Architecture Modelling Standards

Table of Contents

Enterprise Architecture Viewpoints

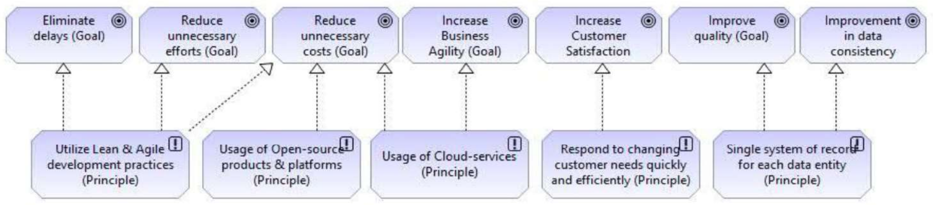
The Enterprise Architecture Viewpoints **define abstractions on the set of models** representing the enterprise architecture, each aimed at a particular type of stakeholder and addressing a particular set of concerns. Viewpoints can be used to view certain aspects in isolation, and to relate two or more aspect, and are grouped into three(3) main categories to distinguish the level of detail at each phase of the architecture engagement. These are:

- ***Domain/Strategic Architecture* - the way that a business comes together from the ground up**. This includes such pieces of information as the stakeholders, the mission and the vision of the company. Other important aspects include the plans for operation, the core strategies and the actions needed to accomplish them
- ***Segment Architecture* - detailed, formal descriptions of areas within an enterprise**, used at the program or portfolio level to organize and align change activity. Segment architecture is driven by business management and delivers products that improve the delivery of services to citizens and agency staff
- ***Solution Architecture* - an architectural description of a specific solution**. SAs combine guidance from different enterprise architecture viewpoints (business, information and technical), as well as from the enterprise solution architecture

Strategic Architecture Viewpoints

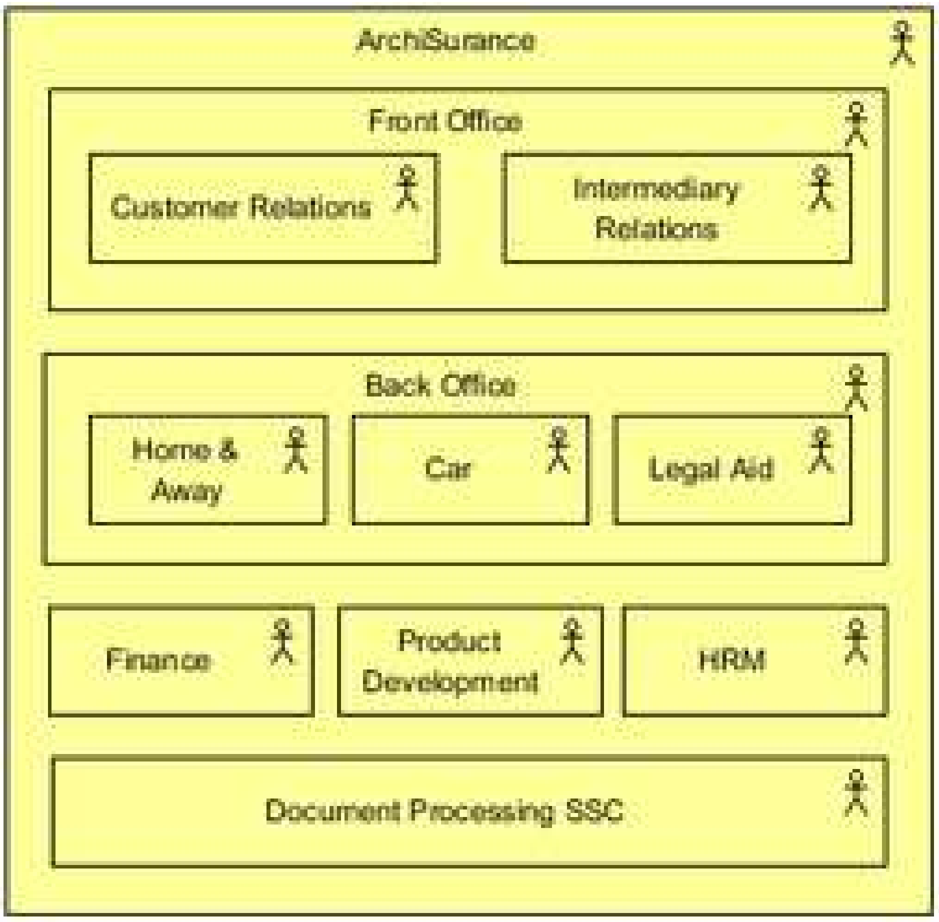
- **Principles Viewpoint** - Describes the principles that are relevant to the design problem at hand, including the relationship between goals that motivate these principles
- **Organizational Structure Viewpoint** - Describes the (internal) organization of a company, department, network of companies, or of another organizational entity.
- **Capability Reference Model** - Gives an overview of what the business does from a pure business perspective
- **Application Reference Model** - Categorizes the system- and application-related standards that support the delivery of business capabilities
- **Technology Reference Model** - Categorizes the technologies to support the construction, delivery, and exchange of business and application components
- **Data Reference Model** - Describes the data and information supporting business line operations
- **Implementation and Migration Viewpoint** - Relate programs and projects to the parts of the architecture that they implement.

Principles Viewpoint


Eliminate delays (Goal)
Reduce unnecessary efforts (Goal)
Reduce unnecessary costs (Goal)
Increase Business Agility (Goal)
Increase Customer Satisfaction
Improve quality (Goal)
Improvement in data consistency
Utilize Lean & Agile development practices (Principle)
Usage of Open-source products & platforms (Principle)
Usage of Cloud-services (Principle)
Respond to changing customer needs quickly and efficiently (Principle)
Single system of record for each data entity (Principle)

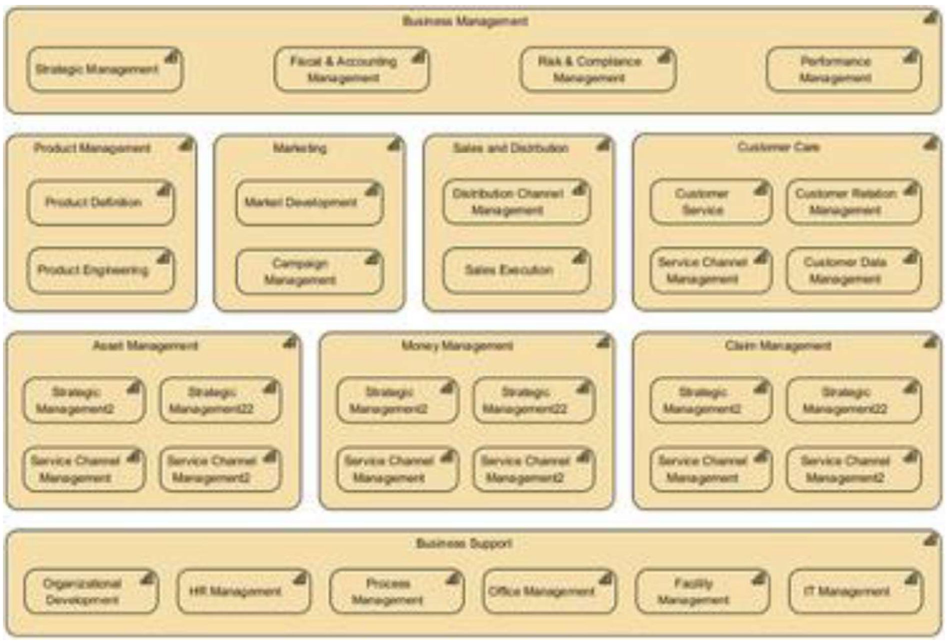
A model with principles that are relevant to the design problem at hand, including the relationship between goals that motivate these principles.

Organizational Structure Viewpoint



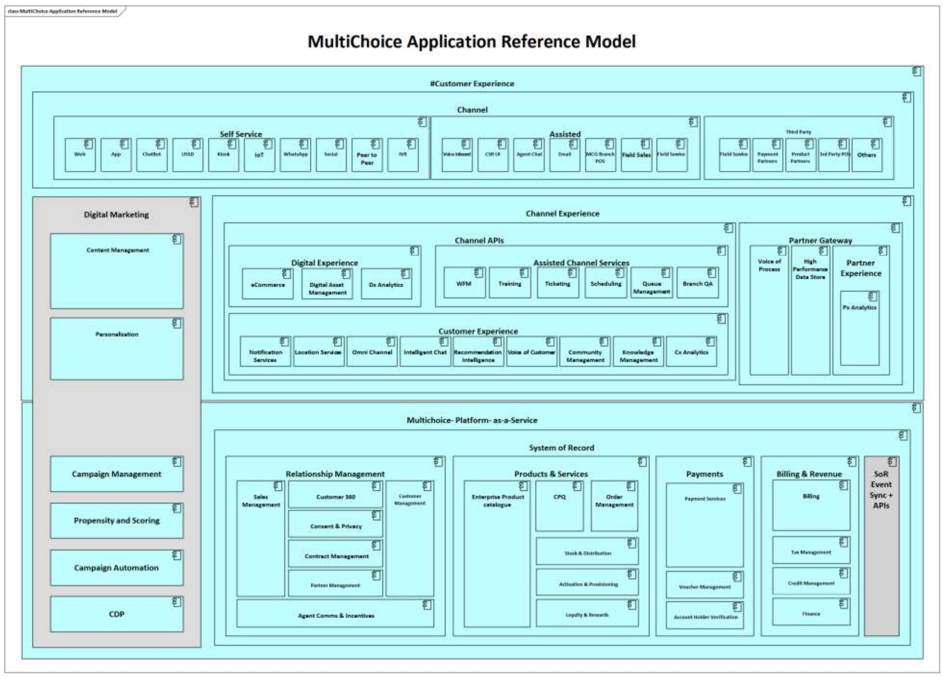
The organization viewpoint focuses on the (internal) organization of a company, department, network of companies, or of another organizational entity. It is possible to present models in this viewpoint as nested block diagrams, but also in a more traditional way, such as organizational charts. The organization viewpoint is very useful in identifying competencies, authority, and responsibilities in an organization

Capability Reference Model



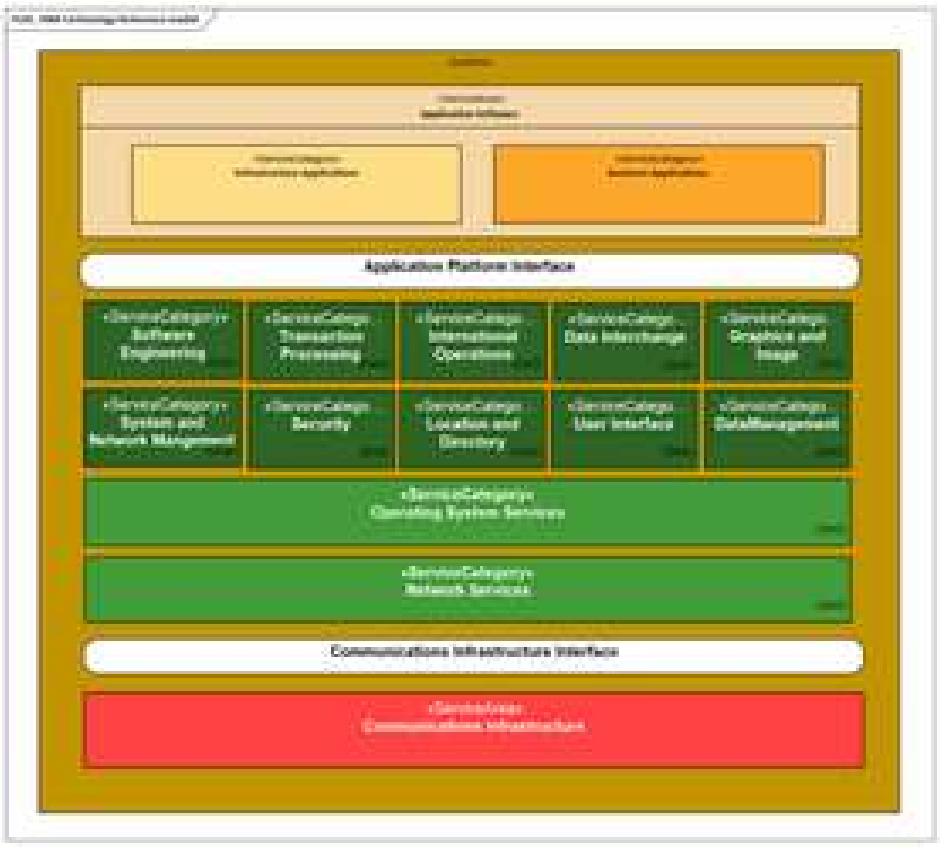
The capability map viewpoint allows the business to create a structured overview of the capabilities of the enterprise. A capability map typically shows two or three levels of capabilities across the entire enterprise. It can, for example, be used as a heat map to identify areas of investment. In some cases, a capability map may also show specific outcomes delivered by these capabilities

Application Reference Model



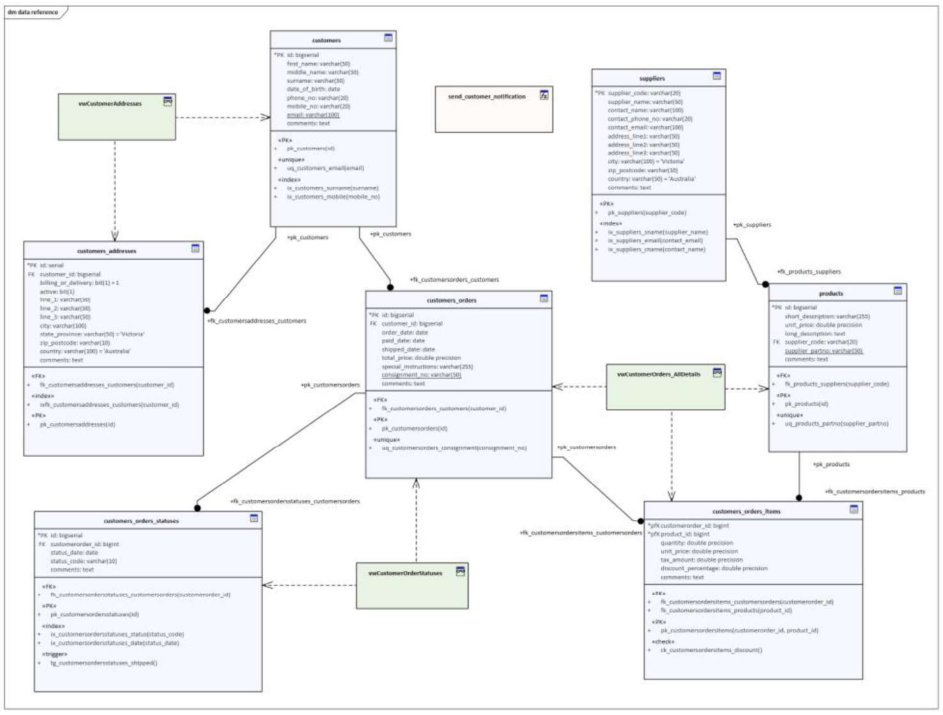
Application Reference Model (ARM) categorizes the system- and application-related standards that support the delivery of business capabilities. This is done by completing the description of the ARM to show only system and application elements affected by this piece of architecture work and the clear link to selected Application Components from TAM in the GT

Technology Reference Model


Application Platform Interface
«ServiceCategory» Software Engineering
«ServiceCatego» Transaction Processing
«ServiceCatego» International Operations
«ServiceCatego» Data Interchange
«ServiceCatego» Graphics and Image
«ServiceCategory» System and Network Mangement
«ServiceCatego» Security
«ServiceCatego» Location and Directory
«ServiceCatego» User Interface
«ServiceCatego» DataManagement
«ServiceCategory» Operating System Services
«ServiceCategory» Network Services
Communications Infrastructure Interface
Communications Infrastructure

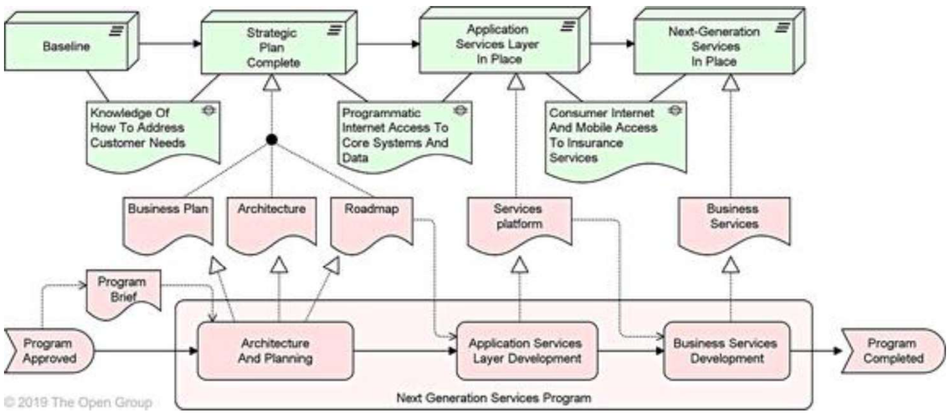
The Technical Reference Model (TRM) provides a foundation to categorize the technologies to support the construction, delivery, and exchange of business and application components. The TRM provides a set of architectural and solution building blocks that will ultimately provide the platform for business and infrastructure applications that will deliver the application and infrastructure services. The Technical Reference Model ensures that architectures are created consistently and repeatedly based on a standard set of elements. The model should be created as part of the set up of the architecture programs but will typically require extending as technology standards are introduced and retired

Data Reference Model



Data Reference Model (DRM) describes, at an aggregate level, the data and information supporting business line operations. The definition of done is a complete description of the DRM. In scope: Only data elements affected by this piece of architecture work.

Implementation and Migration Viewpoint


Baseline
Strategic Plan Complete
Application Services Layer In Place
Next-Generation Services In Place
Knowledge Of How To Address Customer Needs
Programmatic Internet Access To Core Systems And Data
Consumer Internet And Mobile Access To Insurance Services
Business Plan
Architecture
Roadmap
Services platform
Business Services
Program Brief
Program Approved
Architecture And Planning
Application Services Layer Development
Business Services Development
Program Completed
Next Generation Services Program
© 2019 The Open Group

The implementation and migration viewpoint is used to relate programs and projects to the parts of the architecture that they implement. This view allows modeling of the scope of programs, projects, project activities in terms of the plateaus that are realized or the individual architecture elements that are affected. In addition, the way the elements are affected may be indicated by annotating the relationships

This viewpoint is useful in showing direct correlation of work packages to the actual architecture solution developed where;

- The plateaus represent the Program Increments and Iterations with predefined start and end dates

- The Work Packages represent the architectural Epics/Features/User Stories imported directly from ADO
- The Deliverable represents the specific architectural work that has to be done a particular work package
- The Gap highlights the specific differences in the architectures delivered between each program increment/iteration

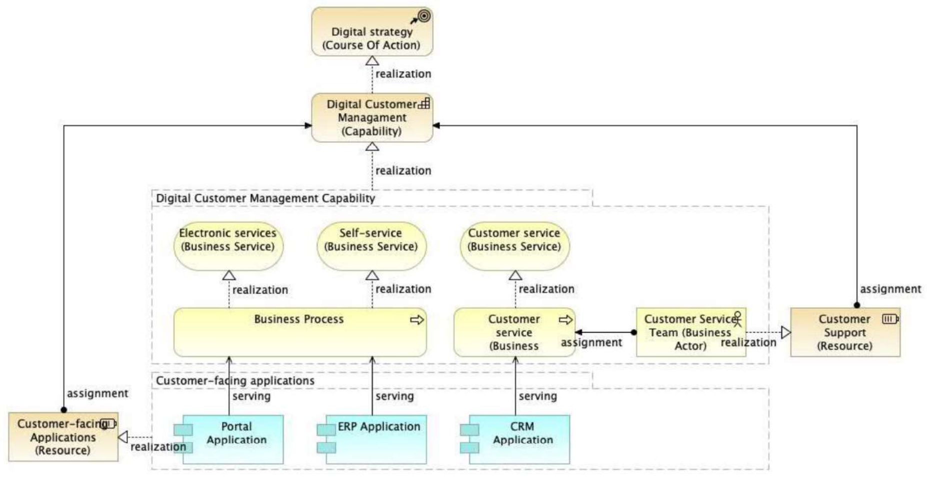
Furthermore, this viewpoint can be used in combination with the programs and projects viewpoint to support portfolio management:

- The *programs and projects viewpoint* is suited to relate business goals to programs and projects. For example, this makes it possible to analyze at a high level whether all business goals are covered sufficiently by the current portfolio(s).
- The *implementation and migration viewpoint* is suited to relate business goals (and requirements) via programs and projects to (parts of) the architecture. For example, this makes it possible to analyze potential overlap between project activities or to analyze the consistency between project dependencies and dependencies among plateaus or architecture elements.

Segment and Platform Architecture Viewpoints

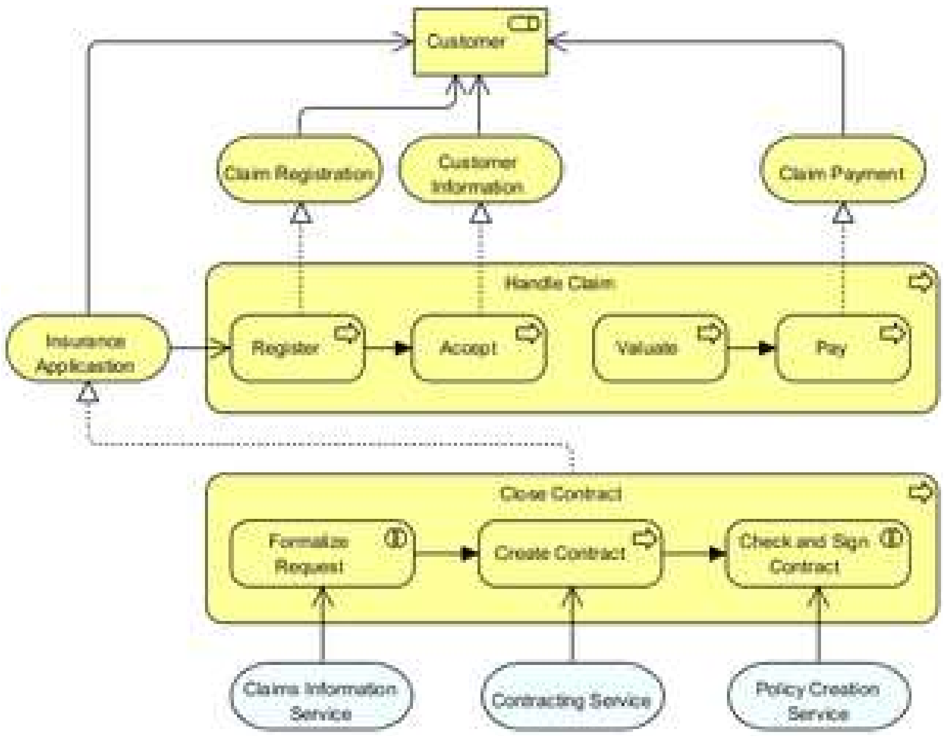
- **Capability Planning Viewpoint** - Describe the planning, engineering, and delivery of strategic business capabilities to the enterprise. It is business-driven and business-led and combines the requisite efforts of all lines of business to achieve the desired capability
- **Business Process Cooperation Viewpoint** - Describe the relationships of one or more business processes with each other and/or with their environment
- **Business Capability Dependency Viewpoint** - Depict the sequence of events when applications are involved in realizing a business capability
- **Application Behaviour Viewpoint** - Describes the internal behavior of an application; e.g., as it realizes one or more application services.
- **Service Realization Viewpoint** - Describe how one or more business services are realized by the underlying processes (and sometimes by application components).
- **Data Object Relationship Viewpoint** - Describes the structure of the information used in the enterprise or in a specific business process or application, in terms of data types or (object-oriented) class structures.
- **Requirements Realization Viewpoint** - Describe the realization of requirements by the core elements, such as business actors, business services, business processes, application services, application components, etc.
- **Implementation and Deployment Viewpoint** - Describe how one or more applications are realized on the infrastructure.

Capability Planning Viewpoint


Digital strategy (Course Of Action)
realization
Digital Customer Managament (Capability)
realization
Digital Customer Management Capability
Electronic services (Business Service)
Self-service (Business Service)
Customer service (Business Service)
realization
realization
realization
assignment
Business Process
Customer service (Business
assignment
Customer Service Team (Business Actor)
realization
Customer Support (Resource)
Customer-facing applications
serving
serving
serving
assignment
Customer-facing Applications (Resource)
realization
Portal Application
ERP Application
CRM Application

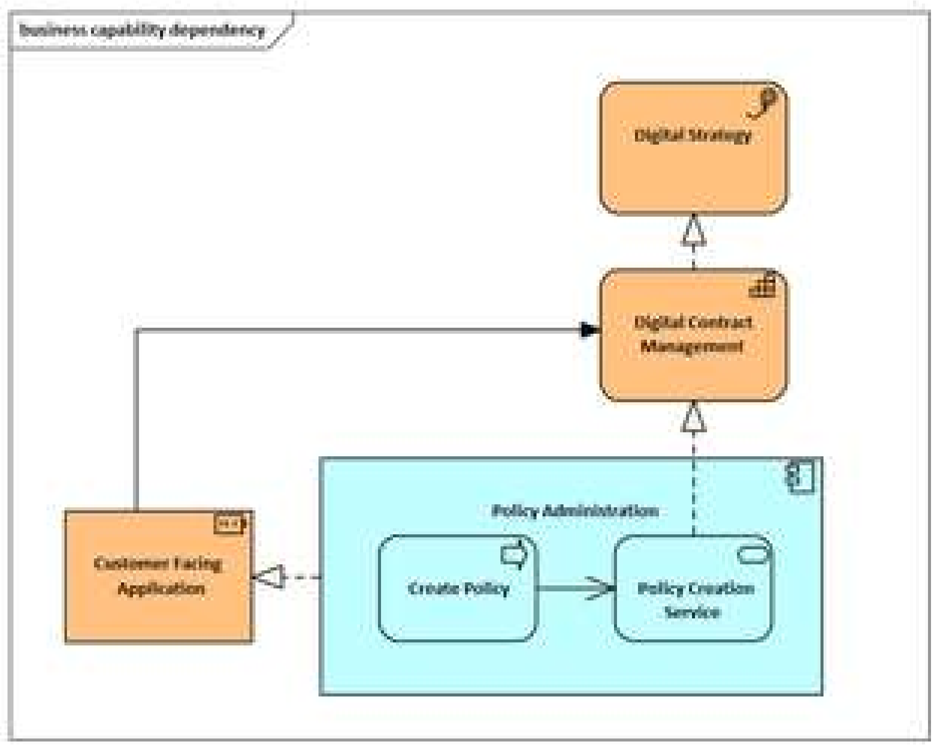
Capability planning view describes the planning, engineering, and delivery of strategic business capabilities to the enterprise. It is business-driven and business-led and combines the requisite efforts of all lines of business to achieve the desired capability. Often the need for these capabilities are discovered and refined using business scenarios

Business Process Cooperation Viewpoint



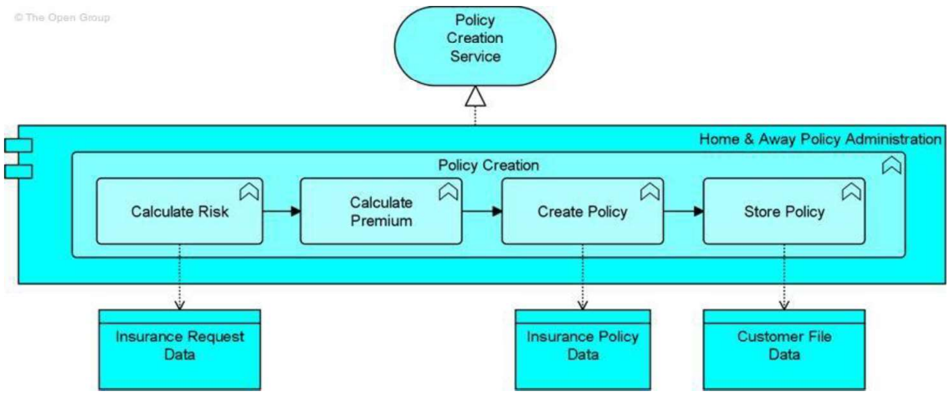
The business process cooperation viewpoint is used to show the relationships of one or more business processes with each other and/or with their environment. It can be used both to create a high-level design of business processes within their context and to provide an operational manager responsible for one or more such processes with insight into their dependencies

Business Capability Dependency Viewpoint


business capability dependency
Digital Strategy
Digital Contract Management
Customer Facing Application
Policy Administration
Create Policy
Policy Creation Service

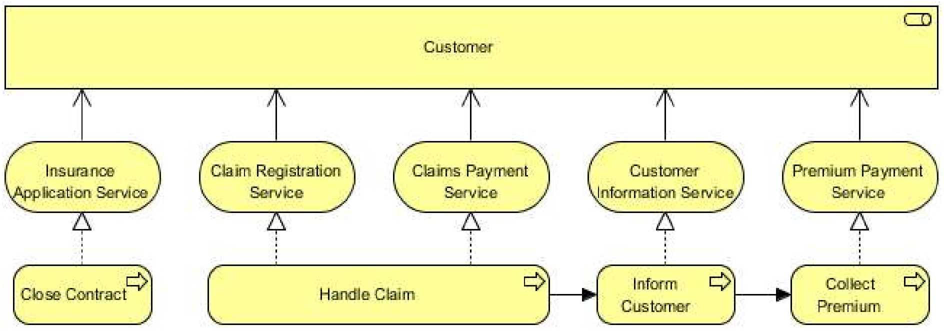
This viewpoint identifies possible rationalization points in the architecture in order to provide more timely information to business users, and also identifying the dependencies of business capabilities to the relevant applications and application processes

Application Behaviour Viewpoint



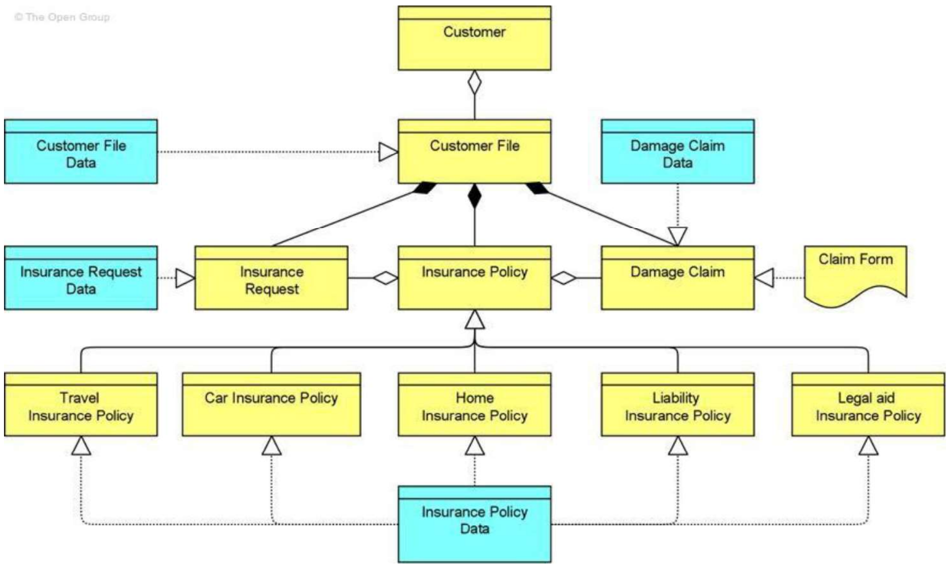
The Application Behavior viewpoint describes the internal behavior of an application; e.g., as it realizes one or more application services. This viewpoint is useful in designing the main behavior of applications, or in identifying functional overlap between different applications

Service Realization Viewpoint



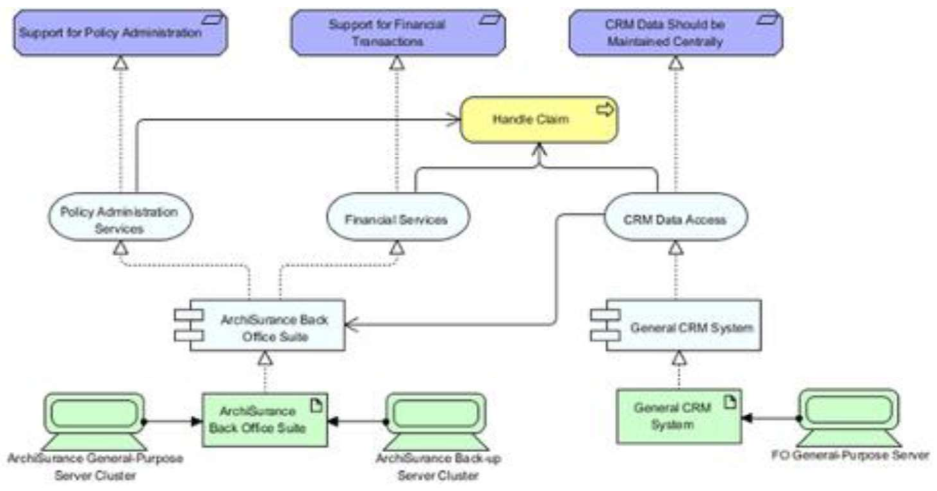
The service realization viewpoint is used to show how one or more business services are realized by the underlying processes (and sometimes by application components). Thus, it forms the bridge between the business products viewpoint and the business process view. It provides a "view from the outside" on one or more business processes.

Data Object Relationship Viewpoint



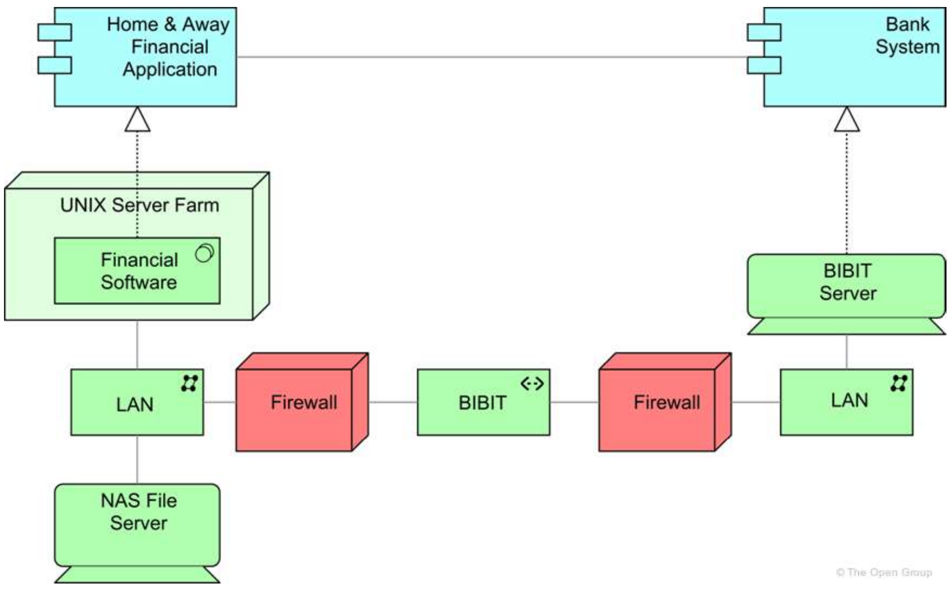
This viewpoint is comparable to the traditional information models created in the development of almost any information system. Describes the structure of the data (information) used in the enterprise or in a specific business process or application, in terms of data types or (object-oriented) class structures. Furthermore, it may show how the information at the business level is represented at the application level in the form of the data structures used there, and how these are then mapped onto the underlying infrastructure

Requirements Realization Viewpoint


Support for Policy Administration
Support for Financial Transactions
CRM Data Should be Maintained Centrally
Handle Claim
Policy Administration Services
Financial Services
CRM Data Access
ArchiSurance Back Office Suite
General CRM System
ArchiSurance Back Office Suite
General CRM System
ArchiSurance General-Purpose Server Cluster
ArchiSurance Back-up Server Cluster
FO General-Purpose Server

The requirements realization viewpoint allows the designer to model the realization of requirements by the core elements, such as business actors, business services, business processes, application services, application components, etc. Typically, the requirements result from the goal refinement viewpoint. In addition, this viewpoint can be used to refine requirements into more detailed requirements. The aggregation relationship is used for this purpose

Implementation and Deployment Viewpoint



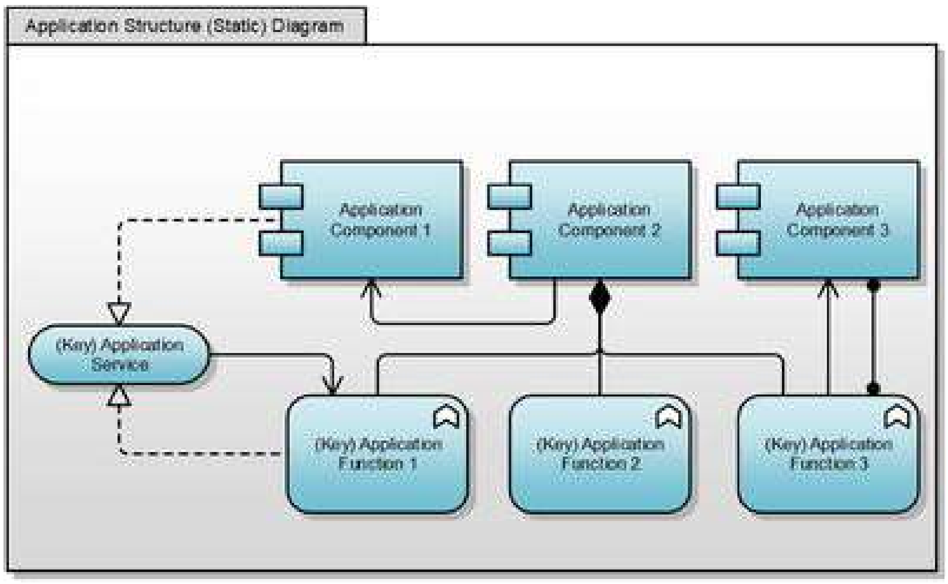
The implementation and deployment viewpoint shows how one or more applications are realized on the infrastructure. This comprises the mapping of applications and components onto artifacts, and the mapping of the information used by these applications and components onto the underlying storage infrastructure

Solution Architecture Viewpoints

There are five (5) main views that are required as minimum (MVP) for the Solution Architecture models that are required in the Wakanda Handover exercise space:

- **Application Structure/ Application Cooperation** - Describe the current application components that will be impacted by the solution architecture
- **Application Usage** - Describe the Applications that are part of the Solution Architecture and which Business Processes will use them
- **Application Collaboration** - Describe the target application components that will be impacted by the solution architecture
- **Data Flow/Logical Data Model** - Describes things about which an organization wants to collect data, and the relationships among these things
- **Technology Usage Viewpoin**t - Shows how applications are supported by the software and hardware technology

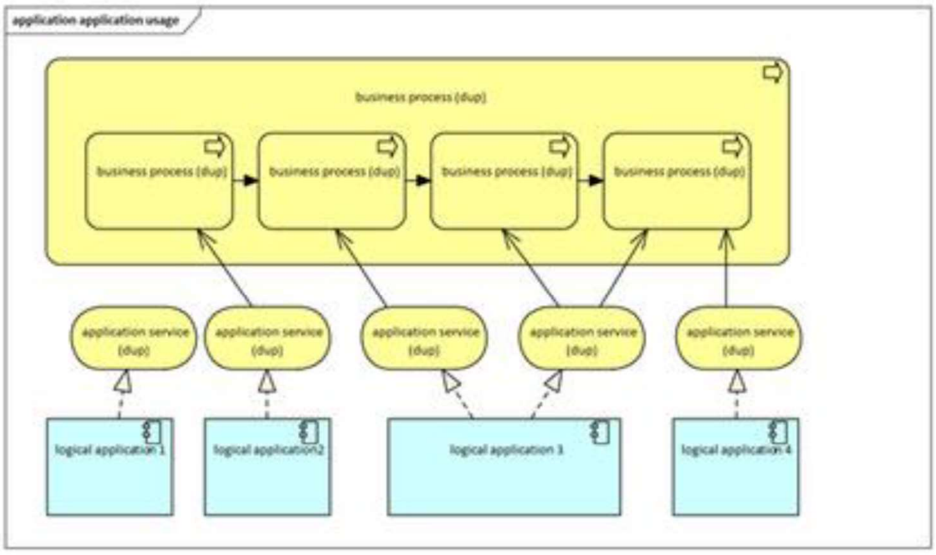
Application Structure/ Application Cooperation Viewpoint

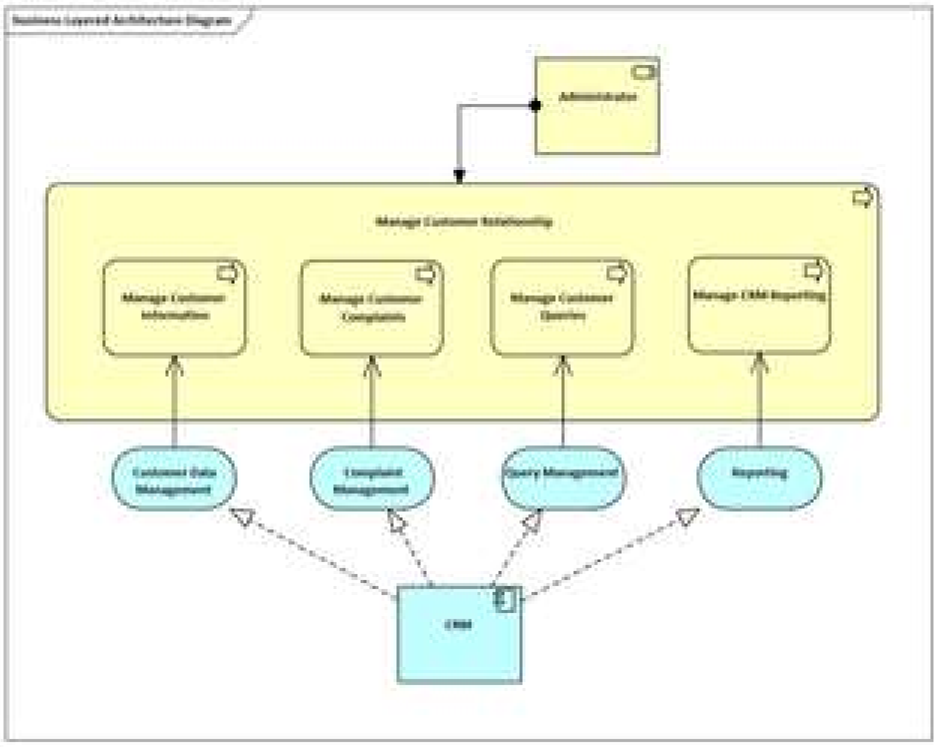

Application Structure (Static) Diagram
Application Component 1
Application Component 2
Application Component 3
(Key) Application Service
(Key) Application Function 1
(Key) Application Function 2
(Key) Application Function 3

The application cooperation viewpoint describes the relationships between applications components in terms of the information flows between them, or in terms of the services they offer and use. This viewpoint is typically used to create an overview of the application landscape of an organization. This viewpoint is also used to express the (internal) cooperation or orchestration of services that together support the execution of a business process (typically modelled in conjunction with the Segment Architect).

NOTE: This may or may not exist

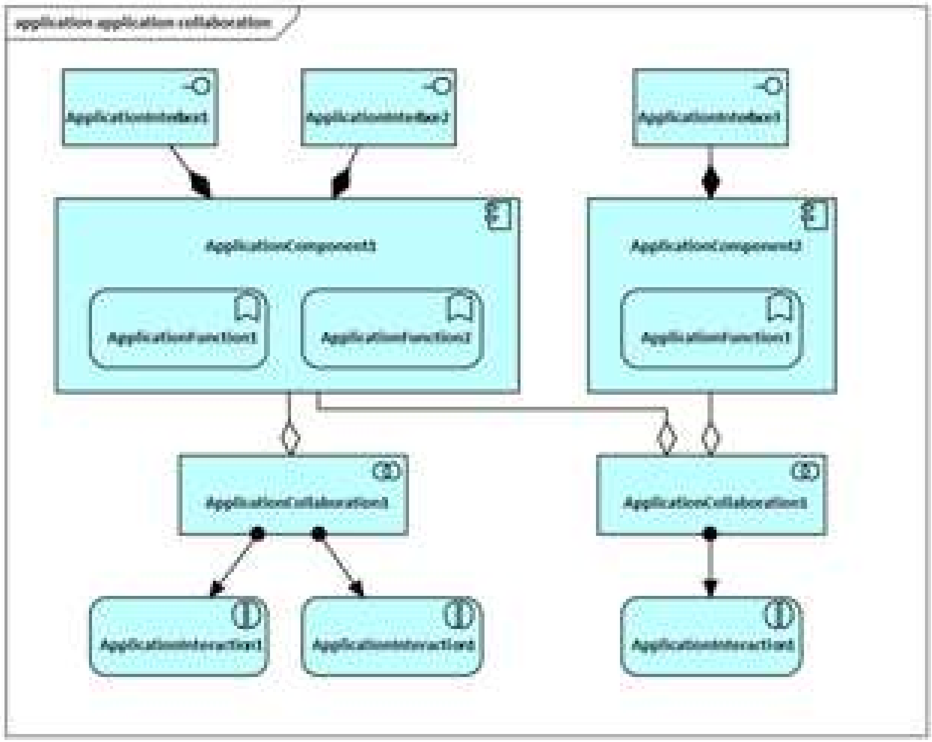
Application Usage Viewpoint

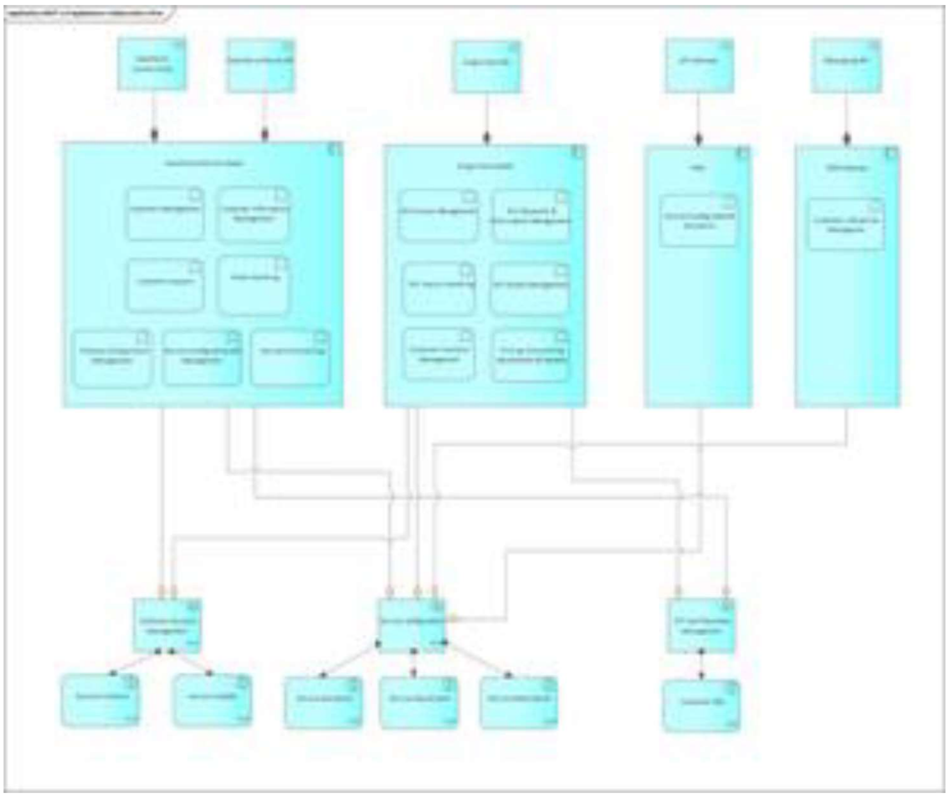




The application usage viewpoint describes how applications are used to support one or more business processes, and how they are used by other applications. It can be used in designing an application by identifying the services needed by business processes and other applications, or in designing business processes by describing the services that are available. Furthermore, since it identifies the dependencies of application/business processes upon applications, it may be useful to operational managers responsible for these processes (typically modelled in conjunction with the Segment Architect & Business Analyst)

Application Collaboration Viewpoint





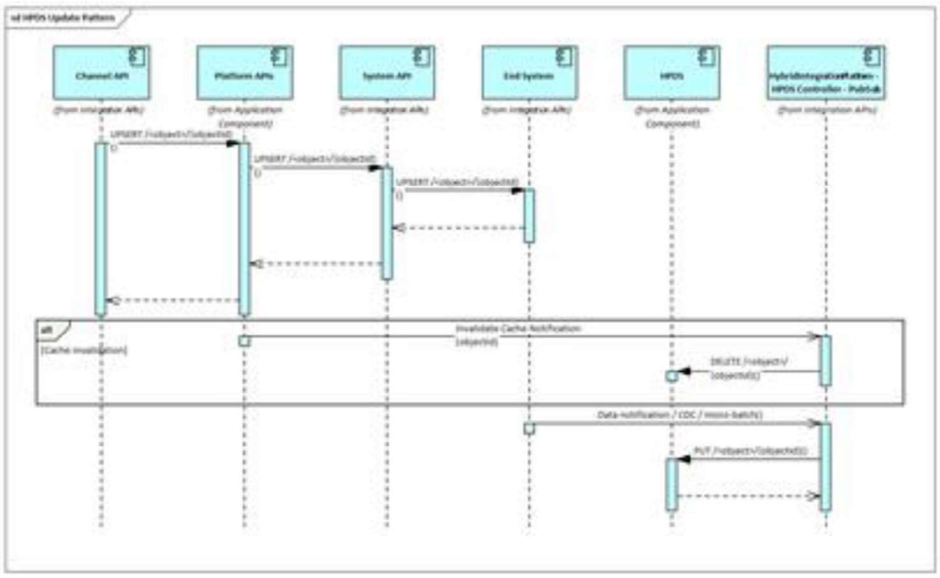
Describes the target application components that will be impacted by the solution architecture (typically modelled in conjunction with the Segment Architect)

The Target Application Architecture reflects each application component and nested are the correspondent eTOM (enhanced Telecom Operations Map) process types from TM Forum's Business Process Framework:

- Each application will be composed for one or multiple interfaces.
- The collaboration between application components will result in an interaction (e.g.: Service Activation), that is linked to an Interaction View

Sequence Diagram

A sequence diagram is a type of interaction diagram because it **describes how—and in what order—a group of objects works together**. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. These are usually mapped as representations from Interaction components

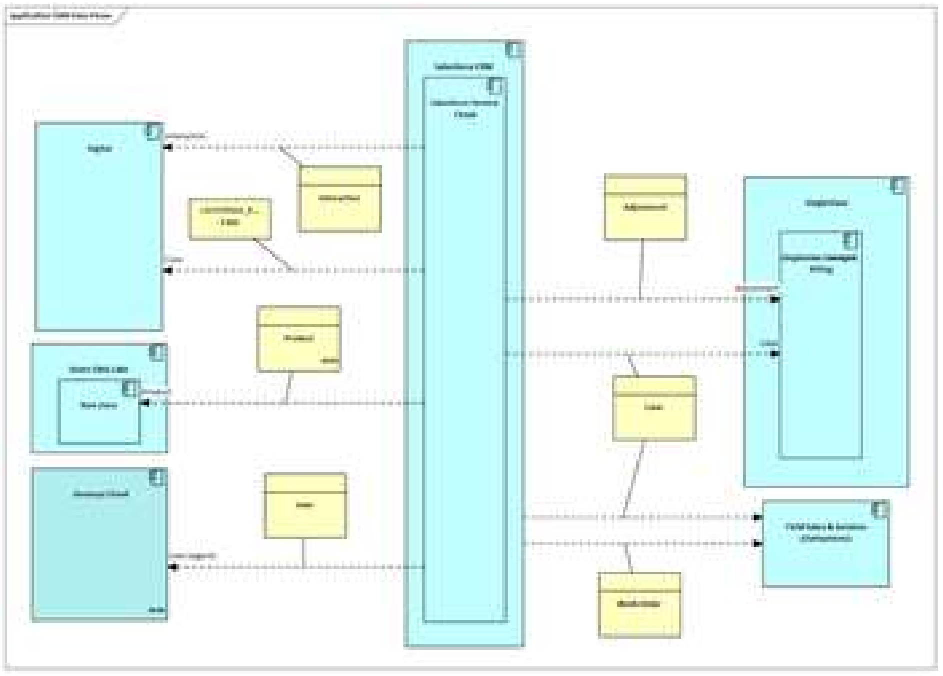

sd HPOS Update Pattern
Channel API
(from Integration APIs)
Platform APIs
(from Application Component)
System API
(from Integration APIs)
End System
(from Integration APIs)
HPOS
(from Application Component)
HybridIntegrationPlatform - HPOS Controller - PubSub
(from Integration APIs)
UPSERT /<object>/<objectId>
UPSERT /<object>/<objectId>
UPSERT /<object>/<objectId>
alt
[Cache invalidation]
Invalidate Cache Notification (objectId)
DELETE /<object>/<objectId(s)>
Data notification / CDC / mini-batch(s)
PUT /<object>/<objectId(s)>

Sequence diagrams can be useful references for businesses and other organizations. Try drawing a sequence diagram to:

- Represent the details of a UML use case.
- Model the logic of a sophisticated procedure, function, or operation.
- See how objects and components interact with each other to complete a process.
- Plan and understand the detailed functionality of an existing or future scenario.

Data Flow

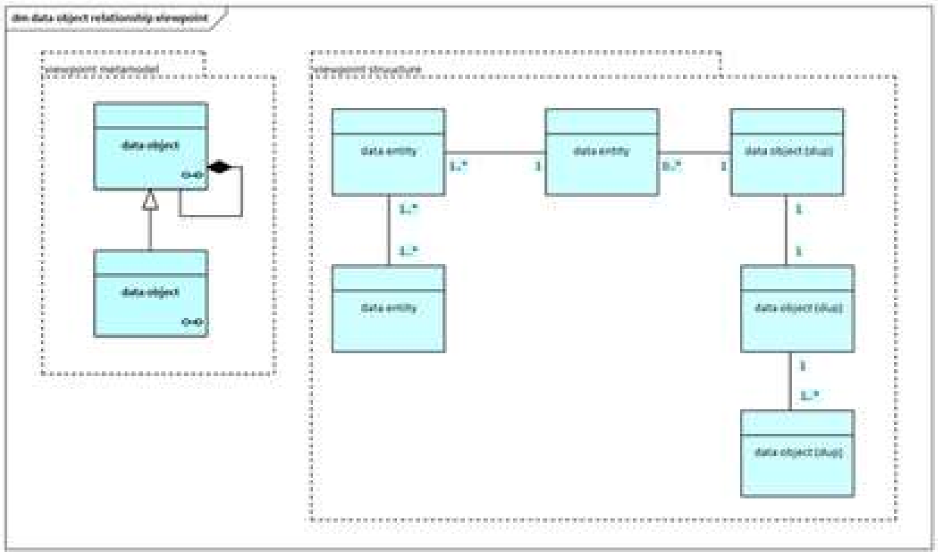
A **data flow** model identifies the highest-level relationships between the different entities, and the flow of data between different components of the system



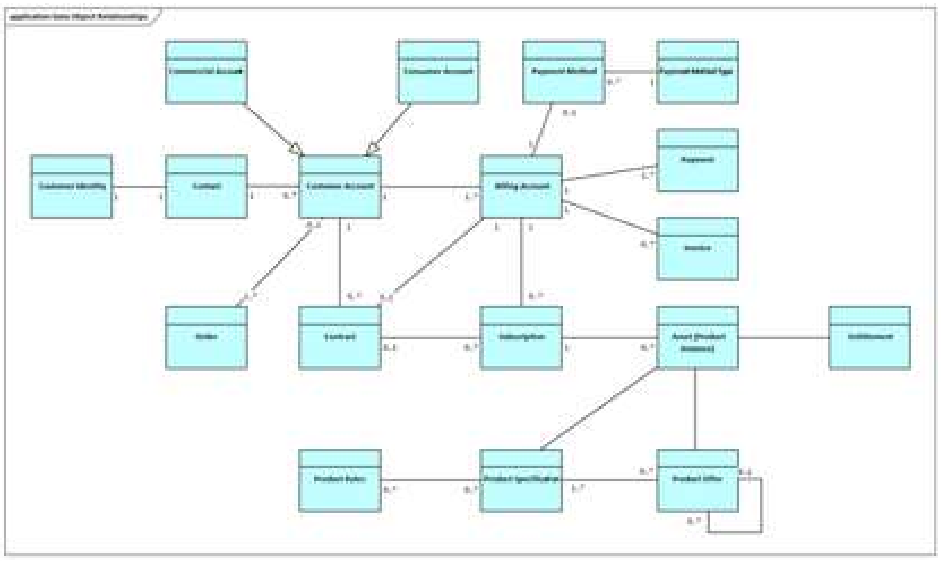
Features of conceptual data model include:

- Includes the important entities and the relationships among them.
- No attribute is specified.
- No primary key is specified.

Logical Data Model



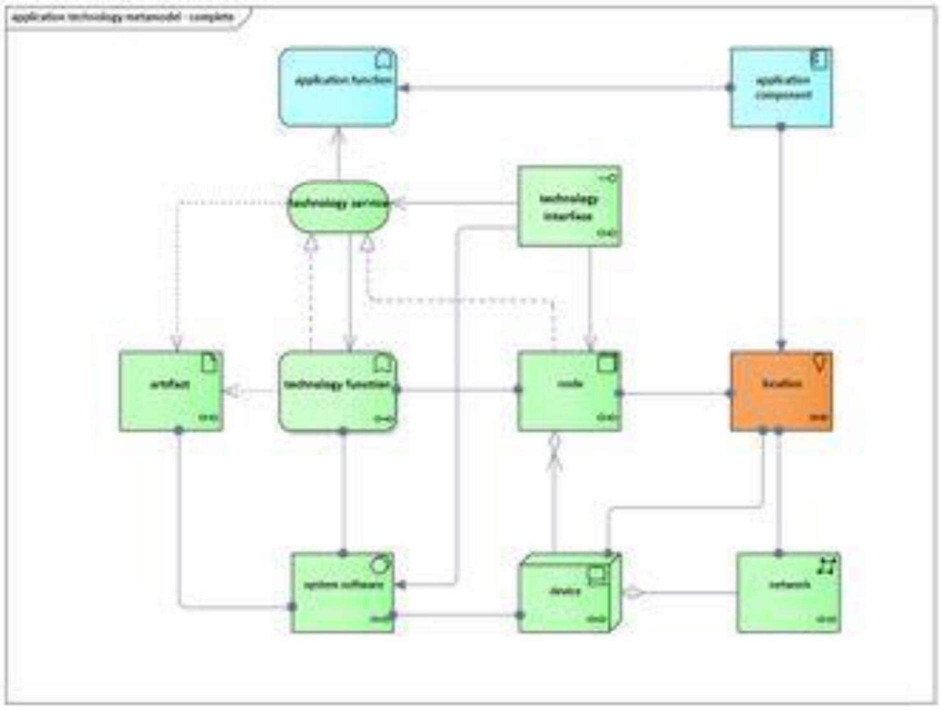
A **logical data** model describes the data in as much detail as possible, without regard to how they will be physical implemented in the database.



Features of a logical data model include:

- Includes all entities and relationships among them.
- Normalization occurs at this level.

Technology Usage Model



The technology usage viewpoint shows how applications are supported by the software and hardware technology: the technology services are delivered by the devices; system software and networks are provided to the applications.

This viewpoint plays an important role in the analysis of performance and scalability, since it relates the physical infrastructure to the logical world of applications. It is very useful in determining the performance and quality requirements on the infrastructure based on the demands of the various applications that use it

ArchiMate Elements

This standard is the specification of the ArchiMate Enterprise Architecture modeling language, a visual language with a set of default iconography for describing, analyzing, and communicating many concerns of Enterprise Architectures as they change over time. The standard provides a set of entities and relationships with their corresponding iconography for the representation of Architecture Descriptions

Link to Archimate Standard

https://pubs.opengroup.org/architecture/archimate3-doc/toc.html

Application Structure Elements

The Application Layer elements are typically used to model the Application Architecture that describes the structure, behavior, and interaction of the applications of the enterprise

Application Component

An application component represents an encapsulation of application functionality aligned to implementation structure, which is modular and replaceable. It encapsulates its behavior and data, exposes services, and makes them available through interfaces.

- may be assigned to one or more application functions.
- has one or more application interfaces, which expose its functionality.

The name of an application component should preferably be a noun

Application Interface

An application interface represents a point of access where application services are made available to a user, another application component, or a node.

- may be part of an application component through composition, which means that these interfaces are provided by that component.
- can serve other application components.
- can be assigned to application services, which means that the interface exposes these services to the environment.

The name of an application interface should preferably be a noun.

Application Function

An application function represents automated behavior that can be performed by an application component.

- may realize one or more application services.
- may access data objects.

The name of an application function should preferably be a verb ending with “ing”; e.g., “accounting”.

Application Process

An application process represents a sequence of application behaviours that achieves a specific outcome.

- may realize application services.
- may access data objects.

The name of an application process should clearly identify a series of application behaviours; e.g., “Claims adjudication process”, or “General ledger update job”.

Application Service

An application service represents an explicitly defined exposed application behavior *(not to be confused with Business Service)*

- may serve business processes, business functions, business interactions, or application functions.
- may access data objects.

The name of an application service should preferably be a verb ending with “ing”; e.g., “transaction processing”. Also, a name explicitly containing the word “service” may be used

Application Interaction

An application interaction represents a unit of collective application behavior performed by (a collaboration of) two or more application components

Application Collaboration

An application collaboration represents an aggregate of two or more application internal active structure elements that work together to perform collective application behavior.

Data Object

A data object represents data structured for automated processing.

- can be used or produced by application services.
- can be accessed by an application function, application interaction, or application service.

- may realize a business object, and may be realized by an artifact.
- may have association, specialization, aggregation, or composition relationships with other data objects.

The name of a data object should preferably be a noun.

Technology Structure Elements

Node

A node represents a computational or physical resource that hosts, manipulates, or interacts with other computational or physical resources

- can perform technology behavior and execute, store, and process technology objects such as artifacts
- can be interconnected by paths.
- may be assigned to an artifact to model that the artifact is deployed on the node.

The name of a node should preferably be a noun. A node may consist of sub-nodes.

System Software

System software represents software that provides or contributes to an environment for storing, executing, and using software or data deployed within it.

- can be assigned to other system software
- can be assigned to artifacts, to model that these artifacts are deployed on that software.
- can realize other system software. A node can be composed of system software.

The name of system software should preferably be a noun referring to the type of execution environment; e.g., “JEE server”.

Location

A location represents a conceptual or physical place or position where concepts are located or performed

- used to model the places where (active and passive) structure elements such as business actors, application components, and devices are located.
- modeled by means of an aggregation relationship from a location to structure element.
- can also aggregate a behavior element, to indicate where the behavior is performed

Communication Network (Object)

A communication network represents a set of structures that connects nodes for transmission, routing, and reception of data

- connects two or more devices

- A communication network realizes one or more paths
- A communication network can consist of sub-networks.
- It can aggregate devices and system software

Technology Function

A technology function represents a collection of technology behavior that can be performed by a node

- describes the internal behaviour of a node

The name of a technology function should preferably be a verb ending with “-ing”.

Device

A device represents a physical IT resource upon which system software and artifacts may be stored or deployed for execution

- is a specialization of a node that represents a physical IT resource with processing capability
- typically used to model hardware systems such as mainframes, PCs, or routers
- may be composite; i.e., consist of sub-devices.
- can be interconnected by communication networks
- can be assigned to artifacts and to system software
- can contain one or more devices.

The name of a device should preferably be a noun referring to the type of hardware; e.g., “IBM System z mainframe”.

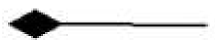
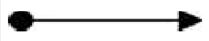
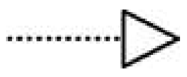
Architecture Object Naming Conventions

In order to keep the repository clean and the architecture models neat, we need to adopt the following naming practices for all architecture objects forming part of our object library collection:

- ***Refer to previous section above on best practices on how to name elements depending on the type***
- Keep the name short and descriptive (try limit the object name to less than 20 words)
- Always capitalize every word in the name
- Avoid hyphenated words (unless absolutely necessary)
- Do not use abbreviations in the name (unless universally understood and agreed upon)
- Do not use special characters in the name **(` ~ ! @ # $ % ^ & * _ = + [{] } \ | ; : ' “ , < > / ?)**

This will also aid in drawing the analytics reports on the health of the repository, minimize cluttered appearances of architecture models and simplify the search of these objects.

Subset of Relationships

Allowed Relationships		Notation	Role Names
Composition	Represents that an element consists of one or more other concepts.		← composed of → composed in
Assignment	Represents the allocation of responsibility, performance of behavior, storage, or execution.		← assigned to → has assigned
Aggregation	Represents that an element combines one or more other concepts.		← aggregates → aggregated in
Realization	Represents that an entity plays a critical role in the creation, achievement, sustenance, or operation of a more abstract entity.		← realizes → realized by
Flow	Represents transfer from one element to another.		← flows to → flows from
Specialization	Represents that an element is a particular kind of another element.		← specializes → specialized by
Serving	Represents that an element provides its functionality to another element.		← serves (used by) → served by (uses)
Access	Represents the ability of behavior and active structure elements to observe or act upon passive structure elements.		← accesses → accessed by

About this Page

Page Owner

The document Owner is accountable for the creation and ongoing maintenance of the document - there may be many contributors however the Owner is always accountable and therefore needs to be identified. The Portfolio Lead by deafalut becomes the Owner if the current Owner rolls off or is unavailable

Owner	Portfolio Lead	Reviewers