White Paper

Control the Costs of your next SAP Upgrade or Enhancement Project with the Scope and Effort Analyzer

WP0219 | December 2015

Ben Parris

Ben Parris is a SAP-certified ALM expert who specializes in SAP Solution Manager and associated 3rd party ALM tools. He works for Rapid ERP (www.rapid-erp.com); providing high quality, innovative SAP consulting services to clients who want to maximize the value that ALM can bring to their SAP operation and drive additional ROI from their SAP investment. Rapid ERP provides the marketplace with the industry's most experienced and knowledgeable SAP ALM consultant.

SAP customers have long since been encouraged to keep their SAP solution up to date, not only to ensure stability of their SAP systems and the business processes that run on them, but also to take advantage of new enhancements delivered on a frequent basis by SAP. Indeed, the move to an Enhancement Package (EhP) concept for SAP's major products, including SAP ERP and SAP CRM, was designed to encourage this further, by enabling new functionality to be technically implemented and the new functionality 'switched on' in a controlled fashion at a later date.

However, whilst the acquisition and implementation of new functionality and technical upgrades has been streamlined to a degree with this concept, there are still some barriers to adoption – largely stemming from the costs of implementation of upgrade and EhP projects still being hard to swallow for many organizations. As a result, many organizations running SAP have been reluctant to upgrade and end up languishing on older releases, not making the most of their investment in SAP and the investments that SAP have made in improving their products over time.

Why is it, that in today's modern world, costs of upgrading key business systems still require significant investment? And, more importantly, what can we do to alleviate these challenges?

Key Challenges

The following figure was taken from a study that SAP undertook to understand customer's perspectives on the primary challenges in running SAP upgrade projects[1]. The results are consistent with the challenges that we hear from our customers when planning these lifecycle events.

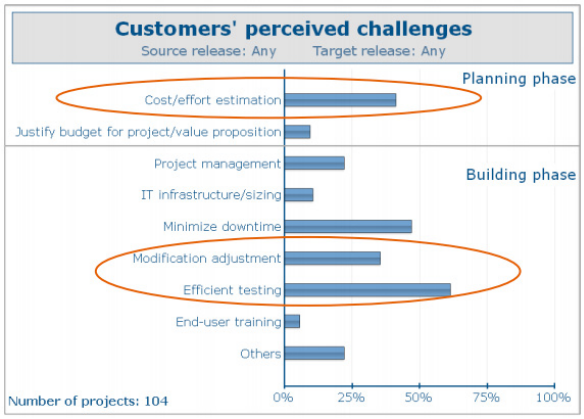

Figure 1: Results of SAP Study into customer's primary challenges in performing SAP upgrades

Let's take a closer look at these challenges:

1. *Difficulty in planning effectively, including accurately predicting the required effort across all phases of the project, including associated costs.*

Until some evaluation is performed (typically by implementing a sandpit system and performing a trial upgrade on this system) it is very difficult to understand the impact of an upgrade on the operation of the system at a technical level. What has changed in SAP standard objects? What impact will the upgrade have on my customizations and custom developments? Until the impact of the upgrade is understood at a technical level, it is nearly impossible to predict the time and effort required to remediate affected developments, test the solution and rectify any testing defects.

In addition, the provision of a sandpit environment itself is a costly activity. The system needs to be provisioned, upgraded and the assessment of the impact calculated.

2. *Identifying and remediating impacted custom developments and modifications*

Typical SAP solutions have an element of customization implemented. This is one of the reasons that SAP is such a popular platform – it can be easily tailored to a customer's specific requirements in a controlled and supportable fashion. However, this causes a significant impact on the cost of upgrade projects. Each customization needs to be assessed

[1] Source: SAP EHP Experience Database, 08/2012, Link: http://service.sap.com/ehp-db

to understand whether it has been impacted at a technical level by the upgrade activity. Many customizations reference SAP standard objects which may have changed during the upgrade and therefore adversely impact the execution or performance of the developments after the upgrade event.

To counteract this, SAP customers often employ a 'fix on fail' approach to customizations, whereby they are picked up and remediated only after a fault has been identified during testing. This approach makes it very difficult to plan the effort required for this phase, leading to a large unknown when planning the project timescales and budget requirements, and actually poses a risk to the project as there could be some customizations that are not known about and therefore not explicitly tested prior to go-live.

3. *Efficiently testing the upgraded solution.*

Testing. It's a necessary cost of any IT project, but more so in terms of an SAP upgrade project. The primary function of the test phase in technical upgrade projects is to ensure that there have been no negative impacts to business operations as a result of the upgrade implementation – we're just checking that everything still works!

Whilst this is clearly valuable to the business in terms of mitigating any adverse impact to operations, the more efficient this phase of the project, the better. Typically, full regression testing is carried out, which even in the simplest solutions typically contributes approximately 60% of the total upgrade project cost . Any saving that can made here will have a significant reduction on the overall project cost.[2]

Now that we understand the main challenges surrounding SAP upgrade projects, what can we do to mitigate them? There is a tool that can help us, and it's called the Scope and Effort Analyzer (SEA).

Introducing the Scope and Effort Analyzer

The Scope and Effort Analyzer tool is the answer to these challenges. We can use the tool to analyze the impact of any given maintenance activity ahead of starting any work on the project. It really is a powerful tool to help plan and execute an upgrade project, and here's why:

Identify the Impact

The SEA tool will look into how you use your SAP solution, pre-upgrade, recording user activity and the execution of both SAP standard and custom technical objects to an incredibly detailed level. In fact, this analysis goes way beyond the executable object layer, also including the entire technical call stack in the analysis provided by the Usage and

[2] Martin Riedel (2009). Managing SAP ERP 6.0 Upgrade Projects. Germany: Galileo Press. 104-105.

Procedure Logging (UPL) statistics. The first result of this analysis is that you now know, to a high level of accuracy, which technical objects are actually used within your SAP solution, enabling us to focus on these objects when it comes to the remediation and testing activities in the project.

The arguably greater benefit of the analysis is that we can use the tool to accurately predict the impact of the proposed technical changes on custom developments. To enable this, the list of objects to be imported as part of the upgrade is provided to the analysis (via a standard Maintenance Optimizer transaction). The SEA tool then uses this list of objects and the detailed execution statistics to understand which custom developments reference objects that will be changed by the upgraded code; to such a degree of accuracy that even the impact to individual developments is returned – for example, the tool can even tell us if a specific custom development will have a syntax error after the upgrade. The result is an identification of all impacted custom developments that are actually used within the SAP solution, without having to perform manual evaluation of a sandpit system, which can be passed to the development community as a worklist. Immediately, this provides a significant cost saving for the project – we no longer need to invest in a sandpit system – and we have reduced one of the big unknowns in a traditional upgrade project of not knowing which developments will be impacted.

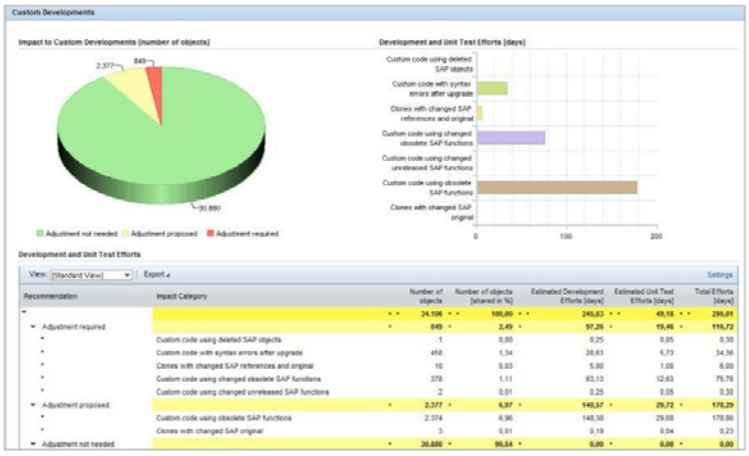

Figure 2: Custom Development Impacts Results in the SEA tool

The same analysis is also performed for modifications to SAP objects, enabling the prediction of SPDD and SPAU lists for the upgrade. As with custom developments, there is a differentiation between used and unused object, enabling the remediation scope to be reduced without introducing significant risk.

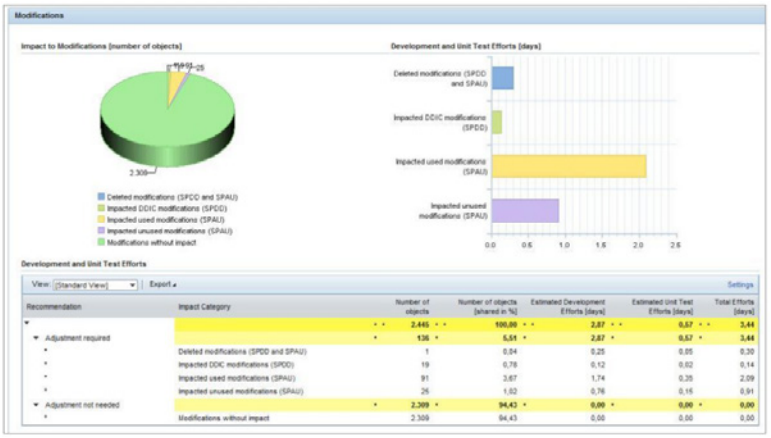

Figure 3: SEA Results for Modifications

In addition, as an input to the analysis we can also specify typical effort durations for across the types of remediation activities and impact categories that the tool identifies. This allows for a prediction of the remediation effort to be generated by the tool, which can help scope and cost this aspect of the project during the planning phase.

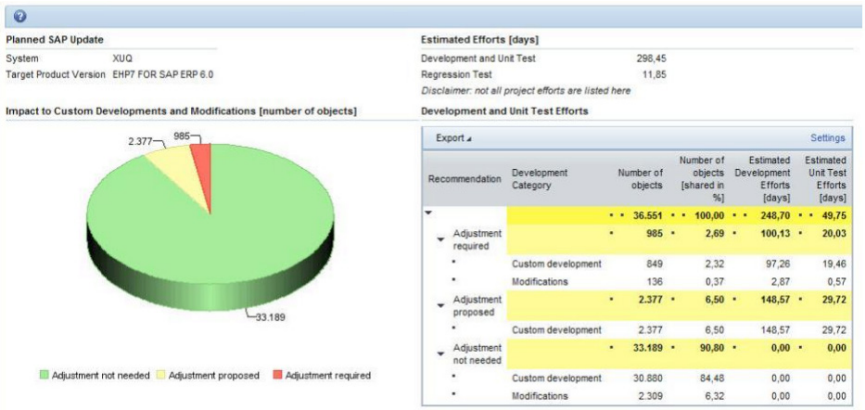

Figure 4: Effort Predictions in SEA Analysis Results

Optimize Your Testing Strategy

So, we've understood the technical impact of the upgrade project, but what about the impact to the business processes? And, more importantly, can we use this technical information to better inform us about the testing phase of the project?

The answer is a resounding 'yes'…

As we now have a detailed understanding of how the technical objects will change in the system, we can map this impact to our business processes. Here's where having well formed, accurate process definitions within SAP Solution Manager can be a significant advantage, as you will have the basis for this analysis already defined. In the process definitions executable objects are assigned to business process steps, to define how the processes are executed technically. This provides us with the mapping required by the SEA tool to perform the analysis and identify the impact to our business processes, and associated test cases.

However, if you haven't yet invested in defining your business processes in SAP Solution Manager, all is not lost – the tool will generate a process structure for you automatically (via the aforementioned UPL stats). This provides enough of a foundation for the SEA analysis, but it is recommended to supplement this generated structure with your own business context.

But what does this actually give us – well, in short, it's the ability to firstly predict, then optimize the testing scope for the project.

Consider that a full regression test would test every object in the system, whether or not it is actively used and whether or not it is actually impacted by the upgrade. Now consider that we have the data to hand to understand which of these objects are a) actually used and b) have actually been impacted by the upgrade. Furthermore, we have a definition of our business processes which maps technical objects to testing scenarios. With this information, the SEA tool can automatically rank the test cases in order of greatest coverage of changed technical objects in the system and provide a cockpit to analyze various 'what if' scenarios concerning the testing phase.

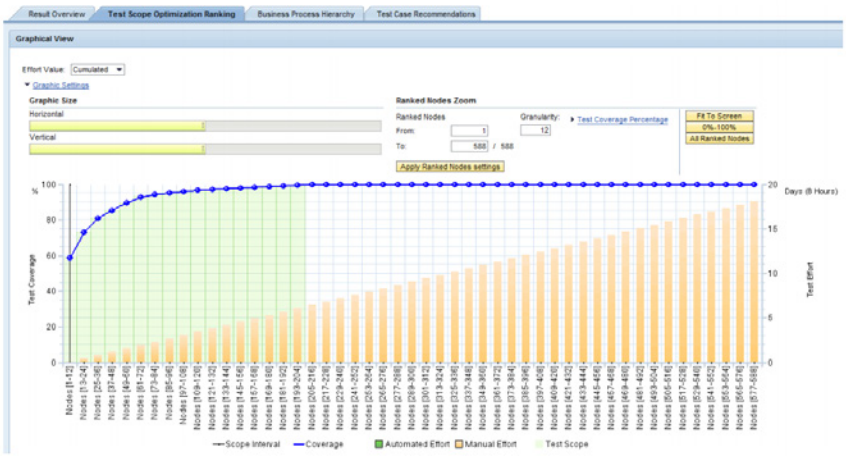

Figure 5: Test Scope Optimization Results in the SEA tool

In the figure above, the horizontal axis represents individual business processes, with the vertical representing the 'Test Coverage' – i.e. the percentage of changed technical objects covered by each testing scenario. The blue line shows the cumulative Test Coverage, and the orange bars show the cumulative effort for the testing to up to that point.

The key information provided by this output is the green shaded area. This area represents where the Test Coverage is below 100%. The area to the right of this, with the white background, shows all testing scenarios which would be performed after the Test Coverage has reached 100% - i.e. after you have already tested everything that has changed in the system. You can reasonably expect that these tests will therefore provide very little value to the project – any tests performed from this section would simply be re-testing objects already tested by the tests in the green shaded area.

What generally astounds SAP customers is that the average optimized test scope provided by the tool is typically in the region of 30-50% of what they would consider to be a full regression test cycle. Given that the testing phase is a significant contributor to overall project cost, this can be incredibly valuable information and enable a major reduction in the cost of implementation.

However, we all know that some business processes are critical to operations, and therefore must be tested explicitly no matter what the tool says. To accommodate this, we can specify that these critical processes must be tested, and then allow the tool to optimize the scope from that point.

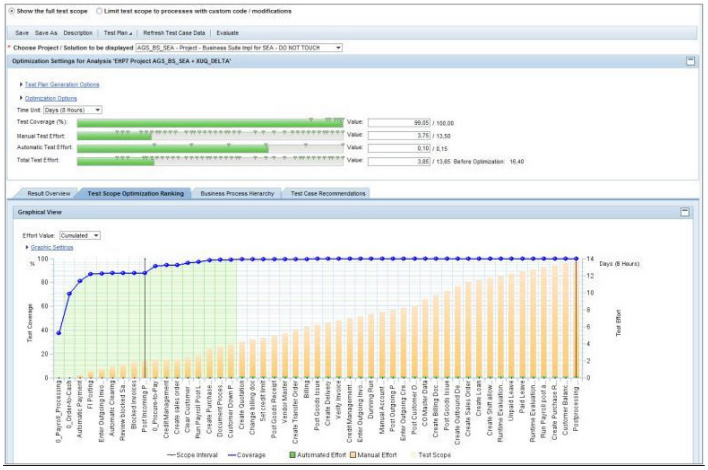

Figure 6: An optimized test scope having specified critical business processes that must be tested

In the example above, the critical processes are specified to the left of the vertical black line. Whilst not as efficient as an unbound optimization, there are still significant savings to be made over a full regression test.

Alternative test optimization strategies supported by the tool include:

- *Risk Based Testing* – Accepting a Test Coverage percentage lower than 100% of the changed technical objects can realize even greater test effort savings, providing that you are willing to accept the risk that not all changed objects will be tested and adopt an appropriate mitigation strategy (e.g. fix on fail in integration testing or even in production, if the objects are of low operational importance).

- *Time-boxed Testing* – By limiting the amount of testing effort that can be afforded to the project (i.e. to conform to budgetary constraints), you can understand the Test Coverage that can be achieved and understand the risk to the project by not testing what is left outstanding.

Benefits

The benefits of using the SEA tool directly address the key challenges SAP customers face when embarking on upgrade projects of any kind:

- Better planning of SAP upgrade projects, due to being informed up front about the impact ahead of making any technical changes to the landscape
- Make informed decisions about the testing strategy and approach to employ on a project by project basis
- Significantly reduce overall project costs by removing the need to implement sandpit systems and employ costly regression testing principles
- Reduce the risk to the project by ensuring that all impacts are known upfront and having confidence that testing has covered either all changed technical objects, or at least all critical objects.
- Direct integration with other SAP Solution Manager functionality – once you have decided on your test scope, automatically generate a test plan directly from the results and execute your testing via the test management workbench, or using integrated third party tools.

Conclusion

The SEA tool should now be the starting point for any Enhancement Package implementation or release upgrade project within SAP environments. The level of detail provided by the tool will help plan the project effectively, with no more guesswork, and can even be instrumental in providing the business cases or feeding a cost-benefit justification.

What's more, this tool is available with SAP Solution Manager and is therefore free to use for SAP Enterprise Support or Premium Engagement (PSLE, Max Attention, etc.) customers! The costs to getting the tool up and running are also low, so there really is no excuse for not utilizing the SEA tool to implement faster, test smarter and plan better on your next SAP upgrade or EhP implementation project.

References

SAP AG. (2012). SAP EHP Experience Database [online]. Available:

http://service.sap.com/ehp-db

Last accessed 10th November 2015

Martin Riedel (2009). Managing SAP ERP 6.0 Upgrade Projects. Germany: Galileo Press. 104-105.

Orbus Software
3rd Floor
111 Buckingham Palace Road
London
SW1W 0SR
United Kingdom

+44 (0) 870 991 1851
enquiries@orbussoftware.com
www.orbussoftware.com

orbus software