An Intuitive Interface for Technical Documentation Based on Semantic Knowledge Graphs

Frieder Loch and Markus Stolze

Department of Computer Science, Ostschweizer Fachhochschule, Rapperswil, Switzerland
frieder.loch@ost.ch

Abstract. Maintaining technical documentation is a challenge. Products are becoming more complex, product lifecycles are getting shorter, and the number of product variants is increasing. Manuals that guide personnel in the use and maintenance of products are critical to their efficient and safe operation. Authoring system for technical documentation therefore increasingly apply semantic models to control the cost of maintaining technical documentation. Working with formal semantic structures is challenging for technical writers who usually work with plain, written text. This paper presents an intuitive interface for a semantic knowledge graph to facilitate the adoption and use of semantic models in technical documentation. The interface allows working with unstructured text and to postpone its semantification. Users can add semantic annotations in an iterative and incremental way. The interface was developed using a user-centered design process and subjected to an evaluation with technical writers. The results indicate that technical writers could use the prototype successfully and enjoyed the underlying concepts. Further iterations will extend the system and, for example, use artificial intelligence to suggest semantic links to improve the quality of the knowledge graph.

Keywords: Human computer interaction · Semantic knowledge graph · Technical documentation · User-centered design

1 Challenges in Technical Documentation

Every product needs comprehensive documentation to describe its operation and maintenance. Creating and maintaining technical documentation for complex technical systems, such as mining equipment or industrial machinery, is challenging because these systems are complex and come in many product variants that require specific documentation depending on their features.

In the past, technical documentation was created using text editors. This approach makes technical documentation hard to update and maintain, since the reuse of plain text documentation fragments or the updating of such information is hard. Software for technical documentation has changed to address these challenges. There has been a trend towards

- *increasing semantics* of the data models and increasing granularity of those models and
- *increasing integration* of data from diverse sources (e.g., engineering, logistics).

The German Standardization Roadmap Industry 4.0 [5] emphasizes the need for semantic unambiguity and interoperability. To meet these requirements, information about technical products must be provided in a semantic and machine-readable structure. Frequently used data structures for semantically annotated data are knowledge graphs and knowledge graph-based ontologies [12].

Over the past twenty years, STAR AG has developed a comprehensive knowledge graph-based meta model and associated tools to capture, manage and distribute semantic information, to support after-sales processes such as maintenance and troubleshooting. This model and the tools are successfully used by companies such as Daimler, Ferrari, Hilti, Liebherr, Vaillant, and Volvo Trucks.

With current tools, capturing product knowledge in knowledge graphs requires significant training. This is an obstacle to scalability to all product information and wider market adoption. The research project Smart Knowledge Capture addresses these challenges and aims to make tools for interacting with knowledge graphs more accessible by developing intuitive interfaces and AI-based assistance. Another goal is to give technical writers a shortcut to information owned by other stakeholders through an intelligent knowledge capturing tool.

This paper contributes a user interface that simplifies capturing instructions in a semantic knowledge graph. The interface allows working with unstructured text and hides the underlying semantic data model. The instructions are incrementally enriched with semantic information. In addition to providing intuitive interaction, this interface should make more transparent how the information in the knowledge graph will be displayed later, for instance in a manual. The interface was developed in an iterative, user-centered process based on contextual inquiry and evaluated with technical writers.

2 Editing of Knowledge Representations

Semantic data models are being applied in many domains. However, ontology authoring tools did not gain traction apart from experts. Ontology engineering tools typically lack a user-centered perspective [15]. These tools have improved, but still have usability issues. The literature proposes an emphasis on the needs of "normal users" instead of "power users" when designing such tools since people with no formal training in knowledge representation are increasingly working with ontologies and other knowledge representations [6, 7]. The study of Dzobor et al. [7] highlights shortcomings of current editing tools such as unclear error messages or a tendency to display too much information at once.

Successful interaction with ontology editing tools depends on knowledge of the low-level languages and frameworks [7]. The Protégé editor is a popular editor for the modelling of ontologies [10]. Semantic Wikis use semantic information to express relations between information to facilitate reuse and semantic queries [3]. Evaluations indicated that the usability of such applications for different stakeholders, especially the ones without expertise in knowledge representation, is an issue [1].

The design of editing interfaces for people without expertise in semantic data models is critical. This is especially true for user-facing applications such as in technical documentation software. However, subject-matter experts or technical writers without training in knowledge representation struggle with existing editing tools.

Several approaches introduce a user-centered approach in the design of ontology authoring tools. Usage patterns in ontology authoring tools such as Protégé are analyzed to derive different user types and, for instance, make the tool adaptive to the user type [14]. Other approaches use ontology visualization. Their aim is to visualize ontologies to facilitate their understanding and editing by users [8]. However, these approaches focus on users with knowledge in ontology engineering and not on the integration into user-facing applications.

The gap between concrete objects and the abstract knowledge representation needs to be bridged. Oberhauser et al. [11] and Stobbe et al. [13] propose an approach that creates description of interactions with software using automated video analysis and the analysis of interactions with a focus on legacy systems. Thereby, semantic information can be created without having to interact with the knowledge graph directly.

According to the previous discussions, a user-centered interface for semantic technical documentation should meet the following requirements:

- **R1: User-Centered Design.** The interface will be developed using a user-centered approach. This ensures the usability of the tool for different skill levels.
- **R2: Interface for non-experts.** The interface should be usable by technical writers without knowledge engineering expertise. Evaluations will be conducted with such users to validate this requirement.
- **R3: Simple and Application-Oriented Interface.** The primary means of interaction is unstructured text. This should make working with the software less abstract.

This paper contributes an interface that facilitates the creation of a semantic knowledge graph for technical documentation. The interface is developed in a user-centered process and evaluated with technical editors.

3 Challenges with Existing Tools

As discussed in the previous section, interacting with a semantic knowledge representation is a challenge. Technical writers must follow the constraints of the data model and decide which components and tools they need before starting to create documentation. For example, to create the instruction "Loosen the battery using a screwdriver" the objects *battery* and *screwdriver* need to be created in the knowledge graph. These parts are transformed to text when they are being published for different media. This forces writers to follow the structure of the semantic data model and requires a steep learning curve. Furthermore, the effects of the edits on the data model are not visible immediately. Our interface reverses this approach. The instruction can be entered first as unstructured text and the tools can be annotated later.

Reusing information through semantic linking is critical to realizing the benefits of the semantic knowledge graph. For example, tasks may require the preparatory action of removing a cover before accessing an engine. If these tasks reference this preparatory action, they can be easily adapted. A new model may have a modified cover that is removed differently. If these tasks replicate the information, the technical writer must adapt the information multiple times. This approach (*Copy, paste, and modify*) can introduce errors. Therefore, it is critical for an interface for technical documentation to facilitate annotating and reusing information.

4 Contextual Inquiry and Interface Design

The user-centered design process started with a contextual inquiry study. The goal of this study was to understand the tasks of technical writers and their characteristics and motivations. From this study, personas were synthesized to guide the development of the prototypes. The prototypes were evaluated in usability walkthroughs and refined accordingly. The final interface was implemented, backed by the semantic knowledge graph, and evaluated with technical writers.

4.1 Contextual Inquiry Study

Contextual design considers data from prospective users as the main criterion for deciding on the structure of the system [9]. The goal is to make sense of real-life working situations to understand the characteristics and motivations of potential users to better empathize with them. This includes the tasks that they perform and the tools they currently use. Various research methods are used, such as interviews, low-fidelity prototypes, and scenarios [2]. We also wanted to collect problems with existing tools. Considering both, their characteristics and problems with existing tools should inform a user-centered design process.
These methods aim to understand the context-of-use. This comprises the human, physical, organizational, historical and social environment in which a technology is used to tailor the application to this context [2].

Contextual inquiry usually depends on being present where the work is done. However, the studies were conducted during the measures to contain the COVID-19 pandemic. Therefore, the contextual inquiry had to be conducted remotely and some methods were not applicable. Instead of a participating observation, we asked participants to provide us with video recordings of common tasks. Based on the results of the video analysis, we designed semi-structured interviews with the same persons to inquire on the tasks in the videos.

Video Analysis. We asked the participants to record videos of regular work tasks. The goal of the video analysis was to understand the tasks of technical writers. We asked the participants to provide additional explanations of the tasks. If possible, the participants provided thinking-aloud information on any issues that they were facing. This approach combined the contextual interview with elements of a diary study [2].

Personal interviews were conducted based on the results of the video analysis. This was done to establish a successful partnership with the people who actually do the work, in order to understand their tasks properly [9].

Four technical writers with various levels of experience and educational backgrounds (e.g., academic training in technical writing or career changers) participated in the study. The participants maintain technical documentation in various domains, such as software or industrial machinery. Each participant recorded about one hour of videos. The results of the video analysis were synthesized as an affinity diagram that clusters recurrent issues under emerging themes.

Results of the Video Analysis. A frequent task in technical writing at our project partner is the migration of technical documentation into the semantic knowledge graph. This is often done when new customers are onboarded. The main challenge is to map the existing documentation to the objects in the semantic knowledge graph. This is challenging due to the variety of objects in the knowledge graph that, for instance, can be used to model different tasks.

We observed that technical writers often relied on the layout preview to verify that their changes to the knowledge graph had the intended effect. Even experienced technical writers were unsure whether they chose the correct semantic structures. Step types tended to be chosen based on the desired appearance in the final manual and not whether they are semantically correct. Later interviews confirmed this observation and suggested that working directly on the abstract structures of the semantic knowledge graph is not intuitive. This observation motivated the idea to allow editing the knowledge graph in a way that resembles the visual layout of the resulting documentation, thereby removing the abstraction of working on the knowledge graph.

Individual Interviews. The video analysis revealed frequent tasks and issues of technical writers. Personal interviews were carried out to ensure the correct interpretation of the incidents in the videos and to clarify possible misconceptions. The interview addressed recurrent themes that were identified in the video analysis. The interview consisted of three parts.

1. **Motivation and Background.** The first part addressed the motivations of the participants with regards to their work as technical writers. It was, for instance, discussed why they decided to work in this domain, what they find satisfying about their job, and what makes good technical documentation.
2. **Processes in Technical Writing.** Participants explained how they organize their work and how they collaborate with colleagues. This should help understanding common work processes.
3. **Working with the Existing Software.** The aim of these questions was to understand how the existing software is used and how it could be improved. This should lead to concrete ideas for features of the interface. Incidents that were observed in the video analysis were discussed as well.

Creation of Personas. Personas represent individual users with distinct motivations and characteristics and should be created based on user research [4]. The process for deriving personas based on research data was based on Cooper et al. [4]. The main part is to create dimensions that describe the characteristics of users. These dimensions were created based on the insights gathered by user research. This process was done in a more qualitative manner than the quantitative approach suggested by Cooper. Below are three example dimensions that characterize our personas.

- One dimension was whether the participants take pride in well-written and well-set text. This aspect varied between technical writers with a background in a technical profession and technical writers with specific academic training in technical writing.
- Another dimension was the extent to which the technical writers were interested in the semantic information model and its quality. This aspect was critical since a high quality of the information model allows, for instance, to reuse information and work more efficiently.
- Another dimension concerned whether the participants enjoy working with innovative technology. This is independent of the concrete technology, but concerns liking a technology for the sake of being a new technology.

The characteristics of the interview participants were located on these dimensions. Clusters of characteristics were identified to create three personas. We developed the following personas.

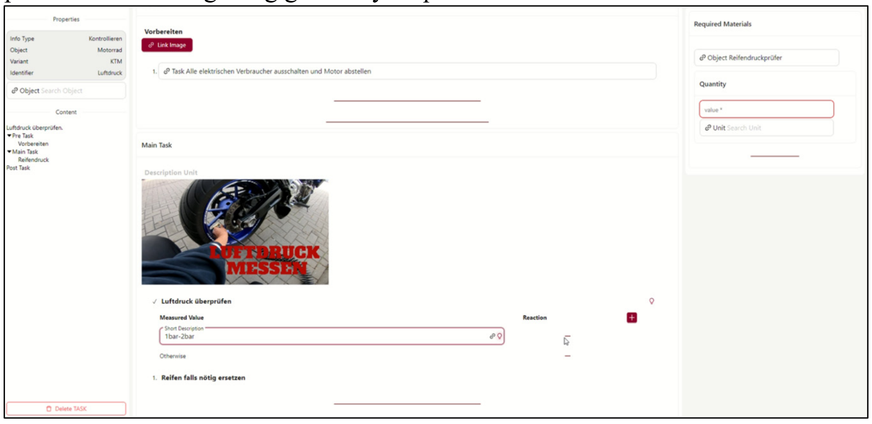
- **Pragmatic.** Someone who is interested in getting the job done and has no strong preferences with regards to the applied tools, if good text can be produced.

- **Semantic Expert.** Someone who is interested in working out the semantic structures and filling them in perfectly. Information must be structured properly while avoiding redundancies as much as possible.
- **Career Changer.** Someone who started working in a technical profession and switched to technical writing as a second career. This person is not too interested in semantic models and has significant technical knowledge about the tasks.

4.2 Design of the Interface

The interface was created based on the results of the contextual inquiry studies that were described in the previous section. The design process was iterative and started with low-fidelity prototypes that were refined later.

The main design idea was to allow for incremental semantification. The user should be able to work with unstructured text and delay the semantification. This should allow use cases such as entering unstructured notes that an editor receives from technicians without having to store them in a note-taking application first. This should hide the complexity of the underlying semantic model until the user decides to use it, and help inexperienced users in getting gradually acquainted with the semantic model.



Info Type	Kontrollieren
Object	Motorrad
Variant	KTM
Identifier	Luftdruck

Object Search Object

Luftdruck überprüfen.
- Pre Task
  - Vorbereiten
- Main Task
  - Reifendruck
- Post Task

Vorbereiten

1. Task Alle elektrischen Verbraucher ausschalten und Motor abstellen

✓ Luftdruck überprüfen

Measured Value — Reaction

Short Description: 1bar-2bar

Otherwise

1. Reifen falls nötig ersetzen

Object Reifendruckprüfer

Unit Search Unit

Fig. 1. Interface of the text-based interface for technical documentation.

Description of the Interface. The interface (see Figure 1) consists of three panels. The left column locates the task in the information model (e.g., by identifying the product, the product variant, and the task) and gives an overview of the steps of the task. The middle column allows the creation and maintenance of the steps of the task. The right column allows editing details for the step that the user has selected in the middle column, such as linking materials that are needed in the selected step. This is a common structure of application interfaces (overview & detail).

The middle column provides functions to create and edit steps. Most steps instruct the user to perform an action, often using a tool. Other steps require checking a value and provide information on how to react to different results. Each task consists of a pre-task a main task and a post-task. The pre-task describes preparatory work before starting the main task. In our case, it is necessary to turn off the engine before checking the air pressure. The post-task concerns work after the main task, such as cleaning up. Different main tasks often share pre- and post-tasks. The information model includes more complex types that were omitted in the prototype.

Horizontal lines indicate where the user can add steps. Hovering over the lines displays a plus-sign that allows adding distinct types of instructions. Instructions can be simple instructions, tasks that structure related instructions, or tasks that describe a check step and what to do in case of different results. A new instruction can also be added by pressing Enter in the previous step. The user can delete, modify, and reorganize the steps using drag and drop.

An important task is linking instructions with materials, such as required tools. This is handled in the right column. Linking materials is desirable since this allows reuse and increases the quality of the knowledge graph. The interface aids the user when creating links. It suggests objects that may be linked to the task. A flashing bulb next to the label of the step indicates that a suggestion for a link is available. The system provides the suggestions using a full text search through the existing data model.

Technical Implementation. The system consists of a frontend and a backend. Both components communicate using REST-interfaces and web sockets. The frontend is implemented using state-of-the-art web technologies. React serves as the frontend framework and MobX for state management. For the design of interface, Ant Design and Tailwind CSS were used. The backend communicates with the semantic data storage using a REST-interface.

5 Evaluation

A qualitative evaluation of the final interface addressed whether technical writers would accept such an interface and whether the interface was suitable for interacting with a semantic knowledge graph. Therefore, we were interested in suggestions for improving the prototype, as well as a comparison of the prototype with the existing tool for technical documentation. A quantitative comparison of the prototype with the existing tool was not performed due to the significant difference in the supported functionality and the complexity of the existing tool.

5.1 Evaluation Procedure

Participants in the evaluation performed a realistic documentation task. They were given an informal, written description of a maintenance task and instructed to formalize this information using our interface. This process is often required when a new product variant is introduced or when a manual is reviewed. Such reviews are often conducted in the field or in the workshop with technicians. The results of these reviews are usually noted on physical printouts of the manuals and later validated and entered into the system by dedicated editors.

Setup and Metrics. The participants received a video introducing the interface prior to the evaluation. This was done to familiarize the participants with the prototype and to reduce the effects of novelty. The evaluation began with a sample task that the facilitator performed to provide an overview of the prototype and introduce its functions. After this introduction, the participants performed the task. Participants completed the evaluation tasks independently. However, the facilitator was available to assist and asked clarifying questions when encountering potential issues.

The facilitator asked the participants to think-aloud during the evaluation. In addition, we used the System Usability Scale (SUS) to evaluate the usability of the interface and compare it to the usability of the tool that they use regularly. The participants completed the SUS after the evaluation.

The evaluations were conducted remotely via a web conference. Since a VPN connection was required to use the prototype, which was not available to the participants, they controlled the prototype using screen sharing software (MS Teams/TeamViewer).

Participants. Nine people participated in the study. All participants worked as technical editors and used the same tool. They worked in different domains, mostly focusing on the documentation of technical equipment.

The evaluation sessions were intended to be individual sessions. Two sessions were conducted in groups due to time constraints. In this case, one participant controlled the system while the other participants observed the interactions and provided comments. Five sessions were conducted in total.

5.2 Results and Discussion

Participants were able to successfully interact with the prototype in the evaluation after only a short training period. All participants could complete the tasks and enjoyed the simple interface and the use of modern interaction techniques (e.g. drag and drop).

Participants mentioned that the interface would be particularly useful for simple tasks and tasks performed by infrequent users, such as subject matter experts. Such an interface could be provided on a mobile device to support the use of the application in the field, for example, when reviewing or updating documentation. In such cases, using the desktop software may be too complex and result in users creating paper notes that must be manually transferred.

Results of the System Usability Scale. The results of the SUS questionnaire supported the positive results of the evaluation. The participants found the application to be not unnecessarily complex (med=3)[1] and to be easy to use (med=3.5). They agreed that the operation of the application can be learned easily (med=4). Most participants would like to use the application on a regular basis (med=3). The resulting SUS score of 65 indicates the good usability of the interface.

The scores for the prototype were superior to those for the current software. The most visible differences concerned that participants perceived the existing software to be harder to learn (med=1 vs. med=4) and that much expertise is required to being able to use the software (med=4 vs. med=1.5). However, the comparison of a full-fledged software to a prototype should be interpreted with caution. Nevertheless, these results indicate that the design ideas were well received.

Suggested Improvements. The evaluations provided ideas for improving the prototype. An important feature is keyboard navigation. It should be possible to create new steps and navigate between steps using keyboard shortcuts. Keyboard navigation not only improves accessibility, but also helps experienced users working more quickly.

[1] Scores range from 0 (fully disagree) to 4 (fully agree). We report median values due to the low number of participants.

Keyboard navigation should use conventions and shortcuts that other applications use. Another consideration was to allow users to store tools and tasks that they use frequently. This should allow to create meaningful shortcuts and customize the application for different tasks.

Further suggestions concerned the visual presentation. For instance, splitting the interface into three panels (see Figure 1) was difficult in some cases. Actions in the right panel that are triggered in the middle panel tend to be overlooked by the participants.

Remote Testing. The remote setup for the usability tests worked well. The use of web conferencing facilitated the recording of the sessions for later analysis and increased the flexibility in scheduling. Participants used the prototype via screen sharing without significant problems. Only the facilitator's ability to interact with the prototype to assist the participants was limited. A thorough briefing of the participants and an opportunity to practice at the beginning is crucial to explain the limitations of the remote testing setup to the participants, such as the latency when using the mouse in a screen sharing session or possible issues with different keyboard layouts.

6 Conclusion and Outlook

This paper described the development of an intuitive interface for the creation and maintenance of technical documentation using an iterative, user-centered process. The interface was intended to replace or complement directly working with the knowledge graph. The interface should facilitate working with semantic structures for users with little or no experience. An evaluation validated the design ideas. The evaluation indicate that the interface can facilitate editing tasks while maintaining the benefits of the underlying semantic model.

The evaluation had a qualitative focus. Future evaluations should include a quantitative comparison with existing applications and be conducted over a longer period. Performance on longer editing tasks should be evaluated as well. This will provide an estimate of the return on investment (ROI) of implementing the interface and introducing it to the existing software.

Work is ongoing to extend the existing interface to other technical editing tasks, such as the creation of new product variants, for example when introducing a new generation of a product. This is a complex operation. It is necessary to decide which information can be reused from the existing variant and which information needs to be adapted to match the characteristics of the new variant. A wizard that guides technical writers can speed up this process and eliminate potential errors. It should also be evaluated whether the concept of the interface can be transferred to other editing tasks (e.g., the creation of product descriptions) without losing its benefits.

Another line of work concerns the introduction of AI-based assistance in the authoring process. This assistance can suggest tools or tasks that can be linked in a particular context of an instruction. The goal is to ease the process of finding and linking resources. By increasing the amount of linked information, the quality of the knowledge graph and the documentation improves. The integration of this assistance into the prototype and the evaluation of the suggestions of the assistance is ongoing. This includes how the user interface should present the suggestions to the user (i.e., confidence score of the suggestion, display of alternatives or explanation of the applied model) and whether the suggestions are useful.

References

1. Barok D, Boschat Thorez J, Dekker A et al. (2019) Archiving complex digital artworks. Journal of the Institute of Conservation 42:94–113. doi: 10.1080/19455224.2019.1604398
2. Benyon D (2013) Designing User Experience: A Guide to HCI, UX and Interaction Design. Pearson
3. Boulos MNK (2009) Semantic Wikis: A Comprehensible Introduction with Examples from the Health Sciences. JETWI 1. doi: 10.4304/jetwi.1.1.94-96
4. Cooper A, Reimann R, Cronin D (2007) About face. The essentials of user interface design. Wiley
5. DIN/DKE (2020) German Standardization Roadmap on Industry 4.0. https://www.din.de/en/innovation-and-research/industry-4-0/german-standardization-roadmap-on-industry-4-0-77392. Accessed 02 Feb 2023
6. Dzbor M, Motta E (2008) Engineering and Customizing Ontologies. In: Hepp M, Leenheer P, Moor A et al. (eds) Ontology Management, vol 7. Springer US, Boston, MA, pp 25–57
7. Dzobor M, Motta E, Aranda C et al. (2006) Developing ontologies in OWL: An observational study. In: OWL: Experiences and Directions Workshop
8. Fu B, Steichen B (2022) Supporting user-centred ontology visualisation: predictive analytics using eye gaze to enhance human-ontology interaction. IJIIDS 15:28. doi: 10.1504/IJIIDS.2022.120143
9. Holzblatt K, Beyer H (2016) Contextual Design. Design for Life. Morgan Kaufmann
10. Horridge M, Gonçalves RS, Nyulas CI et al. (2019) WebProtégé: A Cloud-Based Ontology Editor. In: Liu L, White R (eds) Companion Proceedings of The 2019 World Wide Web Conference. ACM, New York, NY, USA, pp 686–689
11. Oberhauser J, Gieschke R, Rechert K (2022) Automation is Documentation: Functional Documentation of Human-Machine Interaction for Future Software Reuse. IJDC 17:11. doi: 10.2218/ijdc.v17i1.836
12. Rosen R, Fischer J, Boschert S (2019) Next Generation Digital Twin: an Ecosystem for Mechatronic Systems? IFAC-PapersOnLine 52:265–270. doi: 10.1016/j.ifacol.2019.11.685
13. Stobbe O, Rechert K, von Suchodoletz D (2014) Demonstration of an Integrated System for Platform-independent Description of Human-Machine Interactions. In: Coates S, Pearson D, O'Meara E et al. (eds) Proceedings of the 11th International Conference on Digital Preservation, pp 377–378
14. Vigo M, Jay C, Stevens R (2014) Protégé4US: Harvesting Ontology Authoring Data with Protégé. In: Presutti V, Blomqvist E, Troncy R et al. (eds) The Semantic Web: ESWC 2014 Satellite Events, vol 8798. Springer International Publishing, Cham, pp 86–99
15. Vigo M, Matentzoglu N, Jay C et al. (2019) Comparing ontology authoring workflows with Protégé: In the laboratory, in the tutorial and in the ‘wild’. Journal of Web Semantics 57:100473. doi: 10.1016/j.websem.2018.09.004