PROMOpedia

A web-content management-based encyclopedia of software property models

Séverine Sentilles
School of Innovation, Design and Engineering, Mälardalen University
Västerås, Sweden
severine.sentilles@mdh.se

Federico Ciccozzi
School of Innovation, Design and Engineering, Mälardalen University
Västerås, Sweden
federico.ciccozzi@mdh.se

Efi Papatheocharous
Software and Systems Engineering, RISE SICS
Stockholm, Sweden
efi.papatheocharous@ri.se

ABSTRACT

The way software properties are defined, described, and measured, is different across different domains. When addressing these properties, several challenges commonly emerge, among which: synonymity, polysemy, paronymy, incomplete and inconsistent specification. In this paper we introduce PROMOpedia, an online encyclopedia, to tackle these challenges. PROMOpedia uses a web-content management system coupled with crowd-sourcing of scientific contents related to properties and their evaluation methods. The core concepts of PROMOpedia are built upon a property models ontology previously proposed by the authors, and is intended to target the needs of both researchers and practitioners.
Website: http://www.mrtc.mdh.se/promopedia/

ACM Reference Format:
Séverine Sentilles, Federico Ciccozzi, and Efi Papatheocharous. 2018. PROMOpedia: A web-content management-based encyclopedia of software property models. In *ICSE '18 Companion: 40th International Conference on Software Engineering , May 27-June 3, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3183440.3183482

1 INTRODUCTION

The increasing prevalence of software in today's society calls for the ability to develop high quality software at a faster pace. Most existing solutions in software engineering have been quite successful at taming software complexity as far as the sole functionality is concerned. However, the situation is not as satisfactory when quality is concerned. Software quality, embodied through properties, is lagging behind. In [13], Weyuker urges the software engineering community to start focusing more on properties, be they referred to as extra-functional properties, non-functional properties, quality properties, -ilities or just metrics.

Nonetheless, a lot of research on extra-functional properties (EFPs) already exists. A quick search on Google Scholar with the aforementioned terms returns several thousands hits. Unfortunately, they are scattered across many communities, domains and forums, encompassing areas such as testing, requirements engineering, maintainability, decision-making, object-oriented programming, component-based software engineering, model-driven engineering, web-services, software architecture, and many more. This creates many isolated islands of knowledge with limited opportunities for cross-fertilization.

ICSE '18 Companion, May 27-June 3, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5663-3/18/05.
https://doi.org/10.1145/3183440.3183482

As a result, there is currently no unique taxonomy of EFPs [4]. Several classifications and standards do exist but they are typically non-orthogonal, incompatible among themselves, and typically neglect specific characteristics of the properties and the associated evaluation methods [9]. This lack of widely acknowledged and centralized reference leads to loose and improper terminology usage [2, 5, 12], uncontrolled and subjective measurement activities and, more notably, to the following issues which obfuscate the state-of-the-art and practice even further:

(1) *Synonymous EFPs.* One property can be found under many different denominations, which are in fact completely equivalent (e.g., *"Class Coupling"*, *"Coupling between Objects"* [11]).
(2) *Polysemous EFPs.* Two properties can have the same name but refer to two completely different things. The difference can be in semantics, value representation, usage, etc. For example, the *"DC metrics"* in maintainability can refer either to the *"Degree of Cohesion"* or *"Descendants Count"* [11].
(3) *Paronymous EFPs.* Two properties can have the same etymological root, but different meanings (e.g., "composability" and "compositionality", which both derive from "composition", but imply different composition qualities of a component assembly).
(4) *Incomplete and inconsistent specification.* The methods used to evaluate the value of a property may not clearly specify what they measure and how. For example, ambiguous ways to measure software size exist, using the physical Source Lines of Code (SLOC) or the logical SLOC. Even though the definitions of the latter vary, SLOC are stated without a clear definition, and depending on the specific use they might include comments and bracket lines, or not [8].

In this paper, we aim at tackling the lack of a centralized EFP reference by introducing an early prototype of PROMOpedia. PROMOpedia uses a web-content management system coupled with crowd-sourcing of scientific contents related to properties and their evaluation methods. The core concepts leverage the property models ontology (PMO) proposed by Sentilles et al. [12].

Such a tool can play an essential role for both researchers and practitioners. For researchers, it aims at: (i) providing a tool to help them to more systematically formulate the domain knowledge; (ii) fostering on-line communities centred around dedicated properties; and (iii) closing the gap between academic research and industry by making new evaluation methods quickly and widely available for practitioners. For practitioners, it provides a centralized reference that allows them to: (i) browse available properties and evaluation methods to identify the most suited ones for their context; (ii) be automatically informed of new evaluation methods for a property of interest.

The remainder of the paper is organized as follows. In section 2 we introduce PROMOpedia with its core concepts, development concerns and intended users. Its realization with Drupal is described

in section 3, while the validation strategy is outlined in section 4. The contribution brought by the webserver is contextualized and compared to related works in section 5. The paper is concluded with section 6.

2 PROMOPEDIA

The core of PROMOpedia is built upon the property models ontology (PMO) [12] and its key concepts, namely properties, evaluation methods and taxonomies.

Properties are the expression of "how" a software entity performs, i.e., its inherent quality. A software entity can be something as small as a function in a software program or as big as a system-of-systems. Concretely, a property is a, qualitatively or quantitatively, quantifiable characteristic that is specified through: a unique name, the data format of its value with the corresponding reference units, and a documentation written in natural language. The documentation should provide enough information to primarily clarify the meaning of the property as well as its intended usage. Preferably, the documentation is to be supported by scientific references. Examples of properties that we aim to support in this work include both comprehensive ones, such as dependability, performance, cost and resource consumption, and more specific ones, such as worst-case execution time, development effort, class coupling, etc.

Evaluation methods are any approach used to give a value to a given property, be they estimation methods, measurements, mathematical calculations, prediction methods, model-checking, etc. A property can be typically assessed by many different evaluation methods (e.g., COCOMO [3] and expert estimation for the "development effort" property), which are specified through: a unique identifier, the name of the method, the data format of the output with its units, the description of the method, and nine additional attributes (Applicability, Formula, Parameters, Drivers, Assumptions, Advantages, Disadvantages, Sources, Implementations). See [12] for detailed explanations on these attributes.

Taxonomies provides the comprehensive lists all valid values for the different elements of each concept.

2.1 Development Concerns

The following concerns have been considered in the development of PROMOpedia:

- *Functional suitability.* The choice of the development framework shall be guided by its appropriateness expressed through a trade-off between the set of functionalities readily available, its monetary costs and eventual additional required development effort.
- *Usability.* The solution shall be easy to use for all stakeholders (researchers and practitioners, both browsing and contributing with new content).
- *Availability.* The solution shall be easily accessible to any stakeholder, also considering that geographically distributed researchers and practitioners are considered as the primary stakeholders.
- *Maintainability.* The solution shall be easy to adapt to accommodate changes coming from research, without having to re-implement the complete solution.
- *Security and accountability.* It shall be possible to track changes back to users and ensure that a given stakeholder has only access to a corresponding set of functionalities.

2.2 Users and their roles

The target users of PROMOpedia are researchers and practitioners dealing with EFPs and evaluation methods. It is worth noting that roles marked in italics, namely *Contributors* and *Viewers*, are conceptual roles that do not exist in the prototype. They are only used here to highlight the two main categories of authenticated users.

Currently, we consider the following user roles:

- Anonymous users can only browse publicly available contents (e.g., published description of properties and evaluation methods, available taxonomies).
- Authenticated users are users with a registered account who have access to the advanced functionalities of the system.
  - *Contributors*
    - Editors are the initial creator of a new content, and are in charge of the editorial process to publish it (e.g., adding suitable authors, drafting, ensuring completeness and consistency of the content, reviewing and asking for for revisions if needed, and submitting the new content for publication).
    - Authors participate in the writing of new content upon invitation from an editor.
    - Reviewers are responsible to ensure that the contents submitted for publication conform to the ontology and are of sufficient quality. In addition, reviewers are in charge of mapping the content descriptions in natural languages to suitable taxonomy's items. If no suitable item can be found, they can request their addition in the taxonomy.
  - *Viewers*
    - *End-users* can view published contents, compare side-by-side the attributes of a user-selected subset of evaluation methods, post comments, etc.
    - Subscribers are end-users subscribed to selected contents to receive updates upon changes (e.g., addition of new evaluation methods, similar properties, etc.)
- System administrators, who are responsible for setting up the system, configuring it, and maintaining it during its operation.

Any user of the system can have multiple context-specific roles. For example, a registered user can be editor for a content, and only subscriber to another content. This implies that for the latter, the user does not have any editorial role.

3 REALIZATION IN DRUPAL 7

Based on the aforementioned concerns and the need to integrate the core concepts of the property models ontology, we selected Drupal[1] as the main framework to build PROMOpedia as it is a web-content management system, which includes in its core a powerful taxonomy module. The taxonomy module allows to create dedicated internal vocabularies, each of them consisting of lists of hierarchical terms with their definition. The vocabularies and terms can subsequently be used to categorize and classify contents, as well as to search specific contents by pre-defined keywords.

Additionally, it provides a set of out-of-the box core functionalities, which are essential to create a crowd-based web-service whose main focus is on content and its description: tailor-made content types; community modules (comments, feeds, polls, etc.); user-management module with support for accounts, roles, and access permissions; customizable web-based user-interface with user interactions, compliant with standards and multiple types of

[1] https://www.drupal.org/

devices such as computers, tablets and smartphones; and search module.

In addition, Drupal's core functionalities can be easily extended and customized through a wide range of free contributed modules. In particular, the following modules are crucial to the realization:

- *Entity reference module*, which enables a content type to refer to another specific content type.
- *Workflow module* to create a publisher workflow, which allows reviewing contents submitted by users before their publication on-line.
- *Organic groups module* to create "groups" of users related to a specific content.
- *Rules module*, which allows specifying automated actions to be fired in the system, upon the occurrence of certain events such as notifying users of changes in the content.

3.1 Design choices

To integrate the concepts from the property models ontology into Drupal, the following design decisions have been taken:

- "Properties" and "Evaluation methods" are specified as Drupal content type;
- For each attribute from the ontology except "Formula", "Source" and "Implementation", a taxonomy is created as a Drupal's vocabulary through the taxonomy module;
- Each content type consists of two parts: (i) *the description*, which captures the textual description of each ontology's attribute via an associated textfi eld, and (ii) *the keywords*, which map each textfi eld to corresponding taxonomy terms through a number of internalfi elds. A keywordfi eld is created for each attribute of the ontology for which there exists a taxonomy.
- Each property relates to a common "archetype" stored in a property taxonomy, through the *PropertyID* field. This allows handling synonymous, polysemous, and paronymous properties.
- An evaluation method refers to a property through its unique title.

The relationships between the core concepts and the design decisions described above are illustrated in Figure 1.

As identified in [12], it is not currently possible to have a well-defined and complete list of valid terms for the different taxonomies. As a result, the list of keywords is minimal in the current prototype and will be extended with the addition of new contents. The taxonomy module allows us to identify possibly missing vocabulary terms from the content's descriptions, commonly used terms, refining their classification, etc.

Currently, an evaluation method can only be associated with a property. However, this restriction can be easily alleviated later to handle identified cases of polysemic properties or cases of evaluation methods which assess different properties.

3.2 Workflow

The workflow for the creation of new content is depicted in Figure 2. Each content type is actually an organic group, which restricts access to group members only as long as the content is not published. That way only the initial creator of the content, i.e., the Editor, and the collaborators who were invited to participate in the redaction of the new content, i.e., the Authors, can view and edit the content in draft phase. Reviewers are automatically added to the group when the Editor submits the content for publication. When the content is published on-line, any authenticated user of the system can register her interest to the content, and thus receive email notifications upon changes to the content, such as addition of new evaluation methods corresponding to a property.

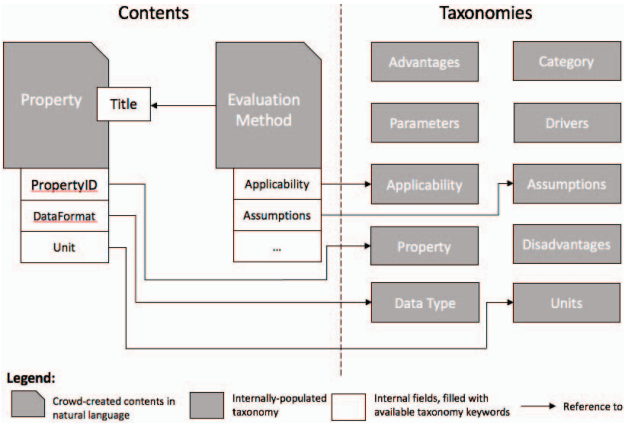

Figure 1: Core concepts and their relationships

4 VALIDATION

The validation of PROMOpedia is composed of the following three main iterations and is currently ongoing:

Iteration 1 (completed) – Evaluation of Drupal's suitability. We evaluated Drupal and its out-of-the-box core features through the realization of afi rst prototype of the PMO focusing on the development concerns stated in Section 2.

Iteration 2 (completed) – Internal use of the enhanced prototype. We experimented the use of the prototype for complete usage scenarios based on data collected from both research literature and interviews with practitioners. During this phase we identified a set of usabilityfl aws as well as inconsistencies which led to further refinements.

Iteration 3 (ongoing) – External use of PROMOpedia. Thefinal iteration consists of validating that PROMOpedia can be used by researchers and practitioners to (i) describe EFPs and evaluation methods, (ii) identify the most suitable evaluation methods according to a set of given criteria, and (iii) alleviate individual bias in selecting and describing an evaluation method. This iteration will be conductedfi rst on a small scale with selected researchers before being tested on a more extensive scale.

5 RELATED WORK

Already from the late 90s, several approaches have focused on a structured and systematic way of describing non-functional aspects. An example is NoFun, introduced by Franch [6], a notation aimed at dealing with extra-functional aspects of software systems at product level specific for component-based applications. NoFun's goal is to improve components selection based on described extra-functional properties. Other works have instead focused on trying tofi nd a common denominator in how software engineers consider non-functional aspects, as in [1].

Similar to our approach can be considered the taxonomy for identification and specification of non-functional aspects for service-oriented development [7]. The taxonomy divides non-functional aspects in three main categories: process, external, and service. The

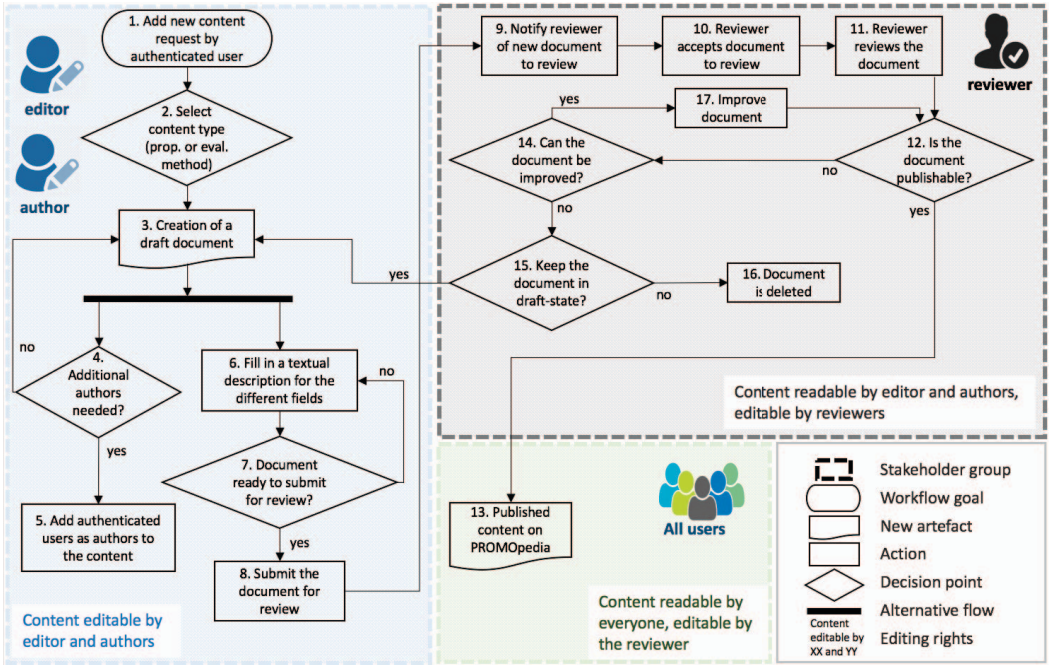

Figure 2: Workflow for creating and publishing new content

taxonomy is meant as a starting point and checklist for handling non-functional issues. Our approach is not domain-specific, does not entail only afi xed set of properties, and does not aim at providing a checklist. The focus is instead on providing a tool for software researchers and practitioners to share knowledge and experience on non-functional properties and related evaluation methods.

Our approach is partially based on crowd-sourcing for enhancement and enlargement purposes. Unlike PROMOpedia, other crowd-sourcing approaches, such as generic ones like Wikipedia[2] or more specific ones like the CVTree3 web server for genome descriptions [14] or the Interactive Tree of Life [10], do not primarily target the research community (i.e., Wikipedia), are not able to control changes once published (even though often possible to track changes), and are not intended to foster research-related discussions.

6 CONCLUSION

In this paper, we have introduced PROMOpedia, a unique web-content management system coupled with crowd-sourcing of scientific contents related to properties and their evaluation methods. The core concepts of PROMOpedia leverage a formalized property models ontology. PROMOpedia is able to control changes to published content and provides a powerful arena for research-related discussions involving both researchers and practitioners. As future work, we plan to investigate solutions to ensure content quality and consistency, avoid content duplication and bloat, as well as identifying means to activate the community to foster discussions through PROMOpedia.

[2]http://www.wikipedia.com

ACKNOWLEDGEMENTS

The work is supported by a research grant for the ORION project (reference number 20140218) from The Knowledge Foundation in Sweden. Furthermore, we would like to thank Kai Petersen for his feedbacks on thefi rst prototype.

REFERENCES

[1] D. Ameller, C. Ayala, J. Cabot, and X. Franch. 2012. How do software architects consider non-functional requirements: An exploratory study. In *Procs of RE*. 41–50.
[2] N.D. Anh, D.S. Cruzes, R. Conradi, M. Höst, X. Franch, and C. Ayala. 2012. *Collaborative Resolution of Requirements Mismatches When Adopting Open Source Components*. 77–93.
[3] B. et al Boehm. 1995. Cost models for future software life cycle processes: COCOMO 2.0. *Annals of software engineering* 1, 1 (1995), 57–94.
[4] I. Crnkovic, S. Sentilles, A. Vulgarakis, and M. R. V. Chaudron. 2011. A Classification Framework for Software Component Models. *IEEE Transactions on Software Engineering* 37, 5 (2011), 593–615.
[5] J. Eckhardt, A. Vogelsang, and D.M. Fernández. 2016. Are "Non-functional" Requirements Really Non-functional?: An Investigation of Non-functional Requirements in Practice. In *Pros of ICSE*. 832–842.
[6] X. Franch. 1998. Systematic formulation of non-functional characteristics of software. In *Procs of RE*. 174–181.
[7] M. Galster and E. Bucherer. 2008. A Taxonomy for Identifying and Specifying Non-Functional Requirements in Service-Oriented Development. In *IEEE Congress on Services - Part I*. 345–352.
[8] C. Gencel, R. Heldal, and K. Lind. 2009. On the relationship between different size measures in the software life cycle. In *Procs of APSEC*. 19–26.
[9] Martin Glinz. 2007. On non-functional requirements. In *Procs of RE*. 21–26.
[10] I. Letunic and P. Bork. 2006. Interactive Tree Of Life (iTOL): an online tool for phylogenetic tree display and annotation. *Bioinformatics* 23, 1 (2006), 127–128.
[11] J. Saraiva, S. Soares, and F. Castor. 2013. Towards a catalog of Object-Oriented Software Maintainability metrics. In *Procs of WETSoM*. 84–87.
[12] S. Sentilles, E. Papatheocharous, F. Ciccozzi, and K. Petersen. 2016. A Property Model Ontology. In *Procs of SEAA*. 165–172.
[13] E.J. Weyuker. 2011. Empirical Software Engineering Research - The Good, The Bad, The Ugly. In *Procs of ESEM*. 1–9.
[14] G. Zuo and B. Hao. 2015. CVTree3 web server for whole-genome-based and alignment-free prokaryotic phylogeny and taxonomy. *Genomics, proteomics & bioinformatics* 13, 5 (2015), 321–331.