Traceability

with

ENTERPRISE ARCHITECT

eaDocX document excellence

Traceability Matters

It's something which most BAs have been asked for. "*I need traceability*". But what does it mean, and how can we deliver it without spending our lives creating even more spreadsheets and reports for our stakeholders?

Talking to our **eaDocX** customers, what most people seem to mean by traceability is the ability to follow the thread through a project: both forwards and backwards, to see where ideas went (forwards) and where they came from (backwards).

Forwards Traceability

A common criticism of IT organisations and especially BAs - who are often the public face of a project - is that we collect requirements, constraints, and suggestions from stakeholders, but the system which gets delivered isn't what they expected. They can't follow those ideas through the development process, and see how they got changed, merged, delivered or de-scoped from the project.

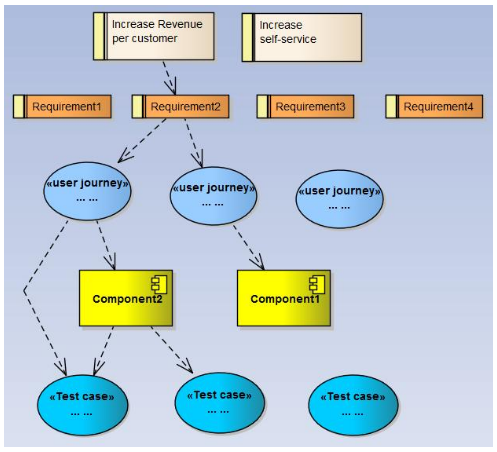
Backwards Traceability

This is a less common requirement, but still an important one. At the end of a project, when a solution is being tested, how can we tell where to concentrate the available testing resources? What's vital to the success of the project, and what's just 'other stuff'? If we can achieve traceability all the way back to the stakeholders' initial ideas, they we can see what *they* thought was important, and focus our efforts much better.

Capturing Traceability with EA

One of the strengths of Enterprise Architect (**EA**) is that we can use it to capture the full range of project ideas. Not just individual requirements and their priorities, but user stories and use cases, classes and components, test plans and test cases.

And we can also capture how those ideas changed over time, by keeping risks, issues and actions, which tell the story of the project, in **EA** as well. Connecting these ideas together can then deliver the traceability which our stakeholders and testers need.



What to Trace

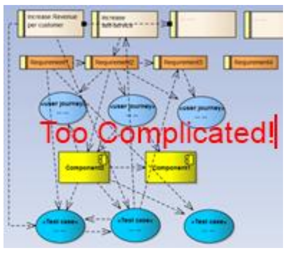
Each project will have a different idea of what 'traceability' means to them, but projects seem to have a few common ideas:

- **Start at the top**. Some ideas are the start of everything. They may be "High Level Requirements" or "Project Drivers", but there will be some. Your project may have 500 requirements, but there are probably a few higher-level ideas somewhere, from which those requirements all came.
- **Be realistic**. We've seen lots of really complicated models, where the traceability is recorded in infinite detail. But mostly, a simple 'X is related to Y' is enough. More doesn't seem to be better.
- **Stick to the core ideas**. If you follow this technique, you might be tempted to link everything to everything. This is just confusing. Decide what the minimum

traceability is, and implement that really well.

Demonstrating Traceability

EA will allow us to capture the raw traceability data, but we can't give complicated EA diagrams to our stakeholders:

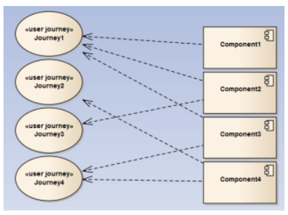


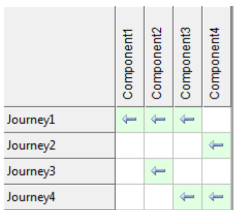
This diagram might be one where we create the traceability links in EA, but it's not for sharing!

Using eaDocX to show Traceability

This is where we can use the eaDocX Document Generator to show the traceability, but in a more readable way. A key capability of eaDocX is that you can make links between EA elements into document hyperlinks. Both forward and backward links. This makes it much simpler for readers to click their way through the document, following the traceability trail.

For example, let's say it's important to us to understand which User Journeys need which solution components. So, we've added these into EA, either via diagrams or with an EA relationship matrix – they show exactly the same information:





	Component1	Component2	Component3	Component4
Journey1	←	←	←	
Journey2				←
Journey3		←		
Journey4			←	←

Now, when we use eaDocX to generate a document, we can show this traceability, along with all the other traceability links we've created. See the example document on the next page, which was generated from EA using eaDocX.

Maintaining Traceability

The reason most projects don't make use of traceability information like this is because maintaining it is just too hard. If we maintained it manually, each time we added a new User Journey, or changed an existing one, we'd have to re-visit our traceability data.

But because all the information is saved in EA, and our documents are generated directly from that model using eaDocX, we just make the EA changes, and re-generate the document. eaDocX will read all the latest information from EA, and create a new version of the document, with all the traceability links refreshed.

An Example

Below is an example of what a document can look like using these ideas. It is some data taken from a simple EA model. In that model, we have created just two kinds of traceability links:

- From Actors to User Journeys, to show who is using what – a very common kind of link
- From User Journeys to the components in the solution which will either implement them or which will need to change. This helps us to check the scope of the change we're proposing to make.

The example document has been generated from the EA model, using eaDocX. The document shows these traceability links as hyperlinks within the document, so a reader can trace through the document just by clicking the links.

EA and eaDocX Sample Document

This example shows how EA and eaDocX can combine to create forward and backwards traceability hyperlinks within a generated document. All the content of this document, including this explanation, come from an EA model.
All the data is presented in tables, but eaDocX allows for a wide variety of different formatting styles.
Also, just for this sample document, we have chosen not to include any of the EA diagrams, to save space. This is also an eaDocX option.

1. Analysis Summary

1.1 Actors

These are the Actors, which are used across many different User Journeys.

Name	Description	User Journeys
Actor1	Actor1 is … …	Journey1, Journey2
Actor2	Actor2 is … …	Journey2, Journey3, Journey4

Comment [I1]: In this sample document, we have decided to print a list of the User Journeys in which each Actor is involved. This way, if there is a Actor who doesn't take part in any Journeys, we can see this easily – that's probably an error in our analysis.

1.2 User Journeys

This is a summary of all the user journeys for the project, showing which are used by which actors, and which existing components are needed by the Journey.

Comment [I2]: For each User Journey, we can show both the backward links to the Actors who use them, and the forward links to the solution Components which implement them. We might also include traceability to other EA data, such as requirements or people, system releases or risks & issues.

Name	Details	Used by Actors	Needs Components
Journey1	As a user, I want to … …	Actor1	Component1, Component2, Component3
Journey2	As a user, I want to … …	Actor2, Actor1	Component4
Journey3	As a user, I want to … …	Actor2	Component2
Journey4	As a user, I want to … …	Actor2	Component3, Component4

1.3 Solution Components

This is a list of the solution components which are currently deployed, along with the User Journeys which need to use them

Name	Used by User Journey
Component1	Journey1
Component2	Journey1, Journey3
Component3	Journey4, Journey1
Component4	Journey4, Journey2

Comment [I3]: This uses the same single EA connection between a User Journey and a Component which was printed in the previous table. Here, we can use it to get an idea of which Components are the most important – used by the largest number of User Journeys.