Task Force Sub Group 3 - Review of Software Quality Attributes and Characteristics

Mario David, Miguel Colom, Daniel Garijo, Leyla Jael Castro, Violaine Louvet, Elisabetta Ronchieri, Massimo Torquati, Laura del Caño, Siew Hoon Leong, Maxime van den Bossche, et al.

▶ **To cite this version:**

Mario David, Miguel Colom, Daniel Garijo, Leyla Jael Castro, Violaine Louvet, et al.. Task Force Sub Group 3 - Review of Software Quality Attributes and Characteristics. European Comission. 2023, https://zenodo.org/record/8221384. ⟨hal-04184377⟩

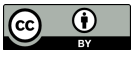
HAL Id: hal-04184377

https://hal.science/hal-04184377v1

Submitted on 21 Aug 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons CC BY 4.0 - Attribution - International License

Task Force Sub Group 3 - Review of Software Quality Attributes and Characteristics

David, Mario (LIP)
Colom, Miguel (ENS Paris-Saclay)
Garijo, Daniel (UPM)
Castro, Leyla Jael (ZB MED)
Louvet, Violaine (CNRS)
Ronchieri, Elisabetta (INFN/CNAF)
Torquati, Massimo (UNIPI)
del Caño, Laura (CSIC/CNB)
Leong, Siew Hoon (CSCS)
Van den Bossche, Maxime (KU Leuven)
Campos, Isabel (CSIC/IFCA)
Di Cosmo, Roberto (INRIA)

August 7, 2023

Contents

1 Quality Characteristics

Software Quality characteristics are taken from ISO/IEC 25010:2011(E) [8] except Scalability and Supportability [10].

Functional suitability: Degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions.

Availability: Degree to which a system, product or component is operational and accessible when required for use, adapted from [7].

Reliability: Degree to which a system, product or component performs specific functions under specified conditions for a specified period of time, adapted from [7]. **Notes**: Limitations in reliability are due to faults in requirements, design and implementation, or due to contextual changes.

Time behaviour: Degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements.

Performance: Performance relative to the amount of resources used under stated conditions. **Notes**: Resources can include other software products, the software and hardware configuration of the system, and materials (e.g. print paper, storage media).

Ease of use (Usability): Degree to which a product or system can be used by specified users to achieve specific goals with effectiveness, efficiency and satisfaction in a specified context of use. **Notes**: Adapted from ISO 9241-210. Usability can either be specified or measured as a product quality characteristic in terms of its sub-characteristics, or specified or measured directly by measures that are a subset of quality in use.

Fault tolerance: Degree to which a system, product or component operates as intended despite the presence of hardware or software faults, adapted from [7].

Security: Degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorisation. **Notes**: As well as data stored in or by a product or system, security also applies to data in transmission. Survivability (the degree to which a product or system continues to fulfil its mission by providing essential services in a timely manner in spite of the presence of attacks) is covered by recoverability (4.2.5.4). Immunity (the degree to which a product or system is resistant to attack) is covered by integrity (4.2.6.2). Security contributes to trust (4.1.3.2).

Maintainability: Degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers. **Notes**: Modifications can include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications. Modifications include those carried out by specialised support staff, and those carried out by business or operational staff, or end users. Maintainability includes installation of updates and upgrades. Maintainability can be interpreted as either an inherent capability of the product or system to facilitate maintenance activities, or the quality in use experienced by the maintainers for the goal of maintaining the product or system.

Recoverability: Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system. **Notes**: Following a failure, a computer system will sometimes be down for a period of time, the length of which is determined by its recoverability.

Operability / Manageability: Degree to which a product or system has attributes that make it easy to operate and control. **Notes**: Operability corresponds to controllability, (operator) error tolerance and conformity with user expectations as defined in ISO 9241-110.

Resource utilisation: Degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements. **Notes**: Human resources are included as part of efficiency (4.1.2).

Safety: Degree to which a product or system mitigates the potential risk to people in the intended contexts of use.

Interoperability: Degree to which two or more systems, products or components can exchange information and use the information that has been exchanged, adapted from [7].

Attractiveness: Renamed as user interface aesthetics. Degree to which a user interface enables pleasing and satisfying interaction for the user. **Notes**: This refers to properties of the product or system that increase the pleasure and satisfaction of the user, such as the use of colour and the nature of the graphical design.

Compatibility: Degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment, adapted from [7].

Installability: Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment.

Technical accessibility: Degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use. **Notes**: The range of capabilities includes disabilities associated with age. Accessibility for people with disabilities can be specified or measured either as the extent to which a product or system can be used by users with specified disabilities to achieve specified goals with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use, or by the presence of product properties that support accessibility.

Portability / Adaptability: Degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another, adapted from [7]. **Notes**: Portability can be interpreted as either an inherent capability of the product or system to facilitate porting activities, or the quality in use experienced for the goal of porting the product or system.

Modifiability: Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality. **Notes**: Implementation includes coding, designing, documenting and verifying changes. Modularity (4.2.7.1) and analysability (4.2.7.3) can influence modifiability. Modifiability is a combination of changeability and stability.

Reusability: Degree to which an asset can be used in more than one system, or in building other assets.

Scalability: 1.) Scalability is the ability of a system to either handle increases in load without impact on the performance of the system, or the ability to be readily enlarged [10] *OR* 2.) Scalability is the capability of algorithms, protocols, and applications to efficiently handle a growing amount of work or the demand of increasing its performance (according to some metrics) by adding resources to the system in which the software is running. Resources can be added to the single nodes (vertical scalability) and to the system as a whole (horizontal scalability) [4].

Supportability: 1.) Supportability is the ability of the system to provide information helpful for identifying and resolving issues when it fails to work correctly [10] *OR* 2.) Existence of a helpdesk or issue tracking, bug reporting, enhancements and general support [15].

Testability: Degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met, adapted from [7].

Confidentiality: Degree to which a product or system ensures that data are accessible only to those authorised to have access.

2 Quality Attributes

We follow the methodology for the survey, proposed by Kitchenham and Charters [9] which has the following steps:

1. Source selection and search: We have searched in the Scopus dataset, including the top five journals in software engineering related to software [1] and articles of the "International Conference on Software Engineering", one of the top venues for Software engineering. We also added documentclass and web resources that the Task Force subgroup considered relevants. The search included the keywords "software quality" in the title of the target publications.
2. Inclusion and exclusion criteria: Excluded journals not in the SE domain. Excluded articles not written in English.
3. Selection procedure: Skim article titles and abstracts. The process was performed by 2-3 people. Final list was agreed upon by the group through discussion about the relevance of the paper and analysis if that paper contains or proposes Software Quality attributes.
4. Review process: After following the selection procedure, we ended up with 19 articles, which have been reviewed in this survey. Some of the articles refer to the ISO/IEC 25010:2011(E) [8] or to its precursor ISO/IEC 9126, have been grouped together.

Journals: IEEE Transactions on Software Engineering, Empirical Software Engineering, Journal of Systems and Software, Software & Systems Modeling, Information and Software Technology, IEEE Software, Software Quality Journal. Query used:

```
TITLE ( software  AND quality )  AND
    ( LIMIT-TO ( EXACTSRCTITLE ,  "Software Quality Journal" )
    OR  LIMIT-TO ( EXACTSRCTITLE ,  "Proceedings International Conference On Software Engineering" )
    OR  LIMIT-TO ( EXACTSRCTITLE ,  "IEEE Transactions on Software Engineering" )
    OR  LIMIT-TO ( EXACTSRCTITLE ,  "Empirical Software Engineering" )
    OR  LIMIT-TO ( EXACTSRCTITLE ,  "Journal of Systems and Software" )
    OR  LIMIT-TO ( EXACTSRCTITLE ,  "Software & Systems Modeling" )
    OR  LIMIT-TO ( EXACTSRCTITLE ,  "Information and Software Technology" )
    OR  LIMIT-TO ( EXACTSRCTITLE ,  "IEEE Software" )
    )  AND  ( LIMIT-TO ( SUBJAREA ,  "COMP" )  OR  LIMIT-TO ( SUBJAREA ,  "ENGI" ) )
```

As a result, we obtained 272 results. Additional filtering: excluding papers with no abstracts, proceedings/workshop summary, and removed those which did not seem related by browsing the abstract and title. Also, removed those papers that did not seem to propose quality dimensions (e.g., if they talk about practices).

The metrics and attributes where subdivided into four categories and for each metric or attribute, a new codename was created by the EOSC Task Force:

- Source Code Metrics (**EOSC-SCMet**):
- Time metrics (**EOSC-TMet**):
- Qualitative Attributes (**EOSC-Qual**):
- DevOps - SW release and management Attributes (**EOSC-SWRelMan**):
- DevOps - Testing Attributes (**EOSC-SWTest**):
- Service Operability Attributes (**EOSC-SrvOps**):

Research Software levels correspond to the four user stories defined in subgroup 1 (Research Software Lifecycle) of this Task Force (Infrastructure for Quality Research Software) [2], and are the following:

1. **Individual** creating research software for own use (e.g. a PhD student): easy to implement, good practice for research software at any level.
2. **Team** creating an application or workflow for use within the team: easy to implement, good practice for research software at any level, useful for some basic coordination when more than one person participates.
3. **OSS** A team / community developing (possibly broadly applicable) open source research software or service platform: Open Source Software in general, all other cases.

[1] `https://research.com/journals-rankings/computer-science/software-programming`
[2] `https://docs.google.com/document/d/13TzhxNpGLtbFWYOmWSKkEHFC22ICmVpB/`

2.1 Source Code Metrics: Attribute type: EOSC-SCMet

EOSC-SCMet-01: Size of the software application: Maintainability

- The software product's rebuild value is estimated from the number of lines of code. This value is calculated in man-years using the Programming Languages Table of the Software Productivity Research Software Productivity Research [2]
- Research Software level: OSS.

EOSC-SCMet-02: % of redundant code: Maintainability, Modifiability

- A line of code is considered redundant if it is part of a code fragment (larger than 6 lines of code) that is repeated literally (modulo white-space) in at least one other location in the source code [2]
- Research Software level: OSS.

EOSC-SCMet-03: # Lines of code: Maintainability

- Number of lines for the whole software project or components/modules/classes/functions/methods [11, 2]
- Research Software level: Individual, Team, OSS.

EOSC-SCMet-04: % of assertions: Maintainability

- Percentage of lines of source code containing assertions. Assertions are used as a means for demonstrating that the program is behaving as expected and as an indication of how thoroughly the source classes have been tested on a per class level. [12]
- Research Software level: OSS.

EOSC-SCMet-05: Cyclomatic Complexity: Maintainability

- Cyclomatic complexity (i.e., the number of linearly independent paths through a program's source code) of the whole software component or modules/components/classes/functions/methods. Cyclomatic complexity is created by calculating the number of different code paths in the flow of the program. A program that has complex control flow requires more tests to achieve good code coverage and is less maintainable. [3] [18, 11, 2, 12]
- Research Software level: OSS.

EOSC-SCMet-06: Cyclomatic Complexity test/source ratio: Maintainability

- Ratio between the sum of cyclomatic complexities of all tests and the source code [12]
- Research Software level: OSS.

EOSC-SCMet-07: Number of arguments: Maintainability

- Number of arguments or parameters used in functions. Functions with many parameters may be a symptom of bad encapsulation [2, 13]
- Research Software level: Team, OSS.

EOSC-SCMet-08: Number of function calls: Maintainability

- Measures the interdependencies between unique classes through parameters, local variables, return types, method calls, generic or template instantiations, base classes, interface implementations, fields defined on external types, and attribute decoration. Good software design dictates that types and methods should have high cohesion and low coupling. High coupling indicates a design that is difficult to reuse and maintain because of its many interdependencies on other types. Source: `https://docs.microsoft.com/en-us/visualstudio/code-quality/code-metrics-values?view=vs-2022`. [11, 13]

[3] `https://docs.microsoft.com/en-us/visualstudio/code-quality/code-metrics-values?view=vs-2022`

- Research Software level: Team, OSS.

EOSC-SCMet-09: Modularity: Maintainability, Functional suitability

- Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components. Also number of modules. [8, 11, 1, 13, 17]
- Research Software level: Individual, Team, OSS.

EOSC-SCMet-10: Number of comments: Modifiability

- Number of lines corresponding to comments for the whole software or per modules/components / classes / functions / methods [18, 11, 13]
- Research Software level: Individual, Team, OSS.

EOSC-SCMet-11: Maintainability Index (MI): Maintainability

- Degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers. Also, the Maintainability Index (MI) calculates an index value between 0 and 100 that represents the relative ease of maintaining the code. A high value means better maintainability. Color coded ratings can be used to quickly identify trouble spots in your code. A green rating is between 20 and 100 and indicates that the code has good maintainability. A yellow rating is between 10 and 19 and indicates that the code is moderately maintainable. A red rating is a rating between 0 and 9. Source `https://docs.microsoft.com/en-us/visualstudio/code-quality/code-metrics-values?view=vs-2022`: [8, 11]
- Research Software level: OSS.

EOSC-SCMet-12: Internal cohesion: Maintainability

- Cohesion describes how related the functions within a single module are. Low cohesion implies that a given module performs tasks which are not very related to each other and hence can create problems as the module becomes large. Source `https://en.wikipedia.org/wiki/Coupling_(computer_programming)` [7, 11]
- Research Software level: OSS.

EOSC-SCMet-13: Test class to source class ratio: Reliability

- Control measure to counter the confounding effect of class size. The coefficient is calculated by TLOC+/SLOC* (TLOC = number of test classes, TLSOC = number of source classes) [12]
- Research Software level: OSS.

EOSC-SCMet-14: Coupling Between Objects (CBO) ratio: Maintainability

- Ratio between the CBO in the tests and the whole source code. The higher the inter-object class coupling, the more rigorous the testing should be [12]
- Research Software level: OSS.

EOSC-SCMet-15: Depth of inheritance Tree (DIT) ratio: Maintainability

- Ratio between the DIT of the tests and the DIT of the source code. A higher DIT indicates desirable reuse but adds to the complexity of the code because a change or a failure in a super class propagates down the inheritance tree. [12]
- Research Software level: OSS.

EOSC-SCMet-16: Weighted Methods per Class (WMC) ratio: Maintainability

- Ratio between the WMC of the tests with respect of the WMC of the whole source code. This measure serves to compare the testing effort on a method basis. WMC is an indicator of fault-proneness [12]
- Research Software level: OSS.

EOSC-SCMet-17: Source lines of code (SLOC) ratio: Maintainability

- SLOC* of the whole project with respect to the minimum SLOC* of its components [12]
- Research Software level: OSS.

EOSC-SCMet-18: Number of include files: Maintainability

- Number of imported modules [13]
- Research Software level: Team, OSS.

EOSC-SCMet-19: Number of conditions per module: Maintainability

- Number of condition per module [13]
- Research Software level: Individual, Team, OSS.

EOSC-SCMet-20: Number of loops per module: Maintainability

- Number of loops per module [13]
- Research Software level: Individual, Team, OSS.

2.2 Time and Performance Metrics: Attribute type: EOSC-TMet

EOSC-TMet-01: Effort required for changes: Reliability, Supportability

- Time and resources dedicated to resolve an issue [11]
- Research Software level: Team, OSS.

EOSC-TMet-02: # Resolved bugs: Supportability

- Number of resolved bugs per period of time [11]
- Research Software level: Team, OSS.

EOSC-TMet-03: # Open bugs: Supportability

- Number of open bugs/issues per period of time [11]
- Research Software level: Team, OSS.

EOSC-TMet-04: Defect rates: Maintainability

- The number of outstanding defects in a product per period of time [5]
- Research Software level: OSS.

EOSC-TMet-05: Integrity: Integrety, Maintainability

- Resource cost expended to solve problems caused by inconsistencies within the system. This may be measured in terms of staff time employed to fix problems and user time wasted. Also, degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data. [8, 6]
- Research Software level: OSS.

EOSC-TMet-06: Maintainability: Maintainability

- Measured by the resources spent in terms of time and cost in keeping a system up and running over a period of time. [6, 3]
- Research Software level: OSS.

EOSC-TMet-07: Adaptability: Reusability. Adaptability

- Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments. [8, 6, 3]
- Research Software level: OSS.

EOSC-TMet-08: # Registered users: Operability

- The metric Number of registered users is to be collected. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-TMet-09: # Active users: Operability

- The metric Number of active users over a given period of time may be collected. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-TMet-10: Amount computing resources: Operability, Performance Efficiency

- The metric amount of computing resources per user or group of users is collected. An example is CPU x hours. Resource utilization. [8, 15]
- Research Software level: Team, OSS.

EOSC-TMet-11: Amount storage resources: Operability, Performance Efficiency

- The metric amount of storage resources per user or group of users is collected. An example is GByte x hours. Resource utilization and capacity. [8, 15]
- Research Software level: Team, OSS.

2.3 Qualitative: Attribute type: EOSC-Qual

EOSC-Qual-01: Complexity of diagrams: Maintainability. Reusability

- Complexity of diagrams (UML) for the whole software or modules/components [11]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-02: Complexity of architecture: Maintainability. Reusability

- Architecture showing modules and interactions [11, 19]
- Research Software level: Team, OSS.

EOSC-Qual-03: Complexity of a use case: Maintainability. Reusability. Usability

- Complexity of use case diagrams (UML) [11]
- Research Software level: OSS.

EOSC-Qual-04: Sustainable community: Supportability

- An active community is behind the software product [1]
- Research Software level: Team, OSS.

EOSC-Qual-05: User satisfaction: Attractiveness

- Degree to which a user interface enables pleasing and satisfying interaction for the user [8, 19]
- Research Software level: OSS.

EOSC-Qual-06: Usability: Usability

- Measured using user surveys. However, it may also be assessed in terms of calls upon support staff, e.g. number o f requests for help or support staff time expended. Also, degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use [8, 19, 6, 3]
- Research Software level: OSS.

EOSC-Qual-07: Reliability: Reliability

- The reliability of systems from the user's point of view is concerned with three things: (1) How often does it go wrong? (2) How long is it unavailable? (3) Is any information lost at recovery? Also, degree to which a system, product or component performs specified functions under specified conditions for a specified period of time. [8, 6, 3]
- Research Software level: OSS.

EOSC-Qual-08: Efficiency: Time behavior, Performance

- Critical resources needed by the operator to carry out the critical tasks can be monitored e.g. if staff time is the critical resource, the elapsed time taken is the best measure o f efficiency. On the other hand, if the computer disk storage is critical, then the percentage of disk space required to carry out the task is the best measure available. [8, 6, 3]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-09: Portability: Portability

- Degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another. [8, 6, 3, 17, 16]
- Research Software level: Team, OSS.

EOSC-Qual-10: Timeliness: Supportability

- Assessed in terms of the costs of non-delivery. These will possibly include staff time and lost sales. It may also be assessed in terms of the number of days departure from the date agreed with client. [6]
- Research Software level: OSS.

EOSC-Qual-11: Cost-Benefit efficiency: Maintainability

- Measured in simple financial terms. The costs of installing and maintaining the system are weighed against the assessment of business benefits [6]
- Research Software level: OSS.

EOSC-Qual-12: Ease of transition: Compatibility

- Assessed in terms of staff time expended. The effectiveness of transition may be assessed in terms of the quality of the resulting system particularly the area of integrity. [8, 6]
- Research Software level: OSS.

EOSC-Qual-13: Accessibility: Technical accessibility

- Degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use. [8, 3]
- Research Software level: Team, OSS.

EOSC-Qual-14: Accountability: Performance, Resource utilization

- Degree to which the actions of an entity can be traced uniquely to the entity. Its usage can be measured; critical segments of code can be instrumented with probes to measure timing, whether specified branches are exercised, etc. [8, 3]

- Research Software level: OSS.

EOSC-Qual-15: Accuracy: Performance

- Code outputs are sufficiently precise to satisfy their intended use [3]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-16: Communicativeness: Ease of use

- Metric which facilitates the specification of inputs and provides outputs whose form and content are easy to assimilate and useful. [3, 16]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-17: Completeness: Supportability, Manageability

- All software parts are present and each part is fully developed. [8, 3]
- Research Software level: OSS.

EOSC-Qual-18: Conciseness: Supportability, Resource utilization

- Redundant information about the software code is not present. [3]
- Research Software level: OSS.

EOSC-Qual-19: Consistency: Maintainability, Interoperability, Compatibility

- Source code contains uniform notation, terminology and symbology within itself [3, 16]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-20: Legibility: Supportability, Maintainability

- Software function is easily discerned by reading the code [3]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-21: Modifiability: Modifiability

- Degree to which a product or system can be effectively and efficiently modified without introducing defects ordegrading existing product quality. [8, 3]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-22: Robustness: Safety

- Software can continue to perform despite some violation of the assumptions in its specification [3]
- Research Software level: OSS.

EOSC-Qual-23: Self-containedness: Supportability

- Software performs all its explicit and implicit functions within itself . [3]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-24: Self-descriptiveness: Supportability, Ease of use

- Software contains enough information for a reader to determine or verify its objectives, assumptions, constraints, inputs, outputs, components, and revision status. [3]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-25: Structuredness: Modifiability, Reusability

- Software possesses a definite pattern of organization of its interdependent parts. [3]
- Research Software level: OSS.

EOSC-Qual-26: Understandability: Supportability

- Software purpose is clear to the inspector. [3]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-27: Intellectual Property: Supportability, Reusability

- There are multiple statements embedded into the software product describing unrestricted rights and any conditions for use, including commercial and non-commercial use, and the recommended citation. The list of developers is embedded in the source code of the product, in the documentation, and in the expression of the software upon execution. The intellectual property rights statements are expressed in legal language, machine-readable code, and in concise statements in language that can be understood by laypersons, such as a pre-written, recognizable license. [17]
- Research Software level: Individual, Team, OSS.

EOSC-Qual-28: Extensibility: Attractiveness

- There is evidence that the software project has been extended externally by users outside of the original development group using existing documentation only. There is a clear approach for modifying and extending features across a in multiple scenarios, with specific documentation and features to allow the building of extensions which are used across a range of domains by multiple user groups. There may be a library available of user-generated content for extensions and user generated documentation on extension is also available. [17]
- Research Software level: OSS.

EOSC-Qual-29: Standards compliance: Functional suitability

- Compliance with open or internationally recognized standards for the software and software development process, is evident and documented, and verified through testing of all components.Ideally independent verification is documented through regular testing and certification from an independent group. [17]
- Research Software level: Team, OSS.

EOSC-Qual-30: Internationalisation and localisation: Usability

- Demonstrable usability: Software has been tested with multiple pseudo or genuine translations. [17]
- Research Software level: OSS.

2.4 DevOps-SW release and management: Attribute type: EOSC-SWRelMan

EOSC-SWRelMan-01: Open source : Supportability, Maintainability, Availability

- Following the open-source model, the source code being produced is open and publicly available to promote the adoption and augment the visibility of the software developments. [14, 16]
- Research Software level: Individual, Team, OSS.

EOSC-SWRelMan-02: Version Control System (VCS): Supportability, Maintainability

- Source code uses a Version Control System (VCS). All software components delivered by the same project agree on a common VCS. [14]
- Research Software level: Individual, Team, OSS.

EOSC-SWRelMan-03: Source code hosting: Supportability, Maintainability

- Source code produced within the scope of a broader development project resides in a common organization of a version control repository hosting service. [14]

- Research Software level: Team, OSS.

EOSC-SWRelMan-04: Working state version: Maintainability

- The main branch in the source code repository maintains a working state version of the software component. Main branch is protected to disallow force pushing, thus preventing untested and un-reviewed source code from entering the production-ready version. New features are merged in the main branch whenever the agreed upon SQA criteria is fulfilled. [14]
- Research Software level: Team, OSS.

EOSC-SWRelMan-05: Changes branches: Maintainability

- New changes in the source code are placed in individual branches. Branches follow a common nomenclature, usually by prefixing, to differentiate change types (e.g. feature, release, fix). [14]
- Research Software level: Team, OSS.

EOSC-SWRelMan-06: Good patching practice: Maintainability

- Secondary long-term branch that contains the changes for the next software release exists. Next release changes come from the individual branches. Once ready for release, changes in the secondary long-term branch are merged into the main branch and versioned. At that point in time, main and secondary branches are aligned. This step marks a production release. [14, 16]
- Research Software level: OSS.

EOSC-SWRelMan-07: Support: Maintainability, Operability

- Existance of an issue tracking system or helpdesk, facilitates structured software development. Leveraging issues to track down both new enhancements and defects (bugs, documentation typos). Applies as well to services to report operational and users issues. [18, 17, 14, 15]
- Research Software level: Team, OSS.

EOSC-SWRelMan-08: Code review: Maintainability

- Code reviews are done in the agreed peer review tool within the project, with the following functionality: (a) Allows general and specific comments on the line or lines that need to be reviewed. (b) Shows the results of the required change-based test executions. (c) Allows to prevent merges of the candidate change whenever not all the required tests are successful. Exceptions to this rule cover the third-party or upstream contributions which MAY use the existing mechanisms or tools for code review provided by the target software project. This exception is only be allowed whenever the external revision lifecycle does not interfere with the project deadlines. [18, 1, 19, 14]
- Research Software level: Team, OSS.

EOSC-SWRelMan-09: Semantic Versioning : Maintainability

- Semantic Versioning specification is followed for tagging the production releases. [14, 16]
- Research Software level: Individual, Team, OSS.

EOSC-SWRelMan-10: Open-source license: Supportability, Maintainability

- As open-source software, source code adheres to an open-source license to be freely used, modified and distributed by others. Non-licensed software is exclusive copyright by default. [14, 16]
- Research Software level: Individual, Team, OSS.

EOSC-SWRelMan-11: Metadata: Supportability, Maintainability, Availability

- A metadata file (such as Codemeta or Citation File Format) exists along side the code, under its VCS. The metadata file is updated when needed, as is the case of a new version. [14]

- Research Software level: Team, OSS.

EOSC-SWRelMan-12: Packaging: Installability

- Production-ready code is built as an artifact that can be installed on a system. [17, 14, 16]
- Research Software level: Team, OSS.

EOSC-SWRelMan-13: Register/publish artifact: Installability

- The built artifact is uploaded and registered into a public repository of such artifacts. [14]
- Research Software level: Team, OSS.

EOSC-SWRelMan-14: Notification upon registration: Installability

- Upon success of the package delivery process, a notification is sent to pre-defined parties such as the main developer or team. [14]
- Research Software level: OSS.

EOSC-SWRelMan-15: Code deployment: Installability

- Production-ready code is deployed as a workable system with the minimal user or system administrator interaction leveraging software configuration management (SCM) tools. [14]
- Research Software level: OSS.

EOSC-SWRelMan-16: Software Configuration Management (SCM) as code: Installability

- A software configuration management (SCM) module is treated as code. Version controlled, it resides in a different repository than the source code to facilitate the distribution. [14]
- Research Software level: OSS.

EOSC-SWRelMan-17: SCM tool: Installability

- All software components delivered by the same project agree on a common SCM tool. However, software products are not restricted to provide a unique solution for the automated deployment. [14]
- Research Software level: OSS.

EOSC-SWRelMan-18: SCM code changes: Installability

- Any change affecting the applications deployment or operation is subsequently reflected in the relevant SCM modules. [14]
- Research Software level: OSS.

EOSC-SWRelMan-19: SCM official repositories: Installability

- Official repositories provided by the manufacturer are used to host the SCM modules, thus augmenting the visibility and promote external collaboration. [14]
- Research Software level: OSS.

EOSC-SWRelMan-20: Installability: Installability

- Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment. Also, a production-ready SW or service is deployed as a workable system with the minimal user or system administrator interaction leveraging Infratructure as Code templates. [8, 15]
- Research Software level: OSS.

EOSC-SWRelMan-21: Preserve immutable infrastructures: Installability

- Any future change to a deployed Service is done in the form of a new deployment, in order to preserve immutable infrastructures. [15]
- Research Software level: OSS.

EOSC-SWRelMan-22: Infrastructure as Code (IaC) validation: Installability

- IaC is validated by specific (unit) testing frameworks for every change being done. [15]
- Research Software level: OSS.

EOSC-SWRelMan-23: Packaging of tarballs: Installability

- Tarballs are not included in the distribution directory. Packaged tarballs are not extracted in the distribution directory [16]
- Research Software level: Individual, Team, OSS.

EOSC-SWRelMan-24: Design for upgradability: Compatibility

- Installation layout support different versions and code releases [16]
- Research Software level: Team, OSS.

EOSC-SWRelMan-25: Provide checksums: Maintainability

- Code releases provide checksums with each binaries (tarballs, RPMs, etc.). [16]
- Research Software level: OSS.

EOSC-SWRelMan-26: Documentation version controlled: Supportability

- Documentation is version controlled. [15]
- Research Software level: Team, OSS.

EOSC-SWRelMan-27: Documentation as code: Supportability, Maintainability, Reusability

- Documentation is treated as code and resides in the same repository where the source code lies. [14]
- Research Software level: Team, OSS.

EOSC-SWRelMan-28: Documentation formats: Supportability, Maintainability, Reusability

- Documentation uses plain text format using a markup language, such as Markdown or reStructuredText. All software components delivered by the same project agree on a common markup language. [14, 16]
- Research Software level: Team, OSS.

EOSC-SWRelMan-29: Documentation online: Supportability, Availability

- Documentation is online available in a documentation repository. Documentation is rendered automatically. [14, 15]
- Research Software level: Team, OSS.

EOSC-SWRelMan-30: Documentation updates: Supportability, Maintainability, Reusability

- Documentation is updated on new software or service versions involving any substantial or minimal change in the behavior of the application. [14, 15, 16]
- Research Software level: Individual, Team, OSS.

EOSC-SWRelMan-31: Documentation updates if inaccurate/unclear: Supportability, Maintainability, Reusability

- Documentation is updated whenever reported as inaccurate or unclear. [14, 15]

- Research Software level: Team, OSS.

EOSC-SWRelMan-32: Documentation production: Supportability, Maintainability, Reusability

- Documentation is produced according to the target audience, varying according to the software component specification. The identified types of documentation and their content are README file(one-paragraph description of the application, a "Getting Started" step-by-step description on how to get a development environment running, automated test execution how-to, links to external documentation below, contributing code of conduct, versioning specification, author list and contacts, license information and acknowledgements), Developer documentations (Private API documentation, structure and interfaces and build documentation), Deployment and Administration documentations (Installation and configuration guides, service reference card, FAQs and troubleshooting) and user documentations (Public API documentation and command-line reference). [1, 17, 14, 15, 16]
- Research Software level: Individual, Team, OSS.

EOSC-SWRelMan-33: Documentation PID: Supportability

- Documentation has a Persistent Identifier (PID). [15]
- Research Software level: Team, OSS.

EOSC-SWRelMan-34: Documentation license: Supportability

- Documentation has a non-software license. [15]
- Research Software level: Team, OSS.

2.5 DevOps - Testing: Attribute type: EOSC-SWTest

EOSC-SWTest-01: Code style: Maintainability, Testability

- Each individual software complies with a de-facto code style standard for all the programming languages used in the codebase. Multiple standard style compliance is possible. [14, 16]
- Research Software level: Individual, Team, OSS.

EOSC-SWTest-02: Unit tests: Maintainability, Testability

- Minimum acceptable code coverage threshold is 70%. Unit testing coverage is higher for those sections of the code identified as critical by the developers, such as units part of a security module. Unit testing coverage may be lower for external libraries or pieces of code not maintained within the product's code base. [1, 12, 3, 17, 14, 16]
- Research Software level: Team, OSS.

EOSC-SWTest-03: Test doubles: Functional suitability, Testability

- When working on automated testing, Test Doubles (i.e., objects or procedures that look and behave like their release-intended counterparts, but are actually simplified versions that reduce the complexity and facilitate testing) are used to mimic a simplistic behavior of objects and procedures. [14, 15]
- Research Software level: Team, OSS.

EOSC-SWTest-04: Test-Driven Development (TDD): Functional suitability, Maintainability, Testability

- Software requirements are converted to test cases, and these test cases are checked automatically. Also, degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met. [8, 5, 19, 14]
- Research Software level: OSS.

EOSC-SWTest-05: API testing: Functional suitability, Testability

- API testing MUST cover the validation of the features outlined in the specification (aka contract testing). [15]
- Research Software level: Team, OSS.
- Research Software type: Service.

EOSC-SWTest-06: Integration testing: Functional suitability, Testability, Interoperability

- Whenever a new functionality is involved, integration testing guarantees the operation of any previously-working interaction with external Services [8, 15]
- Research Software level: OSS.

EOSC-SWTest-07: Functional testing: Functional suitability, Testability

- Functional testing covers the full scope -e.g. positive, negative, edge cases- for the set of functionality that the software or service claims to provide. [8, 15]
- Research Software level: Team, OSS.
- Research Software type: Service.

EOSC-SWTest-08: Performance testing: Functional suitability, Testability

- Performance testing is carried out to check the Software Application performance under varying loads. [8, 15]
- Research Software level: OSS.

EOSC-SWTest-09: Stress testing: Functional suitability, Testability

- Stress testing is carried out to check the Service to determine the behavioral limits under sudden increased load. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SWTest-10: Scalability testing: Functional suitability, Testability

- Scalability testing is carried out to check the Service ability to scale up and/or scale out when its load reaches the limits. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SWTest-11: Elasticity testing: Functional suitability, Testability

- Elasticity testing is carried out to check the Service ability to scale out or scale in, depending on its demand or workload. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SWTest-12: Open Web Application Security Project (OWASP): Security

- Application is compliant with Open Web Application Security Project (OWASP) secure coding guidelines [14]
- Research Software level: OSS.

EOSC-SWTest-13: Static Application Security Testing (SAST): Security

- Source code uses automated linter tools to perform static application security testing (SAST) that flag common suspicious constructs that may cause a bug or lead to a security risk (e.g. inconsistent data structure sizes or unused resources). [14]
- Research Software level: OSS.

EOSC-SWTest-14: Security code reviews: Security

- Security code reviews for certain vulnerabilities is done as part of the identification of potential security flaws in the code. Inputs come from automated linters and manual penetration testing results. [6, 17, 14]
- Research Software level: OSS.

EOSC-SWTest-15: No world-writable files or directories: Security

- World-writable files or directories are not be present in the product's configuration or logging locations. [14]
- Research Software level: Team, OSS.

EOSC-SWTest-16: Public endpoints and APIs encrypted: Security

- The Service public endpoints and APIs are secured with data encryption. [15]
- Research Software level: Team, OSS.
- Research Software type: Service.

EOSC-SWTest-17: Strong ciphers: Security

- The Service uses strong ciphers for data encryption. [15]
- Research Software level: Team, OSS.
- Research Software type: Service.

EOSC-SWTest-18: Authentication and Authorisation: Security, Technical accessibility

- The Service has an authentication and authorization mechanism. The Service validates the credentials and signatures. [17, 15]
- Research Software level: Team, OSS.
- Research Software type: Service.

EOSC-SWTest-19: API security assessment: Security

- API testing includes the assessment of the security-related criteria. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SWTest-20: Service compliance with data regulations (GDPR): Security

- The Service handles personal data in compliance with the applicable regulations, such as the General Data Protection Regulation (GDPR) within the European boundaries. [15]
- Research Software level: Individual, Team, OSS.
- Research Software type: Service.

EOSC-SWTest-21: Dynamic Application Security Testing (DAST): Security

- DAST checks are executed, through the use of ad-hoc tools, directly to an operational Service in order to uncover runtime security vulnerabilities and any other environment-related issues (e.g. SQL injection, cross-site scripting or DDOS). The latest release of OWASP's Web Security Testing Guide and the NIST's Technical Guide to Information Security Testing and Assessment are considered for carrying out comprehensive Service security testing. [15]
- Research Software level: OSS.

EOSC-SWTest-22: Interactive Application Security Testing (IAST): Security

- Interactive Application Security Testing (IAST), analyzes code for security vulnerabilities while the app is run by an automated test. IAST is performed to a service in an operating state. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SWTest-23: Security penetration testing: Security

- Penetration testing (manual or automated) is part of the application security verification effort. [15]
- Research Software level: OSS.

EOSC-SWTest-24: Security assessment: Security

- The security assessment of the target system configuration is particularly important to reduce the risk of security attacks. The benchmarks delivered by the Center for Internet Security (CIS) and the NIST's Security Assurance Requirements for Linux Application Container Deployments is considered for this task. Also, degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization. [8, 15]
- Research Software level: OSS.

EOSC-SWTest-25: Security as Code (SaC) Testing: Security

- IaC testing covers the security auditing of the IaC templates (SaC) in order to assure the deployment of secured Services. For all the third-party dependencies used in the IaC templates (including all kind of software artefacts, such as Linux packages or container-based images) [15]
- Research Software level: OSS.

2.6 Service Operability: Attribute type: EOSC-SrvOps

EOSC-SrvOps-01: Acceptable Usage Policy (AUP): Supportability

- An Acceptable Usage Policy (AUP) is declared. AUP Is a set of rules applied by the owner, creator or administrator of a network, Service or system, that restrict the ways in which the network, Service or system may be used and sets guidelines as to how it should be used. The AUP can also be referred to as: Acceptable Use Policy or Fair Use Policy. [15]
- Research Software level: OSS.

EOSC-SrvOps-02: Access Policy and Terms of Use: Supportability

- An Access Policy or Terms of Use (APTU) is declared. APTU represents a binding legal contract between the users (and/or customers), and the Provider of the Service. The Access Policy mandates the users (and/or customers) access to and the use of the Provider's Service. [15]
- Research Software level: OSS.

EOSC-SrvOps-03: Privacy policy: Supportability

- A Privacy Policy is declared, with a data privacy statement informing the users (and/or customers), about which personal data is collected and processed when they use and interact with the Service. It states which rights the users (and/or customers) have regarding the processing of their data. [15]
- Research Software level: OSS.

EOSC-SrvOps-04: Operational Level Agreement (OLA): Supportability

- The Service includes an Operational Level Agreement (OLA) with the infrastructure where it is integrated. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SrvOps-05: Service Level Agreement (SLA): Supportability

- The Service includes Service Level Agreement (SLA) with user communities. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SrvOps-06: Monitoring service public endpoints: Availability, Reliability

- The Service public endpoints are monitored, such as probes measuring the http or https response time [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SrvOps-07: Monitoring service public APIs: Availability, Reliability

- The Service public APIs are monitored through the use of functional tests. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SrvOps-08: Monitoring service Web Interface: Availability, Reliability

- The Service Web interface are monitored, this can be accoplished through the use of automated and periodic functional tests. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SrvOps-09: Monitoring security public endpoints and APIs: Availability, Reliability

- The Service is monitored for public endpoints and APIs secured and strong ciphers for encryption. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SrvOps-10: Monitoring security DAST: Availability, Reliability

- The Service is monitored with DAST security checks. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SrvOps-11: Monitoring infrastructure: Availability, Reliability

- The Service is monitored for infrastructure-related criteria. [15]
- Research Software level: OSS.
- Research Software type: Service.

EOSC-SrvOps-12: Monitoring with Unit tests: Availability, Reliability

- IaC (unit) tests are reused as monitoring tests, thus avoiding duplication. [15]
- Research Software level: OSS.

References

[1] Mark Aberdour. "Achieving Quality in Open-Source Software". In: *IEEE Software* 24.1 (Jan. 2007), pp. 58–64. ISSN: 0740-7459. DOI: 10.1109/MS.2007.2. URL: http://ieeexplore.ieee.org/document/4052554/ (visited on 06/24/2022).

[2] Robert Baggen et al. "Standardized code quality benchmarking for improving software maintainability". In: *Software Quality Journal* 20.2 (June 2012), pp. 287–307. ISSN: 0963-9314, 1573-1367. DOI: 10.1007/s11219-011-9144-9. URL: http://link.springer.com/10.1007/s11219-011-9144-9 (visited on 06/24/2022).

[3] B.W. Boehm, J.R. Brown, and M. Lipow. "Quantitative evaluation of software quality". In: *Proceedings - 2nd International Conference on Software Engineering, ICSE 1976*. San Francisco; United States, Oct. 13, 1976, pp. 592–605.

[4] André B. Bondi. "Characteristics of Scalability and Their Impact on Performance". In: *Proceedings of the 2nd International Workshop on Software and Performance*. WOSP '00. Ottawa, Ontario, Canada: Association for Computing Machinery, 2000, pp. 195–203. ISBN: 158113195X. DOI: 10.1145/350391.350432. URL: https://doi.org/10.1145/350391.350432.

[5] Lisa Crispin. "Driving Software Quality: How Test-Driven Development Impacts Software Quality". In: *IEEE Software* 23.6 (Nov. 2006), pp. 70–71. ISSN: 0740-7459. DOI: 10.1109/MS.2006.157. URL: http://ieeexplore.ieee.org/document/4012627/ (visited on 06/24/2022).

[6] Alan Gillies. "Modelling software quality in the commercial environment". In: *Software Quality Journal* 1.3 (Sept. 1992), pp. 175–191. ISSN: 0963-9314, 1573-1367. DOI: 10.1007/BF01720924. URL: http://link.springer.com/10.1007/BF01720924 (visited on 06/24/2022).

[7] ISO Central Secretary. "ISO/IEC/IEEE International Standard - Systems and software engineering – Vocabulary". In: *ISO/IEC/IEEE 24765:2010(E)* (2010). ISBN: 9780738162058, pp. 1–418. DOI: 10.1109/IEEESTD.2010.5733835. URL: http://ieeexplore.ieee.org/document/5733835/ (visited on 07/10/2023).

[8] ISO Central Secretary. *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models: ISO/IEC 25010:2011*. ISO. 2017. URL: https://www.iso.org/standard/35733.html (visited on 06/24/2022).

[9] B. Kitchenham and S. Charters. *Guidelines for performing systematic literature reviews in software engineering*. 2007.

[10] Microsoft. *Design Fundamentals*. 2010. URL: https://learn.microsoft.com/en-us/previous-versions/msp-n-p/ee658094(v=pandp.10) (visited on 2010).

[11] Sonia Montagud, Silvia Abrahão, and Emilio Insfran. "A systematic review of quality attributes and measures for software product lines". In: *Software Quality Journal* 20.3 (Sept. 2012), pp. 425–486. ISSN: 0963-9314, 1573-1367. DOI: 10.1007/s11219-011-9146-7. URL: http://link.springer.com/10.1007/s11219-011-9146-7 (visited on 06/24/2022).

[12] Nachiappan Nagappan et al. "Early estimation of software quality using in-process testing metrics: a controlled case study". In: *Proceedings of the third workshop on Software quality - 3-WoSQ*. the third workshop. St. Louis, Missouri: ACM Press, 2005, p. 1. ISBN: 978-1-59593-122-1. DOI: 10.1145/1083292.1083304. URL: http://dl.acm.org/citation.cfm?doid=1083292.1083304 (visited on 06/24/2022).

[13] H. Ogasawara, A. Yamada, and M. Kojo. "Experiences of software quality management using metrics through the life-cycle". In: *Proceedings of IEEE 18th International Conference on Software Engineering*. Berlin, Germany: IEEE Comput. Soc. Press, 1996, pp. 179–188. ISBN: 978-0-8186-7247-7. DOI: 10.1109/ICSE.1996.493414. URL: http://ieeexplore.ieee.org/document/493414/ (visited on 07/28/2023).

[14] Pablo Orviz et al. "A set of common software quality assurance baseline criteria for research projects". In: (2017). In collab. with Digital.CSIC and Digital.CSIC. DOI: 10.20350/DIGITALCSIC/12543. URL: https://digital.csic.es/handle/10261/160086 (visited on 02/10/2022).

[15] Pablo Orviz Fernández et al. "EOSC-synergy: A set of Common Service Quality Assurance Baseline Criteria for Research Projects". In: (June 15, 2020). In collab. with Digital.CSIC and Digital.CSIC.

DOI: 10.20350/DIGITALCSIC/12533. URL: https://digital.csic.es/handle/10261/214441 (visited on 02/10/2022).

[16] Eric Steven Raymond. *Software Release Practice HOWTO*. Jan. 14, 2013. URL: https://tldp.org/HOWTO/Software-Release-Practice-HOWTO/.

[17] John Shepherdson. "CESSDA Software Maturity Levels". In: (Mar. 29, 2019). DOI: 10.5281/zenodo.2614050. URL: https://zenodo.org/record/2614050 (visited on 02/10/2022).

[18] Kamonphop Srisopha and Reem Alfayez. "Software quality through the eyes of the end-user and static analysis tools: a study on Android OSS applications". In: *Proceedings of the 1st International Workshop on Software Qualities and Their Dependencies*. ICSE '18: 40th International Conference on Software Engineering. Gothenburg Sweden: ACM, May 28, 2018, pp. 1–4. ISBN: 978-1-4503-5737-1. DOI: 10.1145/3194095.3194096. URL: https://dl.acm.org/doi/10.1145/3194095.3194096 (visited on 06/24/2022).

[19] Wolfgang Zuser, Stefan Heil, and Thomas Grechenig. "Software quality development and assurance in RUP, MSF and XP: a comparative study". In: *Proceedings of the third workshop on Software quality - 3-WoSQ*. the third workshop. St. Louis, Missouri: ACM Press, 2005, p. 1. ISBN: 978-1-59593-122-1. DOI: 10.1145/1083292.1083300. URL: http://dl.acm.org/citation.cfm?doid=1083292.1083300 (visited on 06/24/2022).