Journal of Web Engineering, Vol. 5, No.2 (2006) 093-120

A *COSMIC-FFP* APPROACH TO PREDICT WEB APPLICATION DEVELOPMENT EFFORT

G. COSTAGLIOLA, S. Di MARTINO, F. FERRUCCI, C. GRAVINO, G. TORTORA, G. VITIELLO

Dipartimento di Matematica e Informatica, Università degli Studi di Salerno Via Ponte Don Melillo, 84084 Fisciano (SA), Italy

E-mail:{gcostagliola,sdimartino,fferrucci,gravino,gvitiello,tortora }@unisa.it

Received April 13, 2005
Revised May 2, 2006

We describe an approach to predict Web application development effort, which is based on the main ideas underlying *COSMIC-FFP* (*Cosmic Full Function Point*). The method is focused on counting data movements and turns out to be suitable for capturing the specific aspects of dynamic Web applications, which are characterized by data movements to and from Web servers. It is based on two measures that can be applied to analysis and design documentation in order to provide early estimations. We also describe the empirical analysis which has been carried out to verify the usefulness of the approach for predicting Web application development effort.

Keywords: Dynamic Web applications, size metrics, empirical validation, effort prediction models
Communicated by: B White & A Ginige

1 Introduction

Web Applications are becoming an essential support for the every-day activities of organizations and institutions which operate in various areas and the demand for these applications is quickly increasing. In the meantime, also the complexity and size of these applications is dramatically augmenting. Thus, the problem of estimating the effort required to develop them represents an emerging issue in the field of Web engineering [6, 40-48, 53-55, 57, 58, 63].

In the context of traditional software engineering many software measures have been defined to gather information about relevant aspects of software products and then manage their development. In particular, several size measures have been conceived to be employed in models to predict the effort and cost needed to design and implement the software. Among them, *Function Points* (*FPs*) have achieved a wide acceptance to estimate the size of business systems and to indirectly predict the effort, cost, and duration of their projects [3]. However, it is widely recognized that this method is no longer adequate for Web-based systems, since it fails in capturing the specific features affecting the size and the effort required for these systems [48, 53, 55, 57]. Nevertheless, the appealing features of the *FP* approach have motivated research efforts to adapt/extend the method to the Web domain. In this context, of special interest are *Web Objects* and *COSMIC-FFP* methods. The former represents an extension of *FPs* characterized by the introduction of four new Web-related components (*multimedia files, Web building blocks, scripts*, and *links*) added to the five

traditional function types of *FPs* [53]. *COSMIC-FFP* is an adaptation of *FPs* originally proposed to size real-time and/or multimedia applications. This method is focused on counting data movements, assumed to require the biggest programming efforts in this specific kind of applications [25]. Nevertheless in the last years several proposals have been formalized meant to apply the *COSMIC-FFP* method to estimate the functional size of object oriented applications [9,27,28,33,34,52], and Web applications [41,55,63]. In the context of object oriented proposals, of special interest is Jenner's approach [33, 34] since it can be easily adopted in any development process exploiting UML *Use Case* and *Sequence Diagrams* during requirement analysis. About the Web domain, Rollo was the first one, in 2000, to advocate the use of the *COSMIC-FFP* method to measure the functional size of Web applications. To support his idea, he provided an example of application for an Internet bank system [55]. Starting from Rollo's suggestion, in 2002, Mendes *et al.* provided a formal method obtained by adapting *COSMIC-FFP* to measure the size of hypermedia Web systems [41]. However, the evolution of Internet technologies, and the consequent shift from "Web content publishing" to "Web applications", is requiring further adaptation of this method, taking into account all the new features gained with technological evolutions.

To address this issue, in this paper we describe an approach meant to apply the *COSMIC-FFP* method to estimate the functional size of dynamic Web applications. Indeed, it is our opinion that, since the *COSMIC-FFP* measure is focused on the counting of data movements, it turns out to be suitable for dynamic applications, which are mainly devoted to handle data movements from users to persistent storage and vice versa.

An appealing feature of our proposal is to allow the Measurer to obtain early size estimation by applying the method on Analysis and Design documents. Indeed, we propose the use of two measures, namely *C-FFPan* and *C-FFPde*. *C-FFPan* is meant to count the data movements at the beginning of the development process to gain a preliminary size estimation, which can be eventually refined by applying *C-FFPde* during design phase. In particular, *C-FFPan* can be obtained from the analysis documents by exploiting the previously cited approach provided by Jenner [33, 34]. In order to count data movements from design documents, we extend the proposals by Rollo and by Mendes *et al.*, defining a set of rules that allow us to measure the functional size of Web applications exploiting the information provided by class diagrams. The diagrams adopt the UML notation for the Web proposed by Conallen [23], which exploits *stereotypes*, *tagged values*, and *constraints* to suitably denote components that are specific for the Web.

In order to assess the effectiveness of the two measures, we also report on the empirical validation we carried out using a dataset of 44 web applications developed by final year academic students. To this aim, the size measures *C-FFPan* and *C-FFPde* have been used as independent variables in an Ordinary Least-Squares (OLS) regression analysis to build effort prediction models. The positive results for prediction accuracy of the derived models encourage us in further investigation.

The remainder of the paper is organized as follows. Section 2 recalls the main concepts of the *COSMIC-FFP* method and the adaptations provided by Jenner for sizing object oriented applications and by Mendes *et al.* for sizing hypermedia systems. Section 3 describes the two measures *C-FFPan* and *C-FFPde*. Section 4 presents the results of the empirical analysis carried out so far while Section

6 contains a description of related work. In the end, Section 7 concludes the paper giving some final remarks and discussion on future work.

2 The COSMIC-FFP Method and some Adaptations

In the present section, we recall the main concepts of the *COSMIC-FFP* method and two adaptations of it, one for sizing object oriented applications [34], and the other for sizing hypermedia systems [41].

2.1 The COSMIC-FFP Method

COSMIC-FFP (COSMIC stands for *COmmon Software Metrics Consortium*, while FFP stands for *Full Function Points*) is a widely adopted method of sizing software, which has been approved as an International Standard (ISO/IEC 19761:2003). It turns out to be particularly suited for real-time and/or multi-layered software. The basic idea underlying this approach is that, for many kinds of software, the biggest programming efforts are devoted to handle data movements, and thus the number of these data movements can provide a meaningful sight of the system size [15]. *COSMIC-FFP* involves applying a set of models, rules, and procedures to Functional User Requirements to obtain a numerical value, which represents the functional size of the software, expressed in terms of *CFSU* (*Cosmic Functional Size Unit*) [25]. In order to apply the method, two models are identified: the *context model* and the *software model*. The former establishes the boundary of the application from its operating environment (see Figure 1.a), and illustrates the generic functional flow of data attributes from a functional perspective. The flow of data attributes is characterized by two directions, back-end and front-end, and by four distinct types of movements: entries and exits, which allow the exchange of data with user, and reads and writes, which allow the exchange of data with the persistent storage hardware.

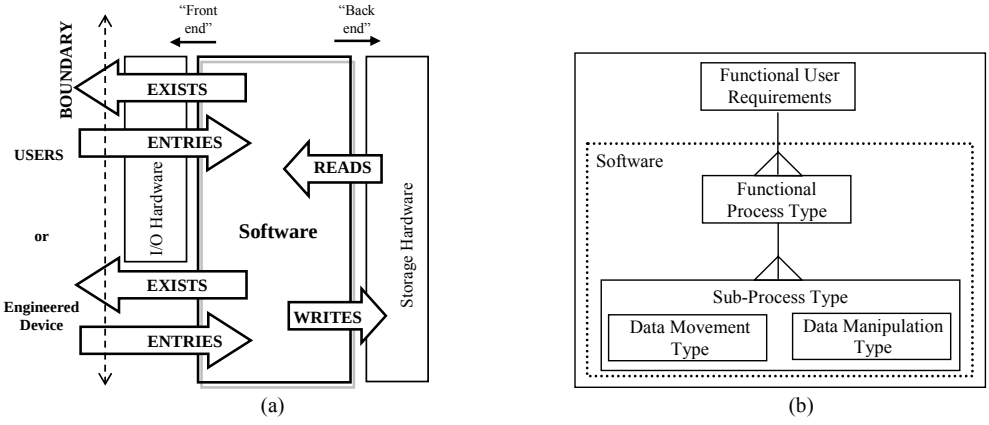


Figure 1. Generic flow of data attributes from functional perspective (a), and generic software model for measuring the functional size (b) [19]

The software model assumes that two general principles hold for the software to be mapped and measured: 1) software takes input and produces useful output to users, and 2) software manipulates pieces of information designated as data groups which consist of data attributes. This software model

allows us to consider the functional user requirements decomposed in a set of functional processes, where each process is a unique set of sub-processes performing either a data movement or a data manipulation (see Figure 1.b). The data movement sub-processes *entry*, *exit*, *read*, *write*, which move data contained in exactly one data group, are considered. In particular, an *entry* moves a data group from a user across the boundary into the functional process (representing the piece of software) that requires it; an *exit* moves a data group from the functional process across the boundary to a user that requires it; a *read* moves a data group from persistent storage to the functional process, which requires it; a *write* moves a data group from the functional process to persistent storage.

According to the idea underlying the *COSMIC-FFP* method, the functional size of software is directly proportional to the number of its data movement sub-processes. This assumption is justified by the nature of the software the method was initially targeted at, namely multi-tiered and/or real-time applications, which are characterized by several data movements.

2.2 Jenner's Adaptation of COSMIC-FFP for Object Oriented Applications

In the last years many proposals have been conceived to apply the *COSMIC-FFP* method to estimate the functional size of object oriented applications [9, 27, 28, 33, 34, 52]. For our purpose Jenner's approach is of great interest since it leads to an estimation of the software size in the early phase of the development. Indeed, Jenner showed that a complete set of use cases, fully specified and represented by their sequence diagrams, can be used to size the corresponding application in terms of *CFSUs* through suitable rules [34]. Figure 2 illustrates the adaptation of the *software model* provided by Jenner.

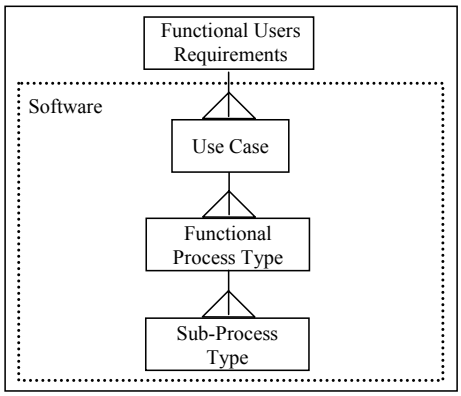


Figure 2. The generic software model to measure the functional size of Web applications provided by Jenner in [34]

To clarify the counting rules of the approach, let us consider the sequence diagram in Figure 3, which is taken from [34] and is referred to an informative system for a library. According to [34], the boundary between the user and the system is represented by the interface objects (e.g., *ReturnWindow* in Figure 3). The rules to count data movements can be summarized as follows.

- Each arrow from the actor to an interface object corresponds to an *entry*, while each arrow from an interface object to the actor corresponds to an *exit*. For example, in Figure 3 the arrows from *Librarian* to *ReturnWindow* determine three *entries*.
- Arrows not involving actors are used to determine *reads* or *writes*. For instance, in Figure 3 the arrow from *ReturnWindow* to *Title* determines a *read*, and the arrow from *ReturnWindow* to *Loan* determines a *write*.

Let us observe that arrows from right to left between intermediary objects representing return of data on a *read* are not counted as further data movements, since they are already considered in the corresponding data request. Indeed, in the sequence diagram of Figure 3 they are omitted. As an example, the number of *CFSUs* obtained from the sequence diagram illustrated in Figure 3 is 8 (3 *entries*, 4 *reads*, 1 *write*).

The method proposed by Jenner seems quite interesting also in the context of Web applications, where several methodologies suggest to exploit use case and sequence diagrams for requirement analysis. However, to the best of our knowledge, Jenner has not provided a systematic empirical analysis to show the effort prediction accuracy of the proposed method neither in the context of object-oriented systems nor for web applications.

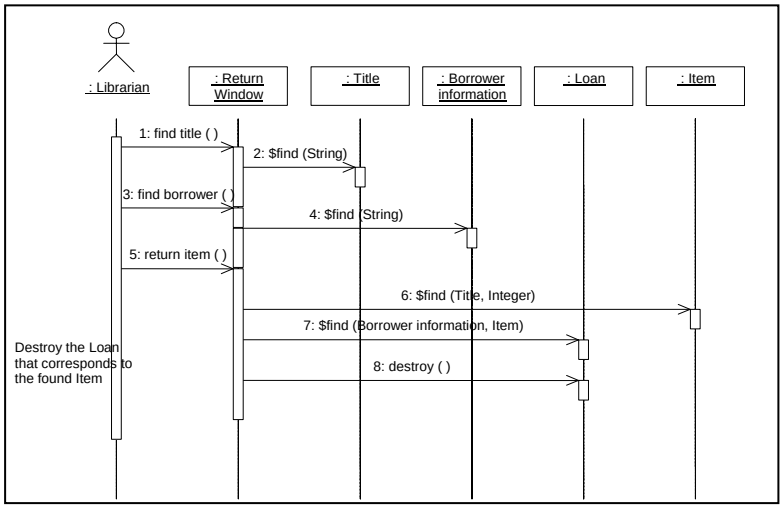


Figure 3. A sequence diagram taken from [34]

2.3 The Adaptation of COSMIC-FFP for Hypermedia Systems Proposed by Mendes et al.

The adaptation of the *COSMIC-FFP* provided by Mendes *et al.* for sizing hypermedia systems uses the context model shown in Figure 4, where *Web Server* is considered as data store [41][1]. They

[1] The model did not present a *write* data movement since the authors did not consider Web applications allowing users to affect the status of the business logic on the Web Server.

focused on the final implementation of hypermedia systems and to count the data movements they provided the following three rules:

1. for each "HREF" tag, count 1 *entry*, 1 *read*, and 1 *exit*;
2. for each Java applet, count 1 *entry* and 1 *exit*;
3. for each JavaScript file, count 1 *entry*.

Using the above rules, 37 Web projects developed by academic students were sized and used to construct an effort prediction model by applying OLS regression. The derived model did not present reasonable prediction, according to the square of the linear correlation coefficient, namely R^2, which determines the goodness of fit of the regression model. Thus, replications of the case study were considered necessary by Mendes *et al.* to find possible bias in the collection of the data and/or for the application of the *COSMIC-FFP* method.

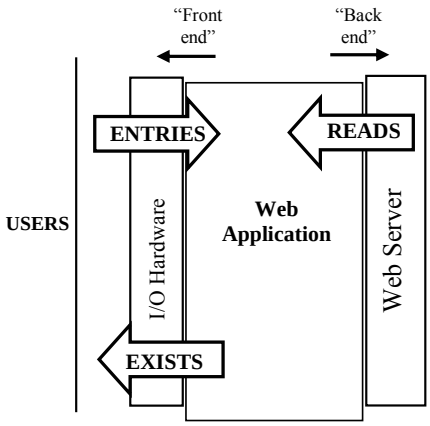


Figure 4: The functional flow of data attributes through Web applications used in [41]

3 Applying the COSMIC-FFP Method to Dynamic Web Applications

In the last years, the Web has become not only a mechanism for sharing information, collaborating and interacting but also a way to access services. The emerging Web technologies leaded to a shift from traditional Web sites, providing navigation mechanisms, to sophisticated and complex Web applications, characterized by functionality affecting the state of the underlying business logic [23]. The growth of complexity and size of the Web applications is motivating the definition of methodologies for supporting Web developers during the development process, and tools for supporting project development planning with reliable cost and effort estimations. In the last decade, several methodologies and visual modeling languages have been proposed for developing Web applications [5, 7, 21, 30, 38, 59]. In particular, we focused our attention on the solution suggested by Conallen [23] whose main concepts are recalled in Section 3.1. In Section 3.2 we provide an approach to estimate the development effort of Web applications, based on the use of the *COSMIC-FFP* method and suitable to be integrated in Conallen's process. Section 3.3 is devoted to illustrate an example of application of the proposed approach.

3.1 Conallen's Proposal for Building Web Applications

Conallen's proposal for building Web applications was especially conceived for applications characterized by client-server interactions. It provides both a symbolic notation and a suitable development process. In particular, the notation is intended as an extension of UML, defined by exploiting the mechanism of *stereotypes*, *tagged values*, and *constraints*. The main goal of the extension is to model appropriate artifacts for the Web at the appropriate level of abstraction and detail, and to enable the interaction between the specific elements for the Web and the rest of the system. The development process suggested by Conallen is based on the *Rational Unified Process* (*RUP*), and provides guidelines about the sequence of developers' activities and the artifacts to be produced at each phase (namely *requirements gathering, requirements analysis, design, implementation*, and *testing*). During requirements gathering and requirements analysis the approach uses Scenarios, Use Case Diagrams, and Sequence Diagrams to capture the concepts of the application domain and to specify the functionality of the systems. Moreover, a class diagram modeling the application domain objects is obtained, which is refined during the design phase in order to make the analysis model realizable in software. Additional classes are added during this phase taking into account the sequence diagrams. The design activity includes the partition of objects in tiers (client, server, and so on) and the definition of client/server Web pages. Thus, a detailed class diagram is obtained, which uses *stereotypes*, *tagged values*, and *constraints* to suitably denote components that are specific to Web applications such as *sever pages*, *client pages*, *forms*, *client script objects*, etc. In Figure 5 the icons denoting some of these components are depicted.

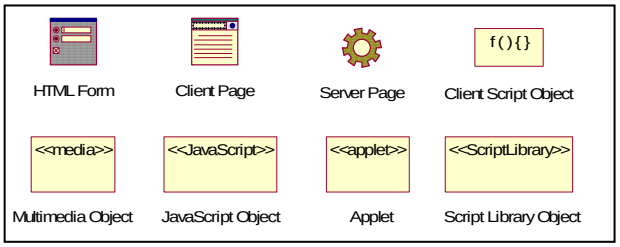


Figure 5. Some icons representing Web components according to Conallen's UML extension

Use Case Insert Contract
ID: InsertContract
Actors: Administrator
Entry condition: The Administrator requests the operation insert contract
Flow of events:
The system presents a form.
The Administrator compiles the form by inputting the information on the contract
The system saves the information on the contract into the database.
Exit condition: the Administrator receives the notification.

Figure 6: The use case modeling the Insert Contract functionality related to an e-procurement system

As an example, let us consider some diagrams modeling part of an e-procurement Web application. A Use Case description for the functionality concerning the insertion of a new contract

by the administrator is provided in Figure 6 while Figure 7 depicts the corresponding sequence diagram.

In particular, the arrow labeled *select()* from the actor *Administrator* to the boundary object *InsertContract* specifies the request for the operation insert contract. Then, the control object *InsertContractControl* creates the boundary object representing the form (named *InsertContractForm*), which is then filled in by the actor *Adminstrator*. Successively, the control object saves the contract in the database which is represented by the entity object *DBContract*. Finally, *InsertContractControl* creates the boundary object *InsertNotifcation* which is displayed to *Administrator*.

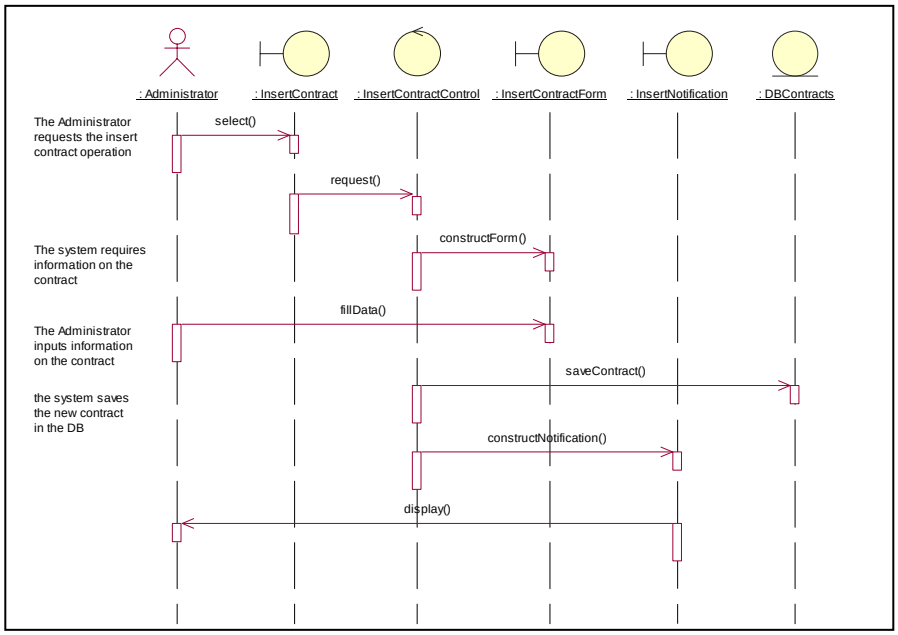
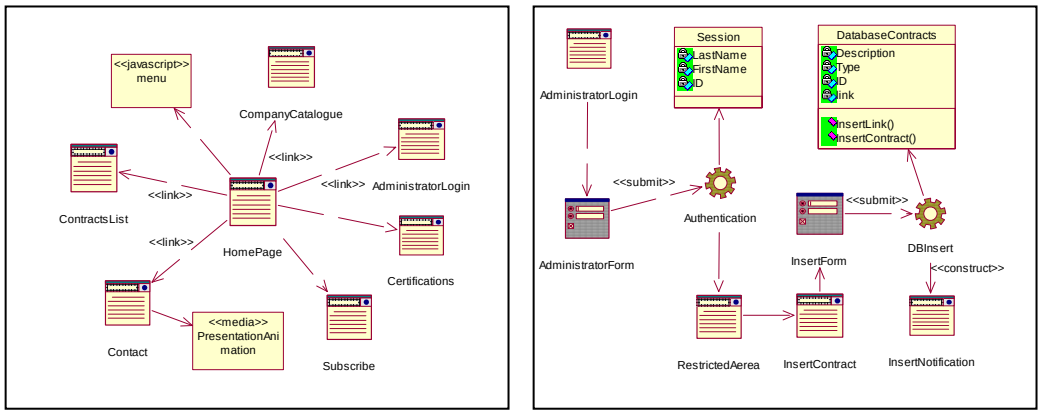


Figure 7: The sequence diagram for the use case Insert Contract of Figure 6

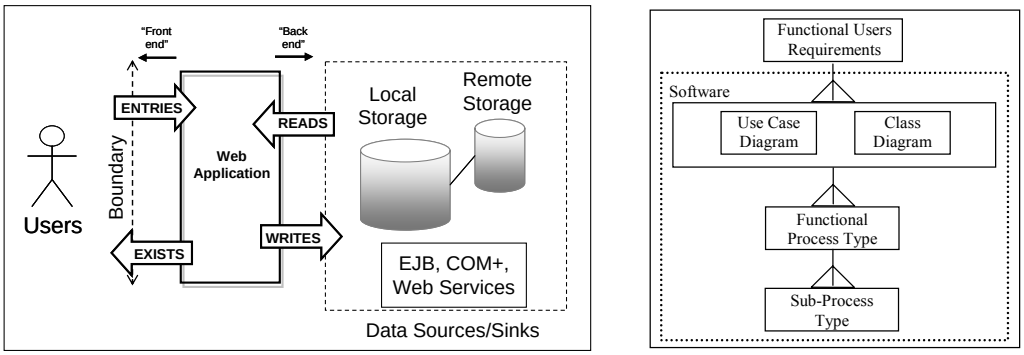
The class diagram depicted in Figure 8.a models the main functionalities provided by the e-procurement system, while the one in Figure 8.b is referred to the specific use case for the insertion of a new contract.

In particular, looking at the class diagram in Figure 8.a, from the client page *HomePage*, six client pages can be reached, namely *Certifications*, *CompanyCatalogue*, *ContractsList*, *Contract*, *Subscribe*, *AdministrationLogin*. *Contract* contains a request for a media which is specified by the stereotype <<media>>, while *HomePage* contains a client script. In the class diagram of Figure 8.b the *Administrator* accesses the restricted area by specifying his/her data through the HTML form *AdministratorForm* contained in the client page *AdministratorLogin*. The server page *Authentication* verifies whether or not the user is registered. When logged the *Administrator* can select the insertion operation and access the client page *InsertContract* which contains the HTML form *InsertForm*. The *Administrator* fills in the form and submits the new contract. Then, the server page *DBInsert* interacts with the database and inserts the information on the new contract. A notification is sent back to the user as an HTML page (i.e., *InsertNotification*).



(a) (b)

Figure 8: The UML class diagram modeling the functionalities of an e-procurement system (a), and the UML class diagram modeling the insertion of a new contract (b)



(a) (b)

Figure 9: The functional flow of data attributes through Web applications (a), and a generic software model to measure the functional size of Web applications (b)

3.2 *The Proposed COSMIC-FFP Approach for Estimating the Development Effort of Web Applications*

The consideration on the fact that dynamic web applications are usually mainly devoted to handle data movements (from users to persistent storage and vice versa), makes them potentially suitable to be measured with the *COSMIC-FFP* approach [55]. In order to apply the method, the *context model* and the *software model* have been suitably adapted. In particular, the *flow of data attributes* for Web applications, executing on a Web server, gives rise to the context model depicted in Figure 9.a.

Indeed, Web applications are bounded in the back-end direction by *Data Sources/Sinks*, and in the front-end direction by the *Users*:

- *Data Sources/Sinks* component encompasses all the modules suited to provide or gain data from the Web application, such as a Web Service, a COM+ or ASPX module, an Enterprise JavaBean, a local file system or a (possibly remote) database.

- *Users* include all the actors suited to provide/consume information managed by the web application.

Compared with the context model provided by Mendes et al. in [41], this context model is characterized by the introduction of the data movement "WRITE" which takes into account the dynamic nature of web applications. The *software model* depicted in Figure 9.b shows that the data movement sub-processes are identified by analyzing fully-specified use case and class diagrams. Indeed, our aim was to provide an approach able to ensure early effort estimation by using analysis and design documents. Thus, we propose the use of two measures, named *C-FFPan* and *C-FFPde*, meant to be applied on the analysis and design documents, respectively. *C-FFPan* allows a Measurer to have a preliminary size estimation, at the beginning of the development process, by exploiting use case diagrams and the corresponding sequence diagrams produced during the requirements analysis. This counting is based on the previously described rules provided by Jenner to determine the data movements (*entry*, *exit*, *read*, and *write*). The application of *C-FFPde* in the design phase adds further details to the estimation obtained during the analysis phase. To this aim we have provided counting rules, intended as an extension of the ones provided by *Mendes et al* in [41], and meant to be applied on UML class diagrams depicted with Conallen's stereotypes for the Web [23]. A description of these rules follows.

1. For each stereotype *Server Page* representing an artifact used to produce a dynamic Web page, count 1 *entry* + 1 *read* + 1 *exit*. In this case, a form allows users to input data and request a dynamic page (*entry*). The Web server elaborates the input of the user through the server-side script or compiled module (*read*) and produces a Web page which is sent to the user (*exit*). Count an additional *read* for each additional elaboration performed to accomplish the functionality (e.g., the checking of login and password, the writing of a cookie, etc…). This information can be obtained by analyzing the corresponding use case description.
2. For each stereotype *Server Page* representing an artifact used to modify persistent data through the Web server, count 1 *entry* + 1 *write* + 1 *exit*. The user inputs data through a form (*entry*), the data is written through the Web server (*write*) and the result is shown to the user (*exit*). Count an additional *read* for each additional elaboration performed to accomplish the functionality. This information can be obtained by analyzing the corresponding use case description.
3. For each stereotype *Client Page*, count 1 *entry* + 1 *read* + 1 *exit*. Indeed, an entry is sent to the application by requesting the client page (*entry*), the page is read from the Web server (*read*) and then shown to the user (*exit*).
4. For each stereotype *ClientScript Object*, count 1 *entry*.
5. For each UML class stereotyped with <<*applet*>>, <<*active X*>>, <<*plug-in*>>, count 1 *entry* + 1 *exit*. The *entry* is considered to run it and the *exit* to show it [22].
6. For each UML class derived from the stereotyped class <<*media*>>, which is visualized after an explicit request of the client, count 1 *entry* + 1 *read* + 1 *exit*. In other words, the media is considered as another Web page downloaded from the server when it is requested.
7. For each stereotype *Script Library*, representing an external application to be integrated/invoked, such as a business tier module (COM+, EJB), a Web Service, or a library

routine, count 1 *entry* + 1 *read* + 1 *exit*. If the reference requires parameter passing, count 1 *entry* + 1 *read* + 1 *write* + 1 *exit* [23].

The sum of all the identified data movements, expressed in terms of *CFSUs*, gives an early size estimation of the Web Application to the Measurer.

It is worth noting that rules 1, 2, and 3 are specifically conceived to deal with dynamic aspects of Web applications, rule 6 refers to multimedia components and rules 3, 4, and 5 take into account elements common to static Web applications. In particular, the latter rules are analogous to the ones provided by Mendes *et al.* in [41] to measure hypermedia Web applications.

To exemplify how to apply the *C-FFPan* and *C-FFPde* measures for sizing dynamic Web applications, let us consider again the UML diagrams depicted in Figure 6-8. Let us determine *C-FFPan* by applying Jenner's rules described in Section 2.2 on the sequence diagram of Figure 7. In particular, the arrow from *Administrator* to *InsertContract* and the arrow from *Administrator* to *InsertContractForm* determine 2 entries, and the arrow from *InsertNotification* to *Administrator* determines 1 exit. The arrow from *InsertContract* to *InsertContractControl*, the arrow from *InsertContractControl* to *InsertContractForm* and the arrow from *InsertContractControl* to *InsertNotification* determine 3 reads, and the arrow from *InsertContractControl* to *DBContracts* determines 1 write. Thus, the number of *CFSUs* obtained from the sequence diagram of the specific functionality is 7 (2 entries, 1 exit, 3 reads, 1 write).

C-FFPde is determined by applying rules 1-7 described above on the class diagrams depicted in Figure 8. In particular, from Figure 8.a, by using rule 3, we obtain 21 *CFSUs* due to the presence of 7 client pages. The presence of a client script in the *HomePage* determines the application of rule 4, and then one more *CFSU*. Finally, the application of rule 6 determines further 3 *CFSUs*, since a media is requested by the client page *Contact*. Thus, for this class diagram we have a total of 25 *CFSUs.* About the class diagram modeling the insertion of a new contract for the e-procurement application (see Figure 8.b), the description of the corresponding use case (see Figure 6) can provide us further insight in the comprehension of the diagram and in the identification of data movements. The presence of the server pages *Authentication* determines the application of rule 1, resulting in 3 *CFSUs*. Rule 2 is instead applied considering the server page *DBInsert*, determining other 3 *CFSUs*. Finally, the presence of the static Web page *InsertContract* which contains the HTML form *InsertForm*, causes the application of rule 3, counting further 3 *CFSUs*. Thus, the total counting for the considered piece of design documentation is 9 *CFSUs*.

4 Empirical Evaluation

An empirical analysis has been performed to establish whether the proposed applications of the *COSMIC-FFP* method can be effectively adopted to predict the development effort of Web based systems. In the following, we first describe how we carried out the case study and analyze the factors which might affect its validity, then we present and discuss the obtained empirical results.

4.1 The Case Study

The case study is based on 44 web applications developed by students of three undergraduate Software Engineering courses held in three subsequent academic years, at the University of Salerno.

The students involved in the case study were familiar with databases design and DBMS, with Java programming and with common Web development technologies (e.g., HTML, Javascripts, CSS, JSP, etc.) since, before the Software Engineering course, they passed the exams on Databases, Programming Languages, and Web Technologies (each course covers 9 ECTSs, according to the *European Credit Transfer System*, where one ECTS corresponds to about 25 hours of workload). In particular, they had practiced the development of web applications as part of the final exam of the course on Web Technologies. The course of Software Engineering (9 ECTSs) aimed to provide them with a systematic approach to Web application design and development. Indeed, during the course, students were instructed on object oriented methodology and UML notation [19], and on Web development by using Conallen's approach [23]. As part of the laboratory activities a coursework was also undertaken: students designed and built a personal Web site with a secure area guaranteeing a restricted access to data stored in a database. This coursework was aimed to reduce learning effects and to classify student's skill on Conallen's notation. The case study is based on the projects developed as final assignment of the course, where students were required to apply Conallen's approach to design and develop a Web application. Students were organized in groups, each composed of 5 undergraduate students of the course on Software Engineering and one postgraduate student acting as project manager. In order to get expertise uniformity among groups, we exploited the information about the grades achieved by the students in the courses of Databases, Web Technologies, and Programming Languages, together with the coursework score. Indeed, the student allocation in the groups was determined in agreement with the following procedure.

- The subjects who had achieved the highest scores in the Web Technologies exam were first distributed among the groups (one for each group).
- The remaining subjects were sorted in decreasing order with respect to their coursework score. Then, starting from top, one subject was randomly assigned to each group.
- Then, again the remaining subjects were sorted in decreasing order with respect to their grades in Databases, and starting from top, one subject was randomly assigned to each group.
- The same process was applied to the others taking into account their grades in Programming Languages.
- Finally, the remaining subjects were randomly assigned to each group.

The project managers were students of a master course on Software Engineering, where they were instructed on project management and size estimation methods. In particular, they carried out a coursework to practice on the application of *COSMIC-FFP* on analysis and design documents.

The developed projects fall in three main categories:

- e-commerce (on-line shop for selling toys, or gifts, or books and cd, flowers or wines, etc…)
- e-learning (web applications modules for self-assessment, management of courseware, etc…)

- Web Portals (job finders, hobbies, dining guides, bed and breakfast guide, book reviews, etc…)

A useful support to carry out the case study was the e-learning platform employed in the Faculty of Science of the University of Salerno. Indeed, it allowed us to keep track of all the deliverables and the crucial data related to the projects. In particular, at the end of each week, any project manager provided us the documentation so far produced and the information about the development effort, expressed in terms of hours, taking into account the reports daily gathered from each team member.

Three milestones were established for the documentation, namely Analysis, Design and Deployment. In the following, we describe the deliverables required for each of them.

1 **Analysis**: The Project Managers had to deliver Analysis documents, namely Scenarios, Use Case Diagrams, Object Diagrams, Sequence Diagrams, Activity Diagrams, and the calculated number of *CFSU*s, obtained by applying the rules related to the *C-FFPan* method.

2 **Design**: The Project Managers had to deliver Design documents, namely Class Diagrams, Component Diagrams, Activity Diagrams and the calculated number of *CFSU*s, obtained by applying the rules related to the *C-FFPde* method.

3 **Deployment**: The Project Managers had to deliver the developed and tested code.

It is worth noting that one of the authors verified the consistency of the information provided by each group with respect to the software documentation and cross-checked all the calculated *CFSU*s.

4.2 The Case Study Validity

As it is widely recognized, several factors can bias the results of a case study in the context of empirical software engineering. In particular, the main factors are related to the subjects involved in the case study, the type of applications developed, the learning effects, and the lack of standardization [8,36,41]. In the following, we will describe how we have tackled them in order to mitigate their biasing effects on our case study.

- ***Subjects involved in the case study***. The scientific literature has often debated on the industrial relevance of results coming from empirical studies with students [6, 12, 20, 31, 41, 45]. However, in the context of web development it is also recognized that professional developers are usually young programmers, coming straight from university [6, 20, 41, 45, 54]. Thus, their skill is very close to the subjects considered in our case study, which were final-year undergraduate students.
- ***Type of Web applications***. In order to obtain significant results, the projects involved in the case study should be representative of the systems developed in the real world. The Web applications considered in our case study are quite similar to the ones typically developed by web companies, both in terms of application domain (e-commerce, e-learning, and Web portals), and in terms of complexity/dimension (in mean 39 server-side pages per project). Consequently, based also on our experience with students' stages in local companies it is our opinion that the considered projects can be representative of small to medium size Web applications.

- ***Learning effects***. As described in Section 4.1 several actions have been undertaken to reduce the learning effects consisting in the development of courseworks and useful projects in previous exams.
- ***Lack of standardization***. Some main difficulties to carry out a case study similar to ours are related to the collection of the information about the software measures and the effort [39, 45, 60]. In order to mitigate the biases in the effort collection each team member daily provided to the project manager the information about his/her development effort, and weekly each project manager reported us the sum of the efforts for the team. The effort was expressed in terms of hours and students were required to include only the actual time spent to realize the project (thus they did not consider possible learning activities). As for the measures, authors defined a template to be filled in by the project managers, in order to collect all the significant information to calculate the number of *CFSU*s. All the project managers were instructed on the rules to apply the *COSMIC-FFP* method. Moreover, one of the authors analyzed the filled templates and the provided analysis and design documents, in order to cross-check the calculation of the *CFSU*s for each project.

4.3 Empirical validation: the method and the accuracy evaluation criteria

In order to perform the empirical validation of the proposed methods, we have applied an Ordinary Least-Squares (OLS) regression analysis, by using as independent variable the measure *COSMIC-FFP*, denoted by *C-FFPan* when it is calculated from analysis documents and by *C-FFPde* when it is derived from design documents. As it is known, the linear regression analysis determines the equation of a line which interpolates data and can be used to predict the development effort. The aim of the validation process is to show that the predicted effort is a useful estimation of the actual development effort required. To this end, we have performed a *multiple-fold cross validation*, partitioning the whole data set into two randomly selected sets: the training set for model building and the test set for model evaluation. Indeed, when the accuracy of the model is computed using the same data set used to build the prediction model, the accuracy evaluation is considered optimistic [18]. Cross validation is widely used in the literature to validate cost estimation model when dealing with small datasets (see, e.g. [15, 17, 29].

To assess the acceptability of the derived effort prediction models, we have used some summary measures, namely *MMRE* and *Pred*(*0.25*) [24], together with *boxplots of residuals* and *boxplots of z* as suggested by Kitchenham *et al.* in [37]. *MMRE* and *Pred*(*0.25*) are considered the de facto standard evaluation criteria to assess the accuracy of software prediction models as stated in [18, 24, 40] and they have been used for many years in many comparisons [11, 15, 16, 17, 34, 35, 49, 50, 57, 58]. In the following we briefly recall the main concepts underlying the *MMRE* and *Pred*(0.25).

The *Magnitude of Relative Error* [24] is defined as

$$MRE = |EFHreal - EFHpred| / EFHreal$$

where *EFHreal* and *EFHpred* are the actual and the predicted efforts, respectively. The rationale behind this measure is that the gravity of the absolute error is proportional to the size of the observations. This value has been calculated for each observation in any test set, using the models derived for the variables *C-FFPan* and *C-FFPde*. Then, for each test set, we have calculated the *Mean of MRE* (*MMRE*) to measure the aggregation of *MRE* over the observations. Moreover, we

have also taken into account *MdMRE*, a measure of the central tendency which is less sensitive to extreme values [45].

The *prediction at level 0.25* [24] is defined as

$$Pred\ (0.25) = k\ /N$$

where k is the number of observations whose *MRE* is less than or equal to *0.25*, and N is the total number of observations. Once accuracy has been separately calculated for each test set, the resulting values have been aggregated across all the sets. According to Conte *et al.*, a good effort prediction model should have a $MMRE \leq 0.25$ and $Pred(0.25) \geq 0.75$ (i.e., at least 75% of the predicted values should fall within 25% of their actual values) [24].

The analysis with the above summary statistics has been complemented by the analysis of boxplots of residuals (*EFHreal—EFHpred*) and the boxplots of *z* (where *z= EFHpred/EFHreal*) [37]. Kitchenham *et al*. recommend these graphical analyses to provide a better insight on the effectiveness of a prediction model. Indeed, they give a good indication of the distribution of the error terms and can help to understand the behavior of the summary statistics [37].

Table 1 The data for the 44 Web development projects

Obs	*EFH*	*C-FFPan*	*C-FFPde*	Obs	*EFH*	*C-FFPan*	*C-FFPde*
1	70	64	99	23	91	66	91
2	129	321	593	24	108	112	241
3	104	173	402	25	128	272	430
4	127	233	419	26	135	101	179
5	171	509	780	27	165	356	421
6	125	240	353	28	133	250	462
7	110	53	165	29	168	202	525
8	102	156	283	30	163	169	745
9	84	47	86	31	172	306	807
10	70	83	111	32	160	277	391
11	159	166	263	33	152	233	530
12	168	299	327	34	99	111	99
13	64	97	85	35	183	248	520
14	119	161	251	36	120	88	269
15	118	196	305	37	111	258	563
16	131	221	302	38	76	49	84
17	141	172	426	39	105	118	187
18	135	242	435	40	170	205	379
19	167	323	414	41	148	178	543
20	72	70	110	42	153	364	577
21	145	378	550	43	135	188	383
22	62	95	115	44	131	205	355

4.4 The Results: Model Construction and Evaluation

Table 1 reports the data of the 44 projects collected from the analysis and design documentations. A descriptive statistics has been performed for the variable *Effort* (denoted by *EFH*), expressed in terms of person-hours, and the variables *C-FFPan* and *C-FFPde*, expressed in terms of *CFSU*s (see Table 2).

Table 2 Descriptive statistics of EFH, and size expressed in *C-FFPan* and in *C-FFPde*

	Obs	MIN	MAX	MEAN	STD. DEV.
EFH	44	62	183	126.80	33.51
C-FFPan	44	47	509	196.70	103.77
C-FFPde	44	84	807	355.80	194.97

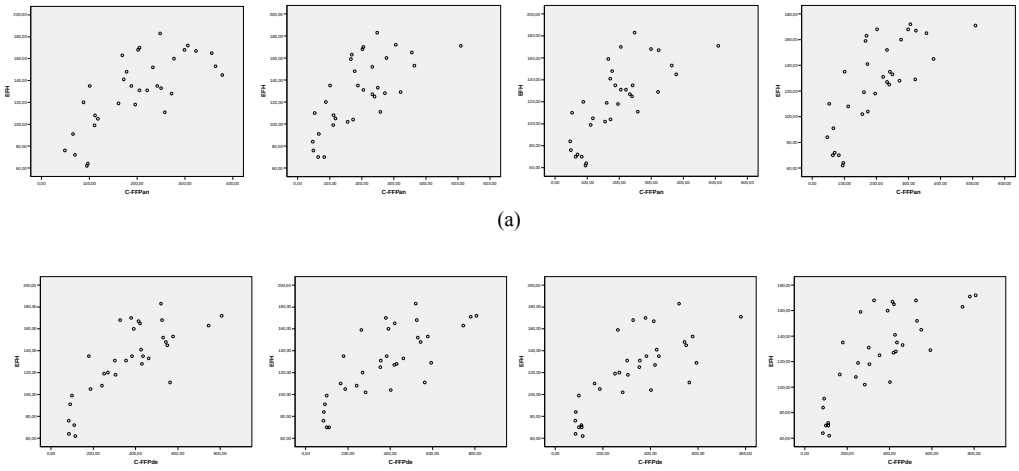
To apply the multiple-fold cross validation, we have partitioned the dataset in 4 randomly test sets of equal size, and then for each test set we have considered the remaining 33 systems as training set in order to build the estimation model.

In order to perform an OLS regression analysis we have verified the following assumptions for each training set: *linearity* (i.e., the existence of a linear relationship between the independent variable and the dependent variable); *homoscedasticity* (i.e., the constant variance of the error terms for all the values of the independent variable); *residual normality* (i.e., the normal distribution of the error terms); *residual uncorrelation* (i.e., error terms are uncorrelated for consecutive observations). In the following we will report the analysis carried out to verify the assumptions.

- ***Linearity***. Figure 10.a and Figure 10.b illustrate the scatter plots obtained from the OLS regression applied to the four training sets, by considering *EFH* as dependent variable and *C-FFPan* and *C-FFPde*, respectively, as independent variables. For the two measures, the scatter plots show a positive linear relationship between the variables involved. It is easy to note that the linearity is more evident for *C-FFPde*.
- ***Homoscedasticity***. From the scatter plots depicted in Figure 11 we can observe that the residuals fall within a horizontal band centered on 0. However, some patterns may be noted in the plots related to *C-FFPan* training sets 2, 3 and 4, and to *C-FFPde* training set 4 (Fig. 11.b). Thus, we further investigated the homoscedasticity assumption, performing a *Breush-Pagan* Test [13], with the homoscedasticity of the error terms as null hypothesis. As reported in Table 3, the assumption can be considered to be verified since the *p-value* of the statistic (i.e., *Sign*) for the four training sets is greater than 0.05 and thus the null hypothesis cannot be rejected.
- ***Normality***. The analysis of Q-Q plots (see Figure 12) reveals that some observations in the training sets 2, 3, and 4 for both *C-FFPan* and *C-FFPde* are not very close to the straight line. Thus, in order to verify the normality assumption, we have used the *Shapiro-Wilk* Test [56], by considering as null hypothesis the normality of error terms. As reported in Table 3, the assumption can be considered to be verified since the *p-value* of the statistic for the four training sets is greater than 0.05 and thus the null hypothesis cannot be rejected.

- ***Uncorrelation***. The uncorrelation of residuals for consecutive observations has been verified by a Durbin-Watson statistic which provided a value close to 2 for each training set (see Table 3). Thus, we can assume that the residuals are uncorrelated.

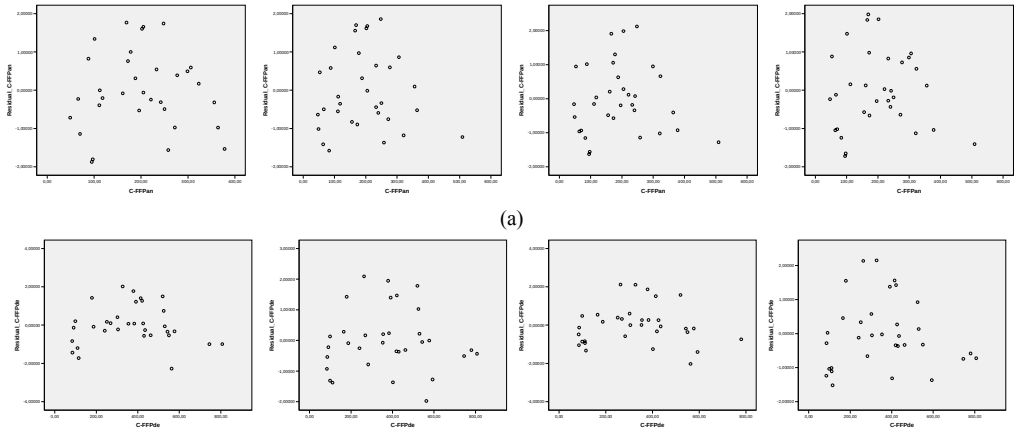
The above observations suggested that a linear regression analysis of *EFH* and *C-FFPde* (*EFH* and *CFFPan*, respectively) can be performed.



(a)

(b)

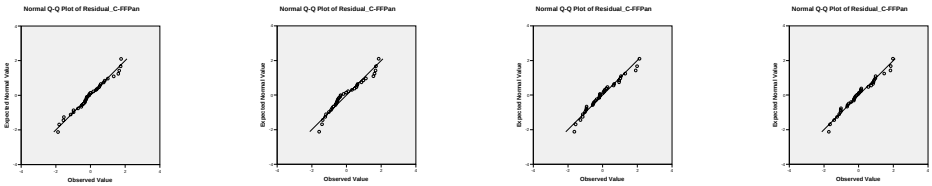
Figure 10. The scatter plots for EFH and C-FFPan (a), EFH and C-FFPde (b), resulting from the OLS regression applied to the four training sets



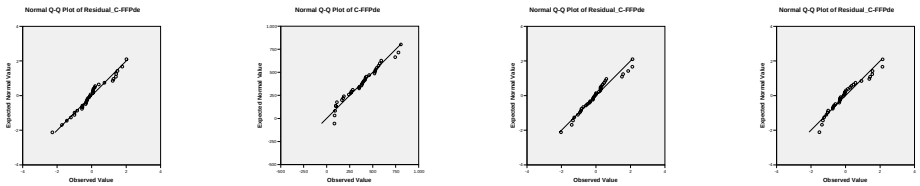
(a)

(b)

Figure 11. The scatter plots for residuals and C-FFPan (a), residuals and C-FFPde (b), resulting from the OLS regression applied to the four training sets



(a)



(b)

Figure 12. The Q-Q plots of the residuals for C-FFPan (a), and C-FFPde (b), resulting from the OLS regression applied to the four training sets

Table 3. The Durbin-Watson values, the Breush-Pagan homoscedasticity tests and the Shapiro-Wilk tests resulting from the OLS regression applied to the four training sets

	Durbin-Watson Value							
	Training set 1		Training set 2		Training set 3		Training set 4	
C-FFPan	1.619		1.522		1.663		1.444	
C-FFPde	2.055		1.714		1.742		1.818	
	Breush-Pagan homoscedasticity Test							
	Statistic	Sign.	Statistic	Sign.	Statistic	Sign.	Statistic	Sign.
C-FFPan	0.307	0.580	0.006	0.939	0.001	0.982	0.740	0.390
C-FFPde	0.010	0.919	0.004	0.952	0.530	0.467	0.761	0.383
	Shapiro-Wilk Normality Test							
	Statistic	Sign.	Statistic	Sign.	Statistic	Sign.	Statistic	Sign.
C-FFPan	0.970	0.476	0.943	0.083	0.960	0.255	0.967	0.394
C-FFPde	0.969	0.463	0.946	0.101	0.958	0.219	0.941	0.072

Table 4 shows the results of the OLS regression carried out with each training set, where the data concerning some crucial indicators are reported. We can observe that the linear regression analysis shows a higher R^2 value for *C-FFPde* with respect to the R^2 value for *C-FFPan*. As an example, let us consider the training set no.1. For *C-FFPde* we have an R^2=0.599, which indicates that 59.9% is the amount of the variance of the dependent variable *EFH* that is explained by the model related to *C-FFPde*, whereas for *C-FFPan* we have an R^2=0.528 indicating that 52.8% is the amount that is explained by the model related to *C-FFPan*. We can observe a high *F* value and a low *p-value* for both measures, which indicate that the prediction is indeed possible with a high degree of confidence, as also shown by the *p-values* and *t-values* for the corresponding coefficient and the intercept (see Table 4). For training set 1 the equation of the regression model for *C-FFPde* is:

(1) $EFH = 0.136 * C\text{-}FFPde + 81.428$,

where the coefficient 0.136 and the intercept 81.428 are significant at level 0.05, as from the T test.

The equation of the regression model for *C-FFPan* is instead:

(2) $EFH = 0.262 * C\text{-}FFPan + 78.729$,

where the coefficient 0.262 and the intercept 78.729 are significant at level 0.05.

Table 4 The results of the OLS regression analysis for evaluating the EFH using CFFPan and CFFPde

***Training set no.1* - Prediction Model Summary**									
		Value	*Std. Err*	*t-value*	*p-value*	**R^2**	**Std Err**	**F**	**Sign F**
C-FFPan	*Coefficient*	0.262	0.044	5.894	0.000	0.528	23.058	34.74	0.000
	Intercept	78.729	9.758	8.068	0.000				
C-FFPde	*Coefficient*	0.136	0.20	6.810	0.000	0.599	21.253	46.377	0.000
	Intercept	81.428	8.185	9.948	0.000				
***Training set no.2* - Prediction Model Summary**									
		Value	*Std. Err*	*t-value*	*p-value*	**R^2**	**Std Err**	**F**	**Sign F**
C-FFPan	*Coefficient*	0.209	0.039	5.377	0.000	0.483	23.433	28.914	0.000
	Intercept	88.412	8.580	10.304	0.000				
C-FFPde	*Coefficient*	0.119	0.018	6.576	0.000	0.582	21.050	43.247	0.000
	Intercept	84.671	7.672	11.036	0.000				
***Training set no.3* - Prediction Model Summary**									
		Value	*Std. Err*	*t-value*	*p-value*	**R^2**	**Std Err**	**F**	**Sign F**
C-FFPan	*Coefficient*	0.235	0.038	6.177	0.000	0.552	23.223	38.159	0.000
	Intercept	76.484	8.310	9.204	0.000				
C-FFPde	*Coefficient*	0.143	0.022	6.502	0.000	0.577	22.561	42.282	0.000
	Intercept	74.529	8.200	9.089	0.000				
***Training set no.4* - Prediction Model Summary**									
		Value	*Std. Err*	*t-value*	*p-value*	**R^2**	**Std Err**	**F**	**Sign F**
C-FFPan	*Coefficient*	0.237	0.037	6.327	0.000	0.564	23.088	40.035	0.000
	Intercept	78.004	8.542	9.132	0.000				
C-FFPde	*Coefficient*	0.134	0.019	7.028	0.000	0.614	21.702	49.395	0.000
	Intercept	78.347	7.724	10.143	0.000				

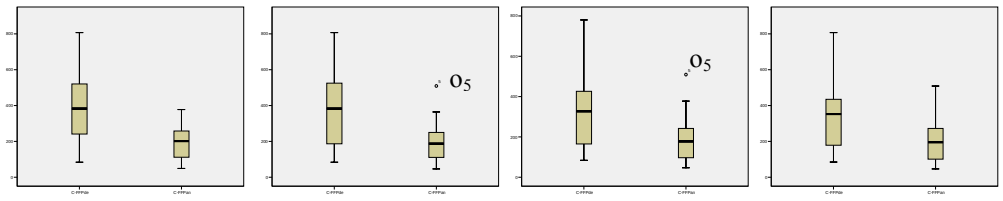


Figure 13. The boxplots derived for the four training sets for variables C-FFPde and C-FFPan

It is worth noting that the analysis we carried out did not reveal extreme values which might unduly influence the models obtained from the regression analysis. Indeed, although the boxplots of

C-FFPde and *C-FFPan* related to each training set (see Figure 13) suggest that system 5 is an "outlier" for training sets 2 and 3, and Cook's distances indicate that it is also an influential observation, the models turn out to be stable in the case we remove this observation (Table 5). This induced us to retain observation 5 in the training sets.

Table 5: The results of the OLS regression analysis for evaluating the EFH using C-FFPan, applied to training sets 2 and 3 by removing observation 5

***Training set no.2* - Prediction Model Summary**									
		Value	*Std. Err*	*t-value*	*p-value*	R^2	**Std Err**	**F**	**Sign F**
C-FFPan	*Coefficient*	0.239	0.046	5.249	0.000	0.479	23.238	27.555	0.000
	Intercept	83.661	9.339	8.958	0.000				
C-FFPde	*Coefficient*	0.121	0.020	6.173	0.000	0.559	21.363	38.102	0.000
	Intercept	84080	8.011	10.496	0.000				
***Training set no.3* - Prediction Model Summary**									
		Value	*Std. Err*	*t-value*	*p-value*	R^2	**Std Err**	**F**	**Sign F**
C-FFPan	*Coefficient*	0.265	0.044	5.987	0.000	0.544	22.976	35.839	0.000
	Intercept	71.794	8.986	7.989	0.000				
C-FFPde	*Coefficient*	0.151	0.025	6.105	0.000	0.554	22.729	37.274	0.000
	Intercept	72.446	8.733	8.296	0.000				

Table 6 The validation result for the first test set

Training set 1							
Obs	***EFF*$_{real}$**	***C-FFPan***	***EFF*$_{pred}$**	***MRE***	***C-FFPde***	***EFF*$_{pred}$**	***MRE***
1	70	64	95	0.36	99	95	0,36
2	129	321	163	0.26	593	162	0,25
3	104	173	124	0.19	402	136	0,31
4	127	233	140	0.10	419	138	0,09
5	171	509	212	0.24	780	187	0,09
6	125	240	142	0.13	353	129	0,03
7	110	53	93	0.16	165	104	0,06
8	102	156	120	0.17	283	120	0,17
9	84	47	91	0.08	86	93	0,11
10	70	83	100	0.44	111	96	0,38
11	159	166	122	0.26	263	117	0,26
		MMRE		0.22	***MMRE***		0.19
		***Pred*(0.25)**		0.64	***Pred*(0.25)**		0.55
		MdMRE		0.19	***MdMRE***		0.17

We have evaluated the accuracy of each model using the corresponding test set and Tables 6-9 report the obtained values of *MRE*, *MMRE*, and *Pred*(0.25). We can observe that the models derived for both the measures *C-FFPan* and *C-FFPde* exhibit an *MMRE* value less than 0.25 (i.e., the acceptable threshold suggested by Conte *et al* in [24]). Moreover, the condition *Pred*(0.25) ≥ 0.75 turns out to be satisfied by the models derived for *C-FFPde*, i.e. based on design documentation, except for the models derived from training sets 1 and 2 which present a *Pred*(0.25) equal to 0.55 and 0.64, respectively. On the contrary, all the models derived for *C-FFPan* have a *Pred*(0.25) less than 0.75.

Table 7 The validation result for the second test set

Training set 2							
Obs	EFF_{real}	***C-FFPan***	EFF_{pred}	***MRE***	***C-FFPde***	EFF_{pred}	***MRE***
12	168	299	155	0.08	327	123	0,27
13	64	97	107	0.67	85	95	0,48
14	119	161	122	0.03	251	114	0,04
15	118	196	131	0.11	305	121	0,02
16	131	221	136	0.04	302	120	0,08
17	141	172	125	0.12	426	135	0,04
18	135	242	141	0.05	435	136	0,01
19	167	323	161	0.04	414	134	0,20
20	72	70	100	0.39	110	98	0,36
21	145	378	174	0.20	550	150	0,03
22	62	95	106	0.72	115	98	0,59
		MMRE		0.22	***MMRE***		0.19
		***Pred*(0.25)**		0.73	***Pred*(0.25)**		0.64
		MdMRE		0.11	***MdMRE***		0.08

Table 8 The validation result for the third test set

Training set 3							
Obs	EFF_{real}	***C-FFPan***	EFF_{pred}	***MRE***	***C-FFPde***	EFF_{pred}	***MRE***
23	91	66	89	0.02	91	88	0,04
24	108	112	101	0.06	241	109	0,01
25	128	272	144	0.12	430	136	0,06
26	135	101	99	0.27	179	100	0,26
27	165	356	166	0.01	421	135	0,18
28	133	250	138	0.04	462	140	0,06
29	168	202	125	0.25	525	149	0,11
30	163	169	117	0.28	745	181	0,11
31	172	306	153	0.11	807	190	0,10
32	160	277	145	0.09	391	130	0,19
33	152	233	134	0.12	530	150	0,01
		MMRE		0.13	***MMRE***		0.10
		***Pred*(0.25)**		0.73	***Pred*(0.25)**		0.91
		MdMRE		0.11	***MdMRE***		0.10

Table 9 The validation result for the fourth test set

Training set 4							
Obs	EFF_{real}	***C-FFPan***	EFF_{pred}	***MRE***	***C-FFPde***	EFF_{pred}	***MRE***
34	99	111	104	0.05	99	92	0,08
35	183	248	137	0.25	520	148	0,19
36	120	88	99	0.18	269	114	0,05
37	111	258	139	0.25	563	154	0,38
38	76	49	90	0.18	84	90	0,18
39	105	118	106	0.01	187	103	0,02
40	170	205	127	0.26	379	129	0,24
41	148	178	120	0.19	543	151	0,02
42	153	364	164	0.07	577	155	0,02
43	135	188	123	0.09	383	130	0,04
44	131	205	127	0.03	355	126	0,04
		MMRE		0.14	***MMRE***		0.11
		***Pred*(0.25)**		0.70	***Pred*(0.25)**		0.90
		MdMRE		0.18	***MdMRE***		0.05

Table 10 Aggregate accuracy evaluation

	Aggregate *MMRE*	**Aggregate *Pred*(0.25)**	**Aggregate *MdMRE***
CFFPan	0.18	0.70	0.15
CFFPde	0.15	0.75	0.10

Once accuracy has been separately calculated for each test set, the resulting values have been aggregated across all four sets. Table 10 reports the results of this analysis. As we can see, the aggregate *MMRE*, the aggregate *Pred(0.25)*, and the aggregate *MdMRE* suggest that *C-FFPde* is good for estimating the development effort based on the threshold provided by Conte *et al* in [24], while *C-FFPan* has a *Pred(0.25)* value less than 0.75. Thus, *C-FFPde* is better than *C-FFPan* to estimate the development effort. This result is consistent with the fact that *C-FFPan* has been obtained from the analysis documents which contain less information on the projects with respect to design documents.

Following the approach suggested in [37] we have also analyzed the boxplots of residuals and the boxplots of *z* (with *z*=estimated effort/actual effort), in order to compare the prediction accuracy of the proposed models built on *C-FFPde* and *C-FFPan* (see Table 4).

The boxplots of residuals and *z* confirm the results obtained with *MMRE*, *Pred*(0.25), and *MdMRE*. Indeed, the tails and the box length of the *C-FFPde* models are smaller than the tails and the box length of the *C-FFPan* models (see Figure 14). This suggests that the models based on *C-FFPde* are characterized by better prediction with respect to the models based on *C-FFPan*. In particular, observing Figure 14.a we can notice that for both measures the boxplots of residuals present a median below zero for training sets 1 and 2 (which indicates that the corresponding models overestimate) and above zero for training sets 3 and 4 (the corresponding models underestimate). Moreover, we can observe that for test set 1 *C-FFPan* model has 2 outliers while *C-FFPde* only one. In test set 2, *C-FFPde* model is characterized by a median closer to zero and a smaller box length. The same situation holds for training set 3. Finally, though for test set 4 *C-FFPde* model has 3 outliers, its box length and tails are smaller than those of *C-FFPan*.

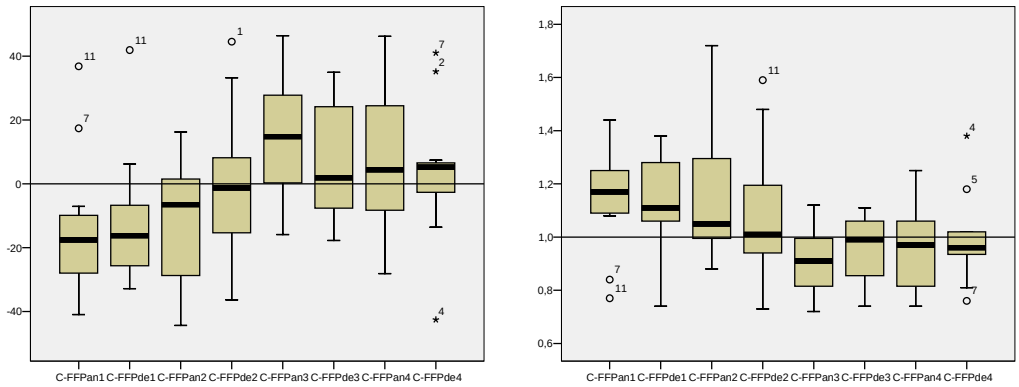


Figure 14. The boxplots of residuals (actual effort – predicted effort) (a), and the boxplots of z (predicted effort/actual effort) (b)

The boxplots of *z* depicted in Figure 14.b show that for both measures the medians present values above one for training sets 1 and 2, and below one for training sets 3 and 4. However, the medians related to *C-FFPan* are more far away from one than those related to *C-FFPde* (especially in test sets 2 and 3). Moreover, though, the boxplots related to *C-FFPde* show outliers for test sets 2 and 4, they are less skewed than the corresponding boxplots associated to *C-FFPan*.

As suggested in [37, 45, 62] we have tested the statistical significance of the results obtained for the proposed models by using paired absolute residuals. To this aim, we have performed both the *Wilcoxon signed rank test* and the *T-Test*, with the following *Null Hypothesis* "the two considered population of absolute residuals have identical distributions". In particular, this kind of test is used to test the hypothesis that the median of the differences in the pairs is zero. The test statistic is the number of positive differences. If the null hypothesis is true, then the numbers of positive and negative differences should be approximately the same. Since the absolute residuals for the model based on *C-FFPde* is not normally distributed (see training set 1 in Table 11) the results obtained with the Wilcoxon signed ranks test should be preferred. As reported in Table 12, the null hypothesis cannot be rejected, except for training set 2, i.e., there is not a statistically significant difference for the absolute residuals obtained with *C-FFPan* and *C-FFPde*.

Table 11. Test of normality for absolute residuals

	Shapiro-Wilk Test			
	C-FFPde		***C-FFPan***	
	Statistic	Sign.	Statistic	Sign.
TS1	0.849	0.042	0.887	0.129
TS2	0.948	0.616	0.934	0.448
TS3	0.903	0.202	0.952	0.673
TS4	0.903	0.202	0.962	0.797

Table 12. Statistical significance tests by using paired absolute residuals of *C-FFPan* and *C-FFPde*

	Wilcoxon signed ranks test		**T-Test**	
	Statistic	Sign.	t	Sign.
TS1	-0.889(a)	0.374	-0.702	0.499
TS2	-2.401(a)	0.016	-3.350	0.007
TS3	-0.978(b)	0.328	-1.113	0.292
TS4	-1.067(b)	0.286	-1.363	0.203

(a) Based on negative ranks. (b) Based on positive ranks.

5 Related Work

The application of the *COSMIC-FFP* method for sizing Web applications has been analyzed by some researchers in the last years [41, 55, 63]. The difficulties of applying the *FP* analysis for sizing an Internet bank system, first suggested to Rollo the use of *COSMIC-FFP*. The successful application of the method to that bank system made Rollo confident that the *COSMIC-FFP* approach is the ideal method for sizing Web applications [55]. However, he did not present an empirical study supporting his thesis, and he did not provide systematic rules to formally apply the method. Subsequently, Mendes *et al.* adapted models and provided some rules for applying the *COSMIC-FFP* measurement to Web hypermedia systems [41]. Their proposal was focused on the final

implementation rather than on analysis and design documentation. They also provided an empirical study carried out with students' projects. The effort prediction model constructed by applying OLS regression did not present reasonable prediction thus suggesting further analysis on the use of the *COSMIC-FFP* method for the effort estimation of Web applications. It is also worth pointing out that they took into account only Web hypermedia systems rather than dynamic Web applications (i.e., applications which affect the status of the business logic on the Web server). The results described in the present paper make us confident that in the case of dynamic Web applications the counting of data movements can be useful for estimating the development effort. This is also intuitive because data movements represent a characterizing feature of this type of applications.

Another interesting proposal has been made by Umbers and Miles who suggested the use of design patterns as a way to simplify the *COSMIC-FFP* measurement process for Web projects [63]. They considered Model-View-Controller, Intercepting Filter, and Data Access Objects as the common patterns used in the context of Web applications, and provided measurement rules to identify the functional processes implemented by each instance of the above patterns. In particular, in the measurement process they identified three types of functional processes (control processes, view processes, and model processes) and provided four principles to count data movements from those processes. Moreover, they used Monte Carlo simulation to mitigate errors judgments in the empirical study based only on three Web applications.

Among the size measures proposed in the last years to predict web application development effort, special interest deserves *Web Objects* [53]. This method proposed by Reifer represents an extension of *Function Points* (*FP*s), the method introduced by Albrecht to estimate software size of business systems in the early phases of the lifecycle [1]. In particular, *Web Objects* introduce four new web-related components (multimedia files, web building blocks, scripts and links), which are used as predictors together with the five traditional function types of *FPs* (external input, external output, external inquiry, internal logical file and external interface file) to compute the functional size of a web application. An empirical study on the application of *Web Objects* was provided in [57], where the results show better performance for the measure with respect to standard *Function Points*. Two cost estimation models based on *Web Objects* have been proposed. In [53] Reifer presented an adaptation of the *COCOMO II* model, named *WebMO*, which uses *Web Objects* as size metrics and 9 cost factors obtained by revising those employed in the context of *COCOMO II* model [10]. Ruhe *et al.* also used *Web Objects* as size metrics in the context of *WebCOBRA* [58], an adaptation of the *COBRA* method [14] for the web.

Besides *COSMIC-FFP* and *Web Objects*, other measures proposed in the literature are extensions/adaptations of the *Function Points*, and aim to exploit the appealing features of the method for predicting the size of Web applications. In particular, Abrahão and Pastor propose the *OOmFPWeb* method [1] which maps the *FP* concepts into the primitives used in the conceptual modelling phase of *OOWS*, a method for producing software for the Web [51]. In a recent work, an initial validation of the proposed size measure has been described [2]. Their analysis differs from that proposed in the present paper since it is focused on the evaluation of the *OOmFPWeb* as functional size measurement method, not as predictor of Web application development effort.

The literature also presents effort estimation modeling techniques which involve the use of variables/predictors specific of Web components. In particular, Mendes *et al.* proposed several size

measures for static and dynamic Web applications, such as number of Web pages, number of graphics, etc. [42, 43, 45, 46, 47]. Using those measures, the authors analyzed the predictive power of several prediction techniques, such as case-based reasoning, linear and stepwise regression, and regression trees using Web applications developed by academic students and by companies.

A different approach has been proposed by Baresi *et al.*, who defined several measures on the basis of attributes obtained from design artifacts [6]. In particular, the authors observed that the design phase is often a critical one and requires an important part of the total development effort, especially when tools for the automatic generation of Web applications are exploited. Indeed, they described a case study with students of an advanced university class to investigate the impact of some attributes, obtained from artifacts designed with W2000, on the total effort required for designing Web applications. However, they were mainly focused at identifying the main attributes affecting the effort required for the design activity performed with W2000. On the other hand, the approach proposed in the present paper aims to identify a measure that taking into account elements of analysis and/or design documents is able to predict the total effort necessary to develop a web application (i.e. the effort for analysis, design, implementation, and testing).

6 Final Remarks

In this paper we have described an approach for estimating the functional size of dynamic Web applications, using the ideas of the *COSMIC-FFP* method during both analysis and design phases. Indeed, as the empirical study shows the measure turns out to be suitable for capturing the dynamic aspects of these applications which are characterized by data movements to and from Web servers. The method we propose has been conceived to be integrated with the development process proposed by Conallen. Thus, during requirement analysis an early effort estimation can be realized by analyzing the sequence diagrams in agreement with the *COSMIC-FFP* rules suggested by Jenner to estimate the size of object oriented systems [33, 34]. During design phase, a more accurate estimation can be performed by examining Conallen's UML class diagrams and by applying a set of rules which have been specifically conceived for dynamic Web applications. These rules represent an extension of the rules proposed by Mendes *et al.* for Web hypermedia systems [41].

The results of the empirical analysis that we have carried out are encouraging, suggesting that the counting of data movements can be useful for estimating the development effort of dynamic Web applications. In particular, the application of the method on design documents exhibits a better performance than its application on analysis documents. This is not surprising, since during analysis fewer details are usually available. For that reason, we suggest to apply the *COSMIC-FFP* method at the beginning of the development process, during the analysis phase, in order to obtain a preliminary effort estimation, which can be later refined during design, when further information is available, by employing the suitable rules defined for class diagrams.

Several research directions can be planned as future work. First of all, further analysis is needed for the assessment of the approach. Data coming from the industrial world are presently being collected in order to verify/confirm the results obtained so far. We intend to use this data also to perform a comparative analysis with respect to *Web Objects* [53]. Finally, it should be interesting to analyze how the method can be integrated in other methodologies for the development of Web applications. Several other methods employ use case and sequence diagrams to gather and analyze

requirements [4, 21, 38]. Thus, for those methods the *C-FFPan* measure can be straightforwardly exploited. As for design documents, the proposed computation of *C-FFPde* is based on the adoption of Conallen's UML notation for the Web. We plan to investigate whether the proposed approach can be generalized in order to apply the data movements counting rules on other visual notations like WebML [21].

References

1 S.M. Abrahão, O. Pastor, "Measuring the functional size of Web applications", in *International Journal of Web Engineering and Technology*, vol. 1, no. 1, 2003.

2 S.M. Abrahão, Geert Poels, O. Pastor, "Evaluating a Functional Size Measurement Method for Web Applications: An Empirical Analysis", in *Proceedings of International Software Metrics Symposium* (*METRICS'04*), Chicago, September 2004, pp 358-369.

3 A.J. Albrecht, "Measuring Application Development Productivity", in *Proceedings of the Joint SHARE/GUIDE/IBM Application Development Symposium*, Monterey, 1979, pp. 83-92.

4 R.D. Banker, R.J. Kaufman, R. Kumar, "An empirical test of object-oriented output measurement metrics in a computer aided software engineering environment", *Journal of Management Information Systems*, vol. 8, no. 3, winter 1991-92.

5 L. Baresi, E. Bianchi, L. Mainetti, M. Maritati, A. Paro, "UWA Hypermedia - User Manual", Technical Report UWA-15, Politecnico di Milano, 2001.

6 L. Baresi, S. Morasca, P. Paolini, "Estimating the Design Effort of Web Applications," in *Proceedings of the International Software Metrics Symposium* (*METRICS'03*), Sydney, 2003, pp. 62-72.

7 L. Baresi, F. Garzotto, P. Paolini, "From Web Sites to Web Applications: New Issues for Conceptual Modeling", in *Proceedings of the International Workshop on The World Wide Web and Conceptual Modeling, co-located with the 19th International Conference on Conceptual Modeling*, Salt Lake City, October 2000, pp. 89-100.

8 V.R. Basili, L.C. Briand, W.L. Melo, "A Validation of Object-Oriented Design Metrics as Quality Indicators," *IEEE Transaction on Software Engineering*, vol. 22, no. 10, pp. 751-761, 1996.

9 V. Bévo V, G. Lévesque and A. Abran, "A pplication de la Mèthode FFP à partir d'une spécification selon la notation UML: compte rendu des premiers essais d'application et questions", *International Workshop on Software Measurement* (IWSM99), Lac Supérieur, September 1999, pp. 230-242.

10 B.W. Boehm, C. Abts, A.W. Brown, S. Chulani, B.K. Clark, W. Horowitz, R. Madachy, D. Reifer, B. Steece, "*Software Cost Estimation with COCOMO II*", Prentice Hall, NJ, 2000.

11 B.W. Boehm, Z. Chen, T. Menzies, D. Port, "Feature Subset Selection Can Improve Software Cost Estimation Accuracy", in *Proceedings of the Workshop on Predictor models in software engineering* (*PROMISE'05*), St. Louis, 2005, pp. 1-6.

12 D. A. Boehm-Davis, L. S. Ross, "Program Design Methodologies and the Software Development Process", *International Journal of Man-Machine Studies*, vol. 36, no.1, pp.1-19, 1992.

13 T.S. Breush, A.R. Pagan, "A simple test for heteroscedasticity and random coefficient variation", in *Econometrica* 47(1979), pp. 1287-1294.

14 L. Briand, K. El Emam, F. Bomarius, "COBRA: A Hybrid Method for Software Cost Estimation, Benchmarking, and Risk Assessment", in *Proceedings of International Conference on Software Engineering* (*ICSE'98*), April 1998, pp. 390-399.

15 L. Briand, K. E. Emam, D. Surmann, I. Wiekzorek, and K. Maxwell "An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques," in *Proceedings of International Conference on Software Engineering* (*ICSE'99*), Los Angeles, May 1999, pp. 313-322.

16 L. Briand, K. E. Emam, I. Wieczorek, "Explaining the Cost of European Space and Military Projects", in *Proceedings of International Conference on Software Engineering* (*ICSE'99*), Los Angeles, May 1999, pp. 303-312.

17 L. Briand, T. Langley, I. Wieczorek, "A Replicated Assessment and Comparison of Common Software Cost Modeling Techniques", *International Software Engineering Research Network Technical Report ISERN-99-15.*

18 L. Briand, I. Wieczorek. Software Resource Estimation. Encyclopedia of Software Engineering. Volume 2. P-Z (2nd ed.), Marciniak, John J. (ed.) New York: John Wiley & Sons, pp. 1160-1196, 2002.

19 B. Bruegge, A. H. Dutoit, "*Object-Oriented Software Engineering: Using UML, Patterns and Java*", (2nd edition), Prentice-Hall, 2003.

20 J. Carver, L. Jaccheri, S. Morasca, F. Shull, "Issues in Using Students in Empirical Studies in Software Engineering Education", in *Proceedings of International Software Metrics Symposium* (*METRICS'03*), Sydney, September 2003, pp. 239-249.

21 S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, M. Matera, "*Designing Data-Intensive Web Applications*", Morgan-Kaufmann, 2002.

22 J. Conallen, "Modelling Web Application Architectures with UML", *Communications of the ACM*, vol. 42, no. 10, pp. 63-70, 1999

23 J. Conallen, *Building Web Applications with UML*, Addison-Wesley Object Technology Series, 1999.

24 D. Conte, H.E. Dunsmore, V.Y. Shen, "*Software Engineering Metrics and Models*", The Benjamin/Cummings Publishing Company, Inc., 1986.

25 COSMIC: *COSMIC-FFP* Measurement manual, version 2.2, http://www.cosmicon.com, 2003.

26 G. Costagliola, F. Ferrucci, C. Gravino, G. Tortora, G. Vitello, "A *COSMIC-FFP* Based Method to Estimate Web Application Development Effort", in *LNCS 3140*, N. Koch, P. Fraternali, and M. Wirsing (Eds.): *ICWE 2004*, Monaco, July 2004, pp. 161-165.

27 H. Diab, M. Frappier, and R. St-Denis, "A Formal Definition of *COSMIC-FFP* for Automated Measurement of ROOM Specifications", in *Proceedings European Conf. Soft. Measurement and ICT Control*, Heidelberg, May 2001, pp. 185-196.

28 H. Diab, F. Koukane, M. Frappier, and R. St-Denis, "McRose: Functional Size Measurement of Rational Rose RealTime", in *Proceedings International Workshop Quantitative Approaches in OO Software Eng*ineering, Malaga, June 2002, pp. 15-24.

29 K.E. Emam, "A Primer on Object-Oriented Measurement", in *Proceedings of IEEE International Sofware Metrics Symposium* (*METRICS'01*), London, 2001, pp. 185-188.

30 F. Garzotto, P. Paolini, D. Schwabe, "HDM: A Model-based Approach to Hypertext Application Design", *ACM Transactions of Information Systems,* vol. 11, no. 1, pp. 1-26, 1993.

31 M. Host, B. Regnell, C. Wholin, "Using Students as Subjects—A Comparative Study of Students and Professionals in Lead-Time Impact Assessment", in *Conference of Empirical Assessment & Evaluation in Software Engineering* (EASE'00), Keele University, UK, 2000.

32 International Function Point Users Group: "Function Point Counting Practices Manual," Release 4.1.1, 2001.

33 M.S. Jenner, "*COSMIC-FFP* and UML: Estimation of the Size of a System specified in UML-Problems of Granularity", in *Proceedings of European Conference Software Measurement and ICT Control*, Heidelberg, May 2001, pp. 173-184.

34 M.S. Jenner, "Automation of Counting of Functional Size Using *COSMIC-FFP* in UML," in *Proceedings of Workshop Software Measurement*, Magdeburg, October 2002, pp. 43-51.

35 B. A. Kitchenham, "A Procedure for Analyzing Unbalanced Datasets", *IEEE Transaction on Software Engineering*, vol. 24 no. 4, pp. 278-301, 1998.

36 B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, K. El Emam, J. Rosenberg "Preliminary Guidelines for Empirical Research in Software Engineering", *IEEE Transactions on Software Engineering*, vol. 28, no. 8, pp. 721 – 734, 2002.

37 B. A. Kitchenham, L. M. Pickard, S. G. MacDonell, M. J. Shepperd, "What accuracy statistics really measure", *IEE Proceedings – Software*, vol. 148, no.3, pp.81-85, 2001.

38 N. Koch, "*Software Engineering for Adaptive Hypermedia Applications*", PhD. Thesis, Reihe Softwaretechnik 12, Uni-Druck Publishing Company, Munich 2001.

39 K.D. Maxwell, "Collecting Data for Comparability: Benchmarking Software Development Productivity", *IEEE Software*, 2001, pp. 22-24.

40 E. Mendes, N. Mosley, S. Counsell, "Do Adaptation Rules Improve Web Cost Estimation?", in *Proceedings of ACM International Conference on Hypertext and Hypermedia*, Nottingham, August 2003, pp. 173-183.

41 E. Mendes, N. Mosley, S. Counsell, "Comparison of Web Size Measures for predicting Web Design and Authoring Effort", *IEE Proceedings-Software* vol. 149, no. 3, pp. 86-92, 2002.

42 E. Mendes, S. Counsell, and N. Mosley, "Measurement and Effort Prediction of Web Applications", in *Proceedings of ICSE Workshop on Web Engineering*, Limerick, June 2000.
43 E. Mendes, N. Mosley, S. Counsell, "Web Metrics – Estimating Design and Authoring Effort", *IEEE Multimedia*, Special Issue on Web Engineering, pp. 50-57, 2001.
44 E. Mendes, I. Watson, C. Triggs, N. Mosley, S. Counsell, "A Comparison of Development Effort Estimation Techniques for Web Hypermedia Applications", in *Proceedings of International Software Metrics Symposium* (*METRICS'02*), Ottawa, Canada, June 2002, pp. 131-140.
45 E. Mendes, I. Watson, C. Triggs, N. Mosley, S. Counsell, "A Comparative Study of Cost Estimation Models for Web Hypermedia Applications", *Empirical Software Engineering* vol. 8, no. 2, pp. 163-196, 2003.
46 E. Mendes, N. Mosley, S. Counsell, "Early Web Size Meausures and Effort Prediction for Web Costimation", in *Proceedings of International Software Metrics Symposium* (*METRICS'03*), Sydney, September 2003, pp. 18-39.
47 E. Mendes, B. Kitchenham, "Further Comparison of Cross-company and Within-company Effort Estimation Models for Web Applications", in *Proceedings of International Software Metrics Symposium* (*METRICS'04*), Chicago, September 2004, pp. 348-357.
48 M. Morisio, I. Stamelos, V. Spahos and D. Romano, "Measuring Functionality and Productivity in Web-based applications: a Case Study", in *Proceedings of the International Software Metrics Symposium* (*METRICS'99*), Boca Raton, November 1999, pp. 111-118.
49 P. Musilek, W. Pedrycz, N. Sun, G. Succi, "On the Sensitivity of COCOMO II Software Cost Estimation Model", in *Proceedings of International Software Metrics Symposium* (*METRICS'02*), Ottawa, Canada, June 2002, pp. 13-20.
50 I. Myrtveit, E. Stensrud, "A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models", *IEEE Transaction on Software Engineering*, vol. 25, no. 4, pp. 510-525, 1999.
51 O. Pastor, S.M. Abrahão, J.J. Fons, "Object-oriented approach to automate Web applications development", in *Proceedings of International Conference on Electronic Commerce and Web Technologies* (*EC-Web'01*), Springer Verlag, Germany, pp. 16–28, 2001.
52 G. Poels, "Definition and Validation of a *COSMIC-FFP* Functional Size Measure for Object-Oriented Systems", *Proceedings of Workshop Quantitative Approaches in OO Soft. Eng*., Darmstadt, Germany, July 2003.
53 D. Reifer, "Web-Development: Estimating Quick-Time-to-Market Software", *IEEE Software*, vol. 17, no. 8, pp. 57-64, November/December 2000.
54 D. Reifer. Ten Deadly Risks in Internet and Intranet Software Development. *IEEE Software*, vol. 18, no. 2, pp. 12–14, 2002.
55 T. Rollo, "Sizing E-Commerce", in *Proceedings of the ACOSM 2000 - Australian Conference on Software Measurement*, Sydney, 2000.
56 P. Royston, "An extension of Shapiro and Wilk's test for normality to large samples", *Applied Statistics* vol. 31, no.2, pp.115-124, 1982.
57 M. Ruhe, R. Jeffery, I. Wieczorek, "Using Web Objects for Estimating Software Development Effort for Web Applications", in *Proceedings of International Software Metrics Symposium* (*METRICS'03*), Sydney, September 2003, pp. 30-37.
58 M. Ruhe, R. Jeffery, I. Wieczorek, "Cost estimation for Web applications", in *Proceedings of International Conference on Software Engineering*, Hilton Portland, May 2003, pp. 285 – 294.
59 D. Schwabe, G. Rossi, "Developing Hypermedia applications using OOHDM", in *Proceedings of Workshop on Hypermedia development Process, Methods and Models, Hypertext´98*, Pittsburgh, 1998.
60 M.J. Shepperd, M. Cartwright, "Predicting with Sparse Data", in *Proceedings of International Software Metrics Symposium* (*METRICS'01*), London, April 2001, pp. 28-39.
61 E. Stensrud, T. Foss, B. Kitchenham, I. Myrtveit, "A Further Empirical Investigation of the Relationship Between *MRE* and Project Size", *Empirical Software Engineering*, vol. 8, no. 2, pp. 139-161, 2003.
62 E. Stensrud, I. Myrtveit, "Human Performance Estimating with analogy and Regression Models: an Empirical Validation", in *Proceedings of International Software Metrics Symposium* (*METRICS'98*), Bethesda, March 1998, pp. 205.
63 P. Umbers, G Miles, "Resource Estimation for Web Applications", in *Proceedings of International Software Metrics Symposium* (*METRICS'04*), Chicago, September 2004, pp. 370-381.