Journal of Web Engineering, Vol. 16, No. 7&8 (2017) 541–570

A TAXONOMY OF WEB EFFORT PREDICTORS

RICARDO BRITTO[a]
Department of Software Engineering, Blekinge Institute of Technology
Karlskrona, 371 49, Sweden
ricardo.britto@bth.se

MUHAMMAD USMAN
Department of Software Engineering, Blekinge Institute of Technology
Karlskrona, 371 49, Sweden
muhammad.usman@bth.se

EMILIA MENDES
Department of Computer Science and Engineering, Blekinge Institute of Technology
Karlskrona, 371 49, Sweden
emilia.mendes@bth.se

Received April 16, 2016
Revised April 3, 2017

Web engineering as a field has emerged to address challenges associated with developing Web applications. It is known that the development of Web applications differs from the development of non-Web applications, especially regarding some aspects such as Web size metrics. The classification of existing Web engineering knowledge would be beneficial for both practitioners and researchers in many different ways, such as finding research gaps and supporting decision making. In the context of Web effort estimation, a taxonomy was proposed to classify the existing size metrics, and more recently a systematic literature review was conducted to identify aspects related to Web resource/effort estimation. However, there is no study that classifies Web predictors (both size metrics and cost drivers). The main objective of this study is to organize the body of knowledge on Web effort predictors by designing and using a taxonomy, aiming at supporting both research and practice in Web effort estimation. To design our taxonomy, we used a recently proposed taxonomy design method. As input, we used the results of a previously conducted systematic literature review (updated in this study), an existing taxonomy of Web size metrics and expert knowledge. We identified 165 unique Web effort predictors from a final set of 98 primary studies; they were used as one of the basis to design our hierarchical taxonomy. The taxonomy has three levels, organized into 13 categories. We demonstrated the utility of the taxonomy and body of knowledge by using examples. The proposed taxonomy can be beneficial in the following ways: i) It can help to identify research gaps and some literature of interest and ii) it can support the selection of predictors for Web effort estimation. We also intend to extend the taxonomy presented to also include effort estimation techniques and accuracy metrics.

Keywords: Web effort predictors, Taxonomy, Knowledge Classification, Web Engineering
Communicated by: M. Gaedke & L. Olsina

1. Introduction

1.1. *Context*

New software development approaches emerge as time goes by and the software industry aims at enhancing existing processes to reduce waste and increase profitability. As a consequence of the

[a]Corresponding author

emergence of the World Wide Web (WWW), new software development approaches were suggested to address the challenges raised in this new scenario. They also led to the creation of a new research field known as Web Engineering [13].

The aim of Web engineering is to develop and maintain quality Web applications by following engineering and management principles and practices in a systematic way [4]. Web engineering differs from software engineering in three main aspects [13]:

- The people involved in the development.
- The Web applications' intrinsic properties.
- The wider audience of the Web applications

Web applications generally have multi-tiered structure and non-code elements, such as images and audio/video, are designed and used in developing these applications [19]. Besides programmers and IT professionals, the development of Web applications also needs the involvement of graphic designers and content writers [10].

The development processes of most Web development companies are ad-hoc and applications are delivered in short development cycles [15, 8, 17]. Effort estimation is carried out in these companies without a formal process [15, 16]. Effort estimation, i.e. the process used to predict the effort needed to develop a particular application, is a project management activity that is fundamental to managing Web project's resources in an efficient way [10]. The estimation of Web applications' development effort involves the selection of an adequate effort estimation technique, as well as the identification and measurement of appropriate Web effort predictors, i.e. size metrics and cost drivers [9].

1.2. *Problem outline*

In the context of Web effort estimation, Mendes *et al.* [11] have designed a taxonomy to classify hypermedia and Web size metrics. A taxonomy is "a scheme of classification"[b] that was initially designed by Carl Linnaeus to classify organism [7], although researchers have used this approach to organizing knowledge in other fields, such as Computer Science [5] and Psychology [14]. Originally, it was devised to classify knowledge in a hierarchical way, although to date many different classification structures have been used to construct taxonomies, e.g. "hierarchy", "tree" and "facet-based" [6].

Knowledge classification can have positive implications for both academia and industry, as follows:

- Facilitates knowledge sharing [21, 23].
- Helps in identifying gaps in a particular knowledge area [21, 23].
- Provides a better understanding of the interrelationships between the factors associated with a particular knowledge area [21].
- Supports decision-making processes [21].

Despite the fact that Mendes *et al.* [11] proposed a taxonomy of size metrics, their work does not account for cost drivers and was based on the relevant literature published between 1992 and 2003.

[b] www.oxforddictionaries.com

Azhar *et al.* [1], through a Systematic Literature Review (SLR) on Web resource estimation[c], identified many Web effort predictors. However, they did not classify the identified Web effort predictors in a meaningful way (see Section 2 for more details).

1.3. *Objective*

The objective of this paper is to design and use a taxonomy to organize the body of knowledge on Web effort predictors (size metrics and cost drivers). To do so, we used the results from Azhar *et al.*'s SLR and Mendes *et al.*'s taxonomy [11] as input, along with expert knowledge.

1.4. *Contribution*

The contribution of this paper is three-fold:

- A Web effort predictors' taxonomy based on evidence from an SLR (updated herein), a previous size metric taxonomy and expert knowledge.
- The validation of the proposed taxonomy by means of comparing it with Mendes *et al.*'s taxonomy [11] and by classifying existing related literature.
- The organized body of knowledge related to Web effort predictors, i.e. 165 predictors identified in 98 studies were classified using the proposed taxonomy.

1.5. *Outline*

The rest of the paper is organized as follows: Section 2 describes the related work. Section 3 presents the research design and methodology employed herein. Section 4 presents our proposed Web effort predictors taxonomy, followed by Section 5 where our taxonomy is used to classify existing literature. Section 6 discusses the implications of our proposal for both researchers and practitioners. Section 7 presents the validity threats associated with this work. Finally, Section 8 concludes the paper and presents our comments on future work.

2. Related work

The taxonomy proposed in this work is based on the studies by Azhar *et al.* [1] and Mendes *et al.* [11]. In this section, we briefly describe these studies and also discuss how our work extends these related papers.

2.1. *SLR by Azhar et al.*

Azhar *et al.* [1] have conducted an SLR to identify techniques, accuracy measures and predictors used within the context of Web resource estimation. Their SLR was based on 84 primary studies on Web resource estimation.

The authors have identified that case base reasoning and regression-based estimation techniques are most frequently used Web resource estimation techniques, with varying degrees of accuracy. They also identified a number of Web resource predictors employed during Web effort estimation. They

[c]Web resource estimation embraces also design, quality, development and maintenance effort, while Web effort estimation deals only with development effort.

categorized the identified size metrics, using the three following categories: length, functionality, and reusability.

We identified the following issues with the classification of size metrics by Azhar *et al.*:

- The identified size metrics are not reported in their study, i.e. only the respective size metric categories are used to classify the identified primary studies, without any further detail about each individual size metric.
- Many Web predictors are classified only as "Tukutuku predictors", disregarding the fact that Tukutuku set of predictors [12] includes both size metrics and cost drivers. In addition, the individual predictors classified in this category are not reported in the paper, i.e. all the identified studies are classified as Tukutuku, without any further details.
- No classification for cost drivers is provided. In addition, the identified cost drivers are not reported in the paper, i.e. all the identified studies are classified as cost drivers, without any further detail.

Therefore, this paper addresses the issues of Azhar *et al.*'s work by proposing a taxonomy of Web effort predictors, which embraces both size metrics and cost drivers, and classifies web effort predictors reported in the included primary studies using the proposed taxonomy.

2.2. *Taxonomy of size metrics by Mendes et al.*

Mendes *et al.* [11] have proposed a taxonomy of hypermedia and Web application size metrics. They designed a facet-based taxonomy, based on a number of measurement concepts and the literature on software size metrics and measurement. These concepts included the motivation of each size metric, harvesting time during the development cycle when a particular size metric should be measured, categories of size metrics (length, functionality, and complexity), entity, measurement scales, computation, validation and model dependency. Their taxonomy was employed to classify the size metrics identified in the relevant studies published between 1992 and 2003.

Our work differs from Mendes *et al.*'s taxonomy in the following ways:

- Our taxonomy covers both size metrics and cost drivers.
- We complemented the literature reviewed by Mendes *et al.* by considering the studies used by Azhar *et al.* and by adding studies published until September 2014. With the discovery of new knowledge, existing taxonomies need to evolve over time.
- Our taxonomy was designed in a systematic way by applying a method, as described in Section 3, while Mendes *et al.*'s taxonomy was developed without following any taxonomy design method.

3. Research design and methodology

In this section, we present the research design and methodology used. The following research question is investigated in this work:

- How to organize the body of knowledge on Web effort predictors reported in the literature?

As mentioned before, we decided to design a taxonomy to organize the body of knowledge on Web effort predictors. Therefore, we carried out the following steps to fully answer our research question:

1. We identified relevant literature to be used as the basis for the Web effort predictors' taxonomy.
2. We employed the process proposed by Usman *et al.* [20] to design the Web effort predictors' taxonomy.
3. We used the designed taxonomy to classify the predictors identified in the existing literature on Web effort estimation.

3.1. *Step 1 - Identify relevant literature*

We used the 84 primary studies included in Azhar *et al.*'s SLR [1] as our starting point. However, Azhar *et al.*'s SLR covered only studies published until February 2012. Thus, we searched for additional literature published between March 2012 and September 2014. Before surveying databases to retrieve additional studies, we slightly modified Damir *et al.*'s search string, because the original search string was designed to fetch Web resource estimation studies and our work was limited to Web effort estimation. The modified search string is as follows:

(Web) **AND** (cost **OR** effort) **AND** (estimation **OR** prediction **OR** forecasting)

We used the modified search string to retrieve studies from Scopus[d] and Compendex/Inspec[e]. The search string was applied on titles and abstracts. These sources cover important databases such as IEEE, Springer, ACM and Elsevier that publish leading SE journals and conference proceedings. In addition, the selected primary sources are able to handle advanced queries. Table 1 presents the surveyed primary sources along with the corresponding number of primary studies returned (439).

Table 1: Summary of search results

Database / Search Engine	**Search Results**
Scopus	129
Compendex and Inspec	310
Total	439

To select relevant studies, we carried out a 2-phase selection process. In the first phase, the two first authors screened together all the 439 titles and abstracts, employing the following selection criteria (adapted from Damir *et al.* [1]):

- Inclusion
  - Studies that investigate Web effort predictors AND
  - Studies that provide an empirical basis for their findings.
- Exclusion

[d] www.scopus.com
[e] www.engineeringvillage.com

- Studies whose full-text are not accessible OR
- Studies that are not written in English OR
- Studies that are not reported in a peer-reviewed workshop, conference, or journal OR
- Studies that lack empirical evidence

As a result of phase 1, 14 studies were deemed as relevant. The aforementioned selection criteria were then applied on the full-text of the 14 studies and all of them were judged as relevant. These 14 studies along with the 84 studies used in Damir *et al.* [1] (98 in total) were employed as one of the inputs for designing our taxonomy.[f]

Note that the extraction phase, like the search phase, was also simplified herein, because we only extracted size metrics and cost drivers from the 14 additional papers, while Azhar *et al.* have extracted also effort estimation techniques and accuracy measures. We did so because only the effort predictors were relevant to design our taxonomy.

3.2. *Step 2 - Design the taxonomy*

To design our taxonomy, we employed the method proposed by Usman *et al.* [20]. This process is a revised version of the method previously proposed by Bayona-Oré *et al.* [2].

Usman *et al.*'s method has 13 activities, which are presented in Table 2. These activities are grouped into 4 phases, as follows:

- **Planning** - This phase is related to defining initial aspects of the taxonomy to be designed, such as the objective of the taxonomy, the software engineering (SE) knowledge [3] associated with the selected subject matter (i.e. the object to be classified [22]) to be classified. The data collection methods [24], classification structure type (hierarchy, tree, paradigm or facet-based [6]) and classification procedure type (qualitative or quantitative [22]) are also selected in this phase.
- **Identification and extraction** - This phase is related to the information extraction and control of terminology describing the extracted terms.
- **Design and construction** - In this phase, the dimensions and categories are identified and described, as well as the relationships between them, leading to a classification scheme. Guidelines for use and evolution of the taxonomy are also defined in this phase.
- **Validation** - In the last phase of this process, the taxonomy is validated. A taxonomy can be validated in three ways [18]:
  - **Orthogonality demonstration** - The orthogonality or mutual exclusiveness of the taxonomy categories and sub-categories is ensured and described.
  - **Benchmarking** - The proposed taxonomy is compared with existing relevant taxonomies, if any.
  - **Utility demonstration** - The utility of the taxonomy can be demonstrated by classifying existing knowledge.

[f] The list of primary studies is available on-line at https://goo.gl/PeXuVV

Table 2: Updated taxonomy design method.

Phase	Id	Activity
Planning	B1	Define SE knowledge area of the study
	B2	Describe the objectives of the taxonomy
	B3	Select and describe the subject matter to be classified
	B4	Select classification structure type
	B5	Select classification procedure type
	B6	Identify the sources of information
Identification and extraction	B7	Extract all the terms
	B8	Perform terminology control
Design and construction	B9	Identify and describe the taxonomy dimensions
	B10	Identify and describe the categories of each dimension
	B11	Identify and describe the relationships
	B12	Define guidelines for using and updating the taxonomy
Validation	B13	Validate the taxonomy

The data extracted from 98 primary studies selected in the previous step, along with Mendes *et al.* taxonomy and the knowledge of the third author of this paper, who is an expert in Web engineering, were used to design our taxonomy. The resulting Web effort predictors' taxonomy is presented in Section 4.

3.3. *Step 3 - Classify identified predictors*

To organize the body of knowledge on Web effort predictors, we classified the 98 primary studies (selected in step 1) using the taxonomy detailed in Section 4. The classification is presented in Section 5.

4. Web effort predictors' taxonomy

In this section, we present our Web effort predictors' taxonomy, whose details are presented according to the phases of the method described in Section 3.

4.1. *Planning*

In this phase, we carried out 6 activities (B1, B2, B3, B4, B5, and B6). The knowledge area associated with the designed taxonomy is *project management* (the outcome of **B1**). The main objective of the

proposed taxonomy is to *define a set of categories that enables to classify the Web effort predictors (size metrics and cost drivers) reported in the existing literature* (the outcome of **B2**). The subject matter of the designed taxonomy is *Web effort predictors* (the outcome of **B3**).

To design the taxonomy presented herein, we chose *hierarchy* to be the classification structure [6] (the outcome of **B4**). We did so because there is an extensive and mature state-of-the-art on Web effort estimation, which allows for designing a hierarchical taxonomy with well-defined categories. The hierarchical structure relates categories and sub-categories in a parent-child relationship [6], which is also the case in our taxonomy (e.g., lines of code *is-a* size metric, and a size metric *is-a* Web effort predictor). In addition, the third author of this paper is an expert on Web effort estimation, and her knowledge helped to identify the best structure to design the proposed taxonomy.

The procedure used to classify the Web effort predictors is *qualitative* in nature (the outcome of **B5**). We selected this classification procedure type because it is an appropriate approach to ensure mutual exclusiveness between categories in a hierarchical taxonomy [22].

The last activity of this phase was to select the method to collect relevant data. As discussed in Section 3, the data used as the basis to design our taxonomy was gathered by means of a *systematic literature review* (the outcome of **B6**).

4.2. *Identification and extraction*

In this phase, we carried out 2 activities (B7 and B8). We extracted all the size metrics and cost drivers (Web effort predictors) from the primary studies included in Azhar *et al.*'s SLR (84 studies) and from the additional 14 studies we identified later on. As a result, we identified 214 *size metrics* and 148 *cost drivers* (the outcome of **B7**).

To carry out activity B8, we used the knowledge of the third author in Web effort estimation as input, which was mandatory to achieve consistent results. Activity **B8** enabled us to control the consistency of the extracted terms (Web predictors). We performed the following actions to improve the terminology associated with the extracted Web effort predictors:

1. The duplicates were removed.
2. Whenever we identified a case where one predictor was represented by different terms in different studies, a terminology unification was performed. To do so, a Delphi-inspired process was employed whereby the authors independently chose a term to name the predictor in question; the deliberations were made to arrive at a consensus; this process was to be repeated until consensus was reached.
3. The terminology associated with many predictors was slightly changed to improve legibility; in many cases, it was nearly impossible to understand the purpose of a predictor without a comprehensive text to describe it.

As a result of actions 1 and 2, the number of predictors was reduced from 362 (214 size metrics and 148 cost drivers) to 165 (88 size metrics and 77 cost drivers). We present the terms that were unified in Table 3, which is the result of the terminology control (see more details in Section 5).

4.2.1. *Design and construction*

In this phase, we carried out 4 activities (B9, B10, B11, and B12). We used the terms extracted and controlled through B7 and B8 to identify and describe the taxonomy dimensions and categories. We

Table 3: Terminology control results.

Final term	**Original terms**
Web page count	Total number of Web pages, page count, number of text pages, Web pages, brochure pages, number of HTML files
Media count	Media count, number of animations in the application, number of audio/video files, medias, high complexity images, medium complexity images, low complexity images, overall complexity images, animation graphics
Component count	Number of used features off-the-shelf, components
Reused component count	Number of reused high effort Fots-adapted, Number of reused low effort Fots-adapted
Feature count	Features programming, animation programming,front end design, graphic design, front end build, context and user base analysis, analysis of on-line demand and offer, newsletter, customization by editorial staff, site findability and positioning verification context architecture management and re-aggregation of tags and keys, system infrastructure, general search engine on site, preparation of bare mock-up, requirements and navigation, content management system, production of logo and corporate image, graphic layout production, creation of ad-hoc texts and pictures/videos, map or background, communicability and social management, templates and navigation system, user role management, multilingualism, DB and internal query creation, report system design, external query, cartographic DB, creation and inclusion of customized maps, clickable maps, file types managed by application, management of reserved areas, external system access, service available outside of application, data input models
Link count	Internal links, number of external links, number of links
Domain experience	Team experience, experience of the developer, personnel experience, lack of employee experience, experience in business area
Availability level	Client unavailability, client time frame
Documentation level	Documentation, project documentation

identified only one dimension, named *Web effort predictor* (the outcome of **B9**) at the very top, making it the eventual parent category of all the sub and sub-sub categories. This was used later on to classify the body of knowledge on Web effort size metrics and cost drivers (Section 5).

To identify the categories of this main dimension, we first looked at Mendes *et al.*'s categories [11]; three categories were identified as relevant (*length, complexity*, and *functionality*). To identify additional categories, we employed an iterative and empirical approach; we examined the extracted Web effort predictors and looked for similarities and differences between them. As a result, 7 additional categories were identified, named *object-oriented, client, development company, project, product, team*, and *technology* respectively (the outcome of **B10**). While carrying out this activity **B10**, we added other two more categories: *size metric* and *cost driver*.

As mentioned before, we selected hierarchy as the classification structure type; such a choice had a direct influence on the type of relationship between categories [6]. Therefore, we used parent-child (also known as is-a or inheritance) relationship in our taxonomy to relate categories and sub-categories (the outcome of **B11**), which resulted in the Web effort predictors' taxonomy displayed in Figure 1.

The taxonomy has 3 hierarchical levels: i) the first level contains the root of the taxonomy, which is the subject matter to be classified (*Web effort predictor*); ii) the second level has two categories: *size metric* and *cost driver*; iii) The third level of the taxonomy further divides the size metrics and cost drivers into sub-categories. The categories of the third hierarchical level are defined as follows:

- Size metrics
  - **Length** - This category includes metrics that directly measure the length of Web applications based on the size/length of their compounding elements.
  - **Object-oriented** - This category embraces metrics that measure indirectly the size of Web applications based on object-oriented properties of their compounding elements.
  - **Functionality** - This category embraces metrics that indirectly measure the size of Web applications based on their feature/functions.
  - **Complexity** - This category embraces metrics that indirectly measure the size of Web applications based on the difficulty associated with their compounding elements.
- Cost drivers
  - **Client** - This category embraces cost drivers related to the client who demands the development of a Web application.
  - **Development company** - This category embraces cost drivers associated with the company hired by a client to develop a Web application.
  - **Product** - This category embraces cost drivers associated with requirements and restrictions associated with a Web application.
  - **Project** - This category embraces cost drivers associated with the setting of a Web application project.
  - **Team** - This category embraces cost drivers that are associated with the development team responsible for carrying out a Web application project.
  - **Technology** - This category embraces cost drivers that are associated with the technologies (programming language, tools, platforms) demanded in a Web application project.

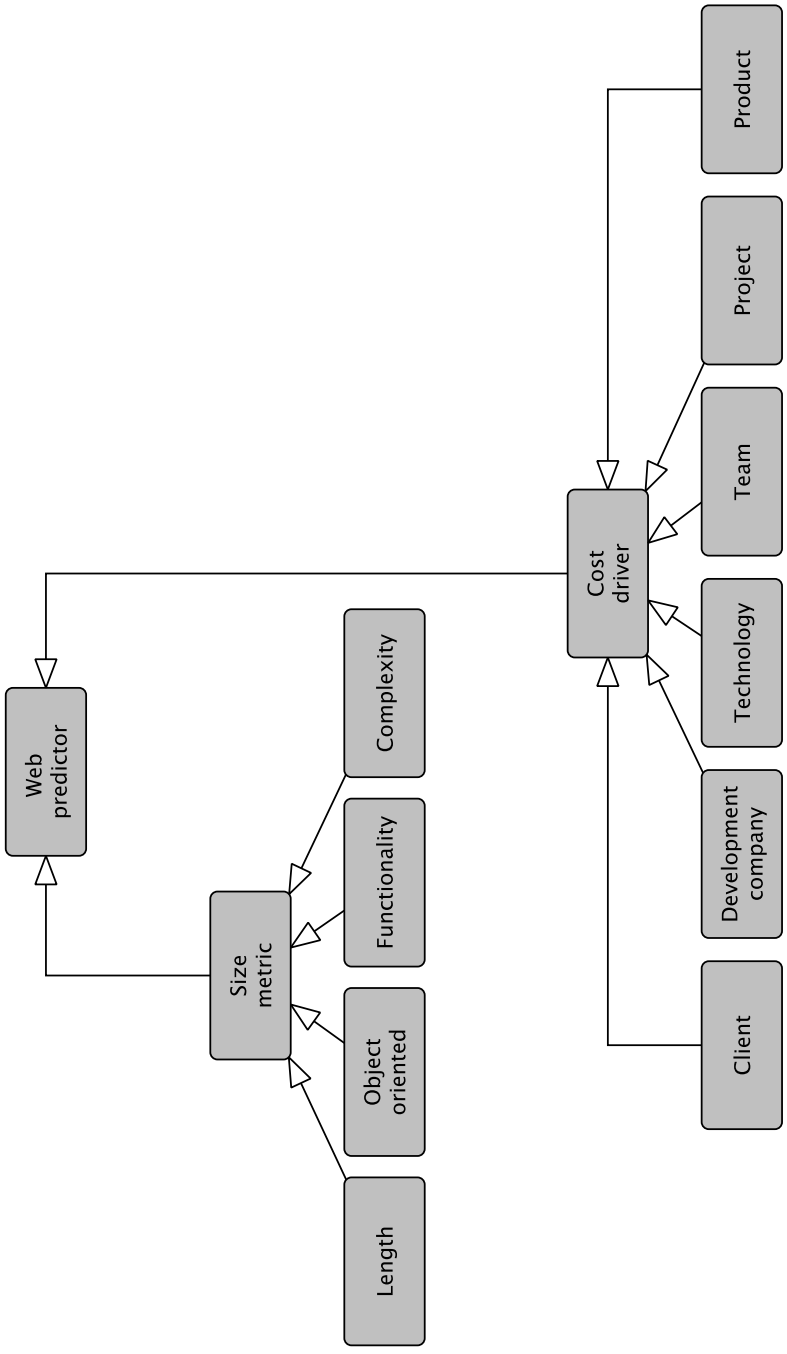

Fig. 1: Web effort predictors' taxonomy.

We believe that the third hierarchical level is the only one that can be updated in future to evolve the taxonomy presented herein. There are just two types of Web effort predictors [9]: size metrics and cost drivers, therefore the corresponding second hierarchical level is complete. In addition, we believe that it is important to keep all the applicable categories in only one hierarchical level, i.e. categories that are to be used to classify Web predictors. In doing so, the taxonomy is easy to understand and apply (the outcome of **B12**).

The description for each category must be consulted whenever a Web predictor is to be classified. This clear category description supports an easy understanding and application of the presented taxonomy (the outcome of **B12**).

4.2.2. *Validation*

The last activity of the employed taxonomy design process validates the taxonomy (**B13**). We conducted this activity as follows:

- **Orthogonality demonstration** - We used a bottom-up approach to identify the categories of the taxonomy presented in this paper: i) first, we identified all the Web predictors reported by the selected primary studies and ii) second, we analyzed those Web effort predictors to identify differences and similarities between them. This process led us to identify orthogonal categories, as presented in Figure 1.
- **Benchmarking** - As discussed in Section 2, only Mendes *et al.*'s taxonomy [11] is similar to the taxonomy put forward in this paper. However, Mendes *et al.*'s taxonomy focuses only on size metrics within both hypermedia and Web application contexts. The taxonomy presented herein embraces both size metrics and cost drivers, focusing only on the Web application context. In addition, our taxonomy is graphically presented and is designed in a systematic way by following the taxonomy design method described in Section 3.
- **Utility demonstration** - To demonstrate the utility of our taxonomy, we classified all the size metrics and cost drivers extracted from the 98 included primary studies. The result from this classification is further detailed in Section 5 and discussed in Section 6.

5. Web effort predictors' classification

In this section, we present the results regarding the classification of the Web predictors identified in the 98 included primary studies (see Section 3). In total, we extracted 165 unique Web predictors, which are detailed in Tables 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 and 18. Note that the column "F" presents the frequencies of each predictor in all the tables.

5.1. *Size metrics*

We identified 88 unique size metrics that were classified according to the following 4 categories (see Section 4): length, functionality, object-oriented and complexity.

5.1.1. *Length*

We identified 54 length size metrics, which represents 61.46% of all size metrics reported herein. The size metrics classified in this category are as follows: Web page count, Media count, New media count,

New Web page count, Link count, Program count, Reused component count, Lines of code, Reused program count, Reused media count, Web page allocation Reused lines of code, Media allocation, Reused media allocation, Entity count, Attribute count, Component count, Statement count, Node count, Collection slot size, Component granularity level, Slot granularity level, Model node size, Cluster node size, Node slot size, Publishing model unit count, Model slot size, Association slot size, Client script count, Server script count, Information slot count, Association center slot count, Collection center slot count, Component slot count, Semantic association count, Segment count, Slot count, Cluster slot count, Cluster count Publishing unit count, Section count, Inner/sub concern count, Indifferent concern count, Module point cut count, Module count, Module attribute count, Operation count, Comment count, Reused comment count, Media duration, Diffusion cut count, Concern module count, Concern operation count and Anchor count. A description of each metric and the primary studies from where they were extracted are presented in Tables 4, 5 and 6.

5.1.2. *Functionality*

We identified 12 functionality size metrics (13.63%) that are as follows: High feature count, Low feature count, Reused high feature count, Reused low feature count, Web objects, COSMIC, IFPUG, OO-HFP, OO-FP, Use case count, Feature count and Data Web points. A description of each metric and the primary studies from where they were extracted are presented in Table 7.

5.1.3. *Object-oriented*

We identified three object-oriented size metrics (3.4%) that are as follows: Cohesion, class coupling and concern coupling. A description of each metric and the primary studies from where they were extracted are presented in Table 8.

5.1.4. *Complexity*

We identified 19 complexity size metrics (21.59%) that are as follows: Connectivity density, Cyclometic complexity, Model collection complexity, Model association complexity, Model link complexity, Page complexity, Component complexity, Total complexity, Adaptation complexity, New complexity, Data usage complexity, Data flow complexity, Cohesion complexity, Interface complexity, Control flow complexity, Class complexity, Layout complexity, Input complexity and Output complexity. A description of each metric and the primary studies from where they were extracted are presented in Table 9.

5.2. *Cost drivers*

We identified 77 unique size metrics that were classified according to the following six categories (see Section 4): client, development company, product, project, team and technology.

5.2.1. *Product*

We identified 35 product cost drivers, which represents 45.45% of all cost drivers reported herein. The cost drivers classified in this category are as follows: Type, Stratum, Compactness, Structure Architecture, Integration with legacy systems, Concurrency level, Processing requirements, Database size, Requirements volatility level, Requirements novelty level, Reliability level, Maintainability level, Time efficiency level, Memory efficiency level, Portability level, Scalability level, Quality level Us-

Table 4: Length size metrics - part 1.

Size metric	Description	References	F
Web page count	Total number of Web pages in the Web application	s3, s4, s5, s6, s7, s8, s9, s10, s11, s14, s16, s17, s19, s21, s22, s24, s25, s32, s34, s39, s40, s42, s43, s44, s46, s47, s48, s50, s51, s52, s53, s54, s55, s64, s65, s66, s67, s71, s72, s76, s81, s84, s85, s90, s91, e5, ss3, ss4, ss7, ss12	50
Media count	Total number of media files (audio, video, animation, images)	s6, s8, s9, s10, s11, s14, s15, s16, s17, s21, s25, s35, s39, s42, s43, s50, s53, s64 s76, e5, ss3, ss12	22
Component count	Number of components	s19, s35, s47, s46, s47, s53, s54, s55, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91, e7	19
Reused component count	Number of reused components	s46, s53, s54, s55, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91, s96, e7	16
New media count	Total number of new media files	s46, s53, s54, s55, s65, s66, s67, s71, s72, s76, s81, s84, s85, s90, s91	15
New Web page count	Number of new Web pages in the Web application	s46, s53, s54, s55, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91	14
Link count	Number of links in the Web application	s3, s4, s5, s6, s7, s8, s9, s10, s11, s15, s19, s76, e5, ss5	14
Program count	Number of classes	s6, s8, s9, s10, s11, s14, s15, s17, s21, s25, s39, e5	12
Lines of code	It represents the total lines of code.	s2, s9, s6, s8, s9, s10, s11, s15, s37, s49, s64, e5	12
Reused program count	Number of reused classes	s6, s8, s9, s10, s11, s14, s15, s17, s21, s25, s39, e5	12
Reused media count	Number of reused media files	s6, s8, s9, s10, s11, s14, s15, s17, s21, s25, s39, e5	12
Web page allocation	Memory space consumed by Web pages	s6, s8, s9, s10, s11, s15, s47, e4, e5	9
Reused lines of code	Number of lines of code that are reused.	s2, s6, s8, s9, s10, s11, s15, e5	8
Media allocation	Memory space consumed by all the media files	s6, s8, s9, s10, s11, s15, e5	7
Reused media allocation	Memory space consumed by the reused media files	s6, s8, s9, s10, s11, s15, e5	7
Entity count	Number of entities in the data model	s16, s19, s47, e4	4

Table 5: Length size metrics - part 2.

Size metric	Description	References	F
Attribute count	Number of attributes per entity in the data model	s16, s19, s47, e4	4
Statement count	Number of statements in the Web application source code.	s14, s16, s57	3
Node count	Number of nodes in the navigation diagram	s16, s19, s47	3
Collection slot size	Average size of slots per collection center	e4, s47	2
Component granularity level	Average component granularity level per entity	e4, s47	2
Slot granularity level	Average granularity of decomposition of slots per component	e4, s47	2
Model node size	Average size of nodes in the model	e4, s47	2
Cluster node size	Average size of nodes per cluster	e4, s47	2
Node slot size	Average size of slots per node	e4, s47	2
Publishing model unit count	Size of publishing units in the model	e4, s47	2
Model slot size	Size of slots in the model	e4, s47	2
Association slot size	Average size of slots per association center	e4, s47	2
Client script count	Number of java script files on client side	s42, s76	2
Server script count	Number of java script files on server side	s42, s76	2
Information slot count	Number of slots in the model	s19, s47	2
Association center slot count	Number of slots per semantic association center in the model	s19, s47	2
Collection center slot count	Number of slots per collection center in the model	s19, s47	2
Component slot count	Number of slots per component in the model	s19, s47	2
Semantic association count	Number of semantic associations in the model	s19, s47	2
Segment count	Number of segments in the model	s19, s47	2
Slot count	Overall number of slots	s19, s47	2
Cluster slot count	Number of slots per cluster	s19, s47	2
Cluster count	Number of clusters	s19, s47	2

Table 6: Length size metrics - part 3.

Size metric	Description	References	F
Publishing unit count	Number of publishing units	s19, s47	2
Section count	Number of sections	s19, s47	2
Inner/sub concern count	Number of inner/sub concerns in a Web application	s37	1
Indifferent concern count	Number of indifferent concerns in a Web application	s37	1
Attribute count	Number of attributes per entity in the data model	s16, s19, s47, e4	4
Module point cut count	Number of module point cut in a Web application	s37	1
Module count	Number of modules in a Web application	s37	1
Module attribute count	Number of module attributes in a Web application	s37	1
Operation count	Number of weighted operations per component	s37	1
Comment count	Number of lines of comments	s11	1
Reused comment count	Number of reused lines of comments	s11	1
Media duration	Play time for audio/video files	s11	1
Diffusion cut count	Number of diffusion cuts in a Web application	s37	1
Concern module count	Number of modules for concern	s37	1
Concern operation count	Number of operations for concern	s37	1
Anchor count	Number of anchors in the navigation diagram	s16	1

Table 7: Functionality size metrics.

Size metric	Description	References	F
High feature count	Total number of high effort features requested by the user	s43, s46, s50, s53, s54, s55, s64, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91, ss3, ss6	19
Low feature count	Total number of low effort features requested by the user	s43, s46, s50, s53, s54, s55, s64, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91, ss3	18
Reused high feature count	It represents the number of reused high effort features requested by the user	s46, s53, s54, s55, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91	14
Reused low feature count	It represents the number of reused low effort features requested by the user	s46, s53, s54, s55, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91	14
Web Objects	A Web development-focused version of function points	s27, s28, s42, s48, s56, s58, s62, s70, s75, s76, s87, s92, s93, s96	14
COSMIC	Function points calculated using COSMIC method	s9, s31, s35, s41, s45, s62, s74, s77, e1, ss2, ss13	11
IFPUG	Function points calculated using IFPUG method	s2, s28, s30, s57, s63, s70, s93, s96	8
OO-HFP	Objected-oriented hypermedia version of function points	s2, s46, s82, ss5	4
OO-FP	Object-oriented version of function points	s2, s57, s69	3
Use case count	Functional metric based on use case descriptions that is calculated using use case points method	s16, e3, ss1	3
Feature count	Functional size metric that counts all the features requested by the user	s64, ss12	2
Data Web points	A functional metric that measures the functionality of Web applications	s26	1

Table 8: Object-oriented size metrics.

Size metric	Description	References	F
Cohesion	Cohesion of the Web application classes	s49	1
Class coupling	Coupling of the Web application classes	s49	1
Concern Coupling	Coupling of the Web application concerns	s37	1

ability, level Readability level Security level, Installability level, Modularity level, Flexibility level, Testability level, Accessibility level, Trainability level, Innovation level, Technical factors, Storage constraint, Reusability level, Robustness level, Design volatility, Experience level and Requirements clarity level. A description of each cost driver and the primary studies from where they were extracted are presented in Tables 10, 11 and 12.

5.2.2. *Client*

We identified four client cost drivers (5.19%) that are as follows: Availability level, IT literacy, Mapped workflows and Personality. A description of each cost driver and the primary studies from where they were extracted are presented in Table 13.

5.2.3. *Development company*

We identified four development company cost drivers (5.19%) that are as follows: SPI program, Metrics' program, Number of projects in parallel and Software reuse. A description of each cost driver and the primary studies from where they were extracted are presented in Table 14.

5.2.4. *Project*

We identified 13 project cost drivers (16.88%) that are as follows: Documentation level, Number of programming languages, Type, Process efficiency level, Project management level, Infrastructure, Development restriction, Time restriction, Risk level, Rapid app development, Operational mode Resource level and Lessons learned repository. A description of each cost driver and the primary studies from where they were extracted are presented in Table 15.

5.2.5. *Team*

We identified 15 team cost drivers (19.48%) that are as follows: Domain experience level, Team size, Deployment platform experience level, Team capability, Programming language experience level, Tool experience level, Communication level, Software development experience, Work Team level, Stability level Motivation level Focus factor, Tool experience level and OO experience level In-house experience. A description of each cost driver and the primary studies from where they were extracted are presented in Tables 16 and 17.

5.2.6. *Technology*

We identified six technology cost drivers (7.79%) that are as follows: Authoring tool type, Productivity

Table 9: Complexity size metrics.

Connectivity density	Average connectivity (links) of Web pages in the Web application.	s6, s8, s9, s10, s11, s14, s15, s16, s17, s21, s25, s39, e5	13
Cyclomatic complexity	Number of linearly independent paths in the Web application source code	s6, s8, s9, s10, s11, s15, e5	7
Model collection complexity	Complexity of the Web application model collections	e4, s47	2
Model association complexity	Complexity of the Web application model associations	e4, s47	2
Model link complexity	Complexity of the Web application model links	e4, s47	2
Page complexity	Average number of different types of media on Web pages	e5, ss12	2
Component complexity	It measures the complexity of components of a model used during the development of a Web application	s47, e4	2
Total complexity	Overall complexity of a Web application	e7	1
Adaptation complexity	Complexity for adapt existing parts of a Web application	s57	1
New complexity	Complexity of new features of a Web application	s57	1
Data usage complexity	Complexity of the usage of the Web application data	s57	1
Data flow complexity	Complexity of the flows of the Web application data	s57	1
Cohesion complexity	Complexity of the Web application data cohesion	s57	1
Interface complexity	Complexity of the Web application interfaces with legacy systems	s57	1
Control flow complexity	Complexity of the Web application control flows	s57	1
Class complexity	Complexity of the Web application source code classes	s57	1
Layout complexity	Complexity of Web pages' layout	s35	1
Input complexity	Complexity of the Web application inputs	s35	1
Output complexity	Complexity of the Web application outputs	s35	1

Table 10: Product cost drivers - part 1.

Cost driver	Description	References	F
Structure	It measures the way the documents of a Web application are linked	s4, s5, s6, s7, s8, s9, s10, s11, s14, s15, s16, s17, s21, s39, s47, e4, e5	17
Time efficiency level	It determines the time efficiency level demanded by the client	s35, s64, s75, s77, s93, ss1	6
Memory efficiency level	It determines the memory efficiency level demanded by the client	s35, s64, s75, s77, s93, ss1	6
Portability level	It determines the portability level demanded by the client	s57, s75, s77, s93, ss1	5
Integration with legacy systems	It measures the presence or absence of integration with existing systems to be considered during the development of a Web application	s35, s75, s77, s93, ss15	5
Stratum	It measures to what degree a Web application is designed to be read in a direct way	s3, s4, s5, s7	4
Compactness	It measures the perceived compactness demand for a Web application	s3, s4, s5, s7	4
Requirements volatility level	It measures how often the Web application requirements change	s27, s35, ss1, ss15	4
Requirements novelty level	It measures how known by the team is the work demanded to develop the Web application	s27, s75, s77, s93	4
Reliability level	It determines the reliability level demanded by the client	s35, s75, s77, s93	4
Maintainability level	It determines the maintainability level demanded by the client	s27, s57, ss1, ss15	4
Scalability level	It determines the scalability level demanded by the client	s75, s77, s93, ss15	4
Quality level	It determines the quality level demanded by the client	s35, ss15	2
Usability level	It determines the usability level demanded by the client	ss1, ss15	2
Readability level	It determines the readability level demanded by the client	s57	1
Security level	It determines the security level demanded by the client	ss1	1

Table 11: Product cost drivers - part 2.

Cost driver	Description	References	F
Type	It measures the type of a Web application, which can be static or dynamic	s81	1
Concurrency level	It measures level of concurrency that the Web application is to handle	ss1	1
Processing requirements	It measures the presence or absence of complex processing to run a Web application	ss1	1
Database size	It measures the size of the database accessed by a Web application	s64	1
Architecture	It measures the architectural model used to develop a Web application	ss15	1
Installability level	It determines the instalability level demanded by the client	ss1	1
Modularity level	It determines the modularity level demanded by the client	s57	1
Flexibility level	It determines the flexibility level demanded by the client	s57	1
Testability level	It determines the testability level demanded by the client	s57	1
Accessibility level	It determines the accessibility level demanded by the client	ss1	1
Trainability level	It determines the training level demanded by the client	ss1	1
Innovation level	It measures the level of innovation inherent to a Web application	ss15	1
Technical factors	It measures all the technical factors associated with a Web application	ss15	1
Storage constraint	It measures the presence or absence of storage constraints	s35	1
Reusability level	It determines the reusability level demanded by the client	s57	1

Table 12: Product cost drivers - part 3.

Cost driver	Description	References	F
Robustness level	It determines the robustness level demanded by the client	s35	1
Design volatility	It measures how often the Web application requirements change	s35	1
Experience level	It determines the average team experience level demanded by the client	ss15	1
Requirements clarity level	It measures how easy is for the team to understand the Web application requirements	ss15	1

Table 13: Client cost drivers.

Cost driver	Description	References	F
Availability level	It measures how often the client will be available to meet with the development team	s35, e7, ss15	3
IT literacy	it measures the knowledge of the client about Information Technology and the domain in which the Web application is developed	s26, ss15	2
Mapped workflows	It measures the presence or absence of mapping about the workflows that are to be incorporated into the Web application to be developed.	ss15	1
Personality	It measures how easy is for the development team to deal with the client, specially regarding the client's personality	e7	1

Table 14: Development company cost drivers.

Cost driver	Description	References	F
SPI program	It measures the presence or absence of a software process improvement program in the company responsible for developing the Web application	s53, s54, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91	12
Metrics' program	It measures the presence or absence of a metrics' program in the company responsible for developing the Web application	s53, s54, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91	12
Number of projects in parallel	It measures the projects carried out at the same time by the company responsible for developing the Web application	s75, s77, s93	3
Software reuse	It measures the extent to which the company employ software reuse practices	s35, ss1, ss6	3

level, Novelty level, Platform volatility level, Difficulty level and Platform support level. A description of each cost driver and the primary studies from where they were extracted are presented in Table 18.

6. Discussion

The Web effort predictors' taxonomy (Section 1) and the organized body of knowledge (Section 5) presented herein have implications for both academia and industry, which are discussed next.

6.1. *Implications for academia*

The taxonomy and the organized body of knowledge (the result from applying our taxonomy to classify the 98 primary studies) can be used by researchers to identify literature of interest. For example, whenever a researcher needs to read literature related to size metrics that consider the number of pages, it is possible to use the results presented in Table 4 as a starting point (50 studies identified).

Researchers can also use the results of this paper to select the predictors to be considered when investigating new effort estimation approaches, as all the studies classified herein have supporting empirical evidence; the most frequent predictors can lead to the investigation of stronger estimation models (empirical wise).

The results from this paper also indicate possible gaps that can be addressed through new research. For example, Table 4 shows that only three studies reported the usage of *node count* as a size metric. This can be interpreted in three different ways: i) this size metric is already considered stable; ii) this size metric was found not to be a relevant predictor to estimate the effort of Web applications; and iii) this size metric still needs to be further investigated within the Web effort estimation context. The

Table 15: Project cost drivers.

Cost driver	Description	References	F
Documentation level	Level of documentation in a Web project	s35, s53, s54, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91, e7, ss12	15
Number of programming languages	Number of different programming languages used in a Web project	s54, s55, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91, ss6	13
Type	The type of a Web project, which can be a new or an enhancement project	s53, s54, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91, e7	13
Process efficiency level	Efficiency of the process employed in a Web project	s26, s56, s63, s87, ss12	5
Project management level	The extent to which good practices of project management are employed in a Web application project	s26, s27	2
Infrastructure	The presence or absence of adequate infrastructure to carry out the Web application project	s26	1
Development restriction	Presence or absence of development restrictions regarding a Web project	ss15	1
Time restriction	Presence or absence of time restrictions regarding a Web project	ss15	1
Risk level	The risk level associated with a Web project	e7	1
Rapid app development	It measures the presence of absence of rapid app development practices	s35	1
Operational mode	The operational model employed to develop the Web application, which can be collocated or distributed	s35	1
Resource level	Level of resources of a Web application project	ss15	1
Lessons learned repository	Presence or absence of a repository to record lessons learned from past finished projects	ss15	1

Table 16: Team cost drivers.

Cost driver	Description	References	F
Domain experience level	The average experience of the team members on the domain in which the Web application is developed	s6, s8, s9, s10, s11, s14, s15, s16, s17, s21, s26, s43, s35, s39, s50, s54, s55, s56, s63, s64, s65, s66, s67, s71, s72, s75, s77, s81, s84, s85, s87, s90, s91, s93, e5, e7, ss1, ss6, ss15	39
Team size	Number of team members	s50, s54, s55, s64, s65, s66, s67, s71, s72, s81, s84, s85, s90, s91, e7, ss15	16
Deployment platform experience level	The average experience of the team members on a deployment platform required in a Web application project. It must be measured for each required deployment platform	s35, s64, s75, s77, s93, ss1	6
Team capability	It measures in average how skillful are the members of a development team	s27, s26, s35, s64, ss6, ss15	6
Programming language experience level	The average experience of the team members on a language required in a Web application project. It must be measured for each required language	s64, s75, s77, s93, ss1, ss15	6
Tool experience level	The experience of the team members on a tool required in a Web project. It must be measured for each required tool	s64, s75, s77, s93, ss15	6
Communication level	Level of effectiveness of the communication within the development team	s27, s35, ss15	3
Software development experience	The average experience of the team members on developing software.	s35, ss1, ss15	3
Work Team level	The ability level to work as team	s26	1
Stability level	The frequency in which members get out and in from a development team	s35	1
Motivation level	Level of motivation of the development team	ss1	1

Table 17: Team cost drivers.

Cost driver	Description	References	F
Focus factor	Commitment level of the team members in a period of time	ss1, ss15	2
Tool experience level	The average experience of the team members on a tool required in a Web application project. It must be measured for each required tool	ss1	1
OO experience level	The average experience of the team members on object-oriented analysis	ss1	1
In-house experience	The average experience of the team members on the company processes	ss15	1

Table 18: Technology cost drivers.

Cost driver	Description	References	F
Authoring tool type	type of the authoring tool used during a Web application development	s6, s8, s9, s10, s11, s14, s15, s16, s17, s21, s39, s40	12
Productivity level	It measures the gain in productivity provided by a technology required in a Web project	s75, s77, s93, s96	4
Novelty level	It measures the novelty of a technology (programming languages, tools and deployment platforms) required in a Web project	s75, s77, s93	3
Platform volatility level	It measures the volatility of a platform required in a Web project	s35, s64	2
Difficulty level	It measures the difficulty to learn how to apply a technology required in a Web project	s26, s57	2
Platform support level	It measures the support that a platform required in a Web project has	s35	1

same is applicable to other predictors, such as data *Web points* (Table 7), *class complexity* (Table 9) and *architecture* (Table 11).

Another point to be highlighted is that a classification scheme, like our taxonomy, is expected to evolve over time, which demands the effort of the research community related to the addressed topic [21]; Mendes *et al.*'s taxonomy [11] was evolved by ours, i.e. our taxonomy can be evolved in the future. So, we believe that as long as new related literature is produced, maybe there will be the need to incorporate new categories into the taxonomy put forward in this paper, so that the body of knowledge will also be updated, accounting for new categories and studies.

6.2. *Implications for industry*

This paper's results can support practitioners in two different ways. First, like researchers, practitioners can identify literature of interest by consulting the studies classified through our taxonomy (see Section 5). Second, it can help practitioners to select predictors to estimate the effort of Web applications.

To exemplify how practitioners can use our results, we present the following example:

> A team needs to estimate the effort required to develop a Web-based management information system for a higher education institution. The team has been developing Web applications for many years and using different expert-based estimation techniques to estimate development effort. Despite the fact that the team is very experienced in both Web software development and estimation, this is the first time the team develops an application in the education domain.

In the aforementioned example, the team is not able to rely only on its own experience, since it has never developed a Web application in the education domain. Therefore, the team could use our organized body of knowledge to select the size metrics and cost drivers to be used as inputs during the estimation process. To do so, the team could select the predictors in two different ways:

- Select the most frequently used predictors, considering a predefined threshold. The team in the aforementioned example could consider all the predictors identified in more than 15 studies, which would result in the following predictors (as per Section 5): *Web page count*, *media count*, *component count*, *reused component count*, *high feature count*, *low feature count*, *structure*, *domain experience level* and *team size*.

- Select relevant predictors based on contextual information. The team in the aforementioned example talked to the client company's personnel and realized that there is a low level of IT literacy and the company personnel does not seem to be willing to closely collaborate with them. So, the team suspects that these two aspects will very likely impact the effort associated with the Web application. Thus, the predictors *IT literacy* and *availability level* are selected, even though they do not comply with the above-defined threshold. Furthermore, they also select *domain experience level* as a predictor as the team is working for the first time in the education domain.

The taxonomy and associated body of knowledge presented herein cover general aspects related to Web software development. However, web applications can be developed in different ways, wherein

parts of an application are based on existing frameworks (e.g. spring[g] and JSF[h]), while other parts are developed from the scratch. Both the taxonomy and the body of knowledge can support practitioners to estimate the effort of web applications that involve both framework-based code and completely new code. This is possible due to the fact that reuse is accounted for in the length-related metrics. Furthermore, the other three types of size metrics and the cost drivers in the aggregated body of knowledge are independent of technology, i.e. they are useful in both situations.

7. Validity threats and limitations

The taxonomy presented herein was mainly based on the results from an SLR, on an existing taxonomy and on expert knowledge (third author expertise). We believe that the main limitations of this study are related to the SLR results and the expert knowledge.

The participation of only one expert is a limitation of this work because the knowledge of more experts might improve the taxonomy's usefulness and correctness. Considering that any knowledge classification scheme is a community effort that evolves over time, we expect the taxonomy presented herein to evolve through the gathering of additional expert knowledge and further research. However, as a counter point, the third author of this paper has been to date the researcher who contributed the most in the Web effort estimation field. Thus, her contribution towards the proposed taxonomy has made a significant impact towards the results presented herein.

Although the taxonomy and aggregated body of knowledge may be useful for practitioners estimating effort of any type of Web application, the impact of frameworks on Web development is not fully incorporated, since it has not been covered by existing literature on Web effort estimation. Thus, we believe that this is a limitation that can be addressed with further research.

We updated the SLR by Azhar *et al.* in this paper, i.e. our investigation also has the following limitations associated with the SLR research method:

- **Coverage of the search string** - This type of threat is related to the efficiency of the applied search string to reach relevant primary studies. To mitigate this threat, Azhar *et al.* [1] designed a comprehensive search string. The adaptation we performed to fetch studies published after February 2012 did not affect the coverage of the original study. On the other hand, the fact that we just applied the search string in two data sources (Scopus and Compendex/Inspec) may have affected the number of returned studies.

- **Study selection** - This type of threat is related to the possibility of a study being classified in different manners by different reviewers. Azhar *et al.* defined clear selection criteria. These selection criteria were adapted by us to only select Web effort estimation related studies published after February 2012. We discussed all the selection criteria to ensure that we shared the same understanding. In addition, all the studies retrieved in this phase were screened at the same time by the two first authors of this paper.

- **Data extraction** - This type of threat is related to the possibility of a study data being extracted in different manners by different reviewers. The first aspect of this work that mitigated this threat is the small number of data elements extracted; only size metrics and cost drivers. In addition, we designed a spreadsheet that was used by the two first authors during the data extraction.

[g] projects.spring.io/spring-framework/
[h] javaserverfaces.java.net

Finally, the data extraction for all the primary studies was carried out at the same by the two first authors.

8. Conclusion

This paper presents a taxonomy of Web effort predictors, which was based on a previous taxonomy [11], on an SLR [1] that was updated herein and on expert knowledge. The proposed taxonomy was designed by using the method proposed by Usman *et al.* [20].

To validate our taxonomy and demonstrate its utility, we benchmarked our proposal to Mendes *et al.*'s taxonomy [11] and we also used it to classify 165 unique Web predictors identified in 98 studies. We also presented some examples of how to use both the taxonomy and the organized body of knowledge (i.e. the classification result of the 98 studies). Our proposal can indicate gaps in the existing literature, as well as it can help researchers and practitioners to identify literature of interest, as well as help practitioners to select predictors, supporting effort estimation of Web applications.

The taxonomy presented herein can be extended in different ways. For example, it can be extended to account for predictors that also impact quality, design, and maintenance of Web applications. It is also possible to extend it to account for other elements related to effort estimation within the Web engineering context, such as effort/resource estimation techniques and effort/resource estimate accuracy metrics. Finally, additional research can provide more details about the impact of frameworks on the effort to develop Web applications. Therefore, we intend to extend our taxonomy accounting for Web predictors, effort estimation techniques and accuracy metrics within the Web resource estimation context.

Acknowledgments

We would like to thank CNPq, UFPI, and INES for partially supporting this work.

References

1. D. Azhar, E. Mendes, and P. Riddle. A systematic review of web resource estimation. In *Proceedings of the 8th International Conference on Predictive Models in Software Engineering - PROMISE '12*, pages 49–58, 2012.
2. S. Bayona-Oré, J. A. Calvo-Manzano, G. Cuevas, and T. San-Feliu. Critical success factors taxonomy for software process deployment. *Software Quality Control*, 22(1):21–48, Mar. 2014.
3. P. Bourque and R. E. Farley, editors. *Guide to the Software Engineering Body of Knowledge v3*. IEEE Computer Society, 2013.
4. A. Ginige and S. Murugesan. Web engineering: An introduction. *MultiMedia, IEEE*, 8(1):14–18, 2001.
5. R. L. Glass and I. Vessey. Contemporary application-domain taxonomies. *Software, IEEE*, 12(4):63–76, Jul 1995.
6. B. H. Kwasnik. The role of classification in knowledge representation and discovery. *Library Trends*, 48(1):22–47, 1999.
7. C. Linnaeus. *System of nature through the three kingdoms of nature, according to classes, orders, genera and species, with characters, differences, synonyms, places (in Latin)*. Laurentius Salvius, -, 10th edition, 1758.
8. A. McDonald and R. Welland. Web engineering in practice. In *Proceedings of the fourth WWW10 Workshop on Web Engineering*, pages 21–30, 2001.
9. E. Mendes. *Cost Estimation Techniques for Web Projects*. IGI Publishing, 2007.

10. E. Mendes. Predicting web development effort using a bayesian network. In *Proceedings of the 11th International Conference on Evaluation and Assessment in Software Engineering*, EASE'07, pages 83–93. British Computer Society, 2007.
11. E. Mendes, S. Counsell, and N. Mosley. Towards a taxonomy of hypermedia and web application size metrics. In D. Lowe and M. Gaedke, editors, *Web Engineering*, volume 3579 of *Lecture Notes in Computer Science*, pages 110–123. Springer Berlin Heidelberg, 2005.
12. E. Mendes, N. Mosley, and S. Counsell. Investigating web size metrics for early web cost estimation. *J. Syst. Softw.*, 77(2):157–172, Aug. 2005.
13. E. Mendes, N. Mosley, and S. Counsell. The need for web engineering: An introduction. In E. Mendes and N. Mosley, editors, *Web Engineering*, pages 1–27. Springer Berlin Heidelberg, 2006.
14. T. E. Moffitt. Adolescence-limited and life-course-persistent antisocial behavior: A developmental taxonomy. *Psychological Review*, 100(4):674–701, Oct 1993.
15. K. Moløkken-Østvold and M. Jørgensen. Group processes in software effort estimation. *Empirical Software Engineering*, 9(4):315–334, 2004.
16. J. Moses and J. Clifford. Learning how to improve effort estimation in small software development companies. In *Proceedings of the 24th Annual International on Computer Software and Applications Conference (COMPSAC)*, pages 522–527. IEEE, 2000.
17. D. J. Reifer. Web development: estimating quick-to-market software. *IEEE software*, 17(6):57–64, 2000.
18. D. Smite, C. Wohlin, Z. Galvina, and R. Prikladnicki. An empirically based terminology and taxonomy for global software engineering. *Empirical Software Engineering*, 19(1):105–153, 2014.
19. P. Umbers and G. Miles. Resource estimation for web applications. In *Proceedings of the 10th International Symposium Software Metrics*, METRICS '04, pages 370–381. IEEE Computer Society, 2004.
20. M. Usman, R. Britto, J. Börstler, and E. Mendes. Taxonomies in software engineering: A systematic mapping study and a revised taxonomy development method. *Information and Software Technology*, 85:43 – 59, 2017.
21. S. Vegas, N. Juristo, and V. Basili. Maturing software engineering knowledge through classifications: A case study on unit testing techniques. *Software Engineering, IEEE Transactions on*, 35(4):551–565, July 2009.
22. G. R. Wheaton. Development of a taxonomy of human performance: A review of classificatory systems relating to tasks and performance. Technical report, American Institute for Research, Washington DC, 1968.
23. C. Wohlin. Writing for synthesis of evidence in empirical software engineering. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '14*, pages 46:1–46:4, New York, NY, USA, 2014. ACM.
24. C. Wohlin and A. Aurum. Towards a decision-making structure for selecting a research design in empirical software engineering. *Empirical Software Engineering*, pages 1–29, 2014.