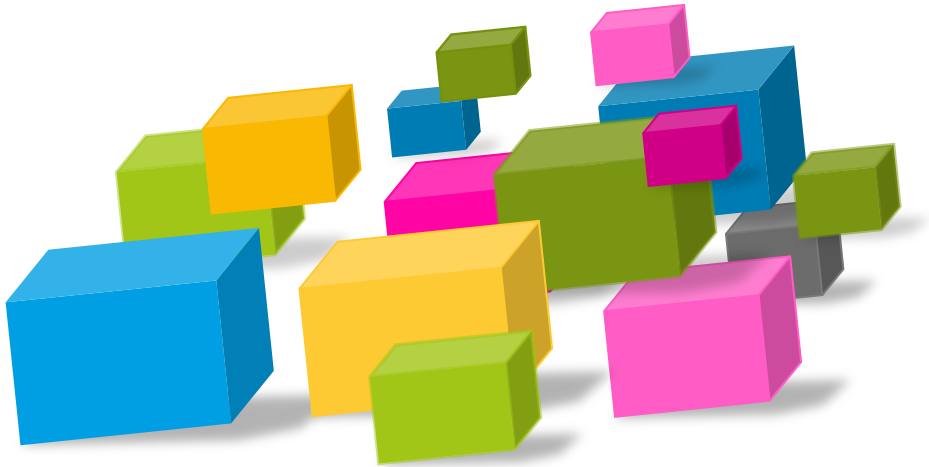
Archimate

Architecture Modelling

Kim Horn

Version 0.3 **3 August 2021**



Why Model Systems ?

Two main reasons for Modelling:

1. To explain something visually;
2. As a tool to work with large complexity, and change.

Realities:

- Models are conceptual, no right answer
- Multiple Models of a Single Reality
- Each stakeholder has a particular *viewpoint*, needs a particular *view*. These is no one set.
- They take effort to build and maintain
- The code cannot explain the architecture: principles, practises, trade-offs, justified decisions, research, the system is more than sum of parts.

Most modelling attempt use a made up language, e.g. PowerPoint Speak. The 'Boxes and Lines', are arbitrary. There is no shared understanding of the grammar, the nouns, verbs etc. Results in poor communication, and confusion. We need a proper language for Architecture.

What is this ?

"This is not a pipe" as it is only the representation of a pipe.

A slide with a photo of a painting of specific person's (Magritte's) representation of a pipe.

A model is not the reality, only a useful abstraction

Agile Architecture

- The UML approach to documenting systems did not work:
  - It reflects code – so why duplicate.
  - Low level diagrams get out of date, become useless quickly.
  - Why document design ?

What's required is the next level up, the architecture. This is not captured in code:

- Requirements, impacting the solution. Non-Functional/Constraints
- Business Process – Interface to business.
- Org structure – Conway's Law
- Capabilities – what the system is automating for the business.
- Integrations/Contracts – the code does no tell us about actors that depends on this system.
- Architecture Decisions, Research/Spike Outcomes.
- Impact of change, across the enterprise:
- Representations (Visual aid) to help gain understanding and commitment from Stakeholders, or to help resolve high level decisions.

EA Architecture Modelling

EA tools:

- allow organizations to examine both the need for, and the impact of, change.
- capture the interrelationships and interdependencies within and between an ecosystem of partners, operating structures, capabilities, processes, applications and technologies.
- provide a central repository to capture knowledge, data and metadata about the artefacts that an enterprise cares about — all types of objects and assets, and their related life cycles.
- capture models to represent the relationships between these objects, and are themselves treated as assets that help describe and shape the future of the enterprise.
- help with investment decisions at the level of both IT and the broader enterprise.

ArchiMate is an open and independent enterprise architecture modelling language to support the description, analysis and visualisation of architecture within and across business domains in an unambiguous way.

- It is a technical standard from The Open Group and is based on the concepts of the IEEE 1471 standard.
- It distinguishes itself from other languages such as UML and BPMN by its enterprise modelling scope.
- Offers a common language for describing the construction and operation of business processes, organizational structures, information flows, IT systems, and technical infrastructure.
  - This insight helps the different stakeholders to design, assess, and communicate the consequences of decisions and changes within and between these business domains.
- It has a linguistic, layered and service-oriented approach to modelling. The higher layers make use of services that are provided by the lower layers.
  - The concepts that are used within each layer are similar, but specific to each layer.

Why was ArchiMate invented ?

The original research says this:

“Because architectures are often complex and hard to understand, architects need ways to express these architectures as clearly as possible: both for their own understanding and for communication with other stakeholders, such as system developers, end-users and managers.

To date, there is no standard language for describing architectures; they are often described in informal pictures that lack a well-defined meaning. This leads to misunderstandings, and makes it very difficult to provide tools for visualisation and analysis of these architectures.”

Layers and Aspects

Core Layers (BAT) supports the Modelling of Solutions

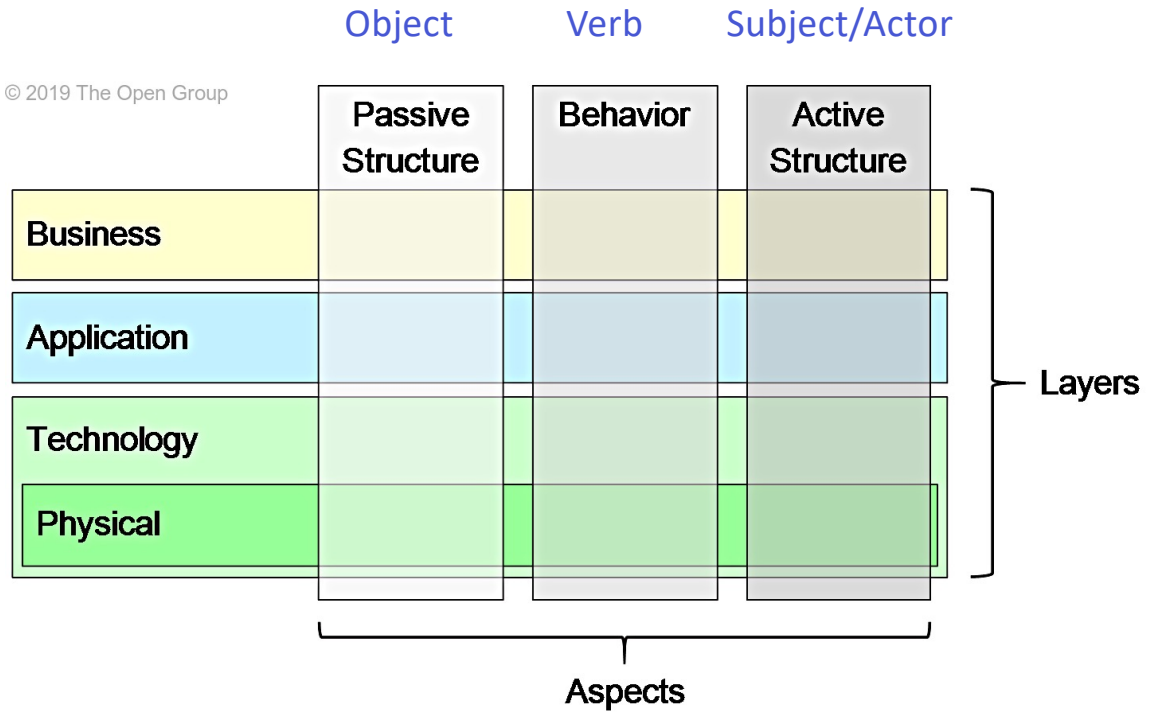
Active Structure Aspect, represents the structural elements, the business actors, application components, and devices that display actual behavior; the "subjects" of activity.

Behavior Aspect, represents the behavior, processes, functions, events, and services, performed by the actors;

- structural elements are assigned to behavioral elements, to show who or what displays the behavior.

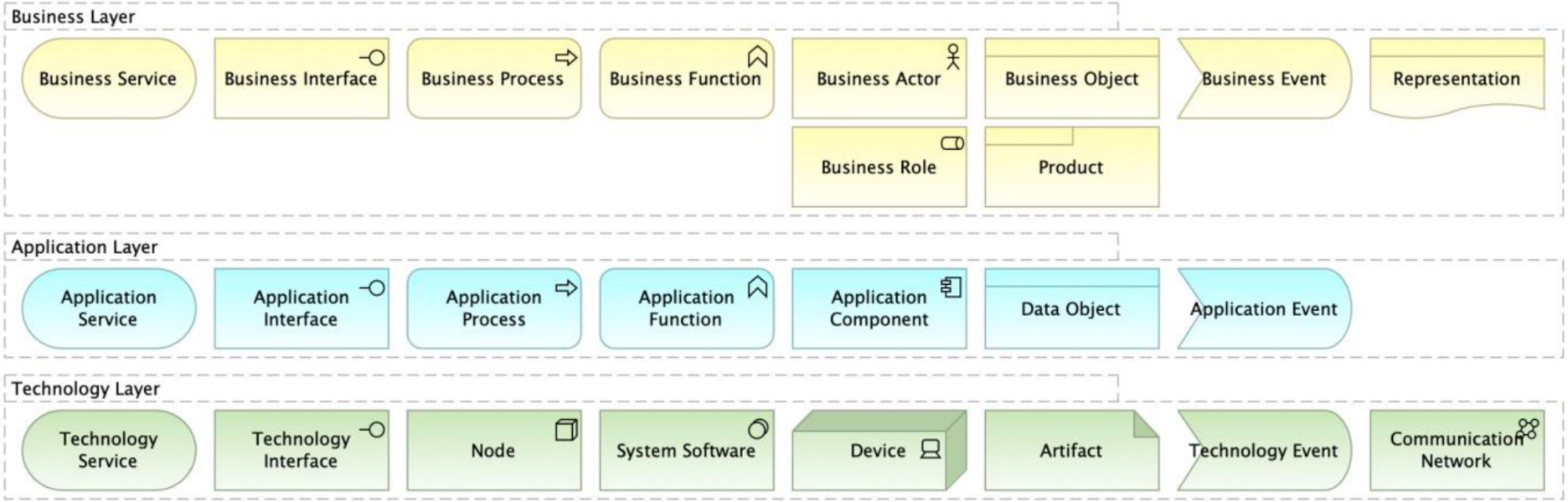
Passive Structure Aspect, represents the objects on which behavior is performed on. Usually are:

- Business Information objects
- Application Data objects
- Physical objects



Note: this is not the full set, others described later.

The BAT Layer Elements



Being able to identify a system by its name is important, but the system's name does not necessarily reflect its purpose. ArchiMate makes a clear distinction between the structural constituent of a system (called Active Structure) and its Behaviour. Active Structure is assigned to its Behaviour

- The default generic notation is to use a *rectangle* (with right angles) for Active Structure elements, and a *rounded rectangle* for Behaviour elements.

Language Based Approach

Active structure element can be regarded as a “subject”, behaviour element as a “predicate” (verb), and passive structure element as an “object”.

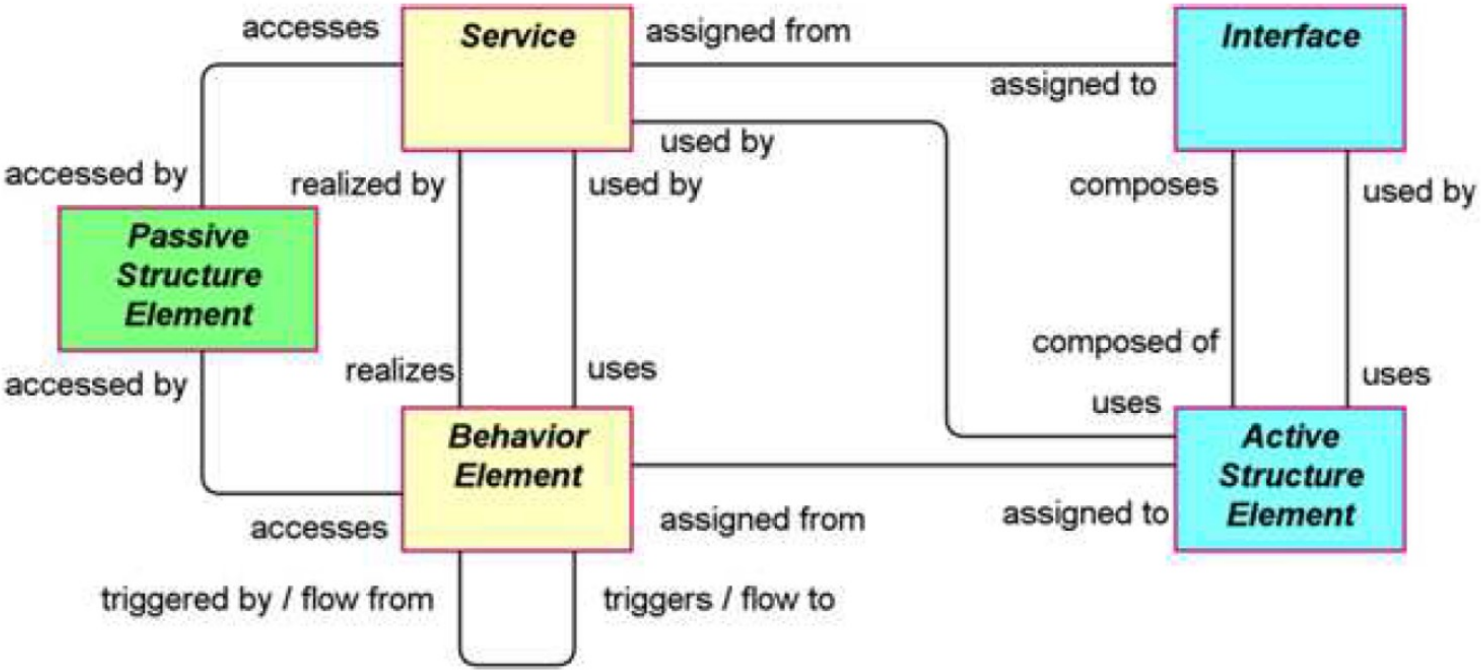
- *Active Structure* elements explain the *who* or *what* of behaviour.
- *Passive* Elements explain what the behaviour acts on

Subject / Actor (Active)	Verb (Behaviour)	Object (Passive)
James	Reads	Book
James	Eats	Breakfast
Business Actor	Business Process	Business Object
Business Role	Business Service	Business Object
Application Component	Application Functions	Data Object
Application Interface	Application Service	Data Object

Identity of an element does not define behavior or the objects it accesses.

Meta Models

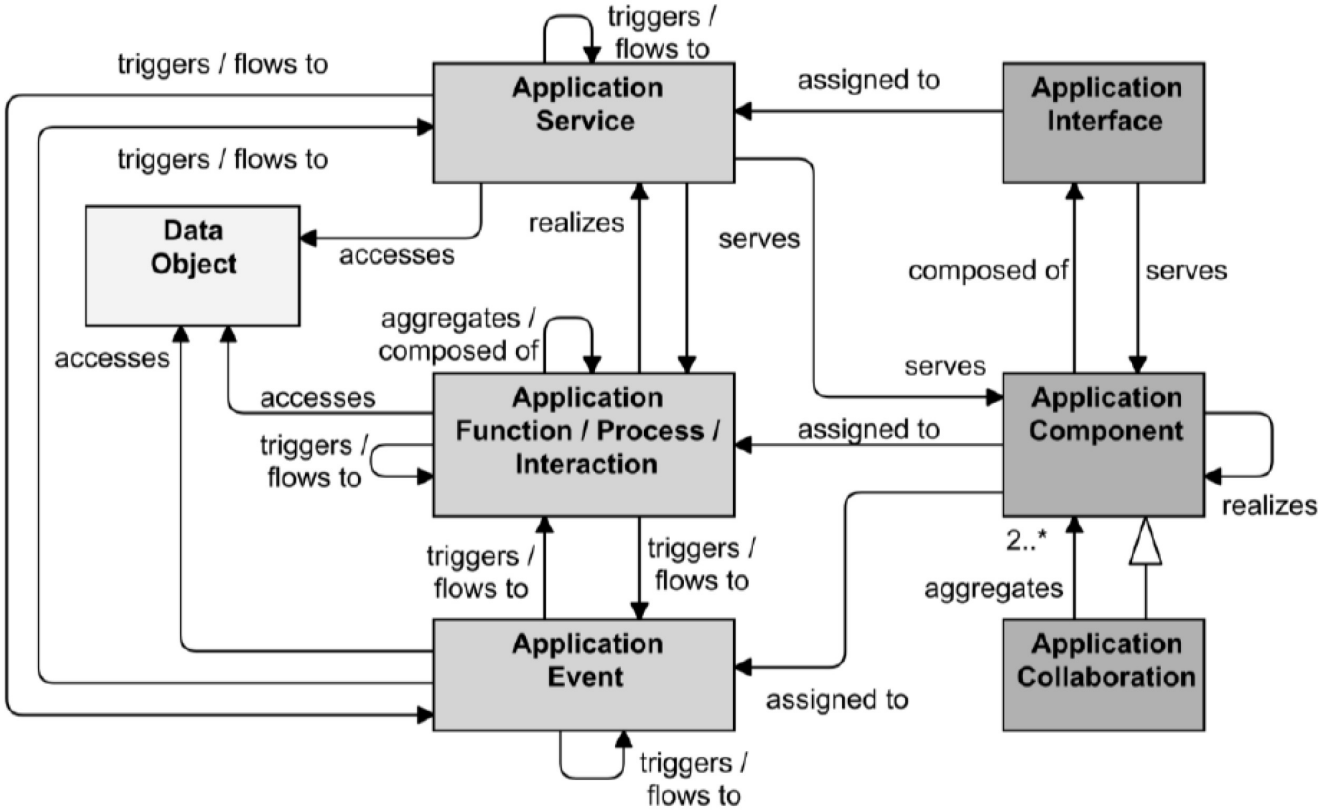
The Core Meta Model



ArchiMate - Core MetaModel

The Application Layer

Passive (light grey)	Behaviour (dark grey)	Active (darker grey)



External and Internal Behaviour

- **Service** is used for modelling the outside-in view on system's behaviour, i.e. the behaviour as it is perceived from the outside of the system.
  - often used to describe what the system must do from the perspective of its users.
  - used for modelling an overview (black box view) of the system which abstracts its internals away.

- **Process** is used for modelling the inside-in view on system's, i.e. the behaviour as it happens inside the system.
  - Process is used for modelling sequences of behaviour, or behaviour that have a beginning and an end.

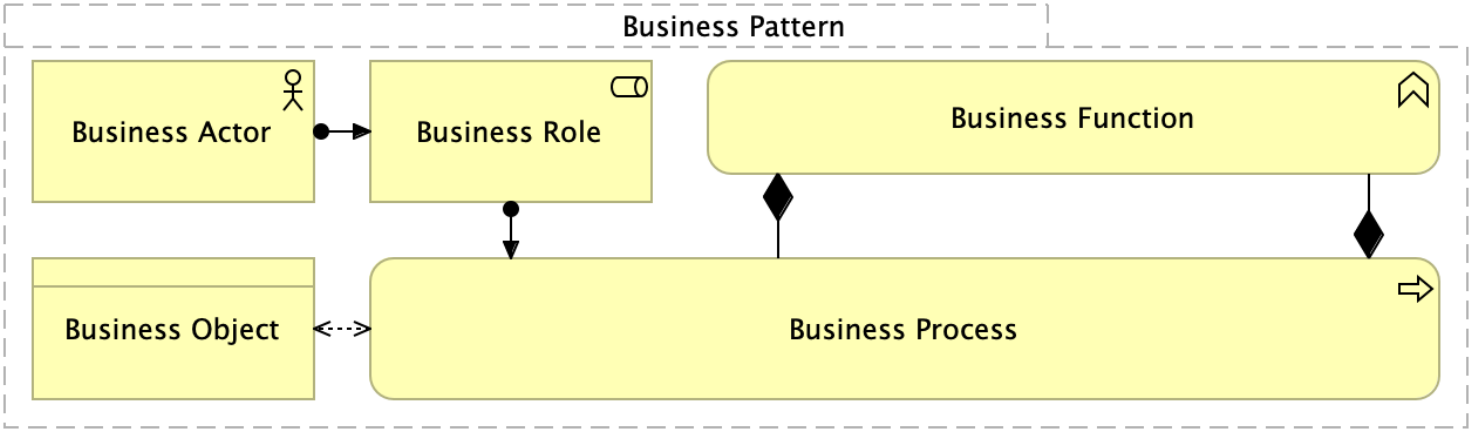
- **Function** is also used for modelling system's internal behaviour. Unlike Process, Function has no beginning nor end. It represents a continuous behaviour or is used to group other behaviours.

- Service can be composed of other Services
- Function can be composed of other Functions, and Processes
- Process can be composed of other Processes, and Functions

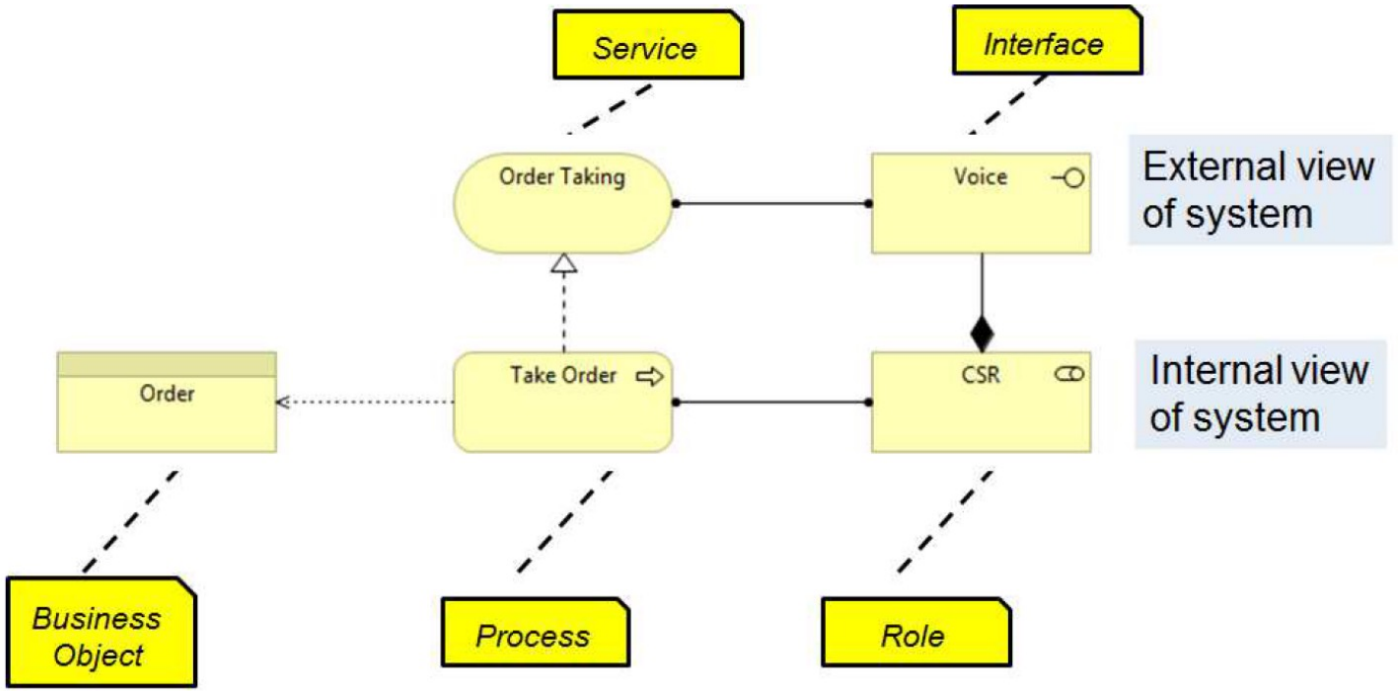
Two views abstractions:
- White Box (internal) = inside-in
- Black Box (external) = outside-in

Patterns by Layer

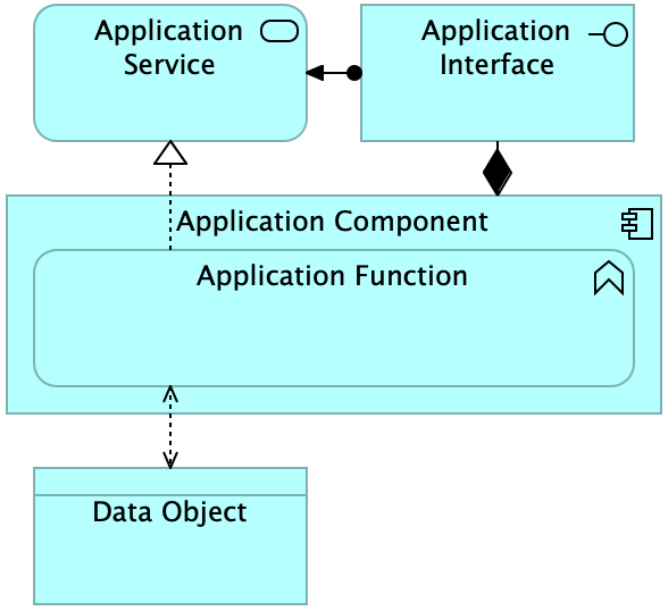
Business Pattern



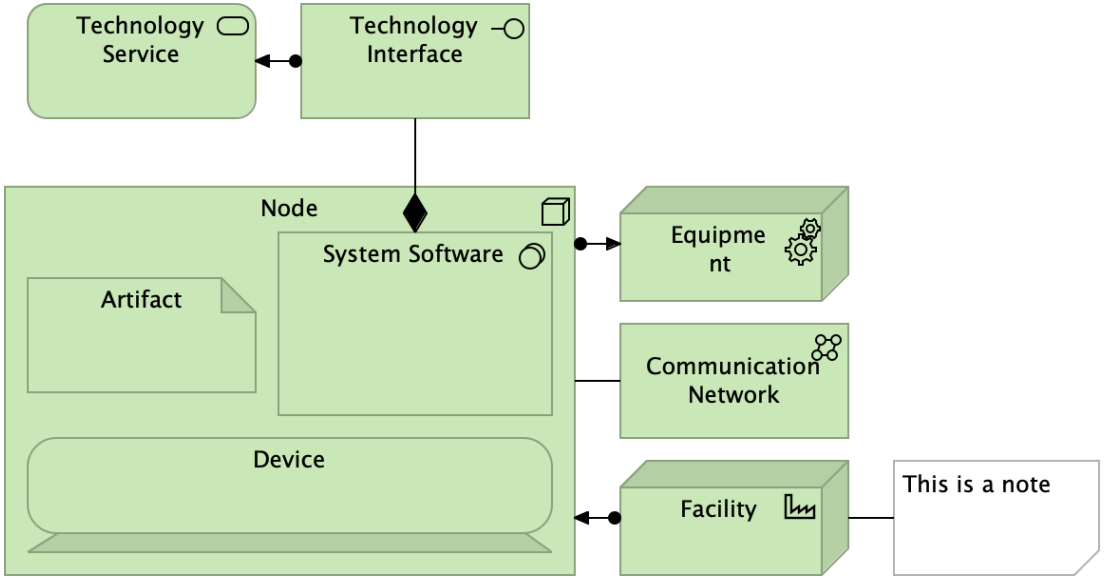
Core Elements



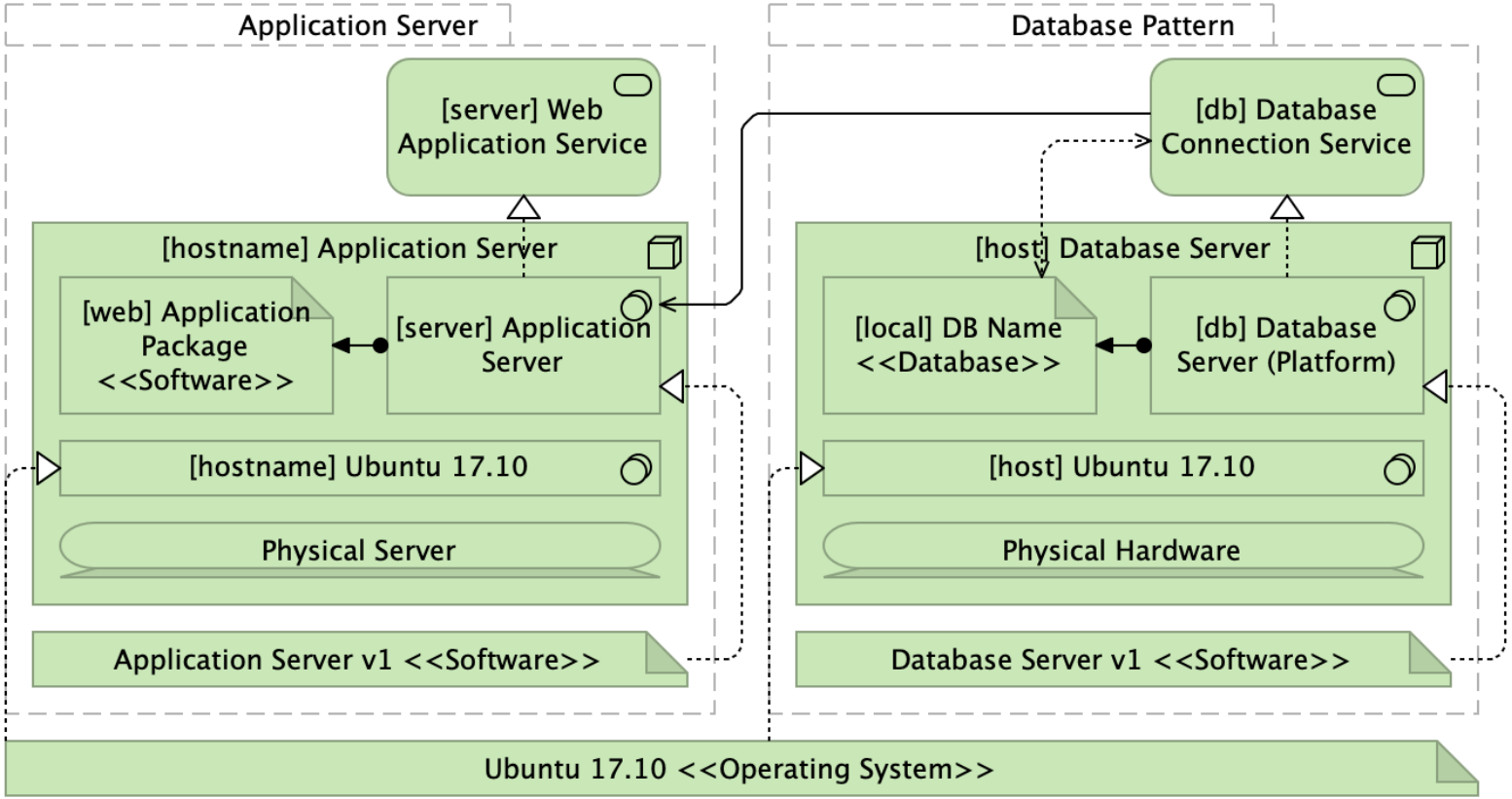
Application Pattern



Technology Pattern

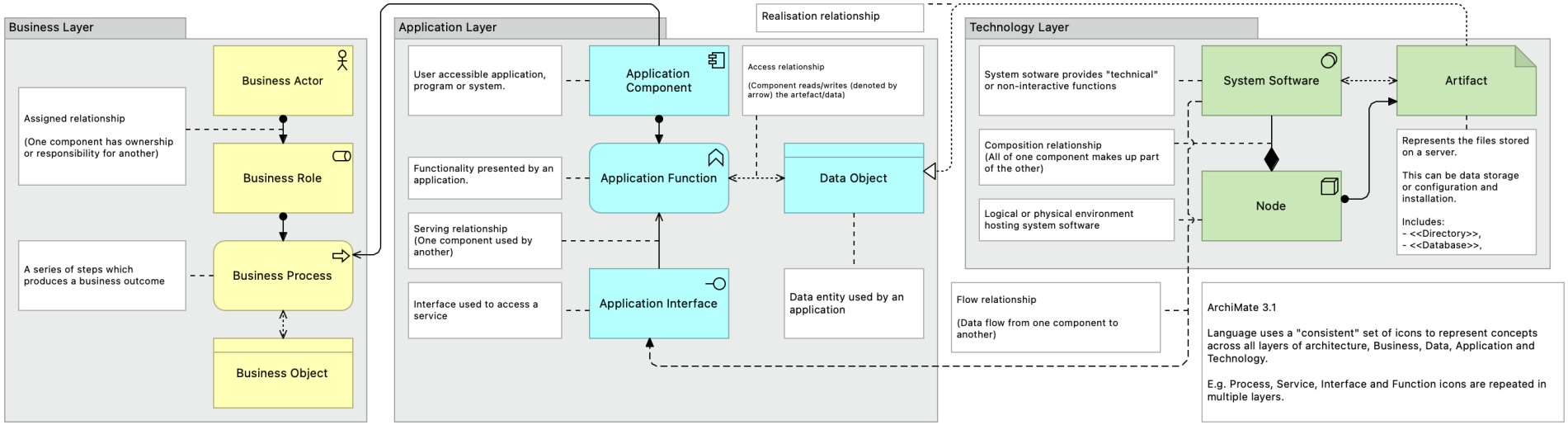


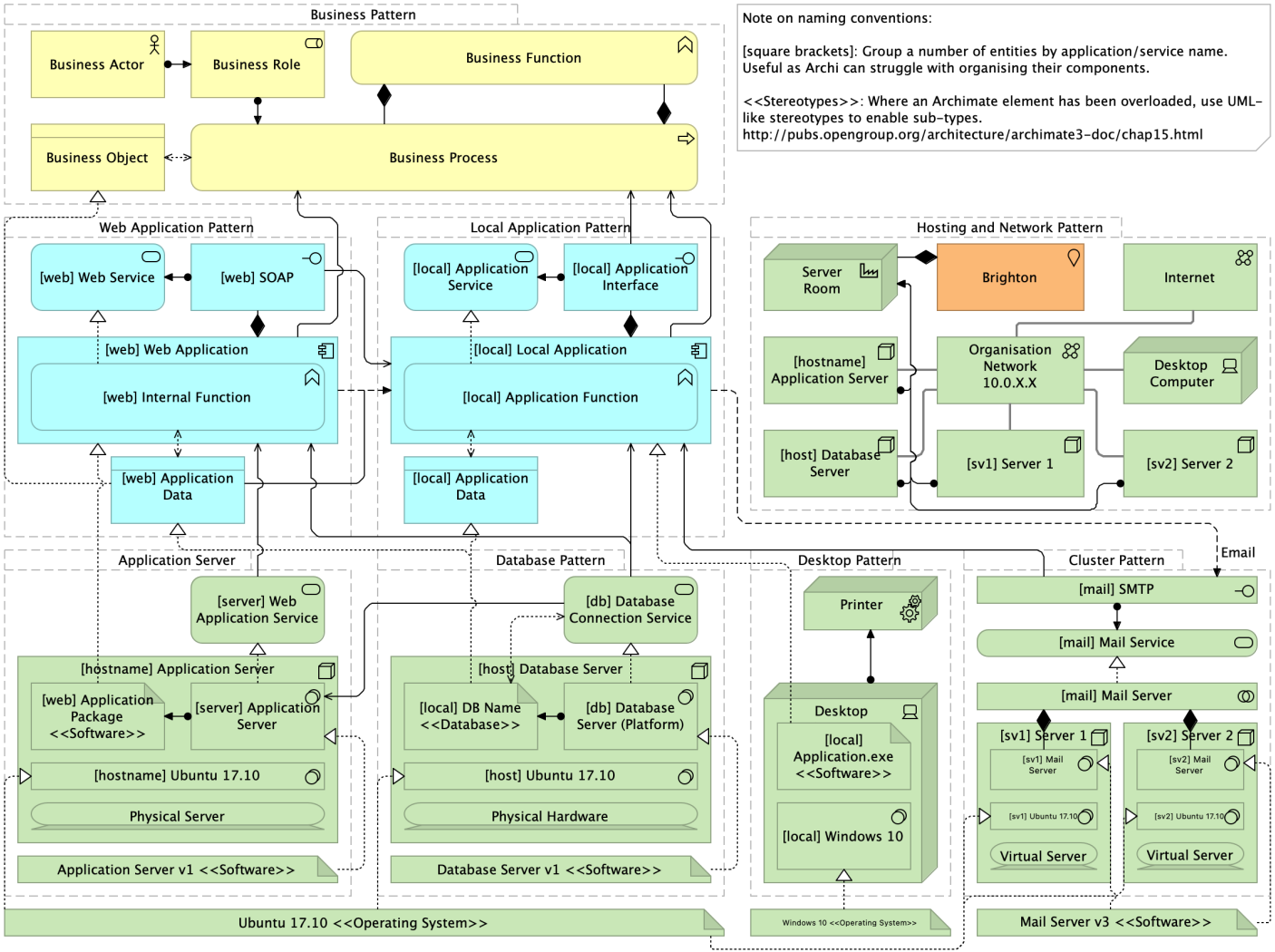
Technology Example



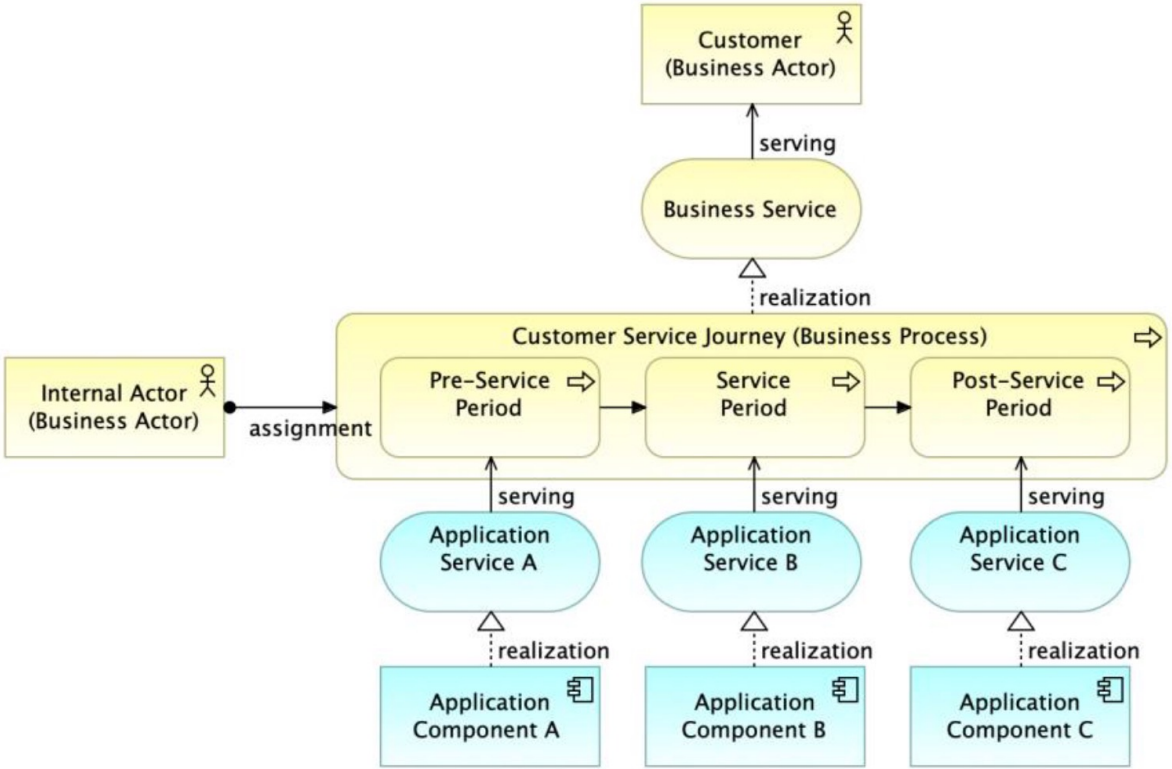
Patterns Across Layers

Across





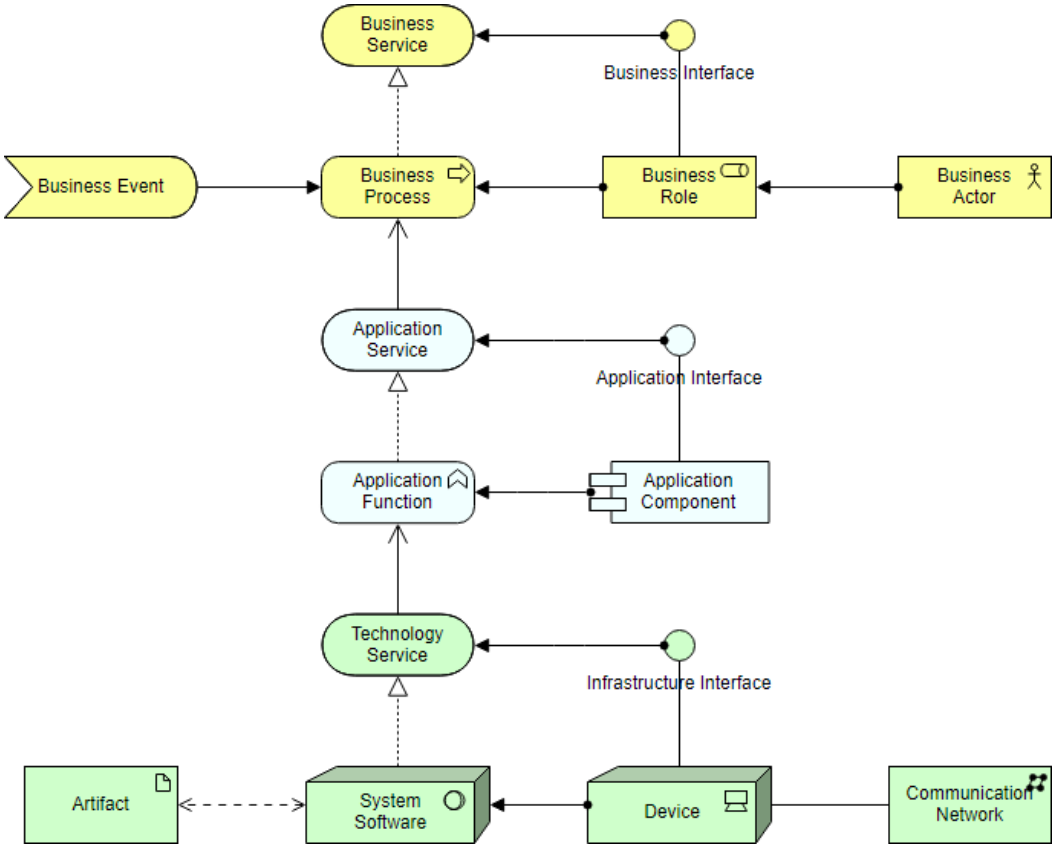
Business To Application



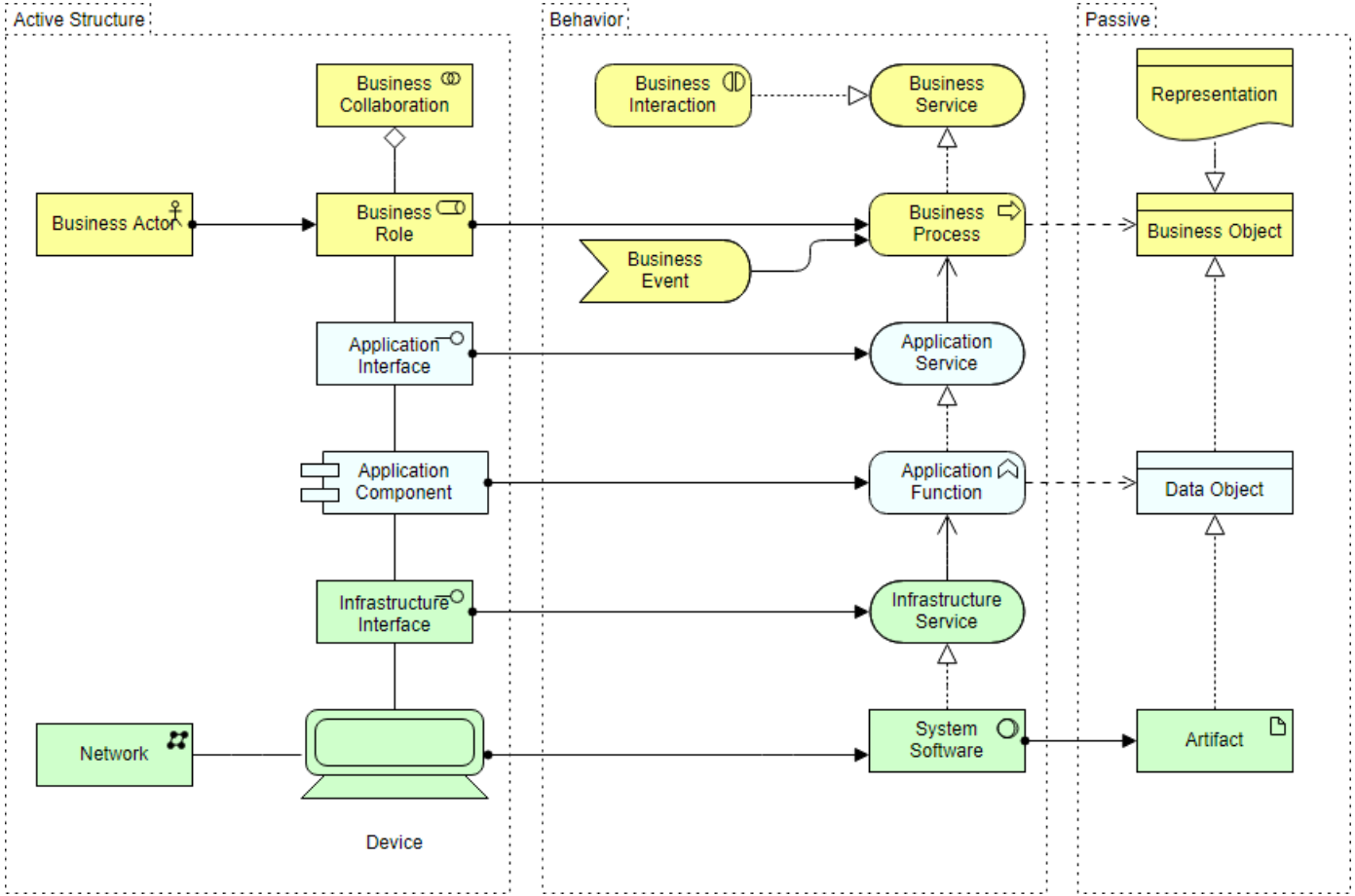
Note the relationships:

- Serving
- Realization
- Assignment
- Triggering
- Composition

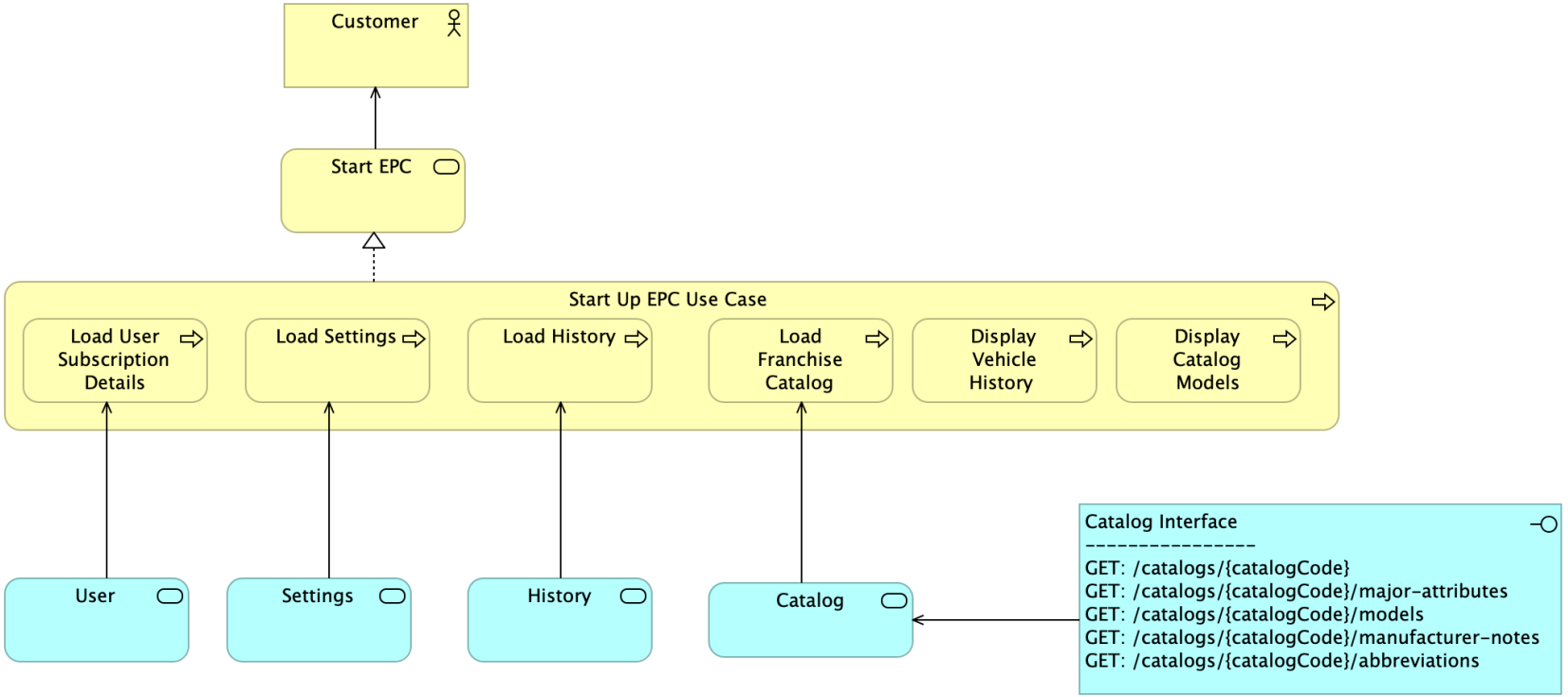
Basic Business Application



Active Structure – Behaviour – Passive

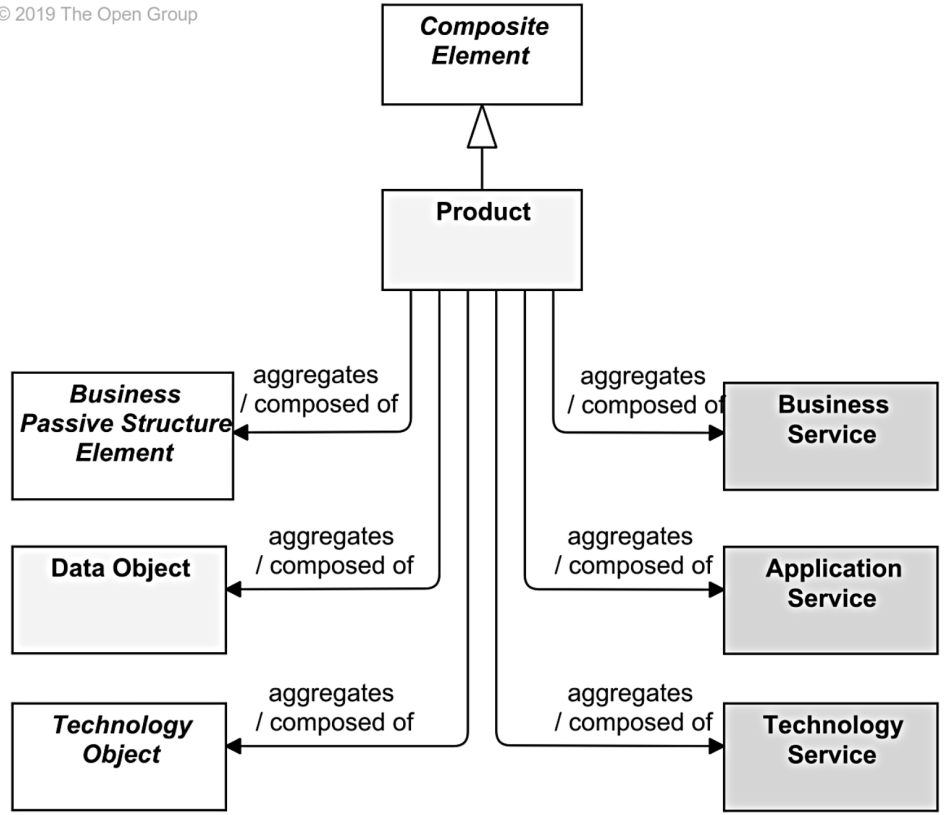


Business Use Case to Application Services – EPC



Business Layer - Product Meta-Model

Page 67



3 Dimensions Summary

1

- **Black Box** – external view from outside
- **White Box** – internal inside view

2

- **Behaviour** – what the system must do
- **Active Structures** – how it does it, the people, applications, and infrastructure

3

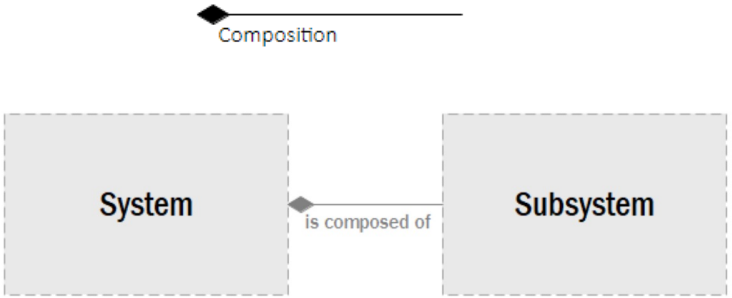
- **Conceptual** – business information
- **Logical** – It structures representing the information
- **Physical** – storage of this information

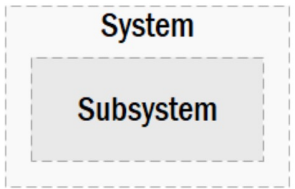
- Logical components of application are the product-independent encapsulations of data or functionality
- Physical components are tangible components or devices.

Relationships

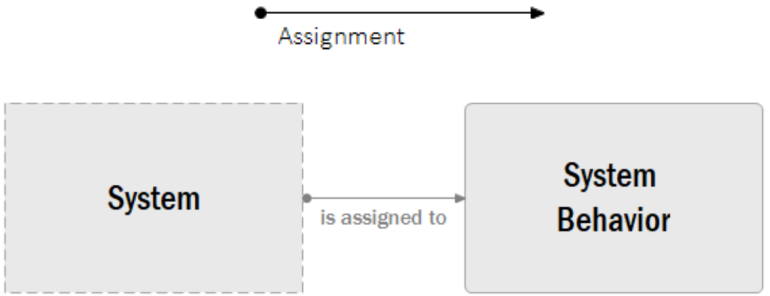
Structural Relationships

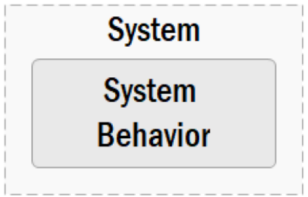
Composition Relationship - decomposing an element into other elements of the same type.
Aggregation allows an element to belong to one or more aggregations





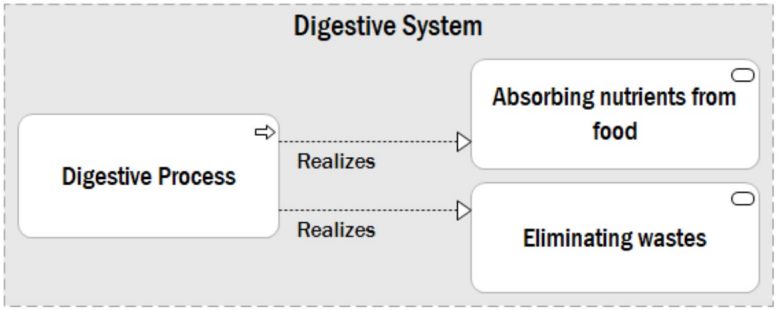
A system's name is important, but does not necessarily reflect its purpose. There is a clear distinction between the structural constituent of a system (called Active Structure) and its Behaviour.
The Active Structure is **Assigned** to its Behaviour

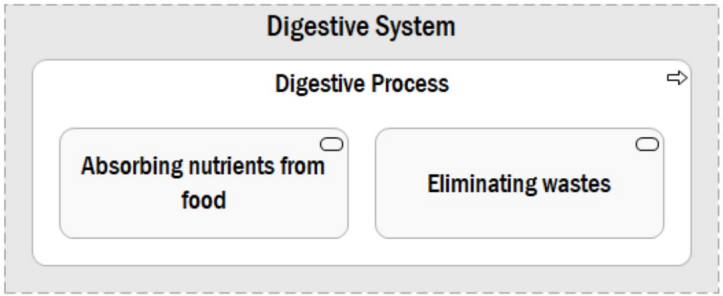




Realization - structural

Service being an external behaviour, it is seen as an abstraction of Functions or Processes. Such abstraction is modelled through the **Realization** relationship:



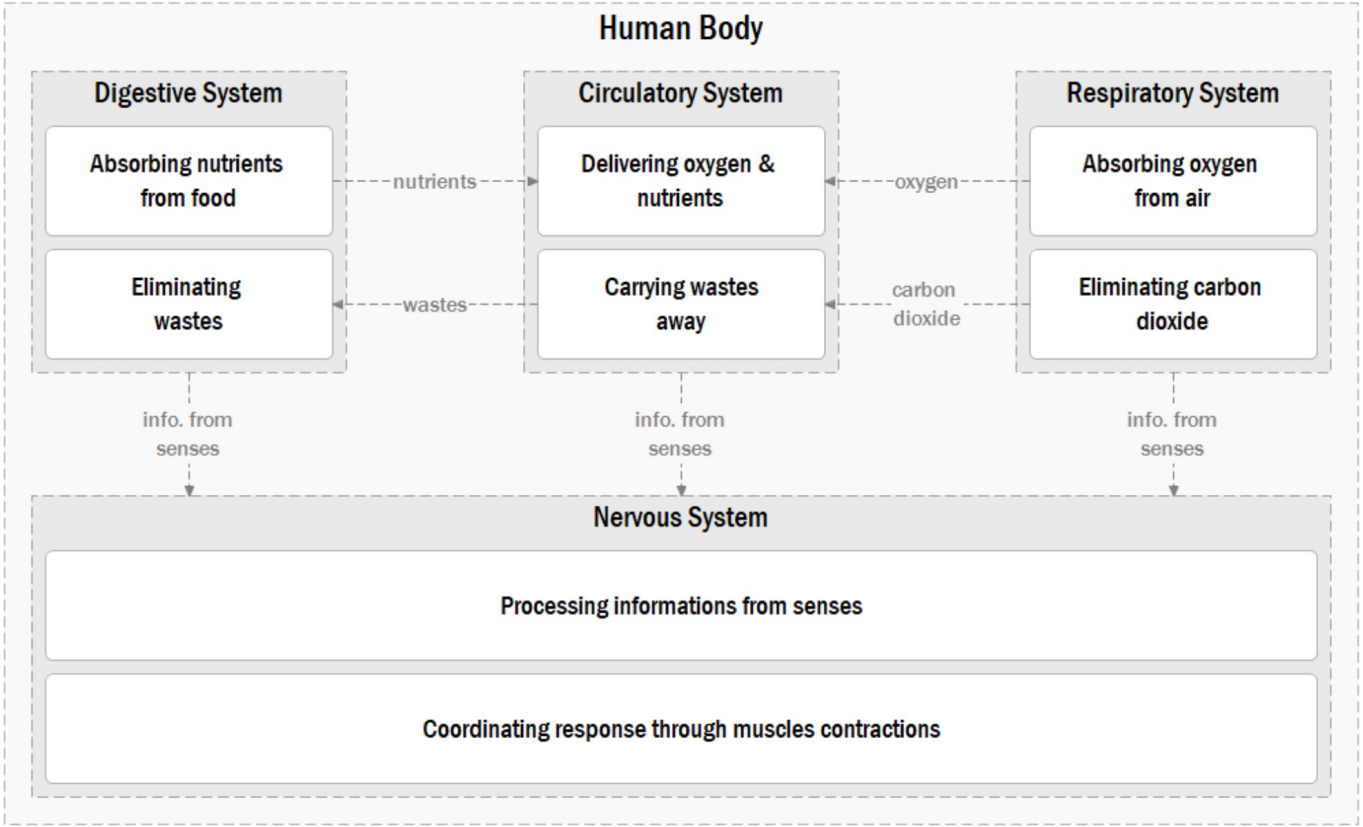


- "Digestive System" is **assigned** to the "Digestive Process”
- "Absorbing nutrients from food" and "Eliminating wastes" are both **realized** by the "Digestive Process”
- Realization allows a Black Box element to be described as White Box.

Flows – dynamic relationship

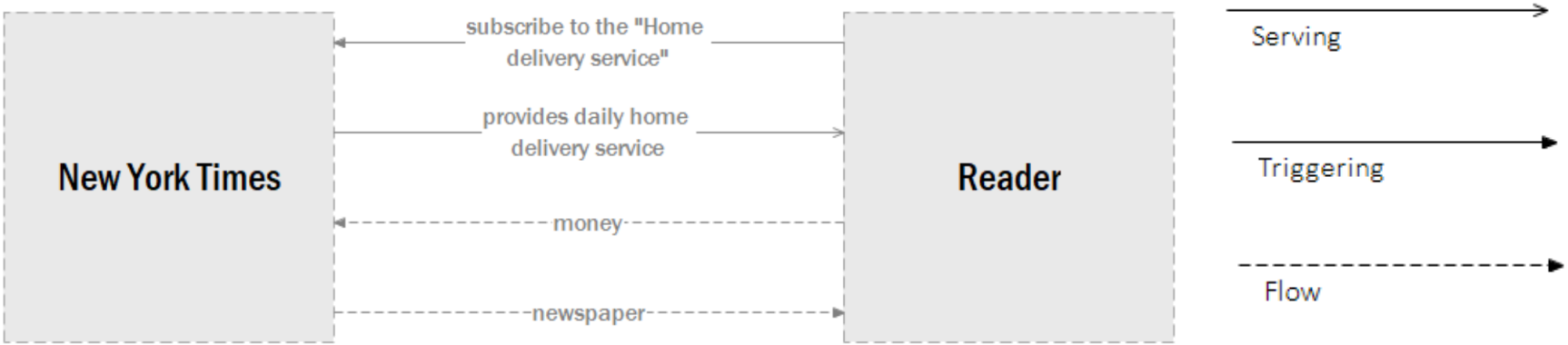
In system dynamics, systems can contain stocks. A stock is anything that exists within the system, performs no behaviour by itself, and is accessed or acted upon by the system (like information, money, goods…).

Flows can be modelled between systems themselves (i.e. Active Structure), between their Behaviours or a mix of both.



Serving – a dependency

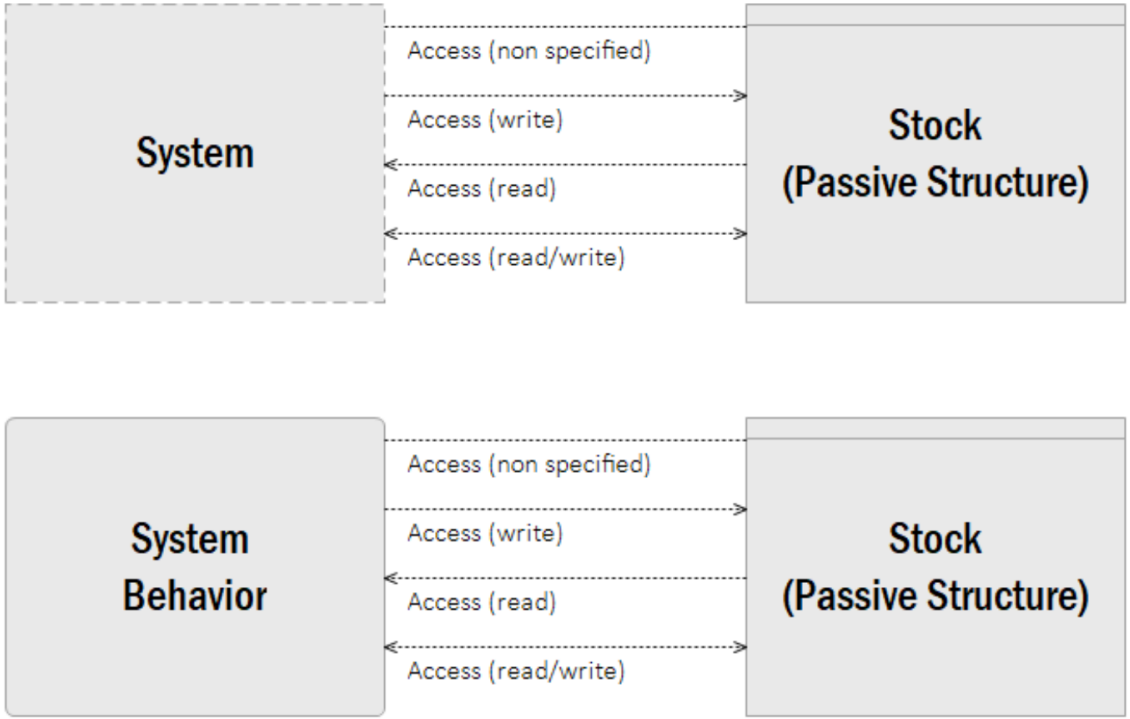
Systems exchanging stocks (i.e. Passive Structure) and affecting each other lead to dependency. The Serving relationship denotes that some systems provide its functionality (i.e. Behavior) to other systems. As with Flows and Triggering, **Serving** can be modelled between systems themselves (i.e. Active Structure), between their Behaviour or a mix of both.



A reader may subscribe to a "Home delivery service" provided by the New York Times and in return, expect the newspaper to be delivered. For this to happen, the reader must pay (i.e. send money) for the service.

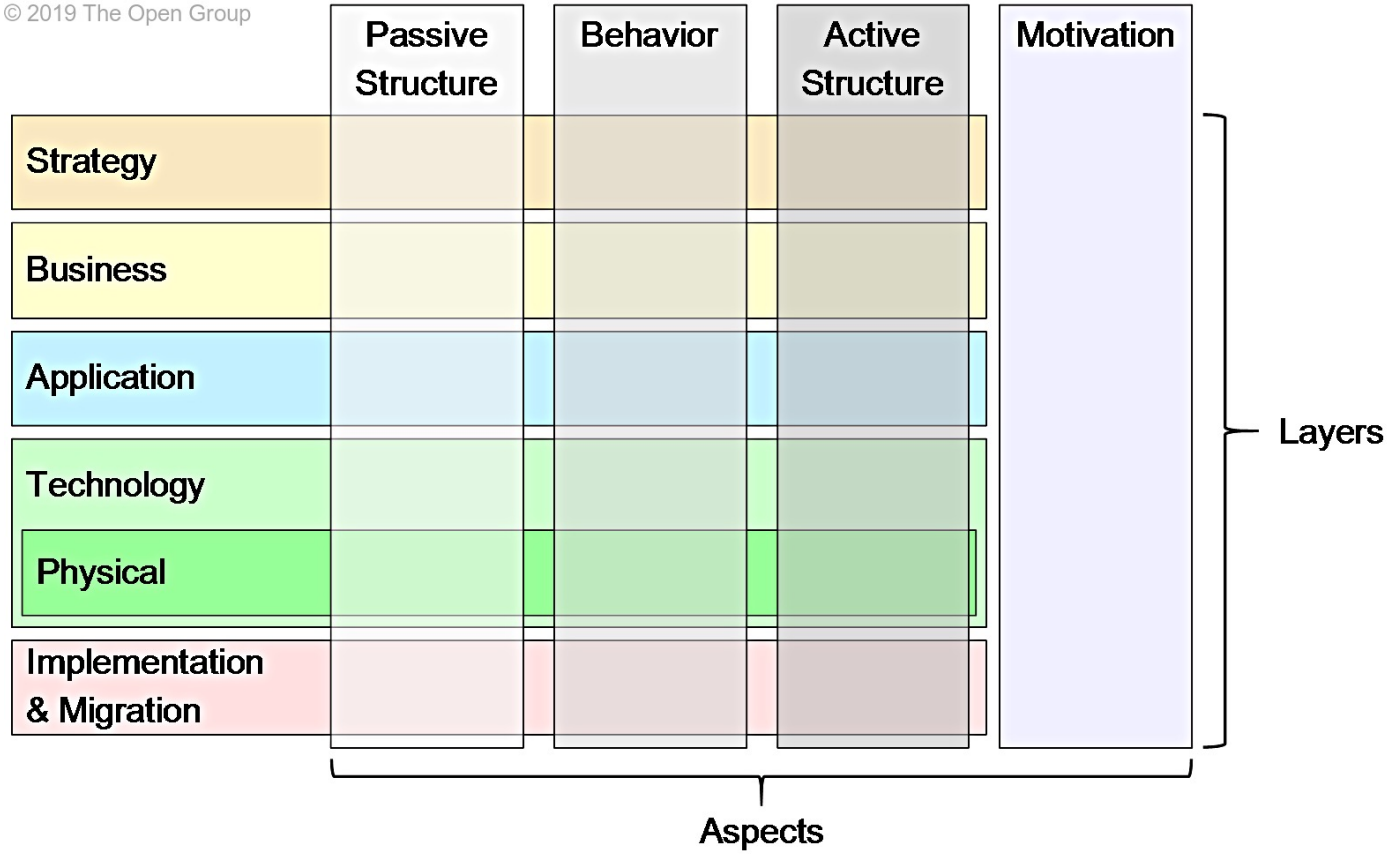
Access – a dependency

Passive Structures are **accessed** by Active Structures or Behaviours. They are tricky because it is always defined from the Active Structure or Behaviour to the Passive Structure, but it visually depicts the type of access



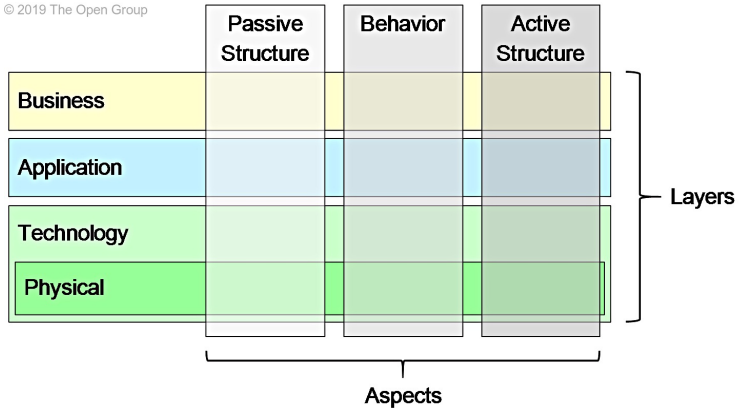
More Layers Aspects

Full Model – Strategy, Motivation, Implementation & Migration

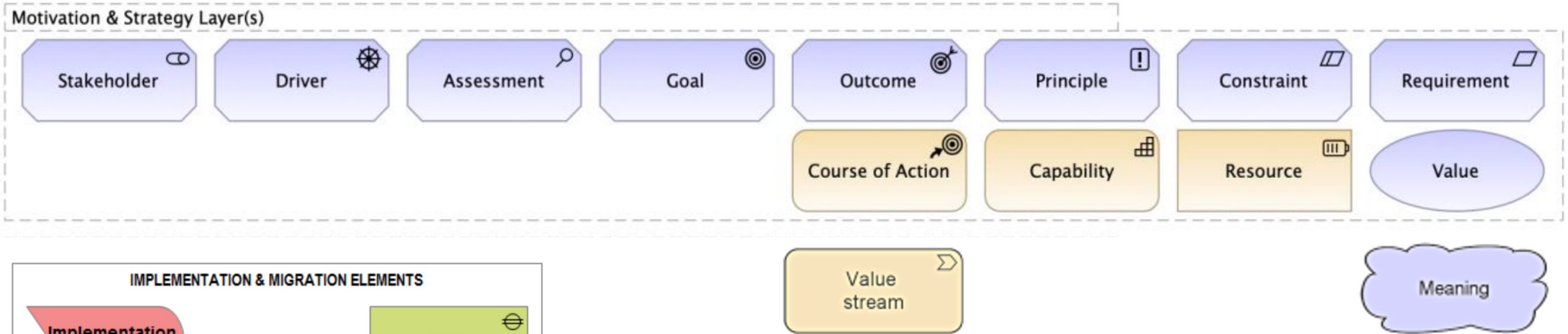


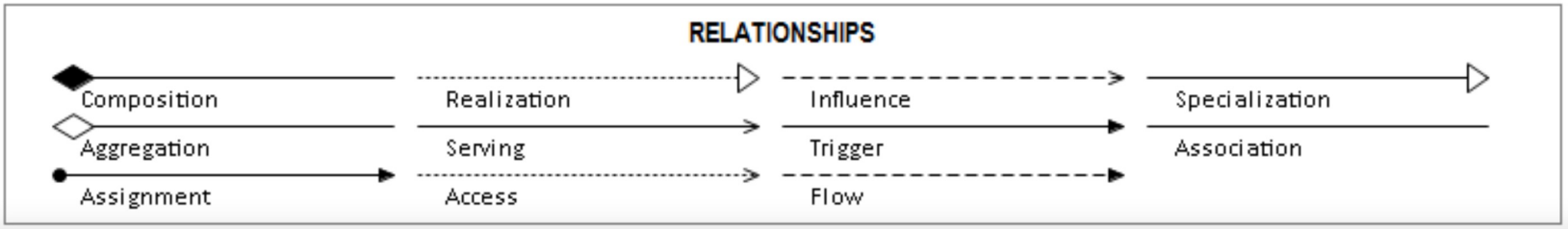
More Domains

- *Motivation* supports the modelling of the reasons that guide the design or change of the architecture.
- *Strategy* supports the modelling of the strategic direction of an enterprise, how it wants to create value for its stakeholders, and the capabilities it needs for that.
- *Core* supports the modelling of the solution itself through three layers (Business, Application and Technology).
- *Implementation and migration* supports the modelling of implementation programs or projects, and migration planning.



Other Layer Elements and Full Set of Relationships

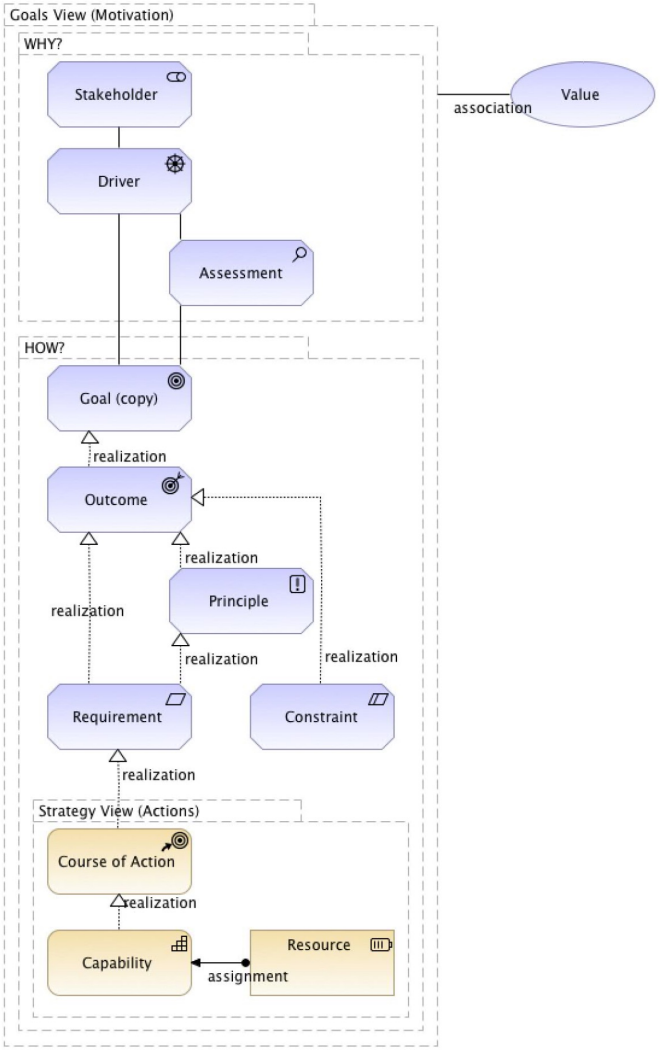




Strategy Meta Model

Links the Why and the How.

- What is the added value for a certain development target

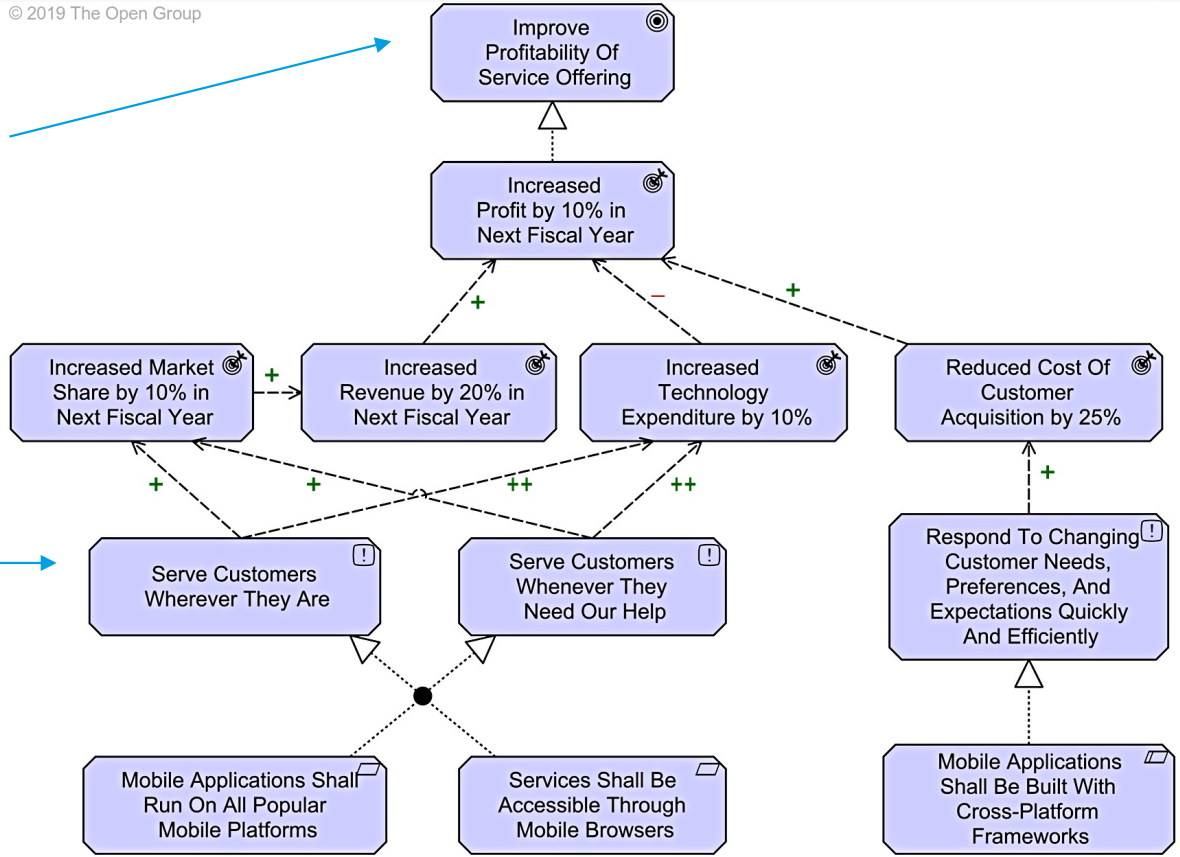


Motivation Example

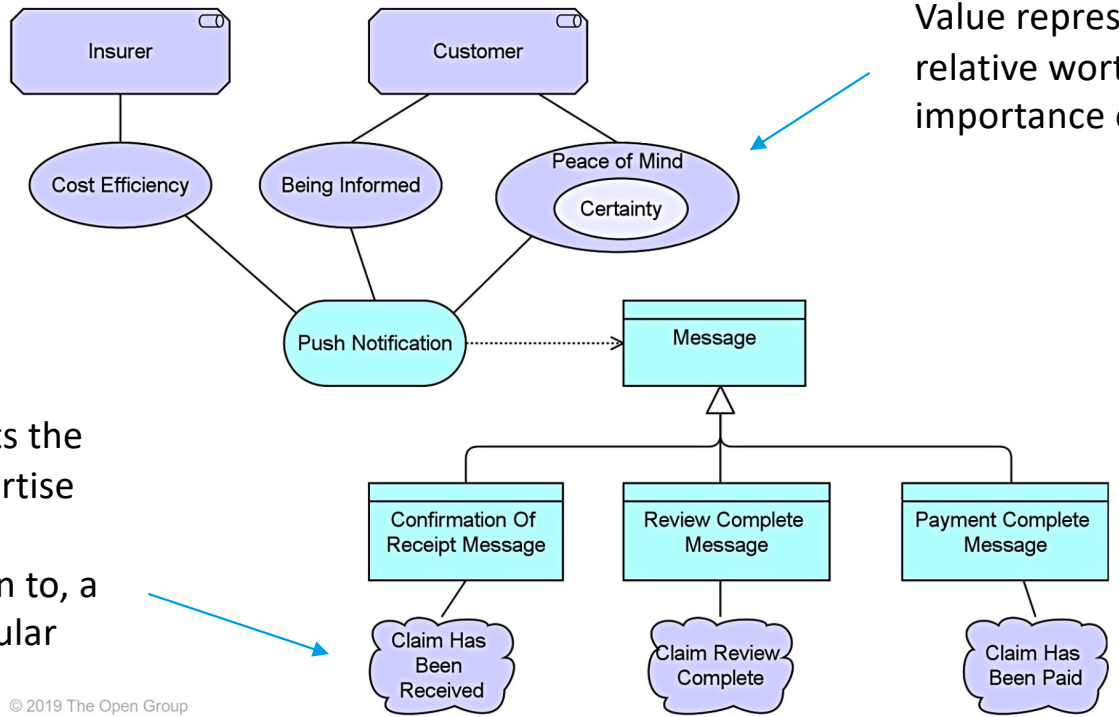
A goal represents a high-level statement of intent, direction, or desired end state for an organization and its stakeholders.

An outcome represents an end result.

A principle represents a statement of intent defining a general property that applies to any system in a certain context in the architecture



Meaning and Value



Value represents the relative worth, utility, or importance of a concept.

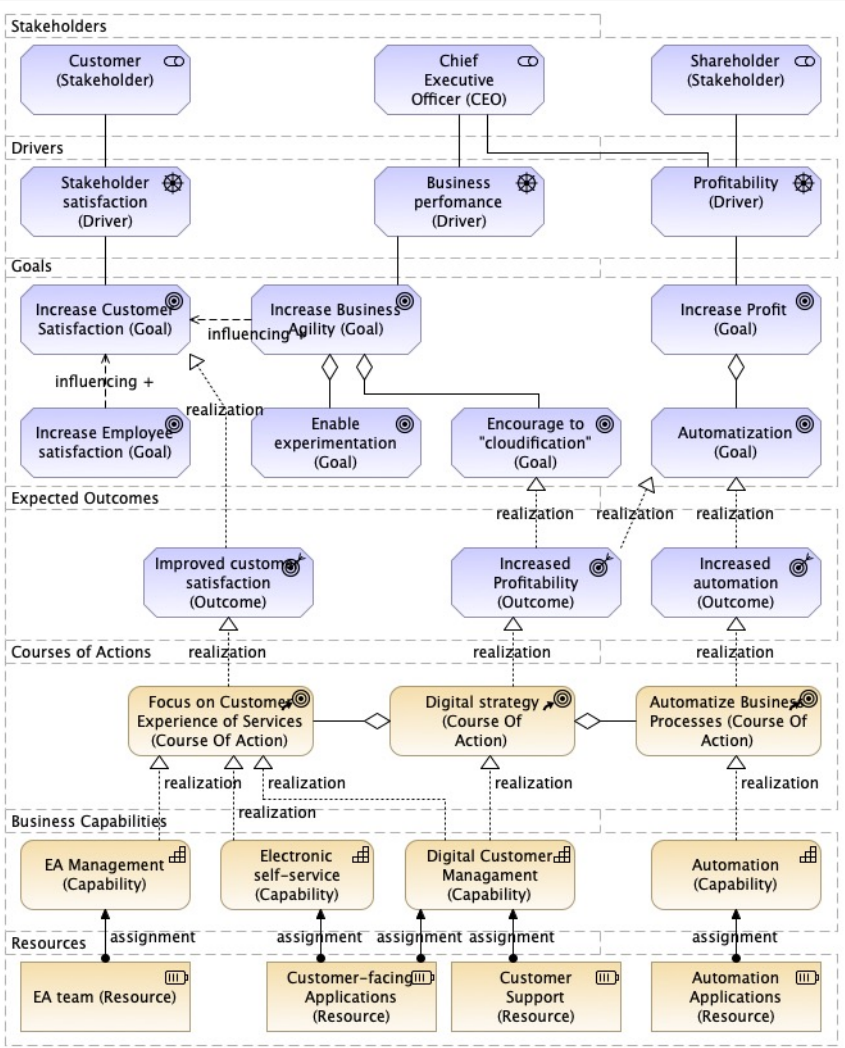
Meaning represents the knowledge or expertise present in, or the interpretation given to, a concept in a particular context

Capability Planning

The Strategy to Capability View can be used for Capability-Based Planning (CBP) purposes, together with other ArchiMate concepts such as “Driver” and ”Goal”.

To support Strategy Planning (and -Execution) purposes.

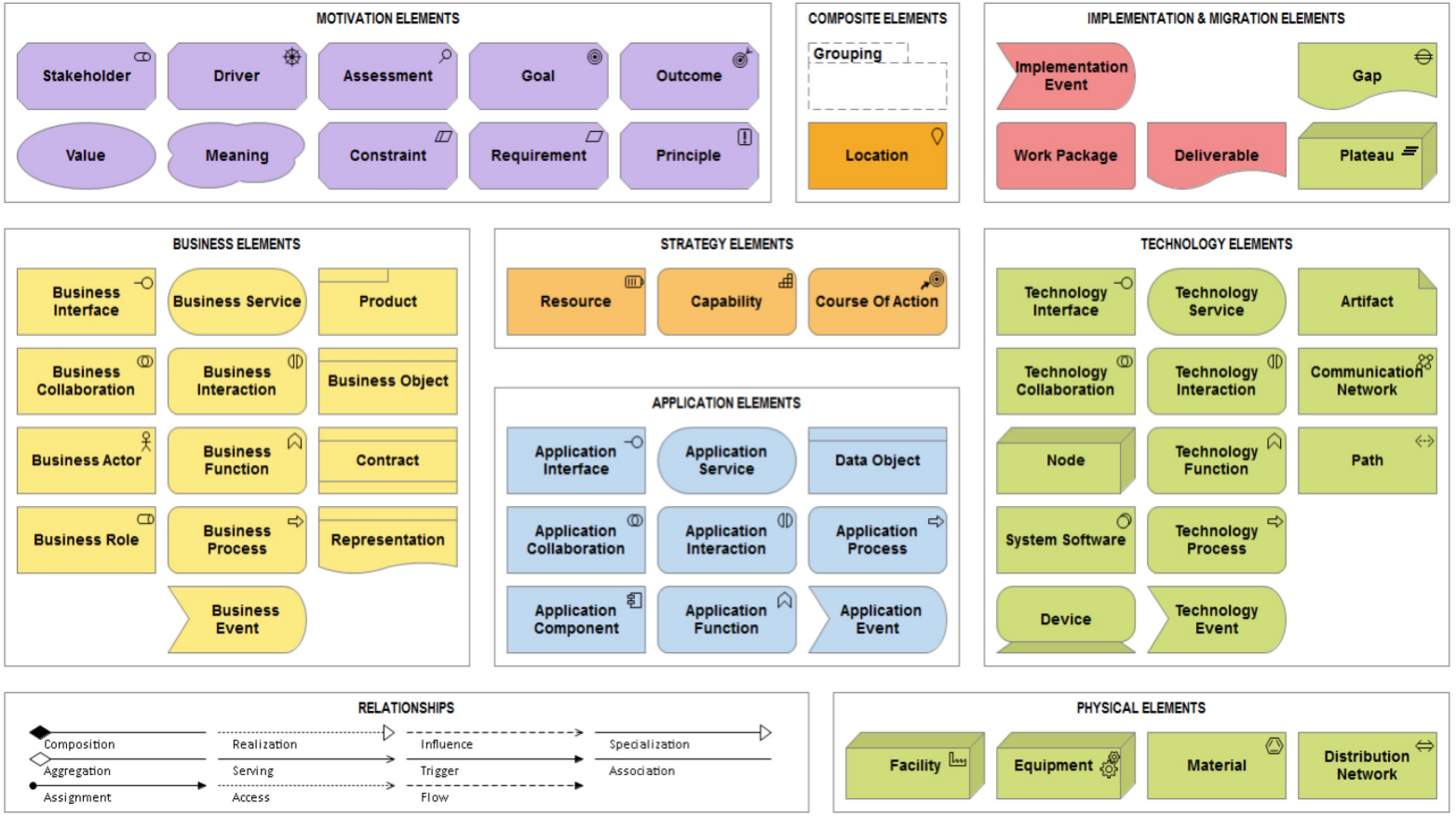
These kind of views can be used in Strategy-to-Capability phase of the operating model (e.g. in the “Strategy-to-Portfolio” value stream of IT4IT value chain)



Archi
archimate modelling

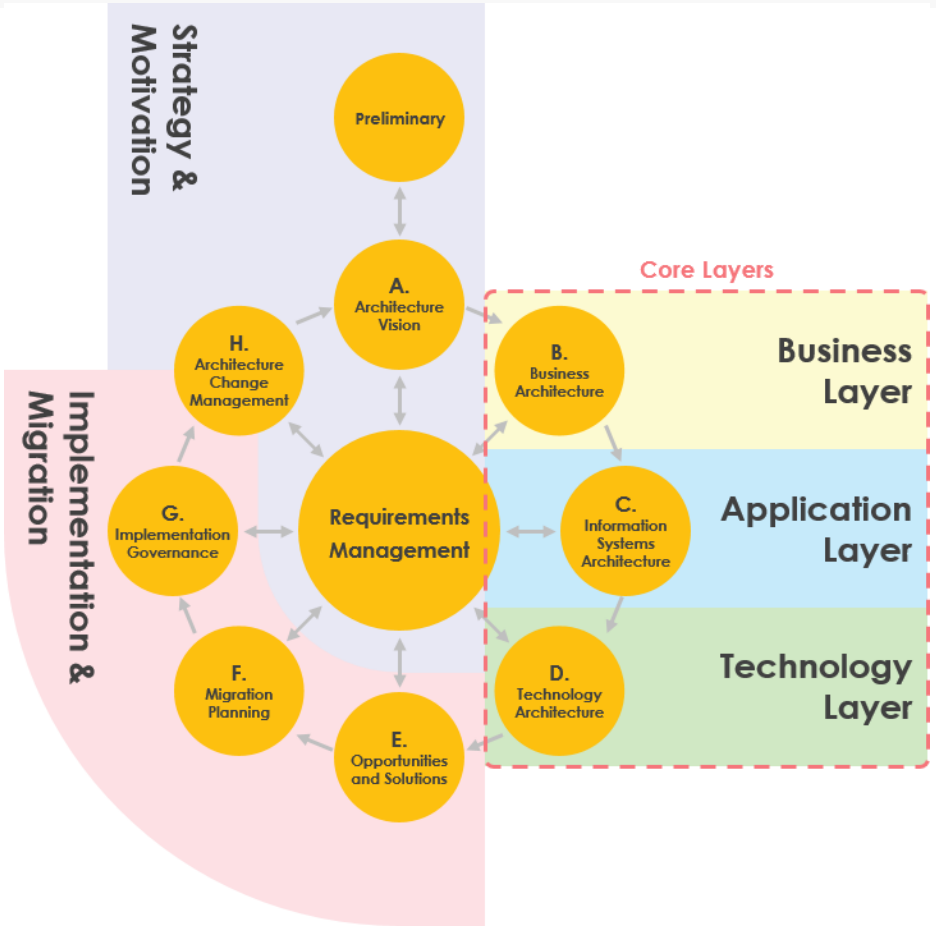
ArchiMate®

THE Open GROUP

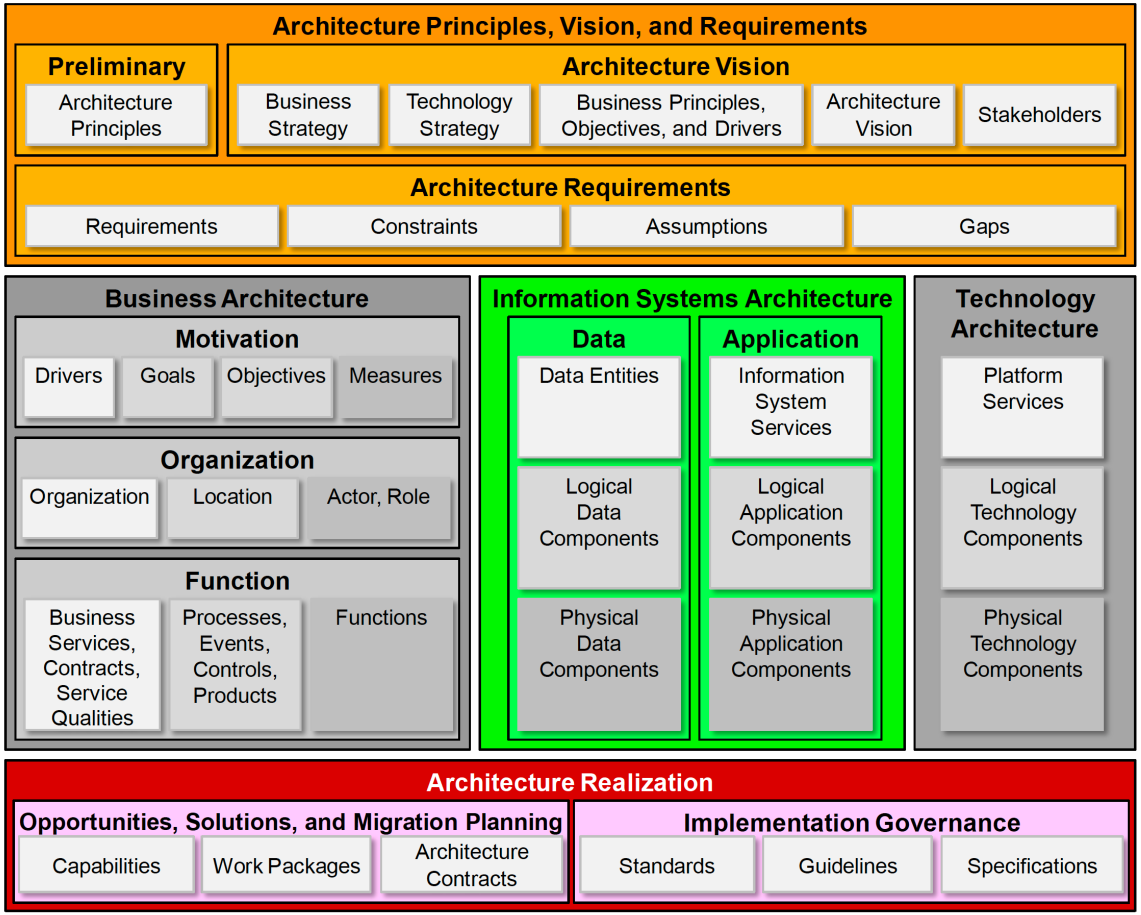


Layers and TOGAF ADM

Archimate covers the full cycle of the ADM



TOGAF Content Meta-Model



ViewPoints

ViewPoints - Purpose

The very first step is to clarify who will observe and read (part of) your architecture description:

- ***Informing*** viewpoints are used to achieve understanding or obtain commitment. Such viewpoints often target a broad audience and should be simplified and straight to the point. The main goal is to elicit feedback to make sure that the communication is effective.
- ***Deciding*** viewpoints support the process of decision-making and often target managers. Gap analysis and scenarios comparison often fall in this category. The main goal is to obtain a decision or a choice between several options.
- ***Designing*** viewpoints support the design process from initial sketch to detailed design. They typically target subject matter experts. The main goal is usually to define or refine a target.

ViewPoints – Content (levels of Detail)

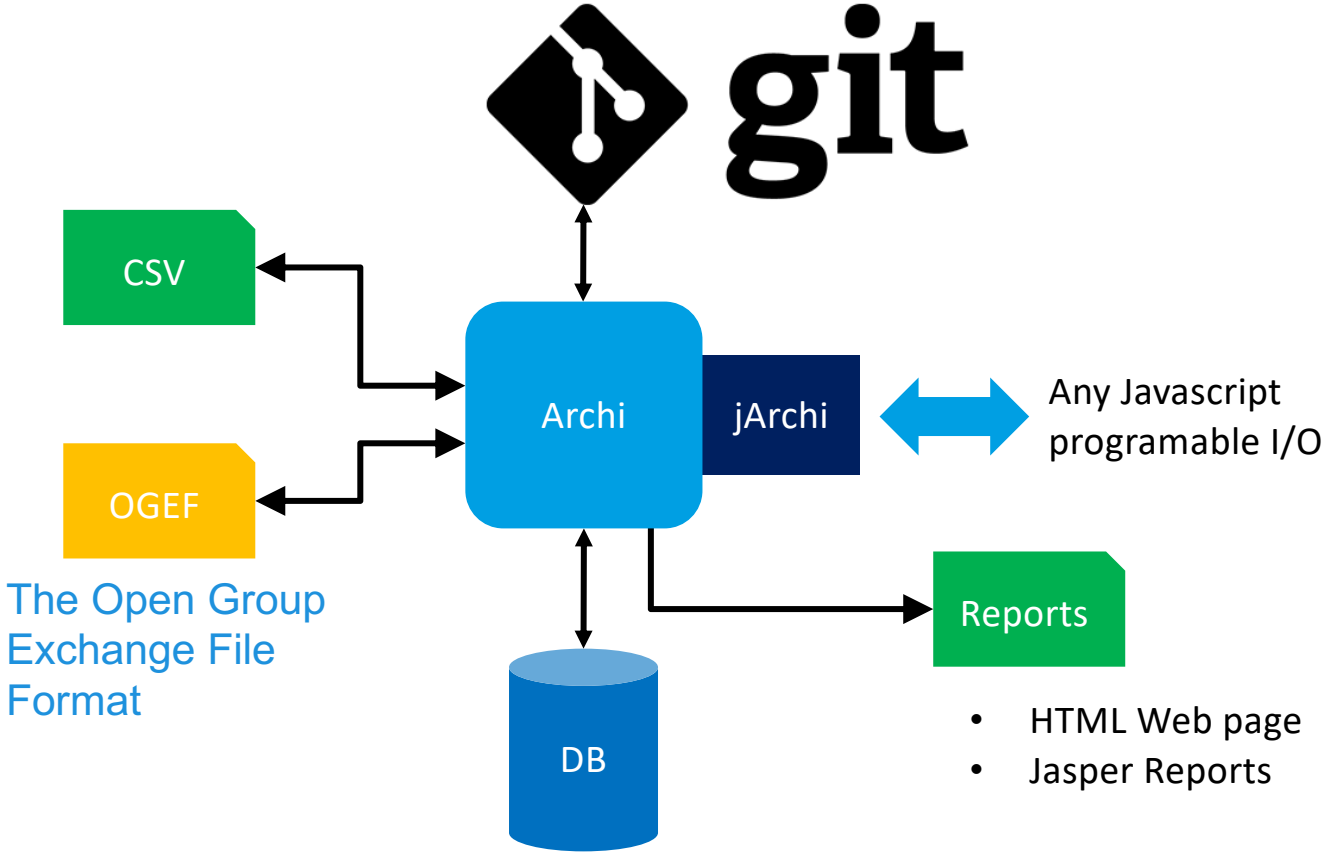
- ***Overview****:* this groups viewpoints providing a helicopter view on a subject which usually mixes multiples domains (such as Strategy, Motivation or Core) and/or layers (Business, Application and Technology). Such viewpoints usually targets decision-makers or enterprise architects.
- ***Coherence****:* the usual goal of coherence viewpoints is to focus on one specific topic, but seen through multiple complementary angles. The emphasis is often on collaboration between people, processes or tools.
- ***Details****:* as its name implies, detailed viewpoints focus on one specific topic and zoom in only one of its aspects. This level of detail usually target subject matter experts or software engineers.

We can now explain the Architecture Journey

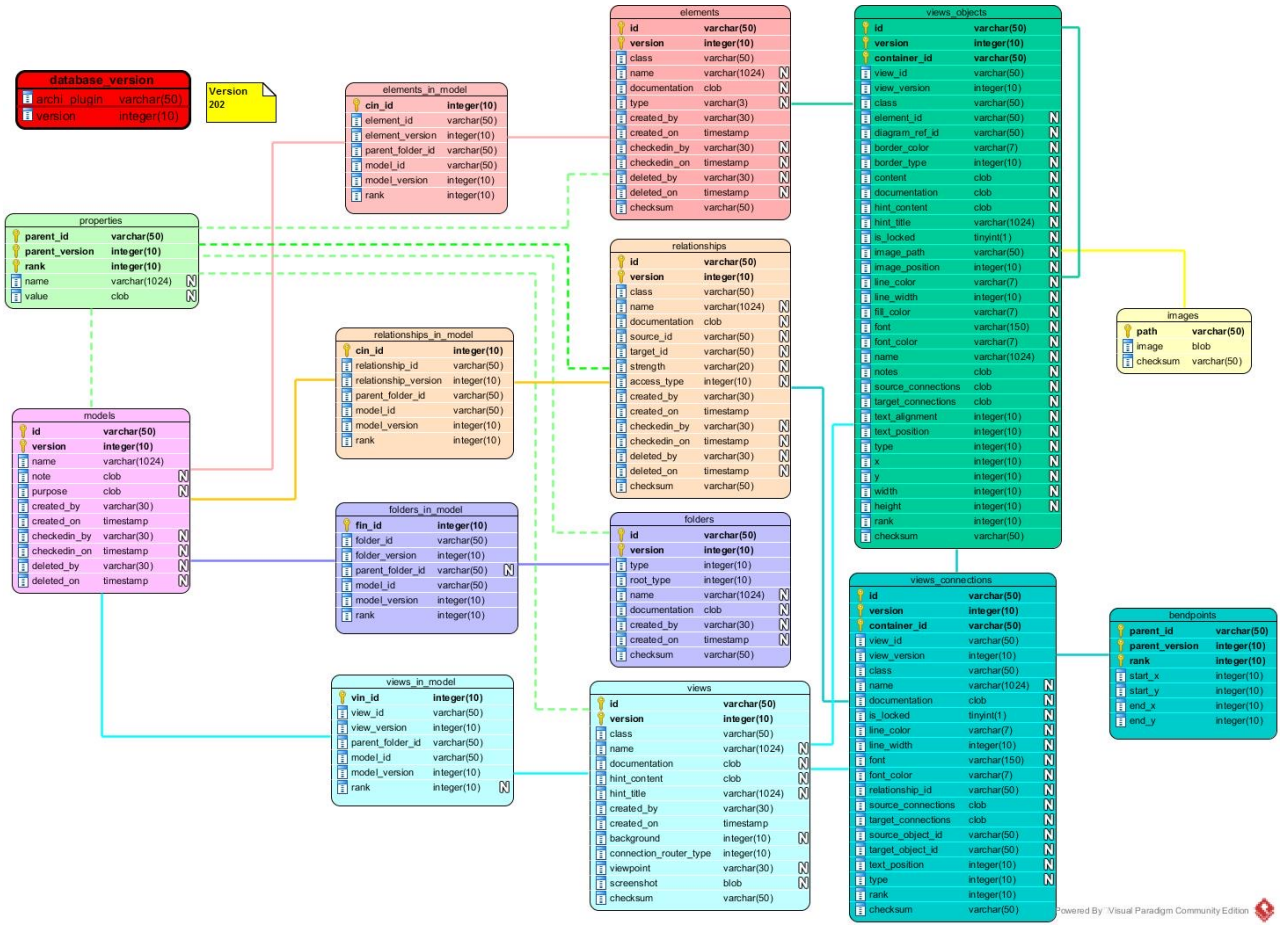
1. Present the organization purpose and motivation, its place into the broader enterprise (organization's **motivation viewpoint**, targets CEO)
2. Present the course of action, strategic goals and outcomes of your organization and how they rely to its purpose and broader motivation (**strategic vision viewpoint**, targets CxO)
3. Present the capabilities impacted by the new strategy (**capability map viewpoint**, targets CxO and top management of impacted capabilities)
4. Show the key business processes affected for each of the capabilities previously identified. Visually, each business process element encapsulate the main applications used to support it, this non standard nesting stands for a reverse serving relationship (**business process impact analysis viewpoint**, targets Business Analysts and Product Owner)
5. For each business object appearing in the process descriptions, provide a more detailed description with an emphasis on personal data (**information model viewpoint, targets** Chief Data Owner and CISO)
6. For each application previously identified, do a gap analysis focusing on added, changed or decommissioned components and flows (**application gap analysis viewpoint**, targets application teams, Product Owners and Lead Developers)
7. For each new application or component, provide a description of the underlying generic technology architecture that will be common to all environments (**logical technology diagram viewpoint**, targets Lead Developers and IT Operations Team)

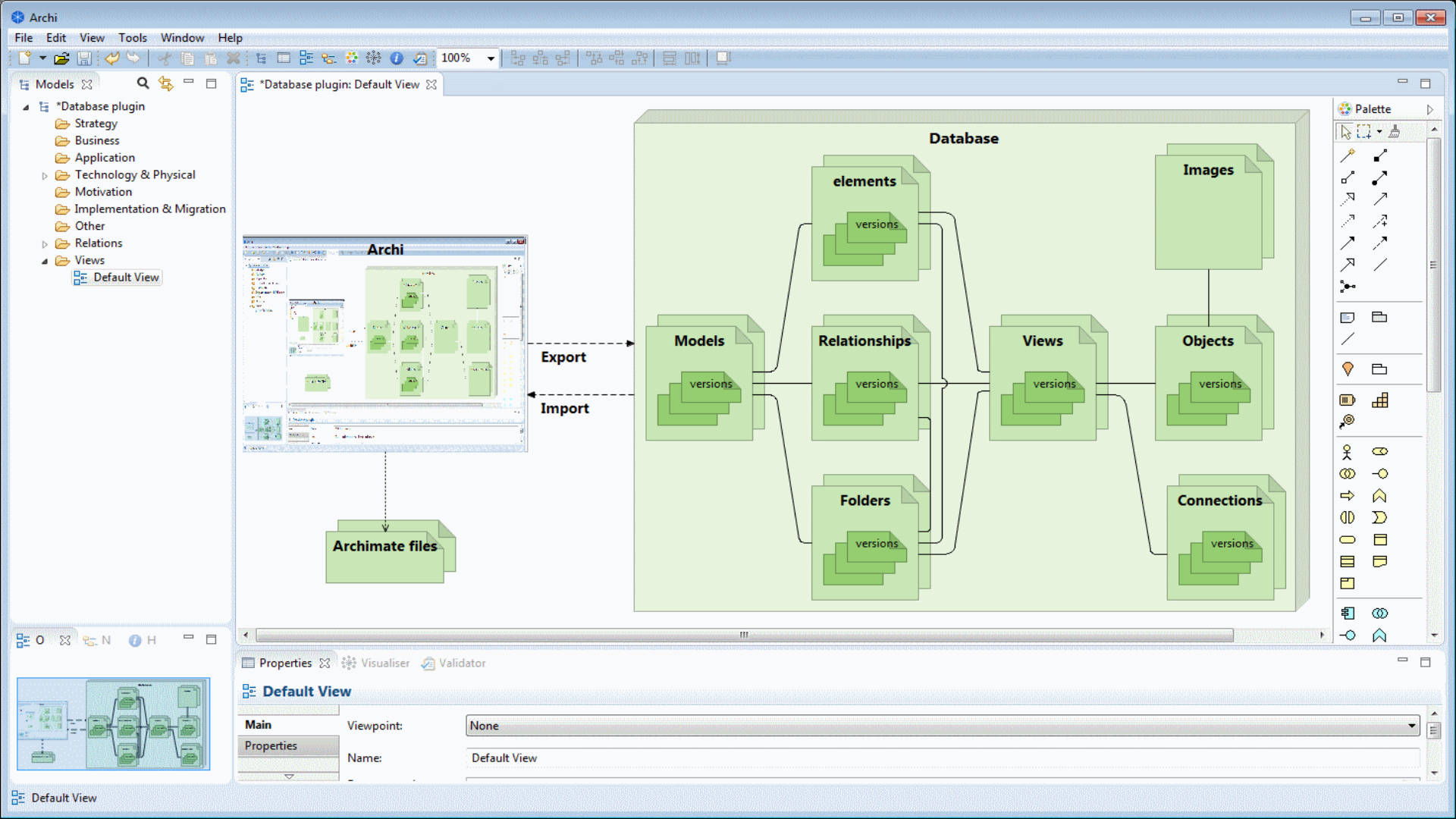
Import/Export

Input and Output



DB Schema



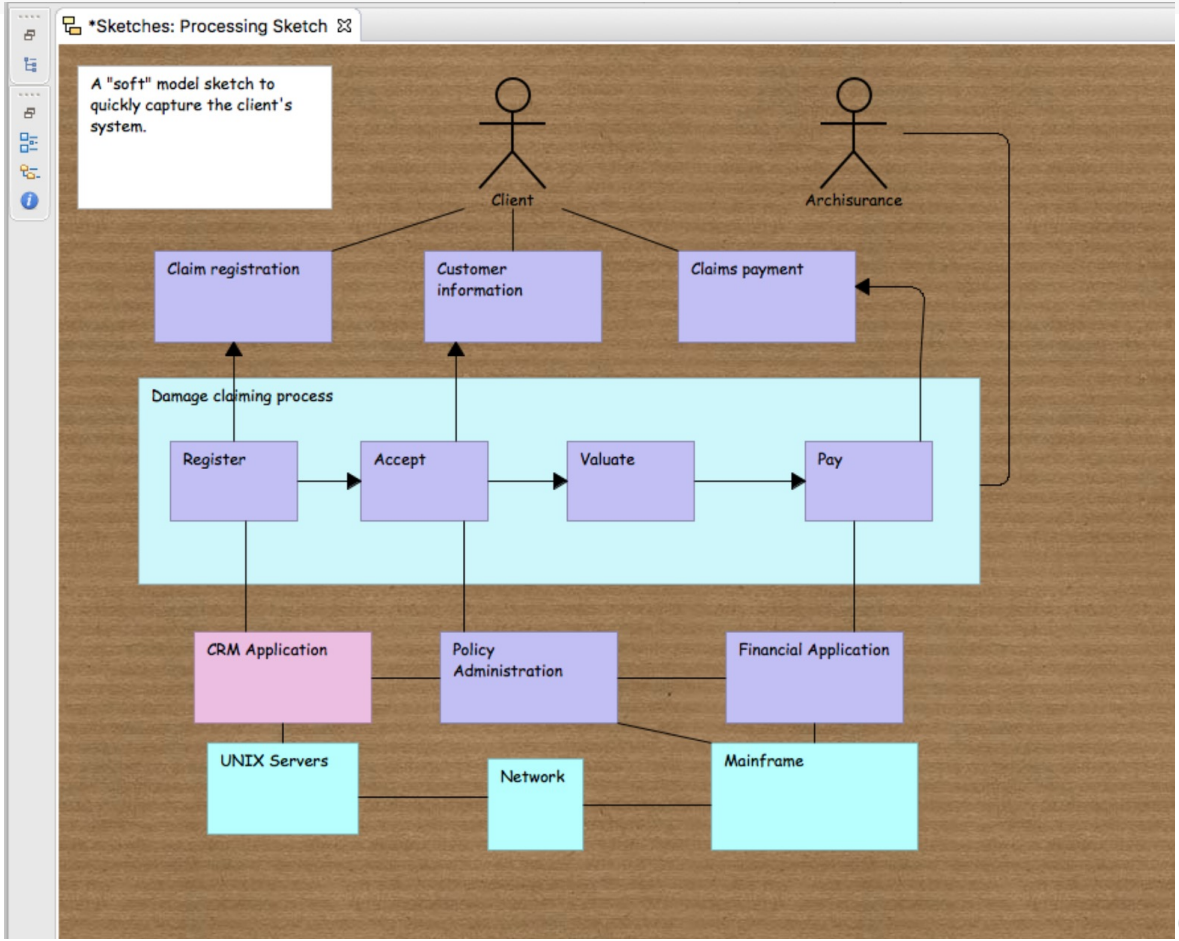

Archi
File
Edit
View
Tools
Window
Help
100%
Models
*Database plugin
Strategy
Business
Application
Technology & Physical
Motivation
Implementation & Migration
Other
Relations
Views
Default View
*Database plugin: Default View
Database
elements
versions
Images
Models
versions
Relationships
versions
Views
versions
Objects
versions
Folders
versions
Connections
versions
Archi
Export
Import
Archimate files
Palette
Properties
Visualiser
Validator
Default View
Main
Properties
Viewpoint:
None
Name:
Default View
Default View

Other Stuff

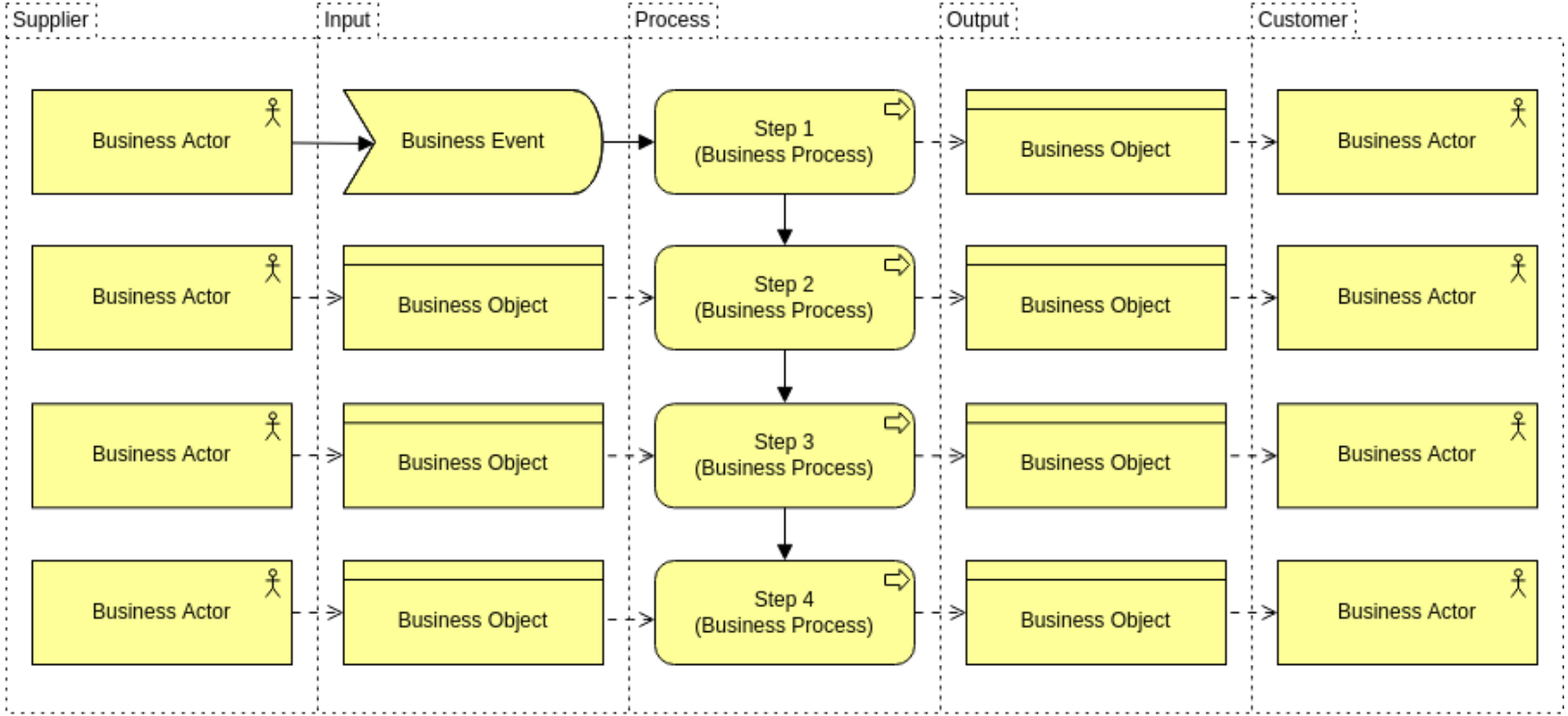
Sketch View

Brainstorm your ideas with elements written on “stickies”. Quickly design and create “soft” models, capturing the essence of your modelling process without worrying about the technicalities of the language.

Share and refine your ideas before transforming them to ArchiMate views.



SIPOCS (Six-Sigma)



The providers of inputs to the process steps.

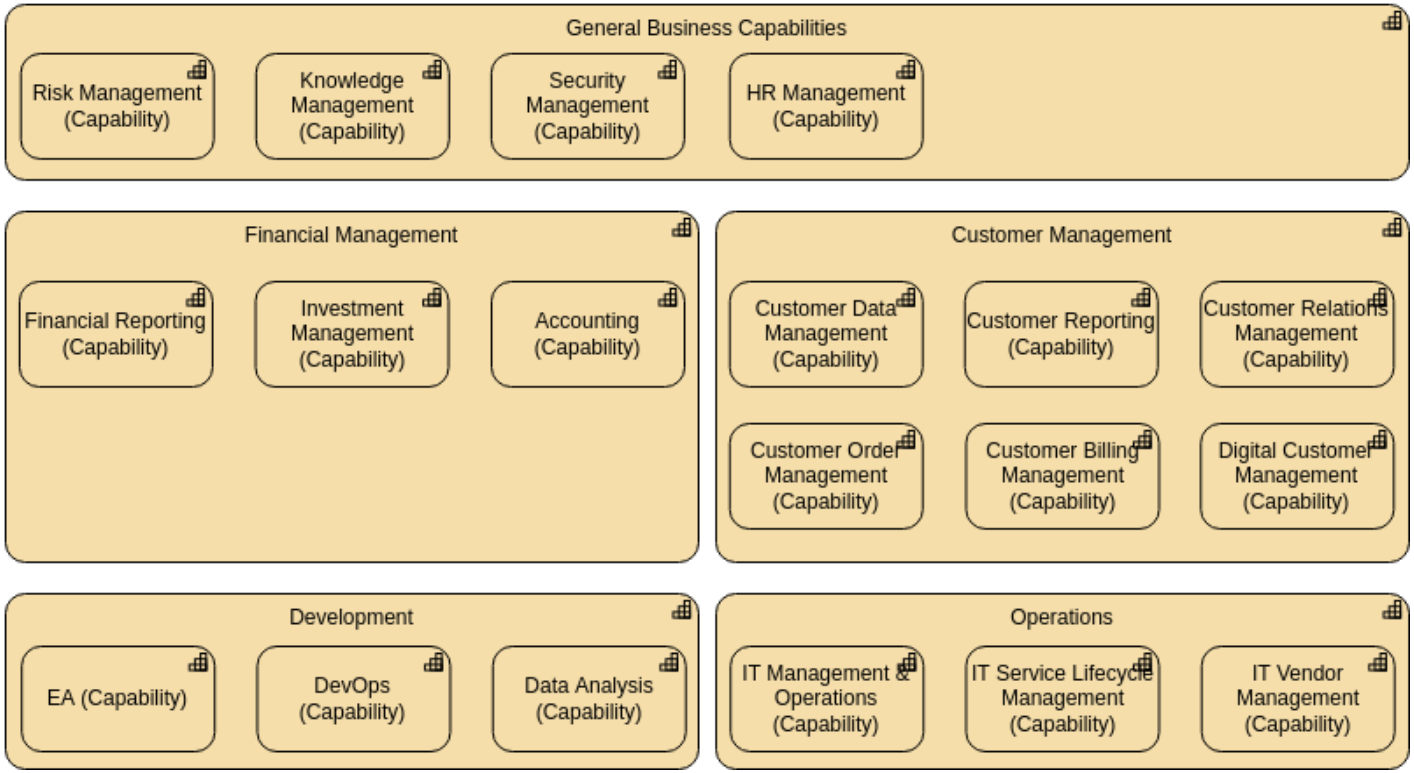
Information, materials, resources required to perform the process.

The steps to perform the inputs into outputs, providing value to customers/ stakeholders.

The deliverable(s) (e.g. product or service, report) to be produced/ delivered from the process steps to the internal or external

The actor/ role that gain benefits from the process.

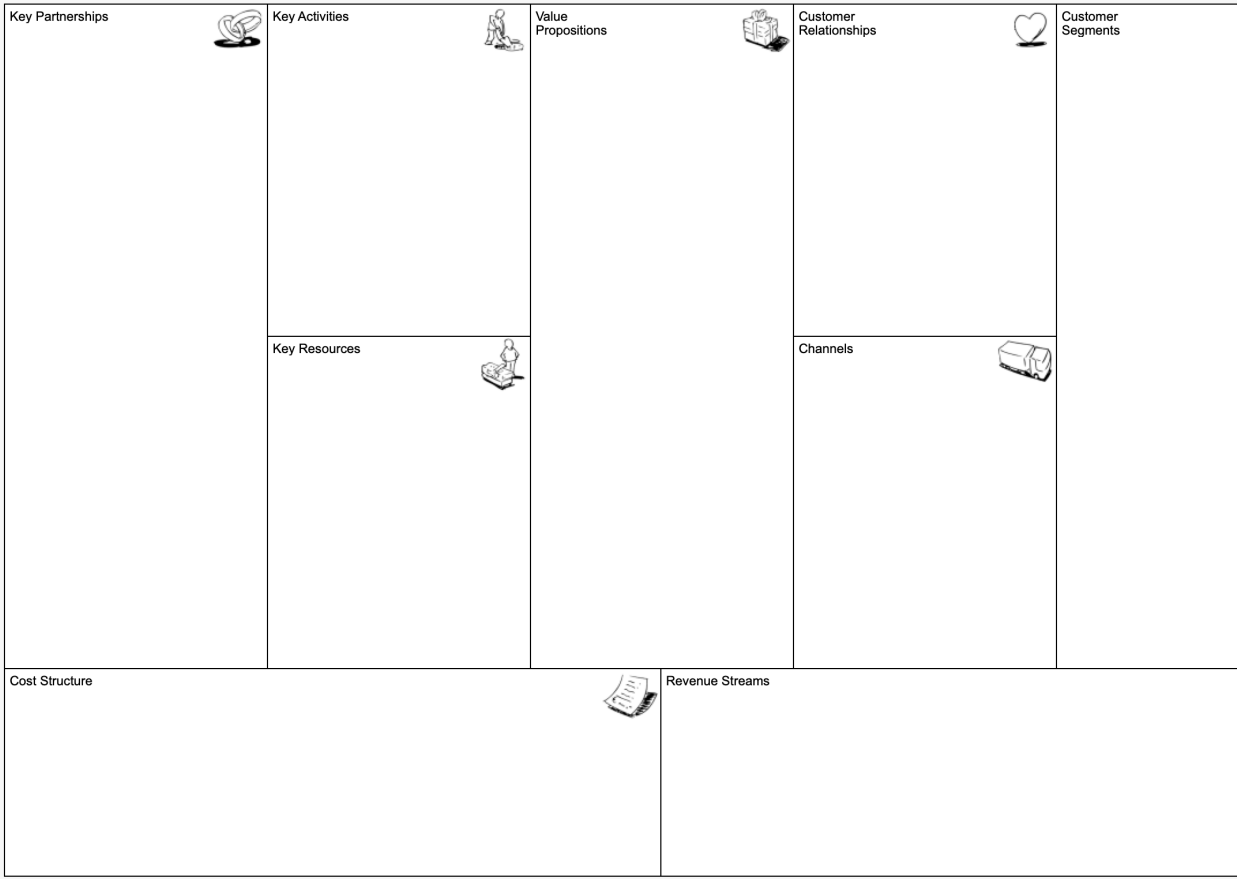
Capability Map



Business Model Canvas

The Business Model Canvas is a strategic management template used for developing new business models and documenting existing ones.

It offers a visual chart with elements describing a firm's or product's value proposition, infrastructure, customers, and finances, assisting businesses to align their activities by illustrating potential trade-offs.

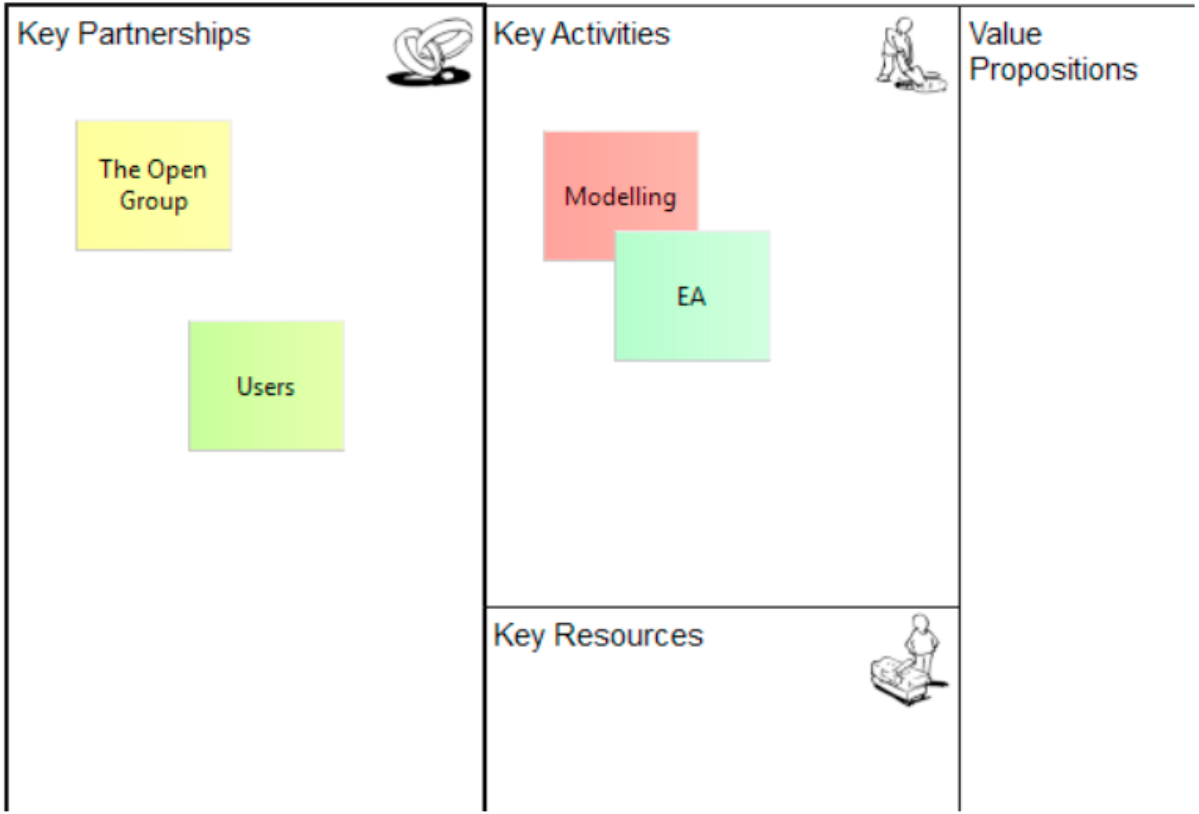


Canvas's

Add Stickies to the Canvas.

Another Canvas type is SWOT Template.

You can also create new Canvas types, with the "Blank Canvas".



Maps

A "Map" View with View References to all the other Views in the model