Journal of King Saud University – Computer and Information Sciences xxx (2017) xxx–xxx

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

An UML profile for representing real-time design patterns

Hela Marouane [a,b,*], Claude Duvallet [a], Achraf Makni [b], Rafik Bouaziz [b], Bruno Sadeg [a]

[a] LITIS laboratory, University of Le Havre, France
[b] MIRACL laboratory, University of Sfax, Tunisia

ARTICLE INFO

Article history:
Received 15 November 2016
Revised 20 June 2017
Accepted 25 June 2017
Available online xxxx

Keywords:
UML profile
Design patterns
Object Constraint Language
Real-time database

ABSTRACT

Systems which manipulate important volumes of data need to be managed with Real-Time (RT) databases. These systems are subject to several temporal constraints related to data and to transactions. Thus, their design remains a complex task. To remedy this complexity, it is necessary to integrate design methods to support data and transactions temporal constraints. Among the design methods, those based on patterns have been widely used in several fields. However, despite their advantages, these patterns present some shortcomings. Indeed, they do not manage efficiently the patterns variability and they do not specify the pattern elements when they are instantiated. To overcome these limitations, we propose, in this paper, a new UML profile to (i) express the variability in patterns and (ii) to identify the pattern elements in its instance. Besides, in order to well-capture the knowledge of the domain, the proposed profile extends UML with concepts related to real-time databases and integrates OCL (Object Constraint Language) to enforce the variation points consistency. Finally, we give an example of a RT pattern that illustrates these UML extensions, where we implement the proposed profile and we validate the pattern diagrams using the constraints we have proposed.

© 2017 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

1. Introduction

Recently, many Real-Time (RT) systems (*e.g.* driver assistance systems and control traffic systems) need to store large amounts of data and to process them in order to operate efficiently. Therefore, an RT database should be used. This database must have not only have the same features of conventional databases (*e.g.,* efficient management of accesses to structured data), but requires also efficient management of data and transactions timing constraints (Ramamritham, 1993). The design methods already proposed for traditional databases cannot be directly applied to model RT database applications since there is no mechanism to deal with the representation of time constraints. Besides, RT database applications become more complex, leading to an extensive conceptual description and to a prohibitive complexity from the practical point of view. Therefore, the design of these applications is a hard process which requires the development of new design methods to support both data structures and the dynamic behavior of RT applications, based on RT databases. In order to successfully design such applications, we believe that a powerful design method (*e.g.* design patterns (Gamma et al., 1995)) may improve the quality of development process.

The design patterns are reusable abstract design elements that can reduce the difficulty of systems modeling. These patterns present mechanisms which successfully capture and promote best practices in the software design. However, despite the benefits of design patterns, the designer may spend a lot of time to understand and to instantiate them. Thus, many researchers proposed several notations in order to facilitate the specification of the patterns and the documentation of their instances. These notations facilitate the understanding of complex concepts. There exist in the litterature different notations for documenting the patterns. For instance, we can mention the natural languages, which are imprecise and too ambigus, and the visual languages (*e.g.* the Unified Modeling Language (UML)). UML language is a standard object-oriented modeling language for general-purpose software. It is a commonly used language for visualizing, specifying and cumenting artifacts of systems by providing a precise semantics of its concepts (Rumbaugh et al., 1999). It provides a set of graphical notations to capture different aspects of the developed system.

* Corresponding author at: MIRACL laboratory, University of Sfax, Tunisia.
E-mail addresses: marouane.hela@gmail.com (H. Marouane), claude.duvallet@univ-lehavre.fr (C. Duvallet), Achraf.Makni@fsegs.rnu.tn (A. Makni), Raf.bouaziz@fsegs.rnu.tn (R. Bouaziz), bruno.sadeg@univ-lehavre.fr (B. Sadeg).

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2017.06.005
1319-1578/© 2017 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Please cite this article in press as: Marouane, H., et al. An UML profile for representing real-time design patterns. Journal of King Saud University – Computer and Information Sciences (2017), http://dx.doi.org/10.1016/j.jksuci.2017.06.005

Among the benefits, these notations (such as diagrams and icons) bring are (i) enhancing the communication between designers (facilitating communication between all participants in the development process), (ii) making easier the understanding of concepts and models. Indeed, it is easier to learn the means to draw diagrams than how to write text, because the formers are more significant and more concrete than text written. Finally, these notations can help people to grasp a lot of information more quickly than written text.

Despite their benefits, graphical notations have some shortcomings. Indeed, they lack clarity and expressive power. In addition, they are sometimes imprecise and ambiguous since they do not offer a well-defined semantics for different UML concepts. For example, UML parameterized collaborations are too limited (they do not explain precisely how to specify patterns with their usage (Sunyé et al., 2000)). Furthermore, in the instantiation level, the graphical notations do not identify accurately the pattern elements and their roles in the pattern. Moreover, the notations are not expressive enough to model specific domains. Indeed, in a particular domain design, UML should take into account the domain specificities. For instance, if we consider RT databases applications, we find that these applications are complex and they have several details that should be considered by the UML notations. They must specify RT database features, such as the time-constrained data and transactions. To overcome the shortcomings of the graphical notations, UML defines a new package, named *Profile*, to extend its syntax and its semantics. This profile provides three extension mechanisms with specific names in order to annotate UML diagrams with quantitative information:

- ≪stereotype-name≫: a stereotype which allows the definition of extensions to the UML vocabulary. It is possible to associate a *tagged value* and constraints to a stereotype.
- A tagged value: which is an attribute associated to a modeling element in order to extend its properties with a certain kind of details.
- A constraint: which is a semantic restriction added to a model element. Usually, constraints are written in OCL (OMG, 2003).

The definition of a profile is very important as it allows adding semantics as well as constraints to UML concepts (*e.g.,* classes, attributes and lifelines). Besides, it provides new vocabulary for a particular domain by giving specific notations related to it. Thus, we define, in this paper, a new profile, which represents a specialized variant of the UML 2.1.2. This profile provides UML extensions to support RT database requirements. In addition, it aims at extending UML with concepts in accordance with design patterns representation. The development of this profile has three motivations:

1. It represents the RT database applications concepts. A RT database has two main features: the notions of (i) data temporal consistency and (ii) the transactions RT constraints (Ramamritham and Pu, 1995). Some of its data must not only be logically consistent, but also temporally consistent, *i.e.* a data must be used during its validity interval and two correlated data must be used within their relative validity interval (a certain temporal window). RT data are classified into: (i) sensor data collected from sensors, and (ii) derived data calculated using the sensor data (Amirijoo et al., 2006). RT data are updated through update transactions which can be executed either periodically (to update sensor data) or sporadically (to update derived data). Therefore, we can deduce that RT databases have their own specificities. Thus, their design need to have appropriate concepts in order to consider factors, such as sensor data, derived data, the quality of data management, temporal semantics of transactions and concurrency control mechanisms in order to meet the timing constraints of RT applications (Idoudi et al., 2008). For this reason, in our work, we define UML extensions to take into account all these concepts.
2. It represents the patterns at the specification level. At this level, our profile is beneficial, as: (i) it offers flexible patterns that allow distinguishing between the fundamental elements and the variable elements, and (ii) it facilitates the comprehension of patterns instantiation.
   It expresses the patterns at the instantiation level. At this level, our profile has two advantages: (i) it ensures the traceability of patterns elements since it identifies clearly the elements belonging to each pattern, and (ii) it avoids ambiguity when composing patterns by identifying the role played by each pattern element.

Moreover, the proposed profile includes OCL constraints that ensure the design patterns diagrams consistency and correctness, *i.e.* the diagram respects all constraints, specified by the designer. In this paper, we focus on the intra-diagram consistency (for a given UML diagram, we check the consistency between its elements). For this end, we propose OCL constraints to deal with the dependence of variable elements in each pattern diagram.

Furthermore, we evaluate the proposed profile based on some criteria and we compare it with other existing profiles. Besides, we illustrate our profile through a RT design pattern defined in Marouane et al. (2012). We also implement the proposed profile using MagicDraw UML tool and we incorporate it into the design pattern diagrams and their instances. After that, we verify if the elements of these diagrams are in accordance with the OCL constraints.

The remainder of this paper is organized as follows. Section 2 overviews recent proposed UML profiles. Section 3 describes our UML profile. This section defines also a set of well-formed rules written in OCL in order to verify the patterns consistency and correctness. Section 4 describes the case study methodology that shows how we have organized and conducted our research. Section 5 illustrates the profile, using a RT sensing pattern defined in Marouane et al. (2012). It gives also two examples of system models instantiating the pattern. Section 6 depicts the implementation of the proposed profile and the verification of the OCL rules on the UML diagrams of the pattern. In Section 7, we give some concluding remarks and outline future work.

2. Related work

2.1. UML Profiles for patterns representation

Several UML profiles have been proposed in the literature to represent design patterns. They can be classified into three categories. The first one reveals design patterns at the specification level (*e.g.,* the profile proposed in Arnaud et al. (2008)). The second category proposes extensions to show patterns at the instantiation level (*e.g.,* the profiles proposed in Dong et al. (2007) and Loo et al. (2012)). In the third category, the extensions are proposed to present patterns at both the two previously-mentioned levels (*e.g.,* the profiles proposed in Reinhartz-Berger and Sturm (2009) and Rekhis et al. (2010)). These UML profiles are evaluated according to a set of criteria for patterns specification (Table 1) and patterns instantiation (Table 2). The criteria used are those defined in Rekhis et al. (2010) (variability, consistency, expressivity, composition and traceability), together with the completeness of the patterns solution.

Please cite this article in press as: Marouane, H., et al. An UML profile for representing real-time design patterns. Journal of King Saud University – Computer and Information Sciences (2017), http://dx.doi.org/10.1016/j.jksuci.2017.06.005

Download English Version:

https://daneshyari.com/en/article/10225928

Download Persian Version:

https://daneshyari.com/article/10225928

Daneshyari.com