Journal of Big Data

RESEARCH Open Access

DD-KARB: data-driven compliance to quality by rule based benchmarking

Mohammad Reza Besharati* and Mohammad Izadi

*Correspondence:
besharati@ce.sharif.edu

Sharif University of Technology, Azadi Avn., Tehran, Iran

Abstract

The problem of *compliance checking and assessment* is to ensure that the design or implementation of a system meets some desired properties and complies with some rules or regularities. This problem is a key issue in several human and engineering application domains such as organizational management and e-governance, software and IT industries, and software and systems quality engineering. To deal with this problem, some different approaches and methods have been proposed. In addition to the approaches such as formal methods, mathematical proofs, and logical evaluations, *benchmarking* can be used for compliance assessment. Naturally, a set of benchmarks can shape an applied solution to compliance assessment. In this paper we propose KARB solution system, i.e. keeping away compliance Anomalies through Rule-based Benchmarking. In fact, in our proposed method the rule-based benchmarking means evaluating the conformity of an under-compliance system to a set of rules. In this solution approach, the under-compliance system is specified symbolically (using formal and logical descriptions). Also, the desired rules are specified formally as the semantic logic in the evaluation process. After reviewing the proposed method, a case study was conducted to demonstrate and analyze the KARB solution. The IR-QUMA study (Iranian Survey on Quality in Messenger Apps) was then conducted to evaluate the quality of some messenger applications. According to the evaluation results, the hybrid DD-KARB method (with a combination of semantics-awareness and data-drivenness) is more effective than solo methods and can compute a good estimation for the messenger application user quality scores. Therefore, DD-KARB can be considered as a method for quality benchmarking in this technical context.

Keywords: Compliance checking of systems, Software quality, Semantic logic, Benchmarking, Messenger applications, Big data

Introduction

This paper copes with the problem of compliance checking and assessment (which we will define it carefully in the first subsection of this introduction). In the technical literature, there are some approaches to deal with this problem. Sharing a theoretical nature, many compliance assessment approaches are based on a systematic, rigorous, and formal theory of proof or evaluation (such as logics, formal languages, proof systems, reference models, and domain models). While the proposed theoretical approaches have high levels of internal integrity but usually they suffer from the lack of supporting tools

and the ability of adaptation to the diversity and complexity of real-world cases. In contrast, sometimes an aggregation of multiple simple tools is more successful than a single, rigid, unified, in-depth-designed, and sophisticated tool.

In some real-world compliance-solving cases, more lightweight approaches to formal specification that support semantic modeling (e.g. generative grammars, production rules, set-theoretic notations, and rewriting logics) can play central roles in overcoming semantic diversity and complexity. These approaches support a kind of semantic compilation of diverse, domain-specific semantic models. For instance, generative rules can be set to define the mapping and composition logic of different, independent semantic models.

In this paper, we propose KARB solution system, i.e. Keeping away compliance Anomalies through Rule-based Benchmarking. In fact, rule-based benchmarking means evaluating an under-compliance system with its symbolic specification and using a set of symbolic rules (on the behalf of the semantic logic of evaluation).

In the remaining part of this first section, we will introduce the primitive concepts of the problem domain and review related works. In the second section, we introduce the KARB solution. Then, in "Case study: software quality evaluation" section, a case study is presented for the application of this method for the issue of compliance to quality in the field of software engineering. In "DD-KARB" section, we introduce DD-KARB, which extends the KARB solution by a data-driven approach. In the last section, "Evaluation and discussion: IR-QUMA study" section, we have an evaluation and discussion based on IR-QUMA study using DD-KARB method.

Compliance checking

Compliance solutions concern assessment, evaluation, verification, validation, and checking of systems, services, processes, products, designs, organizations, or environments with regard to rules, regulations, laws, standards, specifications, policies, guidelines, protocols, methods, principals, and reference-models [1, 2]. The application domains that need and use compliance solutions including organizations and corporates in the following areas: software and IT industry [3], e-governance [2], finance and banking [4], legal sectors and professions [5], commerce and trade [6], highly regulated industries (e.g. food [7] and drug, medical services and devices [8], and construction industry [9]), complex and interdisciplinary products and services [10], emerging technology products and services (e.g. cyber-physical systems [11], self-driving cars [12], cognitive robotics and agents [13], and smart applications [14]).

There are many prominent compliance concerns that have been considered by numerous regulations, standards, laws, and acts. The most important concerns are as follows: security [15], safety [16–18], privacy [19, 20], data protection [21], accountability [22], responsibility [23], transparency [24], competency [25], anti-piracy [26], anti-corruption [27], antitrust [27], accessibility [28], HCI, quality management and assurance [29–31], environmental management [32], sustainability [33], usability [34], human comfort [35], ethics [36], conformance with the disabilities [37], adherence to the children [38], the elderly [39], simplicity [40], and ease of use.

Modern paradigms have amplified the necessity of compliance requirements (paradigms such as standardization in business, automation in industries, artificial

intelligence and ubiquitous computing in society, complex systems engineering, socio-technical systems, ongoing growth in the economy, social complexity, and quality maturity of services/processes).

In [41], a formal definition (as a 4-tuple) was presented for a special kind of benchmarking. There are a few formally-defined frameworks for compliance checking in legal applications, which are defined in theoretical manners such as formal systems [5] as well as conceptual modeling of legal texts [5]. Grammar-like and production-rule formalisms have been suggested for automated compliance checking in legal applications [42–45].

Rules and grammars for architecture conformance checking, especially for "software quality assurance" [46], is another application domain. Some rule-based approaches for architecture selection relate the nonfunctional requirements, domain requirements, and quality characteristics to architectural styles [47], architectural models [48], architectural patterns [49] and architectural aspects [50].

Circuits and flows are considered recurrent modeling approaches in systems engineering. Some researchers regard circuits and flows as a basis for compliance modeling, checking, and benchmarking [51, 52]. There are numerous verification tools and solutions for flow-based models. These tools serve as a means of compliance checking. For instance, agent-coordination protocols for crisis situations could be modeled and checked by these tools and solutions [53].

In software engineering, there are some model-based approaches to compliance assurance [54]. These approaches employ a modeling notation or framework (e.g. UML, KAOS [55], and GSN [56]). A "model-based assurance case" is an approach to safety compliance management. It encompasses a compliance meta-model covering "claims" or "requirements", "evidence", "arguments", and "contexts" [57, 58]. In Kokaly et al. [54], the need for a general model of compliance and compliance activity is addressed as an open-ended problem.

In Zhang and El-Gohary [9, 59], very close approaches were introduced. The meta-model and system architecture of these approaches are comparable with the one proposed in this paper. There are some other meta-models for compliance checking applications and frameworks (see [3, 60–62]).

Benchmarking

A benchmark is the *common or standard infrastructure* employed to *analyze, evaluate, and compare* the reality of solutions, tools, or systems by their *executions* (for a few definitions, see [63–71]. For some instances in other fields, see [30, 72–76]). Sometimes, a measure can simply be used as a benchmark [77]. *Procedures*, *measures*, and *computers* are considered the most common concepts in diverse definitions of benchmarking (Fig. 1).

For some early attempts in the history of benchmarking in IT and computing, see [78–82]. There is also a growing trend in benchmarking for quality assurance, management, and process improvement [83, 84]. It has been a progressive journey so far [85–87].

From a managerial standpoint, benchmarking requires a significant investment in time and perhaps money [84]. Hence, it should be considered a long-term profitable activity and a sort of infrastructure development for a field. In cloud computing, prior investments in performance measuring tools lead to the already available tools for the new

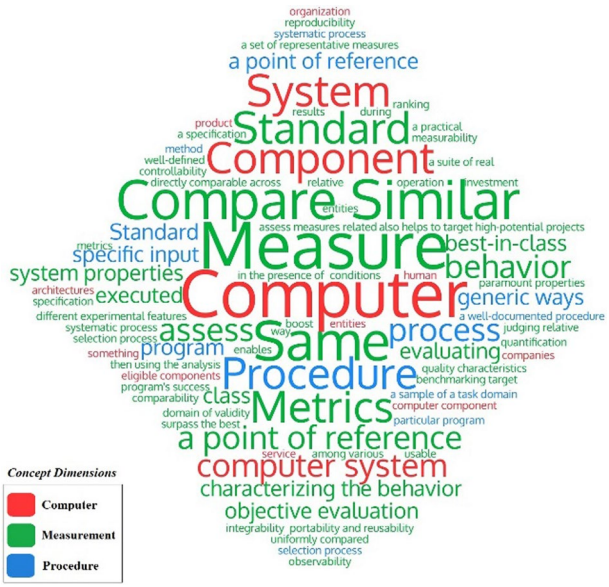


Fig. 1 A segmented word cloud showing the emphasis on different concepts in definitions of benchmarking

filed (for a case, see [88]). It was a chance; however, the dedicated attempts began for defining and developing benchmarks for cloud computing from scratch [89]. Moreover, investment in benchmarking is also important [90], for this decision can benefit all stakeholders [85].

Successful notions of benchmarking (in every field) are characterized by a community that creates, promotes, and uses benchmarks. Benchmarking can also be viewed as an applied manifestation and adoption of community knowledge and expertise [91].

As another instance of the application domains, there are informal guidelines for ensuring the quality of software (in terms of quality attributes such as security, integrity, and maintainability). Formal specification and automatic checking of these guidelines can contribute to the higher quality assurance of software (see [92] as an example for the formalization and automation of security guidelines).

Benchmarks can assess the quality (rather than only functionality) [29, 69, 71]; therefore, they are suitable for formal or systematic qualitative analysis of systems. For instance, security and compliance benchmarks have been reported [66, 93]. Measuring the productivity of an organization is another case that has qualitative dimensions (such as the level of customer satisfaction, the quality of products, or the extent to which an organization has the right group of staff [85]) which can be measured through some systematic approaches [85].

Soft benchmarks

A set of knowledge can be represented once using some tools and techniques, and then be used many times—ontologies are a practical case of this manner of reusability (see [94]). Thus, a community can construct a knowledge representation and use it as a standard and reusable asset. If this asset helps the community share their expertise, analyze their systems and solutions, and evaluate the behavior and other characteristics of

their systems, it can then be considered as a soft benchmark. The "soft" part of the title indicates its knowledge-related nature.

Knowledge representations are not limited to ontologies [95]. Formal specifications such as logical formulations, description logics, semantic networks, and rule-based approaches are considered alternatives [96]. Logical models have a share in compliance checking approaches. For instance, logical modeling of regulations is a method for rule representation and checking automation [97]. Rules can also be used as a paradigm for knowledge representation [98].

A logical theory for a piece of knowledge has the three essential characteristics of a benchmark. (1) It can be considered the *common infrastructure* due to the reusable and defined nature of a formal specification. (2) The results of reasoning indicate an *examination and evaluation* of the studied system and provide a basis for comparison between alternative and competing systems. (3) *Executing* reasoning on a logical theory of a piece of knowledge is an execution of meanings and semantics behind that knowledge. Thinking and mental activities can result in a hypothetical situation. In KARB, the rule-based reasoning schema can be viewed as semantic rules a mimic of these natural procedures (the simulation of human auditing by automated and intelligent compliance audit tools would result in a proper need for the compliance industry [2, 66, 99]).

Object models can act semantic models [100], especially for compliance checking purposes [101, 102]. For instance, Fornax objects capture specific rule semantics for the compliance checking of building designs [101]. These object models include contexts, domains, and sometimes system specifications [100]. As an example, Regarding Fornax, the objects for hospital design semantics differ from those for airport designs [101].

Software patterns are another representation form or media of technical knowledge. Pattern-based solutions to compliance checking have been addressed by some studies [103]. Compliance patterns are a kind of knowledge-capturing tool for compliance assessment.

In KARB, knowledge is represented through the intuitionistic formal semantics known as the "semantic logic". The semantic logic was created to capture the semantics and meanings of text [40]. It is used in KARB for knowledge representation. Any knowledge has its semantics and meaning [96, 104–106]. The knowledge itself is captured if its semantics and meanings are captured. Knowledge also has a specific structure [107]. Therefore, a meaning structure (or a semantic construct) could be a proper candidate for the manifestation of knowledge. Based on and adopted from [40], semantics and meanings are considered as constructions, namely lattices,[1] or systems of realities (= intuitions). Any well-defined formulation of knowledge represents and refers to a combination of entities, things, objects, events, affairs, facts, physics, concepts, cognitions, affections, or any other sorts of basic realities and intuitions. Therefore, knowledge can aggregately and abstractly be considered a combination and construction of basic realities and intuitions (with a glue of operators such as logic, structures, modalities, and any necessary ones). This manner of semantic definition is a constructive and intuitionistic one (Fig. 2).

[1] Lattices are mathematical models for constructs in intuitionistic semantics.

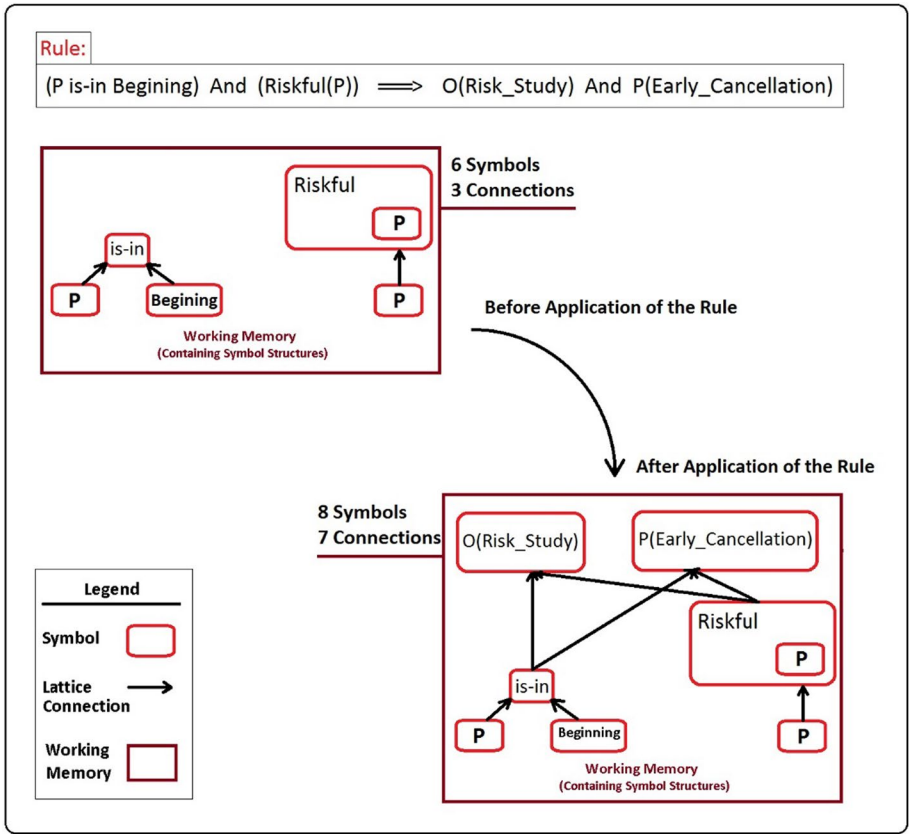


Fig. 2 An instance of a semantic logic rule and effects of its application on working memory. The rule specifies the existence of Risk Study obligation and Early Cancellation probability when a project occurs in a riskful beginning [40]

As an underlying philosophy in KARB, the reality and its meanings are composed of statics-related and dynamics-related meanings. Symbolic constructs capture the statics-related part of knowledge meanings, whereas the generative rules capture the dynamics-related part of knowledge meanings.

The KARB solution

In this section, we introduced our proposed solution. The results of a rule-based benchmarking in KARB differ from those of other compliance assessment approaches. Other approaches yield the results in the form of "yes or no", "correct or incorrect", etc.; however, a pool of quantities (i.e. derived and generated symbols) is considered as the results in KARB. Therefore, the overall state of working memory at the end of each benchmarking process indicates the evaluations of the studied compliance case, whereas a rigorous and reasoned evaluation with diverse dimensions and values would be achieved.

Every aspect of compliance concerns can be addressed with a separate rule-based benchmark. Every benchmark draws and adds a new simple line on the overall picture of compliance assessment scenes. A set of multiple, different, and diverse benchmarks can make an applied and realistic compliance assessment of a complex system. Relying on

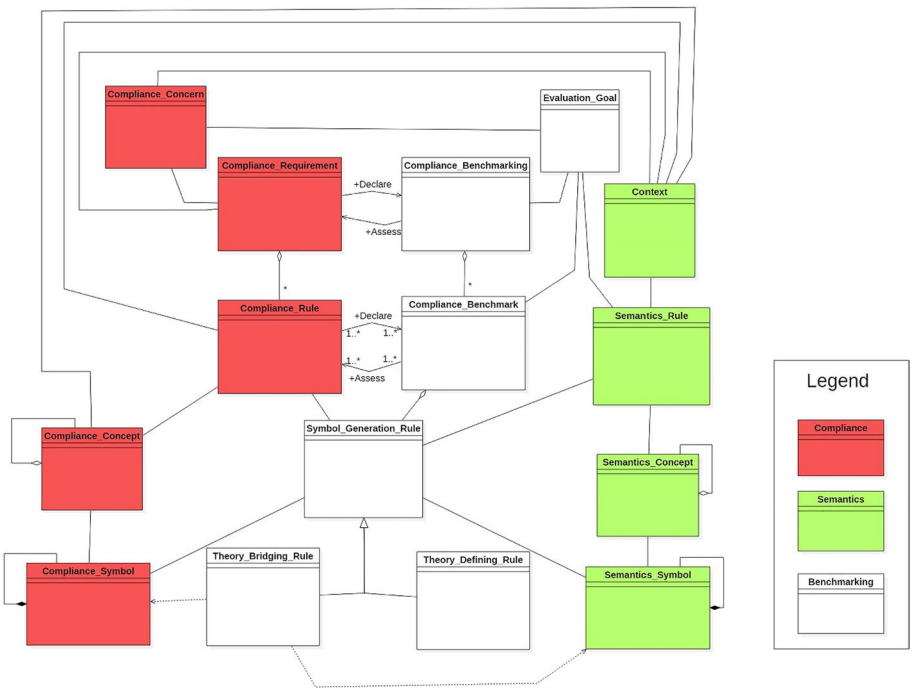


Fig. 3 The meta-model of the KARB solution is depicted as a class diagram (UML Class Diagram is a popular tool for concept modeling and meta-modeling) of elements. There are three main dimensions for KARB elements: compliance-related (red), benchmarking-related (white), and semantics-related (green) concepts

rule-based benchmarks, this experimental and applied approach to compliance assessment provides a new space for new sorts of innovative, creative, and diverse methods for compliance assessment.

Figure 3 illustrates a brief meta-model of KARB. The assessment of every compliance requirement is reified by a compliance benchmark, which in turn consists of some concrete rule-based benchmarks. Therefore, every compliance requirement declares meanings and semantics for a compliance benchmark that assesses it.

A *compliance symbol (CSYM)* abstracts a *compliance concept (CCON)* in a similar sense of atom symbols in LISP, objects in object-oriented languages (e.g., Java), and JSON fragments in NoSQL DBs, all of which are units for compositional parts. Every *compliance concern (CC)* is in association with some *compliance requirements*[2] *(CR)* which capture the notion and attitude of that concern. For instance, *safety* is a compliance concern which can be defined in a zoo as the following notion: *the zoo animals must not be able to harm or threat the visitors* (see Example 1).

Some *compliance rules (CRUL)* aggregately define the operational realization of a *compliance requirement*. Every CRUL defines a more rigorous, concrete, and special obligation than a CR. In KARB, the rules are considered to be finer than requirements. The overall shape of a requirement consists of the limiting lines of its constituting parts (=rules). Every CRUL has some CCONs in its definition. In the computational mechanisms of KARB, a CSYM abstracts a CCON. By using a glue of (logical, structural,

[2] A compliance requirement is a known concept in the compliance checking literature and frameworks.

modality, and any necessary) operators, a formal definition of a CRUL can be constructed from the CSYMs of its CCONs (see Example 1).

Example 1

The system under compliance	A zoo
CC1	Safety
CR1	The zoo animals must not be able to harm or threat the visitors
CRUL1	The cage fences must have proper specifications and conditions
CCONs	Cage, fence, proper specifications, proper conditions
CSYMs	CE, FE, PS, PC
Formal specifications of CRUL1	X [IS-A] FE(CE)⇒O[PS(X)] [AND] O[PC(X)]

In KARB, the manner of formal specification of a CRUL is based on the intuitionistic logic called the "semantic logic". Technically, it can be viewed as an axiomatic system on symbols with Brouwer–Heyting–Kolmogorov interpretations for semantics [108]. Symbols are considered to be on the behalf of basic intuitions (concepts, entities, objects, things, events, values, quantities, qualities, etc.), whereas and the studied system is viewed as a complex construction of basic intuitions.

Formally, it would be sufficient to consider the semantic logic consisting of (1) a set of symbols (on the behalf of basic intuitions) and (2) a set of rules on them. Every rule describes a symbol generation action. When its left-side symbolic structure is ready in the working memory, the right-side symbolic structure is generated and pushed to the lattice of symbols in the working memory (see Fig. 2).

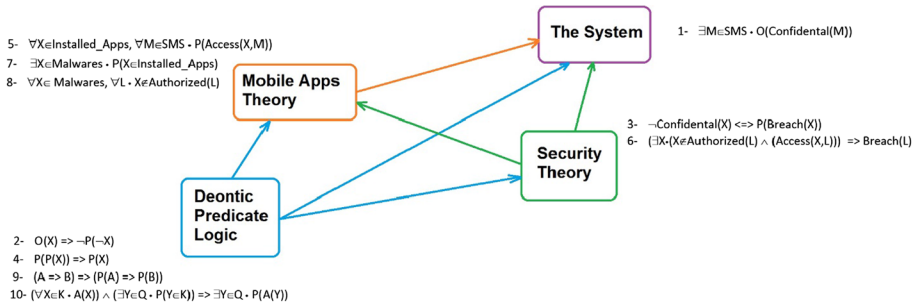
Case study: software quality evaluation

Since the 1990s, there have been various approaches to defining measurable quality such as quality function deployment, goal question metrics, and software quality metrics [109]. These methods seek to shape a general and common framework for quality measurement concerns. However, some quality factors are contextual and user-dependent [110]. For instance, some studies have measured the quality attributes of messenger apps and services from the user perspective (see [111–114]) or based on user behavior (see [115–117]).

Quality definitions can be seen as a hierarchical formal system of interrelated concepts [118] or attributes [119, 120]. This view helps create an explicitly defined conceptual construct for qualities, i.e. a concept-quantized definition of qualities. Hence, a quantification of qualities (which is a well-known but poorly-achieved goal for rigorous software engineering [121]) helps measure and perceive the true level of qualities in each application.

After definition, it is the time for operationalization. Every theoretical concept has its real instances on the ground. User feedback, comments, experiences, requests, requirements, desires, cognitions, and intuitions can help this grounding operationalization. Therefore, a good quality-definition theory needs a good quality-grounding theory.

There is a semantic gap between definition theories and grounding theories. The former has a neat nature, whereas the latter has a scruffy one. How is it possible to



1- ∃M∈SMS • O(Confidental(M))
2- O(X) => ¬P(¬X)
3- ¬Confidental(X) <=> P(Breach(X))
4- P(P(X)) => P(X)
5- ∀X∈Installed_Apps, ∀M∈SMS • P(Access(X,M))
6- (∃X•(X∉Authorized(L) ∧ (Access(X,L))) => Breach(L)
7- ∃X∈Malwares • P(X∈Installed_Apps)
8- ∀X∈ Malwares, ∀L • X∉Authorized(L)
9- (A => B) => (P(A) => P(B))
10- (∀X∈K • A(X)) ∧ (∃Y∈Q • P(Y∈K)) => ∃Y∈Q • P(A(Y))

Fig. 4 The involving semantic theories and the semantic logic for Example 2

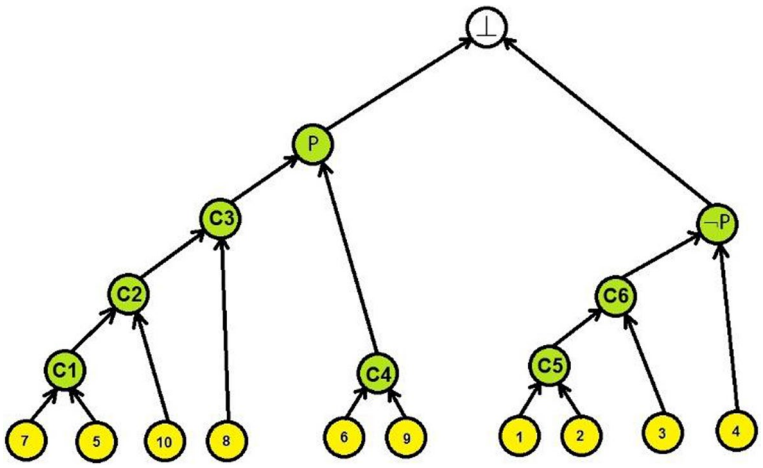


Fig. 5 The proof construct lattice for false value from the scenario semantics

bridge the neat nature and the scruffy nature [96]? A glue model can come forward to resolve this challenge. This model must contain the main conceptual elements of both sides and try to relate them in a gradient conceptual spectrum. Since this is exactly the manner of the KARB solution, it can be used as a method of designing a software quality evaluation technique. It is also a kind of evaluation for KARB, for it demonstrates its usefulness for a real concern or problem in the software engineering community.

Example 2 The semantic logic is provided to model the semantics of this scenario.

Using SMS-Based Dynamic Passwords for E-Banking Transactions This logic contains certain rules and intuitions from four context theories, i.e. mobile apps, deontic

predicate logic, security and system (see Fig. 4). The stateless model-checking of this scenario semantics (by symbolic-value generations) yields a "false" value; hence, there is a contradiction in the scenario. Figure 5 presents the explainable results in the proof construction lattice (= Why and how the overall result was obtained?). Although there are some unmentioned reasoning operations, they are omitted for the sake of simplicity in this preliminary example.

The KARB manifestation for this compliance scenario:

The system under compliance	**Using SMS-based dynamic passwords for e-banking transactions**
CC	User data security
CR	Hackers cannot gain access to users' banking information during SMS exchange
CRUL	It is not possible to breach the bank data of users
CCONs	Breach, data
CSYMs	BR, DT
Formal specifications of CRUL	NOT(BR(DT))

DD-KARB

In order to boost the model pragmatics, a methodic extension of the principal model of KARB is considered. The Data-Driven KARB (DD-KARB) incorporates data-calculated weights (based on Big Data gathered from people) and values to parameterize the rules. An example of a parameterized rule is presented below:

Example 3

Rule:

$$\text{alpha} * (A \Rightarrow B) \Rightarrow \text{beta} * (P(A) \Rightarrow P(B))$$

Interpretation:

If the alpha instances of $(A \Rightarrow B)$ are generated in the semantic solution to the system (or if the weight of $(A \Rightarrow B)$ is equal to alpha), then the beta instances of $(P(A) \Rightarrow P(B))$ must be generated by applying this rule.

Based on expert scores, data examinations, data schemas, AI pre-trained models, and other sources of data-driven models, the semantics-based KARB models can be parameterized, annotated, and enriched with data-driven aspects. The full-fledged methods (by combining the data and semantics aspects) could be better than the solo methods. Each system context or domain of application has its own semantics and data. DD-KARB can be employed to record and adapt both data and semantics aspects. This manner of description or declaration can help define a hybrid semantic core for compliance checking and solving. A hybrid semantics can help approach some hard-to-check compliance requirements through automatic

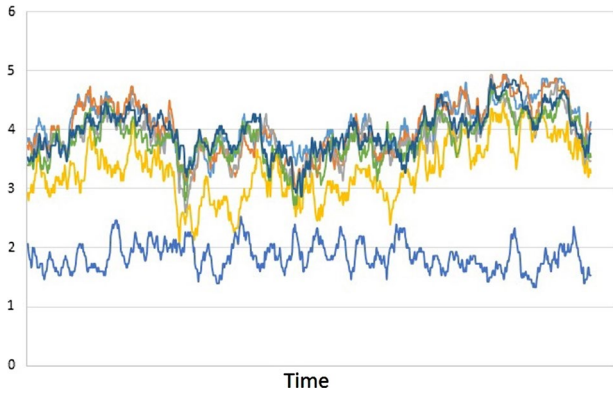


Fig. 6 The running-average series of responses to 7 different questions in the questionnaire

compliance-checking solutions. The goal and related dataset for evaluation are discussed in the next section.

Evaluation and discussion: IR-QUMA study

The case study is a popular evaluation method in software engineering research. Case studies are frequently used in papers to demonstrate the capabilities of new techniques and methods [122]. A case study was conducted in order to demonstrate and analyze the manner of KARB solution. The IR-QUMA study (Iranian Survey on Quality in Messenger Apps) was defined to evaluate the quality of some messenger applications. It consists of these stages:

1. **Selecting messenger applications** The selected applications were Telegram, WhatsApp, Eita, Soroush, Bale, and some other popular mobile messengers in the Iranian cyberspace. They were selected for the IR-QUMA case study due to the access to a large community of their users.
2. **Collecting data** An online questionnaire was designed to collect the opinions of users and trace the specifications of user experiences. The seven main questions concerned "absolute quality", "relative quality", "user satisfaction", "error-freeness", "perceived UI complexity", "rationality of routines", and "accordance and usability". The answer to each question ranged between 1 and 5 to represens choices from "very weak" to "excellent". Figure 6 shows the running-average series of user responses (for a portion of dataset).
3. **Using the KARB solution**

    a. **Elicitation of involving semantic theories**
    b. **Specification of involving semantic theories** The KARB-based specifications were developed for each of the involving semantic theories. Figure 7 presents a detailed map of the involving semantic theories. The emphasis was given to these theories:

        i. KARB-based specification of messenger apps
        ii. KARB-based specification of some quality terms

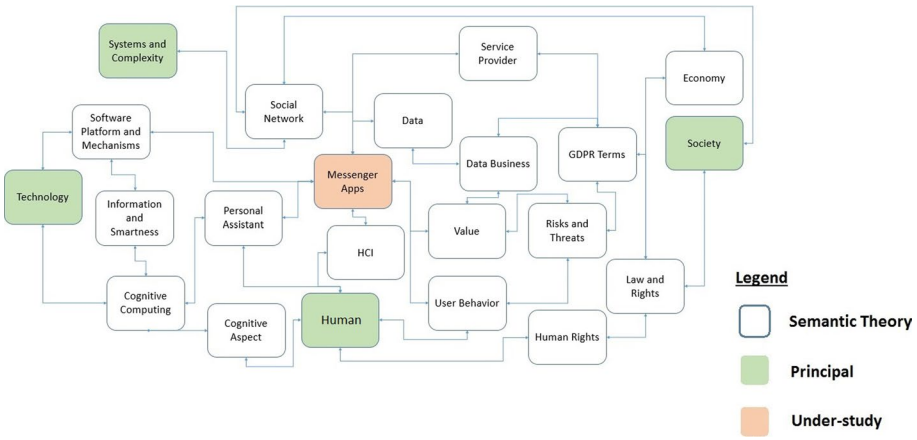


Fig. 7 A detailed map of involving semantic theories in the IR-QUMA study

iii. KARB-based specification of user behavior
iv. KARB-based specification of some pieces of HCI knowledge
v. KARB-based specification of risks and threats
vi. KARB-based specification of software platform and mechanisms
vii. KARB-based specification of cognitive aspects
viii. KARB-based specification of social aspects

c. **Computation and model checking** The KARB solution was employed to compute some of the compliance anomalies.

4. **Evaluating the results** The results were compared in three aspects: expert judgments, IT reports, and user opinions.

The IR-QUMA study details will be published in a separate report. However, this paper we used the collected data and the semantic model to conduct some experiments on deferent quality benchmarks, especially the KARB solution.

IR-QUMA data collection

A questionnaire was designed to evaluate some quality-related measures, metrics, and features from the user experience perspective. The questionnaire was published in the popular channels of Iranian mobile social networks on 10 different messengers (i.e. Telegram, WhatsApp, Instagram, Eita, Soroush, Bale, Gap, iGap, Shaad, and Rubika). More than 40 communities of users on these 10 messengers (which are shaping more than 350 micro-communities based on visiting hours and spatial partitions) contributed to this research questionnaire. The collected data exceeded 7k completed online forms (from more than 7k distinct participants). In the research dataset [123], for the sake of data privacy and protection reasons, the names of these messengers were hashed randomly by assigning the ID codes from M1 to M10.

Different sets of statistical analysis, time series analysis, frequency analysis, cluster analysis, classification analysis, geometry locus of data points, and topological data

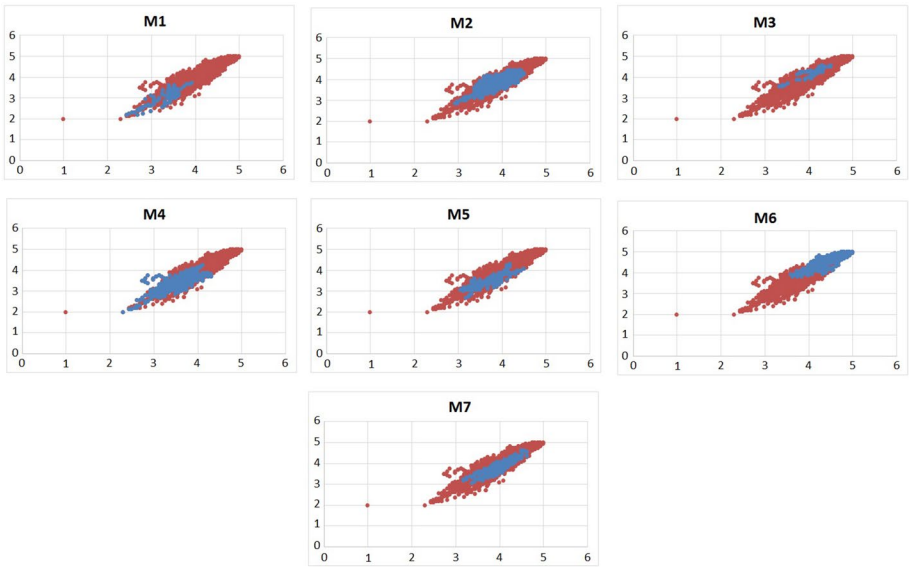


Fig. 8 The correlation between "absolute quality" and "relative quality" from the user point of view for some messenger applications based on the IR-QUMA data. Every point represents average values of one data segment

analysis were based on user opinion data to obtain useful insights. As an analysis example, the data were sorted in a temporal order (which conserved the segregation of micro-communities). A running average method was then adopted (with a window-size = 20). Therefore, 7k data points were obtained from different segments of those 350 micro-communities. Every micro-community with its segment-average had its own footprint in the total space of data-points. Moreover, every messenger app had its own footprint in the total space of data-points.

Figure 8 depicts a correlation locus analysis for two of quality measures for seven different messenger applications in 7k data segments of 350 micro-communities. Every point is in accordance with the measure values obtained from one segment of a micro-community. The blue points refer to the mentioned messengers, whereas the red points indicate the entire data space (for all studied messengers). Every axis demonstrates a 5-level measure value (obtained by averaging user opinions in one data segment).

Figure 9 demonstrates the analysis for two other measures, i.e. correctness vs. quality. Correctness means the error freeness and bug freeness of the messenger application. The results indicate that there is a buffer between "correctness increase/decrease" and "overall quality increase/decrease". This means that the other factors (rather than correctness) can play a key role in the overall quality of software.

Figure 10 illustrates the histogram of score instances of user quality judgments for 10 applications. The topological analysis of these 10 curves indicates six different curve clusters based on change trends during five levels.

Evaluation of the method

The KARB solution is based on the semantic logic specified in previous research steps and its performance to correctly calculate (or mimic) the user opinions (evaluated in terms of error percentage of benchmark-computed quality scores and user opinions

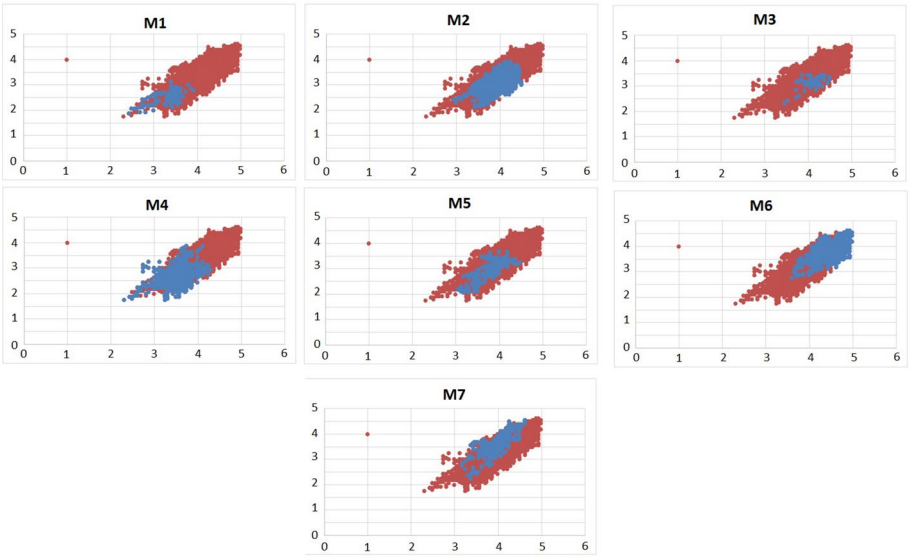


Fig. 9 The correlation between "correctness" and "absolute quality" from the user point of view for some messenger applications based on the IR-QUMA Data. Every point represents average values of one data segment

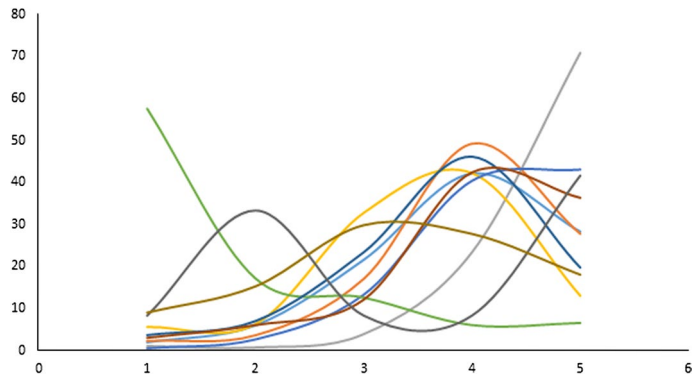


Fig. 10 The histogram analysis of quality score levels (obtained from user judgments) about "absolute quality" for 10 different messenger applications. Every messenger application has five data points for the numeration of 1—very poor, 2—poor, 3—moderate, 4—good and 5—excellent quality scores. These scores reflect the user experience point of view to the quality of messengers

about quality scores). The following experimentation setting was considered: four different methods for quality benchmarking and five different experiments (for five different messengers).

Every user opinion record involves two sections: (1) the user opinion about quality score, which was called the absolute quality score, and (2) the quality context. The quality context includes the factors that can affect or relate to user opinion about quality scores. Age, gender, and other data were gathered from the users stating their experiences with the messengers (the scores also included bug-freeness and error-freeness, perceived UI complexity, rationality of routines, score of usability, etc.). The value options for all scores in the questionnaires were defined in a 5-point Likert scale [124] (the Likert scale

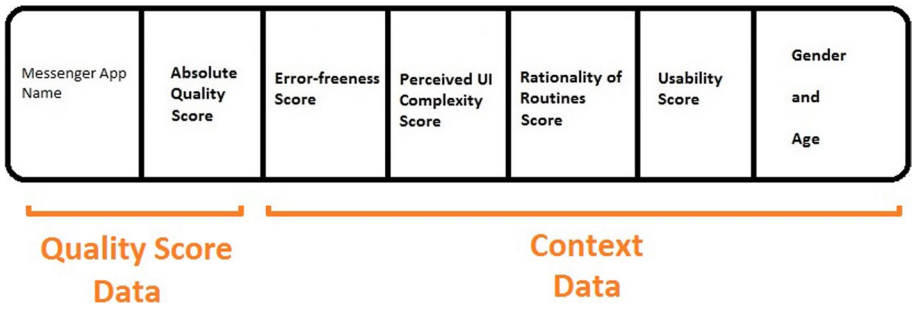


Fig. 11 The structure of a user opinion record

Table 1 Evaluation results (for Experimentation-Plan-ID-1)

Experimentation-plan-ID = 1			Error percentage					
			Exp. 1 Telegram	Exp. 2 Eita	Exp. 1 Whatsapp	Exp. 1 Soroush	Exp. 1 Bale	Average
Type of benchmark	Quantitative, based on stats	IR-QUMA (simple average)	18.4	24.3	26.3	31.8	30.1	26.2
	Quantitative, based on sense	Expert quality scoring	17.5	23.1	25.4	30.2	29.8	25.2
	Declarative, based on analysis	KARB + hill climbing	11.8	15.6	15.8	17.9	16.4	15.5
	Hybrid, based on solving	Data-driven KARB	8.8	13.2	13.5	16.8	14.0	13.3

has been used in various domains of software engineering such as [125]). Figure 11 shows the structure of a user opinion record.

Definition 1 $N_i = number\ of\ user\ opinion\ records\ about\ App_i$

Definition 2

$$Error\_Percentage(App_i,\ Method_j) = \frac{\sum_{k=1}^{N_i} \left| Benchmark\_Computed\_Quality(App_i,\ Method_j,\ Context_k) - User\_Quality\_Opinion_k \right|}{N_i}$$

The KARB manifestation for this compliance situation:

The system under compliance	Messenger apps
CC	Quality form users' viewpoint
CR	Messenger apps must be in the proper quality level for their users
CRUL	Messenger apps must get the proper grade-points from the user-quality-viewpoint benchmark
CCONs	App, Messenger, Proper user-quality-grade-point
CSYMs	APP, MSR, PUG
Formal specifications of CRUL	X [IS-A] APP(MSR)⇒O[PUG(X)]

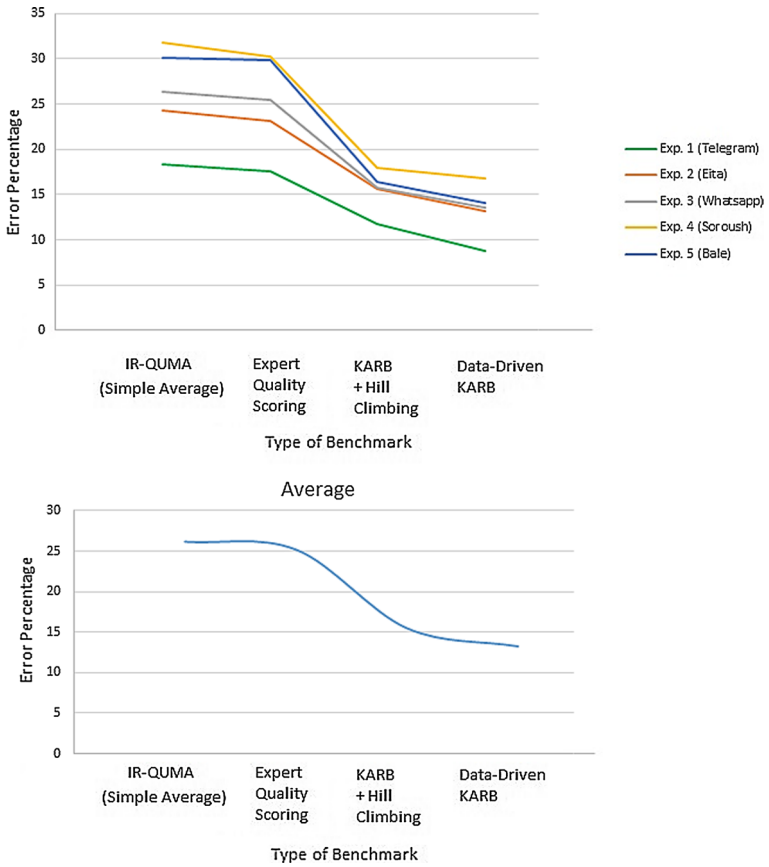


Fig. 12 Error reduction curves for five different experiments and their average

Table 1 reports the evaluation results (in the above-mentioned experimentation setting). Accordingly, the data-driven KARB method reduced the error percentage significantly. Figure 12 shows the error reduction curves (for five experiments). The average curve for these five curves indicate a pseudo-Sigmoid form. In other words, the hybrid DD-KARB method (with combination of semantics-awareness and data-drivenness) is more effective than solo methods and can compute a good estimation for messenger application user quality scores. Therefore, DD-KARB can be considered a method for quality benchmarking in this technical context.

Discussion and conclusion

The first benchmark uses the simple average of the IR-QUMA context data (Fig. 11) as an estimator for quality scores, whereas the second benchmark is based on expert judgments about the quality of messengers. Moreover, the third benchmark is based on an initially voided-filled DD-KARB rules that obtain their weight values from a hill-climbing optimization algorithm for reaching a local minimum of error. These rules compute an estimate for the user-quality-grade-point of the messenger app, based on context data of each opinion record (i.e. an estimate for absolute-quality-score from the other

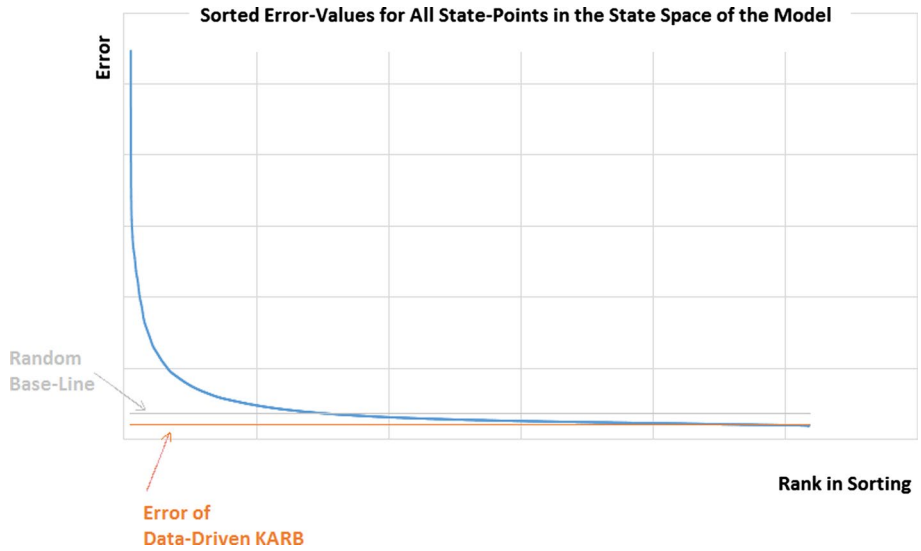


Fig. 13 The optimization performance of the fitting-solving-procedure in the execution of the DD-KARB benchmark in Experiment-1 (Telegram). Every point represents the error of one state in the total state space. The fitting-solving-procedure outperformed the random baseline and found a state near the exhaustive minima

fields of Fig. 11. For further details please review the associated java code in Additional file 1). The last benchmark, but the first in precision, is based on the DD-KARB rules that obtain their weight values from two sources: (1) IR-QUMA data values and (2) a lightweight state-space-checking procedure for finding good fitting parameters for the IR-QUMA dataset and the DD-KARB ruleset. Figure 13 depicts some results of this fitting procedure.

Therefore, the last benchmark incorporates these method features in a hybrid manner including semantics-awareness (by KARB), data-drivenness (by the DD part of DD-KARB and IR-QUMA data), and the fitting solution (by finding fitting parameters with a lightweight state-space-checking procedure).

The intended semantic landscape of this problem (i.e. quality measurement of messenger apps) involves more than 10 semantic theories. Without the semantic framework which the KARB solution provides, it would be impossible to focus on the most relevant parts of the wide semantic landscape of this complex problem. Without using the KARB rules which act as a kind of declarative dimensions in this problem, data-driven solving procedures and verification and model-checking methods would be unable to escape from the "state space explosion" [126] in this landscape.

However, the procedures and methods escaped from the "state space explosion" with the help of KARB in DD-KARB. A 3-min process on a conventional PC (Windows + Java + Intel Core i7 Processor) was successfully able to solve a good fitting 10K-order complexity space (IR-QUMA) to a 10G-order value-space of weight values in the DD-KARB rules of the experiment.

It is concluded that the hybrid nature of the DD-KARB method (in the KARB Solution) can help solve some complex compliance problems in a lightweight manner and yield good results (in terms of a low-error compliance-level quality estimator).

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s40537-022-00654-8.

Additional file 1: DD-KARB Case-Study Java Code.

Acknowledgements
Thanks to more than 7000 IR-QUMA survey participants.

Author contributions
MRB: conceptualization, research, survey, meta-modeling, writing, reviewing and editing, data gathering, programming, visualization. MI: conceptualization, writing, reviewing and editing, supervision. Both authors read and approved the final manuscript. Both authors have the equivalent role of Corresponding Author, in terms of academic regulations.

Funding
Not applicable.

Availability of data and materials
See Besharati and Izadi [123].

Declarations

Ethics approval and consent to participate
Consents to participate were obtained.

Consent for publication
Not applicable.

Competing interests
There is no competing interests.

Received: 25 December 2021 Accepted: 20 October 2022
Published online: 01 November 2022

References

1. Besharati MR, Izadi M. Semantics based compliance solving. In: Fundaments of software engineering (FSEN 2019), Student Poster Competition; 2019.
2. Hashmi M, Governatori G, Lam HP, Wynn MT. Are we done with business process compliance: state of the art and challenges ahead. Knowl Inf Syst. 2018;57:79–133.
3. Turetken O, Elgammal A, Van Den Heuvel WJ, Papazoglou MP. Capturing compliance requirements: a pattern-based approach. IEEE Softw. 2012;29:28–36. https://doi.org/10.1109/MS.2012.45.
4. Brandt C, Santini F, Kokash N, Arbab F. Modeling and simulation of selected operational IT risks in the banking sector. In: ESM 2012–2012 European simulation and modelling conference; 2012. p. 192–200.
5. Ingolfo S, Siena A, Susi A, Perini A, Mylopoulos J. Modeling laws with nomos 2. In: 2013 6th international workshop on requirements engineering and law (RELAW) 2013. IEEE; 2013. p. 69–71. https://doi.org/10.1109/RELAW.2013.6671350.
6. Governatori G, Milosevic Z, Sadiq S. Compliance checking between business processes and business contracts. In: Proceedings of the 2006 10th IEEE international enterprise distributed object computing conference (EDOC'06) 2006; 2006. p. 221–32. https://doi.org/10.1109/EDOC.2006.22.
7. Zasada A, Fellmann M. A pattern-based approach to transform natural text from laws into compliance controls in the food industry. LWA. 2015;15:230–8.
8. Almpani S, Stefaneas P, Boley H, Mitsikas T, Frangos P. A rule-based model for compliance of medical devices applied to the European market. Int J Extrem Autom Connect Healthc. 2019;1:56–78.
9. Zhang J, El-Gohary NM. Semantic-based logic representation and reasoning for automated regulatory compliance checking. J Comput Civ Eng. 2017;31:04016037. https://doi.org/10.1061/(asce)cp.1943-5487.0000583.
10. Bragaglia S. Monitoring complex processes to verify system conformance: a declarative rule-based framework; 2013.
11. Vuotto S. Requirements-driven design of cyber-physical systems. In: CEUR workshop Proc.; 2018.
12. Pek C, Rusinov V, Manzinger S, Üste MC, Althoff M. CommonRoad drivability checker: simplifying the development and validation of motion planning algorithms. In: 2020 IEEE intelligent vehicles symposium (IV); 2020. p. 1013–20.
13. Akinkunmi BO, Babalola FM. A norm enforcement mechanism for a time-constrained conditional normative framework. Auton Agent Multi Agent Syst. 2020;34:1–54. https://doi.org/10.1007/s10458-020-09441-2.

14. Miandashti FJ, Izadi M, Shirehjini AAN, Shirmohammadi S. An empirical approach to modeling user-system interaction conflicts in smart homes. IEEE Trans Hum Mach Syst. 2020;50:573–83.
15. Ranise S, Siswantoro H. Automated legal compliance checking by security policy analysis. In: Lecture notes in computer science (including subseries Lecture notes in artificial intelligence and lecture notes in bioinformatics); 2017. p. 361–372. https://doi.org/10.1007/978-3-319-66284-8_30.
16. Elise GCC, Kift RL. Keeping track of railway safety and the mechanisms for risk. Saf Sci. 2018;110:195–205.
17. Castellanos Ardila JP. Facilitating automated compliance checking of processes against safety standards. Doctoral dissertation, Mälardalen University; 2019.
18. Kupferman O, Vardi MY. Model checking of safety properties. Form Methods Syst Des. 2001;19:291–314.
19. Truong NB, Sun K, Lee GM, Guo Y. GDPR-compliant personal data management: a blockchain-based solution. IEEE Trans Inf Forensics Secur. 2020;15:1746–61. https://doi.org/10.1109/TIFS.2019.2948287.
20. OCR. Summary of the HIPAA privacy rule: HIPAA compliance assistance. Office for Civil Rights; 2003.
21. Lynskey O. The foundations of EU data protection law. Oxford: Oxford University Press; 2015.
22. Butin D, Chicote M, Le Métayer D. Log design for accountability. In: Proceedings of the 2013 IEEE security and privacy workshops, SPW 2013. IEEE; 2013. p. 1–7. https://doi.org/10.1109/SPW.2013.26.
23. Bukhsh FA, Queiroz PGG. Conceptual modeling for corporate social responsibility: a systematic literature review. In: 16th international conference on the economics of grids, clouds, systems, and services; 2019. p. 218.
24. Samavi R, Consens MP. Publishing privacy logs to facilitate transparency and accountability. J Web Semant. 2018;50:1–20.
25. Gay S, Badrick T, Ross J. "State of the art" for competency assessment in Australian medical laboratories. Accredit Qual Assur. 2020;25:323–7.
26. Kesan JP, Gruner RS. Intellectual property compliance: systematic methods for building and using intellectual property. In: The Cambridge handbook of compliance. Cambridge: Cambridge University Press; 2020. https://doi.org/10.2139/ssrn.3506951.
27. Abrantes-Metz RM, Prewitt E. Antitrust compliance 2.0: the use of structural analysis and empirical screens to detect collusion and corruption in bidding procurement processes. Antitrust Chron Compet Policy Int. 2015. https://ssrn.com/abstract=3291651.
28. Baule SM. Evaluating the accessibility of special education cooperative websites for individuals with disabilities. TechTrends. 2020;64:50–6.
29. Correia JP, Visser J. Benchmarking technical quality of software products. In: Proceedings of the 2008 15th working conference on reverse engineering, WCRE'08. IEEE; 2008. p. 297–300. https://doi.org/10.1109/WCRE.2008.16.
30. Baggen R, Correia JP, Schill K, Visser J. Standardized code quality benchmarking for improving software maintainability. Softw Qual J. 2012;20:287–307. https://doi.org/10.1007/s11219-011-9144-9.
31. Lenarduzzi V, Lomio F, Moreschini S, Taibi D, Tamburri DA. Software quality for AI: where we are now? In: International conference on software quality; 2021. p. 43–53.
32. De Craemer S, Vercauteren J, Fierens F, Lefebvre W, Meysman FJR. Using large-scale NO_2 data from citizen science for air-quality compliance and policy support. Environ Sci Technol. 2020;54:11070–8. https://doi.org/10.1021/acs.est.0c02436.
33. Schreiber C. Automated sustainability compliance checking using process mining and formal logic. In: Proceedings of the 7th international conference on ICT for sustainability; 2020. p. 181–4.
34. Alonso-Virgós L, Espada JP, Martínez OS, Crespo RG. Compliance and application tests of usability guidelines about giving information quickly and comprehensibly. Complex Intell Syst. 2020;7:1–21.
35. Zhu M, Wang Y, Pu Z, Hu J, Wang X, Ke R. Safe, efficient, and comfortable velocity control based on reinforcement learning for autonomous driving. Transp Res Part C Emerg Technol. 2020;117: 102662.
36. Christina A, Fort TL. Finding the fit: why compliance and ethics programs should seek to match individual and corporate values. Bus Horiz. 2020;63:451–62.
37. Gavine A, Spillers F. Toward a disability-centric model of user participation in accessibility efforts: lessons from a case study of school children. In: International conference on human–computer interaction; 2020. p. 76–86.
38. Tahir R, Arif F. A measurement model based on usability metrics for mobile learning user interface for children. Int J E-Learn Educ Technol Digit Media. 2015;1:16–31.
39. Bajenaru L, Marinescu IA, Dobre C, Prada GI, Constantinou CS. Towards the development of a personalized healthcare solution for elderly: from user needs to system specifications. In: Proceedings of the 2020 12th international conference on electronics, computers and artificial intelligence (ECAI). IEEE; 2020. p. 1–6. https://doi.org/10.1109/ECAI50035.2020.9223254.
40. Besharati M, Izadi M. Deciding About semantic complexity of text by DAST model. arXiv:1908.09080. 2019.
41. Jacobvitz AN, Hilton AD, Sorin DJ. Multi-program benchmark definition. In: 2015 IEEE international symposium on performance analysis of systems and software (ISPASS); 2015. p. 72–82. https://doi.org/10.1109/ISPASS.2015.7095786.
42. Prasad TVVV, Korrapati RB. Computerized applications of natural language processing in digital economy: a review. Int J Eng Manag Res. 2017;7:239–41.
43. Nash E, Wiebensohn J, Nikkilä R, Vatsanidou A, Fountas S, Bill R. Towards automated compliance checking based on a formal representation of agricultural production standards. Comput Electron Agric. 2011;78:28–37. https://doi.org/10.1016/j.compag.2011.05.009.
44. Maxwell JC, Antón AI. Developing production rule models to aid in acquiring requirements from legal texts. In: Proceedings of the 2009 17th IEEE international requirements engineering conference; 2009. p. 101–10. https://doi.org/10.1109/RE.2009.21.
45. Ingolfo S. Nomos 3: legal compliance of software requirements. Trento: University of Trento; 2015.
46. De Silva LR. Towards controlling software architecture erosion through runtime conformance monitoring. St Andrews: University of St Andrews; 2014.
47. Moaven S, Ahmadi H, Habibi J, Kamandi A. Decision support system environment for software architecture style selection (DESAS v1. 0). In: SEKE09; 2009. p. 147–51.

48. Nawaz F, Mohsin A, Fatima S, Janjua NK. Rule-based multi-criteria framework for SaaS application architecture selection. In: IFIP international conference on artificial intelligence in theory and practice; 2015. p. 129–38. https://doi.org/10.1007/978-3-319-25261-2_12.
49. Alebrahim A, Fassbender S, Filipczyk M, Goedicke M, Heisel M. Towards systematic selection of architectural patterns with respect to quality requirements. In: The ACM international conference proceeding series; 2015. p. 1–20. https://doi.org/10.1145/2855321.2855362.
50. del Mar Roldán-García M, García-Nieto J, Maté A, Trujillo J, Aldana-Montes JF. Ontology-driven approach for KPI meta-modelling, selection and reasoning. Int J Inf Manag. 2019;58: 102018. https://doi.org/10.1016/j.ijinfomgt.2019.10.003.
51. Preidel C, Borrmann A. Refinement of the visual code checking language for an automated checking of building information models regarding applicable regulations. In: Congress on computing in civil engineering, proceedings; 2017. p. 157–165. https://doi.org/10.1061/9780784480823.020.
52. Kokash N, Arbab F. Formal behavioral modeling and compliance analysis for service-oriented systems. In: Lecture notes in computer science. (including subseries Lecture notes in artificial intelligence and lecture notes in bioinformatics); 2009. p. 21–41. https://doi.org/10.1007/978-3-642-04167-9_2.
53. Nowroozi A, Teymoori P, Ramezanifarkhani T, Besharati MR, Izadi M. A crisis situations decision-making systems software development process with rescue experiences. IEEE Access. 2020. https://doi.org/10.1109/ACCESS.2020.2981789.
54. Kokaly S, Salay R, Sabetzadeh M, Chechik M, Maibaum T. Model management for regulatory compliance: a position paper. In: 2016 IEEE/ACM 8th international workshop on modeling in software engineering (MiSE) 2016. ACM; 2016. p. 74–80. https://doi.org/10.1145/2896982.2896985.
55. Van Lamsweerde A. Requirements engineering: from system goals to UML models to software. Chichester: Wiley; 2009.
56. Kelly T, Weaver R. The goal structuring notation—a safety argument notation. In: Proceedings of the dependable systems and networks 2004 workshop on assurance cases; 2004. p. 6.
57. Dardar R, Gallina B, Johnsen A, Lundqvist K, Nyberg M. Industrial experiences of building a safety case in compliance with ISO 26262. In: Proceedings of the 2012 IEEE 23rd international symposium on software reliability engineering workshops, ISSREW 2012; 2012. https://doi.org/10.1109/ISSREW.2012.86.
58. Kelly TP. Arguing safety: a systematic approach to managing safety cases. York: University of York; 1999.
59. Zhang J, El-Gohary NM. Integrating semantic NLP and logic reasoning into a unified system for fully-automated code checking. Autom Constr. 2017;73:45–57. https://doi.org/10.1016/j.autcon.2016.08.027.
60. Siena A, Perini A, Susi A, Mylopoulos J. A meta-model for modelling law-compliant requirements. In: 2009 second international workshop on requirements engineering and law, RELAW 2009. IEEE; 2009. p. 45–51. https://doi.org/10.1109/RELAW.2009.1.
61. Stratigaki C, Loucopoulos P, Nikolaidou M. Designing a meta model as the foundation for compliance capability. In: 1st international workshop on capabilitry-oriented business informatics organized in conjunction with the 16th IEEE conference on business informatics; 2014.
62. Papazoglou MP. Making business processes compliant to standards & regulations. In: Proceedings of the IEEE 15th international enterprise distributed object computing conference, EDOC; 2011. p. 3–13. https://doi.org/10.1109/EDOC.2011.37.
63. Price WJ. A benchmark tutorial. IEEE Micro. 1989;9:28–43. https://doi.org/10.1109/40.45825.
64. Patterson D. For better or worse, benchmarks shape a field: technical perspective. Commun ACM. 2012;55:104–104.
65. Almeida R, Vieira M. Benchmarking the resilience of self-adaptive software systems: perspectives and challenges. In: Proceedings of the 6th international symposium on software engineering for adaptive and self-managing systems; 2011. p. 190–5. https://doi.org/10.1145/1988008.1988035.
66. Whitman M, Mattord H. Readings & cases in information security: law & ethics. Boston: Nelson Education; 2010.
67. Kanoun K, Spainhower L. Dependability benchmarking for computer systems. Hoboken: Wiley-IEEE Computer Society Press; 2008.
68. Friginal J, De Andrés D, Ruiz JC, Gil P. Towards benchmarking routing protocols in wireless mesh networks. Ad Hoc Netw. 2011;9:1374–88. https://doi.org/10.1016/j.adhoc.2011.03.010.
69. Le Goues C, Holtschulte N, Smith EK, Brun Y, Devanbu P, Forrest S, Weimer W. The ManyBugs and IntroClass benchmarks for automated repair of C programs. IEEE Trans Softw Eng. 2015;41:1236–56. https://doi.org/10.1109/TSE.2015.2454513.
70. Tichy WF. Should computer scientists experiment more? Computer. 1998;31:32–40. https://doi.org/10.1109/2.675631.
71. Friginal J, de Andrés D, Ruiz JC, Moraes R. Using dependability benchmarks to support ISO/IEC SQuaRE. In: 2011 IEEE 17th Pacific rim international symposium on dependable computing; 2011.
72. Furia CA, Nordio M, Polikarpova N, Tschannen J. AutoProof: auto-active functional verification of object-oriented programs. Int J Softw Tools Technol Transf. 2017;19:697–716. https://doi.org/10.1007/s10009-016-0419-0.
73. Wang Y. Stream processing systems benchmark: StreamBench. Espoo: Aalto University; 2016.
74. Weide BW, Sitaraman M, Harton HK, Adcock B, Bucci P, Bronish D, Heym WD, Kirschenbaum J, Frazier D. Incremental benchmarks for software verification tools and techniques. In: Working conference on verified software: theories, tools, and experiments. (Lecture notes in computer science); 2008. p. 84–98. https://doi.org/10.1007/978-3-540-87873-5-10.
75. Sattari S, Izadi M. An exact algorithm for the minimum dilation triangulation problem. J Glob Optim. 2017;69:343–67. https://doi.org/10.1007/s10898-017-0517-x.
76. Darmont J. Data-centric benchmarking. In: Encyclopedia of information science and technology. Hershey: IGI Global; 2018. p. 1772–82.
77. Arnett K, Templeton G, Vance D. Information security by words alone: the case for strong security policies. Int J Inf Secur Priv. 2009;3:84–9.

78. Joslin EO. Describing workload for acquiring ADP equipment and software. Comput Autom. 1969;18:36.
79. Joslin EO, Aiken JJ. The validity of basing computer selections on benchmark results. Comput Autom. 1966;15:22–3.
80. Hillegass JR. Standardized benchmark problems measure computer performance. Comput Autom. 1966;15:16–9.
81. Joslin EO, Chairman-Hitti RF. Evaluation and performance of computers: application benchmarks: the key to meaningful computer evaluations. In: Proceedings of the 1965 20th national conference. ACM; 1965. p. 27–37.
82. Yeh J. A report on computer performance evaluation techniques; 1970.
83. Camp RC. Benchmarking: the search for industry best practices that lead to superior performance. In: Benchmarking Search Ind. Best Pract. That Lead to Super. Perform. ASQC/Quality Resources; 1989. https://doi.org/10.5860/choice.27-2173.
84. Pryor LS, Katz SJ. How benchmarking goes wrong. Plan Rev. 1993;21:6–53.
85. Mohapatra S. Information theory and best practices in the IT industry. New York: Springer Science & Business Media; 2012. https://doi.org/10.1007/978-1-4614-3043-8.
86. Scott R. Benchmarking: a literature review. In: Academic excellence centre for learning and development. Joondalup: Edith Cowan University; 2011.
87. Adewunmi YA, Iyagba R, Omirin M. Multi-sector framework for benchmarking in facilities management. Benchmarking. 2017;24:826–56. https://doi.org/10.1108/BIJ-10-2015-0093.
88. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M. A view of cloud computing. Commun ACM. 2010;53:50–8. https://doi.org/10.1145/1721654.1721672.
89. Gillam L, Li B, O'Loughlin J, Tomar APS. Fair benchmarking for cloud computing systems. J Cloud Comput. 2013;2:6. https://doi.org/10.1186/2192-113X-2-6.
90. Zeuch M. Handbook of human resources management. Berlin: Springer; 2016. https://doi.org/10.1007/978-3-662-44152-7.
91. Sim SE, Easterbrook S, Holt RC. Using benchmarking to advance research: a challenge to software engineering. In: Proceedings of the 25th international conference on software engineering; 2003. https://doi.org/10.1109/icse.2003.1201189.
92. Zhioua Z, Roudier Y, Ameur-Boulifa R, Kechiche T, Short S. Tracking dependent information flows. In: ICISSP; 2017. p. 179–89.
93. Antunes N, Vieira M. Assessing and comparing vulnerability detection tools for web services: benchmarking approach and examples. IEEE Trans Serv Comput. 2015;8:269–83. https://doi.org/10.1109/TSC.2014.2310221.
94. Landhäußer M, Weigelt S, Tichy WF. NLCI: a natural language command interpreter. Autom Softw Eng. 2017;24:839–61. https://doi.org/10.1007/s10515-016-0202-1.
95. Guarino N. Formal ontology, conceptual analysis and knowledge representation. Int J Hum Comput Stud. 1995;43:625–40. https://doi.org/10.1006/ijhc.1995.1066.
96. Russell S, Norvig P. Artificial intelligence a modern approach. 3rd ed. Hoboken: Pearson Education, Inc.; 2016.
97. Giblin C, Liu AY, Müller S, Pfitzmann B, Zhou X. Regulations expressed as logical models (REALM). JURIX. 2005;37–48.
98. Nalepa GJ. Modeling with rules using semantic knowledge engineering. Berlin: Springer; 2018.
99. Yip F, Wong AKY, Parameswaran N, Ray P. Towards robust and adaptive semantic-based compliance auditing. In: Proceedings of the 2007 eleventh international IEEE EDOC conference workshop. IEEE; 2007. p. 181–8. https://doi.org/10.1109/EDOCW.2007.33.
100. Fowler M. Domain-specific languages. London: Pearson Education; 2010.
101. Eastman C, Lee JM, Jeong YS, Lee JK. Automatic rule-based checking of building designs. Autom Constr. 2009;18:1011–33. https://doi.org/10.1016/j.autcon.2009.07.002.
102. Ismail AS, Ali KN, Iahad NA. A review on BIM-based automated code compliance checking system. In: 2017 international conference on research and innovation in information systems (ICRIIS). IEEE; 2017. p. 1–6. https://doi.org/10.1109/ICRIIS.2017.8002486.
103. Elgammal A, Turetken O, Van Den Heuvel WJ. Using patterns for the analysis and resolution of compliance violations. Int J Coop Inf Syst. 2012;21:31–54. https://doi.org/10.1142/S0218843012400023.
104. Bhatt GD. Knowledge management in organizations: examining the interaction between technologies, techniques, and people. J Knowl Manag. 2001;5:68–75. https://doi.org/10.1108/13673270110384419.
105. Jonassen DH, Marra RM. Concept mapping and other formalisms as mindtools for representing knowledge. ALT-J. 1994;2:50–6. https://doi.org/10.1080/0968776940020107.
106. Johnsen HCG. The new natural resource: knowledge development, society and economics. London: Routledge; 2016.
107. Ortony A. The representation of knowledge in memory. In: Montague WE, Anderson RC, Spiro RJ, editors. Schooling and the acquisition of knowledge. Hillsdale: Lawrence Erlbaum Associates; 1977. p. 99–135.
108. Sato M. Classical Brouwer–Heyting–Kolmogorov interpretation. In: Algorithmic learning theory. Berlin: Springer; 1997. p. 176–96.
109. Basili VR, Caldiera G, Rombach HD. The goal question metric approach. Encycl Softw Eng. 1994;528–532.
110. Hu H, Wang S, Bezemer CP, Hassan AE. Studying the consistency of star ratings and reviews of popular free hybrid Android and iOS apps. Empir Softw Eng. 2019. https://doi.org/10.1007/s10664-018-9617-6.
111. Arianto R, Gaol FL, Abdurachman E, Heryadi Y, Warnars HL, Soewito B, Perez-Sanchez H. Quality measurement of android messaging application based on user experience in microblog. In: Proceedings of the 2017 international conference on applied computer and communication technologies (ComCom) 2017. IEEE; 2017. p. 1–5. https://doi.org/10.1109/COMCOM.2017.8167099.
112. Lien CH, Cao Y, Zhou X. Service quality, satisfaction, stickiness, and usage intentions: an exploratory evaluation in the context of WeChat services. Comput Human Behav. 2017;68:403–10. https://doi.org/10.1016/j.chb.2016.11.061.
113. Hu H. Studying the perceived quality consistency of cross-platform mobile apps. Kingston: Queen's University; 2017.

114. Jongerius CM. Quantifying chatbot performance by using data analytics. Utrecht: Utrecht University; 2018.
115. Vaziripour E, Farahbakhsh R, O'Neill M, Wu J, Seamons K, Zappala D. A survey of the privacy preferences and practices of Iranian users of telegram. In: Workshop on usable security; 2018. https://doi.org/10.14722/usec.2018.23033.
116. Hashemi A, Zare Chahooki MA. Telegram group quality measurement by user behavior analysis. Soc Netw Anal Min. 2019;9:33. https://doi.org/10.1007/s13278-019-0575-9.
117. Kermani H, Mozaffari A. The study of Iranian users' reasons in preferring Telegram on other instant messaging applications. MEDIA Stud. 2018;13:7–20.
118. Büyüközkan G, Havle CA, Feyzioğlu O. A new digital service quality model and its strategic analysis in aviation industry using interval-valued intuitionistic fuzzy AHP. J Air Transp Manag. 2020;86: 101817. https://doi.org/10.1016/j.jairtraman.2020.101817.
119. Yue C. An intuitionistic fuzzy projection-based approach and application to software quality evaluation. Soft Comput. 2020;24:429–43.
120. Khan M, Ansari MD. Multi-criteria software quality model selection based on divergence measure and score function. J Intell Fuzzy Syst. 2020;38:1–10. https://doi.org/10.3233/JIFS-191153.
121. Schneider Y, Busch A, Koziolek A. Using informal knowledge for improving software quality trade-off decisions. In: Lecture notes in computer science (including subseries Lecture notes in artificial intelligence and lecture notes in bioinformatics). Springer; 2018. p. 265–283. https://doi.org/10.1007/978-3-030-00761-4_18.
122. Perry DE, Sim SE, Easterbrook SM. Case studies for software engineers. In: Proceedings of the 28th international conference on software engineering; 2006. p. 1045–6. https://doi.org/10.1109/icse.2004.1317512.
123. Besharati MR, Izadi M. IR-QUMA. 2020. Mendeley Data. https://doi.org/10.17632/d89gphmnsk.3.
124. Likert R. A technique for the measurement of attitudes. Arch Psychol. 1932;140:1–55.
125. Gren L, Torkar R, Feldt R. The prospects of a quantitative measurement of agility: a validation study on an agile maturity model. J Syst Softw. 2015;107:38–49. https://doi.org/10.1016/j.jss.2015.05.008.
126. Izadi M. Model checking of component connectors. Leiden: Leiden University; 2011.

Publisher's Note