaches
- Human and organisational security considerations
  - E.g. gates, guards and guns
- Data security considerations
  - E.g. encryption
- Application security considerations
  - E.g. user roles, identities, passwords
- Infrastructure security
  - E.g. firewalls



Mapping architecture domains to reference model sections

- The four primary domains are 4, 5, 7 and 9
- Knowledge of the domains numbered 6 and 8 helps