# Software Project Duration Estimation Based on COSMIC Method Applied to Data Flow Diagram

Zoltan Kazi
University of Novi Sad, Technical Faculty, Serbia
zoltan.kazi@tfzr.rs

Ljubica Kazi
University of Novi Sad, Technical Faculty, Serbia
ljubica.kazi@tfzr.rs

**Abstract:** *Business process models are created before the detailed software design, during requirements phase of software development. Even disciplined agile methodology includes business process modeling, before development iterations start. Their use in an estimation of software development duration could be beneficial, because they provide sufficient elements for mapping with software design and development planning. In this paper we propose the method for software project duration estimation based on Common Software Measurement International Consortium (COSMIC) method, applied to data flow diagram. This method is based on data flow diagram analysis and extraction of primitive business procceses,data flows and data stores. The paper contributes with the approach to enhance COSMIC method with calculation of software development process duration based on both data movement-related and data-manipulation-related software functional sub-processes. Data movement-related functional sub-processes are derived from Create, Read, Update, Delete (CRUD) operations assigned to data flows. Calculation of data manipulation-related functional sub-processes duration is derived from number of business processes. The proposed metric enables calculating effort (expressed in Cosmic Functional Points units) and duration (presented with Man/Hours units). An example of the approach application and an empirical study demonstrates the applicability of the approach.*

**Keywords:** *Business process model, CRUD operations, software effort estimation, functional process, data movement, data manipulation.*

*Received September 1, 2020; accepted October 10, 2021*
*https://doi.org/10.34028/iajit/19/4/8*

## 1. Introduction

In software project development process, it is difficult to plan and estimate early because of deficient information [32]. Generally speaking, "early" represents occurring before implementation in any development iteration [8]. Early development stages include activities related to concept, requirements and design [42]. In Software Development Life Cycle (SDLC) models, requirements elicitation and analysis takes primary role among earliest software development phases and activities [8, 10, 38]. In the case of software as part of enterprise information systems [21]), an important part of software requirements analysis is related to the business processes-related knowledge [10], particularly for large-scale projects with special emphasis on activities and data flow [4]. According to [40], domain analysis assists system developers to design valid applications for particular domain, by using modeling. Even model-based approach have some limitations, still it is effective for development risk reduction [42]. A domain model is defined as a "conceptual model capturing the topics related to specific problem domain" [12]. Contemporary business processes modeling is performed within domain engineering (particularly within software product lines [7]) and it results in created reference models [39, 40], in aim to support construction of other models and the development of information systems. Conceptual business process modeling with application of methods such as structured system analysis is used within the software requirements analysis [10, 18], with Data Flow Diagrams (DFD) as a standard notation [21, 44]. Business process models are essential to software development, since other models could be developed based on their generic reference form [19, 39, 40]. They could be mapped into use cases, they are consistent and less variable then use cases and they could be organized hierarchically, with gradual inclusion of details. Aim of this paper is to present a method for calculating software project duration estimation in early phase of software development, based on data flow diagram, which is integrated with Common Software Measurement International Consortium (COSMIC) method.

Khatibi *et al.* [25] presented an overview of the methods for estimating the duration of the software project:

- Algorithmic approaches (number of lines of code; The COCOMO method; Software Lifecycle Management-SLIM); Software Evaluation and Estimation of Resources-SEERSEM; Functional points; COCOMO 2)
- Non-algorithmic approaches (expert assessment;

Classification and Regression Tree (CART); Analogy Based Estimation (ABE).

Contemporary solutions in software estimations include application of soft computing [41], neural networks [33] and machine learning [36].

COSMIC method was developed for measurement of software size, based on functional user requirements mapped to software functions, i.e. functional processes [15]. Functional size of software is calculated according to COSMIC principle of counting data movements in the whole software system, while data movements are not separately addressed. COSMIC method may be used to estimate software project effort, cost, and duration [1, 30] and it been used with business process models as an input, for an early estimation of software size and effort/cost [28] estimation. This is particularly important for ERP systems estimations [34]. Elements of business process models are mapped to COSMIC constructs, with special emphasize on process tree and system boundary communication [24]. Mapping should be performed according to appropriate mapping rules, taking into account ISO standards and BPMN notation [31].

Contributions of this paper are twofold -related to overcoming COSMIC method limitations and proposing improvements comparing to related work. Contributions of this paper are briefly listed:

1. Sources for COSMIC method application, which are business processes and all data flows, external and internal.
2. All data flows (not only those communicating with data stores) are used for data movements calculations. Data movements could be performed with external entities as well, not only internally with processes or with data stores.
3. Data movements functions are derived from actions attached to data flows, where there could be more than one action attached to a single data flow (one graphical data flow from DFD could perform multiple operations upon the data group it consists). In this paper, there is assigned correspondence between graphical data flow with possibly multiple data manipulations.
4. Flexibility of calculation formula with possible application to business processes of any hierarchical level in process decomposition, i.e., available business process knowledge.
5. Flexibility in software development duration estimation, regarding team productivity, with proposing to use variety of speed factors in the calculation formula.
6. Actions that are performed within data movements are expanded with Update and Delete actions.
7. Method for mapping COSMIC elements to DFD business process model elements and formula that uses all elements from DFD to compute functional size and development effort (i.e., software project duration).

Details about contributions are:

1) Sources for COSMIC method application are business processes and all data flows, external and internal. Business processes are taken for data manipulation, while data flows are used for data movement calculations. This way, both data movement and data manipulation segment are included in functional size calculation formula. In COSMIC method, only data movements are taken for calculations and they correspond only to data flows at DFD. Related work [24, 34] consider business processes as a computation source, while data flows are not considered. The need for particular consideration of data manipulation sub-processes is not addressed in [24, 31, 34]. In [31], BPMN data flows are mapped to COSMIC data movements (entry, exit, read, write), but only data movements are used in calculations of software size, just like in COSMIC method.
2) All data flows (not only those communicating with data stores) are used for data movements calculations. Data movements could be performed with external entities as well, not only internally with processes or with data stores. In COSMIC method, only data movements with data stores are included in calculations. Data flows are not considered as a source for computation in [24, 34]. Related work [24] considers system boundaries, but they do not consider data flows with external entities as a source for computation. In [31], incoming flow is mapped to Entry data movement, outgoing flow to exit data movement, while read and write data movements are mapped to resources data flows (i.e., data flows collaborating with internal data stores). Regarding incoming and outgoing flows, it is not clearly explained in [31] if they are related to external entities or they are internal between the system processes.
3) Data movements functions are derived from actions attached to data flows, where there could be more than one action attached to a single data flow (one graphical data flow from DFD could perform multiple operations upon the data group it consists). In this paper, there is assigned correspondence between graphical data flow with possibly multiple data manipulations. In COSMIC method, data manipulation consists of a single data group processed with a single data manipulation action. In related work [24, 34], data flows are not considered as a source calculations. In [31], one business process could include multiple data movements.
4) Flexibility of calculation formula with possible application to business processes of any hierarchical level in process decomposition, i.e., available business process knowledge. COSMIC method has

a horizontally equal set of data movements and gradual inclusion of details according to available knowledge is not supported. In [31, 34], hierarchical decomposition is not addressed. In [31], level of granularity has been emphasized as level of BPMN or Qualigram details, but the concept of primitive process has not been mentioned. In related work [24], process tree is considered and the need for having primitive processes as the most precise source for calculation. It is concluded in [24] that such a precise process tree is not available early, so the calculations in e-Cosmic system support intermediate or top level of hierarchy.

5) Flexibility in software development duration estimation, regarding team productivity, with proposing to use variety of speed factors in the calculation formula, which enables adjustments to particular development team productivity. COSMIC method is not applied to effort estimation, but only to functional size computation, based on functional processes mapping from user requirements. It does not consider any team productivity data, since COSMIC method does not result in computed effort, but functional size. Related work [24, 31] in using COSMIC method with business process models for functional size estimation, but does not perform any effort calculations and, therefore, does not consider any productivity factor.
6) Actions that are performed within data movements are expanded with Update and Delete actions. In [24, 31, 34] there are no other types of actions upon data groups, but in real business-oriented software applications, these operations are supported, as the Create, Read, Update, Delete (CRUD) operations.
7) Method for mapping COSMIC elements to DFD business process model elements and formula that uses all elements from DFD to compute functional size and development effort (i.e., software project duration). In [24, 31, 34], not all business process elements are used, but their methods are very much aligned with COSMIC method having focus on data movements (i.e., data flows and basic CRUD operations-write and read). These related works do not provide any detailed formula for development effort calculations, but they rely of COSMIC principle of counting data movements.

The rest of the paper is organized as follows. Background section explains basic terms-COSMIC method and Data Flow diagram. Related work section provides a short review of software and size estimation methods with special emphasize on methods in software functional requirements elicitation and applications of COSMIC method with business-oriented software. The proposed methodology is introduced with problem definition, which explains the limitations of existing COSMIC method. It also describes approach to applying COSMIC method to data flow diagram elements, mapping of business processes with software functional processes and formula for software project duration calculation. The proposed method is illustrated with an example of method application. Empirical study is described with research methodology and results. Discussion section elaborates the obtained results and the research limitations. Final section provides conclusions and directions for future work.

## 2. Background

### 2.1. COSMIC Method

The COSMIC method is internationally recognized and used for measuring the size of the functional requirements in most software domains [16, 43]. 'COSMIC' stands for the 'Common Software Measurement International Consortium'. Basic idea for moving towards measuring the size of delivered software from software requirements was introduced in 1979 by Albrecht [3] from International Business Machines Corporation (IBM) and the method was named Function Point Analysis (FPA). In aim to improve the FPA, COSMIC group was formed and introduced new method named COSMIC method in 1999. COSMIC method is included in several ISO standards. ISO 19761 defines COSMIC FSM as a standard method for functional size measurements. [31]

The COSMIC method is applied in three phases [15]:

1. *Measurement strategy*-determination what will be measured, which software functional requirements will be included in software.
2. *Mapping*-creating COSMIC model for particular software, where each event ("triggering event") in the world of a user is mapped into a single functional process (that responds to the triggering event).
3. *Measurement*-a functional size is measured in units of Cosmic Function Point (CFP). The size of 1 CFP is defined as a size of a single data movement of any of four types. Measurement is based on application of the COSMIC measurement principle:

"*Functional size of a piece of software is equal to the number of its data movements"* [15].

Each functional process consists of data moving and data manipulating functional sub-processes, but only data movements are used for measurement (Figure 1).

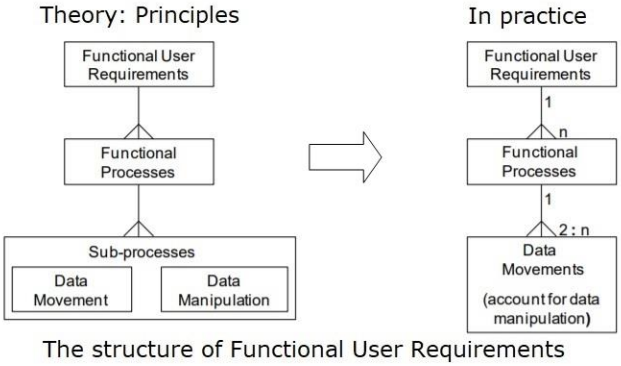



Figure 1. Mapping process in COSMIC method (according to [15]).

Data movement sub-processes move data in and out of the software system with four data movement types: Entry-input of data by functional users; Exit - display of data to the functional users; Write-recording data into persistent storage; Read-reading data from persistent storage (Figure 2).

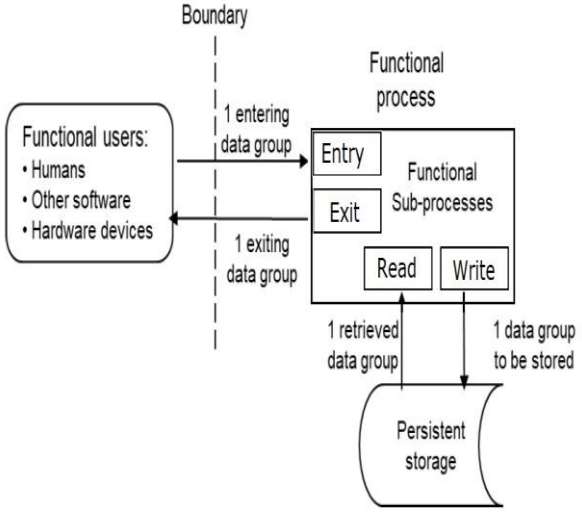



Figure 2. The four data movements in COSMIC method (according to [15]).

## 2.2. Data Flow Diagram

There are several standard business process modelling notations, widely accepted at universities and industrial practice [35]: Business Process Model and Notation, (BPMN) licenced by Object Management Group and currently in version 2.0, EPC (Event-driven Process Chain), Unified Modeling Language-Activity Diagram (UML-AD), Role Activity Diagram (RAD), Integration DEFinition (IDEF). Main process modelling techniques include [21]: flow charts, data flow diagrams (Yourdon's technique [44]), Role activity diagrams, Role interaction diagrams, Gantt chart, IDEF, Petri-net, UML Object-oriented methods, Workflow techniques. Methodologies that include previously mentioned techniques are [21]: Structured Systems Analysis and Design Methodology (SSADM), Soft Systems Methodology (SSM), Graph with Results and Activities Interrelated (GRAI) methodology and Simulation.

Essential artefacts in Yourdon's technique [44] in business process modelling are Data Flow Diagrams (DFD) and Data Dictionary (DD), which describes all objects and elements of the DFD. DFD presents business processes with the data that are shared and exhanged between processes internally and with external entities. Dennis *et al*. [18] described basic elements of DFD: external entities, data flows, business processes, and data stores (presented at Figures 4 and 5). Process modeling is based on the functional decomposition of the system, visualized as a process tree (Figure 3).

Process 0. – context diagram
- Process 1 – first level of decomposition
  - Process 1.1. – second level of decomposition
  - Process 1.2.
  - Process 1.3..
- Process 2
  - Process 2.1.
  - Process 2.2.
- Process3.
  - Process 3.1.
  - Process 3.2.
  - Process 3.3.
  - Process 3.4.

Figure 3. Process tree abstract example.

The process that represents the whole system is presented at the first diagram (context diagram), together with external entities and data flows between the system and external entities. Figure 4. Presents an abstract example of a data flow diagram-context diagram.

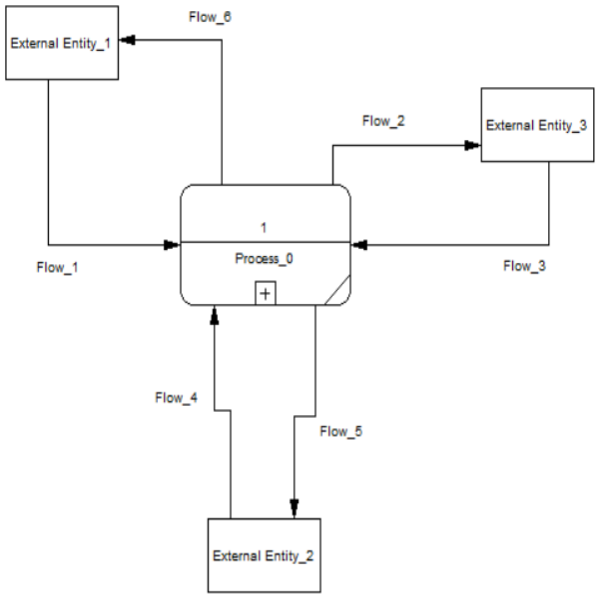



Figure 4. Data flow diagram-context diagram

For each further decomposition there is another data flow diagram, representing the internal organization of the process to be decomposed (Figure 5), with details

that include decomposing processes, data stores (also named„ resources“) and internal data flows. The decomposition process ends with business processes that are elementary and cannot be decomposed (labeled „primitive processes“).

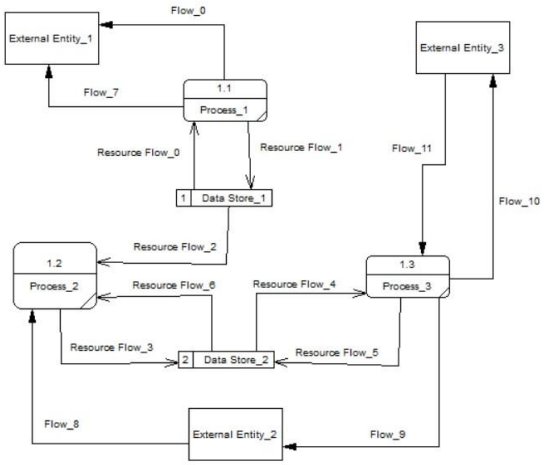



Figure 5. Data flow diagram-decomposition.

Every DFD element has a name and a description, while data flows and data stores have also data items and structure described in the data dictionary. Any business process interact with data stores by data flows (named resource flow). Resource flow properties include attributes related to the access mode (or multiple access modes for one data flow) in interaction with the resource (Figure 6)-read (retreive), enter (create, append), delete (erase), and modify (update, change) (Figure 6).

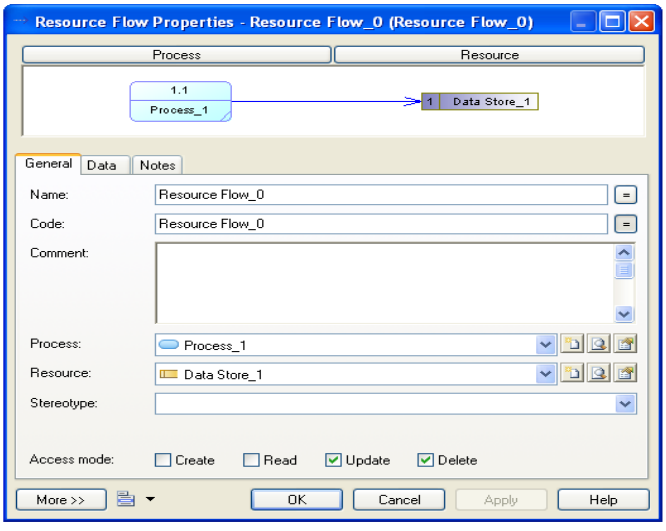



Figure 6. Resource flow attributes and access mode settings.

## 3. Related Work

### 3.1. Software Estimation Methods in Functional Requirements Elicitation

Metrics-based methods use available development artefacts and they are developed for early estimations in software project management. They could address both static and dynamics aspect of a software system [23].

Business process models (such as DFDs) are primarily used to represent core business processes [11], as a basis for multiple technological implementations. Use cases could be derived from business processes represented at business process model [9]. “Business process models, as they are commonly used to gather requirements from the early stages of a project, could be a valuable source of information for functional size measurement” [31]. Issa and Rub [22] emphasized the need to utilize use cases in software project estimation, by deriving them from a business process model, which is technologically independent and represents organizational tasks. Use cases are designed to be automated support for appropriate business processes. Mapping business processes to use cases create set of use cases candidates. Total engagement for software development (based on business process model) is expressed in man-months and can be obtained with using a formula for calculating delivery time based on the number of use cases candidates. Cruz and Da Cruz [19] described the system for automated deriving Use Case and Domain models from BPMN models. Monsalve *et al*. [31] describe a method for using business process models with COSMIC method to compute software functional size. Recent research are directed towards automation of functional size estimation, such as with using UPROM method [6].

Buglione and Gencel [13] describe different types of software size methods that are based on user functional requirements. Function Point Analysis (FPA) was initially designed in 1979 by Albrecht [3] and it is based on the idea of determining size of a software by capturing the amount of functionality presented in software functional requirements. It started a development of methods within the approach of Functional Size Measurement (FSM). Other methods based on functional points are [13]: Common Software Measurement International Consortium Full Function Points (COSMIC FFP), International Function Point Users Group (IFPUG FPA), MarkII FPA, Netherlands Software Metrics Association (NESMA) FSM, Finnish FSM.

Other methods for software size estimation, that are based on functional requirements are methods that present the size in„ Use Case points “[14, 27, 37], „UML points “[26] and„User Story points “[15]. Lavazza *et al*. [29] expressed measurement of software size in functional points by analyzing UML models (use case diagram, component diagram). Angara *et al*. [5] conducted a comparative study of using functional points, COSMIC method and user stories-based functional size measurement.

### 3.2. Using COSMIC Method in Business Software Estimation

COSMIC method has been refined by COSMIC group for business software application domain. The more focused methodology for the functional processes identification is precise within guideline for sizing business application software, version 1.3 (May 2017) [16]:

- Events from the business process environment are mapped into functional processes,
- Data models are analyzed, for each data entity CRUD operations are defined as data movement sub-processes). In this context, different data models are analyzed: E-R diagrams, UML class diagrams and relational database normalization.
- Software user interface design, i.e. screen design, (including analysis of drop-down lists needed for additional data to be supported) were also proposed to be used in analysis.

COSMIC method could be applied to business process-intensive software, such as Enterprise Resource Planning (ERP) and effort estimation, by extraction of business processes, their conversion to CFP and finally to effort, calculated with appropriate conversion factors [34]. Omural and Demirors [34] within the proposed COSMIC EPC method take only business processes and uses different factors in calculations, where factors are selected according to categorization of software functional unit's complexity. Kaya and Demirors [24], propose method e-COSMIC, where elements of business process model are mapped to COSMIC constructs. In e-COSMIC method, key activities are related to detecting system boundary, cross boundary data movements, process tree (hierarchical decomposition) and function allocation diagram.

Monsalve *et al*. [31] proposes a method for measuring software functional size by using International Organization for Standardization (ISO) standard for COSMIC method with business process models from business application domain and real-time domain. This paper proposes modeling rules and mapping rules to make COSMIC elements relate to Business Process Modeling Notation (BPMN) constructs.

## 4. The Proposed Methodology

### 4.1. Problem Definition and Research Goals

There are some limitations of COSMIC method, particularly for the business-oriented software applications:

- Main focus of COSMIC method is on functional sub-processes related to data movements, while data manipulation (data processing with more complex algorithms) were disregarded.
- There is no clear procedure of mapping events from business real-world environment into functional processes of software. It has only been informally mentioned as a needed mapping activity, but without any inclusion of business process modeling.
- In guideline for applying COSMIC method to business-oriented software applications [16], the only models that are mentioned as underlying for functional processes mapping were data models-ER, object-oriented (class diagrams) and relational models (with normalization issues). Data movement functional subprocesses were derived from data entities related to CRUD operations. It could be concluded that [17] gives directions towards data-centric approach.
- Another underlying artefact that was proposed to be used in [17] was user interface screen analysis, particularly for data lists and combo boxes, which require filling and specific processing. It could be concluded that applying COSMIC method, according to [16], requires detailed user interface design.
- Having all previously mentioned in mind, it could be concluded that:
- COSMIC method could not be used in a very early estimation of functional size of a business-oriented software, since it requires more detailed design artefacts (data models, user interface design), which are very close to the implementation. Therefore, usability of COSMIC method in early phases of business-oriented software development could be doubted, according to [16].
- COSMIC method does not consider having business process model as an underlying artefact for deriving functional processes needed for the method application. This is needed for particular use of applying COSMIC method in a very early software development phase, where main focus is on business domain knowledge, not software design.
- As previously noted in [15], it is important to make a distinction between different levels of preciseness in functional requirements specification, which affects granularity and, as well as calculation accuracy. Business process models, especially data flow diagrams, provide organized gradual inclusion of details with the concept of process decomposition and process tree. This way, business process knowledge preciseness could be better organized in levels, which results in calculations within well-established business process modeling methodology.

Based on COSMIC method application, estimation of software development cost could be performed [28]. Effort could be calculated in [28] based on sum of data movements' development efforts.

Research Goals of this paper are:

- To enable better preciseness and flexibility in early software development effort (duration) estimation, based on business process model analysis.
- To propose a method of using business process models in early estimation of software size and development effort (i.e., duration), with using COSMIC method as a basis,
- To overcome limitations of COSMIC method, with enabling all business process elements to be used in software effort estimation.

## 4.2. The Proposed Method

In the proposed method in this paper, business domain knowledge is used for creating business process model. It is presented with data flow diagrams organized hierarchically with process tree and additionally described with data dictionary. If there is enough knowledge available, diagrams are developed to the level where primitive processes are presented. In that case, each primitive business process is directly mapped into a single software functional process.

According to COSMIC, each data movement has appropriate data part (data group) and functional part (i.e. action with that data group). In the approach proposed in this paper, each data flow could contain multiple data movements (i.e., data groups and actions) and actions are not just of Read and Write type, but all CRUD (Create, Read, Update, Delete) operations are supported within data flows.

According to COSMIC method, software functional process consists of data manipulation sub-process and data movement sub-process. In the proposed approach in this paper, each business process is considered to be compound, with having:

- Functional sub-processes dealing with each of data movements (including possibly all CRUD operations support -Create, Read, Update, Delete)
- Functional sub-processes performing data manipulations (such as computations, data transformations and visualizations and other non-trivial actions).

The proposed approach is presented at Figure 7.

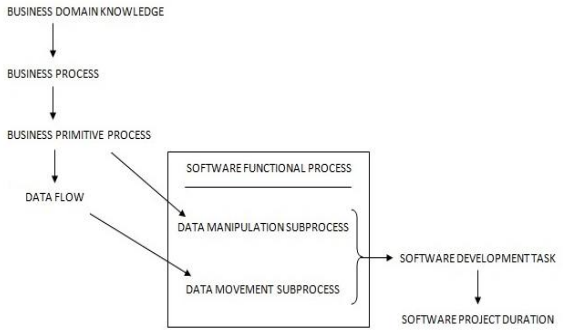



Figure 7. The proposed approach phases.

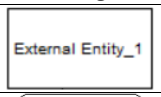
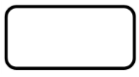
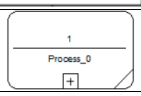
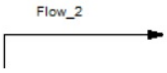
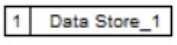
If business process is primitive, it is possible that it could contain a single data movement processing or a single data manipulation processing without any other sub-processes. Then, the computation could be more precise. The best accuracy of computations is possible if the business process model is developed to the primitive business processes level of details, where all relevant processes, data flows, data stores and data entities are specified.

Of course, the proposed method in this paper respects realistic possibilities of having needed knowledge in the early phase of software development. At the very beginning, the high level of business process specification is only possible, having conceptual elements at context diagram available. The proposed method enables application of formula for calculating software project duration at each level of decomposition, according to business process domain knowledge availability. Therefore, the proposed model is scalable and adaptable to different circumstances and availability of domain knowledge data.

### 4.2.1. Mapping Between COSMIC and DFD Elements

The mapping between COSMIC concepts and DFD diagram elements is proposed at Table 1. Table 1 shows the COSMIC concepts: functional user, functional process, data group movements, and persistent storage, represented in conformity with the ISO 5807 standard for symbols in flowcharts and diagrams [2].

Table 1. Mapping DFD and COSMIC diagram concepts.

| COSMIC Concept | COSMIC diagram | DFD diagram |
|---|---|---|
| Functional User (FU) |  |  |
| Functional Process (FP) |  |  |
| Data group movement E/X/W/R |  |  |
| Persistent storage |  |  |

In Table 1, mapping between key concepts of COSMIC method and DFD diagrams are: Functional user (i.e., user of software function) from COSMIC corresponds to External Entity in DFD; Functional process (i.e., software function) from COSMIC corresponds to business process in DFD; Data movement from COSMIC corresponds to data flow from DFD, where one graphical data flow could contain multiple CRUD actions (Entry ~ Create, Exit ~Read, Write~Create, Read~Read, with additional Update and Delete actions).

Generally speaking, one internal data flow (i.e., resource flow-data flow that is connected with data

store) could initiate performing any combination of four CRUD operations (as presented at Figure 6. and Figure 8). Therefore, a resource flow can have 1 or more (up to max 4) operations assigned. If we generalize resource flows to any data flow, we can have any data flow engaged in CRUD operations (initial changes in state of internal data stores or external entities).

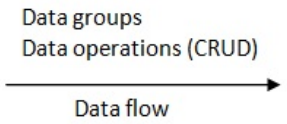



Figure 8. Data flow composition-data group and data operations (CRUD) they initiate.

Communication with external entities could also be active, since external entities could be other software modules or devices, not only persons or organizations. According to COSMIC method and [31], any data flow could have data groups and could be used as a basis for data movements, i.e. calculating functional size of software.

CRUD operations in DFD diagram are: C-Create new data (insert) in a data store or at external entity, R -Reading (retrieve) data from data store or external entity, U-Updating (change) data in a data store or external entity, D- Deleting (erase) data in a data store or external entity.

Usually one "resource flow" collaborates with one data store (resource). If we make generalization, it could be possible for one resource data flow to communicate with more data stores. COSMIC method proposes that each data flow deals with data group related to a single data object (i.e., to have internal cohesion of data within the data flow). The proposed method in this paper allows more general approach - to have multiple data groups within a single data flow. Having generalized approach in this paper, it is also possible that not only resource flows have multiple functions (CRUD operations) and multiple data stores attached, but also data flows that collaborate with external entities could have the same characteristics (multiple data groups, multiple external entities).

### 4.2.2. Formula for Calculating Software Project Duration

The formula for calculating number of all data movements in the whole software includes:

- Number of business processes (with suggestion to prioritize primitive business processes, if possible),
- Number of data flows (having for each data flow, possibly, multiple CRUD operations attached to data groups)
- Number of data stores (or external entities – making no distinction between internal or external communication and making possible for external data flows to perform CRUD operations with external entities, being able to actively collaborate in data exchange).
- Each CRUD operation (Create, Read, Update, Delete) is assigned to a separate data movement operation.

In the proposed formula, each data movement is named "software task", referring to a task for a software developer to work on, i.e. to implement the required software function represented by the data movement function. Therefore, each data movement functional sub-process refers to one software development task.

Total number of all data movement-related software tasks for the whole software development could be calculated with formula:

$$DMoTaskNum = \sum_{i=1}^{4} \sum_{j=1}^{n} (i * \sum_{k=1}^{n} PNj * \sum_{l=1}^{n} DFNk * \sum DSNl)[CFP] \quad (1)$$

Where:

- *DMoTaskNum*- Total number of software development tasks related to data movements, expressed in CFP,
- *PNj*- number of business processes (primitive, if possible),
- *DFNk*- number of data flows,
- *DSNl*- number of data stores (or external entities, if being able to actively collaborate in data exchange),
- *i*- number of CRUD operations that a data flow could perform, values could be: 1, 2, 3, and 4 (which means: one data flow could have assigned any combination of four CRUD operations, so each data flow could have min 1 and max 4 CRUD operations assigned).

Software project duration estimation can be calculated when the number of all software development tasks is multiplied with a single task duration. Effort is expressed in Man/Hours (m-hrs) and duration (time period needed for development of a software task) in hours. Effort is comparable with the term "duration" so in this paper duration is measured with "man/hours" unit.

The proposed approach in this paper considers each business process, presented at DFD, to have both components (data movement and data manipulation component). Therefore, the formula (2) that calculates total software project duration is designed to have two crucial components (DMoTaskNum ~data movements, ProcNum~ data manipulation):

$$SPDE=(DMoTaskNum+ProcNum)*DurPerCFP[m\text{-}hrs] \quad (2)$$

Where:

- *SPDE*-software project duration estimation, expressed in man/hours,
- *DMoTaskNum*-number of data movement software development tasks, derived from DFD elements,

- *ProcNum*-number of business processes in DFD, which is basis for calculating data manipulation software development tasks and development duration,
- *DurPerCFP*-average development duration per 1 CFP.

Average duration per one CFP (*DurPerCFP*) is a factor related to development team productivity and could be separately measured with each specific team and integrated within this formula. Empirical results from Issa and Rub [22] show that 0.67 man-months were calculated as an average engagement per use case. According to Desharnaisa *et al.* [20], it is concluded that effort per COSMIC Functional Point (CFP) is 2 man-hours/CFP.

## 5. Example of the Proposed Method Application

To explain the proposed method application, an illustration is made with a simple abstract example. DFD consists of three processes, two data stores and eight data flows that connect data stores with business processes, as presented at Figure 9.

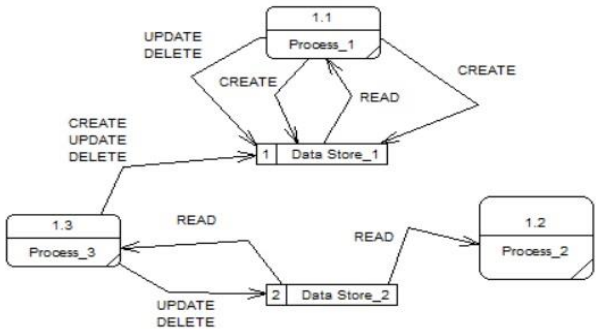



Figure 9. Example-abstract DFD diagram

Corresponding COSMIC graphical representation for DFD from Figure 9 is presented at Figure 10. Data flow movements from Figure 9 are mapped into arrows at Figure 10. At the example presented in this section, it means that there are 12 data movements at the diagram at Figures 9 and 12 arrows at Figure 10.

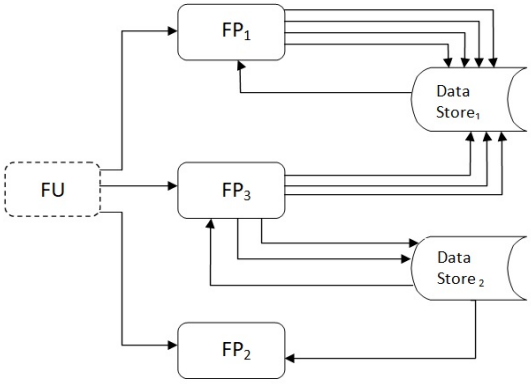



Figure 10. COSMIC graphical presentation of corresponding abstract DFD from Figure 9.

Calculating DMoTaskNum for 1.1 Process_1: two data flows execute create action, one executes only read action, while the fourth process executes update and delete operations over DataStore_1.

For one CRUD operation i=1, number of processes is PN=1, number of data flows with one operation is DFN=3, number of data stores is DSN=1. With CRUD operations i=1 there is only one data flow: i=2, PN=1, DFN=1, DSN=1.

Task number for Process_1 is according to (1):

$$DMoTaskNum_{P1}=(i_1*PN_1*DFN_1*DSN_1)+(i_2*PN_2*DFN_2*DSN_2)$$
$$DMoTaskNum_{P1}=(1*1*3*1)+(2*1*1*1)$$
$$DMoTaskNum_{P1}=3+2$$
$$DMoTaskNum_{P1}=5$$

Calculating *DMoTaskNum* for 1.2 Process_2: only one data flow executes only read action over DataStore_2.

$$DMoTaskNum_{P2}=(i_1*PN_1*DFN_1*DSN_1)$$
$$DMoTaskNum_{P2}=(1*1*1*1)$$
$$DMoTaskNum_{P2}=1$$

Calculating *DMoTaskNum* for 1.3 Process_3: one data flow executes read action and one executes update and delete actions over DataStore_2, while the third data flow executes update, delete and create operations over DataStore_1.

$$DMoTaskNum_{P3}=(i_1*PN_1*DFN_1*DSN_1)+(i_2*PN_2*DFN_2*DSN_2)+(i_3*PN_3*DFN_3*DSN_3)$$
$$DMoTaskNum_{P3}=(1*1*1*1)+(2*1*1*1)+(3*1*1*1)$$
$$DMoTaskNum_{P3}=1+2+3$$
$$DMoTaskNum_{P3}=6$$

Finaly, total estimation SPDE for whole software development, based on DFD presented at Figure 9, is calculated by using sum of all *DMoTaskNum*s (for each business process), multiplied with average effort duration per 1 CFP and summarizing with number of processes multiplied with the same average effort duration per 1 CFP.

For the purpose of simplified illustration in this example, lets accept results from Desharnaisa *et al.* [20], where average duration of software development is 2 m-hrs/CFP. Then, final SPDE for the whole software, based on DFD from Figure 9. Could be calculated as:

$$SPDE=(DMoTaskNum_{P1}+DMoTaskNum_{P2}+DMoTaskNum_{P3}+ProcNum)*DurPerCFP$$
$$SPDE=(5+1+6+3)*2=(12+3)*2=30 \text{ m-hrs}$$

Total software project estimation in this abstract example is 30 Man/Hours.

## 6. Empirical Study

### 6.1. Research Methodology

*Research problem and aims*: since COSMIC method does not include procedures for defining software functionality based on business process models, it is beneficial to integrate COSMIC method measurement

system with business process model, which is firstly a basis for functional units deriving, and secondly, for software development effort and duration estimation. Aim of this empirical research is to present the applicability of the proposed approach (in section 4.) with real examples.

*Research methods*: research sample DFDs was collected from university students practical work. Each DFD is analyzed for the number of DFD elements (processes, data flows, data stores). For each DFD calculation is performed by using the proposed formula (in section 4.).

*Research sample*: empirical study of the proposed method is conducted with 41 different process models (DFDs) collected from the participants at the University of Novi Sad, at Technical faculty, Mihajlo Pupin" Zrenjanin, Serbia. Empirical research sample is divided into two groups. First group of DFDs are results of bachelor students' mid-term exams in Information Systems and Software Engineering, while the second group are DFDs collected from bachelor theses (final exams).

### 6.2. Research Results

Results of DFDs' analysis from first sample group (midterm exams DFDs) is presented at Table 2, while results of DFDs' analysis from the second sample group (final exams, i.e., bachelor theses) is presented at Table 3.

Table 2. Results for midterm exams SPDE calculation

| Estimation element | MAX | AVG | MIN |
|---|---|---|---|
| Total processes in the model | 16 | 6.34 | 5 |
| Total data stores in the model | 6 | 3.46 | 2 |
| Total data flows in the model | 39 | 26.09 | 18 |
| Task number [COSMIC CFP] | 38 | 17.52 | 8 |
| **SPDE [Man/Hour]** | 108 | 47.72 | 26 |

Table 3. Results for final exams SPDE calculation.

| Estimation element | MAX | AVG | MIN |
|---|---|---|---|
| Total processes in the model | 17 | 14.83 | 8 |
| Total data stores in the model | 16 | 6.7 | 2 |
| Total data flows in the model | 75 | 44.83 | 25 |
| Task number (COSMIC CFP) | 140 | 58.17 | 23 |
| **SPDE (Man/Hour)** | 314 | 146 | 62 |

### 6.3. Discussion

Regarding the empirical research results, it can be observed that there is a significant difference in DFD complexity comparing students' exams works (calculated SPDE is between 26 and 108 man/hours) and students' bachelor works (calculated SPDE is between 62 and 314 man/hours).

Another relevant conclusion that could be drawn from empirical research results is that the number of data processes and data stores are not significant factors for calculating estimation of software project duration, while the number of data flows significantly affects the estimation calculated value. It was expected, since the COSMIC method also emphasizes the role of data movements, which are included in data flows. This way, it has been proven that the proposed method in this paper has alignment with basic principles of COSMIC method, but it provides enhancements by including not only data flows, but also other elements of business process models into the calculation formula.

Limitations of the proposed approach are related to:

- Origins of the value for average development duration per 1 CFP, used in formula. The value could be calculated or estimated, based on development team abilities (skills, knowledge, personal working style), used development technology, type of project, etc.
- Simplification of using the same average value of development duration per 1 CFP used for both data movement-related software development tasks and data manipulation-related software development tasks, which in real-world circumstances significantly differ. It is much harder to implement data manipulation software part, then to support a simple CRUD operation.
- Assumption that each business process performs both data movement and data manipulation. When primitive processes are concerned, they usually have a simple task and not a complex structure. Therefore, it is needed to take into the formula the detail regarding the hierarchical level of the process that is included.
- Direct mapping of a business process (preferably primitive business process) to a single software functional process. This mapping could be performed with more details, having 1: N relationship between business process and software functions that support the process. This detailed mapping could not be performed in a very early software project duration estimation, when only rough business process model (i.e. DFD) is available. Detailed mapping could be done within the detailed business process analysis and software design phase, with the consequence of having more precise calculation of software project duration. Aim of this paper was to provide early software project duration estimation, i.e. calculation, where proposed formula is calculated upon the DFD, created with appropriate level of details according to available knowledge.

## 7. Conclusions

This paper describes the approach of using and improving COSMIC method in calculating the estimated software development project duration,

based on DFD. Existing approach of COSMIC method have imprecise source of functional processes, needed for CFP calculation. COSMIC method is focused on data movement-related software functional sub-process, while data manipulation-related functional sub-processes were not included directly in the estimation of functional size of software to be developed.

The proposed method applicability was illustrated with an example, as well as with empirical research, conducted with the students' DFDs as a sample. Empirical research shows that there is a significant impact of data flows number to final calculation results, where it has been proven that the proposed approach is aligned with COSMIC method.

Future research could be performed in several directions, starting with improvements regarding in previously mentioned limitations. The method could be enhanced with integration of method of measurement or calculation of more precise value for the duration per 1CFP, particularly to make distinction between simple data movement tasks and data manipulation software development tasks. Further research could be directed towards using more precise mapping between business processes and software functions, with having the estimation applied in the early software design phase. Further analysis of applicability of the proposed approach could also be performed by conducting comparisons of early software projects duration calculated values with actual project durations from industrial practice, applied with projects of various complexity. Finally, to separate back-end and front-end development efforts is also a significant aspect of future research related to software project duration estimations based on appropriate metrics.

## References


[1] Abran A., *Software Project Estimation*, Wiley, 2015.

[2] Abran A., Symons C., Ebert C., Vogelezang F., and Soubra H., "Measurement of Software Size: Contributions of COSMIC to Estimation Improvements," *in Proceedings of the International Training Symposium*, Bristol, 2016.

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

[4] Ali N., Petersen K., and Scheider K., "FLOW-Assisted Value Stream Mapping in the Early Phases of Large-scale Software Development," *Journal of Systems and Software*, vol. 111, pp. 213-227; 2016.

[5] Angara J., Prasad S., and Sridevi G., "Towards Benchmarking User Stories Estimation with COSMIC Function Points-A Case Example of Participant Observation," *International Journal of Electrical and Computer Engineering*, vol. 8, no. 5, pp. 3076-3083, 2018.

[6] Aysolmaz B. and Demirors O., "Automated Functional Size Estimation Using Business Process Models with UPROM Method," *in Proceedings of the Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement*, Rotterdam, pp. 114-124, 2014.

[7] Bagheri E., Ensan F., and Gasevic D., "Decision Support for the Software Product Line Domain Engineering Lifecycle," *Automated Software Engineering*, vol. 19, no. 3, pp. 335-377, 2012.

[8] Baniassad E., Clements P., Araújo J., Moreira A., Rashid A., and Tekinerdogan B., "Discovering Early Aspects," *IEEE Software Journal*, vol. 23, no. 1, pp. 61-70, 2006.

[9] Bhuiyan M., Haque F., and Shabnam L., "Integration of Organisational Models and UML Use Case Diagrams," *Journal of Computers*, vol. 13, no. 1, pp. 1-17, 2018.

[10] Bourque P. and Fairley R., *SWEBOK v 3.0, Guide to the Software Engineering Body of Knowledge*, IEEE Computer Society, 2014.

[11] Bouwman H. and Solaimani S., "A Framework for the Alignment of Business Model and Business Processes: A Generic Model for Trans-sector Innovation," *Business Process Management Journal; Emerald*, vol. 18, no. 4, pp. 655-679, 2012.

[12] Broj M., *Perspectives on the Future of Software Engineering*, Springer Verlag, pp. 15, 2013.

[13] Buglione L. and Gencel C., "Impact of Base Functional Component Types on Software Functional Size Based Effort Estimation," *in Proceedings International Conference on Product Focused Software Process Improvement*; Monte Porzio Catone, pp. 75-89, 2008.

[14] Carroll E., "Estimating Software Based on Use Case Points," *in Proceedings Companion to the 20th annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications*, San Diego, pp. 257-265, 2005.

[15] Chongpakdee P. and Vatanawood W., "Estimating User Story Points Using Document Fingerprints," *in Proceedings 8th International Conference on Software Engineering and Service Science*, Beijing, 2017.

[16] COSMIC Group, *Introduction to the COSMIC Method of Measuring Software*, Version 1.2, 2019.

[17] COSMIC Group, *Guideline for Sizing Business Application Software*, Version 1.3, 2017.

[18] Dennis A., Wixom B., and Roth R., *System Analysis and Design*, John Wiliey and Sons,

2012.

[19] Cruz E. and Da Cruz A., "Deriving Integrated Software Design Models from BPMN Business Process Models," *in Proceedings 13th International Conference on Software Technologies*, Porto, pp. 571-582, 2018.

[20] Desharnais J., Buglione L., and Kocaturk B., "Using the COSMIC Method To Estimate Agile User Stories," *in Proceedings of the 12th International Conference on Product Focused Software Development and Process Improvement* Torre Canne Brindisi, pp. 68-73, 2011.

[21] Giaglis G., "A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques," *International Journal of Flexible Manufacturing Systems*, vol. 13, no. 2, pp. 209-228, 2001.

[22] Issa A. and Rub F., "Performing Early Feasibility Studies of Software Development Projects Using Business Process Models," *in Proceedings of the World Congress on Engineering WCE2007*, London, pp. 536-540, 2007.

[23] Kama N., Basri S., Ismail S., and Ibrahim R., "Using Static and Dynamic Impact Analysis for Effort Estimation," *The International Arab Journal of Information Technology*, vol. 16, no. 2, pp. 163-168, 2019.

[24] Kaya M. and Demirors O., "E-Cosmic: A Business Process Model Based Functional Size Estimation Approach," *in Proceedings of the 37th EUROMICRO Conference on Software Engineering and Advanced Applications*, Oulu, 404-410, 2011.

[25] Khatibi B., Jawawi D., and Khatibi E., "Investigating the Effect of Using Methodology on Development Effort in Software Projects," *International Journal of Software Engineering and Its Applications*, vol. 6, no. 2, pp. 35-46, 2012.

[26] Kim S., Lively W., and Simmons D., "An Effort Estimation by UML Points in the Early Stage of Software Development," *Software Engineering Research and Practice*, pp. 415-421, 2006.

[27] Kimani M. and Wahid A., "Use Case Point Method of Software Effort Estimation: A Review," *International Journal of Computer Applications*, vo. 116, no. 15, pp. 43-47, 2015.

[28] Kumar G. and Bhatia P., "A Detailed Analysis of Software Cost Estimation Using COSMIC-FFP," *Review of Computer Engineering Research*, vol 2., no. 2, pp. 39-46, 2015.

[29] Lavazza L., del Bianco V., and Garavaglia C., "Model-based Functional Size Measurement," *in Proceedings of the 2nd ACM-IEEE international Symposium on Empirical Software Engineering and Measurement*, Kaiserslautern, pp. 100-109, 2008.

[30] McConnell S., *Software Estimation*, Microsoft Press, pp. 200, 2006.

[31] Monsalve C., Abran A., and April A., "Measuring Software Functional Size from Business Process Models," *International Journal of Software Engineering and Knowledge Engineering; World Scientific Publishing Company*, vol. 21, no. 3, pp. 311-338, 2011.

[32] Nagpal G., Uddin M., and Kaur A., "Grey Relational Effort Analysis Technique Using Regression Methods for Software Estimation," *The International Arab Journal of Information Technology*, vol. 11, no. 5, pp. 437-446, 2014.

[33] Nassif A., Capretz L., and Ho D., "Software Effort Estimation in the Early Stages of the Software Life Cycle Using a Cascade Correlation Neural Network Model," *in Proceedings of the 13th ACIS International Conference on Software Engineering, Artificial Intelligence Networking and Parallel/Distributed Computing*, Kyoto, pp. 589-594, 2012.

[34] Omural N. and Demirors O., "Effort Estimation Methods for ERP Projects Based on Function Points: a Case Study," *in Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement ACM IWSM/Mensura*, Gothenburg, pp. 199-206, 2017.

[35] Pereira J. and Silva D., "Business Process Modeling Languages: A Comparative Framework," *in Proceedings of the World Conference on Information Systems and Technologies*, Recife, pp. 619-628, 2016.

[36] Pillai K. and Jeyakumar M., "A Real Time Extreme Learning Machine for Software Development Effort Estimation," *The International Arab Journal of Information Technology*, vol. 16, no. 1, pp. 17-22, 2019.

[37] Primandari P. and Sholiq P., "Effort Distribution to Estimate Cost in Small to Medium Software Development Project with Use Case Points," *Procedia Computer Science*, *Elsevier*, vol. 72, pp. 78-85, 2015.

[38] Rastogi V., "Software Development Life Cycle Models-Comparison, Consequences," *International Journal of Computer Science and Information Technologies*, vol. 6, no. 1, pp. 168-172, 2015.

[39] Reinhartz-Berger I., Soffer P., and Sturm A., "A Domain Engineering Approach to Specifying and Applying Reference Models," *in Proceedings of the Workshop on Enterprise Modelling and Information Systems Architectures*, Klagenfurt, pp. 50-63, 2005.

[40] Reinhartz-Berger I. and Sturm A., "Utilizing Domain Models for Application Design and

Validation," *Information and Software Technology*, vol. 51, no. 8, pp. 1275-1289, 2005.
[41] Shanker M., Jaya J., and Thanushkod K., "An Effective Approach to Software Cost Estimation Based on Soft Computing Techniques," *The International Arab Journal of Information Technology*, vol. 12, no. 6A, pp. 658-665, 2015.
[42] Smith C. and Woodside M., *System Performance Evaluation: Methodologies and Applications*; CRC Press, 1999.
[43] Vogelzang F., "Using COSMIC-FFP for Sizing, Estimating and Planning in an ERP Environment," *in Proceedings of the International Workshop on Software Metrics and DASMA Software Metrik Kongress, IWSM /MetriKon*, Postdam, pp. 327-342, 2006.
[44] Yourdon E., *Modern Structured Analysis*, Prentice Hall International, 1989.

**Zoltan Kazi** is an associate professor at University of Novi Sad, at Technical faculty "Mihajlo Pupin" Zrenjanin, Serbia. He received his PhD degree in computer science from the Novi Sad University in 2014. His research interests include knowledge based systems, automated reasoning systems, ontology engineering, business intelligence, complex databases, information system modeling and development.

**Ljubica Kazi** is an associate professor at University of Novi Sad, at Technical faculty "Mihajlo Pupin" Zrenjanin, Serbia. She received her PhD degree in computer science from the Novi Sad University in 2016. Her research interests include information system modeling, adaptive software systems, software metrics and distributed information systems development.