Chapter 4
Analysing Enterprise Architecture

Let us reflect what we learned so far in this textbook. In Chap. 1, we started with an introduction to the topic of EA and EAM and also presented idea goals of the purpose of EA and management. We then took a closer look at concepts for describing the business architecture in Chap. 2. We not only covered business processes, but placed strong emphasis on the notion of *business capability* and *business object*. In the subsequent Chap. 3, we looked at how software applications relate to the business architecture. We derived an ideal application landscape from business capabilities and had a look at additional information required for describing an entire application landscape in a corporate environment.

So far the majority of this book was focused on the descriptive aspect of EA. It contains concepts and maps that assist in developing a picture of the EA of an organisation. The application and analysis of those viewpoints will be introduced in this Chap. 4.

Don't worry. This does not require the application of complex mathematics and mathematical models for fancy analysis. Most of the analysis will be facilitated based on visualisations that you know already from previous chapters. We will extend this by presenting basic principles for new viewpoints. We will introduce one very important tool, the so-called business support matrix. This matrix helps with the analysis of the relationship between IT architectures on one side and the business on the other. After completing this section, the subsequent section called Managing EA will then deal with the role of the enterprise architect and how to set up an organisation so that EAM can be performed within an organisation. We will also have a look at typical activities in EAM.

Learning Objectives

At the completion of this chapter, you will be able to . . .

- . . . explain different kinds of maps for visualising EA
- . . . apply common visualisations for EA

J. Jung, B. Fraunholz, *Masterclass Enterprise Architecture Management*,
https://doi.org/10.1007/978-3-030-78495-9_4

- . . . create a business support matrix based on capabilities and applications
- . . . discuss business support based on the business support matrix

As per these learning objectives , after completing this section, you will be able to explain different types of maps for visualising EA (cf. Sect. 4.2). You will further understand data required for creating those visualisations. You will gain an overview and able to apply common visualisations that are used in today's EAM disciplines to visualise the architecture, and also, drive decisions within the company.

Furthermore, we will take a look at a very special tool, the so-called *business support matrix* in Sect. 4.3. You will be capable of creating a business support matrix for a given domain. Based on this business support matrix, you should discuss the implications of what you see there. You will identify and discuss typical issues, and consider follow-up actions for solving the issues identified by the business support matrix.

4.1 Objectives of Enterprise Architecture Analysis

Let's start with the general challenge in today's world for an enterprise architect. Many enterprise architects still very much look like those architects building houses and large buildings: walking around with big blueprints depicting architecture and information. I still remember one of my superiors who was walking around with all the blueprints showing them to his boss. He was showing off the work his department did, and explaining what the top executives can now do with these designs to make decisions based on the information shown there.

It is still like that in many organisations. Especially those that introduced a dedicated EA tool will use it for generating the blueprint automatically. One of the most important hardware tools for architects is the large format plotter capable of creating plots for the large format documents that are then distributed. In fact, it often happens that those outputs are used by top executives. You might see some of the posters attached to the wall when entering their office.

Analysing the EA can cover a variety of aspects in an organisation, especially as some people have a very broad interpretation of EA,[1] it might even cover the analysis of business processes and the corporate strategy. However, this textbook follows a narrow perspective by concentrating on business-IT alignment. We will, therefore, restrict analysis to:

1. application architecture
2. the relationship between application and business architecture

[1] You still remember the three schools of thought on EA identified by LaPalme et al. in [1]. They are briefly introduced in Sect. 1.3 starting from page 18.

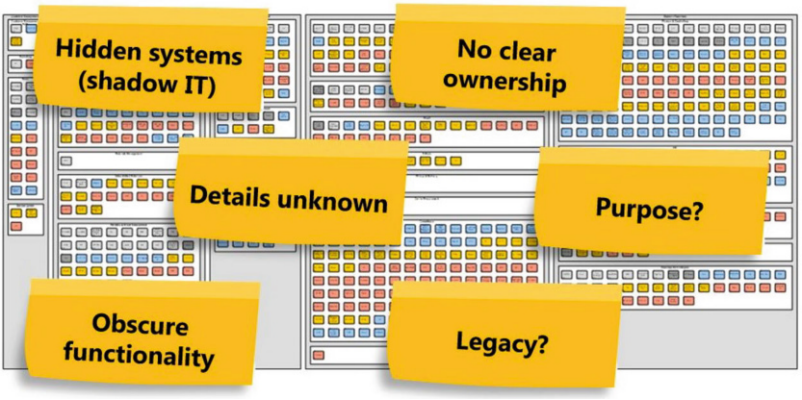


Fig. 4.1 Example issues for EAM

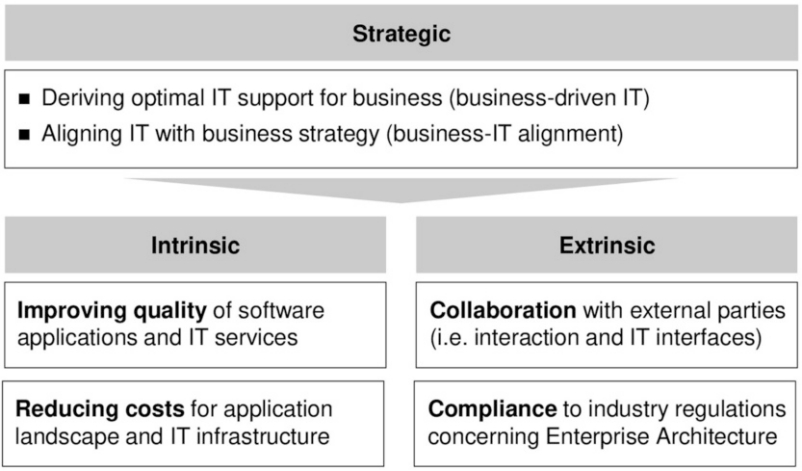


Fig. 4.2 Classification of objectives for EA analysis

We first need to discuss typical objectives that are part of the analysis. Analyse without a clear objective. You should know what you want to achieve with your analysis. It is usually based on some assumptions on existing issues (like for example in Fig. 4.1) that will lead to potential changes. We already presented a couple of objectives in Sect. 1.2, providing examples from current practitioners and decision makers. We will now put them into a structure, a taxonomy for objectives for EAM as shown in Fig. 4.2.

There are **strategic** objectives for applying EAM. This is an executive management decision within the company, because people recognised issues with IT support. We might have operational issues with business processes. Conversely we want to have a holistic approach for optimising our business. Many EA initiatives

have been initiated in companies because of a strong need for transparency and business-IT alignment. Within the strategic section, companies want to understand the business so that they can derive their optimal IT support. This is the so-called *business-driven IT* within corporations as it is based on business requirements following a holistic approach. Another objective is making sure that anything we do within IT is properly aligned with the business strategy and business needs—the so-called business IT alignment.

There are also more **operational** objectives for EAM. These can be further categorised into two different types of objectives. One of them is called *intrinsic* because they are driven by the company and provide an immediate benefit. Beside this, we also have *extrinsic* objectives that are externally driven from the context of the company. Examples for these are legal requirements, industry regulations, business partners or markets.

Intrinsic objectives can be classified as either improving the quality or reducing cost. In the case of EAM this refers to improving the quality of software applications and IT services so that they support the business properly. Even if people are not explicitly talking about reducing cost, in many initiatives, this is one of the implicit objectives for many IT heads in typical corporate environments.

Further **extrinsic** objectives are divided into two categories. One of them refers to collaborating with external parties. This means we need to integrate information systems, but it also refers to changes in our business processes, in our procedures, in the way how we present data to our external parties so that they can work with them.

A company needs to be open to the exchange of information (also data and documents) with external parties. Especially in today's e-commerce environments, in e-business companies collaboration is one of the key capabilities. Companies are not managing the whole value chain for a specific service on their own, but only fraction of the supply chain. Imagine the use case of ordering a packet of tea in China at one of the popular marketplaces.[2] People are offering tea and you can order it on the marketplace. There will be some payment service involved, which is provided by an external payment service provider. This also involves logistics companies which will make sure that your pack of tea is transported from China to your home address.

This simple example already highlights a lot of parties involved in tea delivery. Firstly, the provider of the marketplace, then the company offering the tea on the market place. Then a different company providing the payment service. Then the local logistics service provider in China picking up all the parcels from the tea manufacturer. It will then hand them over to airlines so that airlines can send the parcels to your country. And then your local postal organisation taking the parcel and delivering it at your doorstep. This workflow is even more complicated in reality.[3] But this end-to-end process already involves six business partners. And

[2]Popular marketplaces are Amazon, eBay or tmall and AliExpress in China.

[3]We left out customs, for example.

they all need to interact smoothly so that you do not experience any delays or lost parcels. This simple use case already requires a lot of data exchange between those partners. Therefore, in a networked e-commerce environment, it is not only about checking internal factors for EAM, but also checking how do we connect with our environment.

Business is usually also constraint by legal requirements, fiscal requirements, and industry regulations. These constraints place restrictions on the design of our processes and IT systems supporting the business processes. Beside quality, cost and collaboration, we also need to achieve compliance. Being non-compliant will have a negative impact on the company. Examples of this could be loss of reputation or fines.

Strategic objectives defined by executive management will drive operational objectives. Strategy provides the guidance on how to achieve corporate objectives and also helps with prioritising different options while analysing and improving business performance.

4.2 Enterprise Architecture Visualisation

How is enterprise architect supposed to be working today? Do you remember the diagram in Fig. 4.3? We presented it briefly in Chap. 1 on page 17 when explaining the working mode of an enterprise architect.[4] The enterprise architect in corporate IT is working in a very similar fashion as an architect in construction creating buildings or developing huge maps. The enterprise architect is producing huge posters with information displaying the architecture of the company. Those maps, referred to as viewpoints in this model, are created from a plethora of information that is stored in an EA repository.

One can do it manually, like for example having all the information in office documents or tables, and then draw the maps using common drawing tools. There are also dedicated EAM tools available at the market that support this step automatically. You need to collect all the data about your applications, about your business capabilities, about business objects, and their relationships. It covers any concept presented so far in this book. Information needs to be stored in a structured way in the EA repository. And then the EAM tool can create certain viewpoints (the maps or the posters we are referring to). The visualisation is supposed to support stakeholders in the company with addressing their specific concerns.

[4]The diagram has been created based on a similar discussion in [2, pp. 35] and [3, pp. 5].

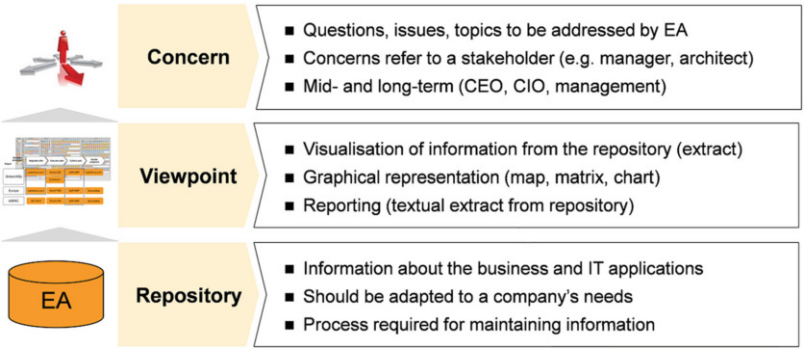


Fig. 4.3 Working mode of an enterprise architect

4.2.1 Types of Maps

There exists substantial research on the visualisation of EA with respect to stakeholders' need. One research group headed by Professor Matthes at the Technical University of Munich is working on providing a sound theory containing an EA repository and the generation of viewpoints. Among others, this group published a research report co-authored by Khosroshahi and others containing typical patterns for describing EA (cf. [3]). The theory encompasses a data model for EA, viewpoints and stakeholders together with their concerns. For the visualisation part, they are defining three different kinds of maps as shown in (Fig. 4.4).

The first type is called the **cluster map**. It is usually used for partitioning elements into logical domains, like for example assigning applications to organisational units or assigning applications to business capabilities. Sounds familiar? Just reiterate the examples provided in this book so far. By doing so you will find a couple of cluster maps.

A second type of visualisation is called the **Cartesian map**. While we can assign elements only to one background or to one kind of information in the cluster map, the Cartesian map is usually two-dimensional. We are mapping elements against two types of informations. Spoiler alert: The business support matrix as presented in Sect. 4.3 is a Cartesian map. You did not experience many examples Cartesian maps yet. These will be part of subsequent paragraphs in this section.

The third type of visualisation is called the **graph layout map**. It looks like a graph in software engineering, showing elements as nodes and the relationships as edges between these. Unlike cluster maps here we do not apply a visual background.

Most of these maps might already sound familiar as we explained EA by drawing on these examples. Such examples can be categorised as one of the types of maps introduced in Fig. 4.4. There will be some more examples in the following section (except for the Cartesian map as it will be introduced in the context of the business support matrix).

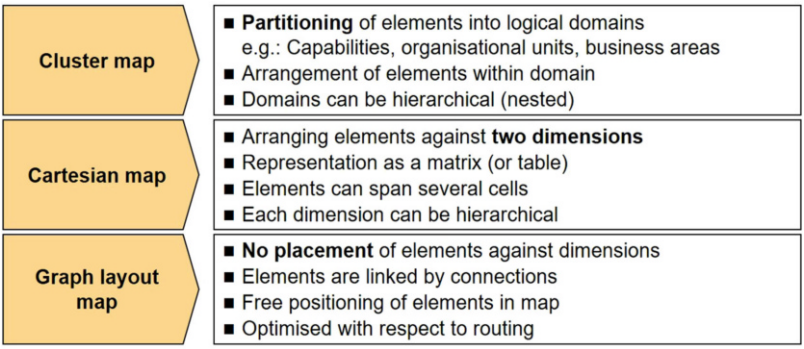


Fig. 4.4 Types of maps

4.2.2 Example Maps

An example cluster map is provided in Fig. 4.5. It might look very familiar as we have been using cluster maps from the beginning of this textbook.

The cluster map shown here is providing applications as elements and mapping these against organisational units. There exists an application called the **Customer portal**, others are called **Order management**, **3rd party booking** and a **Billing system**. An application is visually placed within the organisational unit where it is used. The sales department only requires one application, the customer portal. The operations team uses two applications supporting operations processes. And the finance department has one system, too. Clusters are built by organisational units of a fictional company and applications are mapped against these.

The graph layout map might look familiar to you because we used similar visualisations in this book previously. Figure 4.6 shows a different example for a graph layout map. The applications are the same as in Fig. 4.5. They are connected by data flows, showing which data object is flowing from one application to the other.

You are not restricted to a single type of map but allowed to combine views. Figure 4.7 shows the combination of the previous two maps (from Figs. 4.5 and 4.6). We have the cluster map in the background for detailing the organisational units and then adding arrows for the data flow. This is a combined cluster map and graph-layout map.

4.2.3 Software Cartography

Let us now take a look at the general structure of a map and elements for constructing them. It will be rare to create a pure cluster maps and a pure graph layouts map. Frequently you will mix these visualisations. Pure maps can be helpful

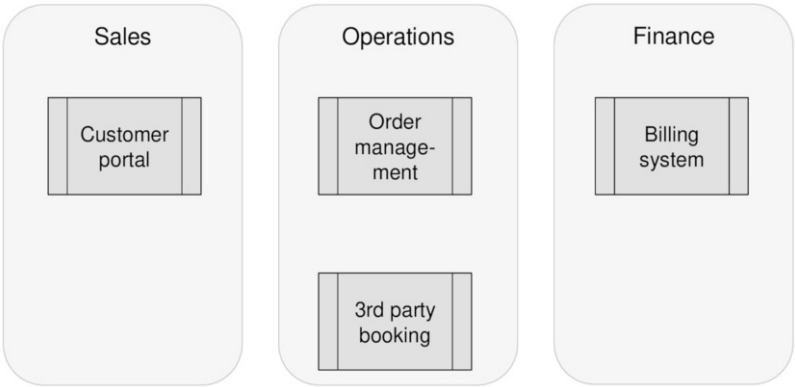


Fig. 4.5 Example cluster map

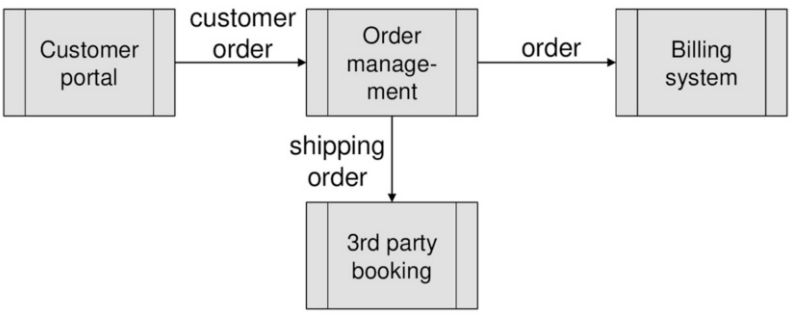


Fig. 4.6 Example graph layout map

(and are widely used), but you will frequently see complex maps displaying different types of information.

Based on the theory provided by André Wittenburg as member of Professor Matthes' team, the general structure for such a map is shown in Fig. 4.8. The background is represented by a **base map**. This could be clusters from a cluster map, for example. You can assign your **elements** against this background. These will be applications most of the time for the purpose of this book. These could also be any other types of element, including business objects, data objects, roles, people—in other words: anything that is relevant for the EA. a base map with clusters together with a layer for elements, we can already construct a cluster map.

An additional layer for **relationships** can be introduced for displaying connections between elements.[5] This results in a combined visualisation are shown in Fig. 4.7.

[5]The two layers for **elements** and **relationships** build a graph-layout map.

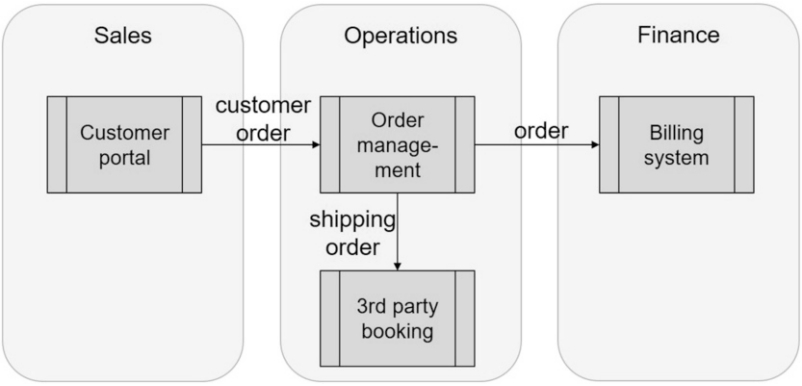


Fig. 4.7 Example combined views

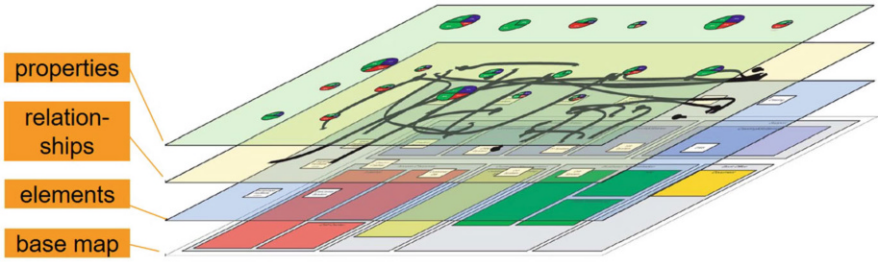


Fig. 4.8 Layers of a map (based on [4, p. 83])

Additional **properties** of the elements can be shown using a fourth layer. You can add for example information like:

- run and maintenance costs of each application
- quality attributes (e.g. technical fitness, business fit)
- planning status (e.g. *planned*, *in use* or *to be decommissioned*
- age of an application (e.g. *legacy*, *state-of-the-art*, *innovator*)

Most maps used in EAM will consist of these four layers. These will have a background and show elements, which are the focus of the investigation. They can display, but are not necessarily limited to, relationships between these elements. Further the properties can provide additional information for the elements shown on the map.

Let us explain this structure by the example map given in Fig. 4.9. This is an extension of the example from Fig. 4.7. The background is build by the three organisational units with software applications assigned to them. The relationships, again, show data flows between them. The top level—represented by the traffic light symbols—adds information on the applications.

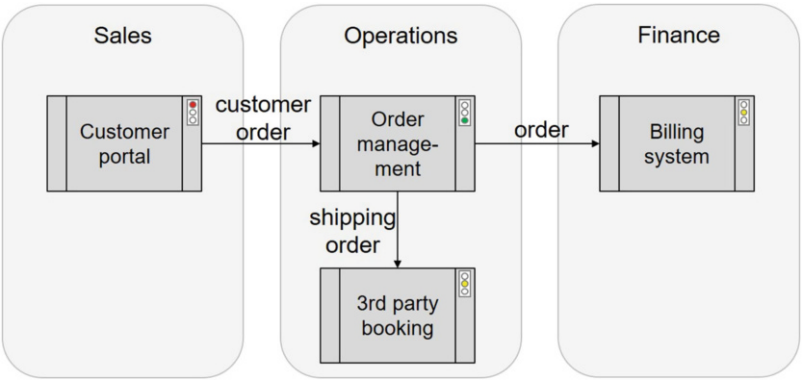


Fig. 4.9 EA viewpoints—example map

The traffic light may indicate the security status of the application:

- *red*: poor level of data protection and security (e.g. **Customer portal** in Fig. 4.9)
- *amber*: average security level and data protection policies
- *green*: high level of data protection and security (e.g. **Order management**)

The red traffic light on the **Customer portal** indicates a severe issue, because a customer portal is accessible from outside the corporate network. It needs to have a high security standard so that intruders cannot use it for getting access to our network.

Amber traffic lights, in this case, might indicate systems with an average security level. They are neither outstanding nor poor. However, the implications here are not clear by just looking at the map. You have to have more information for a precise assessment of the severity. Is it bad having an amber traffic light or not? Do we need to do something? Those symbols can only reflect a highly aggregated view on the application. However, they help in getting an overview on the whole application landscape. Simple visualisations can help top executives with getting a grip on the big picture.

There is no standard way for showing additional information on an EA map. We just provided a simple example using traffic lights. Traffic lights might be associated with statuses like low (red), medium (amber), and high (green) if *high* refers to a positive property. The interpretation needs to be defined by the enterprise architect or by the wider company, so that every person looking at the map can understand what is represented by green, amber, and red. This includes:

1. What does the colour mean?
2. How do we determine the colour?

There are further possibilities for displaying information, like the *Harvey balls* in Fig. 4.10. These are typically used for indicating progress or status of completeness.

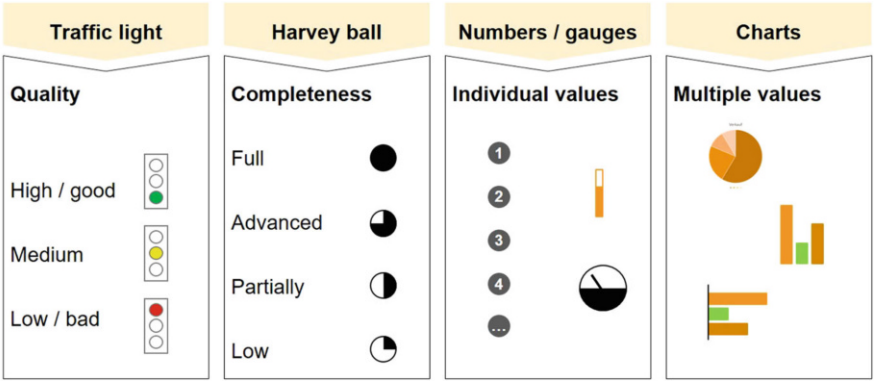


Fig. 4.10 Visualisation of properties

But it can also address a quality criteria with four different degrees. For example, the execution of a project can be indicated by Harvey balls:

- *low*: project just started
- *partial*: progressing
- *advanced*: major work done
- *full*: completed

They can also indicate quality:

- *low*: only a few quality criteria are met
- *partial*: half of quality criteria are met
- *advanced*: most quality criteria satisfied
- *full*: quality criteria fully satisfied

We can only provide examples on how to use Harvey balls and other visualisations. These can be adapted in any organisation just by defining their meaning. They normally will refer to existing properties or KPI.

Alternatively you can assign numbers. If you have security statuses based on numbers, or even if you want to show cost, then you can add numbers. You can even add text if you prefer, but the more text you include, the more complicated the map will become. Architects usually relate to graphical symbols which, of course, need to be interpreted. Further you can include gauges or progress bars to display relative values, like a percentage or some kind of relative values. Such values can, for example, show budget usage (i.e. budget used compared to the overall budget), or the running cost of applications compared to others.

Last but not least, you can also utilise charts for rather complicated maps. Example: The different cost types of an application (e.g. run, support, maintenance, training) can be displayed by a bar chart. Those charts can also provide a summary view on quality attributes (i.e. business fit, technical fitness and architectural fit).

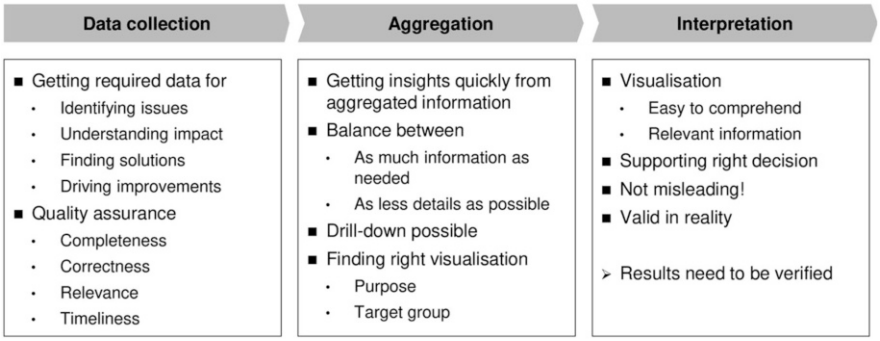


Fig. 4.11 Challenges with visualising properties

Creating these maps requires an EA repository with high quality data. This can be quite a challenging task in the corporate environment if no such repository pre-exists (cf. Fig. 4.11). An enterprise architect needs to start the collection of all data concerning applications, capabilities and business objects. This requires more than simply sending an email to everybody requesting them to provide that information. Architects usually spend a lot of time on sourcing information, contacting people, participating in meetings to obtain all relevant information. Business and IT stakeholders are normally busy with their own responsibilities or might simply refuse to collaborate with EA.

Analysis and optimisation of the application landscape in its business context requires a lot of information. This includes detailed cost figures, usage statistics of applications, number of users and business stakeholders. Obtaining all relevant data can become very tedious. You can keep an army of architects or external consultants busy with meetings to obtain all the relevant detail.

We also have to keep in mind that it is not only a one-time effort to populate the EA repository. We also need to ensure the data quality over time. This requires the data to be complete, correct, relevant and available on time. Furthermore, we need to make sure that we update this data over time. It is an illusion that you are only doing data collection once, and then you are able to simply focus on your EAM work. You need to make sure to update your data frequently whenever the organisation changes. To make matters more complex the application landscape changes too. Processes are also constantly changing. New products are introduced. All this needs to be incorporated into your repository, meaning you need to be aware of the change and constantly collect updated information.

Once you have all the details and data available, then you can think of how to aggregate these. You will never display all the details on one big map because this will be too confusing. You need to decide how to aggregate data, so that it will be helpful for the corresponding stakeholder. The stakeholder should see as much detail as required for addressing concerns. At the same time, you should display as much information as possible, so that all relevant facts are clearly visible.

The more unnecessary information you have, the more tedious it will be for the decision maker. If you show less detail this will adversely impact the decisions made. Aggregation must be planned, but should also be somewhat flexible, so that you can disaggregate data at a later stage. Displaying high level data on a dashboard or an EA map should also allow to drill down and show how this aggregated value has been calculated based on more detailed data. Each aggregation does not exist in isolation. You always have to keep the stakeholder in mind (referred to as target group) and also the purpose of the visualisation.

After you created the map, the stakeholder can use the map not only for visualising and understanding information, but also to support their decisions. Whenever you create the viewpoint, you need to make sure that the map is not misleading. It should not show anything that can be interpreted by the stakeholder in a wrong way. Data, even when aggregated, needs to correspond with the real world.

In real life the process never works by simply collecting data, creating a map, and everyone is happy. Whenever we needed to create maps, it was some kind of iterative process consisting of the following steps:

1. get initial data (might not be complete)
2. make a first draft map
3. discuss draft with stakeholder
    a. Can you use this for your work?
    b. Is it easy to handle?
    c. Is it easy to understand?
    d. Will it support your work?
4. collect additional data based on stakeholders' feedback
5. calculate aggregated data
6. improve data quality
7. refine draft map and continue with step 3

This process shows why traditional EA initiatives have a huge risk of failing. It is not simply an initial set up and then applying standard tools. It is an ongoing collaborative approach together with all stakeholders as organisations, concerns and perception differ vastly. There is no one-size-fits-all map.

Real-life corporations are very complex, thus we need a lot of information in order to properly understand business processes and software applications. You will not always obtain all the data you wish for in a professional environment. You need to start with what you managed to get and discuss first drafts with stakeholders. Their feedback will likely require changes to the map but also result in the need to collect more data (in case they require more information in the map). This is an iterative process until all parties agree on a map or abandon the idea. There is no fixed time-frame until you reach a result. It depends on your knowledge and understanding of the stakeholders for getting a result quickly. But it might also take a couple of months or more.

Many enterprise architects were driven by using existing EA tools. They simply collected the data required by the tool for generating pre-defined reports and maps. This can work but often bears the risk that top executives do not perceive the visualisations as helpful. Pushing for such standard visualisations that are not understood by top executives will be a guarantee for the project to fail. You need to listen to the customers of your work: stakeholders using your maps. This includes top executives, middle management and any relevant stakeholder required for the succesful adoption of EAM.

4.3 Business Support

Even though getting all the information is a challenging task, it needs to be done to support proper alignment between business and IT (as depicted in Fig. 4.12):

- *Effectiveness*: We need to understand the business so that we can provide an adequate application landscape.
- *Efficiency*: We need to know details about applications in order to become more cost efficient.

The world is not simply black and white. The views presented here reflect the perspective of a head of IT who wants to provide an optimal application landscape for the company. Optimisation on the business side is also required from a corporate perspective. With cost reductions, we can increase the efficiency of our business processes. Furthermore, IT is only effective if we make sure that the business is working effectively. From the IT perspective, we want to provide an efficient IT landscape. The remainder of this section will, therefore, provide a closer look at the relationship between business and application architecture. The business support matrix is introduced as a tool for visualisation and analysis.

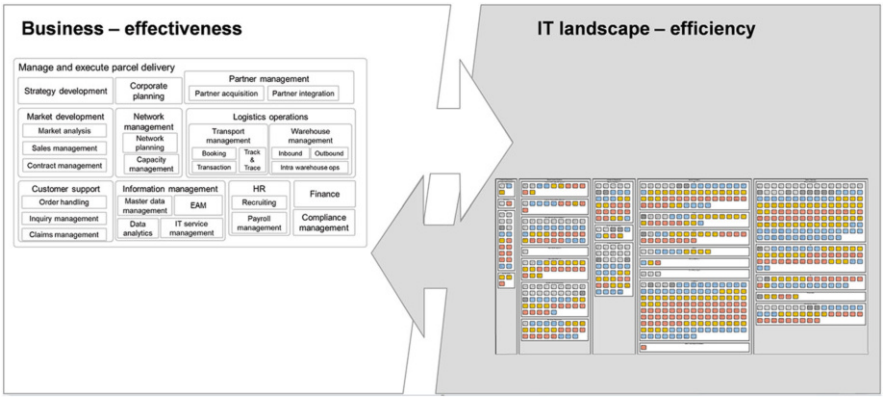


Fig. 4.12 Business and application architecture influencing each other

4.3.1 Business Support Matrix

Instead of starting with a definition, the notion of a business support matrix in EAM is introduced by an example in Fig. 4.13. You might remember this kind of visualisation from Sect. 1.3 like maps in Figs. 1.15 and 1.16. The matrix has two dimensions:

- *Business processes*: value-adding business processes performed within the organisation
- *Region*: organisation of the company based on geographic regions

The example contains the global headquarters and sub-organisations in Europe and in the Asia-Pacific region. Within the matrix, you see applications. **Scheduler+**, for example, is an application supporting **Execute order** in the global organisation. **SAP ERP** is a standard application for cash collection across the whole company. Don't be scared when reading the word *matrix*. We are not talking about mathematics and formal analysis. The matrix is a visual tool for describing, understanding and optimising EA.

A definition of *business support matrix* is provided in Definition 4.1. It is a Cartesian map for putting applications in direct relationship with concepts from the business architecture. This is what we exemplified in Fig. 4.13: applications are mapped against processes and organisational units. We could have done the same with business capabilities, corporate strategy, product or any kind of business-related concept.[6]

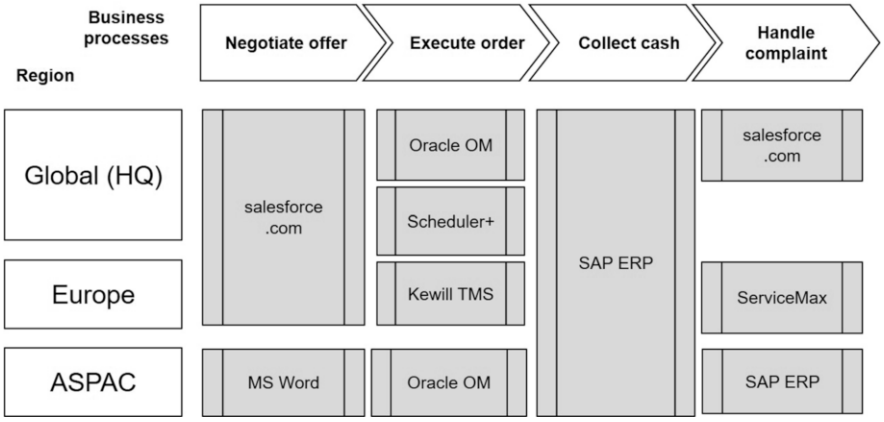


Fig. 4.13 Example business support

[6] An overview on business architecture concepts is given in Sect. 2.4 and in Chap. 2.

Definition 4.1 (Business Support Matrix) A business support matrix (BSM) is Cartesian map for setting applications in direct relationships with concepts from the business architecture (e.g. capabilities, processes or organisational units). There is no standard way but a variance of different matrix representations that can be applied. Hence, each of the two dimensions of the matrix can represent an individual business architecture concept.

There is no standard visualisation of the *business support matrix* or even the *standard matrix*. There is an example in Fig. 4.13 but there are various other representations. When studying textbooks and papers from the EA community, you will find a huge variety of visualisations for the business support matrix. It might not even be called a business support matrix but have a different name like *domain map*, *business map* or *application map*. We continue to use the term *business support matrix* throughout this textbook.[7]

We will now introduce a very specific kind of business support matrix that we will need to analyse the application architecture with respect to its business support. It has the following dimensions:

1. Business capabilities
2. Applications

An abstract example is provided in Fig. 4.14. A cell at the intersection of an application and a capability represents business support between them. If there

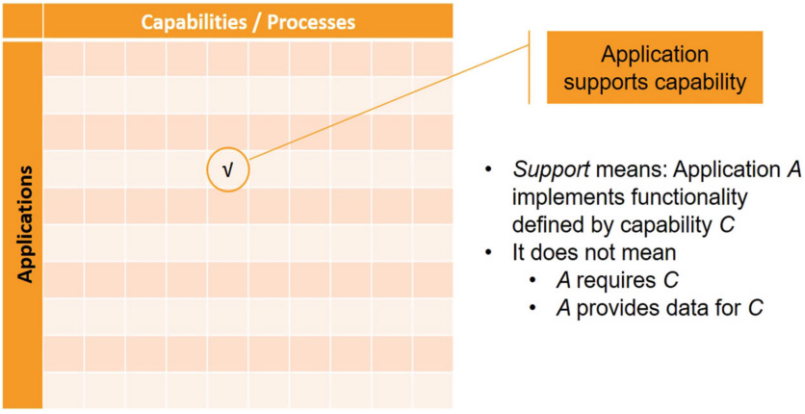


Fig. 4.14 Structure of a business support matrix

[7]This simple kind of matrix does not require a dedicated EA tool or a drawing program but can be easily created using a common spreadsheet application. We used this kind of matrix quite often when data was available in a (relational) database, SharePoint lists or Excel spreadsheets. It can be generated as a pivot table and then any kind of digram offered by Excel can be produced. It took a couple of years until this kind of visualisation emerged as the one most suited for all the stakeholders in the company for understanding the relationship of software applications and the business capabilities. Even though there was a dedicated EAM tool (planningIT), it was not flexible enough as people wanted to have different visualisations. We imported all the required data into an Excel spreadsheet and then used the pivot table to show the business support matrix.

is a tick, then the application does support the capability—otherwise it does not. This refers to the fact that the execution of the capability requires the respective application but not the other way. It does not show whether an application needs a capability. The matrix also represents a clear functional point-of-view: The application implements functionality that is used in the capability. It does not refer to data only.

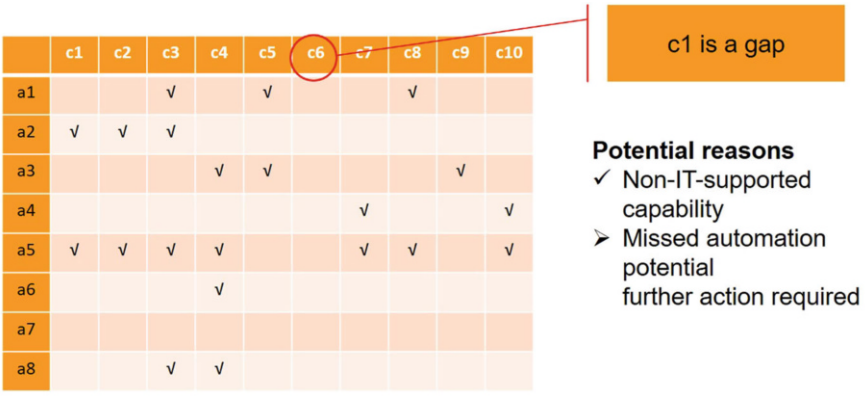
4.3.2 *Analysis Using the Business Support Matrix*

Figure 4.15 shows a more complex, but still abstract, example for a business support matrix. It is not only displaying one tick but all relationships between applications and business capabilities. It lists several applications, named **a1** to **a8**, and 10 different business capabilities just having abstract names, **c1** to **c10**. The specific meaning is not relevant for this example because we just want to discuss the structure. What we can see in this matrix is that application **a1** is supporting capability **c3**, **c5**, and capability **c8**. In the same way, we can see that the business capability **c8** requires two applications, namely **a1** and **a5**.

We will use this example to explain typical issues in the EA that can be identified with the business support matrix. These are:

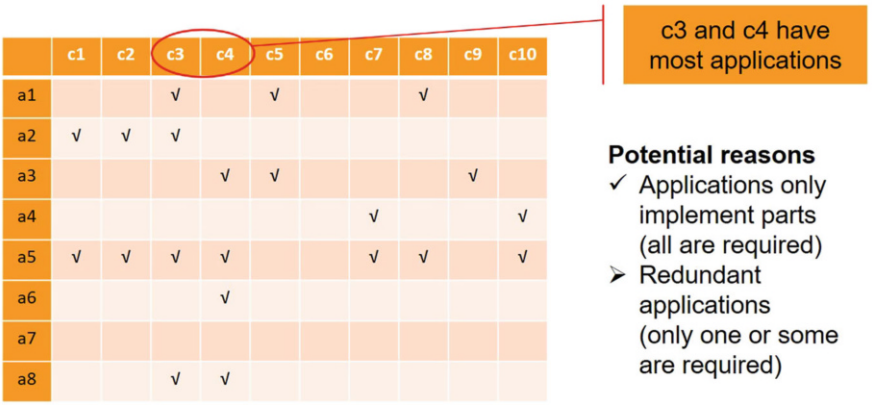
- *Gap*: missed automation potential
- *Redundancy*: too many applications
- *Orphan*: application without any business value
- *Monolith*: lack of modularity in the application landscape

The business capability called **c6** is not supported by any application. This does not necessarily need to be an issue but might be an indication for a missed



	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10
a1			√		√			√		
a2	√	√	√							
a3				√	√				√	
a4							√			√
a5	√	√	√	√			√	√		√
a6				√						
a7										
a8			√	√						

Fig. 4.15 Business support matrix indicating a gap



	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10
a1			√		√			√		
a2	√	√	√							
a3				√	√				√	
a4							√			√
a5	√	√	√	√			√	√		√
a6				√						
a7										
a8			√	√						

Fig. 4.16 Business support matrix indicating redundancies

opportunity for automation or IT support. We need to check the background to understand why it is not supported. Perhaps this is a capability that we do not want to be supported by IT because it is a pure manual activity (e.g. loading parcels into a truck). This root cause analysis requires human actors as the details are not visible in the matrix.

Another kind of typical issue to be identified using the business support matrix is determining capabilities that are using a lot of applications. You can do this very easily by simply looking at the matrix in Fig. 4.16. You can determine some capabilities (e.g. **c9**) only being supported by one application or by two (**c8** and **c10**). However, there are two business capabilities, each of them requiring a lot of applications. **c3** is supported by **a1**, **a2**, **a5**, and **a8**. **c4** is supported by four different applications. Looking at the matrix only allows us the neutral statement that each of them is supported by four applications and this is more compared to other business capabilities.

Is it good or is it bad? We cannot draw any reliable conclusion by just looking at the matrix. We need some more background information, like for example:

- Do each of the applications only perform a small fraction of activities in a complex business capability? Each of them might be very specialised on certain tasks.
- Are these applications required for different products or services? (e.g. manufacturing cars vs. motor bikes)
- Or, are they *redundant*?

Example 4.1 Let us assume **c3** being *Customer Relationship Management*. Having four different CRM applications would in fact reveal a redundancy. CRM is a commodity and we should have one integrated standard application. We should have only one database with customer data. We should not maintain sales training or sales-related training for four different systems, but only have one. In this case, **c3** would be supported by redundant applications.

Example 4.2 Another example is that business capability C4 refers to *logistics operations*. There can be one system for *network planning*, another system for *monitoring* the logistics network, one system for making the *bookings* with external service providers, and one system for *managing customer orders*. In this case, even if it is within one business capability, each of them has a different purpose, and then, four systems might be justified.

Observation number three is that application A7 is not supporting any business capability as highlighted in Fig. 4.17. This means we are maintaining and providing an application that does not provide any business value. This will, of course, hamper efficiency and most heads of IT would challenge this application. *Why are we paying for something that is not useful?*

EAM often identifies applications without supporting any business capability. This will usually drive initiatives investigating for reasons. Such an initiative

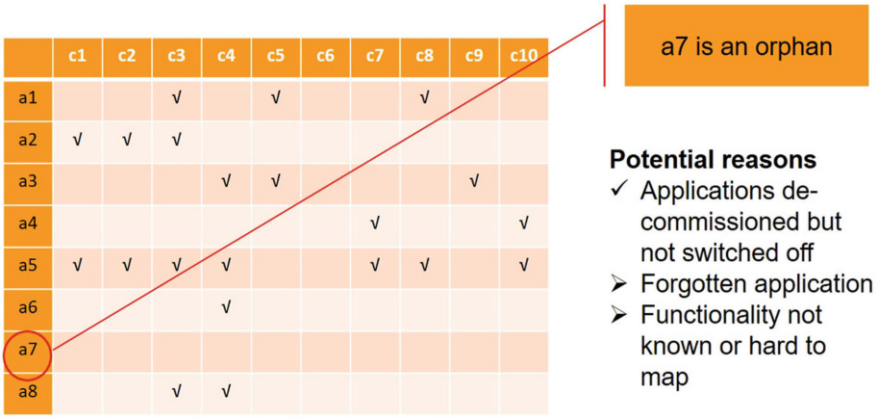


Fig. 4.17 Business support matrix indicating an orphan

will analyse the background of that application and provide a decision for its decommissioning:

1. It is a forgotten application that is not used any more. It can be decommissioned.
2. It is used for activities without business relevance. It might be decommissioned.
3. It is required by relevant business applications (e.g. an integration backbone). It should be kept.
4. It is directly or indirectly required by all capabilities (e.g. office suite). It needs to stay.
5. A business capability is missing (e.g. business collaboration). The capability map needs to be updated.

The business support matrix can support finding certain observations very easily. It is a simple but very powerful tool with respect to identifying common inefficiencies in the application architecture. It takes some effort to obtain all the information required for creating it. But summarising it on one single chart will be a visual aid for decision makers. They start finding issues by counting ticks and perceiving patterns (e.g. empty rows and columns) in the matrix. Also an accumulation of ticks (in a row, a column or a certain area in the matrix) are easy to grasp visually.

However, let us not forget about the fact this is only a highly aggregated representation of a complex system. It only displays aggregated information while leaving out significant detail. An observation in the matrix is primarily only an indication that there might be an issue. A careful analysis has to be conducted in order to reveal valid reasons or proof that it really is an issue. We do not know the reason or the business rationale behind it.

The fourth kind of observation is called a *monolith* as shown in Fig. 4.18. A monolith is a large and self-contained software application with a lot of functionality. Monoliths are often grown over time or introduced in the past. Common problems with monoliths are:

- No modular design and, therefore, no reuse of individual functionality (services)
- Hard to maintain and, therefore, impose a risk on the future application landscape
- Implemented using legacy platforms and, therefore, ...
  - inefficient for users
  - security risk
  - limited performance and scalability

A monolith cannot be split into smaller applications, as it does not have a modular design or is an integrated standard software. You can easily identify monoliths in the business support matrix by the number of tick marks in a row. Application **a5** in Fig. 4.18 is a monolith because it supports four business capabilities that are closely related to each other (**c1** to **c4**) and more.

When recognising a monolith, this observation should be neutral in the first step as with each observation from the matrix. Even though the term *monolith* has a negative connotation, it might be acceptable in many cases.

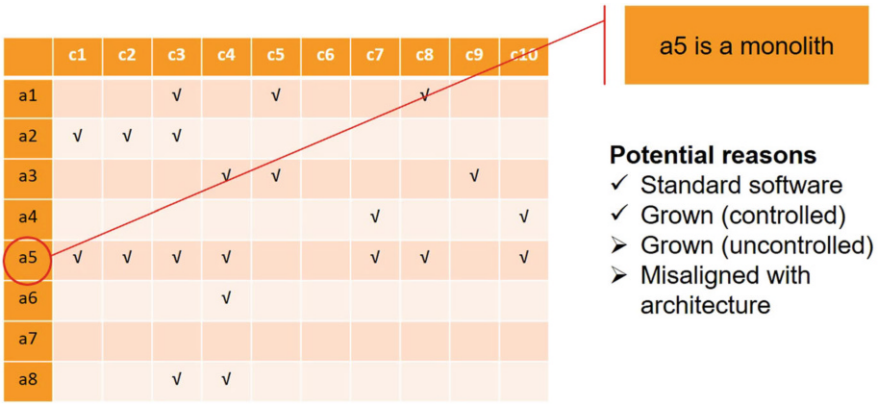


Fig. 4.18 Business support matrix indicating a monolith

There might be a conscious decision for having a monolith in the company. A proprietary core operations system[8] might be developed by the company as a competitive advantage. If you are developing it on your own, you can ensure modularity and maintainability with respect to future business needs. In such a case this is not an issue.

Also an integrated commodity application procured on the market is less of an issue. For example, a customer relationship management suite supports marketing, sales and customer service. If it is a conscious decision that you need only one dedicated system for these capabilities, then it is okay to have a monolith. Before making any judgement about the monolith being good or bad, you need to understand its purpose and why it is there.

4.3.3 Implications from the Business Support Matrix

Let us summarise the implications. The observations[9] from the business support matrix are listed on the left-hand side of Fig. 4.19 together with recommendations how to handle them:

- gap being a business capability without application support
- redundancies, having business capabilities with many applications
- orphan applications or application, not supporting any business capability
- monolith, supporting many business capabilities that are related to each other

[8]The term *core operations system* does not refer to a computer operating system but to an application being used for execution a company's core business processes.

[9]This is only an excerpt of observations you can make in corporate environments. Schlör and Jung [5] introduces some more, which are some kind of special and also require some more detailed analysis.

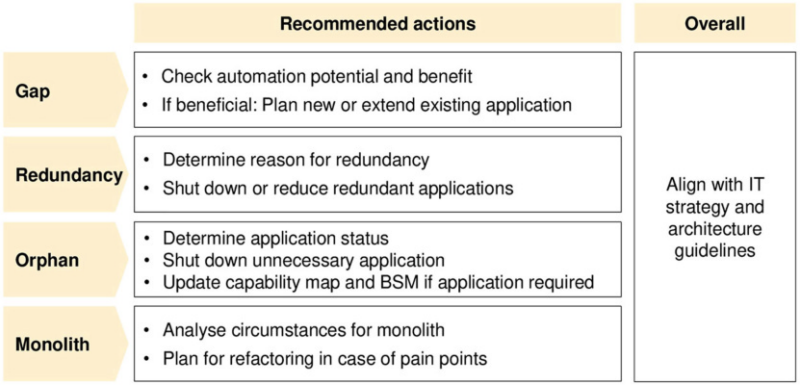


Fig. 4.19 Implications from analysing the business support matrix

In case of a **gap**, we need to check whether the business capability is a completely manual one. If not, there might be missed automation potential. There are two options for filling the gap:

1. introducing a new software application just for this business capability
2. check for an existing application that can support the business capability—even if it requires extensions

The same pattern holds true for **redundancies**. We first need to check the reason behind the it. And if it is perceived as an unnecessary redundancy then it should be eliminated by replacing applications. The same logic applies to **orphans**. Orphans without any reasons can be decommissioned. However, the analysis of the business support matrix may reveal some potential orphans that are still required (see above).

The implication in case of a monolith might be slightly different. You start with its identification in the matrix and then the analysis of the reasons. But, in case of an issue, you cannot get rid of it easily. You need to plan for a step-wise replacement of it. There are some common strategies for this:

1. re-factoring the monolith, which means dividing the software into smaller units
2. introducing one new system as a replacement (in case you need to replace a legacy monolith)
3. introducing several applications, each of which supports a fraction of the functionality (i.e. based on sub-capabilities)

The first option is easier to perform (and less risky) as it is only a redesign of the software. It does not need to be replaced and existing business processes are not affected as they continue using the same system. However, the benefit is quite low as the (legacy) software is still in use. It can be an initial stage before performing option number three.

The second option will consist of a large scale change as also the business processes need to be adjusted to a new software system. Further interfaces of the

old monolith need to be re-established with the new one so that data flows continue to work. This also bears the risk of hampering business during the transition and missing features of the new monolith.

The third option implies a program consisting of individual projects, each of which replaces a part of the monolith. It can be seen as a chance as the application landscape can be optimised by introducing several specialised applications. This *best of bread* approach consists of choosing the best application for each task and integrating them. The division of the monolith (assuming it supports a single capability) can be based on sub-capabilities, hence, the functional split of the software application is derived from the capability tree. the method introduced in Sect. 3.2 can be used here. The complexity of managing this program is quite large as the introduction of several systems need to be aligned and all dependencies must be managed properly.

4.3.4 Dimensions in Business Support

The business support matrix introduced in this textbook consists of two dimensions, allowing for only two concepts from the business architecture. The example provided above was limited to applications and business capabilities. But, what about the other concepts? The business support matrix is a valuable tool showing high-level information. However, we will most likely need additional information for further analysis. This can be incorporated by having additional dimensions for business support (i.e. more information behind each tick) as shown in Fig. 4.20.

Each cell in the matrix (i.e. each business support) can have additional details on time, organisational context, customer segments and products. It also allows for quantifying the degree of business support (e.g. the application only supports the capability by 25%).

Dimension	Application supports capability
1 Time	For a given time **period** (from .. to) or **plan** (as-is, to-be)
2 Organisation	In an organisational unit (including **countries, regions**)
3 Customer type	For specific customer segment (e.g. **consumer, business partner**)
4 Product type	Capabilities may require specialised applications for different products
5 Degree	Degree of business support (partially, completely)

Fig. 4.20 Business support—dimensions

The **time** dimension allows for distinguishing between several stages in the roadmap. A business support can be indicated as *as-is* (the application is currently supporting the capability) or *to-be* (the application will support the capability in the future). Time can also be given as concrete periods of time if there is already a plan for decommissioning an as-is application or when to introduce a to-be application.

We also need to know the **organisation** which is using this application. Remember, there might be a variety of applications for one and the same business capability.[10] We need different applications for the same functionality because of legal or physical reasons. We discussed it in Sect. 3.2 based on customs processing, because of different customs regulations resulting in different requirements. These kinds of redundancies are justified and it would be helpful to have this kind of information included in the business support matrix.

Another piece of information we needed to add to the business support matrix is the **customer type**. There exist significant differences in applications for business customers compared to private customers (i.e. consumers). Order taking for a private customer can be done by phone or a web interface. A corporate customer, submitting thousands of orders every day will ask for an e-business solution.

Product type can results in different requirements for the same capability. Delivering a letter requires a different application compared to delivering a parcel, compared to delivering a huge ocean freight container. Difference in size is not the only reason. It is also the difference in how the delivery is handled. Therefore, different applications are required, but still it is one and the same capability—delivery.

And last but not least, we needed to introduce something that is describing the **degree** of business support. Is this application fully supporting the business capability? Or only half, because they are missing requirements, or there are still some manual steps, or because there are more features required but we cannot extend the application for whatever reason?

Example 4.3 Figure 4.21 shows two examples for business support with different dimensions. The business support matrix is the same as above. Application **a3** is supposed to support business capability **c6**. It is not supporting this capability now. There is a plan (**to-be**) that this application will support the capability beginning May 2022.[11] If you remember the previous version of this business support matrix (Fig. 4.15), the business capability **c6** was a gap as there was no application support. The updated matrix in Fig. 4.21 now indicates the plan to close the gap with application **a3** on a **global** level. a3 is becoming the standard application for **c6** in the entire

(continued)

[10] Reasons for variant applications and common criteria are described in more detail in Sect. 3.2.

[11] Which is in the future when writing this textbook.

Example 4.3 (continued)
company. However, the application is only relevant for **business customers**. It is not intended for private customers. Its usage is also restricted to a certain product type, **production facilities**. But as **a3** is already about to be extended, the objective is to support this capability 100%.

To summarise the example: We identified the gap **c6** and made the plan to extend the application **a3** so that it fully supports the business capability. The change will need some time, so we are planning to support this capability starting May 2022 globally. It will be used by all organisational units for the interaction with business customers only, buying production machinery.

There can be several business support relationships between the same pair of application and capability. The example in Fig. 4.21 also shows the plan for using **a3** for the same capability for private customers (i.e. **consumers**) in **Europe** buying **home appliances** from **January**. Interactions with private customers is less standardised, so that the application only supports the capability to 80%.

This example demonstrates that the real business support behind one tick can be rather complex. In this case one tick already indicates two sets of details for the business support. The requirements for business customers do not apply for supporting consumers—and vice versa.

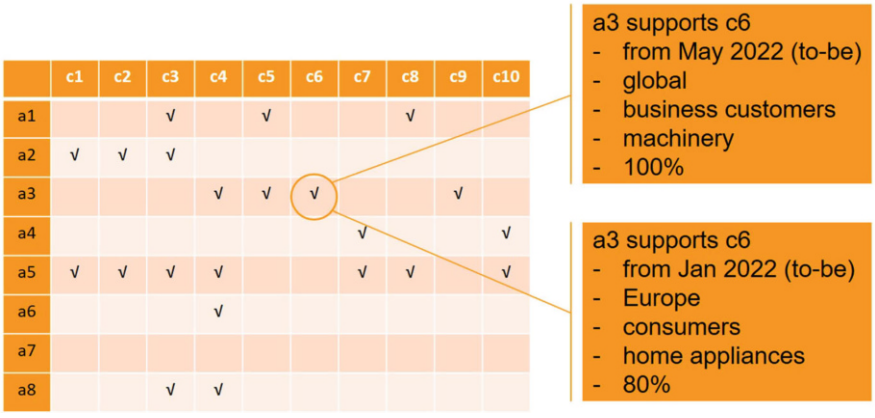


Fig. 4.21 Example dimensions for business support

4.3.5 Summary of Business Support Matrix

There are many more tools and visualisations available for EA. The chair of Prof. Matthes already performed two evaluations of tools showing the range of visualisation in [6] and [7]. Some researcher summarise existing visualisations together with a critical reflection (e.g. [8]) or propose their idea of a complete set of maps (e.g. [9, 10]). We will not discuss these in this book but rather refer to existing publications. Further references will be provided in Chap. 6 as part of the EA frameworks.

Why are we specifically looking at the business support matrix in this book? Because it is a powerful visualisation which can be implemented using a standard spreadsheet application. There are also not many publications presenting it in such detail. A more elaborate comparison of benefits and draw-backs is provided in Fig. 4.22.

It is a very simple tool allowing for high level analysis. It can be used to identify potential issues in the application support and lead to further analysis. Such analysis consists of drilling down into more details—provided in further documents or repositories. Analysis is not only restricted to the current state of your architecture. You can also make decisions for a planned to-be. You can even show intermediate steps for your planning if you want to change your applications landscape step by step. You can have several versions of the business support matrix for then showing also the changes over time (similar to the roadmap presented by Figs. 1.17 and 1.18 in Sect. 1.4).

Most corporate users already have a software tool available for creating and analysing the business support matrix. It is Microsoft Excel or any other spreadsheet application. Microsoft Excel (as an example for a spreadsheet application) supports several scenarios with the business support matrix:

1. creating the matrix from existing data by creating a pivot table
2. printing a poster of the matrix for visual analysis together with corporate stakeholders
3. performing visual analysis together with management by looking for empty rows/columns or clusters of ticks
4. calculating KPIs for applications and business capabilities (e.g. *size of an application or* magnitude *of redundancy)*
5. recording (potential) changes to the application landscape after analysis meetings

It provides an easy graphical visualisation as a table with ticks representing business support. It can be used during workshops hosted by the EA team, business experts or with top executives to indicate improvement potential for the application landscape. The matrix can be used to facilitate face-to-face workshops. You can print it. You can attach it to a wall in a room. You can have a group of people discussing it. And if you printed it, people can draw on it, make comments or highlight issues. A print-out improves interaction between people and can even be extended by sticky notes so that people add comments to the matrix. Having the matrix as a poster on

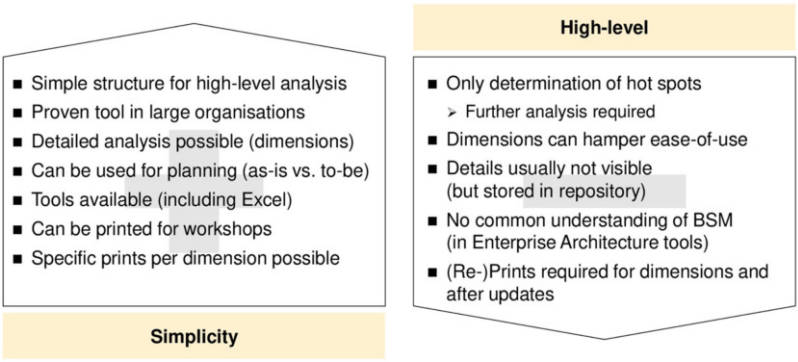


Fig. 4.22 Business support matrix—critical review

the wall makes workshops very interactive so that participants can come up with ideas on how to improve the EA.

Nevertheless, it is only a high level view. Even though it can be complemented by some details, more information is required for decision making. To show the five dimensions introduced above, can make reading the matrix cumbersome. If you show all the details on a hardcopy then it can get very complicated. Consequently, a poster should not contain all the information (applications, capabilities, dimensions) but only what is relevant for the corresponding stakeholder or purpose.

Introducing the business support matrix takes a long time as it relies on data and needs to be established within the corporation. It should also not be pushed by EA but rather be introduced as a tool that can also provide benefits for business people. One might start with simple versions or a matrix for one organisational unit only. If people perceive a value by using the matrix, it can be transferred to other units as well.

There is no standard definition for the business support matrix in the literature. You can find similar visualisations, sometimes by another name. It can also be created with any concept from business and application architecture (e.g. processes and data objects) as well as differing dimensions (e.g. create, read or update data objects). You will have to develop your own version of the standard business support matrix tailored to your organisation.

Using prints to facilitate workshops can lead to some time delay when posters need to be reprinted after a session. When reprinting too often, it may become a waste of resources (paper and print colours). Several versions of out-dated prints may pile-up in offices. Sometimes you need to update your information digitally and then throw away the print-outs.

4.4 Further Reading

There is a conference paper that describes the application of the business support matrix [5]. It was published at a conference on EAM in Stockholm (Sweden) in the year 2018. It will provide more details on the business support matrix, what you need to do in order to establish it, how you can analyse it, and which research questions are still open.

The theory we described in Sect. 4.2 is applied by the chair of Professor Matthes at the Technical University in Munich. This team has a number of publications around this topic. One of them we would like to recommend here is the EAM Pattern Catalogue [3]. In this paper they describe patterns for creating maps and visualisations for EAM. Some of their theory we describe here, like—the layering and the types of maps that are introduced there. In addition to that, Matthes and his team performed evaluations of EAM tools. These evaluations (cf. [6, 7]) are based on typical scenarios and provide plenty of examples for common visualisations.

The third title written by Op't Land, Proper, and others is titled *EA—Creating Values by Informed Governance* [2]. We already recommended this in the first chapter because this provides an overview on the purpose of EAM. They also talk about the analysis and typical objectives when analysing your EAM.

Finally the last book already recommend in the third chapter [9] is on how to describe EA—especially the application architecture. It is a book published by a researcher in Melbourne, who completes a lot of literature reviews and also research about views for EAM. His book, *The Practise of EA—A Modern Approach to Business and IT Alignment* is full of maps and types of maps that can be used within EAM. As we are only capable of discussing a very few within this course, whenever you are interested to check additional maps and visualisations, this book might provide a good starting point.

4.5 Summary

This is the end of the fourth chapter on EAM. We were dealing with the analysis of EA through visualisations, by using maps. We provided some theory on creating maps for EA with respect to analysis. We provided a more detailed view of the business support matrix indicates the business support and helps us with the identification of typical issues. However, the business support matrix is a rather high level tool that—after identifying typical observations—will require more detailed analysis and reasoning of those issues.

The next section will continue as shown in Fig. 4.23. After describing and analysing the EA, we now need to discuss:

- How can we manage the EA?
- How can we make sure that we have an organisational unit that will create all the maps and maintain them?
- How does the organisation look, which will then conduct the changes on the EA?

Fig. 4.23 Following next: managing enterprise architecture

It is not enough having the architecture captured in maps. We also need an organisation for planning and conducting changes on any architectural layer.

4.6 Exercises

Exercise 4.1 (Objectives for Analysis) You are owning a company that is specialised on delivering medical supplies to clinics and hospitals in Greater Melbourne. The current pandemic situation leads to an increase in demand and a shortage of materials from your providers. You are aiming at extending your business so that you also cover imports of supplies from international partners.

Against this background, you have to analyse your EA as you add new capabilities (e.g. import and customs processing) and will have to introduce new application systems for integrating new business partners (e.g. providers or government organisations). List concrete objectives for your analysis. Use the structure given in Fig. 4.2 on page 107 and define an at least two objectives for each category:

1. Strategic
2. Intrinsic
    a. Quality
    b. Cost
3. extrinsic
    a. Collaboration
    b. Compliance