Making Sense of Technical Information Quality — A Software-Based Approach

By Anna Wingkvist, Morgan Ericsson and Welf Löwe

MEASURING THE QUALITY OF TECHNICAL DATA DEPENDS ON DEVELOPING MODELS FROM WHICH METRICS CAN BE EXTRACTED AND ANALYZED. USING AN OPEN SOURCE TOOL THE AUTHORS DESCRIBE ONE APPROACH TO THIS.

Technical information, such as user manuals or maintenance documentation, is an important part of the product experience, no matter if the product is a mobile phone, a warship, or a software system. User manuals often constitute the first line of support when users need help with a problem or want to learn more about the product. Maintenance documentation can be critical when there is a need for updates or repairs.

The quality of technical information is important, since poor quality can reduce its' usefulness and effectiveness. Studies have also shown that the quality of technical information influences the perceived quality of the actual product and the manufacturer [7].

In order to manage the quality of technical information, it needs to be defined in such a way that it can be measured in meaningful ways [9]. There have been several attempts to develop guidelines [3] or models [1, 5] for assessing technical information quality. Few of these quality models are designed to provide automatic assessment. They are often ill defined, or incomplete, in terms of what and how to measure [11].

Based on existing efforts, a top-down approach using a complete and automated quality model for technical information is not feasible. In this article, we suggest a bottom-up approach where technical information is automatically assessed to produce quality indicators. These indicators are communicated to a human expert by interactive visualizations, and the quality model is formed by judgment and experience.

To illustrate the bottom-up approach, we used the Open Source Software (OSS) Java tool, VizzAnalyzer[1]. It was originally designed and implemented as a software quality assessment tool, but the flexible architecture and the similarity of the two domains allowed us to easily modify VizzAnalyzer to support technical information assessment.

The rest of this article is organized as follows. We begin by describing the process and how the human expert will interact with the analyses and visualizations. We then describe the VizzAnalyzer tool and give two examples of how the process and tool have been used to assess quality.

1 The Analytical Process

The various models defined to assess the quality of technical information show that there can be a huge gap between an ideal definition of quality, an approximation of the quality requirements, and what can be automatically analyzed and measured. For example, the quality requirement "easy to understand" can be difficult, or even impossible, to approximate a set of automated analysis and measurements. A quality requirement such as "no spelling errors" can be supported by an analysis and metrics counting errors. Still, there is often the need for human intervention, since such analysis may produce false positives, especially for technical terms. There is also a risk that, as we include more requirements and corresponding analyses and metrics, the quality models grow increasingly complex, and more difficult to understand and use in practice.

Visual analytics is the science of analytical reasoning, facilitated by interactive visual interfaces that display automated analysis results [10]. It can be a helpful way to approach problems that can be difficult to solve using either machine analysis or human analysis, such as those due to size, complexity or fuzzy problem definitions. The combination of visualization, machine analysis and human expertise can help to reduce both the size and the complexity, typically by applying interactive visualizations. Further, Cook et al. [2] state that the goal of the sense-making process is two-fold: detect unanticipated results and confirm the expected. The value comes from being able to cross-reference several visualizations in order to gain greater understanding and highlight anomalies, changes, patterns, and relationships. These results are then

1 VizzAnalyzer can be downloaded from http://www.arisa.se.

presented and assessed to develop new insights about the data. The visual representations make it easy to quickly perceive the salient aspects of the data.

The bottom-up approach to technical information quality that we suggest is, in many ways, similar to visual analytics. We acknowledge that any quality model that exists is constructed by a human expert who explores the technical information and formulates questions against its' structure or content. This suggests a need for a highly interactive and flexible environment.

From the user's perspective, the process starts by importing technical information into the tool and creating an abstraction of it. The actual format, such as Microsoft Word or an XML dialect, should not matter. If needed, it should be easy to add support for additional data sources or file formats. Once the technical information is imported, the user should be presented with a set of predefined analyses and visualizations that provide an overview and insight into possible quality issues. The visualizations are interactive and easy to adjust, for example, with respect to layout, colors, etc. Analyses can be used to provide new views of the technical information that can be visualized in different ways.

The user interacts with the analyses and visualizations to pose and find answers to questions, respectively. In order to support a wide range of possible questions and answers, it should be easy to modify existing analyses and visualizations, or add new ones, if necessary. Given the interactive nature of both analyses and visualizations, they need to be as efficient as possible in terms of execution speed and resource use.

In our approach, visualizations serve as a medium for efficient communication and cooperation between the human assessment of quality and the automated analyses (as supported by metrics). This marriage of computation, visual representation and interactive thinking supports analytic reasoning about technical information quality.

2 VizzAnalyzer

VizzAnalyzer is a quality analysis tool. It was originally designed to assess software quality by applying quality metrics and performing analyses on source code. Here, we discuss how VizzAnalyzer can be used to assess the quality of technical information.

In Section 1, we argued for the need for tool-based support to assess information quality. An important aspect of the tools is flexibility. They should be easy to adjust and reconfigure to support new questions, and existing visualizations and analyses should be easy to reuse and modify.

VizzAnalyzer is a highly flexible tool that supports the human expert and allows the user to adjust and reconfigure the system by providing:

- A common repository that captures abstractions of documentation according to a common meta-model that is shared by all types of technical information
- A mechanism to define and adjust mappings from common, real-world documentation formats, such as XML dialects, to the common meta-model
- A mechanism to define and use analyses and quality metrics that can be reused
- Visualizations of the abstractions of the documents, the metric values, and the analyses

2.1 A Model of Documentation and its Meta-Model

In order to perform different analyses on, or to visualize different aspects of, technical information, it needs to be captured in a repository using a suitable representation and a common format. When a document is captured in the repository, certain information, such as layout, etc., may be omitted. This abstracted version of the document is referred to as the model, and each document in the repository is a model. The model is still close to the original document, and every possible model is described by a common meta-model. This common meta-model represents a set of elements and their relationships.

For instance, if we want to represent a structured document consisting of a single document that contains sections and paragraphs, the meta-model should contain a *Document* element, a *Section* element and a *Paragraph* element, as well as a *Structurally contains* relationship to express that a certain element is contained within another. For example, a **Paragraph** is **Structurally contained** within a **Section**. Note that the elements and relationships in the meta-model are general for all technical information. Since the meta-model should be able to describe any technical information, and we cannot be sure that a single instance of it is complete, it is left as a configuration parameter of VizzAnalyzer. The meta-model can be described using an Entity/Relationship (E/R) diagram, which is used to instantiate the software as, for example, classes and associations, or database schemas used to form the common repository.

2.2 Mapping Documents to a Meta-Model

A model of a document is instantiated from documents that are represented using, for example, one of the many common real world file formats such as plain text or dialects of XML, or from the storage engines of content management systems. This instantiation is a mapping from the actual format to the common representation used by the common repository, using the elements and relationships of the meta-model.

Technically, the mapping is performed by traversing the technical information in document-order and adding it to the repository by, for example, invoking the constructors of the repository classes accordingly. Predefined methods and classes do exist to traverse most of the common document formats, e.g., XML processing libraries, so implementing support for a new format is often only a matter of inserting the corresponding construction actions according to the visitor pattern.

For each new documentation format that is added to VizzAnalyzer, a mapping must be defined. There may be a need to extend or change the meta-model to accommodate the format. This may require some programming, but based on our experience it is a simple task that requires less than a person-day. Note that this only needs to be done once for each new format.

2.3 Analyses and Metrics

One of the main purposes of the common repository and the common meta-model is to allow for analyses and metrics to be defined within the models (i.e., the representation of the documentation in the system). The analyses and metrics are defined using the elements and relationships of the meta-model, adding information to the models. Existing analyses (such as cluster analysis and clone detection) and metrics (such as size, cohesion and coupling counts) can thus be reused for different documents. Similarly, new analyses and metrics can be added when there is a need.

The metrics and analyses operate on views of the models of the common repository rather than on the actual models. As discussed in Section 2.2 the meta-model may need to be extended. The metrics and analyses required to support this would need to be extended and changed accordingly. By defining them for views that abstract the information from the model, they are reusable as long as there is a transfer function from the model to the view.

Consider the example from Section 2.1. Suppose we define a simple metric that counts the number subsections within a section. A naive implementation of such a metric might be implemented by counting the number of children that are Subsection elements of every Section element in the common repository. Now assume that we extend the meta-model so that a section contains a *Header paragraph* and a *Body element* which, in turn, contains subsections (Section elements). As a consequence, our metric would not find any subsections that are direct children of a Section element and, hence, would return zero subsections for every section. If we instead define the metric for a view of the meta-model that only contains sections and subsections (i.e., filters out the Body element), then the metric can be reused without change.

2.4 Visualizing the Results

Visualizations are used to illustrate the models, the results of the analyses, and metric values. Technically, these are nothing but additional analyses that generate special views that highlight the elements and relationships of the common repository.

Visualizations can be considered as mappings from the analysis space (the elements, relationships, values computed by metrics, etc.) to a visual space (shapes, colors, textures, position, etc.). For example, the number of subsections per Section element is a metric value that can be mapped into one bar in a bar chart. The visual objects and their attributes convey information about the documentation and its quality.

The mapping between the analysis space and the visual space can be configured. We generally map elements from the repository to visual objects of different shapes or color, and map the quality attributes to visual attributes of these objects. There are several examples of such mappings in Section 3.

A more technical presentation of adaptation and configuration opportunities associated with VizzAnalyzer is presented in Löwe and Panas [6]. A formal definition of the different models, meta-models and meta-meta-models (model to describe the meta-model, i.e., the meta-model specification language), including a discussion of meta-model modifications (which do not affect views) is presented in Strein et al. [8].

3 Quality Assessment in Practice

This section shows two examples of how VizzAnalyzer was used to assess the quality of technical information. Each example discusses the potential quality issue, how the common meta-model was designed and how the analysis was implemented. We also discuss the findings of each analysis. For more examples, see Wingkvist et al. [11].

3.1 Text clones

A text clone is a block of text that is repeated in various degrees of similarity across the documentation (i.e., redundant text). Redundant text is not necessarily an indication of poor quality, since it can make the text easier to read and understand by increasing the cohesion and reducing the number of cross-references, for example. But it can also increase the cost of maintaining and translating the documentation. In order to investigate the degree of text clones in real world technical information, we implemented a clone detection analysis and used it to assess the non-classified parts of the technical information of a warship. Figure 1 depicts the result of the clone detection.

The clone detection analysis determines how similar two documents are by first comparing the text, and then the document structure. In order to implement this analysis, the common meta-model should contain at least the text and the structure. We used a *Topic* element to represent text and a *Section* element to represent sections on different levels. The structure of the documentation is represented using a *Structural containment* relationship between the elements. The clone detection analysis compares two models of different documents using metrics that compare the text of the Topic elements and the Structural containment relationships and compute the relative size of the unique parts of two documents.

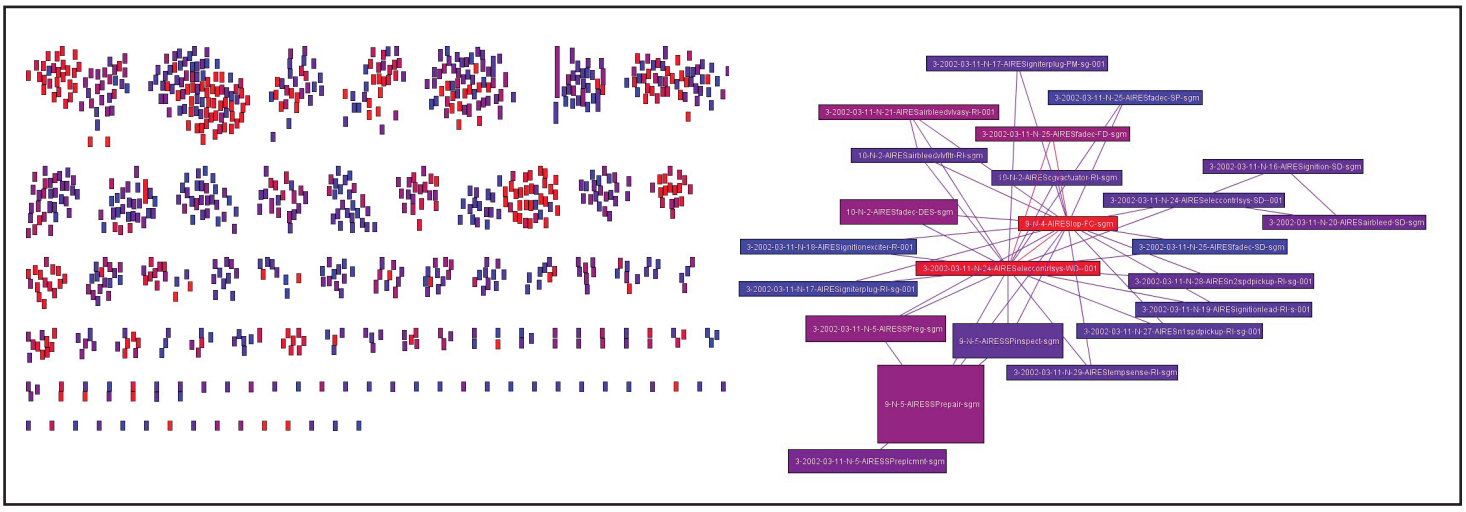


Figure 1: The result of a clone detection analysis of the technical information of a warship is visualized as clusters of similar documents (left). Each cluster can be viewed in more detail (right). The color of a box represents the degree of similarity from low (blue) to high (red).

We analyzed 913 XML documents from the warship documentation and found that only 6 of them were unique. Twenty documents were exact clones of another document, and the remaining documents were similar to some extent. On average, a document was 54% unique. The clone detection analysis is quite efficient; running it on the warship documentation was almost instantaneous.

3.2 Use of Meta-Information

Meta-information can be used to improve the usability of technical information by introducing topics or keywords. However, improper use of these can reduce the usability by, for example, attaching the topics to the wrong parts of the documentation.

In order to investigate how well meta-information is used in real world technical information, we implemented a meta-information analysis and applied it to the documentation of a mobile phone. Figure 2 depicts the result.

A section can have a number of applicability tags that describe the content. The mobile phone uses tags such as Call or SIM-card. These tags are categorized and, as an example, the tag *Call* belongs to the *Actions* category. We model the tags and the categories by introducing *Applicability* and *Category* elements and *Has a* and *Is tagged by* relationships to the common meta model. The former describes the relationship between applicabilities and categories, and the latter the relationship between sections and applicabilities. We can now define analyses (visualizations) that extract all the applicabilities for each category, and the applicabilities (and categories) for each section.

We analyzed 12,286 XML documents from the mobile phone technical information and found that some categories

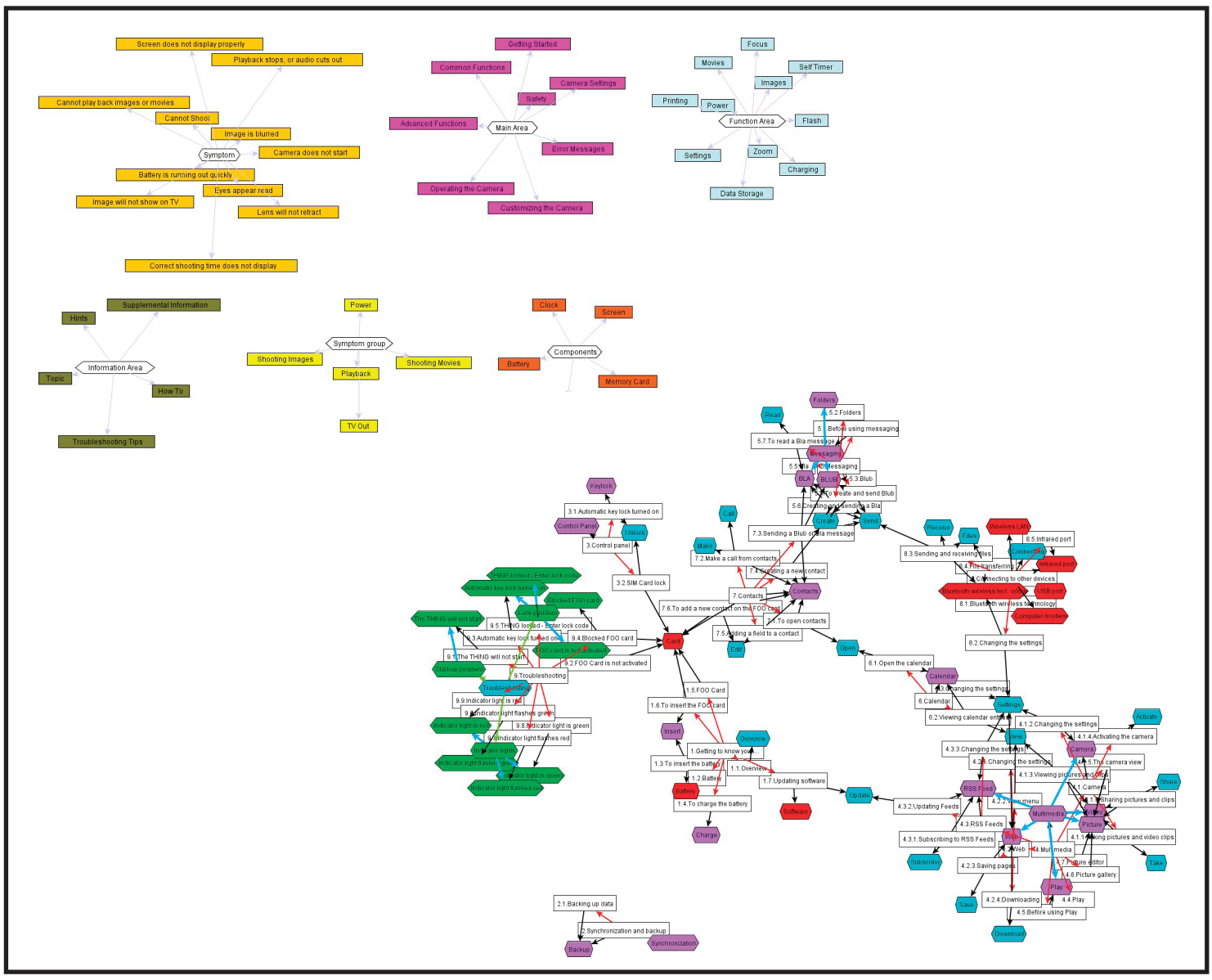


Figure 2: A meta-information analysis of technical information for a mobile phone. The visualizations show the structure of the meta-information (left) and the relationship to information (right). The color of an applicability tag identifies the category it belongs to.

concentrated on certain parts of the technical information, while other categories were spread out. For example, the category *Troubleshooting* was concentrated in the technical information related to troubleshooting, while the category *Actions* was spread throughout the documentation. These observations can indicate quality issues, but the technical writers confirmed that this was intentional and part of the design.

The common meta-model used for the clone detection and meta-information analyses can be described by the E/R diagram depicted by Figure 3. Note that certain technical details present in the E/R diagram were left out of the discussion in this section, to keep it simpler.

4 Conclusions

The assessment of technical information quality requires complete quality models, which are very difficult to define. We argue that a bottom-up approach, where we provide a human expert with:

- a repository that contains the technical information,
- a way to import existing documents from files or other data sources,

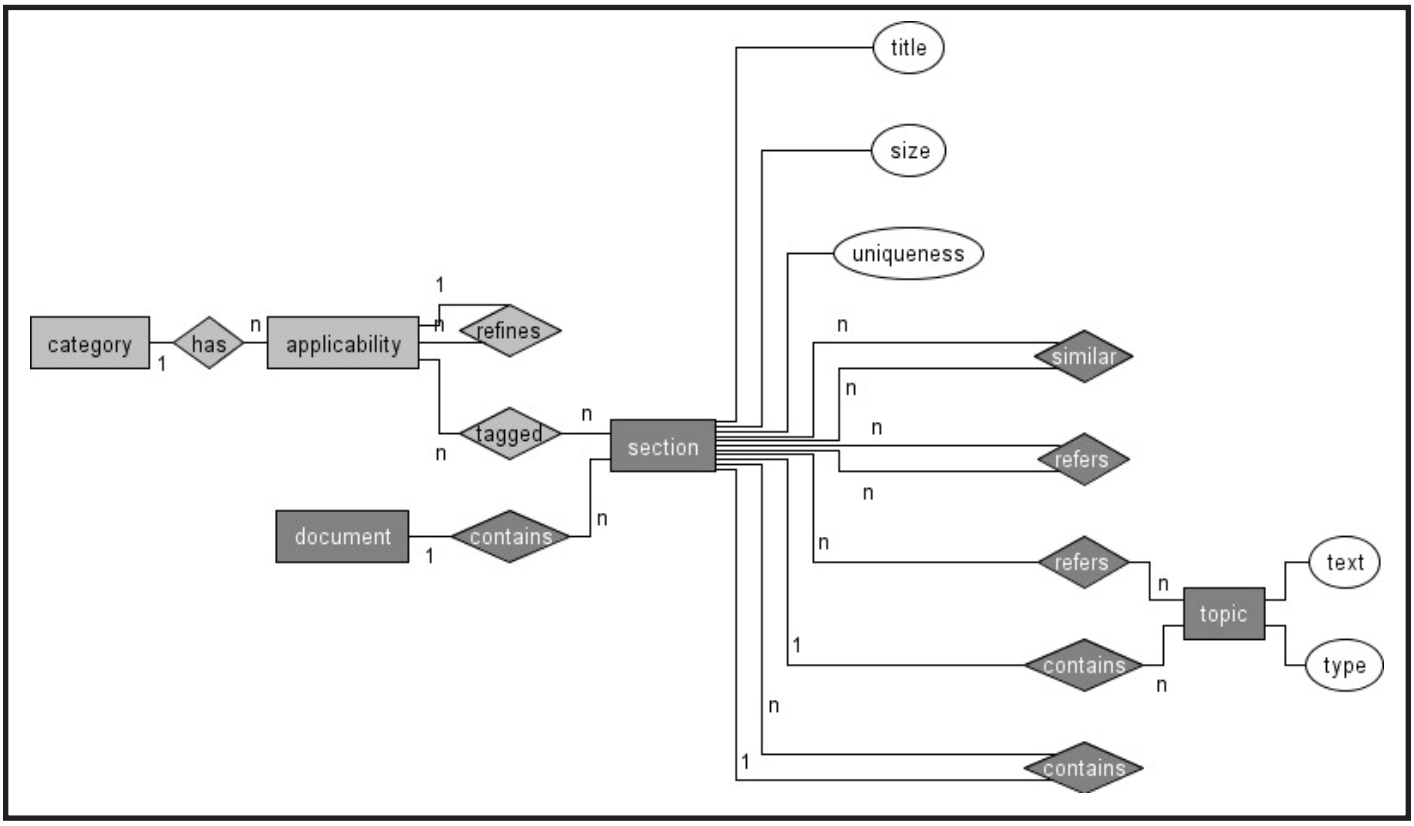


Figure 3: An E/R diagram that shows the common meta-model used for the examples. The dark grey entities and relationships are used to capture document data, while the light grey captures meta-data.

- a number of analyses and metrics that can be used to ask questions, and
- visualizations of the answers to these questions,

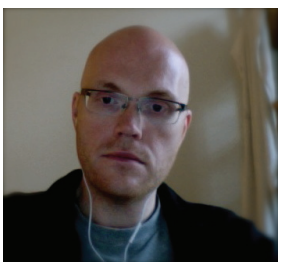
can be an effective way to approach assessment of technical information quality.

The bottom-up approach requires flexible software tools that support a wide range of document formats, analyses and metrics, and visualizations. The OSS tool, VizzAnalyzer, offers this flexibility and has proven useful in quality assessment. We have used it to analyze several instances of real world technical information from different domains and have, in collaboration with domain experts, identified possible quality issues.

In the bottom-up approach, human experts define the quality model in use through analyses and visualizations. These quality models may be specific to a project or a domain, but parts of them may be general enough to form best practices.

About the authors:

Anna Wingkvist is a postdoctoral researcher in Computer Science at Linnaeus University, Växjö, Sweden. Her academic background is in information systems development, methodological and research methods reasoning, and project management. Since earning her PhD in 2009, her scientific interest and publications are mainly in the information quality domain. In June 2011 she was awarded a prestigious, multiyear research grant from the Swedish Governmental Agency for Innovation Systems, VINNOVA. Email: Anna.wingkvist@lnu.se



Morgan Ericsson is an associate professor in Computer Science at Linnaeus University, Växjö, Sweden. After completing his PhD in 2008 he was a postdoctoral researcher at the Department of Information Technology, Uppsala

University, Sweden. His main research interest is how methods and tools from software technology and database research can be used to assess quality. He is a skilled programmer and has contributed to the development of programming models and frameworks for software and information analysis. Email: Morgan.ericsson@lnu.se

Welf Löwe is a professor in Computer Science and he has held a Chair in Software Technology at Linnaeus University since 2002. His experience with numerous projects in industry and research has made him an expert in software and information analysis as well as software and information quality management in general. Email: Welf.lowe@lnu.se

References

1. J. D. Arthur and K. T. Stevens. Document quality indicators: a framework for assessing documentation adequacy. Journal of Software Maintenance, 4: 129–142, Sep 1992.

2. K. Cook, R. Earnshaw, and J. Stasko. Guest editors' introduction: Discovering the unexpected, computer graphics and applications. 27(5):15–19, 2007.

3. G. Hargis, M. Carey, A. K. Hernandez, P. Hughes, D. Longo, and S. Rouiller. Developing quality technical information – A handbook for writers and editors (2nd ed.). Upper Saddle River, NJ: Pearson Education, 2009.

4. D. Keim, F. Mansmann, J. Schneidewind, and H. Ziegler. Guest editors' introduction: Discovering the unexpected, computer graphics and applications. Proc. of Information Visualization (IV 2006), IEEE, 27(5):9–16, 2006.

5. S. A. Knight and J. Burn. Developing a framework for assessing information quality on the World Wide Web. Informing Science, 8:159–172, 2005.

6. W. Löwe and T. Panas. Rapid Construction of Software Comprehension Tools. International Journal of Software Engineering and Knowledge Engineering, 15(6):905–1023, Dec 2005. Special Issue on Maturing the Practice of Software Artifacts Comprehension, Ed. Nenad Stankovic, World Scientific Publishing.

7. K. Smart, J. Madrigal, and K. Seawright. The effect of documentation on customer perception of product quality. IEEE Transactions on Professional Communication, 39(3):157–162, Sep 1996.

8. D. Strein, R. Lincke, J. Lundberg, and W. Löwe. An extensible meta-model for program analysis. Software Maintenance, 2006. ICSM '06. 22nd IEEE International Conference on, pages 380–390, Sep 2006.

9. B. Stvilia, M. B. Twidale, L. C. Smith, and L. Gasser. Assessing information quality of a community-based encyclopedia. In Proceedings of the International Conference on Information Quality, pages 442–454, 2005.

10. J. Thomas and K. Cook. Illuminating the Path: The Research and Development Agenda for Visual Analytics. National Visualization and Analytics, 2005.

11. A. Wingkvist, M. Ericsson, and W. Löwe. A visualization-based approach to present and assess technical documentation quality. The Electronic Journal of Information Systems Evaluation, 14(1):150–159, 2011.