Applying Requirements Templates in Practice: Lessons Learned

The beauty of a simple natural language (NL) template is that it can be applied by anyone with no tool support. Whilst many requirement notations rely on the use of expensive specialist tools, a simple NL template can be used on the back of a cigarette packet, in a simple application such Word or Excel, or with dedicated tool support. This makes templates a very cost-effective improvement to requirements practice.

The simplicity of these templates means that they can be learned in an hour and people will immediately write better requirements. This seems to be true for both novices and experienced requirements authors. Indeed, I have received a lot of positive feedback from experienced authors, who will often revisit and improve an existing set of requirements once they start using the template.

To help embed the learning, some coaching and support is beneficial. I encourage first-time users of the template to write 10 requirements and then let me review them. With the review comments in mind, I ask them to write another 10. Using this type of approach, any systematic errors can easily be avoided so that when a full document is produced it will contain less requirements errors. This also makes document review easier and reduces the need for rework.

In practice, requirements are almost always written iteratively. Authors will usually start with a subject about which a requirement may be needed. For example, for an aero engine, subjects could include Noise, Vibration, Temperature, Weight and Reliability. Authors will use these seeds to draft and then improve requirements, potentially several times as development proceeds. It is important that stakeholders understand this need for iteration and do not expect to get every requirement right first time.

Once using a template, authors will begin to use the style and language of the template, automatically fitting emerging requirements into the template structure without even realising it. Surprisingly quickly, using a template will improve the quality of first draft requirements. A well-formulated template therefore serves to enhance the capability of requirement authors.

The structure of a requirement written with a template tends to help with requirements decomposition. This is because the clauses of a system requirement will often suggest the need for an element of functionality that has to be provided at a sub-system level. For example, where a system level requirement is triggered by a change in temperature, there must be some means of detecting that temperature change. This will yield a derived requirement at sub-system or component level to measure temperature.

NL templates provide guidance for writing each individual requirement more clearly, but they can also help with requirements coverage. One example is in requirements pairing, such as where a requirement for wanted behaviour will naturally imply the need for some requirements to cover associated unwanted behaviour. Similarly when a requirement is written for normal operation, authors will be likely to consider the behaviour required in back-up operation. Examples for an aero engine would be a requirement for engine dedicated generator power supply and a corresponding requirement for aircraft back-up power supply.

Scenarios are a useful mechanism for grouping related requirements into logical chunks of functionality. NL templates work particularly well with scenarios, which also helps to ensure requirements coverage. Scenarios can be used during requirements discovery, as the basis for structuring a requirements document and during system test [5].

Simple NL templates obey one of my key rules for stakeholder engagement; to keep things as simple as possible. Whenever I introduce an approach I do so in small steps, without explicitly labelling it as a new technique. For example, I would avoid the use of terms such as syntax or swimlane diagram, which may not be commonly understood. Instead I would use more everyday language like style or flow chart. Over time, more terms can be introduced, adding rigour to techniques and models, which will incrementally build stakeholders' knowledge and capability.

Human beings are naturally resistant to change which can hinder new initiatives.

Therefore in my experience it can be best to introduce an NL template by stealth. A simple NL template is a lightweight improvement that can be introduced without an official launch. When introduced quietly and allowed to grow and spread by word of mouth, an NL template can significantly improve requirements practices within an organisation.

The purpose of a requirements document is not to be an entertaining read. The purpose is to make it as easy as possible for the reader to understand what the system must do. If a requirement must be read several times and the meaning is still unclear, then it is a poorly written requirement. One key benefit of NL templates is to drive consistency into requirements. This makes each requirement more alike and therefore easier to understand, since humans are very good at pattern recognition. However, this consistency makes reading a requirements document rather repetitive. I do not see this as a problem, since I doubt that anyone has ever read a requirements document for pleasure.

We developed our template at Rolls-Royce because the majority of people write most of their requirements in NL. However, some requirements are better expressed in other formats, such as truth tables, graphics, mathematical formulas or Boolean algebra. It can actually be counter-productive to write these requirements using NL, even with the aid of a template. So, my final piece of advice is to accept and embrace some diversity of requirements expression. A document may include 95% NL requirements, but the other 5% will be consciously documented using other more suitable notations.

Alistair Mavin (Mav)

Mav is an Independent Requirements Specialist based in the UK and is the lead author of the Easy Approach to Requirements Syntax (EARS) natural language requirement template [1, 2, 3, 4]. EARS was developed whilst Mav worked at Rolls-Royce PLC. EARS and MASTER were generated independently, but both build on similar work and share the same basic principles. EARS is a single template, the application of which produces requirements in a small number of patterns. In this Expert Box, Mav shares some insights and lessons learned from applying this type of natural language template in practice. More details on EARS can be found at www.alistairmavin.com and you can contact Mav at mav@alistairmavin.com

1 “Easy Approach to Requirements Syntax (EARS)”, Mavin, A., Wilkinson, P., Harwood, A. and Novak, M., Proceedings of RE09, IEEE, August 2009.

2 “BIG EARS (The Return of Easy Approach to Requirements Syntax)”, Mavin, A. and Wilkinson, P., Proceedings of RE10, IEEE, September 2010.

3 “Listen, then use EARS”, Mavin, A., IEEE Software, IEEE, March 2012.

4 “Listens Learned (8 Lessons Learned Applying EARS)”, Mavin, A., Wilkinson, P., Gregory, S. and Uusitalo, E., Proceedings of RE16, IEEE, September 2016

5 “Scenarios, Stories, Use Cases - Through the Systems Development Life-Cycle”, Alexander, I. & Maiden, N. (eds), Wiley, 2004