Plecka, Przemyslaw; Bzdyra, Krzysztof

Article

Usefulness of software valuation methods at initial stages of ERP implementation

Foundations of Management

Provided in Cooperation with:
Faculty of Management, Warsaw University of Technology

Suggested Citation: Plecka, Przemyslaw; Bzdyra, Krzysztof (2013) : Usefulness of software valuation methods at initial stages of ERP implementation, Foundations of Management, ISSN 2300-5661, De Gruyter, Warsaw, Vol. 5, Iss. 3, pp. 33-48, https://doi.org/10.2478/fman-2014-0018

This Version is available at:
https://hdl.handle.net/10419/184560

USEFULNESS OF SOFTWARE VALUATION METHODS AT INITIAL STAGES OF ERP IMPLEMENTATION

Przemysław PLECKA*, Krzysztof BZDYRA**

Faculty of Electronic and Computer Engineering, Koszalin University of Technology, Koszalin, Poland
*e-mail: przemek.plecka@gmail.com
**e-mail: krzysztof.bzdyra@tu.koszalin.pl

Abstract: This work discusses the problem of selecting methods for valuing the costs and estimating the time of implementing computer systems in cases when system modification is necessary. The methods presented in literature are reviewed and the stages of strategic phase of implementation characterised. On the basis of the analysis of data required by each method and the data obtained at different stages, appropriate selection of methods for each stage was proposed.

Keywords: ERP, IT project, COCOMO, Delphi method, function points, implementations.

1 Introduction

The first ERP-class information systems (IS) were available in large companies only [1]. High implementation and maintenance costs were a barrier for their proliferation. With the decrease of costs, the group of users also covered medium and small companies. The first systems were tailor-made for individual customers. Their functions suited the organisations they were made for but their production costs were excessively high. Gaining experience from work with different clients, the producers selected a set of functions that reappeared in most versions and offered it as a standard version of their product. At the moment, all significant producers have their standard product: SAP, Business Suite; Microsoft, Dynamix AX; JD Edwards, EntrepriseOne; and so on.

During trade talks while selling standard ERP systems, the parties (suppliers and clients) reach a conclusion that the organisation of processes in the company does not fully overlap with the processes supported by the computer system that is available [2]. There are a group of processes that are not represented in any functionality in standard ERP system. This generates a need for adapting IS to a company. The costs of modifications increase the value of the contract (implementation). In some cases, it is the company that adapts processes to the system; however, the costs of organisational changes are an additional burden to the client. It is only when clients recognise the costs of system implementation (including adaptations) that they incline to consider changes in their organisations. In such cases, either the value of contract will be higher and the system will overlap with the processes in the company or the value of contract (costs for the client) will be lower and the client will need to adapt the organisation to the IS to a certain extent. For this reason, cost estimation at very early stages of implementation is crucial for system suppliers. Employing appropriate methods at each stage will allow the suppliers make earlier and more precise estimations of costs. As a result, they will generate lower costs of concluding a contract and increase the chances for a successful implementation of the project.

The aforementioned adaptations of IS to the company are the modifications that involve redefinition or broadening processes or structures of data implemented in IS. Standard sets of functionalities are similar for all software providers. The differences concern additional functionalities for different businesses. For instance, margin analysis in construction industry may be based on projects (revenue and project costs), whereas in metal production it can be defined on the basis of assortment groups (income from selling articles from a given group and production costs). It is important to define standard functionalities, as they will be the object of modifications [3]. These changes do not concern any other than ERP-class systems. Single-activity IS (e.g. sales with limited and closed functionality) do not need to be modified. Prospective clients (small companies) select readily available software by analysing correspondence with the processes in their company.

The methods facilitating valuation of software production are known and discussed in literature, e.g. by McConell [4]. However, due to changes in information technologies, the popularity of their use also

changes. The use of algorithmic methods at initial stages of information projects is difficult.

At this stage, there are no analytic or project documentations whose components facilitate estimating algorithms. Despite the fact that the uses of algorithmic methods at early stages of information projects can be found in literature [5, 6] the practice of information project suppliers indicates a common use of non-algorithmic methods as faster (i.e. cheaper) and easier. One can find suggestions for using cost evaluation methods for information projects, starting with statements that any combinations of methods should be used, through views about when and what methods should be used, and finishing with "step by step" procedures [7]. There are however, no guidelines advising a given method depending on the stage of implementation.

Negotiations with ERP system suppliers and clients concern implementation costs and time. For estimating the cost of software, one may use such time-consumption measures as man hours, man days or man months. With a given cost of a working unit of time for implementation, it is possible to calculate the cost in a given currency and the time (dates) of implementation, with consideration for possible simultaneousness of works.

What is known are the stages of software lifecycle [8] and software valuation methods [4]. The range of the problem was limited to ERP-class IS. The question is which of the evaluation methods produces most appropriate results of costs and time at a given stage of project implementation. The limitation is in the quality of necessary input data at given stages.

Section 2 of this article includes the description of stages in the strategic phase of implementation project with consideration for the data available for valuation. Section 3 is a review of algorithmic valuation methods. Section 4 includes the description of non-algorithmic methods. The final section presents the conclusions resulting from the connection of effects from lifecycle stage and the data necessary for software valuation. This is how alternative uses of methods at each stage are proposed.

The use of symbols in Figures 1, 3, 4 and 6 is in accordance with BPMN 2.0[1], even if full schemes may not be coherent with the notation [9].

[1] BPMN (Business Process Model and Notation) – a graphic notation for describing business processes.

2 Lifecycles stages of ERP system implementation

Numerous authors describe software lifecycles focusing on software production or writing software on an individual client's order [8, 10]. None of the presented models corresponds entirely to implementation process of ERP-class software in a middle-sized company. They do not consider "movability" of the end of strategic phase (concluding a contract) and possibilities of having one additional stage – feasibility study. Feasibility study is not significant for software lifecycle; however, it provides information for project valuation.

The stages of IS implementation may be categorised as follows:

1. initial trade talks,
2. pre-implementation analysis,
2'. feasibility study,
3. system change project,
4. implementation of changes and testing,
5. test installation and initial import of historic data,
6. system validation,
7. training,
8. final installation and proper data import,
9. user assistance,
10. in-use changes.

The grouping of the stages in phases is presented in Figure 1.

Considering cost evaluation, one should remember that in the sales process, the moment of contract conclusion is significant. Up to this moment, the supplier estimates the costs, while later they verify and predict the costs hoping that they will not exceed the income. Contract conclusion may happen right after stage 1 but not later than stage 4. This period is called the strategic phase. It is in IS supplier's interest to get the contract signed as soon as possible, as the implementation of subsequent stages increases the costs with the risk of failure of concluding the agreement. However, early estimation of costs involves higher risk of estimation error. A convenient situation, from the supplier's point of view, in which the client agrees to sign an agreement for implementation analysis is very rare. It is only after its completion (and valuation) that the parties sign an implementation agreement for other stages. For this reason, stages 1–3 are important and these are the ones that are going to be discussed in subsequent sections of the present work.

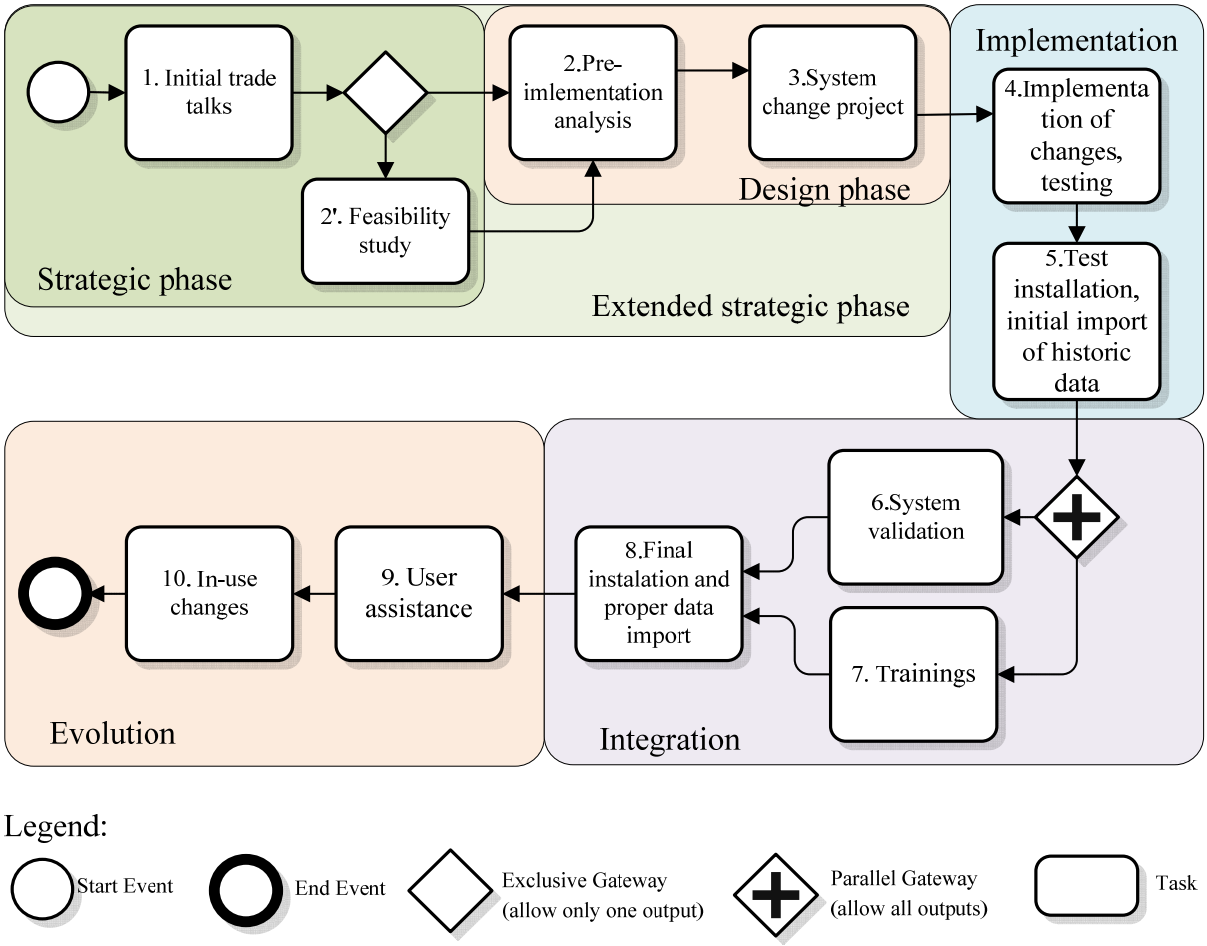

Figure 1. ERP-class information system lifecycle implemented in a company

.

2.1 Initial trade talks

The supplier has meetings with a prospective client in order to define the range and value of the contract. Usually, it is conducted in one or two initial meetings followed by two or three presentation meetings. Some of the elements of work range are identified quickly and precisely. These concern primarily the computer hardware, network infrastructure and licences for individual ERP modules. Some elements, e.g. IS modifications that result from non-typical users' requirements are difficult to define.

At this stage, the supplier cannot fully identify the needs that are not satisfied by the standard version of ERP system. As clients' knowledge on IS comes from trade presentations, they cannot define precisely which requirements are not standard. As an example, the general range of contract is presented below:

A. hardware – supply: 12 computer work stations, 1 server in accordance with the specification

B. information infrastructure – service: 25 electric and logical access points and two WLAN access points in client's office building

C. licences: financial module – 4 users; personnel module – 3 users; logistics module – 12 users: sales module – 4 users; production module – 11 users

D. adapting IS in for logistics and production processes

E. training: 30 man days (3 days on financial area, 6 days on personnel, 12 days on logistic, 4 days on sales, 5 days on production)

F. assistance for users at work: 25 man days.

Except for part D, the supplier has sufficient information to present a price offer to the client. Moreover, using the description of the range in such a form as e.g. in part E or F, they secure against the client's changes in requirements for this range. If it happens in the course of implementation that the client will need 45, rather than 25 hours of assistance, and the amount in the contract stipulated at 25 hours, the supplier will have the right to additional pay.

The situation is different with IS modifications from part D in the above range of contract. At this stage, the supplier holds a set of client's general and specific requirements. The reason for non-uniformity of specificity of the requirements (general and specific) is the client's lack of experience and knowledge. Specific requirements are encountered sporadically and are most frequently related to presentation of data in the system (prints, listings, etc). One should remember that the aim of this stage is not collecting the requirements but concluding a contract. The requirements are obtained, as if "by chance", from presentations and talks with the client. Thus, the supplier is aware that there is still a subset of undisclosed requirements.

The supplier who decides to evaluate IS adaptation must consider the above "faults", including undisclosed requirements. Estimation error and the risk of underestimation are usually high.

2.2 Feasibility study and pre-implementation analysis

If the supplier was unable to evaluate system adaptation (modifications), works aimed at clarifying and specifying client's needs must be conducted. Then, a pre-implementation analysis or feasibility study is done [11]. Although both solutions are aimed at specifying the data for the evaluation, the basic purpose of each is different. If the supplier estimates the chances for signing a contract as high, they order a feasibility study. The work is less expensive and in case of concluding the contract, some effects may be used in pre-implementation analysis.

Feasibility study includes information on the company in a form of a systematic document based on economic facts [12]. The information concerns economic, organisational and technical aspects [13]. The aim of the study is to define the range of works (including modifications) and the costs of the project. The document is used by supplier's decision-makers while analyzing economic aspects of project implementation.

Pre-implementation analysis includes exclusively the information concerning the computer system in the context of a given company. The result is a report including the following components: functional range of the implementation, list and description of business processes, functions and data advised to be included in the functional range of the system, organisational range of the implementation, proposed aims of the implementation, expected business benefits, and schedule of work [14]. For example, two requirements may appear as in the following:

- L.03.12.02 – invoice for clients from outside EU should include an amount to pay in the client's currency with the number of Polish National Bank currency exchange table that was used to calculate the amount in the field "comments",
- L.03.12.05 – operator issuing VAT invoice should be able to print the document, save it as PDF file or e-mail as an attachment in PDF file to the address from "e-mail" field in client's database.

The numbering of requirements from the example is in accordance with WBS[2] [15] and may mean: L – logistics, 03 – group of requirements concerning sales processes, 12 – subgroup of sales invoices and 02 – the number of subsequent requirement in subgroup 12.

Even in a medium-sized production company, the recording all user requirements would be very time-consuming and expensive (from a few up to over a thousand requirements). Moreover, in most cases, they would overlap with the records in ERP system documentation. Therefore, suppliers make a differential analysis that includes only those elements that are not covered in a standard IS. Such a procedure shortens the time of stage implementation but also allows the client to see the documentation of a standard version with the pre-implementation analysis in order to get an idea of the future system.

At this stage, the supplier assumes that the requirements are complete and the level of their specificity meets the deed of the designers to whom the document is addressed. At the same time, the documentation of implementation analysis (specification of requirements) is going to be used for labour valuation.

2.3 Project of system changes

Project of IS is an intermediate phase between defining the requirements and the implementation. The documentation that is produced is indented exclusively for internal use of the supplier (software departments).

[2] WBS (Work Breakdown Structure) – a basic technique in managing projects that allows defining and organising the range of project with a hierarchic tree-structure.

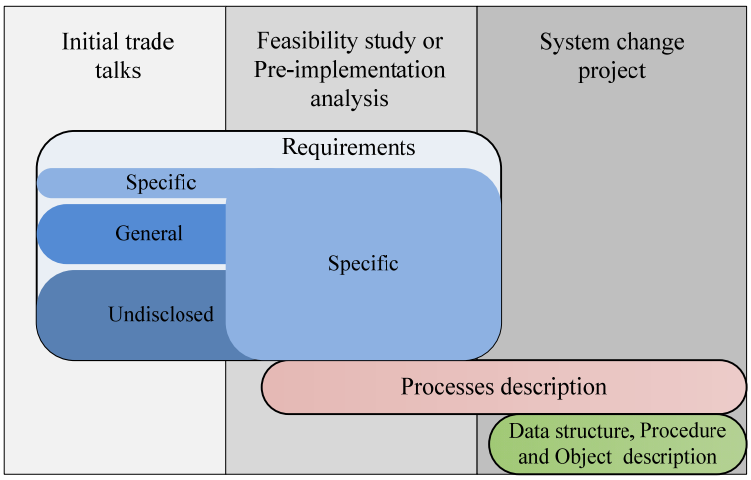

Figure 2. Input information necessary for valuation at initial stages of lifecycle.

Depending on the methods of implementation (structured, object-oriented programming, agile software development, etc), project documents may include different elements [8]. Some ERP system developers worked out their own specific methodologies. In such cases, the documentation will be specific. One such example is Select Perspective methodology[3] [16, 17] or ARIS[4] [18]. However, there are always common elements for evaluating software.

The first element of software developing is to specify the requirements resulting from implementation character. The level of requirement specificity must determine the manner of implementation in an unambiguous way. Despite this, project documents include the elements describing data structures and procedures of processes. There are a number of methods for presenting project information: from DFD, Entity-Relationship Diagrams [19], through Unified Modelling Language (UML) models [20]. Each of them is an appropriate source of data for software evaluation.

2.4 Summary of lifecycle stages

With subsequent stages of software lifecycle, the supplier's knowledge of the client's organisation enlarges. In the first two stages, only requirements are obtained and after the project stage such elements as data objects (tables, fields) and interface windows are also known. At the same time, the supplier's costs will be rising. If a contract with client is concluded, the costs will be included in the contract value; if not, they will be the supplier's cost. Input information necessary for making valuation at the first three stages of project lifecycle is presented in Figure 2.

3 Algorithmic methods of software evaluation

3.1 COCOMO II method

Constructive cost model (COCOMO) method was proposed by Barry Boehm in 1981 [22]. Since then, a number of versions and types of this method have been developed, e.g. COCOMO81 and COCOMO II [22]. It is used to calculate Person per Month (PM) on the basis of Kilo Source Line of Code (KSLOC) (process 1 in Figure 3). KSLOC calculation is done on the basis of project components. Because for many contemporary uses, the use of source lines does not correspond with PM, the method was modified by using function point (FP) analysis, as presented in section 3.2 [23] (process 2 in Figure 3). The complete sequence of process realising COCOMO method is presented in Figure 3.

The first activity is defining five scale factors (SFs) (process 3 in Figure 3), whose value is determined empirically in five classes, depending on the level of complexity (from very low to very high), which is presented in Table 1.

[3] Select Perspective – formalised by Select Business Solutions Inc, a set of best practices supporting software development and its controlling.

[4] ARIS (Architecture of Integrated Information Systems) – a method of analysing and modelling industrial processes that lead to building an integrated system of information processing.

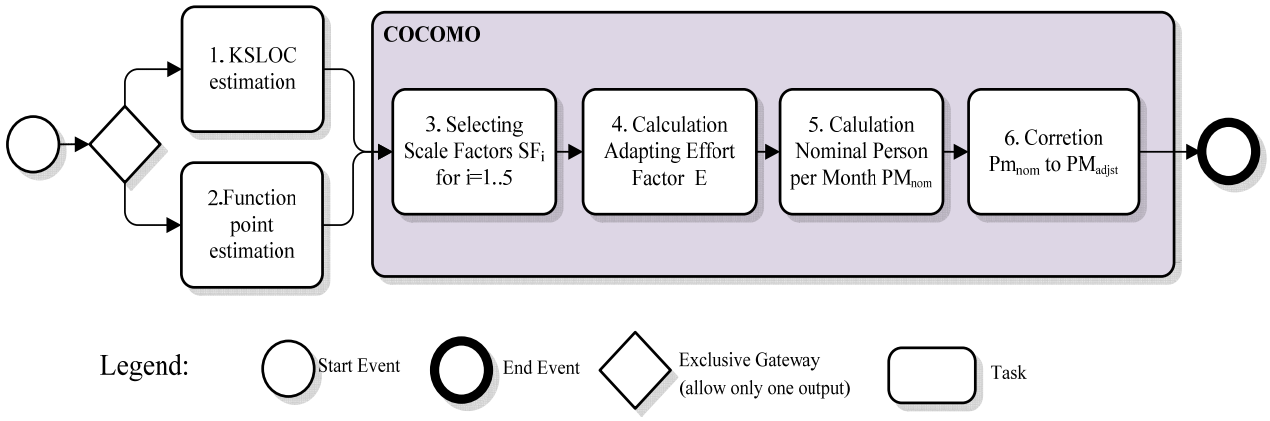

Figure 3. Sequence of processes in constructive cost model method

Table 1. The value of scale index [23]

i	Scale factor	Very low	Low	Normal	High	Very high	Extra high
1	typicality	6.20	4.96	3.72	2.48	1.24	0.00
2	flexibility	5.07	4.05	3.04	2.03	1.01	0.00
3	risk management	7.07	5.65	4.24	2.83	1.41	0.00
4	team maturity	5.48	4.38	3.29	2.19	1.10	0.00
5	process maturity	7.8	6.24	4.68	3.12	1.56	0.00

Knowing the SFs, one may determine the indicator adapting effort E in accordance with formula (1):

$$\mathrm{E} = \mathrm{B} + 0{,}01 \cdot \sum_{\mathrm{i}=0}^{5} \mathrm{SF_i} \tag{1}$$

where:

B – a constant 0.91 for COCOMO II model [23].

For instance, for the project in which typicality is low, flexibility must be high, risk management is very low and team maturity is normal and process maturity, the adapting factor, will equal:

$$\mathrm{E} = 0{,}91 + 0{,}01 \cdot (4{,}96 + 2{,}03 + 7{,}06 + 3{,}29 + 3{,}12) = 1{,}1146$$

Then, $\mathrm{PM_{nom}}$ nominal PM is calculated in accordance with formula (2) (process 5 in Figure 3):

$$\mathrm{PM_{nom}} = \mathrm{A} \cdot (\mathrm{Size})^{\mathrm{E}} \tag{2}$$

where:

Size – the number of code lines in KSLOC unit

A – a constant determined on the basis of previous projects = 2.94 [23].

Following the previous example, it is possible to calculate nominal implementation time for 8 KSLOC which equals ~ 30 man months.

$$\mathrm{PM_{nom}} = 2{,}94 \cdot 8^{1{,}1146} = 29{,}85 \approx 30 \text{ osobomiesiące}$$

For models from the first stages of Application Composition Model, Early Design Model [23] nominal time should be corrected (process 6 in Figure 3) in accordance with formula (3).

$$PM_{adjs} = \mathrm{PM_{nom}} \cdot \prod_{\mathrm{i}=1}^{7} \mathrm{EM_i} \tag{3}$$

where:

$\mathrm{EM_i}$ – effort multiplier.

For the models in another lifecycle stage, Post-Architecture Model (when system project is known), the formula for nominal PM was enriched by indicators. They define $\mathrm{PM_{nom}}$ changes, depending on system reliability, database size, product complexity, recycling level, analysts' and developers' skills, and so on. Similar to SF values, EM was determined empirically. The complete list can be found in method documentation [23]. The literature includes a number of examples of adapting COCOMO method [24, 25] with the use of fuzzy logic, inter alia [26–28].

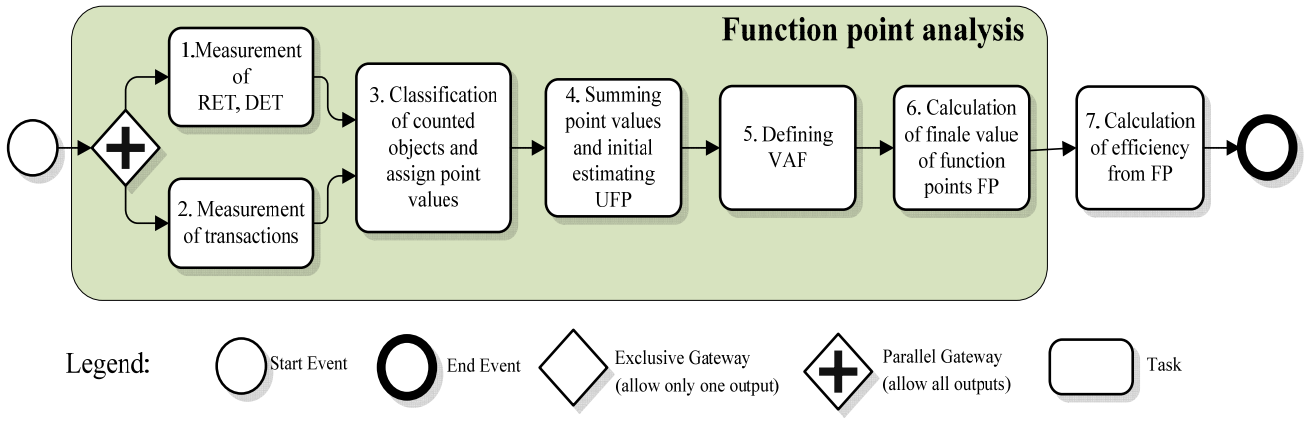

Figure 4. Evaluation process with the use of function points (FPs)

In practice [29], the COCOMO method is not used often. The reason for this is the need to use the size of code, which is known only at the stage of project documentation. Moreover, in project implementation, it is more frequent that the existing code is modified, rather than written from scratch. In case of code changes, COCOMO does not produce correct estimations. However, if the supplier uses FPs, evaluation can be started earlier – at the stage of implementation analysis. On the other hand, knowing the number of FPs inclines towards using evaluation by analogy method (presented in section 4.5), rather than COCOMO. This is because the supplier operates in known developer environment, in similar conditions and it is easier to determine multiplication factor and use it for every evaluation, rather than to do calculation according to COCOMO method each time.

3.2 FP analysis

The evaluation method proposed by A. Albrecht [30] requires the calculation of the number of FPs on the basis of specific requirements. FPs are a conventional measure complexity of function offered by the program. The number of FPs depends on the number of distinguished objects (reports, interfaces, etc), with the possibility of objects from one class (e.g. enquiries to database) having different values due to the level of complexity. For example, the value of FPs of a report referring to one table and two fields will be lower than the report referring to ten tables and eight fields. In the subsequent stage, COCOMO method, presented in section 3.1 or *evaluation by analogy* presented in section 4.5 can be used to calculate PM or costs. *FP analysis* is used to only to estimate software complexity or costs and its effects are not used at next stages of implementation (e.g. in the project). The method may be used for:

- software development,
- software modification,
- finished software product.

The evaluation process with the use of FP analysis is presented in Figure 4.

The FPs method is based on selecting five classes of objects in requirements or the ready program.

1) Internal Logic File (ILF) – a set of objects defining internal system data, e.g. a table in relational database,
2) External Interface File (EIF) – a set of objects exchanging data between IS e.g. an interface allowing data import from internet application,
3) External Inputs (EI) – responsible for inputting the data from the outside, e.g. screens, dialogues with the user,
4) External Outputs (EO) – responsible for outputting the data outside from the inside, e.g. prints, files send outside,
5) External Inquires (EI) – processes transferring internal data without modifying them, e.g. SELECT inquiry in Structured Query Language (SQL)[5].

Relations between classes of objects in the context of the environment are presented in Figure 5.

[5]SQL (Structured Query Language) – a structural language of enquiries used to build and modify databases and import or export data to and from the database.

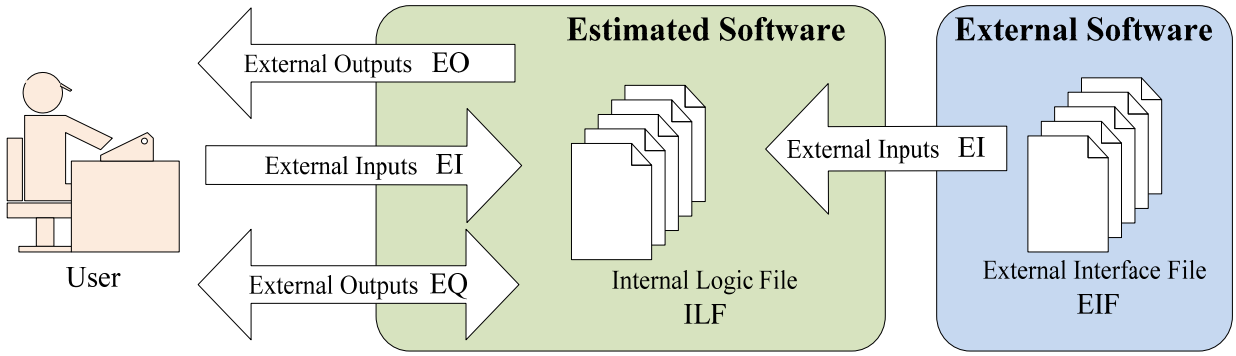

Figure 5. Classes of objects in function point analysis

The first two classes are related to data, and the other three to transactions. To make estimations in the first stage, the following indicators are used:

- RET (Record Element Type) – a unique, recognisable subgroup of elements given in ILF or EIF, which correspond to the record in the table;
- DET (Data Element Type) – a unique, identifiable field in ILF or EIF, which corresponds to the field in record;
- FTR (File Type Referenced) – recognisable by users, logically related data, which corresponds to files or relationally connected files.

All objects in classes must be identified and attributed with appropriate value of indicators (process 3 in Figure 4). ILF and EIF are described with RET and DET, while EO, EI and EQ with FTR and DET. For example: ILF #1: RET=3, DET=15. In this way, the values of unadjusted FP of a given object are read from table of weights (Table 2). For example, the aforementioned ILF #1 will have a value of 10 UFP.

Table 2. Weights of object classes, depending on DET and RET or FTR

ILF	**DET ∈<1;19>**	**DET ∈<20;50>**	**DET > 50**
RET=1	7	7	10
RET ∈<2;5>	7	10	15
RET > 5	10	15	15
EIF	**DET ∈<1;19>**	**DET ∈<20;50>**	**DET > 50**
RET=1	5	5	7
RET ∈<2;5>	5	7	10
RET > 5	7	10	10
EI	**DET ∈<1;4>**	**DET ∈<5;14>**	**DET > 15**
FTR < 2	3	3	4
FTR 2	3	4	6
FTR > 2	4	6	6
EQ	**DET ∈<1;5>**	**DET ∈<6;19>**	**DET > 19**
FTR < 2	3	3	4
FTR 2	3	4	6
FTR > 2	4	6	6
EO	**DET ∈<1;5>**	**DET ∈<6;19>**	**DET > 19**
FTR < 2	4	4	5
FTR ∈<2;3>	4	5	7
FTR > 3	5	7	7

Table 3. Example of cost estimation with the method of summing, computing and evaluating

Summed elements	Calculated number of objects [items]	Estimated Person per Month [h]	Total [h]
SQL queries	14	6	84
User interface windows	8	3	24
Printouts	6	6	36
		Total	144

Summing *UFP* values of all objects in all classes, the total value of unadjusted FPs is obtained (process 4 in Figure 4).

Value Adjustment Factor (VAF) considers the internal system complexity, unrelated to its functionality. Defining the value entails giving the impact of 14 factors, which may raise the system complexity (process 5 in Figure 4). The list of factors can be found in method documentation [31]. The estimation is done by an expert. Impact estimation requires attributing each of 14 categories with an impact factor from 0 to 5 (where 0 – no impact, 5 – strong impact). VAF value is calculated from formula (4):

$$\mathrm{VAF} = \mathrm{B} + 0{,}01 \sum_{i=0}^{14} C_i \tag{4}$$

where:

B – empirically determined constant value 0.65 [31],

C_i – impact value of i-th factor.

On the basis of VAF, the final values of FPs are calculated by correcting the unadjusted FPs (process 6 in Figure 4) according to formula (5):

$$\mathrm{FP} = \mathrm{VAF} \cdot \mathrm{UFP} \tag{5}$$

Knowing the FP value, efficiency can be determined with two methods (process 7 in Figure 4):

- calculating into KSLOC with empirically determined values from the calculation table [32] and then using the COCOMO method to define PM
- if the organisation owns historic data, FP value can be directly calculated into PM, using Estimation by Analogy method.

The source of complete and updated documentation of the method is the website of International Function Point Users Group [33]. The idea of FPs was used in User Case Point [34] that is reliant on modelling agreeing with UML[6].

[6] UML (Unified Modelling Language) – a formal language used to model different systems, developed by the Object Management Group.

The condition of using FP is knowing the complete set of specific requirements. Thus, it can be used only from the stage of implementation analysis onward. The algorithm guarantees objective and repeatable evaluations. Unfortunately, the level of evaluation complexity, in comparison with non-algorithmic methods, is a barrier for its common use.

4 Non-algorithmic methods of software evaluation

4.1 Summing, computing and evaluating

The method concerns searching the available documentation (e.g. pre-implementation analysis, feasibility studies) and quantifiable objects, e.g. requirements, functions, use cases, stories, reports, windows, database tables and classes. Each identified object that can be summed is attributed (computed) with estimation constituent (cost or time). The estimated values are the function (6) of the objects constituting an information project:

$$f(x) = \sum_{i=1}^{N} C(x_i) \tag{6}$$

where:

x – calculated object,

N – the number of summed objects,

C – computer cost of the object.

An example of its use can be the valuation of developing a sales reporting module. If the authors managed to select the premises, such as SQL queries, interface windows, users and printouts, it is possible to define a unit PM. Thus, one can evaluate the costs of developing the whole module, as presented in Table 3

Table 4. An example of evaluation with individual expert evaluation method

Valuated work	Most optimistic value [h]	Most likely value [h]	Least favourable value [h]	Calculated value
SQL queries	45	81	108	84
User interface windows	14	22	32	24
Printouts	25	33	45	36
			Total	144

The cost of works depends on their value, in the above case, man hours in a given supplier's organisation.

The method is complex provided the source documentation allows determining the summed objects. One of the failures is the high risk of omitting objects or ranges of work that influence the value of the whole project, for example, ignoring supplementary tables or costs of developing filtering inquiries while evaluating the costs of interface windows. An important stage in this method is the evaluation of individual objects' costs. This can be done with individual expert evaluation method or group expert evaluation method. The method is efficient in projects with a small number of object types identified but are plentiful, e.g. 30 reports, 25 SQL inquiries and 18 interfaces. These types of data are infrequent in the stage of trade talks, but often present in feasibility studies and pre-implementation analysis.

4.2 Individual expert evaluation

The method of valuation by individual expert estimation is the most frequently used method, not only in software development [35], but also in other IT enterprises such as implementations and modifications. Research conducted in the USA in 2002 showed that as many as 72% of the valuations are done with this method [36]. In the first stage, the method requires selecting experts with appropriate knowledge and experience. Then experts evaluate the ranges they were bestowed. In order to reduce the evaluation errors, the method was modified with multiple evaluations for different versions of implementation. Such a technique, called PERT (Program Evaluation and Review) [15, 37], involves analyses of the most optimistic case, the most probable case and the worst case. However, it is different from critical path analysis (CPM [38]) because it is used to evaluate independent tasks only. After previous decomposition processes, the information about relations between tasks was lost. The expected evaluation has the following form:

$$f(x) = \sum_{i=1}^{N}(Cp(x_i) + 4 \cdot Co(x_i) + Ck(x_i))/6 \quad (7)$$

or, considering experts' tendencies to lower their evaluation:

$$f(x) = \sum_{i=1}^{N}(Cp(x_i) + 3 \cdot Co(x_i) + 2 \cdot Ck(x_i))/6 \quad (8)$$

where:

Cp – the most optimist value of the i-th task

Co – the most likely value of i-th task

Ck – the most probable value of i-th task

N – the number of tasks in the project.

Valuation of sales reporting module can be an example of the method. If the authors managed to select the premises, such as SQL queries, user interface windows and printouts, the expert can estimate the most optimistic, the most likely and the least favourable amount of work load. Then, one can determine the costs of programming work on the module (additionally, considering the inclination to lower the evaluation), as presented in Table 4.

The precision of results depends exclusively on expert's experience. The criteria of expert selection are determined imprecisely. The influence of personality has such importance that longer experience does not guarantee more precise evaluations. It happens that experts who are known for overvaluation or undervaluation are unpredictable.

4.3 Group expert evaluation

The method involves presenting the same range of work to more than one expert. In an unstructured version of the method (group review), the experts decide about the valuation or its range as a group. In a structured version called Wideband Delphi [15, 16], the experts' work is done in a formalised way and its result is a scoring evaluation.

Table 5. An example of valuing with the construction and decomposition method

No.	Range of work	Estimated value [h]
A.	Preparatory work:	
A.1	• software installation	
A.1.1	- application server software installation	5
A.1.2	- server database software installation	4
A.1.3	- user software installation	14
A.2	• database import	
A.2.1	- export from "old" verification system	8
A.2.2	- import to the new system	11
A.2.3	- reconstructing indexes and data verification	16
A.2.4	- back-up copy parameterisation	2
B.	Modification movement:	
B.1	• modification movement in the area of finances	34
B.2	• modification movement in the area of personnel	21
B.3	• modification movement in the area of production	120
C.	Trainings:	
C.1	• financial departments	16
C.2	• HR department	16
C.3.1	- hull production staff	4
C.3.2	- wind station staff	4
	Total	**275**

The work of experts in groups is more expensive than individual work; however, a method's advantage over individual evaluation is the decrease of personality factors' importance. In spite of different experience, characters and inclinations, experts will either reach a common ground or, as in case of Widebrand Delphi type, the conclusion of problem is reached by attributing pre-selected points. Wanting to minimise the costs of first stages of implementation, suppliers decide to engage a larger number of experts to do the same calculations.

The estimation method is used frequently at initial stages of IT projects in situations of high uncertainty of requirements.

4.4 Decomposition and reconstruction

Decomposition and reconstruction is a popular method due to its intuitiveness and universality. It is used in situations when whole project evaluation generates difficulties, e.g. resulting from work heterogeneity. In the practice of IT project implementation [29], there are very few projects that can be evaluated without this method.

The method involves decomposing the range into a number of components. The method of division is arbitrary and depends on project specifics. Consultants frequently undertake evaluation with the Work Breakdown Structure (WBS) method. Having done the division, the parts of objects are estimated and undergo further division with the same or another method. Even though the literature lists this method as equal to others [4], its role in the evaluation process is different. Project evaluation is started in this method, but after decomposition, other methods of elemental evaluation are selected. The depth of division depends on the method that is going to be used at another stage. A detailed description of decomposition method according to WBS can be found in literature [15, 39–42].

One example of this approach is the evaluation of IT system version change. The works can be decomposed in the manner presented in Table 5.

Table 6. Example of calculating multiplication index in evaluation by analogy

Parts of decomposed project	Completed project [h]	New project (estimation) [h]	Multiplication index
Database table	60	42	0,70
User interface	43	18	0,42
Reports	54	32	0,59
SQL queries	85	54	0,64
Basic classes	28	14	0,50

A significant element in this method is the manner of division. Suppliers with little experience in cost evaluation can do this division in a way that will generate significant errors in elemental evaluations. If the ranges of works in basic parts of the project are too excessive or heterogeneous in terms of the technology of development, the methods will generate unreliable results. The practice of modification estimation [29] suggests that the effects of decomposition method should be the works evaluated in a few or a dozen man hours.

4.5 Evaluation by analogy

This method involves dividing the project into components that already exist in a completed project. Evaluating the selected parts, one may calculate the ratio of two project sizes (new and the completed one). Knowing the relations between the sizes and the costs of the completed project, one may estimate the value of the new project.

One example of this method is presented in Table 6. The average multiplication factor for the above example is 0.57. Knowing this result and the value of the completed project, one can estimate the value of works.

The difficulty lies in collecting historic data from similar projects and structure as the evaluated project. An additional problem is the selection of a representative part of the decomposed project, which is a basis for multiplicity factor. Ignoring significant objects may increase the evaluation error.

The input data for this method come from project documentation, e.g. interfaces, reports and SQL queries. Only for this type of objects can the multiplication factor be calculated. The use of requirements, even the specific ones, does not allow for calculating the multiplication factor, and thus the whole evaluation must be performed. Evaluation by analogy is also used to calculate PM on the basis of FPs.

4.6 Valuation based on substitution

Similar to the previous method, this method requires the knowledge of costs of previously completed projects in the organisation of standard objects (interfaces, reports, etc). Depending on the version of method, the objects can be grouped differently. For example, Putnam [37] and Humphrey [43] selected different classes of objects: very small, small, medium, large and very large. Another method of classifying the objects is a standard component method [4] used to evaluate object software. The division can then be as follows:

- dynamic WWW websites,
- static WWW websites,
- data tables,
- reports,
- business rules.

If the IS system supplier uses extreme software or close to Agile methods [44], the so-called “stories” might be a standard element.

Then, the groups of objects are attributed with average cost values, e.g. number of lines of code, man hours or man days. The objects from a new project must be classified in the same manner. Then their sum can be calculated.

An example of such an approach is the project of white goods’ sales. The cost estimation is presented in Table 7.

Table 7. An example of valuation by substitution

Standard classes of components	Average value of costs [h]	Number of objects in a class	Value of costs [h]
dynamic WWW websites	7	5	35
static WWW websites	2	18	36
data tables	7	16	112
reports	5	9	45
business rules	12	5	60
		Total	288

The supplier determines the average cost of works in a given class, e.g. the cost of building one static website – 2 man hours – on the basis of previous historic data. In a new project, the works are attributed with appropriate classes, e.g. dynamic websites – 5 pcs. Reports – 9 pcs. Then the old objects are substituted with the new ones.

It is only the experienced organisations that are able to use this method. Not only does it require having historic data but also weights must be attributed to them. One should remember that the weights will be valid only till the developer's tools or programming style is changed.

As in the previous method, this should be used when classes of programming objects are known. One exception is the organisations using extreme or agile software. In this case, the cost of "stories" documented at the stage of talks to clients may be substituted with historic data. The practice of evaluations [29] indicates that the method can be incidentally used at an earlier stage (pre-implementation analysis), when only the requirements are known.

5 Conclusions

In conclusion, one should notice that implementation of the first stages of software lifecycle provides more and more information about the planned solution, on the one hand, and there are a number of evaluation methods available, on the other.

At the stage of trade talks, the supplier holds a complete list of client's requirements. The set includes general requirements, a few specific requirements and a subset of undisclosed requirements. Specific requirements, such as specific printouts, reports and inquiries, may be evaluated in summing, computing and evaluating method. Other requirements, especially undisclosed ones, will be appropriate only for the individual expert method and group expert method. Experts, who have historic knowledge, may predict that if the client specifies one requirement, it will implicate a set of other functions that might be revealed only at another stage of works.

After completing feasibility study or a pre-implementation analysis, the suppliers hold a complete set of specific requirements. COCOMO methods are used in case of lack of reasons for using KSLOC. For *FP* analysis there are sufficient input data. Summing, computing and evaluating and evaluation by substitution can be used in project valuation; however, using requirements as input data will generate significant errors. Individual expert analysis and group expert analysis with complete requirements generate significant errors in results. At this stage, it is too early for evaluation by analogy because requirements as input data will generate errors.

Project of changes provides additional information related to implementation – data structure, information on processes, objects and so on. Only from this stage can KSLOC be estimated in order to use COCOMO method. For summing, computing and evaluating and evaluation by substitution methods, input data significantly limit the errors of results. Only at this stage can evaluation by analogy be used. For other methods, the quality of estimation is increased only slightly by including additional data. For individual expert analysis and group expert analysis, the data taken in project documentation are inappropriate. Deliberate use of these methods causes significant errors.

By classifying the possibility of using evaluation methods in the following way, we can present them in a table (Table 8):

0 – impossible to use, lack of data,

1 – usable but generating significant errors, because the data are inappropriate,

2 – usable and generating satisfactory results

Table 8. Usefulness of evaluation methods

Method \ stage	Trade talks	Feasibility study / pre-implementation analysis	Project of system changes
COCOMO	0	0	2
Function point analysis	0	2	0
Summing, computing and evaluating	1	1	2
Individual expert analysis	1	2	0
Group expert analysis	1	2	0
Evaluation by analogy	0	0	2
Evaluation by substitution	0	1	2

An alternative use of the evaluation method is presented In Figure 6. Because method usability depends on input data, quality and type of requirements and project components can be the basic criteria for choosing a given method.

Due to different characters of elements constituting IS (business rule software, data interface software, reports, etc), evaluation of cost and implementation time should start with decomposition of work into tasks and groups of tasks. Then, still in the phase of trade talks, they can make attempt evaluation by experts (group of experts), if the data obtained from client are of appropriate quality. If specific requirements are identified, the supplier can additionally use summing, computing and evaluating method.

Fundamentally, the data of quality that allow for evaluation of cost and implementation time are obtained in subsequent stages, i.e. feasibility study and pre-implementation analysis. Then, *FP analysis* or *evaluation by analogy* can be used.

At the system development stage, the analysis of implementation costs can be based on very detailed data, and thus it is burdened by serious errors. The array of methods can be supplemented by *evaluation by analogy* or COCOMO.

The methods of selecting the manners of evaluating costs of IS implementation in the context of input data quality are going to be the topic of future research. The authors will also search for methods for quick definition of precise input data, even before the stage of feasibility study.

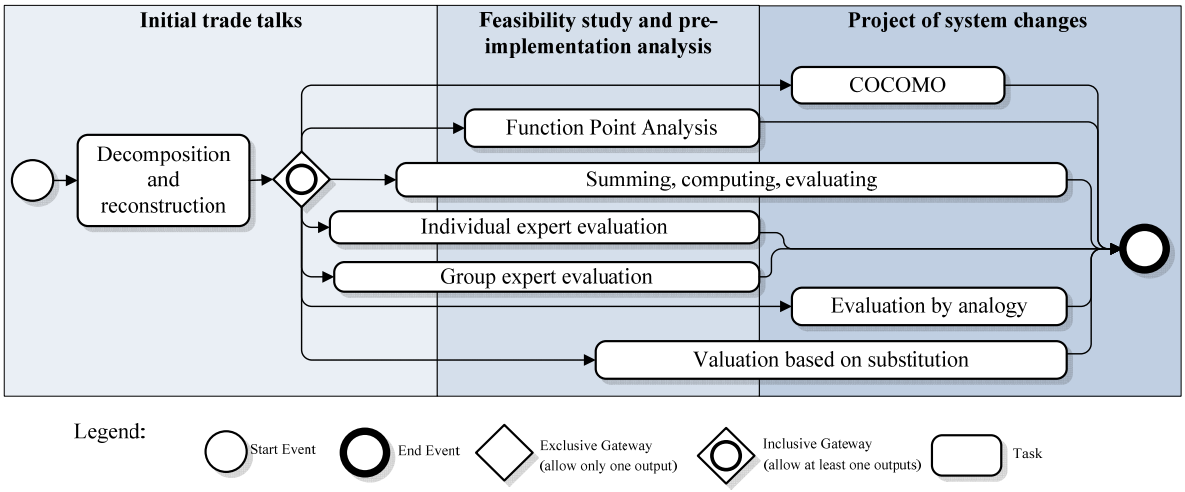

Figure 6. Suggestion for an alternative use of evaluation methods. COCOMO, constructive cost model

6 Bibliography

[1] Burns M. - *How to select and implement an ERP System*. 2005. [Online]. Available: http://www.180systems.com/ERPWhitePaper.pdf.

[2] Kotarba M. - *Problems occurring during modification to the standard ERP system and ways overcome them on the example of the implementation of Oracle.* JD Edwards EnterpriceOne enterprise in the food industry. Support Systems Organization, Katowice, 2007.

[3] Lech P. - *Integrated management systems ERP / ERP II. Use in business, implementations.* Difin, Warszawa 2003.

[4] McConell S. - *Software Estimation: Demystifying the Black Art*. Microsoft Press, 2006.

[5] Meli R. - *Early Function Points: a new estimation method for software project* [in] WSCOM97, Berlin 1997.

[6] Santillo L., Conte M., Meli R. - *Early &Quick Function Point: Sizing More with Less* [in] Metrics 2005, 11 th IEEE Intl Software Metrics Symposium, Como, Italy, 2005.

[7] Boehm B., Abtsi C., Chulani S. - *Software Development Cost Estimation Approaches – A Survey*. Annals of Software Engineering, tom 10, nr 1-4, pp. 177-205, 2000.

[8] Jaszkiewicz A. - *Software Engineering.* Helion, 1997.

[9] *BPMN*. Object Management Group, 2013 [online]. Available: http://www.bpmn.org/.

[10] Sacha K. - *Software Engineering*. Wydawnictwo Naukowe PWN, 2010.

[11] Frączkowski K. - *IT project management.* Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2003.

[12] Phillips J. - *IT Project Management: On Track from Start to Finish*. McGraw-Hill, Inc. Nowy Jork, 2004 .

[13] Radosiński E. - *Systems in dynamic analysis of decision-making. Decision support systems. Simulation modeling. Intelligent Techniques*. Wydaw-nictwo Naukowe PWN, Warszawa 2001.

[14] Justynowicz K. - *Pre-Implementation analysis becoming popular.* Oct. 2007. [online]. Available: http://www.bcc.com.pl/akademia-lepszego-biznesu/analiza-przedwdrozeniowa-coraz-popularniejsza.html.

[15] Stutzke R.D. - Estimation Software-Intensive Systems, Upper Saddle River, New York: Addison-Wesley, 2005.

[16] Allen P., Frost S. - *Component-Based Development for Enterprise Systems, Applying the Select Perspective*. Cambridge: Cambridge University Press, 1998.

[17] *Select Business Solution* [Online]. Available: http://www.selectbs.com.

[18] *ARIS* [Online]. Available: http://www.softwareag.com.

[19] Yourdon E. - *Death March.* Yourdon Press, 2003.

[20] Booch G., Rumbaughi J., Jacobson I. - *The Unified Modeling Language*. User Guide., Rational Software Corporation.: Addison Wesley Longman, Inc, 1990.

[21] Boehm B. - *Software Engineering Economics*. New York: Englewood Clifs, 1981.

[22] Boehm B.W. - Software Cost Estimation with COCOMO II, Prentice Hall, 2000.

[23] Baik J. - *COCOMO II, Model Definition Manual*. Version 2.1, Center for Software Engineering at the University of Southern California, 2000.

[24] Hele J., Parrish A., Dixon B., Snith R. - *Enhancing the Cocomo estimation models*. Software, IEEE, Volume:17, Issue: 6 , 2000, pp. 45-49.

[25] Aljahdalii S., Sheta A. - *Software effort estimation by tuning COOCMO model parameters using differential evolution* [in] Computer Systems and Applications (AICCSA), IEEE/ACS International Conference on, Hammamet, 2010.

[26] Fei Z. - *f-COCOMO: fuzzy constructive cost model in software engineering*. [in] Fuzzy Systems, IEEE International Conference on, San Diego, CA, 1992.

[27] Satyananda R.C. - *An Improved Fuzzy Approach for COCOMO's Effort Estimation using Gaussian Membership Function*. Journal of Software, Vol. 4, No. 5, July 2009, pp. 452-459.

[28] Attarzadeh I. - *Improving estimation accuracy of the COCOMO II using an adaptive fuzzy logic model* [in] Fuzzy Systems (FUZZ), 2011 IEEE International Conference on, Taipei, 2011.

[29] Plecka P. - *Selected Methods of Cost Estimation of ERP Systems' Modifications.* Zarządzanie Przedsiębiorstwem, nr 3/21013.

[30] Albreht A. - *Measuring Application Development Productivity* [in] Proceedings of the Joint SHARE, GUIDE, and IBM Application Development Symposium, Monterey, California, USA, 1997.

[31] *IFPUG*, Function Point Counting Practices: Manual Release 4.1, Westerville, OH: IFPUG, 1999.

[32] *The QSM Function Points Languages Table.* QSM, 2013. [online]. Available: http://www.qsm.com/resources/function-point-languages-table.

[33] *International Function Point Users Group* [online]. Available: http://www.ifpug.org/.

[34] Clemmons R.K. - Project estimation with use case points. The Journal of Defense Software Engineering , pp. 18-22, 2006.

[35] Jorgensen M. - *A Review of Studies on Expert Estimation of Software Development Effort*. Journal of Systems and Software, Vol. 70, No. 1-2, February 2004, p. 37–60,.

[36] Kitchenham B., Pfleeger S.L., McColl B., S. Eagan - *An empirical study of maintenance and development estimation accuracy*. Journal of Systems and Software, Vol. 64, No. 1, 2002, pp. 57-77.

[37] Putnam L.H., Myers W. - *Measures for Excellence: Reliable Software on Time.* Within Budget, Englewood Cliffs, NY: Yourdon Press, 1992.

[38] Fondahl J.W. - *The History of Modern Project Management Precedence Diagramming Methods: Origins and Early Development*. Project Management Journal, Vol. XVIII., No. 2, 1987.

[39] NASA - *ISD Wideband Delphi Estimation*. 09 2004. [online]. Available: http://software.gsfc.nasa.gov/assetsapproved/PA1.2.1.2.pdf.

[40] Tausworthe R. - *The work breakdown structure in software project management*. Journal of Systems and Software, Vol. 1, 1984.

[41] Norman E., Brothertoni S., Fried R. - *Work Breakdown Structures: The Foundation for Project Management Excellence.* John Wiley & Sons, 2010.

[42] Haugan G. - *Effective Work Breakdown Structures*. Project Management Institute, 2002.

[43] Humphrey W.S. - *A Discipline for Software Engineering*. Addison Wesley, 1995.

[44] Cohn M. - *Agile Estimating and Planning.* Upper Side River, NY: Prentice Hall PTR, 2005.