







Quick Technical Architecture Method - QTAM

Table of Contents

 Change Log	1
 Introduction to the Method	1
 Actions	2
 Design the Architecture Vision	2
 Design the Business Architecture	4
 Design the Information Systems Architecture	6
 Design the Technology Architecture	8
 Consolidate the Work Packages	10
 Design the course of actions	11
 References	12
 License	13



Change Log

When?	Who?	What?
2025-06-23	Thibault Morin	Initial version of the Quick Technical Architecture Method (QTAM).



Introduction to the Method

The Quick Technical Architecture Method (QTAM) is a streamlined approach to technical architecture. It is designed to help practitioners quickly analyze and design technical architectures in a way that is both efficient and effective.







The method is based on the TOGAF standard and, optional but recommend, the ArchiMate modeling language. It provides a set of actions that can be performed to analyze and design technical architectures.

The method is tailored for practitioners who have an entry-level understanding of the Business Architecture domain and a good understanding of the Information Systems and Technology Architecture domains. That means, professionals with a string technical background, such as software engineers, system architects, or IT consultants, who are looking to quickly analyze and design technical architectures.

The method doesn't require any specific tool or software. It can be performed using any text editor,

diagramming tool, or modeling software.

The actions are:

1.  Design the Architecture Vision
2.  Design the Business Architecture
3.  Design the Information Systems Architecture
4.  Design the Technology Architecture
5.  Consolidate the Work Packages
6.  Design the Course of Actions

The practitioner shall start with the first action and then proceed to the next one. The actions are designed to be performed in a sequential manner. However, due to the iterative nature of architecture work, the practitioner may need to revisit previous actions as new information becomes available or as the architecture evolves.

The main deliverables of the method are a set of views that inform stakeholders about the work to be done and a course of action that outlines how to achieve the desired architecture.

To provide the main deliverables, the practitioner must design intermediate views. Despite their significance to the architecture work, these views are not considered deliverables. They simply serve as the rationale leading up to the main deliverables.

Actions

Design the Architecture Vision

What are we doing?

We ensure that the expected outcome of the architecture work is clearly understood. This means everyone involved knows what success looks like before the work begins.

We also confirm that the scope of work required for the analysis is approved. Without this agreement, we risk wasting time on unclear or misaligned expectations.

Who is participating it?

We, as practitioners, work together with the stakeholders involved in the architecture process. The action is mainly performed by the architecture practitioner.

Stakeholders are important to confirm a shared understanding of the scope and objectives. Moreover, their involvement ensures that the work stays relevant and grounded.

Why are we doing it?

We do this to gather upfront as much information as possible about the context of the architecture work.

When we have this clarity from the start, it helps us reduce misunderstandings, align with stakeholder expectations, and define boundaries that are both realistic and achievable.

How do we do it?

The action is composed of the following steps:

1. Design a view to capture the context of the architecture work: Motivation Viewpoint (ArchiMate) or similar approach.
2. Optionally, design a view to highlight the outcomes of the architecture work: Strategy viewpoint (ArchiMate) or similar approach.
3. Optional, design a view that describes what will exist once the architecture is implemented: Solution Concept diagram (TOGAF) or similar approach.
4. Document the main objective of the architectural work.

Motivation Viewpoint (ArchiMate)

The motivation viewpoint allows the designer or analyst to model the motivation aspect, without focusing on certain elements within this aspect. For example, this viewpoint can be used to present a complete or partial overview of the motivation aspect by relating stakeholders, their primary goals, the principles that are applied, and the main requirements on services, processes, applications, and objects.

— ArchiMate® 3.2 Specification: [Motivation Viewpoint](#)

The motivation viewpoint is used to capture the context of the architecture work. It helps to:

1. Identify the stakeholders and their concerns.
2. Identify the internal/external drivers like requirements, constraints and/or principles associated to their concerns.
3. Identify the goals/outcomes from the stakeholder point of views.

Strategy viewpoint (ArchiMate)

The strategy viewpoint allows the Business Architect to model a high-level, strategic overview of the strategies (courses of action) of the enterprise, the capabilities, value streams, and resources supporting those, and the envisaged outcomes.

— ArchiMate® 3.2 Specification: [Strategy Viewpoint](#)

Solution Concept diagram (TOGAF)

A Solution Concept diagram provides a high-level orientation of the solution that is envisaged in order to meet the objectives of the architecture engagement. In contrast to the more formal and detailed architecture diagrams developed in the following phases, the solution concept represents a “pencil sketch” of the expected solution at the outset of the engagement.

This diagram may embody key objectives, requirements, and constraints for the engagement and also highlight work areas to be investigated in more detail with formal architecture modeling.

Its purpose is to quickly on-board and align stakeholders for a particular change initiative, so that all participants understand what the architecture engagement is seeking to achieve and how it is expected that a particular solution approach will meet the needs of the enterprise.

— TOGAF® Standard — Architecture: [3.6.3.3. Solution Concept Diagram](#)

Objective of the architectural work

The objective of the architectural work should be SMART:

- S – Specific: The objective should be clear and unambiguous, detailing exactly what is to be accomplished.
- M – Measurable: There should be criteria for measuring progress and success. It answers the question: *How will you know when it's done?*
- A – Achievable: The objective should be realistic, considering available resources and constraints.
- R – Relevant: The objective should align with stakeholder expectations and practitioner(s) skills.
- T – Time-bound: The objective should have a defined timeline or deadline to create urgency and focus.

Design the Business Architecture

What are we doing?

We're creating a small set of views that describe the business landscape. This could include the current state (also called the Baseline), the future state (also called the Target), or both. We use these views to understand what needs to change and how that change affects the business.

Who is participating it?

Mostly the practitioner, but we involve stakeholders when we need to assess alternatives or

validate the business impacts. Their insight is crucial to avoid blind spots.

Why are we doing it?

Because architecture changes don't happen in a vacuum. They impact how value is delivered, who is responsible for what, and how the business operates.

By understanding the business side, we make sure our technical decisions stay relevant and useful.

How do we do it?

The action is composed of the following steps:

1. Design a view to capture the business concepts and the underlying wording : Information Map (TOGAF) or similar approach.
2. Optionally, design a view to capture the organization structure: Organization Viewpoint (ArchiMate) or similar approach.
3. Optionally, design a view to capture interactions, behaviors, and business rules about digital products and services: Product Viewpoint (ArchiMate) or similar approach.
4. Document the gaps between the starting point (BASELINE) and the target state (TARGET).

Information Map (TOGAF)

A collection of information concepts and their relationships to one another. Information concepts, in effect, reflect the business vocabulary; e.g., client, account, or product. Mapping information in the Business Architecture starts with listing those elements that matter most to the business as well as how they are described in business terms.

— *TOGAF® Standard — Architecture: [3.6.4.29. Information Map](#)*

The information mapping technique is well described in the TOGAF series guide: [Information Mapping](#).

Organization Viewpoint (ArchiMate)

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.

— *ArchiMate® 3.2 Specification: [Organization Viewpoint](#)*

Product Viewpoint (ArchiMate)

The product viewpoint depicts the value that these products offer to the customers or other external parties involved. It shows the composition of one or more products in terms of the constituting (business, application, or technology) services, and the associated contract(s) or other agreements. It may also be used to show the interfaces (channels) through which this product is offered, and the events associated with the product. A product viewpoint is typically used in product development to design a product by composing existing services or by identifying which new services have to be created for this product, given the value a customer expects from it. It may then serve as input for business process architects and others that need to design the processes and ICT realizing these products.

— ArchiMate® 3.2 Specification: [Product Viewpoint](#)

Design the Information Systems Architecture

What are we doing?

We're designing a small set of views to describe the Information Systems Architecture—that means we capture how systems interact and how data flows. This includes the baseline situation, the target scenario, or both.

Who is participating it?

This is mainly done by us, the practitioners.

But we should involve stakeholders, especially when evaluating alternatives. Their feedback ensures our assumptions are grounded and our designs are aligned with expectations.

Why are we doing it?

We do this for two reasons:

- To understand the impact on the information systems.
- To confirm that changes in other architecture domains—like business or technology—are actually relevant to the stakeholders.

This is not just about documenting apps and data. It's about making sure the systems architecture supports the change we're trying to achieve.

How do we do it?

The action is composed of the following steps:

1. Design a view to capture structure and relationships between applications: Application

Structure Viewpoint (ArchiMate), C4 Model diagrams, or similar approaches.

2. Design a view to capture the data structures and their relationships: Information Structure Viewpoint (ArchiMate) or similar approach.
3. Optionally, design a view to capture the relationships between applications and data: Application/Data Matrix (TOGAF) or similar approach.
4. Document the gaps between the starting point (BASELINE) and the target state (TARGET).
5. Manage possible alternatives by documenting their differences, establishing criteria, and conducting a benchmark.

Application Structure Viewpoint (ArchiMate)

The application structure viewpoint shows the structure of one or more applications or components. This viewpoint is useful in designing or understanding the main structure of applications or components and the associated data; e.g., to break down the structure of the system under construction, or to identify legacy application components that are suitable for migration/integration.

—ArchiMate® 3.2 Specification: *Application Structure Viewpoint*

C4 Model diagrams

The C4 model is a framework for visualizing the architecture of software systems. It provides a way to describe the system at different levels of detail, from high-level context diagrams to detailed component diagrams.

More information can be found on the C4 Model website: <https://c4model.com>.

Information Structure Viewpoint (ArchiMate)

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 business processes upon applications, it may be useful to operational managers responsible for these processes.

—ArchiMate® 3.2 Specification: *Information Structure Viewpoint*

Application/Data Matrix (TOGAF)

The purpose of the Application/Data matrix is to depict the relationship between applications (i.e., application components) and the data entities that are accessed and updated by them.

Applications will create, read, update, and delete specific data entities that are associated with them. For example, a Customer Relationship Management (CRM) application will create, read, update, and delete customer entity information.

The data entities in a package/packaged services environment can be classified as master data, reference data, transactional data, content data, and historical data. Applications that operate on the data entities include transactional applications, information management applications, and business warehouse applications.

The mapping of the Application Component-Data Entity relationship is an important step as it enables the following to take place:

- Assign access of data to specific applications in the organization
- Understand the degree of data duplication within different applications, and the scale of the data lifecycle
- Understand where the same data is updated by different applications
- Support the gap analysis and determine whether any of the applications are missing and as a result need to be created

The Application/Data matrix is a two-dimensional table with Logical Application Component on one axis and Data Entity on the other axis.

— *TOGAF® Standard — Architecture: [3.6.5.3. Application/Data Matrix](#)*

Design the Technology Architecture

What are we doing?

We create a few views describing either the BASELINE, the TARGET, or both.

We're creating a few views that describe either the Baseline, the Target, or both. These views allow us to understand how technology and infrastructure will evolve—and ensure those changes support the broader architecture work.

Who is participating it?

Mainly, this action is driven by us, the practitioner. But key stakeholders are also involved, especially when we evaluate alternatives to decide which solution makes the most sense.

Why are we doing it?

We're doing this for two key reasons.

Firstly to understand the impact on technology usage and infrastructure implementation.

Secondly, to confirm that technology changes align with the rest of the architecture—business, data, application—and are meaningful to stakeholders.

How do we do it?

The action is composed of the following steps:

1. Design views to capture the context of the technology changes: Technology Viewpoint (ArchiMate), C4 Model diagrams, or similar approaches.
2. Design views to capture the context of the physical elements: Physical Viewpoint (ArchiMate), C4 Model deployment diagrams, or similar approaches.
3. Document the gaps between the starting point (BASELINE) and the target state (TARGET).
4. Manage possible alternatives by documenting their differences, establishing criteria, and conducting a benchmark.
5. Optionally, design views based on code analysis to highlight changes in the codebases.
6. Optionally, perform quick security-focused analyses such as STRIDE, DREAD, etc.

Technology Viewpoint (ArchiMate)

The technology viewpoint contains the software and hardware technology elements supporting the Application Layer, such as physical devices, networks, or system software (e.g., operating systems, databases, and middleware).

—ArchiMate® 3.2 Specification: *Technology Viewpoint*

C4 Model diagrams

The classical C4 Model diagrams can be used to capture the technology architecture. The C4 model provides a way to describe the system at different levels of detail, from high-level context diagrams to detailed component diagrams.

It can be conflictual with the Information Systems Architecture where the C4 Model diagrams can also be used for modeling. It is because the C4 Model mix both concerns: the information systems architecture and the technology architecture.

More information can be found on the C4 Model website: <https://c4model.com>.

Physical Viewpoint (ArchiMate)

The physical viewpoint contains equipment (one or more physical machines, tools, or instruments) that can create, use, store, move, or transform materials, how the equipment is connected via the distribution network, and what other active elements are assigned to the equipment.

— *ArchiMate® 3.2 Specification: [Physical Viewpoint](#)*

C4 Model deployment diagrams

The C4 Model deployment diagrams can be used to capture the technology architecture or the physical architecture. Again the boundary between both layers Physical and Technology doesn't exist in the C4 Model.

More information can be found on the C4 Model website: [Deployment diagram on c4model.com](#).

Threat Modeling

Threat modeling is an approach to identify and mitigate potential security threats in the architecture. It can be performed using various techniques such as STRIDE or DREAD.

STRIDE is a mnemonic that stands for Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. It helps identify potential threats in the architecture.

DREAD is a risk assessment model that stands for Damage, Reproducibility, Exploitability, Affected Users, and Discoverability. It helps prioritize threats based on their potential impact. However, DREAD is subject to criticism for its subjectivity and lack of standardization.

STRIDE and DREAD are often used in conjunction.



Consolidate the Work Packages

What are we doing?

We're consolidating all the gaps we identified in the previous steps. And for each gap, we define work packages.

A work package is pieces of work needed to address the gap.

We create views, especially a table, that show what needs to be done, and how those tasks relate to one another.

Who is participating it?

Mainly us, the practitioners. But we should definitely bring in stakeholders for feedback—especially when it comes to refining work effort estimates and checking dependencies.

Why are we doing it?

Because, at the end of the day, stakeholders don't care about diagrams—they care about timing. About scope. About delivery.

Work packages are the unit of measurement for architectural impact. They let us turn analysis into action.

Also, when we model these work packages, we often uncover hidden dependencies or risks. That makes this step essential for risk mitigation, too.

How do we do it?

1. List all gaps identified in each previous actions.
2. Group gaps in work packages.
 - a. At the beginning grouping is mostly based on the impacted components.
 - b. Later, grouping is based on the nature of the work.
3. Refine the work packages.
 - a. Identify the actions which are expected to fill the gaps.
 - b. Identify the dependencies between work packages.
 - c. Estimate the work effort for each work package.



Design the course of actions

What are we doing?

We're creating views that explain how to get from where we are, to where we want to be—the Course of Action.

It's a rationalized plan. It connects the work packages we identified earlier into a coherent path toward the target state.

This plan includes estimates, dependencies, and ideally, risks and possible mitigations.

Who is participating it?

Mainly us, again, the practitioners. But you'll need stakeholders for feedback, validation, and sometimes approval.

Why are we doing it?

Because stakeholders need to plan.

They need to assess impact.

And they need to understand the deliverables, timeline, and potential risks of the architecture work.

How do we do it?

1. Identify the phases based on the grouping of work packages.
2. Sort the work packages by priority and dependencies.
3. Design views to show the execution of the course of actions over time: Implementation and Migration Viewpoints (ArchiMate) or similar approaches.
4. Optionally, perform a risk assessment to identify potential risks and mitigations.

Implementation and Migration Viewpoints (ArchiMate)

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, and 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.

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. — *ArchiMate® 3.2 Specification: [Implementation and Migration Viewpoint](#)*



References

- [togaf] The TOGAF® Standard: <https://pubs.opengroup.org/togaf-standard/>
- [archimate32] ArchiMate® 3.2 Specification: <https://pubs.opengroup.org/architecture/archimate32-doc/>
- [c4model] C4 Model: <https://c4model.com/>

License

Quick Technical Architecture Method © 2025 by Thibault Morin is licensed under CC BY-NC 4.0. To view a copy of this license, visit <https://creativecommons.org/licenses/by-nc/4.0/>