A Long Way to Quality-Driven Pattern-Based Architecting

Gianantonio Me[1,2](✉), Coral Calero[1], and Patricia Lago[2]

[1] Universidad de Castilla La Mancha, Ciudad Real, Spain
{gianantonio.me,coral.calero}@uclm.es, g.me@vu.nl
[2] Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
p.lago@vu.nl

Abstract. The relation between architectural patterns (or styles) and quality attributes has been widely addressed in the literature. However, the knowledge is fragmented over a wide range of heterogeneous studies. Our aim is to build a systematic body of knowledge to support architectural decision-making and design. If available, this knowledge helps architects in addressing quality requirements consciously and more explicitly, i.e. in *quality-driven pattern-based design*. In order to build that body of knowledge we carried out a systematic literature review. We identified 99 primary studies for the analysis. The resulting data shows a wide spectrum of approaches encompassing patterns and quality attributes. In this study we (1a) present in which way patterns and quality attributes interact and (1b) provide quantitative data on the frequency of appearance for both patterns and quality attributes; (2) give an overview of the approaches we elicited from the analysis; and (3) provide our insights regarding a specific challenge (combination of patterns). Our analysis is a first step toward a theory on the architectural patterns and quality attribute interaction.

Keywords: Architectural patterns · Architectural styles · Quality attributes · Decision making

1 Introduction

Architectural patterns and styles are recurrent solutions to common problems. Among others, they include knowledge on quality attributes (QAs) [1]. For the sake of simplicity, throughout the paper we use the term architectural pattern to mean both. In fact, according to Buschmann [2], patterns and styles are very similar as every architectural style can be described as an architectural pattern. However, some differences can be considered as essential, the most relevant being that patterns are more problem oriented, while styles do not refer to a specific design situation [2]. Accordingly, in our analysis we make explicit if and why authors adopt the term pattern or style. We observe a similar problem with the definition of quality attribute. Again, for the sake of simplicity, we adopt the term quality attribute. In our analysis, if necessary, we make explicit the term

B. Tekinerdogan et al. (Eds.): ECSA 2016, LNCS 9839, pp. 39–54, 2016.
DOI: 10.1007/978-3-319-48992-6_3

used by the authors such as non-functional requirement, quality property, quality dimension, etc. Architectural patterns include knowledge on quality attributes. Architects rely on that knowledge for effective architectural decision-making. Increasing that knowledge means increasing the role of patterns in satisfying quality attributes. The aim of this paper is to present our results of the Systematic Literature Review (SLR), hence providing some conceptual building blocks on patterns and quality attributes interaction. Those conceptual blocks can be used for building a systematic theoretical framework. We also aim to encourage the discussion in the software architecture community.

Paper Overview: Section 2 offers a description of background knowledge and related work. Section 3 presents our study design. Section 4 presents our analysis and results, while Sect. 5 includes threats to validity. Section 6 summarizes conclusions and future work.

2 Background and Related Work

In the literature there are several works that, to various degrees, address the interaction between architectural patterns and quality attributes. Many have been included as primary studies of our SLR. In this section, we focus on two additional works, Buschmann [2] and Harrison and Avgeriou [1], holistic in nature and hence providing an excellent starting point for our SLR. Buschmann [2] is the cornerstone of architectural patterns and many later publications refer to its taxonomy of patterns. The approach is holistic. Firstly, software architecture design is considered more than a simple activity with a limited scope. Software architecture design has system-wide goals. Secondly, it aims at providing systematic support beyond that of a single pattern. As the title of the book suggests, patterns are framed in a *system of patterns*. For our purpose, we have considered the work of Buschmann in a pattern-quality interaction perspective, i.e. with a special focus on such interaction. In particular, the relationship between patterns and quality is based on a quality model that includes Changeability, Interoperability, Reliability, Efficiency, Testability and Reusability. Several quality attributes (called in [2] non-functional properties) present one or more sub-characteristics. Each quality attribute has been exemplified by means of scenarios. Some good fitting solutions (pattern-quality attribute) are given, for instance an example of fitting solution for changeability is the pattern Reflection. Trade-off and prioritization of quality properties have been mentioned. Non-functional properties can be classified according to the architectural techniques for their achievement. Patterns provide a support for building high-quality software system in a systematic way given some quality properties and functionalities. According to [2], the final assessment of quality properties in software architecture is still a difficult task. Indeed, although quality properties are crucial for the design, we still have to solve problems in their measurement. The lack of quantification makes the choice mostly based on the intuition and knowledge of software architects [2]. Similar to [2], Harrison and Avgeriou [1] holistically cover architectural patterns; differently, they propose to extend patterns with the knowledge about their impact on quality attributes: by knowing the consequences of adopting a certain pattern, architects would ideally make better-informed design decisions. The authors provide some evidence regarding the impact of patterns and quality attributes. Their ultimate goal is to organize a body of knowledge in a way that is accessible and informative for architects, in order to support an architectural decision making process. This goal is shared with ours. There are other studies that aimed to address the interaction between architectural patterns and quality attributes. For instance, Babar [3] focuses on the synergy between architectural patterns, quality attributes and scenarios. He provided a framework for collecting and representing the knowledge of that synergic interaction. His motivation is the lack of systematic knowledge about that synergy that might support software design. Babar proposes a valuable template, but he does not go beyond a methodological proposal, without bringing experimental evidence. In [4], Zdun focuses on pattern combinations. He proposes a pattern language grammar in order to keep track of patterns relationships. The formalized pattern language grammar has been considered also with effects to quality goals. This work fits with our purpose. However the level of analysis is on design patterns; as such it does not qualify as primary study for our SLR, because, we decided to explore only the highest level of abstraction (patterns and styles) excluding design and idiom level. This decision was necessary to scope the amount of information to a manageable size within a single SLR. The rationale is the amount of information would have been difficult to manage in a single systematic review study. Weyns [5] explains how patterns capture expert knowledge in the domain of multi-agent systems. The knowledge accumulated over years of practice and research has been represented by a pattern language. The interaction between patterns and quality attributes appear in a primary representation that includes quality attributes, constituent elements, responsibilities, interfaces explaining how elements have been used together and design rationale behind the architectural choices. Finally, Costa et al. [6] built a collection of scenarios useful for a particular architectural style evaluation. Such methodological approach can be extended to other patterns, or pattern combinations (system of patterns).

3 Study Design

Our systematic literature review has been carried out according to Kitchenham guidelines [7]. Few studies focus exactly on the interaction between quality attributes and architectural patterns. Therefore, we have decided to carry out this SLR with the motivation of detecting the widespread knowledge and build it in a systematic theoretical framework.

3.1 Research Questions

It is widely known that architectural patterns and quality attributes are not independent by implying (explicitly or not) significant interactions [8]. Such interactions can be represented as reusable knowledge elements. In this line of reasoning, for instance, Layered architecture presents a trade-off between efficiency and maintainability, where the second quality attribute is better fit [1]. Architects in search for assuring a high maintainability for their software architecture might take decisions on the basis of the knowledge reported above, and hence adopt a Layered pattern, but sacrificing something regarding efficiency. In this light, the knowledge on the interaction between patterns and quality attributes is a foundation for architectural decisions. Therefore, this study aims to assess if that type of reusable knowledge elements is widely accepted in the literature or if there are substantial differences in evaluating which pattern is more adequate for achieving specific quality attributes (QAs). Accordingly, we will address the following research questions (RQs):

- RQ1: What types of relations exist between architectural patterns and quality attributes?
  This research question has two goals. Firstly, it aims to explore the characterization of those relations (e.g., impact, dependencies, interaction, synergies or quantitative). Secondly, the type of relations can be evaluated and classified according to frequency of various patterns and QAs and related combinations.
- RQ2: What types of approaches address the relations between architectural patterns and quality attributes?
  This research question aims to understand and classify the various methodologies, frameworks, models, etc. available in the literature that addresses the relation between pattern and QAs.
- RQ3: What are the most important challenges for a quality-driven and pattern-based design?
  This research question aims to identify the most important challenges for building a theory of pattern-QAs interaction. We consider challenges as specific issues that emerge from the primary studies and for which better/explicit knowledge can help in addressing them better.

3.2 Data Sources and Search Strategy

Piloting the review protocol is essential [9]. We identified a set of 12 pilot studies. This set includes key studies we knew upfront as relevant and expected to find back in our systematic search, and at least one study on every single architectural pattern considered for the analysis. In this way it is possible to assess if the generic term architectural pattern can catch specific patterns (for instance *Layered*). Firstly, we have been collecting keywords for shaping the search string from the following studies. In case authors keywords were not available, we selected keywords by reading the abstract. Secondly, the search string has been tested on Google Scholar and other customary search engines in order to verify if the pilot studies would be detected. Our final search string is: (*architecture pattern* **OR** *architectural pattern* **OR** *architecture style* **OR** *architectural style*) **AND** (*quality attribute* **OR** *quality characteristic* **OR** *quality properties* **OR** *non-functional requirement* **OR** *no functional requirement* **OR** *quality dimension*).

3.3 Study Selection

We run the search string on the following search engines with the corresponding results: ACM Digital Library (422 studies); IEEE eXplore (129); SpringerLink (1395); Scopus (499); Web of Science (79) and Science Direct (418). The total was of 2942 hits collected in November 2015, covering a time span of 26 years (1990–2015). Subsequently we merged the hits in a reference manager database (Mendeley). SpringerLink has been analyzed on a spreadsheet due to some technical difficulties in importing references to Mendeley. The primary-study selection was organized in four rounds (Round 1: based on title and abstract; Round 2: skimming reading; Round 3: Full reading; Round 4: Snowballing. One level of Snowballing has been performed on the citations of the included studies). The Round 1 and Round 2 were aimed to clear the set of studies from out of scope works and duplicates. We started the Round 3 with 283 studies and we applied the Inclusion and Exclusion criteria. Inclusion and exclusion used the following criteria. Inclusion criteria are: (1) A study that offers knowledge elements on the interaction between at least one architectural patterns and at least one quality attribute; (2) A study that is carried out by either academics or practitioners; (3) A study that is written in English. Exclusion criteria are: (1) A study that does not provide directly or indirectly any description for the quality attributes taken into account; (2) A study that does not provide directly or indirectly any description for the architectural patterns taken into account; (3) A study that does not focus on architectural patterns of applications; (4) A study that focuses on pre-pattern or anti-patterns; (5) A study that the analysis is at design or idiom level. We focus only on a higher level of abstraction (patterns/styles) and (6) A study that is not available, or is a book or a workshop note. After applying inclusion and exclusion criteria through a skimming reading we had back 160 studies that where reduced to 88 after a whole reading. By snowballing we retrieved other additional 11 studies. So doing, study selection resulted in a total of 99 primary studies.

4 Analysis and Results

We extracted the data from all the primary studies by using a structural coding procedure. Structural coding captures a conceptual area of the research interest [10]. All the knowledge has been classified in four main categories: Decision-Making, Patterns, Quality Attributes and Patterns-QAs Interaction. We decided for four categories according to our previous work [11]. Subsequently, data analysis has been reported with a descriptive approach. Due to the extensive amount of knowledge gathered, for the focused scope of promoting the discussion in the community, in this work we present our most interesting preliminary results.

4.1 RQ1: On Pattern-QA Relations

As previously stated, RQ1 has a dual goal. First we want to uncover how the various studies characterize the interaction between patterns and quality attributes.

That interaction remains mostly undefined. Other main important characterizations are indirect by means of tactics and according to quantitative measures. In the first case the interaction between patterns and quality is supported by tactics. In the second case quantitative models shape the interaction according to specific QAs measures. Table 1 shows the summary of this analysis part. In particular, it shows the various ways of addressing the interaction between quality attributes and patterns.

Table 1. Interaction patterns-quality attributes

Number of studies per type	Type of interaction described	Total number of studies
35	Undetermined	35
12	Tactics	12
8	Measurability	8
7	Fitness & satisfaction	7
6	Interaction as knowledge	5
5	Trade-offs	5
3	Scenario-based; characterization of patterns with a QA	6
2	Functional-Non-functional; Markov Model; Views and Viewpoint; Materialization; Technical; Real world requirements (QAs)-Systems specification (Patterns);	12
1	Responsibilities; Appropriateness; Capturing; Actors and dependencies; Relationship; Representability; Softgoals; Problem space-Solution space interaction; Transformation	9

The highest frequency is the category Undetermined. In this case it was not possible to identify one specific type of interaction. Category Tactics provides an intermediate mechanism between quality attributes and patterns. Category Trade-offs focuses on specific techniques for comparing and assessing several quality attributes at the same time. Different types of interactions work at different levels of abstraction. For instance: while Scenario-based provides the (external) context for analyzing the interaction, Measurability addresses the (internal) quantification of such interaction. In other words, we may have types of interactions relating to the external context and others capturing the internal functioning of a certain system. A type of interaction pursues another goal: assessing the quantitative value of a quality attribute inside a specific system solution (pattern). The second goal of RQ1 is to single out how patterns and quality attributes interact as witnessed in the primary studies. Firstly we have

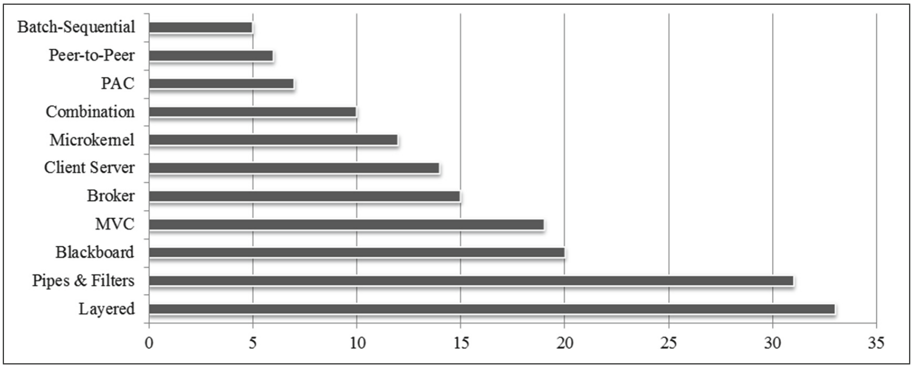

Fig. 1. Most frequent architectural patterns

gathered the frequency of both patterns and attributes. Figure 1 shows the patterns identified in the primary studies with the related frequency of appearance.

According to Fig. 1, the most frequent patterns (with a frequency of five or higher) are those 10 enlisted patterns plus a set of pattern combinations (see Combination). In fact, there are 44 additional patterns (not displayed in the figure) with frequency between 1 to 5. Among these less frequent patterns, multi-agent system patterns show a good potential for further research. Our online protocol provides the frequency table for the full list of patterns.

We performed a similar analysis about the found quality attributes, which provide a similar picture with an extended landscape of exotic quality attributes. In this case we have found 43 quality attributes (plus a residual category of not recognizable QAs) with a frequency mean of 15,6. Figure 2 includes only the quality attributes that appear at least 13 times in our primary studies. Like for patterns, the less frequent QAs are available in the online protocol[1]. Finally we have combined the two data pools (Patterns-QAs Frequency, see Table 2). In particular, we identified 711 couples pattern-QA. Of these, 422 (62 %) are couples composed by one of the most frequent patterns and one of the most frequent quality attributes. Interestingly, 166 couples out of 711 (23 %) are composed by one of the most frequent quality attributes listed in Fig. 2. Other combinations are much lower, for instance the couple "most frequent patterns-less frequent quality attributes" appears just in 62 cases (9 %) and expectably the couple "less frequent patterns-less frequent QAs" appears in even less cases (41, corresponding to 6 %).

Regarding the frequency of patterns and QAs, we observe that the set of most frequent quality attributes covers 85 % of all identified couples pattern-QA. Only 70 % of the identified couples are composed by a pattern belonging to the set of most frequent patterns. This might suggest that the set of most frequent QAs is mature enough to be considered as a backbone for an architectural quality model. On the other hand, patterns as a category is to be considered as

[1] www.s2group.cs.vu.nl/gianantonio-me/.

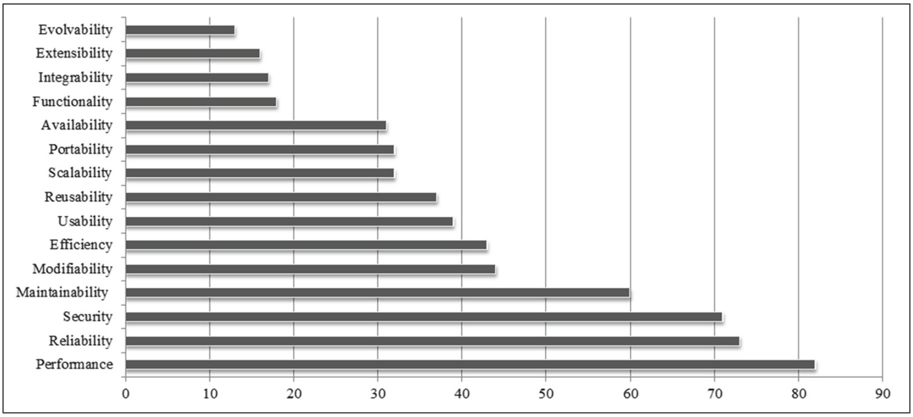

Fig. 2. Most frequent quality attributes

Table 2. Patterns and quality attributes combinations

Combinations	Most frequent QAs	Less frequent QAs	Total
Most frequent patterns	62 %	9 %	71 %
Less frequent patterns	23 %	6 %	29 %
Total	85 %	15 %	100 %

potentially unlimited: combinations of patterns or new patterns might be continuously created. This poses a challenge on how to capture the heterogeneity in a continuum, and represent it in a body of reusable knowledge.

4.2 What Do We Learn? (Answer to RQ1)

We identified several ways of characterizing patterns and QAs interaction. The highest frequency belongs to an Undetermined interaction, which means that the elements provided by the study were not clear or sufficient for defining the patterns-QAs interaction. One important mechanism for addressing quality in patterns, however, is the architectural tactic. Other studies focus on how to measure the interaction between patterns and quality, by offering quantitative knowledge for supporting architectural decisions. We also identified both a set of most frequent patterns and QAs. We discovered that most quality attributes frequent cover the large part of the couples pattern-QA we identified in the literature.

4.3 RQ2: On Classifying the Approaches

Table 3 provides an overview of the types of approaches identified in the 99 primary studies. We clustered the different approaches according to the characterizing elements for each study. For instance, Decision-Making approaches highlight the role of architectural decisions. Tactics might belong to the Decision-Making category because they highlight a specific mechanism for design decision-making. Studies that focus on quantitative modeling for prediction aim to support architectural decisions by measuring the QAs. Therefore, overlaps of approaches are very likely to arise.

Table 3. Types of approaches

Type of approach	Nr. of studies	Focus on
Decision-making	22	How to support architectural decisions: e.g. hierarchies of QAs for prioritizing decisions
Quantitative-prediction and/or formal model	16	Support of architectural choice with quantitative assessment of QAs Design Method 8 Holistic method, focus on the process of designing systems architecture
Knowledge based	8	Reusable and well-known knowledge on patterns and QAs
Evaluation method	7	The focus is on the process of evaluating architecture Pattern QA characterization 7 Patterns are characterized by a single QA (e.g. Security Broker)
Ontology-pattern language topology-taxonomy	6	The focus is on the description and definition both of patterns, or quality attributes (Taxonomies)
Specific domain method	6	Those studies focus on a specific context, e.g. multi-agent system
Views-scenario based	6	Those studies extract information on patterns-QAs interaction using scenarios as particular instance of the system
Quality driven method	5	Those studies consider the entire process of architecting as achieving quality
Business process-real world oriented	3	Those studies explore how patterns and QAs can effectively address specific real world challenges
Functionality oriented	3	Those studies explore the link between functionalities and quality
Technical method	2	Those studies explore the patterns and QAs in terms of how to capture technological complexity

Basically, each type of approach represents an element potentially common to other approaches. For instance, studies that focus on Business Processes and/or

Real World needs shed light on an intrinsic goal of the other methodologies, namely to design systems that match business processes needs.

We have identified 13 different types of approaches. However, overlaps are very common (e.g. knowledge-based and decision-making). A reason for this overlap is that the approaches are at different levels of abstraction. For instance, scenario-based approaches provide the space were patterns and quality will be assessed; functionality-oriented approaches zoom in how the considered pattern both satisfies functionality and quality, zooming in implementation level. Overall we noticed that Decision-Making elements are widespread in all the identified approaches. Many studies have the goal to provide support for decisions, so decision-making can be considered as a cross-characterizing element for all the methodologies. In the same line of reasoning, knowledge-based approaches present a body of reusable knowledge for adopting decisions. In general, we observe redundant elements proposed as new/different methodologies. Table 4 proposes a possible key of reading the holistic relation we uncovered in the 13 approaches we identified.

Table 4. Unified framework for pattern-quality based architecting

<table>
<tr><th>Type of approach</th><th colspan="2">Meaning</th></tr>
<tr><td>Decision-making</td><td colspan="2">Goal</td></tr>
<tr><td>(Pattern) quality driven method</td><td colspan="2">Rationale</td></tr>
<tr><td>Design method</td><td colspan="2">General framework</td></tr>
<tr><td>Evaluation method</td><td colspan="2">General framework</td></tr>
<tr><td>Specific domain method</td><td colspan="2">Context</td></tr>
<tr><td>Business process-real world oriented</td><td colspan="2">Context</td></tr>
<tr><td colspan="2">Knowledge-based</td><td>Support for decision-making</td></tr>
<tr><td rowspan="6">Knowledge-based contents</td><td>Quantitative-prediction formal model</td><td>Architectural knowledge element</td></tr>
<tr><td>Pattern QA characterization</td><td>Architectural knowledge element</td></tr>
<tr><td>Ontology-pattern language topology-taxonomy</td><td>Architecture description technique</td></tr>
<tr><td>Views- scenario based</td><td>Architectural evaluation technique</td></tr>
<tr><td>Functionality oriented</td><td>Architectural knowledge element</td></tr>
<tr><td>Technical method</td><td>Architectural implementation technique</td></tr>
</table>

The essence of architecting is taking decisions. Therefore, (effective) *Decision-Making* represents the main goal of the overall process of architecting. From top to bottom:

- We aim to a unified framework where the rationale behind *Decision-Making* is quality- and pattern-driven. There are two ways to organize decision-making in a methodological framework: evaluation and design method.
- *Evaluation methods* focus on assessing how and how much system architectures achieve quality.
- *Design methods* focus on the process of architecting, defining the system architecture.
- Those methodological frameworks are intertwined. Architecting is contextualized into a specific domain (see Specific Domain Method), a Business Process or in general into a Real World need.
- Concrete support for decision-making is provided by reusable knowledge. Knowledge-based approaches encompass several knowledge elements or techniques. They can be used in combination or in isolation, according to the needs of the system in focus. Knowledge-based contents does consider the type of interaction between patterns and quality attributes, but mostly implicitly.

4.4 What Do We Learn? (Answer to RQ2)

We identified 13 main types of approaches. Each of them is characterized by a specific element. We observed multiple overlaps of approaches. For instance, Decision-Making aspects can be identified in all other approaches, although they focus on other specific elements. Our proposal, rather than invent a new approach, is to unify in a holistic framework all the essential and shared elements widespread in several, apparently different, approaches. We offered a prototype of that holistic approach, by isolating and highlighting the characterizing aspects of each single approach.

4.5 RQ 3-Challenges: Combination of patterns

In looking for the interaction between architectural patterns and QAs it emerged that a quality-driven combination of architectural patterns is among the most important challenges in developing modern software systems. We zoomed into the effect that combining multiple patterns may have on the overall quality delivered by the combination. I.e., while individually two patterns may contribute (or hinder) a certain quality attribute, their combination might have a positive (or conflicting) impact on the same. "Combination of patterns" can find a place in our Unified Framework among the Knowledge-based elements. Interestingly enough, among our 99 primary studies, we found only 8 papers mentioning such a combination, as described in the following.

Background Works on Combination of Patterns. Study [1] considers the research on combination of patterns as a great challenge, considering the lack of knowledge we have on the interaction between combinations of architectural patterns and quality attributes. In [2] combinations of patterns are considered crucial: patterns do not operate in isolation. However, according to [2] a combination of patterns is not software architecture yet because more refinements are required. Finally in [4] the focus is also on pattern combinations. In order to keep track of patterns relationship a pattern language grammar has been proposed. The formalized pattern language grammar has been considered also with effects to quality goals. Relationships between patterns are also in [12]. However those last two studies work on design pattern level of abstraction that, at least for the moment, is out of scope for our research.

Examples of Combinations of Patterns. The best source of information for combination of patterns is [13]. The study offers a wide list of combinations of patterns and some quantitative data. Table 5 summarizes the knowledge on the interaction between combination of architectural patterns and quality attributes.

4.6 Combination of Patterns

Lee et al. [14] present a method for evaluating quality attributes. This uses conjoint analysis in order to quantify QAs preferences. It can be used in combination with the ATAM. In this study the decision of a Layered+ MVC architecture is the result of a composition of customers needs. The approach of [14] suggests a conceptual building block where combinations of patterns reflect the result of negotiation between stakeholders.

In [15] the authors provided a knowledge base for architecting wireless services. They propose a knowledge-based model with a service taxonomy, a reference architecture and basic services as backbone. Regarding combinations of patterns, in [15] the focus is on service sub-domain. Combinations of patterns are solutions to achieve quality attributes in specific sub-domain. They are applied to basic services and shape the reference architecture. The approach of [15] selects the Layered as a main pattern for building the software architecture. The rationale is in the type of quality attributes supported and the popularity among engineers. This study offers the conceptual idea that a combination of patterns can be classified according the main pattern.

In [17] the authors are aiming for an architectural pattern language for embedded middleware systems. The core architecture is a Layered+ Microkernel.

In [18] the authors proposed a framework for early estimation of energy consumption, according to particular architectural styles, in distributed software systems. In their experiment styles have been tested in isolation. Further, one combination of them has been assessed regarding energy consumption. The combination of patterns (called in the study hybrids) showed less energy consumption and overhead with the same amount of data shared respect to each single pattern. In that case the impact on the quality attributes is not merely addictive; indeed combining patterns reduces the energy consumption of a single pattern.

In [16] a full model for architectural patterns and tactics interaction has been analyzed, with the aim of linking strategic decisions (decisions that affect the

Table 5. Pattern combinations and quality attributes

N	Combination of patterns	Quality Attributes	Study	Approach and/or type of interaction according to Tables 1 and 2
1	Layered, model view controller	Performance, usability, availability, modifiability and security	[14]	Measurability; quantitative-prediction formal model
2	Model view controller, broker	Modifiability, interoperability and reusability	[15]	Interaction as knowledge; knowledge based
3	Layered, blackboard, presentation abstraction control	Interoperability, integrability, portability and modifiability	[15]	Idem
4	Pipes & Filters, presentation abstraction control	Simplicity and integrability	[15]	Idem
5	Broker, repository, layered	Performance (capacity, response time), reliability (availability and fault tolerance)	[16]	Tactics; decision-making
6	Layered+, microkernel	A wide set of QAs	[17]	Measurability; quantitative-prediction formal model
7	Public subscribe, client server	Energy efficiency	[18]	Undetermined; quantitative-prediction-formal model
8	Pipes & Filters, model view controller	Flexibility	[19]	Undetermined; evaluation method
9	Reflective blackboard	Performance, maintainability, manageability and reusability	[20]	Undetermined; specific domain

overall architecture) and tactics (clear-cut implementations that achieve specifically a quality attribute). Regarding combination of patterns the study shows a Broker combined with a Repository and a Layered. In the case study the overall level of performance has been augmented by the introduction of a new component. The new component allows Broker to bypass some Layers and this increases performance. Tactics for Fault Tolerance can be implemented in the Broker, without changing the overall structure. The valuable knowledge element from this study is that tactics can support pattern combinations.

Study [19] focuses on a specific software architecture style for applications performing distributed, asynchronous parallel processing of generic data streams. The combination of patterns here presented highlights data stream and user interactivity. This leads to increased flexibility. Unfortunately, the study does not provide enough information to support generalization about combinations of patterns. Finally, study [20] shows an interesting motivation for combination of patterns. The analysis is framed in the context of multi-agent systems. The combination of Reflection pattern and Blackboard allows effective separation of concerns, contributing to high manageability of several agents.

4.7 What Do We Learn? (Answer to RQ3)

In spite of its systematic nature, the SLR does not provide enough knowledge for building either univocal types of interactions between given couples pattern-QA or pattern combinations. Usually, if a given pattern addresses a particular QA positively that interaction would be replicated in the combination. Generally all the combinations reported above address QAs in the same way of each single pattern. That means, for instance, that a Layered pattern addresses positively Portability (according to [1]) also when Layered is combined with other patterns. Similarly, combination 3 [15] supports portability as well. The only clear (reported) exception regards Energy Efficiency, for which the QA measure seems better if the patterns are combined instead of implemented in isolation. More evidence is needed to confirm this result [21]. Finally, an interesting and promising research path is to consider combinations of patterns as specific design-solutions for real world problems.

5 Threats to Validity

As customary, for the analysis we followed a SLR protocol. However, there are potential threats to validity. Firstly, the search string might not catch all the relevant papers available. We mitigated this risk by adding a snowballing phase, checking references of primary studies. Secondly, the process of inclusion and exclusion criteria applications has been conducted by only one researcher. Thirdly, there are almost no studies that explicitly address the focus of our analysis. This means that the knowledge is widespread in a heterogeneous spectrum of studies. Relevant information might be hidden in studies not detectable by a sound search string. To cope with this issue we performed a pilot study for testing and refining the search string. Finally, the threats to validity for the analysis results and conclusions might be considered as a problem of generalization. Since we are in search of a theory, our results should be generalizable to different contexts. Our strategy mitigation for this issue has been the adoption of a coding procedure. However, the context specifications still represent an important challenge for this research work.

6 Conclusion

We performed a systematic literature review in order to shed light on the interaction between architectural patterns and quality attributes. We answered three main research questions. For the first research question we identified the ways of addressing the interaction between quality attributes and patterns. We discovered that relation remains mainly unexplored, with a high number of studies showing an Undetermined type of interaction. We also analyzed the frequency of recurring patterns and recurring quality attributes. The main finding was that the set of most frequent quality attributes covers 85 % of the identified couples patterns-QAs. We can conclude that the set of quality attributes we found can act as backbone for a quality model. The second research question was answered by identifying different types of approaches for addressing quality through architectural patterns. We observed redundancy and overlapping, so we described basic elements for a pattern-quality driven architecting and we unified them in a holistic framework. The third research question, about challenges in quality and patterns interaction, allowed us to explore combinations of patterns. We realized that we still lack extended knowledge on this specific challenge in particular. Overall, the knowledge gathered so far puts the basis for a further development of a theory for pattern-quality driven architecting. However, in spite of architectural patterns and quality attributes being both widely explored and practiced, there is still a lot to learn on their interaction—a long way to go.

Acknowledgement. This work is part of the GINSENG (TIN2015-70259-C2-1-R) project (funded by the Spanish Ministerio de Economía y Competitividad and by FEDER-Fondo Europeo de Desarrollo Regional) and by VILMA (PEII11-0316-2878) project (funded by the Junta de Comunidades de Castilla-La Mancha and by FEDER-Fondo Europeo de Desarrollo Regional).

References

1. Harrison, N.B., Avgeriou, P.: Leveraging architecture patterns to satisfy quality attributes. In: Oquendo, F. (ed.) ECSA 2007. LNCS, vol. 4758, pp. 263–270. Springer, Heidelberg (2007). doi:10.1007/978-3-540-75132-8_21
2. Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M.: A System of Patterns. Wiley, Hoboken (1996)
3. Babar, M.A.: Scenarios, quality attributes, and patterns: capturing and using their synergistic relationships for product line architectures. In: Software Engineering Conference, 11th Asia-Pacific, pp. 574–578. IEEE (2004)
4. Zdun, U.: Systematic pattern selection using pattern language grammars and design space analysis. Softw.-Pract. Exp. **37**(9), 983 (2007)
5. Weyns, D.: Capturing expertise in multi-agent system engineering with architectural patterns. In: Weyns, D. (ed.) Architecture-Based Design of Multi-Agent Systems, pp. 27–53. Springer, Heidelberg (2010)
6. Costa, B., Pires, P.F., Delicato, F.C., Merson, P.: Evaluating rest architectures? Approach, tooling and guidelines. J. Syst. Softw. **112**, 156–180 (2016)

7. Kitchenham, B., Charters, S.: Procedures for performing systematic literature reviews in software engineering. Keele University & Durham University, UK (2007)
8. Harrison, N.B., Avgeriou, P.: How do architecture patterns and tactics interact? A model and annotation. J. Syst. Softw. **83**(10), 1735–1758 (2010)
9. Brereton, P., Kitchenham, B.A., Budgen, D., Turner, M., Khalil, M.: Lessons from applying the systematic literature review process within the software engineering domain. J. Syst. Softw. **80**(4), 571–583 (2007)
10. Saldaña, J.: The Coding Manual for Qualitative Researchers. Sage, Thousand Oaks (2015)
11. Me, G., Calero, C., Lago, P.: Architectural patterns and quality attribute interaction. In: Working IEEE/IFIP Conference on Qualitative Reasoning about Software Architectures (QRASA), WICSA 2016. IEEE (2016)
12. Zimmer, W., et al.: Relationships between design patterns. Pattern Lang. Prog. Des. **57** (1995)
13. Harrison, N.B., Avgeriou, P.: Analysis of architecture pattern usage in legacy system architecture documentation. In: Seventh Working IEEE/IFIP Conference on Software Architecture, WICSA 2008, pp. 147–156. IEEE (2008)
14. Lee, K.C., Choi, H.-J., Lee, D.H., Kang, S.: Quantitative measurement of quality attribute preferences using conjoint analysis. In: Gilroy, S.W., Harrison, M.D. (eds.) DSV-IS 2005. LNCS, vol. 3941, pp. 213–224. Springer, Heidelberg (2006). doi:10.1007/11752707_18
15. Niemela, E., Kalaoja, J., Lago, P.: Toward an architectural knowledge base for wireless service engineering. IEEE Trans. Softw. Eng. **31**(5), 361–379 (2005)
16. Harrison, N.B., Avgeriou, P., Zdun, U.: On the impact of fault tolerance tactics on architecture patterns. In: Proceedings of the 2nd International Workshop on Software Engineering for Resilient Systems, pp. 12–21. ACM (2010)
17. Wu, C., Chang, E.: Comparison of web service architectures based on architecture quality properties. In: 2005 3rd IEEE International Conference on Industrial Informatics, INDIN 2005, pp. 746–755. IEEE (2005)
18. Seo, C., Edwards, G., Malek, S., Medvidovic, N.: A framework for estimating the impact of a distributed software system's architectural style on its energy consumption. In: Seventh Working IEEE/IFIP Conference on Software Architecture, WICSA 2008, pp. 277–280. IEEE (2008)
19. Francois, A.R.: A hybrid architectural style for distributed parallel processing of generic data streams. In: Proceedings of the 26th International Conference on Software Engineering, pp. 367–376. IEEE Computer Society (2004)
20. Silva, O., Garcia, A., Lucena, C.: The reflective blackboard pattern: architecting large multi-agent systems. In: Garcia, A., Lucena, C., Zambonelli, F., Omicini, A., Castro, J. (eds.) SELMAS 2002. LNCS, vol. 2603, pp. 73–93. Springer, Heidelberg (2003). doi:10.1007/3-540-35828-5_5
21. Procaccianti, G., Lago, P., Vetrò, A., Fernández, D.M., Wieringa, R.: The green lab: experimentation in software energy efficiency. In: Proceedings of the 37th International Conference on Software Engineering-Volume 2, pp. 941–942. IEEE Press (2015)