Introducing the Practice Scenarios

We will introduce the three scenarios related to real-life examples. The company that we will talk about is a fictional company, which we will name **ABC Trading**, but it can represent any company that you have or will be working for.

The purpose of the scenarios is to make up simple and generic stories that can fit within your real-life work environment, even if your organization is working in a different business domain. The scenarios will be detailed enough and as close as possible to problems that you have faced or will face as an enterprise architect. The goal here is to introduce our proposed agile **Enterprise Architecture** (**EA**) approach for implementing EA.

The three scenarios will address three different aspects of the enterprise that concern different stakeholders, and each requires a different set of EA artifacts:

- *First scenario:* Describes an **application architecture** project where your contribution is required to help the **solutions architect** in designing and documenting a new mobile application
- *Second scenario:* Describes a **technology architecture** project where the **Chief Technology Officer** (**CTO**) needs to reduce the IT costs and needs your help in articulating some decision-supporting information
- *Third scenario:* Describes a **business architecture** project where the **Chief Executive Officer** (**CEO**) needs your contribution to a plan for introducing a new service to customers

Before moving ahead into the details, let's first learn more about ABC.

A brief on ABC Trading

Today, all companies have business, application, and technology requirements and there are always projects here and there to address these requirements. You have been hired by ABC Trading to establish an **Enterprise Architecture Office** (**EAO**). Your goals are to establish the foundations of the EAO, create tangible value, and gain the trust of the enterprise stakeholders. You need to make the EAO a trustworthy consulting reference for all types of projects within the organization.

ABC Trading is a mid-sized wholesale network that has been in business for 50 years. They distribute their goods through a network of retail partners, and they do not have their own retail stores. However, they do have showrooms in each warehouse to showcase their goods and provide samples to retailers. ABC Trading delivers goods from their warehouses to retailers' warehouses through a fleet of trucks owned by the company.

ABC Trading has about 1,000 employees working at different business units. The IT function at ABC Trading is just enough to get the job done, but they do not have fancy up-to-date solutions and technologies. They have a mixture of a few remaining old mainframe, client-server, and web-based applications, all hosted in a centralized data center owned and managed by the company.

There was a previous attempt to establish an EAO a few years ago, but the management decided to stop the project due to low return on investment. Sparx was purchased long ago but no one really used it or got tangible value out of it. There are several licenses available and one of them has been given to you. There will be more architects to join the EAO once established but unfortunately, you will start by yourself. You will be reporting to the **Chief Information Officer** (**CIO**) who has direct contact with all other C-level executives.

The three projects that were introduced take place at three different business units and every **Product Owner** wants to get the job done as soon as possible so they can move on to their next assignments. They are not willing to compromise the quality of work for sure, but you must keep in mind that they have no time to listen to EA speeches or to any topic they do not have an interest in. They might have already done that a few years ago when the EAO was first established and all that they remember are presentations they never understood and terminologies they never heard before. Expect their first question to you to be *what's in it for me?*, which is also known by the short abbreviation **WIIFM**. The situation is challenging, but this is your chance to show what a practical enterprise architect can do.

For each of the scenarios, we will do the following:

1. Provide the background and introduce the problem.
2. Introduce the stakeholders of each.
3. Convert the requirements into agile user stories.
4. Identify the tasks for each user story at a high level.
5. Add the story to the artifacts backlog.

You can use any software to manage the backlog and the user stories, such as **Jira**, **Microsoft Planner**, or even Sparx itself, we will use a simple bulleted list.

Now let us explore the first scenario and extract the user stories and tasks from it.

First scenario – application architecture

The first project that you will be involved in is the development of a new mobile application that can enable tracking shipments from the time they leave ABC Trading warehouses to the time they get delivered to retailers' warehouses. It will be a mobile application that truck drivers will install on company-provided smartphones that gives the location of the truck at any moment in time.

The reason for choosing an application architecture example is because many business solutions today involve the use of business applications. Whether your organization prefers to build or buy solutions, knowing how to define an application architecture and document it, will be required in all cases.

We will describe the scenario in more detail by putting more context around it, then we will extract the user stories and tasks to build the artifacts backlog. Let's start by looking more closely at the problem.

Scenario description

Today, ABC Trading retail partners use a web application on the ABC Trading website known as the *Trading Web* to check items' inventory and place orders online. Each retailer uses a user ID and a password for authentication before they can place orders. Payments can be made online as well as by checks in the mail. Retailers can check their orders' status to know if they have been processed, shipped, or are having issues, but they cannot know the location of their shipment at a point in time. For some retailers, shipments can take days and possibly weeks before arriving, depending on the distance, traffic, weather conditions, and many other variable factors, which makes it difficult for them to give accurate delivery dates to their customers.

Within the scope of this phase of the project, the targeted mobile application – which we will name the *Tracking App* – is supposed to provide the shipment tracking functionality only. However, it will be good to see other features being migrated from other ABC Trading applications to the mobile application in upcoming phases in the future but that requires different scoping.

The Trading Web application is aging in terms of technology, but it is still doing the essential parts of serving the business in terms of functionality and performance, so ABC Trading has no real requirement to replace it in the near future. The Trading Web application has some modern architecture patterns in it such as modularization and **Application Programming Interfaces** (**APIs**) for providing or receiving information. It is not comparable to modern shopping websites in terms of technologies and architecture, but it is not as obsolete as a monolith mainframe system.

Now we have sufficient information about the problem, we will extract the user stories from it.

Artifacts backlog

Your task as an enterprise architect is to write an **Application Architecture Document** (**AAD**) in which you will describe the targeted Tracking App in terms of high-level designs. This document will be used to guide the design and development of the application either internally or externally, so its content needs to be ready to be part of a **Request for Proposal** (**RFP**) or similar software procurement documents.

Our targeted audience of the AAD is the solutions architect who will use it as a high-level design and requirements document, to guide and influence the way they will design and develop the application. There is no solid line that defines where the enterprise architect role stops and where the solutions architect role starts as they overlap in many areas. But in general, the design details, the development of programming user stories, system requirements, test cases, and development schedule will be taken care of by the solutions architect and the development team so the AAD needs to stay at a conceptual-to-logical level of detail that is equivalent to **business use cases**. The AAD needs to contain at least the following sections:

- A definition of the business requirements that translates the problem description section into formal requirements
- A definition of what the target application shall do in terms of services and functionalities that users and other applications can use
- A definition of integration requirements with the existing systems

- A definition of the data that will be used in or produced by the application
- A definition of the required infrastructure technologies that will support the application

Let's see how many user stories and tasks we can extract from the previous requirements.

User story 1

To start with, we need to restate the problem definition in the form of a user story or stories. Our target audience (stakeholder) is a solutions architect, so the user story can be stated and described as follows:

As a solutions architect, I need to have an AAD describing the targeted Tracking App at a conceptual level that identifies the **Requirements**, **Application Services**, **Application Interfaces**, **Data Objects**, and **Technology Services**, so that I can build the Tracking App mobile application to realize the identified requirements and design principles and constraints.

For now, let's build the artifacts backlog:

1. First, we need to provide a brief description of the Tracking App and who will be using it, which means that our first task is developing the **Application Component** context diagram.
2. After that, we need to build the requirements list and bear in mind any existing design principles and constraints that can influence the application, which means we need to define the **Application Requirements** catalog.

Then, we need to describe what this application component will provide to its context, and what it will receive from or provide to other applications. This can be translated into the following three tasks:

1. Define the application services catalog for the Tracking App.
2. Define the data objects catalog for incoming and outgoing data.
3. Define the application interfaces in which integrations with other applications will take place.

Finally, we need to describe what technology (or infrastructure) services are required to run and operate the application. Remember that we do not have to name the actual products that will provide the technology services at this level.

4. We only need to identify what services are needed, so in other words, we need to define the technology services catalog for the Tracking App.

Since our EA repository in Sparx is a **green field**, which means there are no formal guidelines that we need to follow when developing our artifacts, we need to build these guidelines as we build and develop the catalogs. Please remember that every artifact needs to be based on a metamodel to tell the architects what elements and what relationships can be included. It will help in maintaining the integrity and consistency of your EA repository. With that said, we need to develop six focused metamodels, so each will have a task under the user story:

1. Build the application component focused metamodel.
2. Build the application requirements focused metamodel.
3. Build the application services focused metamodel.
4. Build the application interfaces focused metamodel.
5. Build the data objects focused metamodels.
6. Build the technology services focused metamodels.

Based on the size of the organization that you work for in real life and the complexity of its systems, fitting each of the preceding stories into a two-week period may not be possible, so you would break them down into smaller stories, but for the sake of the example, we will keep them as they are.

Remember that we are just beginning to build our EA repository so it is empty and we need to define the processes that are required for creating, modifying, and deleting artifacts, and this will be our second user story.

User story 2

The other thing that we need to define and build within our repository is the operating processes of the repository itself. We do not have to be sophisticated and detailed at the beginning because this may result in the scope. At the same time, we need to put in place some guidelines and documented processes of how to do things right.

Since the operating processes will be used across all the user stories that we have defined and will be defining in the future, it is a good idea to have a separate user story for each of the operating processes with a separate set of tasks.

As a lead enterprise architect, I need to define the proper guidelines and processes for creating, modifying, and deleting artifacts from the EA repository, so that every other contributing architect, including myself, can follow them to maintain the integrity and consistency of all EA artifacts.

The user story has already identified three of the tasks that need to be added to the backlog:

1. Define the process of EA artifact creation.
2. Define the process of EA artifact modification.
3. Define the process of EA artifact deletion.

Note that the term *artifact* is not limited to the diagrams only but includes the metamodels and the operating processes that are created as part of architectural work as well.

We have defined two user stories for the application development scenario, and it could be any application development project. Let's move ahead to the second scenario and create more user stories and tasks out of it.

Second scenario – technology architecture

With a tough year like 2020, ABC Trading is planning for a 20% budget cut, and many aspects of the organization, including the IT expenditures, must be re-evaluated and reduced to meet the new budget. IT expenditures are categorized into two main categories: **Capital Expenses** (**CapEx**) and **Operational Expenses** (**OpEx**). CapEx are the expenses on the purchase and acquisition of new assets such as new hardware, software, or owned facilities. OpEx on the other hand are the periodical and continuous expenses that occur as a result of operating IT resources such as human resource salaries, subscription-based software and infrastructure, utility bills, rented facilities, and other similar types of *expenses*.

This example has been chosen to illustrate a simple example of changes that can occur at the **technology layer**, which in your case could be a cloud migration project, an upgrade to the existing technology stack, or the shutting down of an entire facility. The artifacts might differ slightly based on requirements but the idea is the same.

Scenario description

The CTO has decided to analyze the existing IT assets, including all the software and hardware components, and identify whether there is any possibility to replace, merge, eliminate, or reduce the number of licenses of each in order to meet the cost reduction targets. Components that provide similar functionalities need to be identified and evaluated for possible elimination or merging. Components that can be replaced by other components performing the same functionality but at a lower price need also to be identified. Deprecating unused or rarely used components can also reduce the OpEx, and replacing some of the on-premises components with the equivalent cloud components is an additional option to consider.

The CTO would also like to have a dependency report showing which application uses which IT asset to help in analyzing the impact of making any changes to the existing technology stack.

Note

Please remember that this scenario has been created to demonstrate how to create EA artifacts in Sparx and is not meant to be used as a guideline to reduce IT budgets in any way. There could be better and more efficient ways to reduce IT budgets than the ones used in the example.

Now we have our problem defined and our requirements stated, our next step is to extract more user stories out of them and add them to the backlog.

Artifacts backlog

The CTO is looking for a document addressing the following list of requirements:

- Identify the existing IT assets including hardware and software components.

- Identify the services provided by each of the components and find any duplicated services.
- Identify the dependencies of applications on the existing IT assets to estimate the impact of changing, merging, or replacing any of them.
- Provide a **To-Be technology architecture** showing the proposed changes.
- Provide a plan for implementing the proposed changes.

From the preceding requirements, we can see that two user stories can be defined: an **As-Is technology architecture**, and a **To-Be technology architecture** with a list of **gaps** and a **roadmap**. The gaps will identify what is missing between the As-Is and the To-Be architectures, while the roadmap will identify the set of initiatives and projects that will be needed to close these gaps.

Note

Gaps do not always indicate something necessarily is missing from the As-Is that needs to be added to the To-Be. Gaps indicate differences between two states of the architecture regardless of which state has more elements than the other. If the As-Is has more components than the To-Be, that is still considered a gap even though we need to remove elements to bridge it.

We already have two user stories in our backlog so the next will be user story 3.

User story 3

The third user story is an **As-Is technology architecture**.

As CTO, I need to have a list of all the technology assets in the enterprise, know what they provide, and identify what application uses which asset so that I can identify the redundant services and know the impact of replacing or retiring some of these assets.

To start with, we need to translate the CTO terminology into the equivalent EA ones. Lists are **catalogs**, and the technology assets are the hardware and software that support the enterprise, so architecturally they are called **technology nodes**. These technology nodes provide **technology services** to the enterprise including the **application components** that depend on them.

Note

It is very important for an enterprise architect to listen to stakeholders' requirements in their own language and to not attempt to enforce EA acronyms on them. It is your job to make that translation, not theirs, because you are the one bridging the communication gaps across the enterprise, not creating new gaps.

Having the translation part done, the third user story will have the following tasks:

1. Define the technology nodes catalog.
2. Define the technology services catalog.
3. Additionally, we need to show the relationship between these two catalogs, so we will define the technology-nodes-to-technology-services matrix.
4. We also need to define the relationships between application components and technology nodes, so we need to define the application components catalog.
5. Then we need to define the application-components-to-technology-services matrix to build the relationship between the two catalogs.

e need to build the metamodels that guide the development of each artifact. Therefore, we need three focused metamodels, one for each:

1. Build the technology node focused metamodel.
2. Build the technology service focused metamodel.
3. Build the application component focused metamodel.

Notice that *user story 1* has a task for building the technology service focused metamodel so most probably this is a duplicate of the same task. Let's keep both in the backlog for now and decide which one to remove later.

User story 3 looks good now with sufficient details and tasks to start working on it, so let's move on to the next user story.

User story 4

The fourth user story is about defining some To-Be artifacts describing the targeted technology architecture with the proposed changes and a plan for implementing them.

As CTO, I need to have a list of the duplicated services and those services no longer being used, along with the possible impact of deprecating, merging, or replacing the technology assets that provide them, so that I can create an action plan for implementing the changes.

Since we have already developed the technology services and technology nodes catalogs in the previous user story, we need to utilize them in this user story to derive smaller but more specific catalogs out of them. The duplicated and unused technology services catalogs are subsets of the main technology services catalog:

1. Define the duplicated technology services catalog.
2. Define the unused technology services catalog.

Then we need to define what it looks like if we remove, merge, or replace these services.

3. Define the To-Be technology services catalog.
4. Define the To-Be technology nodes catalog.

Define the **To-Be application components** to **technology services** matrix so we know the impact of making these changes on the existing applications' gaps between them and build a plan for bridging them. Therefore, we need to add two more tasks to the user story.

5. Define the **technology gaps** catalog.
6. Define the **work packages** catalog.

And of course, define the mapping between the gaps and the work packages so we need the following.

7. Define the **work packages** to **technology gaps** matrix.

Think of work packages as a group of actions that need to be addressed to resolve a common problem. They can be whole projects or smaller components of projects. Since we already have tasks for building the technology nodes- and technology services focused metamodels, we do not have to repeat them here. We will need, however, to build the focused metamodels for the gaps and the work packages:

8. Build the gap focused metamodel.
9. Build the work package focused metamodel.

Do not forget that we need to define the governance model for the artifacts created in *user stories 3 and 4*. Looking back at *user story 2*, we can see that its tasks are generic for any artifact so there is no need to duplicate or customize any of them.

Now we have four user stories in the backlog, we can move on to the last scenario and extract more stories out of it.

Third scenario – business architecture

ABC Trading has always been a wholesale company and has never sold any items directly to end consumers. The relationship with their retail partners has been excellent and nothing has emerged to alter the – so far successful – business model. However, the only constant thing in this world is change and major changes occurred in the world in 2020 that affected businesses no one thought would ever be affected by crises.

This scenario is about a business problem that needs to be resolved not only with technology but with **people**, **business processes**, **information**, and **technology**, which together enable your business to provide services to its consumers. In other words, these four aspects combined make your organization *capable* of providing services. Therefore, it is called **business capability modeling** and it is an immensely powerful business planning tool that helps you to understand how changes to the provided services require changes to the components that support them and vice versa.

You can map this example to any similar changes to the business of your organization – if your business is in manufacturing and wants to introduce a new product or possibly to discontinue an existing one, or if your business provides financial management services and is planning to introduce a consulting line of business, you can use the exact same concept that will be provided here.

Scenario description

The COVID-19 pandemic has changed the way people work, study, and shop. People used to enjoy going to shops to buy what they need, but with everyone staying home, the demand for online shopping and delivery services has increased dramatically and has made clear the difference between a successful business and a struggling business.

ABC Trading's retail partners were no exception from the effects of the pandemic and some of them went out of business because they were not ready to change. ABC Trading's revenue went down by 20% as a result, and all the credit for it remaining in business goes to the retailers with established online shopping channels. With that said, ABC Trading has decided to establish online retail sales to sell goods directly to customers and reduce its dependency on retail partners. A feasibility study was done a few months ago and showed promising profit figures, so the CEO has decided to proceed with the initiative. Part of the proposed risk mitigation plan is to reuse or reallocate the existing resources if possible before considering procuring new ones.

The CEO has asked you to develop a high-level business architecture document describing the new online sales service, what ABC Trading needs to do to start providing the service, how much can be reused from the existing resources, what is missing, and what needs to be built and developed.

Artifacts backlog

To start with, we need to break down the CEO's requirements:

- Define how the new business service will be provided.
- Identify how the existing business services are provided and find the components that can be reused for the new service.
- Define the gaps that need to be closed to be able to provide the new service.
- Define a roadmap with the required actions to take.

The next step is to build user stories based on these requirements. Two user stories can be identified.

User story 5

This user story is all about business capability modeling for the new service.

As CEO, I need a model that shows what we need to be able to provide the online shopping service to customers so that I can identify the required changes to make.

For now, we must continue building the artifacts backlog, so we need to identify the required tasks under this user story but using architectural terminologies:

1. Build and develop the To-Be **business capability model** for the online shopping **business service**.
2. Build the To-Be **business actors** catalog that defines what human resources are needed for providing the service.
3. Build the To-Be **business processes** catalog that defines how the new services will be provided.
4. Build the To-Be **application services** catalog that defines the required application services.
5. Build the To-Be **data entities** catalog that defines the required data and information.
6. Build the To-Be **technology services** catalog that defines the required technology services.

We already have some user stories for some focused metamodels, so we need to add the following ones:

1. Build the business capability focused metamodel.
2. Build the business service focused metamodel.
3. Build the business actor focused metamodel.
4. Build the business process focused metamodel.
5. Build the data entity focused metamodel.

Now we can add *user story 5* to our backlog and move on to *user story 6* to find what components we can reuse to provide the new service.

User story 6

User story 6 is a continuation of *user story 5* but because the latter was already too long, it needed to be split into two stories instead of one.

As CEO, I need to know what the closest existing services to the new ones are and what their components are, so that I can identify what can be reused and what actions need to be taken.

The architectural tasks for this user story are as follows:

1. Identify the **As-Is business capability model** for the selected business service.
2. Identify the **As-Is business actors** catalog that defines what human resources currently provide the current service.

3. Identify the **As-Is business processes** catalog that defines how the current services are provided.
4. Identify the **As-Is application services** catalog that defines what application services support the current business service.
5. Identify the **As-Is data entities** catalog that defines the data and information that are required currently for providing the service.
6. Identify the **As-Is technology services** catalog that defines what technology services support the current business service.
7. Identify the **gaps** catalog between the **As-Is** and **To-Be** models.
8. Identify the **course of actions** catalog to define required actions to take to be able to provide the new service.

As with the other stories, we need to build a focused metamodel for the artifacts that we are planning to produce. The metamodels can be used for both the As-Is and the To-Be artifacts so we do not need to repeat the tasks that were defined earlier.

9. The only new element that does not have a focused metamodel is the **course of action**, so we need to build the **course of actions** focused metamodel.

All the required governance processes have been identified in *user story 2* and there is no need to change or customize any of them to fit the additional scenarios now, so we will keep all of them as they are defined in *user story 2*.

With a backlog like the one we have defined, it can take any time between 6 months and a year depending on the size of the organization.