Preface

The creation of Enterprise Architecture Views has always been critical to successful delivery of an organisation wide Transformation / Modernisation programme. Over the last 10 years or so many organisations decided that architecture was not important and in many cases was an inhibitor to rapid progress. These organisations looked on Agile as a replacement for proper architecture rigor. This is a bit like deciding to build an office on the fifth floor from the top of a skyscraper and having no idea how tall the building will be, or starting a journey with enough supplies to get you to the end of the road and hoping a resupply point will be available as required. A significant number of these organisations are now regretting this decision and moving back to re-formalise architecture disciplines in their transformation programs.

However architecture disciplines have not helped themselves in the perception of value of the deliverables. In many cases the “beauty of the diagrams” and overly complicated vision of “end-state” allied with high levels of bureaucracy and governance hindered progress rather than assisting it.

For a transformation programme to be successful an Architecture Landscape “To-Be” state is a critical and foundational element that needs to be designed and agreed early in the programme.

This document provides a Point of View on designing a “To-Be” or “Target” architecture. It is not intended to be an instruction set, but rather an approach of what needs to be considered, and who should be involved.

If you are to take one thing away it is this;

"Vision without execution is just hallucination" - Thomas Edison.

The design of "To-Be" / "Target" architecture must ultimately be _*Deliverable*_.

It must be fully aligned and driven by the medium term business strategy(s)

It should reflect the realities of

- budget availability
- degree and rate of change the organisation can consume
- risk and complexity around deliverability
- compliance rules
- acknowledgement of the starting point ("As-Is")

Above all it must reflect what the business needs and when it is required.

Architecture State Representation

Describing Architecture State within IT has traditionally been about expressing how pieces of the puzzle connect, what technologies & systems are required and what their relevance is to the overall technology landscape.

Architecture State generally falls into three categories, “As-Is”, “To Be” and “Transitional”.

“As-Is” is in most terms a Technical focused (and biased) representation of the current state of play. It is generally used to figure out where we are coming from (how big the mess is) and to assist in the assessment of challenges and difficulties in delivering both tactical and strategic innovation.

“To-Be” is a desired end state within a medium term timeframe. It is a representation of where the technology landscape of an organisation needs to be in order to meet the business strategy. The “To-Be” state is an evolving target in that it should always be aligned with the business strategy to remain fit-for-purpose in delivering to the needs of the organisation. This does not mean it should be fluid, just that as with business strategies which have a three to five year horizon, so should the Architecture “To-Be” state.

“Transitional” states represent steps in HOW to evolve from “As-Is” to “To-Be”. Transitional states are probably the most important element of Architecture as they;

- Prove it is actually technically possible to get from A to B
- Identify how much it would cost
- Illustrate the degree of change focusing on Budgets, Acceptable-Change Steps, Business & Operational Processes, Risk, etc that each step will require
- Identify “Throw Away” tactical elements needed for step-change
- Assist in assessing where “good-enough” lies in determining the business end-state

Within each of the three categories above there are a number of “Views” to be considered

Business Architecture - Domain, Capability, Information
Enterprise Architecture - Conceptual, Logical, Physical
EA Areas of Focus - Integration, Data, Security, etc

Enterprise Architecture - Conceptual View

The intent of a conceptual architecture view is to direct attention at a layer of abstraction of the business without delving into the details of applications or meaning of data transfer. Key elements, components & constructs are identified, including significant architectural elements such as components and relationships among them, as well as architectural mechanisms. Architectural mechanisms are designed to address cross-cutting concerns (i.e., those not localized within a single component).

By focusing on key constructs and abstractions rather than a proliferation of technical details, conceptual architecture provides a useful vehicle for communicating the architecture to non-technical audiences, such as management, marketing, and in some cases users. It is also the starting point for Logical Architecture, which elaborates the component specifications and architectural mechanisms to make the architecture precise and actionable.

The conceptual architecture diagram identifies the people/roles, components and general-connections between components, and the accompanying descriptions document the responsibilities of each component. Structural choices are driven by tradeoffs among interacting or even conflicting component properties or qualities, and the rationale section articulates and documents this connection between the architectural requirements and the structures (components and connectors, or mechanisms) in the architecture.

Enterprise Architecture - Logical View

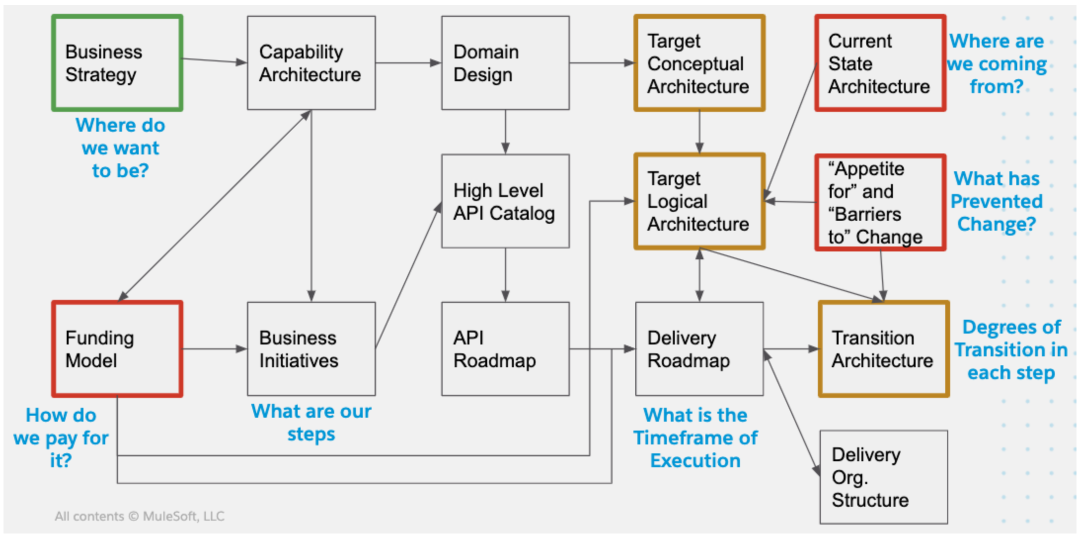
The intent of the logical architecture is to elaborate the conceptual architecture, adding precision through more detailed definition of key architectural constructs, for example, adding interface flow detail for every architectural component.

Logical architecture provides a useful vehicle for communicating the architecture to business and technical audiences. The component specifications act as end-state-vision contracts, allowing for independent development and use of components. This is important for larger-scale projects, allowing teams to work in relative independence. A well-specified logical architecture is especially important when the architecture is intended to support multiple systems, as in the case of product families (product development) and solutions that will be customized for different business units (IT) or markets (consulting).

The logical architecture diagram identifies the application components and their interfaces, and the accompanying specifications define the responsibilities of each component. Again, structural choices are driven by tradeoffs among interacting or even conflicting system properties or qualities, and the rationale section articulates and documents this connection between the architectural requirements and the structures (components and connectors, or mechanisms) in the architecture.

Getting Started

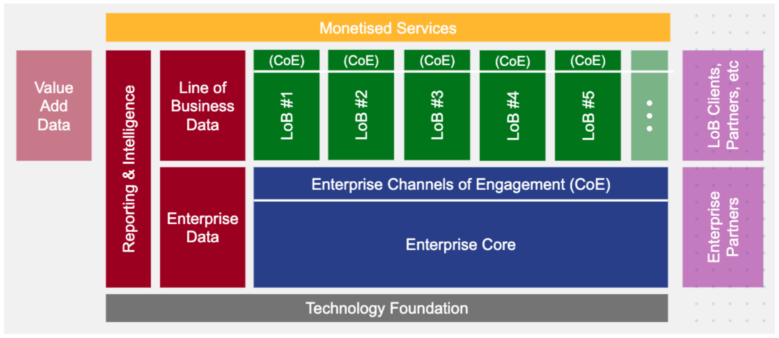
The following process map shows a recommended path to follow in order to reach a “To-Be” (or Target) architecture.


Business Strategy
Where do we want to be?
Capability Architecture
Domain Design
Target Conceptual Architecture
Current State Architecture
Where are we coming from?
High Level API Catalog
Target Logical Architecture
“Appetite for” and “Barriers to” Change
What has Prevented Change?
Funding Model
How do we pay for it?
Business Initiatives
What are our steps
API Roadmap
Delivery Roadmap
What is the Timeframe of Execution
Transition Architecture
Degrees of Transition in each step
Delivery Org. Structure
All contents © MuleSoft, LLC

A “To-Be” (Target) Logical Architecture view is Driven by Business Strategy and Business Initiatives, however it may also be constrained by current state, organisational dynamics (e.g. culture) and most of all available budget.

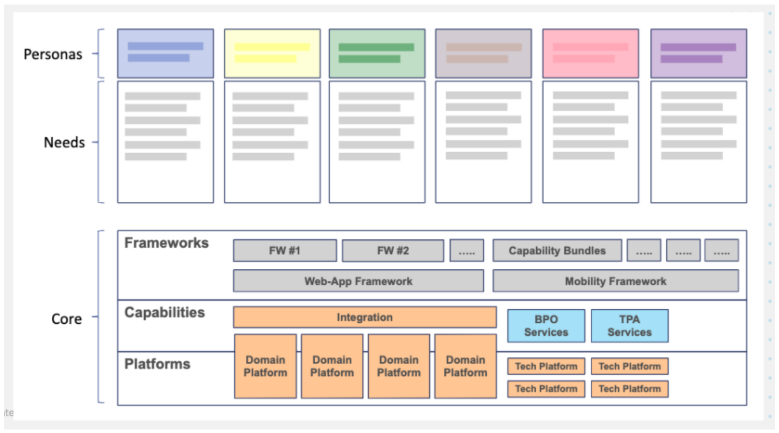
The person/team building out the “To-Be” state should be aware of the strategy, constraints and dependencies before commencing the conceptual model which leads into the logical model.

There is also a need to determine which reference architecture model is best suited to the organisation. Two example models are illustrated below.



Enterprise Platform Model

The **Enterprise Platform Model** looks to consolidate common technologies in an enterprise layer allowing only Line of Business technologies to be managed exclusively by the LoB’s. This pattern works well for organisations that have a high degree of capability commonality across the LoB's but is not useful in organisations where the LoB's are in differing business verticals as the only commonality is generally HR, Finance, Security, Email.



Persona Model

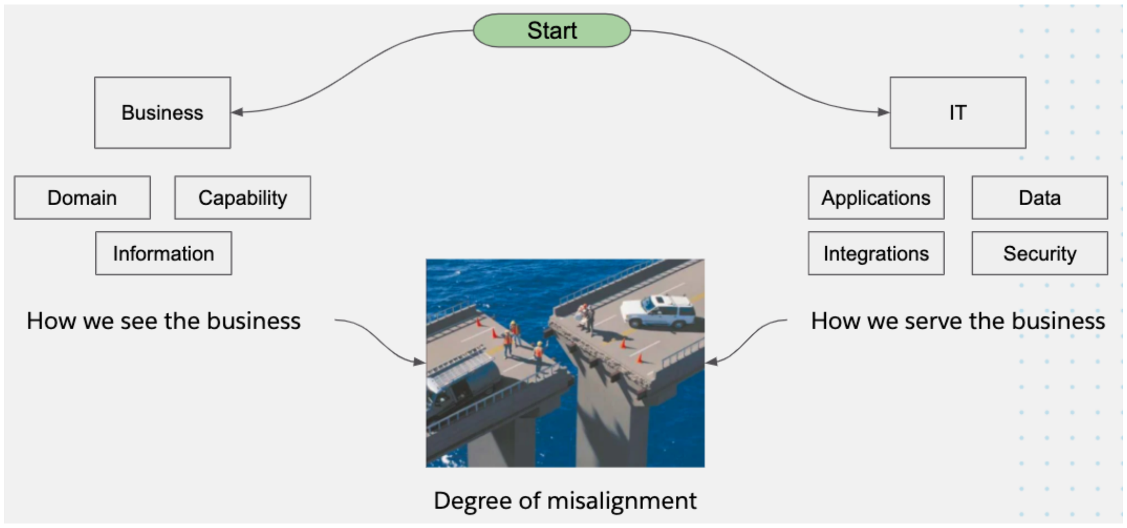
The Persona Model is becoming more popular and is used in companies such as AirBnB and Uber. It has a base layer of commonality, but the pillars are focused on the persona's of the business entity model. Generally in these situations there are separate teams focused on each persona experience and they can do this independently apart from Business Strategy/Initiative driven touch points.

These are just an example of the models you could use, there are a few more and also many mash-ups of approaches. The Model you chose as a focal point for your architecture target state should resonate with a company's culture, organisation, ability to change (if necessary) and budgetary considerations. There is no right model for an organisation but there are models which can deliver more efficiently that others. Also you must be 100% clear that the model you choose has the backing of business and IT leadership.

Know where you are starting from - The "As-Is" (Current) State

The "As-Is" (Current) state is Documented not Designed.

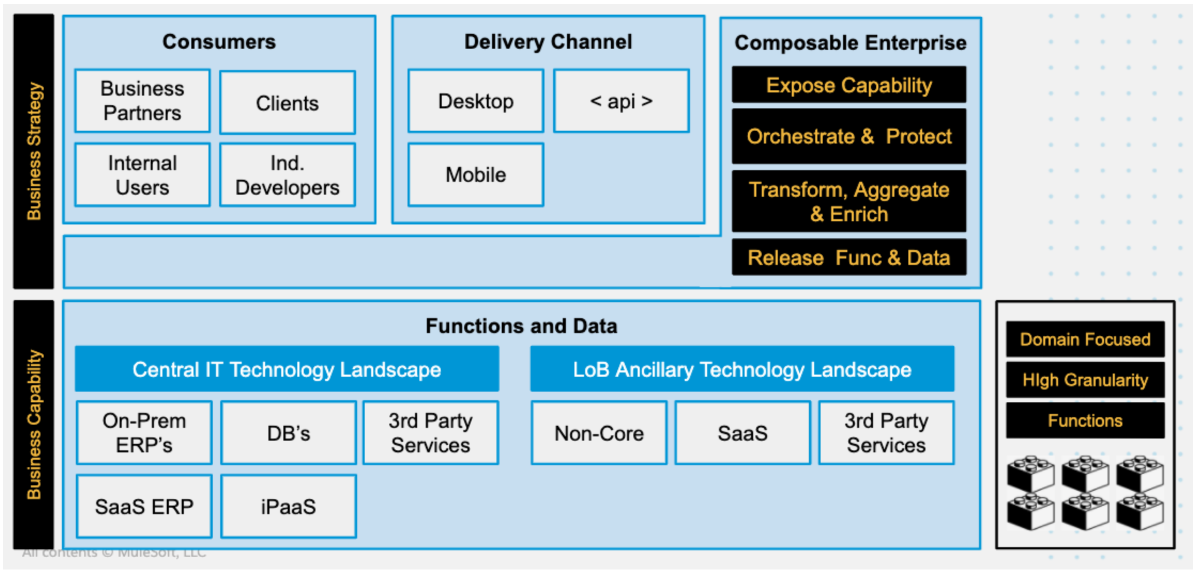
When documenting an "As-Is" architecture it is advisable to approach from both the business and Enterprise perspectives. By doing this and then comparing the results you will easily see the degree of misalignment that has evolved and also the degree of technical debt, shadow IT, solution fitness for purpose, etc.



The artefacts produced to describe the “As-Is” state should be fit for purpose and nothing more. They do not have to be beautified and endlessly cross referenced. They help to figure out “where we are now” and as such are a backward looking set of artifacts with a limited shelf life.

The “To-Be” (Target) state is Influenced by Business Strategy

When building a “To-Be” architecture it must be driven Top-Down. The future state should be conceived as a result of the Business Strategy and Business Objectives. The “To-Be” architecture is in effect “Business Driven”.



(Enterprise) Architecture has a number of core tenets that should be followed in order to provide true value to the business.

- Architecture Decisions are Driven by Business Strategy and the Business Capabilities to be deployed as part of that strategy.

- The “To-Be” (Target) Architecture State is a vision of a point in time, it will evolve based on changes in business strategy, external environment (think Covid-19), cost, technology innovation, etc.

- The “To-Be” (Target) Architecture State must be achievable based on the following limitations, constraints, considerations;
    - Budgets
    - Organisation Culture
    - Channels of Consumption (Users, Clients, Partners, etc)
    - Channels of Delivery (Web, Mobile, Social, API, etc)
    - Capacity for change within the organisation
    - Marketplace
    - Regulatory Conditions

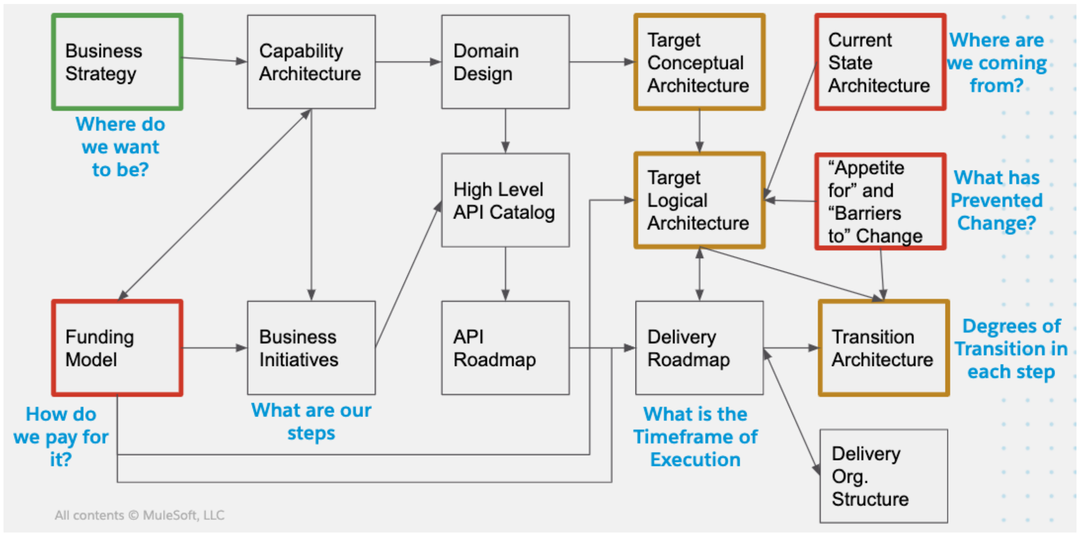
- The “To-Be” (Target) Architecture State must take into account the following aspects of Data Management;
    - Data Sovereignty - Generally dictated by governments
    - Data Residency - Generally dictated by customers and/or the organisation
    - Data Classification - PII, PCI, PHI
    - Data Longevity
    - Data Ownership
    - Data Privacy
    - Data Use

- The “To-Be” (Target) architecture state must be deliverable via a series of transition architecture states

Building a “To-Be” (Target) Architecture State is not about envisaging a perfect end-state. It is about defining a model aligned with the business strategy that can be delivered within the constraints of budget, culture, etc and most importantly can adapt to the changing needs of the business.

Logical Architecture Inputs

Going back to our “Process Map” for building a logical architecture there are a number of key inputs.


Business Strategy
Where do we want to be?
Capability Architecture
Domain Design
Target Conceptual Architecture
Current State Architecture
Where are we coming from?
High Level API Catalog
Target Logical Architecture
“Appetite for” and “Barriers to” Change
What has Prevented Change?
Funding Model
How do we pay for it?
Business Initiatives
What are our steps
API Roadmap
Delivery Roadmap
What is the Timeframe of Execution
Transition Architecture
Degrees of Transition in each step
Delivery Org. Structure
All contents © MuleSoft, LLC

The directional input to a Logical architecture are;

- Business Strategy
- Capability Architecture
- Domain Map
- Conceptual Architecture

The constraints input to a Logical architecture are;

- “As-Is” (Current) State
- Capability / Barriers to change
- Budgets

Logical Architecture Outputs

We live in a world where tangibles—like deliverables—are demanded, but intangible outcomes are what fundamentally matter. Of course, architecture deliverables are essential to important outcomes. But the point is this: we need to think about what outcomes we want to achieve, so that we can shape expectations as to what deliverables we should produce. This allows us to focus on deliverables that will actually make a difference in achieving our desired outcomes.

The key questions your “To Be” (Target) Logical Architecture outputs need to answer are;

- How is the business strategy will be supported via technology
- How will it be delivered

- What is the transition timeframe from “as-is” to “to-be”
- Is it extensible to changing needs
- How is current technical debt addressed
- What are the tactical or throwaway components required
- What organisational changes are required to reach the “to-be” state
- Cost of the transition

Audience

We need to consider the concerns of our primary stakeholders and tailor views that address specific concerns. We also need to tailor communication formats to match the communication styles and needs of the stakeholders. In the sections below, we cover deliverables that are generally useful.

Documents

Executive Briefing: Architecture Strategy. The audience for this document is primarily executive management. The briefing covers how we will deliver on the business and product strategy producing differentiating capabilities or features. It also identifies concerns such as tough issues faced on similar past projects, or new risks accompanying new technical, market or organizational opportunities and directions, and articulates how we will address them.

Technical Briefing: The audience for this document is primarily the IT community, and technical project management. The goal of the technical briefing is to provide just enough information to get the technology landscape view and essential architectural strategies across. Make the briefing interesting to the technologist. Identify the key tough problems the architecture addresses. Outline our technical approach to addressing these. Refer to white papers for details on technical approaches.

Architecture Requirements Overview. This contains all requirements models, descriptions, notes, etc. which fully document and explain all decisions regarding technology landscape, scope, architecture objectives, priorities, and architecturally significant requirements including functional requirements and characterizations of required components.

“To-Be” State.

“Transition” States.

Decision Log. The Decision Log is your complete record of architecture decisions. It covers how we have organized our technology landscape at a conceptual and logical level: what are the components, what are their relationships and externally visible properties?

White Papers. The audience is both the business and technical community, and the purpose is to explain and achieve buy-in. Write or obtain publically available write white papers motivating and explaining significant (sets of) decisions, such as the conceptual design of architectural mechanisms or key aspects of how the technology landscape delivers to the business strategy.

Presentations

Documents provide a record and a reference, but presentations are essential to getting the architecture communicated and “sold.” Use every chance, formal and informal, to present key aspects of the architecture to targeted audiences. And encourage others to do so. The more freely you allow others in your organization to present “your” architecture ideas, the more exposure those ideas will get. If others take ownership for those ideas, they are fully bought into them. It is more helpful to see that as an achievement than as a threat.

Outcomes

A key outcome is confidence that "this can be done." We know where we are headed, and we have confidence we can get there. Specifically, the architect (or team of architects) is confident that there is good understanding of what capabilities the system needs to have, and these capabilities can be built in the planned timeframe with the allocated resources. Moreover, the architect has shared this path with management and key influencers in the development community, and gained their confidence by showing the approach that will be taken to build these capabilities, and articulating believable strategies to address the "gnarly" issues that beset such systems.