Meta-ArchitecturE Document

**Table of Contents**

[How To Use This document Template 3](#_Toc379555488)

[Architectural Vision 3](#_Toc379555489)

[Principles 4](#_Toc379555490)

[Styles 5](#_Toc379555491)

[Patterns & interconnection Mechanisms 5](#_Toc379555492)

[Pattern [NAME] 5](#_Toc379555493)

[Interconnection Mechanism [name] 6](#_Toc379555494)

[Philosophies & Preferences 6](#_Toc379555495)

[Guidelines & policies 6](#_Toc379555496)

[Additional Information 6](#_Toc379555497)

Meta-ArchitecturE document

[TEAM NAME]

# How To Use This document Template

Meta-Architecture as an activity is gathering ideas from other architectures and past experience so that you don’t repeat mistakes and waste time reinventing solutions.

Meta-architecture is the first set of decisions oriented toward addressing system-level concerns. Meta-architecture strongly influences the architecture, but is not the architecture itself. This document should therefore include statements of architectural vision, principles, preferences, styles and patterns of structure and interconnection. Its focus is high-level decisions that will strongly influence the structure of the system; it rules certain structural choices out, and guides selection decisions and tradeoffs among others. These decisions may be expressed as:

* Guiding Principles, Strategies, Styles, Philosophies and Values
* Structural, Behavioral Patterns or Patterns of Interaction: especially patterns that are repeatedly applied to solve some issue, and simplify the architecture
* New system concepts or abstractions
* Best Practices/Lessons Learned
* Thinking about alternative approaches

The level of formalism should suit the project it describes. Making these ideas explicit, rather than leaving them implicit, helps ensure that a team shares a unified vision and goal.

# Architectural Vision

Describe your vision for the architecture here. What is the architecture going to do for your team, or your business?.This should be drawn from your vision for the project.

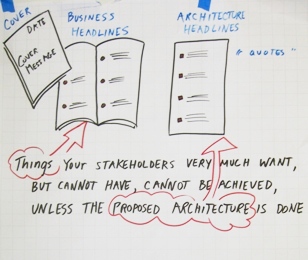


Figure 1. Cover story vision for the project

Figure 1. is used by permission from Bredemeyer Consulting.

# Principles

By “principle” we mean “a behavior or practice that leads to good things”. Example: “Conceptual Integrity can be achieved by uniformly applying a limited number of design forms”.

Guidelines on from for architectural principles:

* Each principle clearly states a fundamental belief of the organization.
* Each principle should be stated in a way that you will know if the architecture has the characteristics expressed by the principle
* Each principle should have a counterargument; that is, they should not be platitudes or general features that are desirable regardless of the system.
* Principles should be simply stated and understandable
* Principles need to be rationalized, stating why the principle is preferred; draw on business-related factors where possible
* The implications of adopting the principle should be identified if possible

You should include one table for each principle. To include new principles, cut and paste copies of the template below.

|  |  |
| --- | --- |
| Principle Name | **Keep it simple** |
| Description | Architectural decisions should make implementation as simple and direct as possible . |
| Rationale/Benefits | We have limited experience and compressed timeframe. Also, simplicity would enable later modifications. |
| Implications | Functionality would limited in the beginning and try to add on later. |
| Counterargument | Could do something cool and unique with custom code everything. |

|  |  |
| --- | --- |
| Principle Name | **Think about scalability specifically multiple clients and group chats.** |
| Description |  |
| Rationale/Benefits |  |
| Implications |  |
| Counterargument |  |

|  |  |
| --- | --- |
| Principle Name | **User friendly** |
| Description |  |
| Rationale/Benefits |  |
| Implications |  |
| Counterargument |  |

# Styles

Here is definition of style that helps distinguish it from Patterns, although at this level there is not much distinction. Style is determined by

* A set of component types that perform some function at runtime
* A topological layout of these components indicating their runtime relationships
* A set of connectors that mediate communication, coordination, or cooperation among components (function call, RPC, data stream, socket, etc.)
* A distinctive name that describes the collection.

We can identify styles by answering the following questions, from Shaw and Garlan:

* What is the design vocabulary—the types of components and connectors? Clients and server/database, REST, Serializable objects
* What are the allowable structural patterns? Three tiered,
* What is the underlying computational model? Client-server, peer-to-peer
* What are the essential invariants of the style? Clients must register to be seen or initiate
* What are common examples of its use? To talk to people
* What are the advantages and disadvantages of using the style?
* What are some common specializations of the style?

# Patterns & interconnection Mechanisms

## Pattern [Three tiered]

### Summary Description –

### Context of USE (INTENT)

### Problem Statement

### Solution Description

### Variants and Related Patterns

### Known Uses -

### Consequences

Rationale:

|  |  |
| --- | --- |
| Class: | Collaborator: |
| Responsibility: |  |

Three types of patterns are of interest:

* Structural Patterns
* Interconnection Patterns
* Decoupling Patterns

These follow directly from describing architecture in terms of a set of components and their relationships.

Patterns are described in terms of a Pattern Form (see below), and one or more CRC “cards” (see below), which describe each of the major pieces of the pattern. Cut and paste copies of the Pattern Form and CRC table (as needed), for each Pattern or Interconnect Mechanism described. You may not think of every pattern the first tie through. You may struggle with rationale. That’s OK. Plan time in your project to revisit this document.

If you are thinking of following a reference architecture, this is the place to describe and consider it.

## Pattern [MVC]

### Summary Description –

### Context of USE (INTENT)

### Problem Statement

### Solution Description

### Variants and Related Patterns

### Known Uses -

### Consequences

Rationale:

|  |  |
| --- | --- |
| Class: | Collaborator: |
| Responsibility: |  |

## Interconnection Mechanism [REST]

This is the place to consider standard interconnection technology and solutions that you may use, such as COM, CORBA, JavaBeans, Mule Enterprise Service Bus, MS Kinect.

# Philosophies & Preferences

Often there are ideas, philosophies or cultures that affect teams and projects. If it doesn’t fit anywhere else, put it here.

# Guidelines & policies

* Here you want to put general guidelines and policies that:
* Might affect the architecture & design of the system
* Don’t fit other places in this document

# Additional Information

Include comments or ideas that you want here, but that don’t fit anywhere else. Keep this section to a minimum, because you want a document people can use easily, not a tome of esoteric lore.