University of Zurich^UZH

Requirements Engineering II

Assignment 3

Traceability and Requirements Evolution

Prof. Dr. Martin Glinz
Dr. Samuel Fricker
Cédric Jeanneret

I. Tasks

- Read the mandatory items in the reading list
- Be prepared to answer the questions given below in class
- Prepare a 15 minutes presentation (5-10 slides) on the theme assigned to your course group. Browse/read additional papers and/or web pages where necessary.
- Generate a requirements-to-stakeholders traceability matrix for the requirements document that you produced in exercise 2 of Requirements Engineering I.

II. Reading List

Mandatory reading
Traceability was first treated in [Gotel 1994]. [Jarke 1998] and [Dick 2005] further motivate and introduce traceability, while [Ramesh 2001] establishes reference models for it.

Theme-specific reading
[Egyed 2002], [Ben Charrada 2010]: Traceability and Software Execution.
[Hayes 2006], [Cleland-Huang 2007]: Traceability and Information Retrieval.
[von Knethen 2003], [Jönsson 2005]: Impact Analysis with Traceability.

III. Questions

- What is requirements traceability?
- What is the benefit of requirements traceability and what does it cost?
- How can one establish and maintain traces?
- Did the understanding of traceability evolve over time?
- What is the role of tools?

IV. Themes for Presentation

Themes will be assigned by the assistant who tutors this course; your group can apply for a theme.

A. Traceability and Software Execution

How can traceability links be generated by observing the execution of software? How can high-level tests be used to propagate changes in source code to requirements?

B. Traceability and Information Retrieval

How can traceability links be generated with information retrieval techniques? Can and should humans be replaced for defining traceability?

C. Impact Analysis with Traceability

What is post-requirements traceability used for? How can and should such information be maintained?

References

Ben Charrada, E., M. Glinz (2010). An automated hint generation approach for supporting the evolution of requirements specifications. *Joint ERCIM Workshop on Software Evolution and International Workshop on Principles of Software Evolution (IWPSE-EVOL '10)*, Antwerp, Belgium. 58-62.

Cleland-Huang, J., R. Settimi, E. Romanova, B. Berenbach, S. Clark (2007). Best Practices for Automated Traceability. *IEEE Computer* **40,** 6 (Jun. 2007). 27-35.

Dick, J. (2005). Design traceability. *IEEE Software* **22**, 6 (Nov. 2005). 14-16.

Egyed, A., P. Grünbacher (2002). Automating requirements traceability: Beyond the record replay paradigm. *17th IEEE International Conference on Automated Software Engineering (ASE 2002)*, Edinburgh, UK. 163-171.

Gotel, O., A. Finkelstein (1994). An Analysis of the Requirements Traceability Problem. *1st International Conference on Requirements Engineering*, Colorado Springs. 94-101.

Hayes, J. H., A. Dekhtyar, S. K. Sundaram (2006). Advancing Candidate Link Generation for Requirements Tracing: The Study of Methods. *IEEE Transactions on Software Engineering* **32**, 1 (Jan. 2006). 4-19.

Jarke, M (1998). Requirements Traceability. *Communications of the ACM* **41**, 12 (Dec. 1998). 32-36.

Jönsson, P., M. Lindvall (2005). Impact Analysis. In Aurum, A., C. Wohlin. *Engineering and Managing Software Requirements*. Springer. 117-142.

Von Knethen, A., M. Grund (2003). QuaTrace: A Tool Environment for (Semi-)Automatic Impact Analysis Based on Traces. *International Conference on Software Maintenance*, 2003. 246-255.

Ramesh, B., M. Jarke (2001). Toward Reference Models for Requirements Traceability. *IEEE Transactions on Software Engineering* **27**, 1 (Jan. 2001). 58-92.