Research Topic Handout

Daniel Szymczak
McMaster University

June 28, 2018

Note: This handout was created as a companion-piece to my supervisory committee meeting presentation for June 28, 2018

1 Too much duplication!

In any piece of software, the same knowledge will appear across many different artifacts. Manually entering this knowledge multiple times increases the potential for errors to occur.

From our case study on a fuel pin in a nuclear reactor, we see h_g – a symbol which appears in the Software Requirements Specification (SRS), the Literate Programmer's Manual (LPM), and the source code.

The following table shows a (simplified) definition of h_g taken from the SRS.

Number	**DD1**
Label	h_g
Equation	$h_g = \frac{2k_c h_p}{2k_c + \tau_c h_p}$
Description	h_g is the gap conductance ...
...	

$$h_g = \frac{2k_c h_p}{2k_c + \tau_c h_p} \tag{1}$$

The defining equation for h_g also appears in Equation 1, taken from the LPM, and it appears again in the corresponding C code:

```
double calc_hg(double k_c,double h_b,double tau_c)
{
 return (2*(k_c)*(h_p)) / ((2*(k_c)) + (tau_c*(h_p)));
}
```

This kind of duplication is all too common and writing one piece of information in multiple places is tedious and time-consuming. Wouldn't it make more sense to encode a definition once and automatically reuse it wherever required in our software artifacts?

2 Inter-/intra-artifact consistency

3 (Re-)Certification is expensive

Software certification is necessary by law in certain fields, particularly in safety-critical applications. Certifying bodies exist across domains and each have their own list of requirements. Looking at some examples [1, 2, 3, 5] there are many common requisite documentation artifacts including, but not limited to:

- Problem definition
- Theory manual
- Requirements specification
- Design description
- Verification and Validation (V&V) report

Overall certification is an expensive process. The exact costs can be hard to estimate [4], but many person-hours are spent working on ensuring full traceability and consistency between software artifacts. Getting re-certified should anything need to be updated is a similarly long and costly process as all artifacts must be updated and re-verified manually.

While we are not proposing to automatically verify our software artifacts, we want to relieve the burden of artifact creation. What if instead of manually maintaining each artifact, we could automatically generate them?

References

[1] Center for Devices and Radiological Health, CDRH. General principles of software validation; final guidance for industry and FDA staff. Technical report, US Department Of Health and Human Services Food and Drug Administration Center for Devices and Radiological Health Center for Biologics Evaluation and Research, York, England, January 2002.

[2] CSA. Quality assurance of analytical, scientific, and design computer programs for nuclear power plants. Technical Report N286.7-99, Canadian Standards Association, 178 Rexdale Blvd. Etobicoke, Ontario, Canada M9W 1R3, 1999.

[3] CSA. Guideline for the application of N286.7-99, quality assurance of analytical, scientific, and design computer programs for nuclear power plants. Technical Report N286.7.1-09, Canadian Standards Association, 5060 Spectrum Way, Suite 100, Mississauga, Ontario, Canada L4W 5N6, 1-800-463-6727, 2009.

[4] J. Hatcliff, M. Heimdahl, M. Lawford, T. Maibaum, A. Wassyng, and F. Wurden. A software certification consortium and its top 9 hurdles. *Electronic Notes in Theoretical Computer Science*, 238(4):11–17, 2009.

[5] U.S. Food and Drug Administration. Infusion pumps total product life cycle: Guidance for industry and fda staff. online, December 2014.