The hassle-free way to maintain mappings

How to maximise 'maintainability' right from the start

Nobody wants to maintain

Everybody wants to build, but nobody wants to maintain, as the saying goes. But like it or not, we all spend a lot of time carrying out maintenance of one kind or another. This is just as true in our area of specialisation – message transformation – as any other. Both internal business changes and external standards updates regularly force existing mappings to be revised.

Since maintenance is an inevitable fact of life, can we identify some ways to make the job easier? Yes we can. At Trace Financial we thought long and hard about these issues when we were designing our Transformer solution. In this article we outline the key points that lead to a hassle-free maintenance cycle.

The right way to start

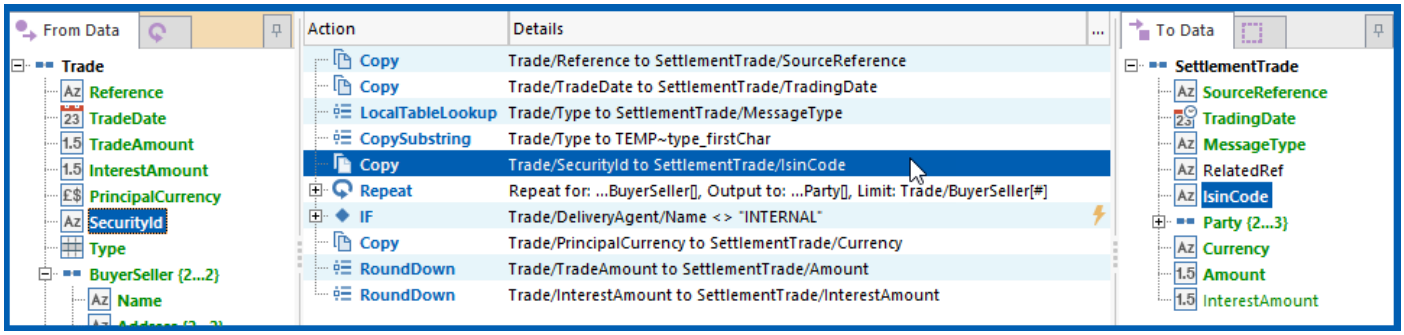
The first step in changing an existing mapping is to understand what it currently does. In a traditional development environment this means finding the original specification document, which was created some time before the mapping itself by an analyst. Unfortunately those original specs are often either lost or no longer in sync with the actual mapping. This forces the analyst into a preliminary investigation to try and find out what the mapping does now. Developers will need to get involved if the mapping is implemented in code or script form.

There is a better way. Our approach is to build in what we call 'maintainability' right from the start. This somewhat cumbersome word points to some real and practical qualities that should be built into the

Trace Financial Limited, 224-232 St John Street, London EC1V 4QR
Tel: +44 (0) 207 825 1000 email: info@tracefinancial.com web: www.tracefinancial.com

mapping in the first place. They help to make sure that any mapping you create is understandable – logically and clearly laid out, easy to read at a non-technical level and so on.

With Transformer, the mapping is fully self-documenting. Unlike some mapping tools we do not use graphic lines to connect source and target fields, because in any reasonably complex mapping that approach can often produce a tangle of lines most of which begin or end above or below the portion of the mapping being viewed. Instead of this we show the user a list of mapping actions, with the source field(s) to the left and the target field(s) to the right of each mapping action:



A typical Transformer mapping

Select one mapping action and the relevant fields on either side are highlighted. Technical syntax details are kept out of the way and the fields and mappings are described in business-friendly terms. In this way it is easy for a business analyst to 'read' a Transformer mapping and understand what it means.

Minimising the impacts

For any given change it is essential to understand the full impact on relevant mappings, to make the necessary set of changes in a coherent way, and to make sure no other changes are accidentally introduced.

Much mapping maintenance work is driven by changes to external standards such as SWIFT MT or MX formats rather than by internally-driven business needs. This means that there usually has to be an initial assessment stage in which the primary aim is generally to minimise the impact on existing mappings.

At this point we can mention another way to build in maintainability – by maximising re-use. If we define a field (or higher-level component) just once and then re-use it in many places, it will be all the easier to see the impacts resulting from a change to that field or component. Re-use helps to make Transformer's 'Where Used?' function a valuable tool at this stage. It is able to highlight all of the message definitions and mappings that will need attention because a given field or component has changed in some way.

When the time comes to actually make the changes that have been identified as necessary it is important to keep the maintainability level high. In other words it must be possible to make changes in ways that preserve the original clarity of the mapping. In this respect it's important to note that there is no coding stage with Transformer even when the required data transformation is highly complex. Because there is no need to drop into programmatic code or scripting, projects remain clearly

Trace Financial Limited, 224-232 St John Street, London EC1V 4QR
Tel: +44 (0) 207 825 1000 email: info@tracefinancial.com web: www.tracefinancial.com

articulated and easy for all stakeholders to understand. Removing the old-fashioned 'spec handover' from analyst to developer in itself eliminates a major source of misunderstanding and delay.

Regression testing

After a mapping has been tested, Transformer saves both the input test data and the resulting output, along with the mapping itself. The test can then be repeated later in 'regression mode', which highlights any differences between the current test's output and that produced previously. Transformer test scripts can also be run automatically, e.g. by a version control system, after a new version of a library is checked in. This greatly speeds up the task of checking the continued correctness of a large number of message mappings following minor updates to just one or two elements.

Summary

If maintenance is done badly it gets progressively harder to do over time. In a large organisation with hundreds of mappings it can become a nightmare. But when you can create mappings so that they possess the quality of self-documenting clarity that we call 'maintainability', the task starts to look much more achievable. When you also have the means to rigorously assess, control and check the impact of changes, then maintenance becomes an efficient procedure, releasing resources for other projects.

Transformer is designed to build in maintainability right from the start, and has all the tools you need to carry out a high quality maintenance task in far less time than other methods.

Trace Financial Limited, 224-232 St John Street, London EC1V 4QR
Tel: +44 (0) 207 825 1000 email: info@tracefinancial.com web: www.tracefinancial.com