Electronic Communications of the EASST
Volume 85 Year 2025

deRSE25 - Selected Contributions of the 5th Conference for Research Software Engineering in Germany

Edited by: René Caspart, Florian Goth, Oliver Karras, Jan Linxweiler, Florian Thiery, Joachim Wuttke

SMECS: A Software Metadata Extraction and Curation Software

Stephan Ferenz, Aida Jafarbigloo, Oliver Werth, Astrid Nieße

DOI: 10.14279/eceasst.v85.2708

SMECS: A Software Metadata Extraction and Curation Software

Stephan Ferenz[1 2], Aida Jafarbigloo[2], Oliver Werth[2], and Astrid Nieße[1 2]

[1] stephan.ferenz@uol.de, astrid.niesse@uol.de
Department for Computer Science
Carl von Ossietzky Universität Oldenburg, Germany

[2] aida.jafarbigloo@offis.de, oliver.werth@offis.de
Energy Division
OFFIS, Germany

Abstract: Providing metadata for research software plays a crucial role in adopting the FAIR (Findable, Accessible, Interoperable, and Reusable) principles for research software. Metadata enable findability and reusability. However, creating high-quality metadata can be resource-intensive for everyone who develops research software like researchers and research software engineers. To address this challenge, the creation of metadata can be partly automated by gathering distributed existing metadata. Therefore, we developed the Software Metadata Extraction and Curation Software (SMECS) which integrates the extraction of metadata from existing sources together with a user-friendly interface for metadata curation. SMECS extracts metadata from online repositories such as GitHub and GitLab. It presents the extracted metadata to its users through an interactive interface for further curation and allows to export the metadata as a *CodeMeta* file. The usability of SMECS was evaluated through usability experiments which confirmed that SMECS provides a satisfactory user experience. Through simplifying the creation of metadata for research software, SMECS enables the adoption of the FAIR principles for research software.

Keywords: Software Metadata, Metadata Extraction, Metadata Curation, FAIR4RS, CodeMeta, Usability Evaluation, Research Software

1 Introduction

Research software plays a vital role in science [GKL+21] by enabling researchers to simulate real-world systems, control and monitor experiments, prove concepts, and collect, analyze, and interpret complex data [HDB+25]. Gruenpeter et al. defined research software as software that is "created during the research process or for a research purpose" [GKL+21].

Considerable challenges persist when dealing with research software [CKB+22]. For example, missing descriptions of research software often hinders the reproducibility of research which is one of the key elements of science [HCH+20]. Also, it is often not possible to build on existing research software which slows down further research [HCH+20]. From an organizational point of view, it is highly challenging to keep an overview on the research software developments within an institution.

The FAIR (findability, accessibility, interoperability, and reusability) principles for research software [CKB+22] provide important guidance on how to improve the management of research software and make it FAIR. An important aspect to achieve FAIR research software is good metadata describing the research software. This metadata should provide descriptive domain-specific information, e.g., on the purpose, functionality, and limitations of the software. This allows to find software and assess whether it is suited for reuse. Also, the metadata should include administrative information, e.g., on the license, to assess reusability from a legal perspective. Additionally, the metadata should include provenance information, a persistent identifier (PID), and a version number [CKB+22].

Yet, creating meaningful metadata for software is a labor-intensive task, as it involves collecting and mostly manually re-entering multiple pieces of information. Meanwhile, a lot of information is already available in multiple sources, such as GitHub metadata or README files [MGF19]. However, this information is distributed across these diverse sources and is often incomplete. Especially when a research software has existed for a long time, the information is often not fully known by a single person. To address these challenges, there is a need for tools that can assist individuals who develop research software, e.g., researchers and research software engineers (together referred to as users in the following), in creating metadata [FN23]. For example, such tools should be able to make use of available information from different sources.

Several tools already address different aspects of research software metadata, such as creation (CodeMeta Generator [Cod25]), extraction (SOMEF [KG21]), and publication (HERMES [KMD+25]). However, these existing tools focus on specific aspects and do not provide a comprehensive solution that connects all aspects.

To fill this gap, we present SMECS (Software Metadata Extraction and Curation Software), a novel tool that addresses the challenges of metadata creation for research software. SMECS combines extraction capabilities with a curation interface, aiming for a simple metadata creation process. Curation includes reviewing and improving the extracted metadata as well as adding additional information in accordance with the defined schema. We evaluated SMECS through a comprehensive user study based on the System Usability Scale (SUS) by Brooke [Bro95] and semi-structured interviews. The evaluation showed that users already perceived SMECS as a usable software while they wished for more automation, like automatic generation of descriptions, and reuse of more existing metadata.

With this paper, we contribute as follows:

- We provide a comprehensive overview of existing approaches for the creation, extraction, and curation of research software metadata in Section 2.
- We introduce our solution SMECS and its architecture in Section 3.
- Based on the evaluation method, described in Section 4, we present a qualitative evaluation of SMECS in Section 5.
- Based on the discussion of our results, we provide general implications for further research on tooling for research software metadata and an outlook for further developments of SMECS in Section 6.

2 Related Work

This section provides an overview of existing schemas and tools that support the work with research software metadata. We begin by introducing relevant metadata schemas for research software in Section 2.1, followed by a review of the existing tools in Section 2.2.

2.1 Metadata Schemas for Research Software

There exist multiple metadata schemas for research software. Some of them are domain-specific (e.g., *biotoolsXSD* [IIR+21], *Software Ontology* [MBL+14], *ontosoft* [GGMR16]) while other are non-domain specific (e.g., *CodeMeta* [JBM+23], *CFF* [DSC+21], *Software Description Ontology* [GOK+19], *DataDesc* [KGK+24]). Ferenz et al. provide a general overview of many of them in [FWN25]. In the following, we will only highlight the most relevant.

The citation file format (*CFF*) [DSC+21] provides the most important information for properly citing research software. In this way, it covers some relevant metadata information on research software without defining a proper metadata schema.

CodeMeta [JBM+23] is the de-facto standard for research software metadata covering multiple aspects of research software. It reuses many elements of *schema.org*, which is an ontology describing general concepts and is mainly used to semantically describe things (like products) on websites.

The *Software Description Ontology* [GOK+19] is an ontology to describe research software, which is based on the geoscience-specific ontology *OntoSoft* [GGMR16] and *CodeMeta*. Thereby, it provides more metadata elements than *CodeMeta* and better semantic web functionalities.

2.2 Tools for Curation and Extraction of Research Software Metadata

For the existing tools, we mainly focus on the abilities to extract metadata, allow curation, and upload the metadata to relevant databases. With respect to the extraction, we differentiate between the extraction of structured and unstructured metadata. Structured metadata are already present in a machine-readable format, e.g., as an API response or a JSON file following a specific schema. In contrast, unstructured metadata appear as free text, e.g., in a README file. Regarding curation, we additionally characterize whether the tool uses controlled vocabularies for certain metadata elements, e.g., a predefined list for licenses. Tools that exclusively serve as metadata repositories for research software, without offering extraction or curation functionalities, are not within the scope of our analysis.

The CodeMeta Generator [Cod25], developed by the Software Heritage initiative[1], is a specialized web tool to simplify the creation of *CodeMeta* metadata, which is available online.[2] The user-friendly form covers most elements of *CodeMeta* and includes drop-down menus for license and development status. The tool also allows users to export a *CodeMeta* file, as well as import and validate existing *CodeMeta* files. The tool does not extract any metadata from any source.

Betty's research engine [SRW24b, SRW24a] is a specialized search engine designed to facilitate the discovery of research software. It aggregates structured metadata from multiple sources,

[1] https://www.softwareheritage.org/, last access 2025-04-07
[2] https://codemeta.github.io/codemeta-generator/, lass access 2025-04-16

Table 1: Overview of existing approaches to create research software metadata based on the following characterization: Extracts structured metadata refers to the extraction of machine-readable metadata accessible from APIs or by reading structured files (e.g., JSON); Extracts unstructured metadata refers to the extraction of unstructured information included in text files (e.g., README); Curation refers to the ability to allow the user to edit and add metadata information; Supports controlled vocabularies refers to the ability to choose metadata from predefined lists (e.g., for licenses); Upload metadata refers to the ability to directly upload the created metadata to a certain database (e.g., software repository or registry)

✓: fulfilled (✓): partly fulfilled ✗: not fulfilled

Tool	Scope	Extracts structured metadata	Extracts unstructured metadata	Curation	Supports controlled vocabularies	Upload of metadata
CodeMeta Generator [Cod25]	Metadata creation	✗	✗	✓	(✓)	✗
Betty's research engine [SRW24b, SRW24a]	Metadata extraction for search	✓	✗	(✓)	✗	✓
HERMES [DBJ+22, KMD+25, MDB+25]	Metadata publication	✓	✗	(✓)	✗	✓
SOMEF [KG21, MGF19]	Metadata extraction	✓	✓	✗	✗	✗

including GitHub, GitLab, Zenodo, Open Alex, DataCite, and Open Citations. The engine uses a cascading search approach to iteratively improve the search results. Although, it does not systematically incorporate metadata standards such as *CodeMeta* or *CFF*, it provides a unique platform for searching for research software. Betty's research engine allows users to export search results to the Open Research Knowledge Graph (ORKG) [ADF+25] with the option to edit the JSON file online before submission [SRW24b]. A demonstrator is available online.[3]

HERMES [DBJ+22, KMD+25, MDB+25] is a tool for automating the publication of metadata using Continuous Integration/Continuous Deployment (CI/CD) pipelines. Its workflow consists of five key steps: harvest, process, curate, deposit, and post-process. HERMES uses *CodeMeta* as a data model to store metadata between the steps. In the harvest step, it extracts structured metadata from *CFF* files, *CodeMeta* files, local git repositories, and pyproject.toml files. The curation step is facilitated through Pull/Merge Requests on GitHub or GitLab, without providing a dedicated curation interface. HERMES allows metadata to be uploaded to InvenioRDM[4] based platforms such as Zenodo [EO13]. Overall, HERMES is built in a modular way by allowing plugins to be added for all steps of the workflow. This enables easy extendability [KMD+25] and

[3] https://nfdi4ing.rz-housing.tu-clausthal.de/, last access 2025-04-16
[4] https://inveniosoftware.org/products/rdm/, last access 2025-04-16

longer interoperability with these extensions.

SOMEF [MGF19, KG21, GMD^{+}25] is a tool designed to capture metadata from software projects and export it in a structured manner. Its primary goal is to provide broad extraction mechanisms that extract both structured and unstructured metadata. The sources for the structured metadata include the GitHub API and various package management files such as setup.py, pyproject.toml, and package.json. Unstructured metadata are leveraged from README files, licenses, and Docker files. SOMEF does not provide any curation capabilities. It can export metadata as a *CodeMeta* file (JSON-LD) and as an RDF file using the *Software Description Ontology*. The tool is designed to achieve high semantic-web compatibility, enabling the creation of knowledge graphs, as demonstrated in [KG21]. There is also a web demonstrator for SOMEF called SOMEF-Vider [FG21] which is available online[5] [KG21].

In this section, we provided an overview of existing tools supporting the extraction and/or curation of research software metadata as summarized in Table 1. Betty's research engine highly differs from the other approaches but shows interesting extraction sources which are not yet covered by the other tools. The CodeMeta Generator and SOMEF focus on specific aspects of the process, namely metadata creation and extraction. Meanwhile, HERMES covers the entire workflow, including extraction, curation, and publication. However, HERMES lacks a user-friendly curation interface and has limited extraction capabilities compared to SOMEF. Our goal is to bridge this gap by developing a comprehensive tool that spans the entire process, features an intuitive interface, and reuses the existing tools as far as possible. Thereby, we address the current limitations and provide a more seamless experience for users.

3 Description of SMECS

We planned SMECS to combine the extraction and curation of research software metadata. Therefore, the workflow of SMECS consists of four sequential phases: **start**, **extraction**, **curation**, and **export**, as illustrated in Figure 1. Generally, SMECS addresses the individuals developing research software as main user group. These individuals may be researchers and/or research software engineers.

In the **start phase**, users provide two initial inputs. The associated initial interface is shown in Figure 2. The first input prompts users to enter the repository link from either GitHub or GitLab, from which metadata extraction is desired. The second input requests the user's personal access token from the corresponding platform. GitHub and GitLab support personal access tokens as authentication method for their APIs and allow users to create those tokens in their accounts. SMECS can also operate without user-provided tokens for certain repositories, using internal default tokens when needed. A user-provided token is always required for other GitLab instances. Additionally, a user-provided token may enable to extract more metadata for certain repositories.

Based on the entered input, metadata from the repositories are extracted in the **extraction phase**, which is further outlined in Section 3.1. Afterwards, the extracted metadata are shown to the user in the **curation phase**. In this phase, the user can curate and refine the extracted metadata and add additional metadata. We provide more details on this phase and the user interface in Section 3.2.

[5] https://somef.linkeddata.es/, last access 2025-04-07

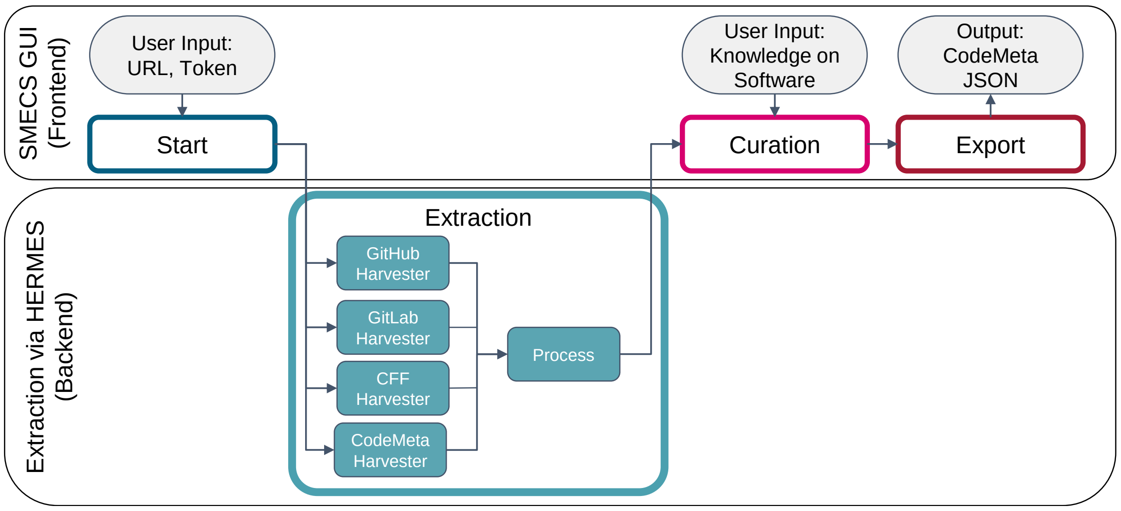


Figure 1: Process diagram of SMECS, which shows the four phases of SMECS: **Start**, **Extraction**, **Curation**, and **Export**

Once completed, the curated metadata can be downloaded as a *CodeMeta*-based JSON file (following *CodeMeta* 3.0) in the **export phase**. The exported metadata can be included by the user in their repository to make their research software more FAIR or be used for other purposes, e.g., for uploading their metadata to a software registry.

SMECS is available on GitHub [FJ25a] under an AGPL license. The GitHub repository also includes the setup instructions. Additionally, the latest version is available on Zenodo [FJ25b].

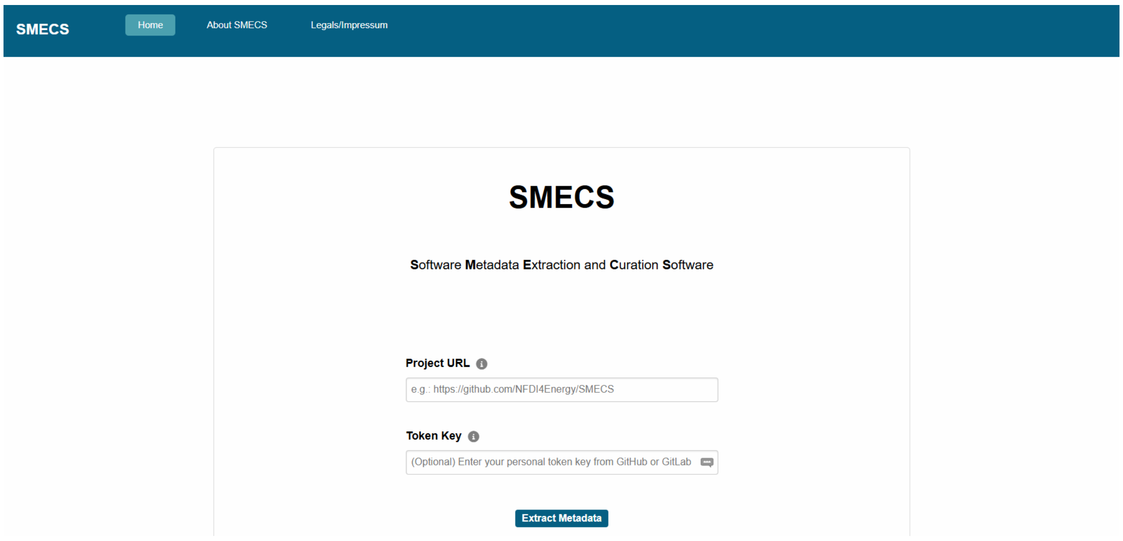

SMECS
Home
About SMECS
Legals/Impressum
SMECS
Software Metadata Extraction and Curation Software
Project URL
e.g.: https://github.com/NFDI4Energy/SMECS
Token Key
(Optional) Enter your personal token key from GitHub or GitLab
Extract Metadata

Figure 2: SMECS landing page for the **start phase**, which allows the user to provide the URL of a software repository (e.g., on GitHub or GitLab) and optionally to enter a personal access token

Table 2: Overview of the supported extraction sources via HERMES and the reference for their mapping to *CodeMeta*

Source	Description	Source for mapping
GitHub	Metadata are extracted from GitHub API	Official *CodeMeta* crosswalk[6]
GitLab	Metadata are extracted from GitLab API	Self-created Crosswalk
CFF	*CFF* files in a repository	Crosswalk by HERMES
CodeMeta	*CodeMeta* files in a repository	No Crosswalk needed

SMECS is based on the Meta Tool by Reiner Lemoine Institut [HHHM22], which is a web application enabling the creation and validation of metadata for research data based on the Open Energy Metadata standard. Building on this foundation, we developed SMECS as a Python-based tool using the Django framework [Dja23] (Version 4.2.22). The web frontend uses JavaScript, HTML, and CSS.

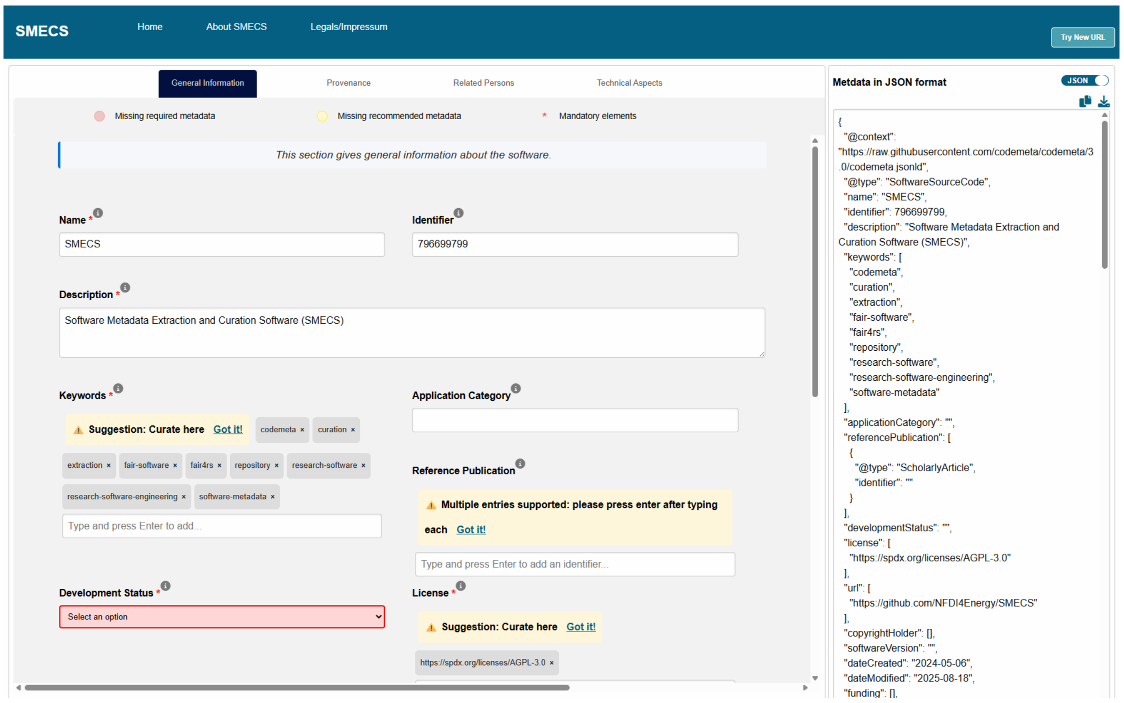
3.1 Extraction Phase

Upon entering the input data on the initial page in the **start phase** and initiating the **extraction phase**, the tool retrieves metadata. This process uses parts of HERMES [DBJ+22, KMD+25, MDB+25] which serves as the core of the backend as shown in Figure 1. Only small changes to HERMES (Version 0.8.1) were made, which will be included in the official HERMES release. For additional functionalities, plugins for HERMES were created, allowing long-term interoperability with future HERMES versions.

First, metadata are extracted from different sources using the harvesting step of HERMES. The extraction process is done by using four harvesters in HERMES (GitHub, GitLab, *CFF*, and *CodeMeta*). Table 2 gives an overview of the used sources for metadata. We extended the existing *CFF* and *CodeMeta* harvesters of HERMES to deal with remote repositories and implemented new harvesters for GitHub and GitLab. Up to 16 metadata elements can be extracted from the GitHub or GitLab API. These metadata are more focused on project information (such as repository name, description, and programming languages), collaboration, and activity (such as issues, pull requests, commits, and contributors). However, these sources do not cover citation metadata. On the other hand, *CFF* and *CodeMeta* files are more oriented toward software citation metadata and metadata related to software dissemination and reuse (such as author, versioning, publishers, licensing, and project identifiers like Digital Object Identifier (DOI)). Overall, the quality of the extracted metadata highly depends on the available information in a repository. All harvesters map their extracted metadata to *CodeMeta*. For this task, we used existing crosswalks from *CodeMeta*[6] and HERMES as well as created one crosswalk by ourselves, as shown in Table 2.

The extracted metadata are processed and merged using the process step of HERMES, resulting in a common set of metadata. These metadata are subsequently presented in the user interface in the **curation phase**. By using HERMES, we achieved an interoperable and modular architecture for the **extraction phase**, allowing it to easily integrate more harvesting sources.

[6] https://codemeta.github.io/crosswalk/github/, last access 2025-04-10



This section gives general information about the software.

Name *: SMECS

Identifier: 796699799

Description *: Software Metadata Extraction and Curation Software (SMECS)

Keywords *: Suggestion: Curate here — codemeta, curation, extraction, fair-software, fair4rs, repository, research-software, research-software-engineering, software-metadata — Type and press Enter to add...

Application Category:

Reference Publication: Multiple entries supported: please press enter after typing each — Type and press Enter to add an identifier...

Development Status *: Select an option

License *: Suggestion: Curate here — https://spdx.org/licenses/AGPL-3.0

```
{
  "@context":
"https://raw.githubusercontent.com/codemeta/codemeta/3
.0/codemeta.jsonld",
  "@type": "SoftwareSourceCode",
  "name": "SMECS",
  "identifier": 796699799,
  "description": "Software Metadata Extraction and
Curation Software (SMECS)",
  "keywords": [
    "codemeta",
    "curation",
    "extraction",
    "fair-software",
    "fair4rs",
    "repository",
    "research-software",
    "research-software-engineering",
    "software-metadata"
  ],
  "applicationCategory": "",
  "referencePublication": [
    {
      "@type": "ScholarlyArticle",
      "identifier": ""
    }
  ],
  "developmentStatus": "",
  "license": [
    "https://spdx.org/licenses/AGPL-3.0"
  ],
  "url": [
    "https://github.com/NFDI4Energy/SMECS"
  ],
  "copyrightHolder": [],
  "softwareVersion": "",
  "dateCreated": "2024-05-06",
  "dateModified": "2025-08-18",
  "funding": [],
```

Figure 3: SMECS curation page which allows the user to review and edit the extracted metadata and to add additional metadata

3.2 Curation Phase

Based on the metadata from the **extraction phase**, the extracted metadata are visualized in the **curation phase**. In Figure 3, this visualization is showcased based on an example GitHub URL. The extracted metadata are organized within an interface resembling a fillable form and are displayed across four main tabs: General Information, Provenance, Related Persons, and Technical Aspects. Each tab includes a brief description to help users quickly understand its purpose and the type of metadata it contains.

In general, some metadata elements are marked with an asterisk symbol to indicate that they are required for downloading the final output. When the automatic extraction of a required metadata element was not possible for the provided repository, the corresponding field is visually highlighted in red, indicating where manual input is needed to complete the metadata. Certain metadata elements, like reference publication, license, url, keywords, Funding, and programming languages, are accompanied by a small yellow suggestion box immediately after extraction. This color-coded indicator encourages users to review or curate these elements for improved accuracy, as their quality can often be enhanced through manual input. The suggestion box appears only once and disappears from the interface once accepted by the user. These suggestions are context-specific, offering targeted guidance for each metadata element. For example, in the case of reference publication, the hint informs users that multiple entries are supported.

To enhance the user experience when curating metadata elements such as programming languages and licenses, a list of suggestions is shown to the user. On the frontend, a programming language JSON file[7] was utilized, offering a comprehensive selection of programming languages. Additionally, a list of licenses[8] was fetched to provide users with a valid and up-to-date set of license options for repositories. Dynamic filtering enables users to quickly locate and select an option from a manageable subset, avoiding the overwhelm of a large, unfiltered list.

Once the metadata curation is completed in the **curation phase**, users can download the finalized result as a JSON file or copy that through the copy-to-clipboard feature in the JSON viewer section in the **export phase**. The *CodeMeta*-based metadata format facilitate easy integration with a variety of tools and applications for further use or analysis.

4 Research Design and Method, Data Collection and Analysis

For the evaluation, we focused on the aspect of usability, a core requirement for SMECS. Accordingly, we applied usability testing that mainly concentrated on the user interface in the **curation phase**.[9]

Usability testing is a commonly employed method to evaluate how easy a product is to use. This method allows for the identification of problems in the software development stage, focusing on users' ability to understand and interact with the designed functions [GG04]. Our usability testing aimed to assess how well users could navigate the user interface, understand the tool's functionalities, and interact with the available features.

For the usability testing, we recruited six researchers from our own research institute[10] as participants to conduct the study on-site, which enabled us to let the participants directly use SMECS on a test machine. All six participants (P1 to P6) were energy researchers at PhD level and engaged in medium to large-scale energy-related research projects. They mostly held master's degrees in computer science, with one participant holding a degree in engineering physics. The participants had varying levels of familiarity with metadata, rating their knowledge between 2 and 4 out of 5 (5 = most knowledge).

Our usability testing was structured in the following way. First, we gave the participants a short introduction to the session including promising confidentiality of the interview transcripts to avoid possible response biases [MN07]. Afterwards, the participants were asked to perform predefined tasks designed to evaluate key aspects of the user experience while also covering the main features of the tool. These tasks were structured around two scenarios: one using a familiar repository (provided by the participants themselves) and one using an unfamiliar repository (provided by us). In the first scenario, participants were instructed to enter a GitLab or GitHub repository URL along with an optional personal token and start the extraction. Afterwards, they were ask to review and curate the extracted information, add missing elements such as the license, and export the curated metadata as JSON file. They were also encouraged to explore the interface

[7] https://gist.github.com/calvinfroedge/defeb8fc6cdc0068e172, last access 2025-04-16

[8] https://raw.githubusercontent.com/spdx/license-list-data/master/json/licenses.json, last access 2025-04-16

[9] The evaluation was conducted on a previous version of SMECS (Commit number: aefc7a9). Since the evaluation, several changes have been made, including integrating with HERMES, adding features to support multiple inputs for one element, expanding the curation page by additional metadata elements, redistributing the elements over tabs, and improved responsiveness of the tool.

[10] OFFIS - Institute for Information Technology, https://www.offis.de/, last access 2025-04-16

and provide feedback throughout the process. In the second scenario, participants repeated a similar sequence using an unfamiliar repository, including obtaining a token if needed. They were instructed to compare the extracted metadata against a printed reference sheet, add metadata about contributors, and export the results. Finally, participants were asked to provide overall feedback on their experience, including suggestions for improvement and reflections on features.

In this phase, the think-aloud method was used to capture feedback, insights, and thoughts. In the think-aloud method, participants verbalize their thoughts while completing a specific task or recall their thoughts immediately after finishing the task [EA17]. All the ideas and thoughts expressed during the think-aloud method were audio-recorded in the session while testing the tool. This qualitative data was analyzed and documented in the same way as the data collected through semi-structured interviews.

To evaluate participants' perception of the user interface of SMECS, quantitative data was gathered through the System Usability Scale (SUS) questionnaire by Brooke [Bro95]. Employing the SUS allows for an evaluation of the perceived usability of a system with a limited sample size, providing a reliable assessment of user perceptions of a system or product [TS04]. The SUS questionnaire consists of ten statements, each on a five-point scale from "Strongly Disagree" to "Strongly Agree" [BKM09], and is used to assess usability in websites, software, and mobile applications [PPP13].

Additional qualitative data on the tool's usability, functionality, and overall effectiveness was obtained through semi-structured interviews with participants after they completed the SUS questionnaire. The interview guide covered participants' backgrounds, initial expectations, first-time experiences, specific features, encountered challenges, and overall impressions.

All experiments were conducted in English between September and October 2024 using the experimenter's laptop in an on-site setting. Each session lasted between 45 and 60 minutes. The audios of the interviews were recorded and later transcribed using Whisper [RKX+22]. After transcribing the audio recordings, we reviewed the transcripts, highlighted key points, and categorized the feedback into positive and negative opinions, reported bugs, and participant suggestions.

5 Results

Based on the evaluation method explained in Section 4, the main results of our study are presented in this section.

The SUS scores from the usability experiment are illustrated in Figure 4 as box plot[11]. They indicate that participants had a positive experience using the tool. The scores are consistently high, with a mean of 92.08 and a median of 92.5, reflecting a good user satisfaction. Five participants rated the tool within the "Excellent" range (above 85, based on [BKM09]), and one participant scored slightly lower, though still indicating high usability. This consistent pattern suggests the tool aligns well with user needs and expectations.

The qualitative feedback from the think-aloud method and semi-structured interviews offers helpful insights into user experiences and perceptions. Overall, impressions were positive with notable comments about clarity, navigation, and ease of use. Participants described the navigation

[11] The box plot was generated using [BBP22].

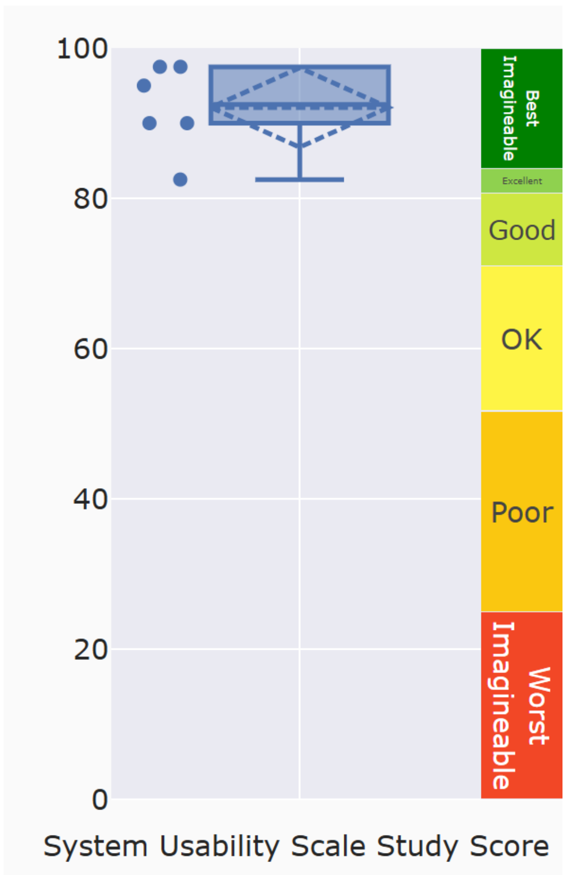


Figure 4: SUS scores for the usability experiment of SMECS as box plot with a mean of 92.08

through SMECS as intuitive (P1: “It’s very easy to use, very fast. You can change your information quite fast and it already gives you a lot of information.”, P2: “The tool is plain, but in a good sense.”). Two participants mentioned that the tool’s smooth and responsive functionality mirrors common online tools (P4: “The tool feels organized, easy to use and feels like an online tool”).

All participants noted that this segmentation of metadata and visualizing in different categories (tabs) made the page clean, organized, and easy to curate relevant metadata (P2: “The curation form looks really good. It looks clean in my opinion and less intimidating for users that are not technically inclined.”, P5: “I think it helps not to have too much information on one page, especially when the list of contributors gets very long. If all the metadata is on a single page, you might have to scroll a lot before reaching the next point.”). They acknowledged the color-coded clues (P3: “Colors in color coding perfectly matching to the expectation and they are perfectly matching to the standard, the typical things you would see on the Internet and you are used to.”). The participants liked the clarification of roles for contributors and authors, noting that they could distinguish these roles when adding or curating the corresponding metadata. All participants appreciated having both options for visualizing metadata during the **curation phase**, through the form and the JSON viewer (P1: “The live view of the JSON is pretty nice, and makes it easy to edit metadata in the form and see the result simultaneously.”).

The feedback included suggestions for incorporating advanced features, such as automated license selection and software description generation. Another proposed improvement was the feature to import a metadata set, obtained from SMECS or other software, into SMECS for the curation of existing research software metadata. Additionally, other potential improvements include implementing validation for specific metadata types and verifying the metadata presented in the JSON viewer. Furthermore, there were recommendations on including detailed information about authors and contributors, specifying what each provided. This can include whether they worked on code, documentation, writing, etc., similar to the way academic publications clarify the roles of each author.

As part of further development on the features and functionalities, we removed the copy feature from the contributors table. This feature was intended to help users to add a contributor to the authors table, however, its purpose was mostly unclear and misunderstood in the experiment. Most users did not understand the main purpose of this feature. Because of that, we introduced a different approach and decided to merge contributors and authors, presenting related information in a single table for both. This new design allows users to select or deselect roles, meaning a person can be defined as a contributor, an author, or both.

6 Discussion and Outlook

Although SMECS has already been rated as usable by researchers in the evaluation (see Section 5), the evaluation and its comparison to related work (see Section 2) showed several areas where it can be further extended.

Firstly, multiple existing functionalities can be further expanded. The extraction functionality can be enhanced, for example, by integrating SOMEF. Furthermore, the export functionality can be extended to provide more than the current download to a JSON file. For example, exports to the ORKG (similar to Betty’s research engine), software repositories (e.g., GitHub and GitLab, as merge requests like in HERMES), and/or software registries (similar to HERMES’ upload to invenioRDM) can be added.

Secondly, there are also additional use cases to use SMECS. Besides the export to software registries, it would also be interesting to test SMECS as an input tool for registries by directly integrating it into the registries as envisioned in [FN23]. This would simplify the creation of metadata in software registries. Still, we assume that manual curation will be needed to ensure the quality of the metadata. Furthermore, software management plans are an emerging topic in the area of research software [ABC+21]. Since software management plans contain a lot of metadata about research software, it would be an interesting approach to use SMECS or parts of it to semi-automate their creation. Additionally, there is potential for utilizing SMECS for more specific metadata schemas, such as the one currently being developed for energy research software [FN23, FWN25].

Our user study as presented in Section 4 and Section 5 was designed to evaluate the usability of SMECS. While the results provide valuable insights into the performance of the tool, some limitations of the evaluation must be acknowledged. Firstly, our study had a small number of participants with homogeneous backgrounds in computer science, which is not representative for all researchers. Despite this limitation, we were able to gain a good understanding of the tool's usability and have already incorporated some of the findings (e.g., metadata from existing *CodeMeta* files in a repository can now be extracted). Future studies should aim to recruit a more diverse sample to ensure that the results are generalizable to a broader group of researchers.

In addition, the use of predefined tasks may have limited the scope of the evaluation, but it also allowed us to test the usability of specific features of SMECS. Furthermore, the most recent version of the tool was not used in the usability study. Consequently, the findings may not be entirely applicable to this latest version. Moreover, the overall functionality of SMECS could be further analyzed by systematically testing a high number of repositories. In this way, the extraction quality could be analyzed for the different elements.

Overall, further research is necessary to explore the capabilities of SMECS and its usability by researchers. Also, the general impact of tools like SMECS on researchers' behavior with respect to the metadata creation remains an area for further exploration. Understanding the effectiveness of SMECS and similar tools in promoting metadata creation and publication is relevant for developing strategies to increase the adoption of metadata for research software.

In conclusion, we showed that SMECS is a usable tool for creating metadata for research software based on extraction. By building on HERMES, it achieves a high degree of interoperability with existing tools. Nevertheless, there are still opportunities for improvement, particularly in refining its extraction and export capabilities. We will focus on these aspects in the upcoming project ConnOSS [CMN25]. Additionally, more research, such as integrating other metadata schemas or conducting studies with more diverse participants, is required to better understand what is needed to enable researchers to create meaningful metadata for their research software.

Acknowledgements: The authors would like to thank the German Federal Government, the German State Governments, and the Joint Science Conference (GWK) for their funding and support as part of the NFDI4Energy and the Base4NFDI consortia. The work was partly funded by the German Research Foundation (DFG) – 501865131 and 521453681 within the German National Research Data Infrastructure (NFDI, www.nfdi.de).

Additionally, this research was partly funded by the Lower Saxony Ministry of Science and Culture under grant number 11-76251-13-3/19–ZN3488 (ZLE) within the Lower Saxony "Vorab" of the Volkswagen Foundation. It was supported by the Center for Digital Innovations Lower Saxony (ZDIN).

Bibliography

[ABC+21] R. Alves, D. Bampalikis, L. J. Castro, J. M. F. González, J. Harrow, M. Kuzak, E. Martin, F. E. Psomopoulos, A. Via. ELIXIR Software Management Plan for Life Sciences. Oct. 2021.
doi:10.37044/osf.io/k8znb

[ADF+25] S. Auer, J. D'Souza, K. E. Farfar, M. Y. Jaradeh, A. Jiomekong, O. Karras, A. Oelen, L. Snyder, M. Stocker, L. Vogt. Open Research Knowledge Graph: A Large-Scale Neuro-Symbolic Knowledge Organization System. In *Handbook on Neurosymbolic AI*

and Knowledge Graphs. Pp. 385–420. IOS Press, 2025.
doi:10.3233/FAIA250216

[BBP22] J. Blattgerste, J. Behrends, T. Pfeiffer. A Web-Based Analysis Toolkit for the System Usability Scale. 2022.
doi:10.1145/3529190.3529216

[BKM09] A. Bangor, P. Kortum, J. T. Miller. Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. *Journal of Usability Studies archive*, May 2009.
https://www.semanticscholar.org/paper/Determining-what-individual-SUS-scores-mean%3A-adding-Bangor-Kortum/3399f83ff6149dc65b52600f52ed372be5a6aa86

[Bro95] J. Brooke. SUS: A Quick and Dirty Usability Scale. In *Usability Eval. Ind.* Volume 189. CRC Press, 1st edition edition, Nov. 1995.
doi:10.1201/9781498710411-35

[CKB+22] N. P. Chue Hong, D. S. Katz, M. Barker, A.-L. Lamprecht, C. Martinez, F. E. Psomopoulos, J. Harrow, L. J. Castro, M. Gruenpeter, P. A. Martinez, T. Honeyman, A. Struck, A. Lee, A. Loewe, B. van Werkhoven, C. Jones, D. Garijo, E. Plomp, F. Genova, H. Shanahan, J. Leng, M. Hellström, M. Sinha, M. Kuzak, P. Herterich, Q. Zhang, S. Islam, S.-A. Sansone, T. Pollard, U. D. Atmojo, A. Williams, A. Czerniak, A. Niehues, A. C. Fouilloux, B. Desinghu, C. Richard, C. Gray, C. Erdmann, D. Nüst, D. Tartarini, H. Anzt, I. Todorov, J. McNally, J. Moldon, J. Burnett, K. Belhajjame, L. Sesink, L. Hwang, M. Roberto, M. D. Wilkinson, M. Servillat, M. Liffers, M. Fox, N. Lynch, P. M. Lavanchy, S. Gesing, S. Stevens, M. Cuesta, S. Peroni, S. Soiland-Reyes, T. Bakker, T. Rabemanantsoa, V. Sochat, Y. Yehudi. FAIR Principles for Research Software (FAIR4RS Principles). Technical report, FAIR4RS WG, May 2022.
doi:10.15497/RDA00068

[CMN25] L. J. Castro, B. Mathiak, A. Nieße. Connected Open Source Software - ConnOSS - Proposal. Technical report, Zenodo, June 2025.
doi:10.5281/zenodo.15616384

[Cod25] CodeMeta Generator. Software Heritage, Jan. 2025.
https://github.com/codemeta/codemeta-generator

[DBJ+22] S. Druskat, O. Bertuch, G. Juckeland, O. Knodel, T. Schlauch. Software Publications with Rich Metadata: State of the Art, Automated Workflows and HERMES Concept. Jan. 2022.
doi:10.48550/arXiv.2201.09015

[Dja23] Django. Django Software Foundation, Lawrence, Kansas, 2023.
https://www.djangoproject.com/

[DSC+21] S. Druskat, J. H. Spaaks, N. Chue Hong, R. Haines, J. Baker, S. Bliven, E. Willighagen, D. Pérez-Suárez, A. Konovalov. Citation File Format. Aug. 2021.
doi:10.5281/zenodo.5171937

[EA17] D. W. Eccles, G. Arsal. The Think Aloud Method: What Is It and How Do I Use It? *Qualitative Research in Sport, Exercise and Health*, Aug. 2017.
doi:10.1080/2159676X.2017.1331501

[EO13] European Organization For Nuclear Research, OpenAIRE. Zenodo. 2013.
doi:10.25495/7GXK-RD71

[FG21] V. Fernandez-Lancha Aranda, D. Garijo. SOMEF VIDER. Dec. 2021.
https://github.com/SoftwareUnderstanding/SOMEF-Vider

[FJ25a] S. Ferenz, A. Jafarbigloo. Software Metadata Extraction and Curation Software (SMECS). Carl von Ossietzky Universität Oldenburg, May 2025.
https://github.com/NFDI4Energy/SMECS

[FJ25b] S. Ferenz, A. Jafarbigloo. Software Metadata Extraction and Curation Software (SMECS). Zenodo, Aug. 2025.
doi:10.5281/zenodo.16892288

[FN23] S. Ferenz, A. Nieße. Towards Improved Findability of Energy Research Software by Introducing a Metadata-based Registry. *ing.grid* 1(2), Nov. 2023.
doi:10.48694/inggrid.3837

[FWN25] S. Ferenz, O. Werth, A. Nieße. Requirements for a Metadata Scheme to Enable FAIR Energy Research Software. *Electronic Communications of the EASST* 83, Feb. 2025.
doi:10.14279/eceasst.v83.2594

[GG04] L. Guarascio-Howard, R. Gray. Using Kinematics to Assess Software Usability and Ergonomic Risk Factors. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 48(10):1184–1188, Sept. 2004.
doi:10.1177/154193120404801008

[GGMR16] Y. Gil, D. Garijo, S. Mishra, V. Ratnakar. OntoSoft: A Distributed Semantic Registry for Scientific Software. In *2016 IEEE 12th International Conference on E-Science (e-Science)*. Pp. 331–336. Oct. 2016.
doi:10.1109/eScience.2016.7870916

[GKL$^+$21] M. Gruenpeter, D. S. Katz, A.-L. Lamprecht, T. Honeyman, D. Garijo, A. Struck, A. Niehues, P. A. Martinez, L. J. Castro, T. Rabemanantsoa, N. P. Chue Hong, C. Martinez-Ortiz, L. Sesink, M. Liffers, A. C. Fouilloux, C. Erdmann, S. Peroni, P. Martinez Lavanchy, I. Todorov, M. Sinha. Defining Research Software: A Controversial Discussion. Technical report, Zenodo, Sept. 2021.
doi:10.5281/zenodo.5504016

[GMD$^+$25] D. Garijo, A. Mao, H. Dharmala, C. Diwanji, J. Wang, A. Kelley, M. A. García, J. Ciucio-Kiss, J. Mendoza. A Framework for Creating Knowledge Graphs of Scientific Software Metadata. 2025.
doi:10.1162/qss_a_00167

[GOK+19] D. Garijo, M. Osorio, D. Khider, V. Ratnakar, Y. Gil. OKG-Soft: An Open Knowledge Graph with Machine Readable Scientific Software Metadata. In *2019 15th International Conference on eScience (eScience)*. Pp. 349–358. Sept. 2019.
doi:10.1109/eScience.2019.00046

[HCH+20] W. Hasselbring, L. Carr, S. Hettrick, H. Packer, T. Tiropanis. From FAIR Research Data toward FAIR and Open Research Software. *it - Information Technology* 62(1):39–47, Feb. 2020.
doi:10.1515/itit-2019-0040

[HDB+25] W. Hasselbring, S. Druskat, J. Bernoth, P. Betker, M. Felderer, S. Ferenz, B. Hermann, A.-L. Lamprecht, J. Linxweiler, A. Prat, B. Rumpe, K. Schoening-Stierand, S. Yang. Multi-Dimensional Research Software Categorization. *Computing in Science & Engineering*, pp. 1–10, 2025.
doi:10.1109/MCSE.2025.3555023

[HHHM22] H. Huysksens, L. Hülk, J. Huber, A. Michaltsis. Meta Tool. Reiner Lemoine Institut, Aug. 2022.
https://github.com/rl-institut/meta_tool

[IIR+21] J. Ison, H. Ienasescu, E. Rydza, P. Chmura, K. Rapacki, A. Gaignard, V. Schwämmle, J. Van Helden, M. Kalaš, H. Ménager. BiotoolsSchema: A Formalized Schema for Bioinformatics Software Description. *GigaScience* 10(1), 2021.
doi:10.1093/gigascience/giaa157

[JBM+23] M. B. Jones, C. Boettiger, A. C. Mayes, A. M. Smith, M. Gruenpeter, V. Lorentz, T. Morrell, D. Garijo, P. Slaughter, K. E. Niemeyer, Y. Gil, M. Fenner, K. Nowak, M. Hahnel, L. Coy, A. Allen, M. Crosas, A. Sands, N. Chue Hong, P. Cruse, D. S. Katz, C. Goble, B. Mecum, A. Gonzalez-Beltran, N. Ross. CodeMeta: An Exchange Schema for Software Metadata. Version 3.0. 2023.
https://codemeta.github.io/terms/

[KG21] A. Kelley, D. Garijo. A Framework for Creating Knowledge Graphs of Scientific Software Metadata. *Quantitative Science Studies* 2(4):1423–1446, Dec. 2021.
doi:10.1162/qss_a_00167

[KGK+24] P. Kuckertz, J. Göpfert, O. Karras, D. Neuroth, J. Schönau, R. Pueblas, S. Ferenz, F. Engel, N. Pflugradt, J. M. Weinand, A. Nieße, S. Auer, D. Stolten. DataDesc: A Framework for Creating and Sharing Technical Metadata for Research Software Interfaces. *Patterns*, p. 101064, Oct. 2024.
doi:10.1016/j.patter.2024.101064

[KMD+25] S. Kernchen, M. Meinel, S. Druskat, M. Fritzsche, D. Pape, O. Bertuch. Extending and Applying Automated HERMES Software Publication Workflows. *Electronic Communications of the EASST* 83, Feb. 2025.
doi:10.14279/eceasst.v83.2624

[MBL+14] J. Malone, A. Brown, A. L. Lister, J. Ison, D. Hull, H. Parkinson, R. Stevens. The Software Ontology (SWO): A Resource for Reproducibility in Biomedical Data Analysis, Curation and Digital Preservation. *Journal of Biomedical Semantics* 5(1):25, June 2014. doi:10.1186/2041-1480-5-25

[MDB+25] M. Meinel, S. Druskat, O. Bertuch, O. Knodel, D. Pape, K. Sophie, N. Heeb. Hermes. Mar. 2025. https://github.com/softwarepub/hermes

[MGF19] A. Mao, D. Garijo, S. Fakhraei. SoMEF: A Framework for Capturing Scientific Software Metadata from Its Documentation. In *2019 IEEE International Conference on Big Data (Big Data)*. Pp. 3032–3037. Dec. 2019. doi:10.1109/BigData47090.2019.9006447

[MN07] M. D. Myers, M. Newman. The Qualitative Interview in IS Research: Examining the Craft. *Information and Organization* 17(1):2–26, Jan. 2007. doi:10.1016/j.infoandorg.2006.11.001

[PPP13] S. C. Peres, T. Pham, R. Phillips. Validation of the System Usability Scale (SUS): SUS in the Wild. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 57(1):192–196, Sept. 2013. doi:10.1177/1541931213571043

[RKX+22] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, I. Sutskever. Robust Speech Recognition via Large-Scale Weak Supervision. Dec. 2022. https://arxiv.org/abs/2212.04356v1

[SRW24a] V. Seibert, A. Rausch, S. Wittek. Betty Research Engine. Nov. 2024. https://gitlab.com/tuc-isse/public/betty-research-engine

[SRW24b] V. Seibert, A. Rausch, S. Wittek. Betty's (Re)Search Engine: A Client-Based Search Engine for Research Software Stored in Repositories. *ing.grid* 1(2), May 2024. doi:10.48694/inggrid.3953

[TS04] T. S. Tullis, J. N. Stetson. A Comparison of Questionnaires for Assessing Website Usability. In *Usability Professional Association Conference*. 2004. https://dgs.ie/wp-content/uploads/2016/12/A-Comparison-of-Questionnaires-for-Assessing-Website-Usability-.pdf