Systematizing the Relationship Between Business Processes' and Web Services' Non-functional Requirements

Camila F. Castro Jr.[1], Marcelo Fantinato[1(✉)], Ünal Aksu[2], Hajo A. Reijers[2], and Lucinéia H. Thom[3]

[1] School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, Brazil
{marcos.freitas,m.fantinato}@usp.br
[2] Department of Information and Computer Sciences, Utrecht University, Utrecht, The Netherlands
{u.aksu,h.a.reijers}@uu.nl
[3] Institute of Informatics, Federal University of Rio Grande do Sul, Porto Alegre, Brazil
lucineia@inf.ufrgs.br

Abstract. We propose in this paper a conceptual framework for the hierarchical decomposition of Non-Functional Requirements (NFRs) from the business process level to the web service level. This framework seeks to reduce the dependence on a particular IT expert's knowledge by simplifying the dialog between the business and IT areas. The proposed framework relies on a structure of NFRs interdependence. The main reference was the ISO/IEC 25010 Product Quality Model, extended by additional software quality models and particular QoS attributes. This framework is accompanied by an extensive dictionary of non-functional requirements for both business processes and web services that can serve as a reference for researchers and industry practitioners. We assume that orchestrating web services to run business processes requires a rigorous definition of the functional requirements and NFRs of these web services. Web service NFRs are often defined as Quality of Service (QoS) attributes, which is done at the implementation level by IT teams. The definition of QoS attributes should consider the business process NFRs, since misinterpretations of web service NFRs may affect the behavior of the web services and hence achieving the business goals. The approaches proposed so far in the literature are still heavily dependent on an IT expert's knowledge to identify the appropriate QoS attributes required to meet particular business process NFRs. However, defining appropriate QoS attributes without reference to business process-level NFRs may be a costly, time-consuming task.

Keywords: Non-functional Requirements · Service Level Agreements · Quality of Services · Web services

J. Filipe et al. (Eds.): ICEIS 2019, LNBIP 378, pp. 473–497, 2020.
https://doi.org/10.1007/978-3-030-40783-4_23

1 Introduction

Software Engineering has fostered approaches that reuse software components to implement business functionalities to reduce cost, time and effort throughout the software life-cycle. A modern and popular approach is Service-Oriented Architecture (SOA): a framework in which business functionalities are built, deployed and integrated as autonomous services [24]. Services provided and accessed over the web are denominated web services, and their widespread use by organizations has made them the most popular implementation of SOA. Through web services, business processes can be implemented and executed by assembling and coordinating business activities among corresponding web services, using a concept denominated web service composition or orchestration [24]. Web services invoke software code that should execute a corresponding business activity.

To ensure the success of executing a business process through a web service orchestration, functional requirements and Non-functional Requirements (NFRs) of the web services should be considered. Web service NFRs are often defined as Quality of Service (QoS) attributes, which are formalized in Service Level Agreements (SLA) established between web service providers and consumers. QoS attributes defined in SLAs are propagated from specific business goals [24], for instance: a business goal related to agility may require QoS attributes such as adaptability, scalability and extensibility. Therefore, different web services require different QoS attributes, and what attributes are required depends on the business domain, intended use and user requirements [2].

Seeking strategic alignment, the definition of QoS attributes in SLAs should rely on business process NFRs. Business process NFRs can be formalized in Business Level Agreements (BLA), which are should be defined by business or requirements analysts and capture business process-level NFRs useful later for web service provisioning [7,27,28]. However, a decomposition of BLAs into SLAs would depend on an IT expert's knowledge to identify the appropriate QoS attributes required for a web service, based on implicit business process NFRs.

As an illustrative example, Fig. 1 shows a fragment of a business process model for assessing loan against property applications [11]. Once received the customer application form from the *Loan Officer*, the *Financial Officer* needs to check the customer's credit history to assess the loan risk, while the *Property Appraiser* appraises the property. When both of them complete these activities, the *Loan Officer* is able to assess the customer's eligibility for the requested loan. This set of activities is susceptible to some NFRs. For example, the execution of these activities may include BLAs related to: *(i)* security, since private customer data in other institutions should be accessed; *(ii)* performance efficiency, since a rapid response must be sent to the customer so as not to lose this business opportunity; and *(iii)* compliance, as regulatory rules may need to be met. The web services implementing these activities are also susceptible to related NFRs since the effective execution of the business process depends directly on the effective execution of the web services.

Translating the business process NFRs exemplified above into appropriate NFRs to provide web services to support the execution of such a business process

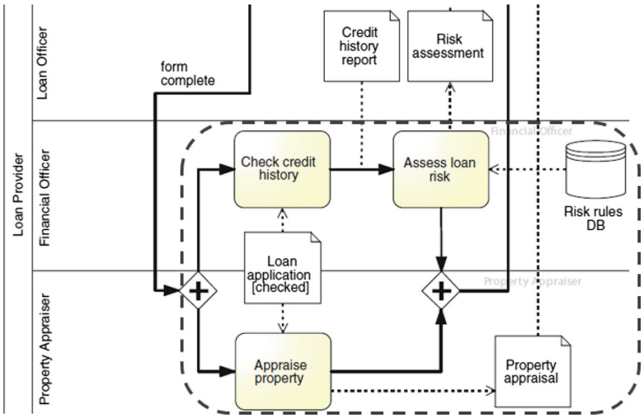

Fig. 1. Fragment of a business process for assessing loan applications [11].

is a challenge. Assuming the business team defines a BLA stating the execution of these three activities together should not exceed 5 min: what QoS properties for which web services should be defined to ensure achieving this business goal? The challenge addressed here is how IT teams can define which SLAs (and for what web services) are needed to meet a BLA defined by business teams.

We present here a conceptual framework to hierarchically decompose NFRs from the business process level to the web service level. This framework relies on the NFR-interdependence among NFRs. Given a business process-level NFR, this framework describes the related NFRs that could be considered at the web service level to meet business goals. This decomposition is not automated, but an approach to help IT teams in breaking down preferences of business process NFRs into more detailed preferences related to web service QoS attributes. Our motivation is the importance of an appropriate web service execution to meet business goals, which requires pertinent QoS attributes based on business needs.

To achieve this goal, we first developed a dictionary of NFRs, including both business process and web service levels. Then, we defined the interdependence among the NFRs at both levels via UML (Unified Modeling Language) class diagrams. The interdependence framework considers the relationships among business process-level NFRs, and between the business process and web service levels in a top-down strategy. The ISO/IEC 25010 Product Quality Model is the main reference, enriched by extra quality models on software and QoS attributes.

A preliminary version of this paper has already been published in [8], which is extended here with the full version of the web services' non-functional requirements dictionary. The main contributions of this work are several:

– Extensive gathering of NFRs related to business processes and technical aspects of web service provisioning, and the definition of their interdependence relationships to support those who want to systematize their decomposition.

- Conceiving a conceptual framework that, while designed primarily for the context of business process automation through web service orchestration, is generic enough to be used or adapted to other areas considering NFRs.
- Presenting an approach that, while not automatic, can be further used to support semi-automated decision making.
- Delivering an unprecedented compilation of NFR for both levels - business process and web services, with a hierarchical view among them, which can be used as a reference by researchers and industry practitioners.

The rest of this paper presents the following sections: underlying concepts, research method, obtained results, related work and concluding remarks.

2 Underlying Concepts

2.1 Business Process Automation

A business process comprises a set of activities executed in a specific order to achieve a common organizational goal [32]. Business Process Management (BPM) refers to overseeing how work is performed in an organization, to ensure consistent outcomes and take advantage of improvement opportunities [11]. The BPM life-cycle involves the identification, analysis, redesign, implementation and monitoring of business processes.

Considering the expected strategic alignment between business and Information Technology (IT) as background, activities of business processes can be outsourced by services provided by business partners, using the SOA-style architecture. A service implements a well-defined piece of business functionality, with a published and discoverable interface [24]. Through SOA, requesters can reuse services to implement business functionalities, and the only part they need to know is the service description, i.e., they can abstract the details of the underlying logic and implementation [24].

The popularity of SOA increased as services become available on the web, giving rise to web services. A web service can be as simple as performing a credit card number check or as complex as dealing with a mortgage application [19].

Using web services to implement business functionalities allows automating the execution of business processes by orchestrating web services, i.e., orchestrating heterogeneous web services to achieve a common business goal [2,12]. A web service orchestration strategy requires the pre-modeling of a business process in terms of both functional requirements and NFRs. This modeling is relevant, considering that the behavior of an individual activity performed through a web service can influence the entire business process and hence the achievement or not of an organizational business or strategic goal.

2.2 Non-functional Requirements

In software engineering, Non-functional Requirements (NFRs) are defined as constraints on services or functionalities offered by a system, including characteristics related to software behavior and constraints imposed by standards [30].

These NFRs are defined based on user needs, budget constraints or external factors, such as regulatory and legislative determinations [30]. Examples are performance, security and availability.

In BPM and SOA, NFRs represent quality aspects expected in the provisioning of services responsible for executing business processes. These quality aspects define guarantee levels which allow comparison among distinct services offering the same functionality [3]. Web service NFRs are often expressed as QoS attributes and their specification is usually made through SLAs.

SLAs refers to a commitment between web service providers and consumers, whereby the exact quality conditions that guide the web service provisioning are systematically defined [27][1]. An SLA could include, for instance, a QoS attribute for availability with a target of 99% and a QoS attribute for response time with a target of 5 ms. In SLAs, penalties and rewards are defined and imposed depending on the breach of pre-defined guarantee terms.

SLA terms are defined by IT considering technical aspects of web service provisioning [7]. However, business aspects should be also considered, mainly in the context of business process automation via web service orchestration [5]. A different type of agreement is then required, what can be done through BLAs.

The structure of a BLA is like of an SLA, including penalties and rewards. The main difference lies in their scopes: while SLAs are associated with web services and consider mainly technical aspects involved in web service provisioning, BLAs are associated with business process activities that will be executed in the form of web services [5,27]. BLAs are defined during business process analysis and modeling whereas SLAs are determined during business process implementation and execution [27].

The differences between BLA and SLA are illustrated by Salles et al. [28] who exemplify a BLA goal with *"the business subprocess starting in the activity [1] and ending in the activity [4] must be concluded within 24 h"* whereas the corresponding SLAs goals are exemplified with *"the web service invoked to execute the activity [1] must be completed within 2 h"* and *"the web service invoked to execute the activity [4] must have 95% of availability"*.

A BLA can be mapped to a set of SLAs, each BLA related to a specific business process activity automated through a set of web services with their SLAs [13]. Assuming that all the guaranteed terms of each SLA are satisfied, the corresponding BLA is expected to be satisfied accordingly.

2.3 Software Quality Models

According to the IEEE Standard Glossary of Software Engineering Terminology [15], software quality means the degree to which a system, component or business process meets specific requirements. Specifying functional requirements

[1] In this work, only technical aspects involved in a web service provisioning are considered in SLAs; i.e., IT outsourcing or out-tasking web services for higher-level tasks, including human tasks, are not part of the scope.

and NFRs for software is not a trivial task; a common approach is to use software quality models as a reference to describe and assess software requirements.

Quality models support the identification of relevant quality characteristics to be used as requirements and their corresponding satisfaction criteria and measures [17]. Quality models provide the foundation for software evaluation, providing consistent QoS terminology and supporting software measurement [6].

There are several software quality models proposed in the literature from international standards to several domain and company-specific models. One of the most popular approaches is the ISO/IEC 25010 System and Software Quality Model, which is a part of the ISO/IEC 25000 Software Product Quality Requirements and Evaluation (SQuaRe) model and results from evolving several other standards, especially the ISO/IEC 9126 [17]. ISO/IEC 25010 addresses a set of QoS attributes for software product quality and software quality in use.

Figure 2 shows the structure of ISO/IEC 25010 Product Quality Model, depicting its eight main quality characteristics and 31 related sub-characteristics. Alternative software quality models were proposed [4,10,21]. Detailed information about quality models can be found in [22,29,31].

Fig. 2. ISO/IEC 25010 Product Quality Model [17].

Regarding BPM and SOA, quality models can be used as a reference to define requirements related to business processes and web services, allowing for the overall quality improvement of SOA-based applications. To the best of our knowledge, there is no general standard accepted as a quality model for web services being orchestrated to automate business processes. However, web services and software modules share the same set of properties; therefore, if software components can be replaced by web services, then the quality requirements of both solutions must be compatible with [1]. As a result, software quality models can also be used to address quality characteristics of web services.

As the ISO/IEC 25010 quality model is a recognized quality standard for any type of software, it can be used to provide QoS attributes for web services [1]. Other quality models used in the context of web services were proposed [2,23].

3 Research Method

This work was developed following principles of the *design science* research paradigm, which considers the creation and evaluation of artifacts to solve

identified organizational problems [14]. These artifacts need to address an unsolved problem or propose an improvement for an existing solution to more significantly contribute with science and practice [14]. In this research, the problem refers to the lack of a systematic structure to support a straightforward decomposition of NFRs, from business to QoS attributes related to web services. However, contrasting the paradigm, this research did not include a validation work to ascertain the results, thus resulting in a theoretical research based on literature analysis. In this context, developing an conceptual interdependence framework for NFRs included two major activities: *(i)* the elaboration of a dictionary of NFRs, considering NFRs for both business process and web service levels; and *(ii)* the definition of interdependence relationships between identified NFRs, taking relationships between NFRs at the same level (for the business process level) and relationships between NFRs at different levels (from the business process level to the lower levels).

Regarding the dictionary of NFRs, an exploratory literature study was conducted to elicit a set of quality characteristics related to business or technical aspects of web service provisioning. The structure of characteristics and subcharacteristics of the ISO/IEC 25010 Product Quality Model was the main reference due to its wide use and acceptance by business and IT practitioners [1]. This base structure was expanded through extra research on software and web services quality models and studies related to SOA and QoS attributes.

The definition of the interdependence relationships among the dictionary's NFRs was based on the studies of McCall, Richards and Walters [21] and Zulzalil et al. [35] which were used as the main references to describe the relationships between the business process-level NFRs. Although these studies predate the publication of the ISO/IEC 25010 Product Quality Model, both share the same evaluated characteristics.

With respect to the relationships between NFRs from the business process level to the web service level and also between NFRs at the web service level, the structure of characteristics and sub-characteristics of the ISO/IEC 25010 Product Quality Model was also used as the main reference. For most of the NFRs got from other references during the elaboration of the dictionary, the corresponding studies already incorporated some hierarchical classification that could be the basis to define the decomposition structure.

Remaining relationships were determined via logical inference based on empirical analysis. The authors conducted iterative brainstorming meetings to discuss potential relationships between the NFRs mapped in the dictionary. The ideas that came up during these meetings were refined resulting on a final set of relationships, which are presented as follows.

4 NFR Decomposition

Considering business processes being automated through web service orchestration, a conceptual NFR decomposition framework is proposed to support a straightforward definition of web service QoS attributes. The definition of QoS

attributes is carried out based on constraints determined by business areas at process modeling time. Using this approach, IT teams are given hints of which QoS attributes they can assign to a web service to meet a business demand. Thus, the expected users for this framework are IT teams working on the perspective of a web service provider and hence involved in the definition of web service SLAs to be executed by business units.

The designed NFR decomposition framework is presented below. An explanation of the framework is first given, with details on the dictionary of NFRs and the interdependence diagram, followed by the decomposition diagrams.

4.1 Conceptual Framework Overview

The set of NFRs is organized into a dictionary structure. The dictionary of NFRs is formed by two sections: one for business process NFRs (cf. Table 1) and another for web service NFRs (cf. Tables 2, 3, 4, 5, 6, 7, 8, 9). The structure of both sections comprises four attributes: **ID**, a numerical NFR identification; **Name**, the NFR name; **Definition**, a brief description of the NFR; and **Reference**, the references of the works from which the NFR was extracted from. Synonyms are identified and grouped using a unique ID. Specifically for web service NFRs, there is an extra attribute, **Measurement Unit**, which identifies the primary unit used to measure a quantitative NFR. The measurement unit was filled in the dictionary only when found in the literature.

Table 1. Dictionary of business process NFRs [8].

ID	Name	Definition	Ref.
1	Performance efficiency	Degree to which a business process can efficiently use an amount of resources (such as software, products, hardware and generic materials) under stated conditions	[17]
2	Compatibility	Degree to which a business process can exchange information with other business processes, and/or perform its activities while sharing the computing environment	[17]
3	Usability	Degree to which a business process can be used by specified users to achieve specific goals with effectiveness, efficiency and satisfaction	[17]
4	Reliability	Degree to which a business process performs specified activities under specified conditions for a period	[17]
5	Security	Degree to which a business process can protect information and data from unauthorized access	[17]
6	Maintainability	Degree of effectiveness and efficiency with which the activities of a business process can be modified	[17]
7	Portability	Degree of effectiveness and efficiency with which a business process can be configured in an environment and transferred from one environment to another	[17]
8	Compliance	Degree to which a business process is compliant with internal procedures of an organization and external guidelines	[30]

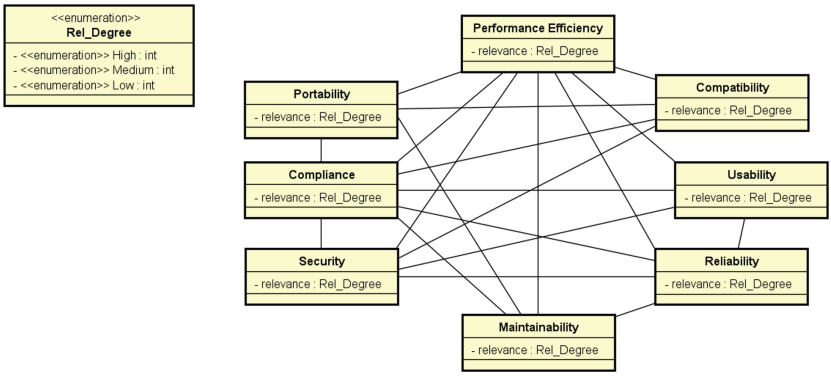

Fig. 3. Interdependence relationships for business process NFRs [8].

Considering the relationships between the NFRs, they are represented through UML class diagrams (cf. Figures 3, 4, 5, 6, 7, 8, 9, 10, 11). Each class in the diagram represents an NFR included in the dictionary. Relationships between NFRs at the same level are represented using the *association* bidirectional connector (e.g., Fig. 6, association between *Confidentiality* and *Access Control*). Relationships between NFRs of different levels are defined through *aggregation* connectors, i.e., lower-level NFRs contribute to those at a higher level, although they exist independently (e.g., Fig. 6, aggregation between *Confidentiality* and *Encryption*).

Each class in the diagram includes a configurable attribute denominated **relevance**, which considers three values: high, medium or low. Business and IT areas should use this attribute to show which NFRs are how likely relevant when creating SLAs in an organization, based on the business domain and previous experiences. As the proposed decomposition framework has been developed to be generic enough to be considered in different organizational contexts, no prior definition of relevance for each NFR is provided. As a result, each organization willing to use this framework should define its own relevance values considering its own context and historical data.

Table 2. Web services' dictionary of NFRs – performance efficiency.

ID	Name	Definition	Meas. Unit	Ref.
1	Capacity	Maximum limits of a service (i.e., concurrent users, stored data etc.) for which performance is guaranteed		[17]
2	Execution time	Time for a service to execute a sequence of activities and process a request	Time	[18]
3	I/O utilization	Measurement of estimated I/O utilization to complete a specified task	Number of buffers	[16]
4	Latency time	Round-Trip Delay (RTD) between the dispatch of a request and receive of a response for a service	Time	[3]
5	Memory utilization	Measurement of the estimated memory size a service will occupy to complete a specified task	Bytes	[16]
6	Resource utilization	Degree to which the amount and type of resources used meet requirements in service provisioning		[17]
7	Response time	Time necessary to complete a certain service request, from the moment it is dispatched until a response is received	Time	[18]
	[Average and maximum response time]	Mean time needed for the packet of control data to get to the provider's server and return to the requester	Time	[3]
	[Execution duration]	Expected delay from the dispatch of a service request until the result is received by the client	Time	[34]
8	Scalability	Degree to which a service operates correctly, without degradation of other quality attributes, when the system is changed in size or in volume in order to meet users' needs		[24]
9	Throughput	Measurement of the number of service requests served in a given time interval	Processed rqts/time	[18]
	[Maximum Throughput]	Maximum number of services that a platform providing services can process for a unit time		[23]
10	Time behavior	Degree to which the response and processing times and throughput rate meet requirements in service provisioning		[17]
11	Timeliness	Degree to which a service meets deadlines, i.e., to process a request in a deterministic and acceptable amount of time		[24]
12	Transaction time	Time that passes while the service is completing one complete transaction. This concept may depend on the definition of a service transaction	Time	[18]
13	Transmission utilization	Estimated amount of transmission resources utilized by a service	Bits/time	[16]

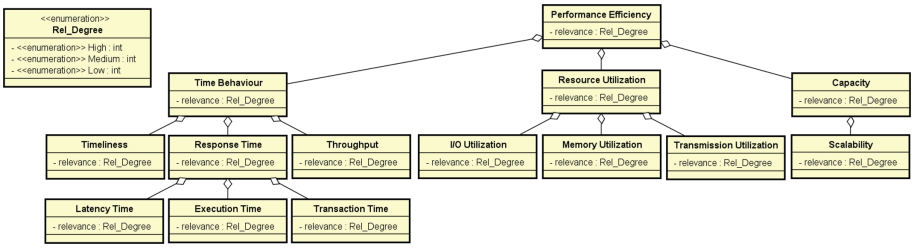

Fig. 4. NFR decomposition diagram – performance efficiency [8].

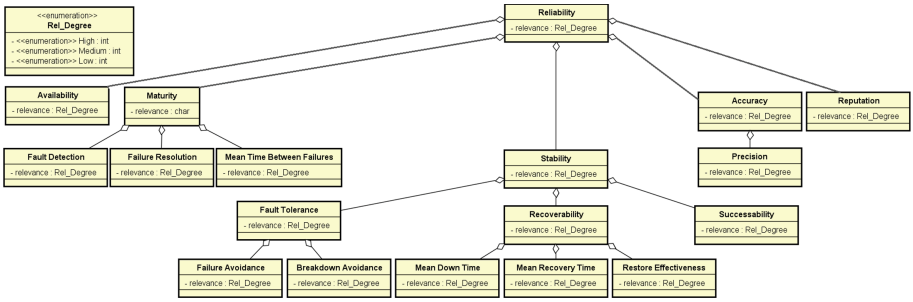

Fig. 5. NFR decomposition diagram – reliability [8].

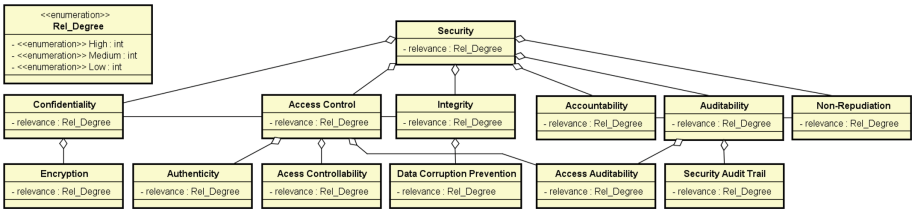

Fig. 6. NFR decomposition diagram – security [8].

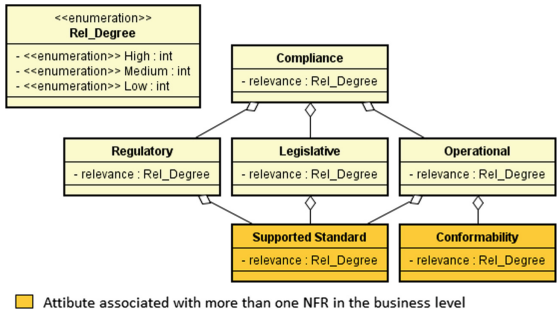

Fig. 7. NFR decomposition diagram – compliance [8].

Table 3. Web services' dictionary of NFRs – reliability.

ID	Name	Definition	Meas. Unit	Ref.
1	Accuracy	Degree of precision to which a service provides right outcomes or effects		[29]
2	Availability	Proportion of total time during which a service is operational and accessible when required for use	Percentage	[17]
3	Breakdown avoidance	Proportion of breakdowns in production environment in comparison to the total number of failures	Percentage	[16]
4	Failure avoidance	Proportion of fault patterns identified by the service to avoid critical and serious failures, considering the number of executed test cases of fault patterns during testing	Percentage	[16]
5	Failure resolution	Proportion of resolved fault conditions that do not recur in relation to the number of failures detected	Percentage	[16]
6	Fault detection	Measurement of failures detected by a service in a certain period of time	Detected failures	[16]
7	Fault tolerance	Degree to which a service operates as intended despite the presence of hardware or software faults		[17]
	[Error tolerance]	Degree to which a service provides continuity of operation under abnormal conditions		[21]
8	Maturity	Degree to which a service meets needs for reliability under normal operation		[17]
9	Mean down time	Mean time a service stays unavailable when a failure occurs before it gradually starts up	Time	[16]
10	Mean recovery time	Mean time a service takes to complete recovery from initial partial recovery	Time	[16]
11	Mean time between failures	Mean time between failures of a service in operation	Time	[16]
12	Precision	Measurement of the frequency to which users encounter results with inadequate precision	Inaccurate results/ Time	[16]
13	Recoverability	Degree to which, in the event of an outage or failure, a service can recover directly affected data and restore the desired operational state		[17]
14	Reputation	Measurement of a service's trustworthiness in terms of service quality, user's satisfaction and reliability on its operation		[20]
15	Restore effectiveness	Proportion of successful restorations meeting the target restore time in comparison to the number of restorations required	Percentage	[16]
16	Stability	Degree to which a service can deliver continuous, consistent and recoverable services despite increased throughput, congestion, system failures, natural disasters and intentional attacks		[23]
17	Sucessability	Degree to which services yield successful results over request messages	Percentage	[23]

Table 4. Web services' dictionary of NFRs – security

ID	Name	Definition	Meas. Unit	Ref.
1	Access auditability	Proportion of user accesses to a service recorded in the access history database	Percentage	[16]
2	Access control	Degree to which a service restricts unauthorized user's access by using services security token or similar approach		[23]
3	Access controllability	Proportion of illegal operations detected by the service, in comparison to the number of illegal operations defined in the specification	Percentage	[16]
4	Accountability	Degree to which the actions of a user can be traced uniquely to the user		[17]
5	Auditability	Degree to which a service keeps sufficiently adequate records in the database to support financial or legal audits		[24]
6	Authenticity	Degree to which the identity of a subject or resource can be proved		[17]
7	Confidentiality	Degree to which a service ensures that its data is accessible only by authorized users		[17]
	[Data confidentiality]	Degree to which a service protects data against unauthorized disclosure		[23]
	[Privacy]	Degree to which access to sensitive data by unauthorized people can be controlled		[21]
8	Data corruption prevention	Measurement of data corruption events identified and prevented by the service	Corruption events	[16]
9	Encryption	Degree to which a service's data is encrypted, making it unreadable without special knowledge		[25]
10	Integrity	Degree to which a service prevents unauthorized access to, or modification of, functions and data		[17]
	[Data integrity]	Degree to which a service ensures that data has not been altered or destroyed in an unauthorized manner		[23]
11	Non-repudiation	Degree to which action or events in a service can be proven to have taken place, so that they cannot be repudiated later		[17]
12	Security audit trail	Degree to which a service records a log of attempted attacks in order to evaluate its vulnerability		[23]

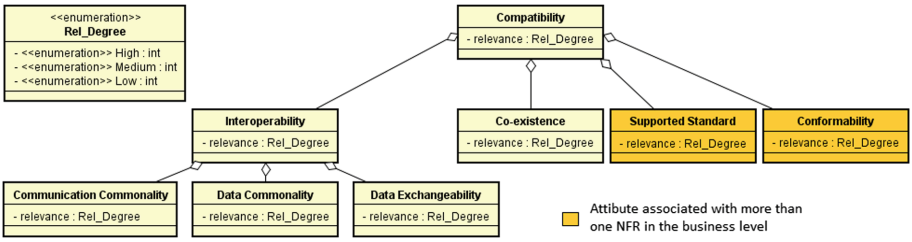

Fig. 8. NFR decomposition diagram – compatibility [8].

Table 5. Web services' dictionary of NFRs – compliance.

ID	Name	Definition	Meas. Unit	Ref.
1	Conformability	Degree to which a service uses the standard technology for services (i.e. SOAP, WSDL e UDDI)		[24]
2	Legislative	Legislative requirements that must be followed to ensure that the service operates within the law		[30]
3	Operational	Operational process requirements that define how a service should be used		[30]
4	Regulatory	Regulatory requirements that set out what must be done for the service to be approved for use by a regulator		[30]
5	Supported standard	Degree to which a service operation complies with standards (e.g. industry specific standards)		[26]

Table 6. Web services' dictionary of NFRs – compatibility.

ID	Name	Definition	Meas. Unit	Ref.
1	Co-existence	Degree to which a service can perform its required functions while sharing environment and resources with other products		[17]
2	Communication commonality	Degree to which a service uses standard protocols and interface routines for communication		[21]
3	Conformability	[as defined for *compliance*]		
4	Data commonality	Degree to which a service uses standard data representations		[21]
5	Data exchangeability	Proportion of successful data transfers between the target service and other services	Percentage	[16]
6	Interoperability	Degree to which two or more components or services can exchange information and use the information that has been exchanged		[17]
	[Integrability]	Degree to which two or more components and services of a system are integrated		[25]
7	Supported standard	[as defined for *compliance*]		

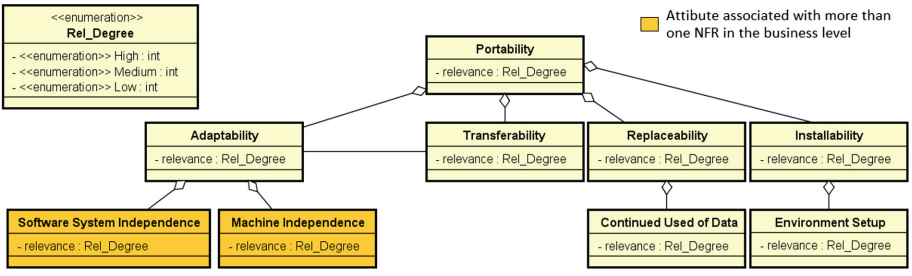

Fig. 9. NFR decomposition diagram – portability [8].

Table 7. Web services' dictionary of NFRs – portability

ID	Name	Definition	Meas. Unit	Ref.
1	Adaptability	Degree to which a service can effectively and efficiently be adapted for different hardware, software or other operational environments		[17]
2	Continued use of data	Proportion of data that can be used in the same way after service migration or replacement, in comparison to the number of total data items required to be used from old services	Percentage	[16]
3	Environment setup	Ease with which a service can be configured to be used in a certain environment		–
4	Installability	Degree of effectiveness and efficiency with which a component of a service can be successfully installed and/or uninstalled in a specified environment		[17]
5	Machine independence	Degree of service dependency on the hardware system		[21]
6	Replaceability	Degree to which a service can replace another for the same purpose in the same environment		[17]
7	Software system independence	Degree of service dependency on the software environment, including operating systems, utilities, input/output routines, etc		[21]
8	Transferability	Ease of moving a computer program from one computing environment to another		[21]

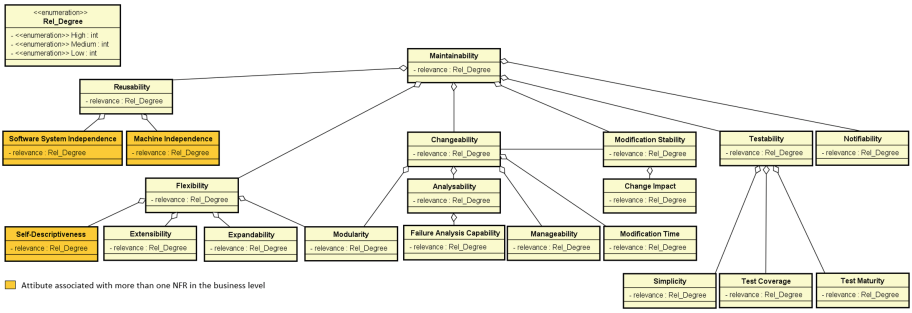

Fig. 10. NFR decomposition diagram – maintainability [8].

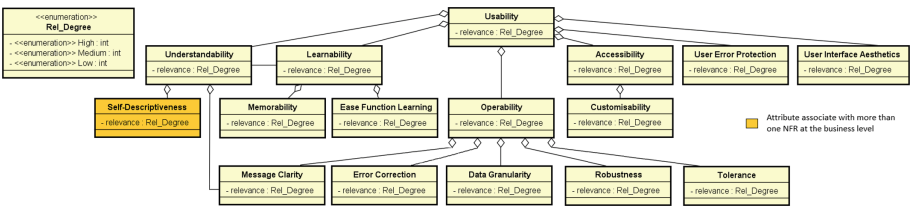

Fig. 11. NFR decomposition diagram – usability [8].

Table 8. Web services' dictionary of NFRs – maintainability.

ID	Name	Definition	Meas. Unit	Ref.
1	Analyzability	Degree of effectiveness and efficiency with which a change in parts of a service is evaluated		[17]
2	Change impact	Proportion of detected adverse impacts after the modification of a service, in comparison to the number of modifications performed	Percentage	[16]
3	Expandability	Degree to which a service can be expanded relative to data or function requirements		[21]
4	Extensibility	Ease with which service features can be extended without affecting other services/systems		[24]
	[Extendibility]	Degree to which a service can be expanded to new specification changes or other domains		[29]
5	Failure analysis capability	Proportion of times a user can identify which operations caused a service to fail considering the total number of failures detected	Percentage	[16]
6	Flexibility	Effort needed to change an operational service		[29]
7	Machine independence	[as defined for *portability*]		
8	Manageability	Degree to which a service lends itself to efficient administration of its components		[21]
9	Changeability	The ease with which a service can be modified		[17, 22]
	[Modifiability]	Degree to which a service can be effectively and efficiently modified without introducing new defects or degrading the product quality		[17]
10	Modification stability	Degree to which a service can avoid unexpected effects from modification of the software		[17, 22]
11	Modification time	Time required to modify a part of a service, from identifying the new requirement or modification to its implementation and validation	Time	[21]
12	Modularity	Degree to which a service is composed of discrete components such that a change to one component has minimal impact on other components		[17]
13	Notifiability	Degree to which service providers notify changes on a service's functions and resources to an external quality manager or any other stakeholder		[23]
14	Reusability	Degree to which an asset can be used in more than one service or to develop other services		[17]
15	Self-Descriptiveness	Ease with which a service's functions and documentation can be understood by humans		[21]
16	Simplicity	Degree to which the functions of a service are implemented in the most understandable manner, avoiding practices that increase complexity		[21]
17	Software system independence	[as defined for *portability*]		
18	Test coverage	Degree to which a service is effectively tested in terms of source code statements executed	Percentage	[30]
19	Test maturity	Proportion of test cases that have been successfully performed during testing in comparison to the total number of test cases	Percentage	[16]
20	Testability	Degree of effectiveness and efficiency with which test criteria can be established for a service and tests can be performed to determine whether those criteria have been met		[17]

Table 9. Web services' dictionary of NFRs – usability.

ID	Name	Definition	Meas. Unit	Ref.
1	Accessibility	Degree to which a service can be used by people with distinctive characteristics and capabilities to achieve a specified goal in a context of use		[17]
2	Customizability	Proportion of operations and procedures of a service that can be customized by the user	Percentage	[16]
3	Data granularity	Granularity of data a service provides in response to user requests		[24]
4	Ease of function learning	Mean time a user takes to learn to use a service function correctly	Time	[16]
5	Error correction	Mean time a user takes to correct an error on a task while using the service	Time	[16]
6	Learnability	Degree to which a service can be used by specific users to achieve specific learning goals with effectiveness, efficiency and satisfaction		[17]
7	Memorability	Degree to which users remember the operations of a service over time	Time	[25]
8	Message clarity	Proportion of messages with clear explanations provided by a service, from the total number of messages implemented	Percentage	[16]
	[Communicativeness]	Degree to which a service provides useful inputs and outputs which can be assimilated by users		[21]
9	Operability	Degree to which a service has attributes that make it easy to operate and control, in conformance with user expectations		[17]
	Ease of use	Degree to which a service is easy for users to operate and control		[16]
10	Robustness	Degree that represents the ability of the service to act properly even if some of the input parameters are missing or incorrect		[3]
	[Exception handling]	Degree that represents how well a service handles exceptions on data inputs		[26]
11	Self-Descriptiveness	[as defined for *maintainability*]		
12	Tolerance	Degree to which a service accepts different forms of the same data as valid or supports some input variation without malfunction or rejection		[21]
13	Understandability	Degree to which a user understands the logical concept of a service and its applicability		[25]
	[Appropriateness Recognizability]	Degree to which users can recognize whether a service is appropriate for their needs		[17]
14	User error protection	Degree to which a system protects users against making errors		[17]
15	User interface aesthetics	Degree to which a user interface enables satisfying interaction for the user, such as the use of color and the nature of the graphical design		[17]

4.2 Decomposition Diagrams

The NFR decomposition framework considers attributes at business process and web service levels, defined by business and IT areas, respectively. For business process NFRs, the first section of the dictionary has eight attributes (cf. Table 1).

To identify the business process-level NFRs, the characteristics proposed in the ISO/IEC 25010 Product Quality Model [17] were considered as describing generic aspects of product quality to be selected by business areas [9]. From eight characteristics in the ISO/IEC 25010 model (cf. Fig. 2), seven were adapted to be added in the dictionary as business process NFRs: *Performance Efficiency*, *Compatibility*, *Usability*, *Reliability*, *Security*, *Maintainability* and *Portability*. Only *Functional Suitability* was not considered as it addresses functional requirements and not NFRs which is the purpose of this dictionary.

ISO/IEC 25010 and other related software quality models describe quality characteristics only from the product perspective. Aiming at completeness for the business context, the dictionary of business process NFRs was extended with an attribute addressing regulatory, legislative and operational aspects involved in business process enactment. This NFR is *Compliance*, adapted from the types of NFRs for software systems presented by Sommerville [30].

The attributes in Table 1 relate to each other. For instance, the conversion from standard protocols to ensure compatibility may affect performance efficiency [21], while a fast maintenance implies in higher recoverability in the presence of errors, improving reliability levels. Identifying interdependence relationships between business process NFRs is relevant to recommend a more complete set of web service NFRs to be defined in SLAs. For example, when business areas define a constraint related to compatibility, the IT team could take care of performance NFRs as well. Figure 3 shows the interdependence relationships between the business process NFRs that were identified in this work.

Some relationships shown in Fig. 3, such as *performance efficiency* vs. *compatibility* and *reliability* vs. *maintainability*, were defined based on studies found in the literature [21,35]. Others, such as *reliability* vs. *security* and *compliance* vs. *security*, were defined via logical inference based on an empirical analysis by the authors of this work. Table 10 presents a brief explanation of the meaning of the interdependence relationships shown in Fig. 3.

Regarding the web service NFRs, the characteristics and sub-characteristics proposed in the ISO/IEC 25010 Product Quality Model [17] were considered describing technical aspects of web service provisioning to be defined in SLAs by IT teams. This model was refined by extra works on software quality models and QoS attributes and, as a result, the dictionary of web service NFRs is formed by 93 requirements, each of them related to at least one business process NFR. The dictionary of web service NFRs, with requirements definition, references, measurement unit and additional details regarding its elaboration.

The NFR decomposition framework is split into eight decomposition diagrams offering a better visualization of the relationships between the attributes. Figures 4, 5, 6, 7, 8, 9, 10, 11) show one diagram for each business process NFR (cf. Fig. 3).

Table 10. Details of interdependence relationships between business process NFRs.

NFR1	NFR2	Relationship explanation	Ref.
Perf. Effic.	Compatib.	The conversion from standard protocols to ensure compatibility between web services may affect performance efficiency	[21]
Perf. Effic.	Usability	The additional code and processing required to ease an operator's task or provide more usable output may affect performance efficiency	[21]
Perf. Effic.	Maintain	The need of using modularity, instrumentation and well commented high-level code to increase maintainability may affect performance efficiency	[21]
Perf. Effic.	Security	The additional code and processing required to control the access of a web service or data may affect performance efficiency	[21]
Perf. Effic.	Portability	Using direct code or optimized system software to increase performance may affect web service portability	[21]
Perf. Effic.	Reliability	The implementation of strategies to increase web service's availability may affect performance efficiency	—
Compatib.	Security	Coupled systems or web services can be accessed by different users, increasing the potential for accidental access of sensitive data and thus affecting security requirements	[21]
Compatib.	Portability	The guarantee of the web service's compatibility may affect portability requirements in terms of platform independence	–
Usability	Reliability + Security	The implementation of error prevention functions in a web service's interface may affect its maturity and stability in terms of fault detection and fault tolerance	[35]
Reliability	Maintain	Increasing maintainability usually affects a web service's reliability, as it turns easier for a web service to be maintained in case of breakdown	[35]
Reliability	Security	The security of a web service may affect its reliability in terms of stability and reputation	—
Maintain	Portability	Increasing maintainability can affect the effort to move a web service from one operating environment to another	[35]
Compliance	*	Regulatory, legislative or operational guidelines can be applied to all seven remaining NFRs included in the framework	–

For example, the NFRs shown in Table 2 are related to *performance efficiency*, with which 13 NFRs describing web service's time behavior, resource utilization and capacity are associated. Contrasting the dictionary of business process NFRs (cf. Table 1), the measurement unit is defined here for some web service NFRs. Synonyms from different references are grouped in a unique ID as for *Response time*, *Average and maximum response time* and *Execution duration*.

Figure 4 shows the decomposition diagram for *performance efficiency*. The NFRs and most of the relationships between them were extracted from works from the literature [3,16–18,23,24]. For instance, the relationships between the attributes related to *resource utilization*. Other relationships were mapped from related studies or defined via logical inference. The attributes *latency time*, *execution time* and *transaction time*, for instance, are associated with *response time* considering the definition of the latter: "response time is defined as the time required to complete a web service request" [18]. Thus, response time should include the round-trip delay for the network propagation (i.e., the latency time) plus the execution time required to process the request in the provider (i.e., the execution time). In addition, if transactions are processed, it should also consider the time to complete the transaction (i.e., the transaction time).

Figures 5 and 6 show the decomposition diagrams for *reliability* and *security*, respectively. As for reliability, 17 web service NFRs are presented, which are mainly related to web service availability and stability in the presence of failures. The NFRs and most of the relationships were extracted from [16,17,21,23,29,34]. For instance, the attributes *fault detection*, *failure resolution* and *mean time between failures* as associated with *maturity*. Other relationships were defined via logical inference, such as associating *fault tolerance*, *recoverability* and *successability* in meeting requests as attributes related to web service *stability*.

Regarding security, 12 web service NFRs are presented, which are mainly related to data confidentiality and integrity, access control and traceability. The NFRs and some relationships were extracted from works from the literature [16, 17,21,23–25]. Other relationships, such as the bidirectional association between *confidentiality* and *access control*, were defined via logical inference. For the latter, we considered that confidentiality requires that data should be read only by those with access to it, implying in access control. The association between *access control* and *integrity*, on the other hand, was proposed in the literature with a similar argument [21].

Figures 7 and 8 show the decomposition diagrams for *compliance* and *compatibility*, respectively. Regarding *compliance*, the NFRs were extracted from works from the literature [23,26,30,33] and all the relationships were defined via logical inference. On the other hand, the NFRs and most of the relationships for *compatibility* were extracted from works from the literature [16,17,21], as the attributes associated with *interoperability*. The attributes *supported standard* and *conformability* were identified by the authors as being related to both *compliance* and *compatibility* and hence are shown in both diagrams in orange color. The definition of this dual association considered a scenario where technical standards must be addressed in web service provisioning, as a demand of regulators, organizations or the government itself. When these standards are related to the communication between systems, this attribute may also be defined in terms of compatibility. Conformability is the degree to which the pre-defined standards are met and hence considered in both cases.

Figures 9, 10 and 11 show the decomposition diagrams for *portability*, *maintainability* and *usability*, respectively. The NFRs and relationships in these diagrams were mainly extracted from the literature [3,16,17,21–25,29]. The attributes *software system independence* and *machine independence* are related to both *maintainability* and *portability*, and shown in both diagrams in orange color. This dual classification considered that, the more a system is independent of the computational environment, the easier it is to adapt its operation to different environments and reuse its components in different contexts. Likewise, the attribute *self-descriptiveness* is classified as related to both *maintainability* and *usability*, considering that the clearer a web service and its documentation is, the easier it is to maintain and operate it.

4.3 Examples of NFR Decomposition

Using the illustrative example of business process presented in Fig. 1, some NFRs can be defined to the web services that will implement such a process.

Consider the BLA related to *performance efficiency* associated with the set of highlighted activities. Some levels for web service QoS attributes should be pursued in order to ensure that this BLA is met, and the decomposition framework proposed herein can be used for this purpose. Considering that for each activity one or more web services can be used, different QoS attributes can be defined for each web service, depending on the needs identified by the analysts involved. Per Fig. 4, to meet the BLA *performance efficiency* for these three activities, 13 distinct QoS attributes may be associated with the web services that will be used to implement them. For example, for some web services, QoS attributes related to *response time* or *throughput* may be defined, associated with some target values; i.e., the QoS levels. IT teams may also understand that, for some web services, they should use a more specific QoS attribute related to *latency time*, *execution time* or *transaction time*.

Besides the direct relationships addressed in the previous example, indirect QoS attributes can also be defined as they can also interfere in the performance efficiency of these three activities. For example, according to Fig. 3, *performance efficiency* is related to *reliability*. Thus, web services that implement one or more of the three activities in Fig. 1 may also have QoS attributes associated with *reliability*, as they may also affect the performance efficiency of the business process, as explained in Table 10. An example would be to define a QoS attribute related to *availability* (cf. Fig. 5) because the business process will be delayed if the web service is not available.

Still taking the example in Fig. 1, another BLA associated with the three activities being addressed is related to *security*. Using Fig. 6 as the main reference for the decomposition of security-related NFRs, 12 QoS attributes are suggested as ideas for the IT team to add in the corresponding SLAs. From these 12 QoS attributes, eight are leaf nodes, i.e., the most specific attributes, whereas four are intermediate nodes, i.e., more generic ones. Any attribute level can be used, depending on the needs perceived by the IT team. The business team should provide detailed information related to the security BLA, explaining exactly

what the requirement means, so IT teams can choose the most appropriate QoS attributes to be associated with the web services, such as *access control*, *encryption*, *auditability* and so on. The information from the business area is also relevant to allow identifying which web services will need QoS attributes or not. Other QoS attributes can be chosen by referring to Figs. 4, 5, 7, 8 and 11 as the attributes related to *performance efficiency*, *reliability*, *compliance*, *compatibility* and *usability* are indirectly related to *security* (cf. Fig. 3).

5 Related Work

Several studies have addressed business process automation with SOA. However, only a few of them discusses the relationship between business process NFRs and web service QoS attributes. Still in 2005, the particularities involved in using quality requirements originally defined as software quality models were discussed for SOA [24]. SOA and underlying concepts were explained in detail, examining their impact on meeting business goals in organizations. A structured list of QoS attributes used in SOA was provided.

In 2008, the ISO/IEC 9126 quality characteristics (ISO/ IEC 25000 predecessor) were investigated to develop web-based applications [35]. Eliciting information from stakeholders with an online survey, interactions between pairs of quality characteristics were identified, considering three possible relationships: positive, negative and independent. This approach enabled to understand how software quality aspects influence each other, contributing to elaborate a quality model that combines individual QoS attributes based on specific relationships.

In 2009, a related approach was proposed to address quality requirements expressed at the level of SOA applications and break them down to the level of components used to create the applications, i.e., at the web service level [1]. The structure used for decomposition considers two ontologies: *(i)* SQuaRE-based SOA Quality Ontology, with 14 high-level quality characteristics extracted from ISO/IEC 25000 SQuaRE quality model and is used by business users; and *(ii)* Semantic Web Service QoS Ontology, with a set of qualitative and quantitative QoS attributes related to web services. This approach supports a more direct decomposition of high-level QoS attributes into detailed preferences, in a task less dependent on an IT expert's knowledge.

Discussions for the business process level were introduced by showing the need for defining functional requirements and NFRs of web service provisioning in a different type of agreement, which is the aforementioned BLA [7].

The Strategic Alignment with BPM (StrAli-BPM) framework was proposed in 2013 to foster the strategic alignment between business and IT, in organizations executing business processes via web service orchestration [27,28]. StrAli-BPM was extended [5] and is formed by four components, one of which is particularly related to the work being presented herein – the BLA2SLA component. BLA2SLA considers a top-down strategy for a generic decomposition of business process NFRs (represented by BLAs) into web service QoS attributes (represented by SLAs). The definition of BLAs and SLAs is supported by metamodels, each including a set of attributes related to the business process and web

service levels, respectively. BLA2SLA enables the use of a standardized structure to define NFRs for web services based on business needs.

Many of the related work focuses on only discussing individual quality aspects of software components and web services, regardless of their relationship to NFRs of the business processes being automated. The BLA2SLA [27,28] component is closer to the aim sought in this work but using a loose relationship between the addressed concepts: business areas define a BLA, but IT teams only receive this BLA as a reference and must use their experience to define the SLAs considered more appropriate to address this BLA.

The decomposition model proposed by Abramowicz et al. [2] considers the relationships between high-level and low-level quality characteristics to semi-automate the derivation of QoS attributes. They assume NFRs defined from scratch by business users in the SOA development. This assumption contrasts with our work, which assumes that specific activities of business processes (to be automated through web services) are associated with business goals. Their proposed structure considers mainly product quality characteristics to be selected by business users, disregarding organizational and external requirements (i.e., regulatory and legislative) that might be demanded by business areas in software applications. For some of those characteristics, no related QoS attribute was mapped to be defined at the web service level. Ultimately, the relationships among high-level quality characteristics themselves were not considered, resulting in each characteristic being examined only individually.

In the approach of Zulzalil et al. [35], only the relationships among high-level quality characteristics themselves were explored, disregarding their relationship with web service QoS attributes. Finally, despite presenting an analysis of QoS attributes in SOA, O'Brien, Bass and Merson [24] did also not consider the relationships with business process NFRs.

6 Concluding Remarks

With this paper, we aimed to present a solution that systematically supports IT teams that need to define web service NFRs based on business process NFRs. Our solution is a conceptual framework that drives a breakdown of business process NFRs (which may or may not be formalized into BLAs) into web service NFRs (which may or may not be formalized into SLAs that group QoS attributes). The conceptual framework comprises one high-level decomposition diagram (for business process NFRs) and eight low-level decomposition diagrams (for web services). In addition, the nine decomposition diagrams are accompanied by an extensive dictionary with about 100 NFR attributes. This framework facilitates the decomposition of BLAs into SLAs and can be used to mitigate the dependence on specialized human knowledge about business-IT integration.

As future work, we would like to delve deeper into validating and adjusting the framework being proposed. This includes the relationships between the identified NFRs and an improved definition of each NFR attribute. First, we would like to extend the tool prototype presented in Salles et al. [27,28] to incorporate the decomposition diagrams proposed in this paper so that the interdependence relationships between NFRs are specifically addressed in the context of

the StrAli-BPM approach. In terms of validation, other possible actions that can be taken, individually or jointly, are through case studies and interviews with experts in organizations that may be interested in the proposed approach.

References

1. Abramowicz, W., Haniewicz, K., Hofman, R., Kaczmarek, M., Zyskowski, D.: Decomposition of SQuaRE-based requirements for the needs of SOA applications. In: International Conference on Advances in Communication Technologies and Engineering Science, pp. 81–94 (2009)
2. Abramowicz, W., Hofman, R., Suryn, W., Dominik, Z.: SQuaRE based web services quality model. In: International MultiConference of Engineers and Computer Scientists (2008)
3. Abramowicz, W., Kaczmarek, M., Zyskowski, D.: Duality in web services reliability. In: Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services, pp. 165.1–165.6 (2006)
4. Boehm, B.W., Brown, J.R., Kaspar, H., Lipow, M.: Characteristics of Software Quality. TRW Software Technology, Amsterdam (1978)
5. Borges, E.S., Fantinato, M., Aksu, U., Reijers, H.A., Thom, L.H.: Monitoring of non-functional requirements of business processes basedon quality of service attributes of web services. In: 21st International Conference on Enterprise Information Systems (ICEIS) (2019)
6. Botella, P., Burgués, X., Carvallo, J.P., Franch, X., Pastor, J.A., Quer, C.: Towards a quality model for the selection of ERP systems. In: Cechich, A., Piattini, M., Vallecillo, A. (eds.) Component-Based Software Quality. LNCS, vol. 2693, pp. 225–245. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45064-1_11
7. Bratanis, K., Dranidis, D., Simons, A.J.H.: Towards run-time monitoring of web services conformance to business-level agreements. In: 5th International Academic and Industrial Conference on Practice and Research Techniques, pp. 203–206 (2010)
8. Castro, C.F., Fantinato, M., Aksu, Ü., Reijers, H.A., Thom, L.H.: Towards a conceptual framework for decomposing non-functional requirements of business process into quality of service attributes. In: 21st International Conference on Enterprise Information Systems, pp. 481–492 (2019)
9. de Castro, C.F., Fantinato, M.: Dictionary of non-functional requirements of business process and web services. Technical report 003/2018, Graduate Program on Information Systems, University of São Paulo (2018)
10. Dromey, R.G.: Software product quality: theory, model, and practices. Technical report, Software Quality Institute, Griffith University, Brisbane, Australia (1999)
11. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. 2 edn. (2013)
12. Garcia, D.Z.G., de Toledo, M.B.F.: Quality of service management for web service compositions. In: 11th International Conference on Computer Science and Engineering, pp. 189–196 (2008)
13. Goel, N., Kumar, N.V.N., Shyamasundar, R.K.: SLA monitor: a system for dynamic monitoring of adaptive web services. In: 9th European Conference on Web Services, pp. 109–116 (2011)
14. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Q. **28**(1), 75–105 (2004)

15. IEEE: IEEE standard glossary of software engineering terminology, IEEE Std 610.12-1990 (1990)
16. ISO/IEC: ISO/IEC 9126 software product quality, IEEE Std 9126:2002 (2002)
17. ISO/IEC: ISO/IEC 25010 system and software quality models (2010)
18. Lee, K., Jeon, J., Lee, W., Jeong, S.H., Park, S.W.: QoS for web services: requirements and possible approaches. Technical report NOTE-ws-qos-20031125, W3C Korea Office (2003)
19. Leymann, F., Roller, D., Schmidt, M.T.: Web services and business process management. IBM Syst. J. **41**(2), 198–211 (2002)
20. Liu, Y., Ngu, A.H., Zeng, L.Z.: QoS computation and policing in dynamic web service selection. In: 13th International WWW Conference, pp. 66–73 (2004)
21. McCall, J.A., Richards, P.K., Walters, G.F.: Factors in software quality. volume-iii. preliminary handbook on software quality for an acquisiton manager. Technical report RADC-TR-77-369, Defense Technical Information Center (1977)
22. Miguel, J.P., Mauricio, D., Rodriguez, G.D.: A review of software quality models for the evaluation of software products. Int. J. Softw. Eng. Appl. **5**(6), 31–54 (2014)
23. OASIS: Quality model for web services (2005). http://www.oasis-open.org/committees/download.php/15910/WSQM-ver-2.0.doc
24. O'Brien, L., Bass, L., Merson, P.: Quality attributes and service-oriented architectures. Technical report CMU/SEI-2005-TN-014, Software Engineering Institute, Carnegie Mellon University (2005)
25. Pettersson, A.: Service-Oriented Architecture (SOA) Quality Attributes - A Research Model. Master's thesis, Department of Informatics, Lunds University, Sweden (2007)
26. Ran, S.: A model for web services discovery with QoS. ACM SIGecom Exch. **4**(1), 1–10 (2003)
27. Salles, G.M.B., Fantinato, M., de Albuquerque, J.P., Nishijima, M.: A contribution to organizational and operational strategic alignment: incorporating business level agreements into business process modeling. In: IEEE International Conference on Service Computing, pp. 17–24 (2013)
28. Salles, G.M.B., Fantinato, M., Barros, V.A., de Albuquerque, J.P.: Evaluation of the strali-bpm approach: strategic alignment with BPM using agreements in different levels. Int. J. Bus. Inf. Syst. **27**(4), 433–465 (2018)
29. Sheoran, K., Sangwan, O.P.: An insight of software quality models applied in predicting software quality attributes: a comparative analysis. In: 4th International Conference on Reliability, Infocom Technologies and Optimization, pp. 1–5 (2015)
30. Sommerville, I.: Software Engineering, 9th edn. Pearson, Addison-Wesley, London (2010)
31. Tomar, A.B., Thakare, V.M.: A systematic study of software quality models. Int. J. Softw. Eng. Appl. **2**(4), 61–70 (2011)
32. Weske, M.: Business Process Management: Concepts, Languages, Architectures. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28616-2
33. Yoon, S., Kim, D., Han, S.: WS-QDL containing static, dynamic, and statistical factors of web services quality. In: International Conference on Web Services, pp. 808–809 (2004)
34. Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J., Sheng, Q.Z.: Quality driven web services composition. In: 12th International Conference on WWW, pp. 411–421 (2003)
35. Zulzalil, H., Ghani, A.A.A., Selamat, M.H., Mahmod, R.: A case study to identify quality attributes relationships for web based applications. Int. J. Comput. Sci. Netw. Secur. **8**(11), 215–220 (2008)