Student Research Abstract: Fuzzy ontology-driven web-based framework for supporting architectural design

Francesco Nocera
Polytechnic University of Bari
Electrical & Information Engineering Department
Via E. Orabona,4 - Bari - Italy
Tel:+ 39 080 596 3641
francesco.nocera@poliba.it

1. PROBLEM AND MOTIVATION

Since 2005 when Anton Jansen and Jan Bosch [3] gave a modern definition of Software Architecture as a composition of a set of explicit design decisions, a new perspective concerning software architectures design decisions, quality and goals evaluations have been dominating the scientific literature in this field. Designing the software architecture of non-trivial systems belonging to several application domains, namely industrial automation, defense telecommunication, financial services, and so on, is not an easy task, and requires highly skilled and experienced people. Beyond these, new challenges in the design and in architectural models are derived from self-managing and self-adaptive capabilities that are typical of many modern and emerging software systems, including the industrial Internet of Things, cyber-physical systems, cloud computing, and mobile computing. The satisfaction of quality requirements and the appropriate options for future changes are among the major goals of software architectures. In defining and modeling software architecture through patterns, a challenging issue is also concerned with the number of different available decisions depending on the fact that patterns can cooperate, are composable, are complementary or exclusive with respect to a given problem.
To solve the challenging problem of choosing a set of patterns, some structures have been proposed, supported by pattern languages with a given syntax and style.

2. BACKGROUND AND RELATED WORK

Non-functional requirements (NFRs) play a crucial role in software development also as available choices in decision making procedures for architectural solutions. Several quality models and taxonomy are available to classify and characterize quality requirements (e.g. Furps model, ISO 9126 and 25010). Anyway these quality models and taxonomy do not provide a formal representation of semantic relationship between quality attributes. Moreover, an explicit definition of their treatment in software modelling and development is needed. In self-adaptable models but also in classical software architectures, the link between architecture and design-time features modeling and the relationship between non-functional requirements, patterns and design decisions should be made more flexible.
The knowledge on NFRs is usually owned by designers and not formalized in a structured way. The idea is to provide an approach to more faithfully reproduce the existing relationships in order to formalize the extent to which they are guaranteed in the design of software architecture. Design patterns are interrelated in families and grouped for problem areas. Each problem area addresses a specific topic related to build distributed systems and, specially intrinsically addresses some specific non-functional requirements. Hence by considering a pattern in a given family we state that the pattern could have given NFRs.
Several surveys are available in the literature regarding the support software engineering development applications. An automatic optimization process for adaptation space exploration of service-oriented applications based on trade-offs between functional and extra-functional requirements is proposed in [6]. Kruchten models architectural design decision in software-intensive systems [5] using an ontology to model design decision having a specified category and attachments or relationships with other design decisions. The ontology is supported by a tool able of listing decision and relations, visualizing design structure and temporal view of design decisions. Kampffmeyer et al. propose DPIO, an ontology to formalize relationships among *Gang of Four's* (GoF) design patterns in [4], where the intent of design pattern is formalized and the ontology is used to suggest a pattern to solve a given design problem. Extended version of DPIO ontology to suggest patterns to solve integration problems is in [2]. Formalization of web design pattern is modeled in [7] using ontologies.

3. APPROCH AND UNIQUENESS

In order to encode all the information related to NFRs, patterns and corresponding families, a formalization via a Fuzzy Description Logic (DL) for the Semantic Web is proposed for modeling *(i)* relationships and sequences of patterns and *(ii)* the taxonomy that relates patterns with ensured NFRs

SAC 2016, April 04-08, 2016, Pisa, Italy
ACM 978-1-4503-3739-7/16/04.
http://dx.doi.org/10.1145/2851613.2852014

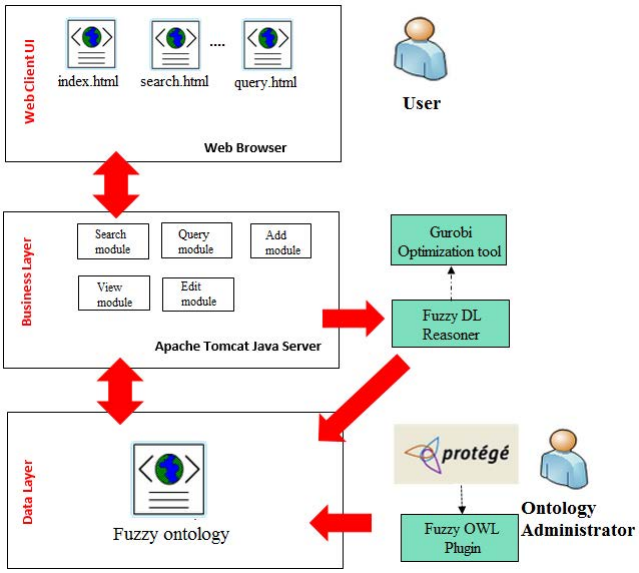

Figure 1: Model of proposed framework, and linked components.

for given application contexts. The idea is to develop a Fuzzy ontology-driven web-based framework and define a novel reasoning task to solve the following task: *"Given a set of non functional requirements $R = \{r_1,...., r_n\}$, retrieve the minimal subset of Patterns that better satisfies them."* The fuzzy version of DLs is needed in order to represent both ontological relations and factual ones. In the former case we may model that *portability and adaptability are directly proportionate* while *stability and adaptability are inversely proportionate*. For the latter we may represent that *the Adapter pattern has a high portability*. We see that the combination of ontological and factual knowledge allows a reasoning procedure to infer new information about a pattern. With reference to the previous example, we may infer that *the Adapter pattern has a high adaptability and a low stability*. We are also allowed to formally define that *a high adaptability implies a medium maintainability*. This means that we prefer patterns with a high value of a specific functional requirement r_i to those with a medium or low one. If there is no pattern with high value for r_i, than we prefer patterns with a medium value to those with a low one. In such patterns, fuzziness is evident: terms such as high, medium and low can be defined in terms of fuzzy sets. This is an ongoing work and it will be researched to what extent these ontological relations can be used. In the overall architecture and linked components proposed (see Figure 1), ontology is used as knowledge container to encode all the Fuzzy DL statements described. The ontology can be built using Protégé [1] with a Fuzzy OWL Plugin to build Fuzzy OWL ontologies (see Figure 2 for ontology model). To effectively deal with fuzzy ontologies in practice, we need a fuzzy DL reasoner [1]. The reasoning algorithm uses a combination of a tableaux algorithm and a Mixed Integer Linear Programming (MILP) optimization problem thanks to

[1]http://protege.stanford.edu/

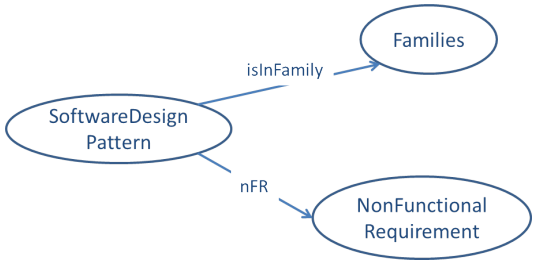

Figure 2: A graphical representation of the high-level ontology proposed.

Gurobi optimization tool [2]. The framework supports the following main functionalities: Insertion of a new pattern or a new NFR, insertion of fuzzy DL statements (relations intercurring between NFRs), editing of ontology individuals (Pattern or NFR), Check consistency. A query module implements the novel reasoning task previously suggested.

4. RESULTS AND CONTRIBUTIONS

The main contribution of this student research abstract is the presentation of a framework to represent and reason on mutual relations between NFRs and design patterns. The declarative proposed approach makes possible to maintain the above mentioned knowledge by keeping the flexibility and fuzziness of modeling thanks to the use of fuzzy concepts as high, low, fair.

5. REFERENCES

[1] O. Bobillo and U. Straccia. FuzzyDL: An expressive fuzzy description logic reasoner. In *In Proc. FUZZ-IEEE-2008. IEEE Computer Society*, 2008.

[2] D. Harb, C. Bouhours, and H. Leblanc. Using an ontology to suggest software design patterns integration. In *Models in Software Engineering*, pages 318–331. Springer, 2009.

[3] A. Jansen and J. Bosch. Software architecture as a set of architectural design decisions. In *Software Architecture, 2005. WICSA 2005. 5th Working IEEE/IFIP Conference on*, pages 109–120. IEEE, 2005.

[4] H. Kampffmeyer and S. Zschaler. Finding the pattern you need: The design pattern intent ontology. In *Model Driven Engineering Languages and Systems*, pages 211–225. Springer, 2007.

[5] P. Kruchten. An ontology of architectural design decisions in software intensive systems. In *2nd Groningen Workshop on Software Variability*, pages 54–61. Groningen, The Netherlands, 2004.

[6] R. Mirandola, P. Potena, and P. Scandurra. Adaptation space exploration for service-oriented applications. *Sci. Comput. Program.*, 80:356–384, 2014.

[7] S. Montero, P. Díaz, and I. Aedo. Formalization of web design patterns using ontologies. In *Advances in Web Intelligence*, pages 179–188. Springer, 2003.

[2]The math programming library focused on MILP, MIQP, LP and QP, is available at www.gurobi.com.