OrbusSoftware

Enterprise Architecture **Gap Analysis**

Contents

Gap Analysis

On the surface, gap analysis is a simple technique used in enterprise architecture (EA), primarily to examine the differences between one architecture state and another

But there are many ways to consider gap analysis and many techniques that can be combined with it to increase effectiveness. Additionally, gap analysis in EA should be much more than a simple “spot the difference” game between the current and target states.

This eBook provides an overview of the main points that you may want to address for effective EA gap analysis.

What Comes First?

The general practice in gap analysis suggests that you start by looking at the current state and move on to what you need in the future state. While this may work in some business situations, with EA it is slightly different.

Often, unless we know the desired architectural state we won't know what is wrong with the current state. For example, one company was driven by a desire to standardize its technology stacks. Once they had a clear statement of this policy, and defined options for standard platforms, they could return to the current state from a business architecture perspective, and identify any concerns and complications from the proposed technical changes. In this case, definition of the future technology architecture preceded analysis of the current state.

And then there are intermediate or transition states. Again, widespread practice suggests that each transition state is produced through sequential or linear analysis, based on a chronological time line. But if you are dealing with a major EA transformation it may make sense to pick a mid-point between the current and target states, followed by further mid-points until the full sequence emerges.

An organization making the significant switch from a "bricks and mortar" business model to a "bricks and clicks" model might plan this as a five year change. It might be easier to define the two year transition state, before the 12 month and six month transition states, rather than start with the six month state and work forwards.

As a general rule, work back and forth between the various states and gaps in order to develop a coherent EA roadmap. The following diagram illustrates the typical process of gap analysis, and emphasizes this iteration between different states. It also highlights the overall need to include feedback at the end of each major change, ensuring the overall process follows a clear strategic vector.

Figure 1: A Typical Process for Gap Analysis

Current State

We need to know where we are now and where we will start. In particular, given the defined concerns and requirements – outlined in a request for or statement of architecture work – we must explain the ways in which the current architecture is constraining or limiting.

It's also important to remember that although there may be problems and issues with the current architecture, it's not all bad news. We are unlikely to sweep away the entire architecture, so we must identify which components work and should be keep. It might be useful to conduct a SWOT analysis to look at the strengths, weaknesses, opportunities, and threats that are inherent in the current architecture state.

When we examine the current state we also need to consider the attributes or characteristic features of the architecture that need to be improved. Although we might also discover attributes when we examine the future state, they are more likely to be apparent here.

It is these attributes or characteristics that help us to identify the particular EA components, configurations, relationships, dependencies, or behaviors that need to be improved. They will also be the things we can measure to know how bad things are or how much we have improved them.

For example, a bank characterized its account opening procedures as inconsistent and contradictory, because each product had a separate process for setting up a new account.

Current State

We can measure these attributes in two basic ways: quantitative or qualitative. A quantitative measure might be that we have more than 400 applications supporting the customer relationship management (CRM) function; a qualitative measure might be that the application landscape is too complex. The measures describe how we know things aren't right.

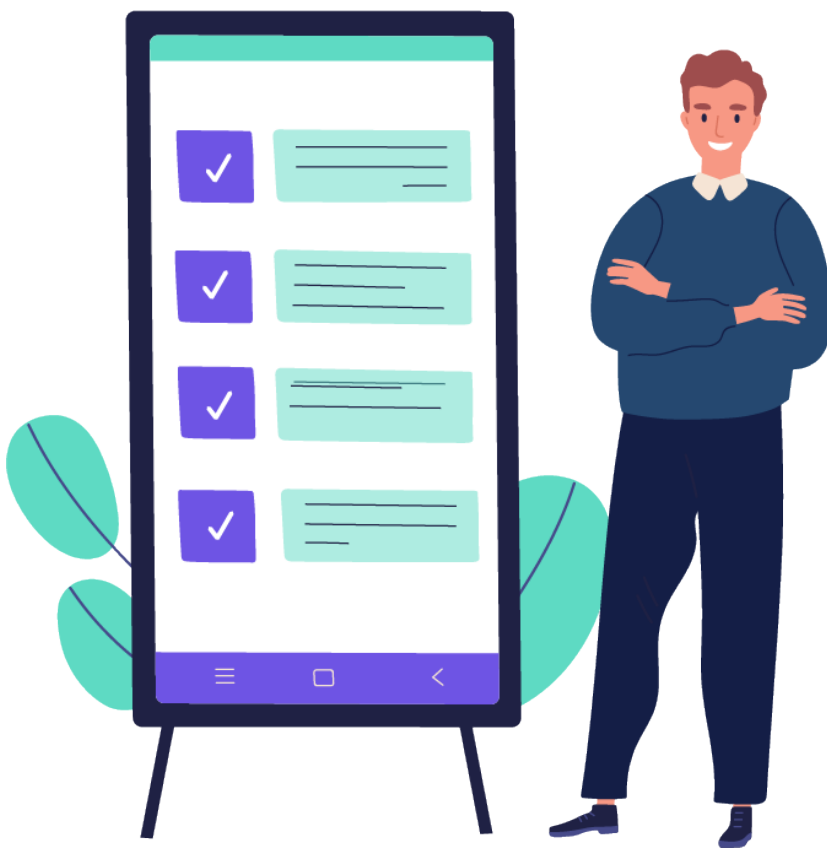
Describing the current state is not about blame, but it can be very helpful to give a bit of background or history. We might have more than 400 applications for CRM because past mergers and acquisitions simply extended and added to the architecture, without any rationalization or simplification.

The focus should always be on the concerns. Whether the scope is wide or narrow is determined by how much the architecture must change in order to address stakeholder concerns. Sometimes a small concern will require a big architectural change, and vice versa.

A senior executive thought that becoming customer-centric was a simple cultural change, but the EA team knew that it required significant changes in just about every architectural corner.

The other thing to remember about the current state is to look for the weaknesses and constraints. What is it that the architecture can't do, but should be able to do? You must have enough detail, and it should be specific and accurate, even if you only need to explain how the current architecture limits or restricts the capabilities that stakeholders need.

Just one final reminder: sometimes you won't recognize the weaknesses until you know more about future possibilities, which is where you have to move on to target states.



Target State

The target state is all about where we would like (or need) to be. A good enterprise architect will devise a strong future vision and a roadmap to get there; the best enterprise architects supply a range of alternative options and give decision-makers all the information they need to be able to choose a sustainable evolutionary path. Best practice ensures that all investments contribute towards the strategic vectors driving the enterprise.

What does this actually mean? Firstly, it is rare that only one possible vision or target will address the current concerns, but many EA teams work towards a single, obvious goal. Secondly, there can be a true long-term ideal and the best visions are exactly that; they are powerful visualizations of the future enterprise that show imagination, insight, and farsightedness. Third, such visions need to be grounded to a realistic, shorter-term plan that has well-defined, achievable intermediate steps.

For gap analysis, this takes us away from a simple transition from current to target, and replaces it with a more sophisticated decision-making tool that allows an enterprise to weather unpredictable and often uncharted seas. For example, a telecommunications company has three alternative longrange EA visions that it is working towards, because it cannot be certain that one particular option is “right”.

The EA team constantly assess trends and market pressures, using this information to adjust their short-term course. If there is an unexpected change, then they revisit their transition architectures and route maps, but the overall direction is guided by an ongoing, dynamic gap analysis process that leads the enterprise along defined strategic vectors.

The best way to work towards dynamic and evolutionary targets is to focus on the characteristics or attributes of the target state. Review the attributes from the current state, and add any additional attributes for the target state. Each attribute should then be given a specific quantified target for example to reduce the number of CRM applications to 10 or fewer; or a generic qualitative target, such as simplifying the CRM application landscape.

Target State

Note that sometimes you don't know (exactly) what the target state will be, so gap analysis helps to bring the future to life and be more precise.

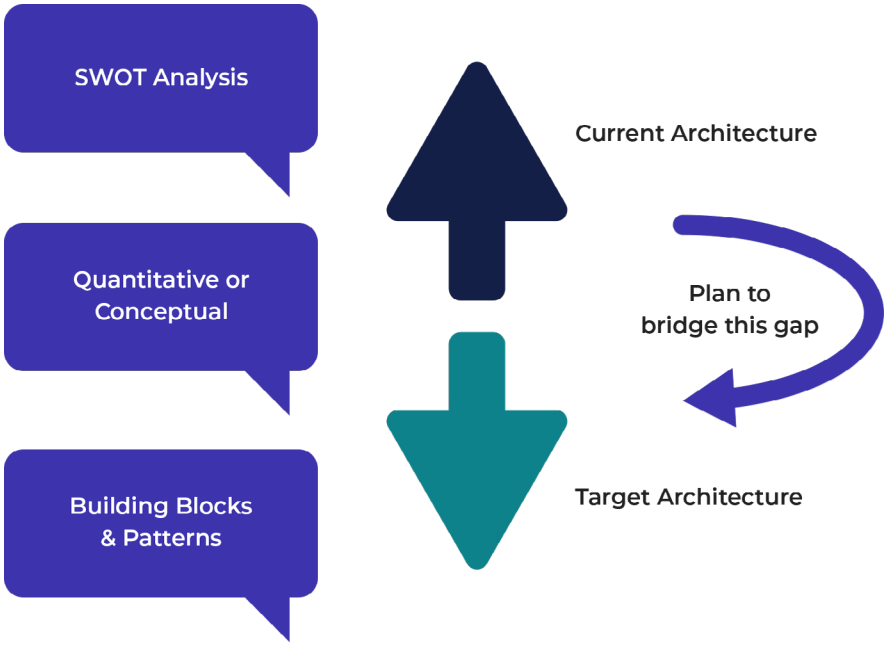


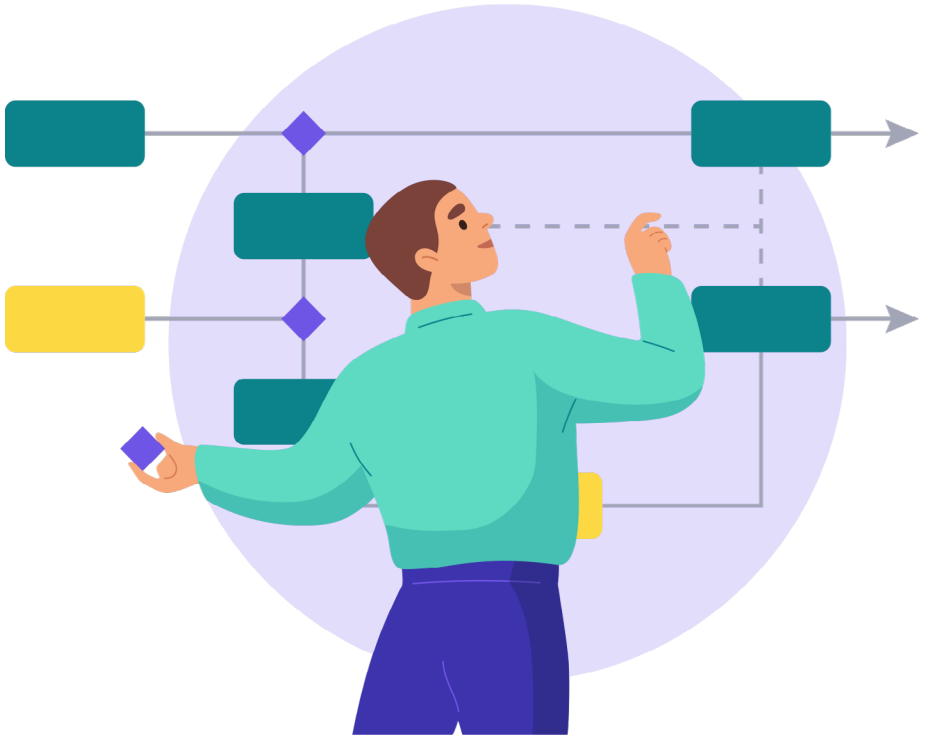
Figure 2: Techniques That Help Bridge EA Gaps

Identifying the Gaps

Once we have both a start point and end points, we can identify and bridge the gaps. The first task here is to be precise about what the gaps are, starting by describing the gaps in terms of the difference in their attributes or characteristics.

For example, we need to reduce the number of applications supporting CRM from more than 400 to less than 10. This gives us something to focus on; we can identify which applications provide the requisite core functionality, and be more precise about the capabilities these applications deliver.

The guiding question should be: do we really know the EA root cause? We are looking at the components or building blocks that make up the EA to find out how the selection of one component over another, their structure and configuration, or their interactions and behaviours cause the problem in the first place. Then we decide if we can fix the root cause, or whether we are merely putting a fix in place.



Bridging the Gaps

It is really helpful to look at the different types of gap. Here are some of the possible gaps:

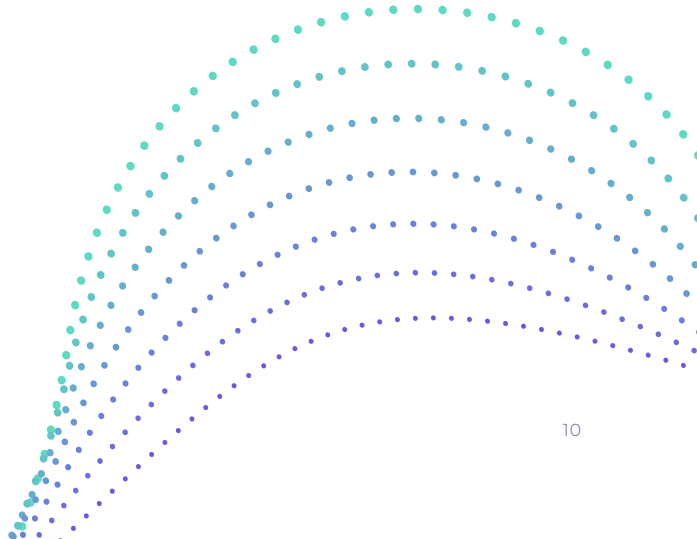
- We have something that we don't need (or that is causing harm or damage) – remove it
- We have something but it's not right – replace it
- We already have what we need – no changes required
- We have part of what we need – we must add something to what we've already got
- We have a total mess – we must radically redesign and replace everything
- We have a mix of good bits and bad bits – we need to selectively improve

The basic options are to remove, to keep, to reconfigure, and to replace. Note that in pretty much every case we are looking at replacing an old pattern with a new one. Patterns are an excellent tool for analyzing differences and showing gaps.

Also note that the gap can be a full gap or a partial gap, and that partial gaps are more common. In some instances there is no architectural gap, even though there is a concern or issue. A good case in point is where an outdated version of purchased software is causing problems with maintenance costs or contracts;

Also note that the gap can be a full gap or a partial gap, and that partial gaps are more common. In some instances there is no architectural gap, even though there is a concern or issue. A good case in point is where an outdated version of purchased software is causing problems with maintenance costs or contracts; upgrading it will resolve the problem, but doesn't change the architecture in any way.

- Are there any alternative patterns that will bridge the gap?
- How should decision-makers decide between one pattern and another? What metrics or measurements do they need?
- In what sequence or order should we plug the gaps? Which ones have the highest stakeholder priority? What is the optimum order to build the best EA?
- Are there any alternatives roadmaps?
- How should decision-makers decide between one roadmap and another?



Conclusions

Two key points are that

1. Analysis must be from an architectural perspective.

2. The best EA teams examine a range of options for future states and roadmaps to get there.

If gap analysis doesn't highlight why and how the architecture constrains or enables required enterprise capabilities, then it is not effective as EA gap analysis.

There is one other important observation about gap analysis. The maturity of the EA team determines how well it performs gap analysis and, more importantly, how well it guides architectural evolution to bridge these gaps in the most effective way. This is almost too obvious to state, but there are many EA teams facing this conundrum: that a high degree of EA maturity is necessary to successfully address big gaps, but that it is usually low maturity that causes the big gaps in the first place. This is illustrated in our final figure.

EA teams therefore need to work on improving their gap analysis skills, but it must be done in conjunction with building stronger EA capability in general. In particular, they need to build skills in using enterprise patterns, strategic themes and strategic vectors, road-mapping, SWOT analysis, qualitative and quantitative or conceptual metrics, presenting options, and stakeholder management, to mention but a few.

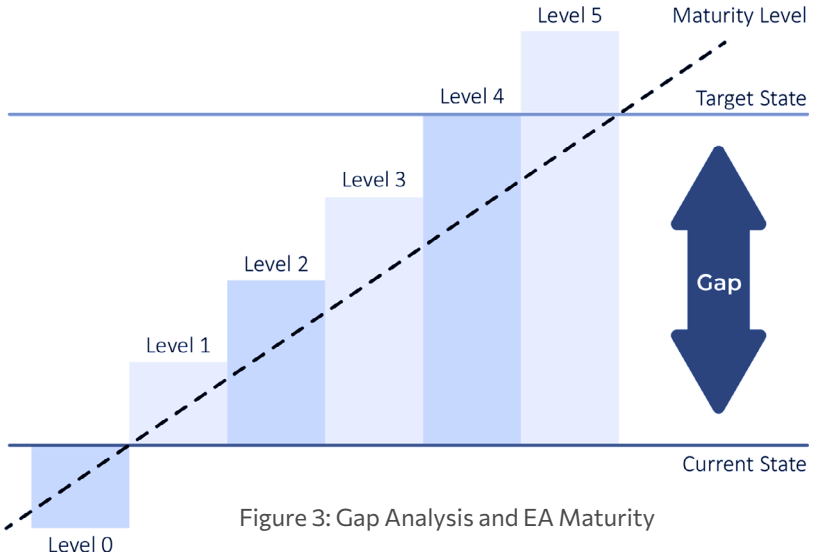


Figure 3: Gap Analysis and EA Maturity

With regards to gap analysis, it is easier to apply to EA components and their structure. It becomes harder to examine gaps in the behavior and dynamic aspects of EA, and it is even harder on interpretative use of EA components, where components are used in unexpected ways.

Another aspect of maturity is whether the enterprise itself is open to architectural change. Will the enterprise stay the longer-term course? How comfortable is the enterprise with architectural change? Is the enterprise willing to treat the causes, or only the symptoms? And is the enterprise willing to look far enough into the future, or look accurately at the future?

About Orbus Software

Orbus Software is a leading global provider of enterprise transformation solutions. We aim to empower customers with a strategic decision-making platform to successfully manage complex change. Our OrbusInfinity platform enables leaders to deliver business objectives, innovate faster, and ensure enterprise resiliency, while supporting them to make more informed, responsible, and sustainable business decisions.

orbussoftware.com

EBK0110/1025