International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:10, No:2, 2016

Software Engineering Inspired Cost Estimation for Process Modelling

Felix Baumann, Aleksandar Milutinovic, Dieter Roller

Abstract—Up to this point business process management projects in general and business process modelling projects in particular could not rely on a practical and scientifically validated method to estimate cost and effort. Especially the model development phase is not covered by a cost estimation method or model. Further phases of business process modelling starting with implementation are covered by initial solutions which are discussed in the literature. This article proposes a method of filling this gap by deriving a cost estimation method from available methods in similar domains namely software development or software engineering. Software development is regarded as closely similar to process modelling as we show. After the proposition of this method different ideas for further analysis and validation of the method are proposed. We derive this method from COCOMO II and Function Point which are established methods of effort estimation in the domain of software development. For this we lay out similarities of the software development process and the process of process modelling which is a phase of the Business Process Management life-cycle.

Keywords—Cost Estimation, Effort Estimation, Process Modelling, Business Process Management, COCOMO.

I. Introduction

THE creation and editing of business processes is handled in the industry partially unsystematic in the sense that no standardised method for process model creation is used. Furthermore, initial estimation on the cost and effort for process creation is not performed. Through dynamically changing markets and induced pressure for change in businesses, process management has become a dominant topic for the enterprises [1],[2],[3],[4].

The modelling of processes and as a consequence thereof the IT support with the execution, monitoring and especially the improvement is already common practice [1],[3].

Albeit metrics for quality and faultiness exist, those quality attributes enter existing tools and software and the actions of users only slowly. Closely related to these metrics is the question for cost and effort estimation for the creation, maintenance and adaption of process models [5],[6].

Contrary to related fields, like software engineering (SE) or software development (SD), an accepted practice for such estimations is missing. As process models or processes in general can be regarded as highly critical to the success of an enterprise as well the claim that the quality of a product results from the quality of the underlying process - "Process equals Product" [7], the modelling and documentation associated with it and the process itself is vital to the leadership of an enterprise.

This work gives an overview of current research efforts into cost and effort estimation in general. Based on the findings we derive a method to estimate the cost of process modelling beforehand.

Felix Baumann is with the University of Stuttgart, Germany (e-mail: felix.baumann@informatik.uni-stuttgart.de).

Business process modelling is a sub task of the general methodology of business process management which is to be described by Müller[3] as the effort to document, create and improve business processes and the IT support for it.

Business processes - hereinafter processes, as we are only having these as the focus of our research - are described as connected working steps that produce a specified output using a specified input [4].

In this work, we do regard cost and effort as interchangeable, as the cost for modelling is mainly based on the cost for the modellers time, other cost inducing factors like licences, equipment, soft- and hardware cost and support are regarded as irrelevant in this paper[8].

A. Business Process Management

A business process is the sequence of correlated tasks that produce a defined achievement using defined inputs[4]. Business processes contain knowledge about customer(groups), applications, success factors, strengths and weaknesses, competition strategies and business goals. With the knowledge of these aspects it is possible to increase the added value of an enterprise[4]. A business process begins and ends at a customer according to Gadatsch[9] and Schmelzer[10]. Customers are not the only stakeholders that formulate requirements for Business Process Management. Business leadership also have their requirements on BPM mostly in the form of key performance indicators (KPIs) in order to inquire about their efficiency and effectiveness which enables them to evaluate their actions and ideas[10]. BPM is regarded as a bridge between business administration and IT and helps enterprises to be more successful due to IT support. We use the term of organization interchangeably for public sector entities and private enterprises in this work. We consider small and medium sized enterprises (SME) at the focus of this work as they conduct a large number of BPM projects[1].

1) BPM Life-Cycle: The flow of BPM is described by the BPM life-cycle[11]. This life-cycle clarifies the procedure of BPM and explains the associated phases. The phases of the BPM life-cycle are as following:

1) Analysis
2) Modelling
3) Implementation
4) Execution
5) Monitoring
6) Optimization

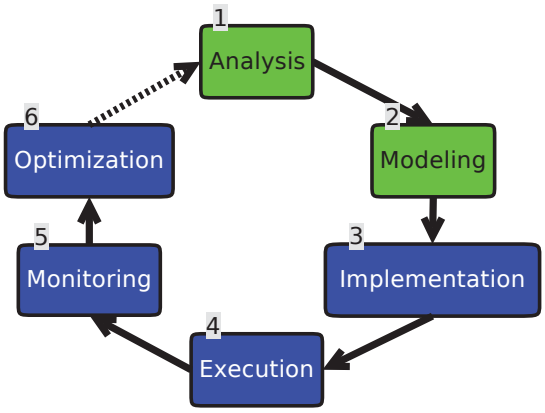

Fig. 1 BPM Life-Cycle

The cycle starts with the analysis phase (sometimes called design phase) - See Fig. 1 indicated with a 1. In this phase existing processes are examined as well as new processes to derive value from are being identified. Furthermore, KPIs are analysed and requirements for the description or improvement of the processes are determined. The result of this phase is an artefact that is referred to as process-as-is model - describing the current available process - as well as a model that describes how the process is to become. The definition of process-as-is model is not entirely truthful according to Freund[1] as this is not the model that will be modelled in the modelling phase. We will refer to this model as process documentation (which can also be a process model but does not necessarily have to be).

In the modelling phase the process model itself is being modelled. This is done in a modelling language or notation that has to be decided upon beforehand. Tool support is available for most modelling languages, albeit their quality differs significantly. Tool support varies from graphical modelling environments, modelling repositories, web-based graphical modelling environments with cloud and collaboration support to drawing tools without embedded intelligence or support. Knowledge from the first phase (Analysis) is incorporated in this second phase. We describe the implementation phase as an additional phase in order to distinguish the modelling from the implementation. Some authors regard this as one single phase as modelling and implementation are being used interchangeably by some authors [1] this is also dependent upon the selected modelling language, tools and the modeller. We draw the line between those two phases as we see it necessary to refine the process model in order for it to be deployed on a workflow management system (WFMS) or business process management system (BPMS) and executed using IT support [12].

The execution phase is when the process model deployed in a BPMS if the BPM is backed by IT support or it can be executed without IT support as standalone processes where involved parties are themselves responsible for following the given process model.

As BPM is based on a continuous improvement process[1] there is always a monitoring phase in which data is collected during execution. Analysing this data is not to be confused with testing, which can be regarded as a separate phase after implementation. We do not describe the testing phase as this is out of scope of this work. Data acquired in the execution phase can be used in the optimization phase to improve process models. From this phase the life-cycle can start over but with every increment of the life-cycle the process is to be improved as more data is available for better decisions and analysis in every phase.

B. State of the Art

The topic of business process management (BPM) is currently discussed not only academically but also from a wide range of practitioners. It is being developed and refined almost daily which can be observed by the crowing number of participants in the BPM-network [1], the occurrence of a multitude of new blog posts [2], the emergence of new languages and modelling standards rooting in BPMN - like CMMN or DMN. Furthermore, BPMN 2.0 is evolving and the acceptance and prevalence of this method is growing [1]. BPM can be regarded as established and mature enough for production[9].

Surveys like the yearly global CIO-study from IBM [3] and the survey on situation analysis by Müller [3] at the University of Stuttgart give further indication that BPM is researched academically and further refined.

C. Research Method

For this work, we have researched in university libraries, the German Nationalbibliothek [4], journals[5], with Google Scholar [6] and searched the Internet using Google [7] for the following search terms:

- (Effort|Cost) Estimation (Method*) (Software Engineering|Business Process Management|BPMN|Business Process Modelling)

We have discovered the following number of results per search term (using Google Scholar).

- "Effort Estimation" "Software Engineering" **8190**
- "Effort Estimation" "Business Process Management" **543**
- "Effort Estimation" BPMN **112**
- "Effort Estimation" "Business Process Modelling" **166**
- "Cost Estimation" "Software Engineering" **15200**
- "Cost Estimation" "Business Process Management" **861**
- "Cost Estimation" BPMN **326**
- "Cost Estimation" "Business Process Modelling" **413**
- "Effort Estimation Method" "Software Engineering" **315**
- "Effort Estimation Method" "Business Process Management" **5**

[1]https://network.camunda.org
[2]http://www.column2.com,http://www.bpm-plus.de,http://www.bpmn-buch.de
[3]http://www-935.ibm.com/services/us/en/c-suite/csuitestudy2013/
[4]http://www.dnb.de
[5]Business Process Management Journal, Knowledge and Process Management, International Journal of Business Process Integration and Management
[6]https://scholar.google.com
[7]https://www.google.com

- "Effort Estimation Method" BPMN **2**
- "Effort Estimation Method" "Business Process Modelling" **4**
- "Cost Estimation Method" "Software Engineering" **344**
- "Cost Estimation Method" "Business Process Management" **10**
- "Cost Estimation Method" BPMN **7**
- "Cost Estimation Method" "Business Process Modelling" **6**

From these results we have analysed the top 10 results (sorted by relevance in Google Scholar) for applicability and discarded non-related findings. It is obvious that the more mature field of software engineering yields more results than the more specialized query for business process modelling or even for the language specific query for BPMN.

Through reviewing these articles it is obvious that the topic of effort estimation is mentioned in scientific literature but not extensively researched. Furthermore, consulting companies seem to have developed internal models and methods due to the nature of their operation but those are likely to be not scientifically backed and founded as well as not published. Based on these observations it can be concluded that there are currently no established and recommended standardized practices of how to estimate the cost of business process model creation within BPM.

The proposed forecast models published are not to be applied in the modelling phase as they require input values that are only present after the modelling phase. Effort estimation is only possible in the implementation phase as earliest [13],[6],[14],[15],[16].

D. Approaches and Methods

A complete estimation for BPM projects is described by Bankhofer and Nissen [17], Çulha and Dogru[18] and Thiemich and Puhlmann [19] by combining all phases of the BPM life-cycle or by combining the modelling phase and implementation phase. In these works no direct calculation of the modelling phase is described. It can only be derived as a proportion of the total cost. Further research is conducted to identify factors that influence the modelling effort and should be considered relevant for our proposed method. The following influencing factors have been identified and are considered in our estimation approach. The aspect of process maturity as described by Allweyer [20] and Fisher [21], input factors and model size [13], [14],[15], complexity [22], [5], [6], risk management [23], [24], teamwork and personnel [1], [9], [25], [4], [26] and best practices [27],[28].

The identified relevant works base their proposals for effort estimation on the existence of an already existing process model. This reflects to having completed the modelling phase and estimating the implementation phase. Furthermore, this process model has to be complete and correct[15]. Real world situations differ from these assumptions as many companies do not have correct process models available, processes and their documentation are incomplete or obsolete or in other cases do not exist at all [1], [29].

The generation of a process model is similar to conducting a software creation project, as a process model is akin to software in the regard that it can be executed on a BPMS (as it is the case with BPMN 2.0) or can be displayed as code (see XML representation of BPMN 2.0 [8]). Another example is the WS-BPEL modelling process which is according to the specification[9] and partially lacking graphical support in modelling tools (see Eclipse BPEL Designer [10] not primarily visual but code centric[30].

We compare the phases from the BPM life-cycle (Fig.1) with the phases from software development and display the proposed matching in Table I. This matching encourages us to see similarities between cost estimation methods from software development which we will discuss further in this work. Based on this possible adoption we contrast methods and processes of effort estimation from SD to BPM and evaluate their portability to this domain.

TABLE I
MAPPING OF BPM AND SD LIFE-CYCLE PHASES (ADAPTED FROM [13])

Phases BPM life-cycle	Phases Software Development
1. Analysis	1.1 Requirements Specification
2. Modelling	1.2 System and Software Design
3. Implementation	2.1 Implementation and Unit Test 2.2 Integration and Systems Test
4. Execution 5. Monitoring 6. Optimization	3. Operation and Maintenance

We will discuss selected effort estimation methods from SD regarding their ability for adaption for BPM. We omit the discussion of agile methods as they require existing process models as a basis which are considered missing for our method [13], [18]. Baklizky et al. [13] propose in "Business Process Points-A Proposal To Measure BPM Projects" a method to enumerate BPM project points which allows them to estimate the project size. This method also requires existing models and accompanying documentation to be present and is not applicable to the modelling phase. This method is applicable for the implementation phase.

E. Effort Estimation Methods from Software Development

We briefly describe a selection of identified effort estimation methods from the domain of software development. This list contains methods that we discuss for further adaption for our problem setting.

1) Function Point: Function Point (FP) was developed in 1973 by Albrecht [31], [32] and is an analogy and weighting method that estimates the effort for a project based on its size and complexity. Function Point is being used by IBM since 1981 [32] which displays its acceptance. For FP the functions of a software are counted and in classified in categories. Based on the five categories of a function (Input, Output, Files, Inquiry, Interface) the user is able to get a number of "Unadjusted Function Points" (UFP) depending on the weighting of each function in one of the categories of complexity (Low, Average, High). In the next step the degree of influence [32] is calculated over 14 technical complexity

[8]http://www.omg.org/spec/BPMN/2.0
[9]https://www.oasis-open.org/committees/tc_home.php
[10]https://eclipse.org/bpel

factors where each factor is weighted between 0 (no influence) and 5 (high influence). There is a maximum of 70 points for this factor. Afterwards the Total Complexity Factor (TCF) is to be calculated by the division of the sum of all factors by 100 and then adding 0.65. Afterwards the UFP are multiplied by the TCF in order to receive the adjusted Function Points. The Result is therefore in the range of 65% and 135% of the original UFP. The factors are based on empirically research studies by Albrecht. With the help of an experience table, initially created by the IBM, it is possible to translate the Adjusted Function Points to person months (PM). Enterprises can develop their own experience table which possibly reflects their own situation better and increases accuracy over more conducted projects.

2) Use Case Point: The Use Case Point (UCP) method was developed in 1993 by Karner [33] and is based on the procedure of the Function Point method. UCP is applicable for object oriented software development but FP and UCP are not directly mappable [34]. Frohnhoff describes the Use Case Point method as follows: "A Use Case covers a contract between the stakeholders of a system regarding the system's behaviour. The Use Case describes the system's behaviour under various conditions as a reply to request from a stakeholder, called actor. An Actor triggers an interaction with the system in order to reach a specific goal. The system replies under consideration of the interests of all stakeholder. Depending on the special system request and the request context there can be a different ordering of the system's behaviour or different scenarios. The Use Case encompasses all possible different scenarios". Due to the similarity of Use Case Point to Function Point in its execution procedure we omit its description in this work.

3) COCOMO: The Constructive Cost Model (COCOMO) was developed by Boehm [35] in the early 1980s. Input values for the effort estimation are KLOC (Kilo Lines of Code) [11]. The Project Effort (PE) in person months (PM) is defined by (1):

$$\mathrm{PE} = m * \mathrm{KLOC}^{n} \tag{1}$$

The factor **m** and the metric number **n** are based on empirical data. Projects are classified in three complexity classes:

- "organic mode" for easy projects (**m=2.4**, **n=1.05**)
- "semi-detached" for medium difficult projects (**m=3**, **n=1.12**)
- "embedded" for complex projects (**m=3.6**, **n=1.2**)

The COCOMO model contains a description of how each project is to be classified according parameters like tool support, environment, team size and schedule. Boehm[36] developed an update to COCOMO which is called COCOMO II with the intention to improve the estimation capability and to map different project categories. In COCOMO II three sub-models are introduced

- Application Composition Model
- Early Design Model
- Post-architecture Model

[11]Boehm uses Kilo Delivered Service Instruction with is identical to KLOC

II. Proposed Method

The three discussed methods are each taking the input size as a base for the estimation. For Use Case Point and for Function Point these are points that are weighted using empirical values from a table. The weighted points are then indirectly or directly translated into an effort estimation [34]. Constructive Cost Model (COCOMO) II uses Lines of Code (LOC) as input size. There is a distinction between SLOC (Single Line of Code) and KSLOC (Kilo Single Line of Code) for a stack of 1000 lines of code in the original COCOMO method.

The lines of code of a process model are hard to estimate and are dependent upon the modelling language or method. Furthermore, it is not possible to make a qualified statement about the lines of code of a process model at the situation where there is not process documentation available. The process has to be documented first. Therefore, it is necessary to employ other methods to derive the input size for the estimation model. We have identified the following input values and factors as a possibility to derive the input size (See II-A).

The early design model of COCOMO II is intended for the planning phase in software development and can therefore be transferred to the modelling phase of the BPM life-cycle III. As there is a high overlap in this association it is suitable for further investigation.

The post-architecture model is intended for the investigation of maintenance efforts and is applied after software development has started.

Both the early design model and the post-architecture model (hereinafter called COCOMO II level 2 and level 3) are based on the same mode of calculation. COCOMO II like COCOMO uses KLOC as input size. Differences are the naming of the exponential factor (formerly noted as n) as scale factors and the linear factor (formerly noted as m) as cost driver. Furthermore, Boehm[36] introduce a factor A that is based upon experience values and recommended being set to **2.94** as initial value. The formula to calculate the effort is described in (2).

Despite the early design model is considered most promising for further inquiry we will discuss the scale factors and cost drivers from the post-architecture model. Those scale factors and cost drivers are also included in the early design model. This enables us to give an outlook on possible factors for the maintenance cost. Costs for the implementation, execution, maintenance and optimization phase are not part of this work and are therefore not discussed.

A. Input Values

In process modelling it is not directly possible to estimate LOC beforehand. One possibility to estimate LOC is the enumeration of tasks using a method to be specified and derive LOC from this dependent upon a chosen modelling language. Business process modelling allows for graphical and for textual modelling, both result in source code. In the case of BPMN 2.0 and BPEL this source code is related to XML (See 1). In this source code every modelled element (task,

TABLE II
LINES OF CODE FOR BPMN OBJECTS

Object	XML-LOC
Activities	6
Artefacts	6
Messages	10
Events	8
Pools	7
Lanes	7
Average	**7 (rounded down)**

interaction, message flow etc.) is represented with a certain (minimum) number of lines of code. The number of lines of code per specific element is dependent upon the language and further factors like implementation (e.g. BPMN extensions) and tool support. For our proposed method there is derived LOC (See Table II) and points available as input values.

Listing 1 BPMN XML Description of an Activity (From [37])

```
<xsd:element name="task" type="tTask"/>
 <xsd:complexType name="tTask">
 <xsd:complexContent>
  <xsd:extension base="tActivity"/>
 </xsd:complexContent>
</xsd:complexType>
```

The simplest method to derive points is to enumerate all visual elements of the process modelling that will be created during modelling and summing them up. The multiplication of this sum with a factor that has to be defined would yield a derived amount of lines of code.

When modelling in BPMN 2.0 the following objects are mainly used [1] and need to be enumerated:

- Activities and Transactions
- Events
- Gateways
- Pools
- Lanes
- Data objects
- Incoming and outgoing communication

Activities and transactions, as well as involved participants (in the form of pools and lanes) can be estimated reasonably well beforehand but this is not true for the number of gateways needed. Heuristics can be considered as a basis to derive the number of gateways from a given number of activities and participants. We consider such heuristic models for future research.

B. Scale Factors

Besides the input value there are further factors to be considered: Not every process with the same amount of activities and events is equal. They can differ vastly in their complexity, process duration and the degree of parallelism. This influences the effort necessary to model these processes. More effort can be necessary for modelling more complex models as those models can be modelled semantically correct by the modeller but result in run time errors when executed [1]. Additional awareness during modelling is indicated in such cases as errors found during run time are more costly to correct than errors not introduced into the model in the first place. Complex process models warrant thorough testing. This is also an example of the so called soft factors like adeptness of the modeller. A skilled or adept modeller knows the limitations and pitfalls of the given modelling language and can produce valid and adequate models with less effort than an inept modeller.

Furthermore, we consider scale factors like team cohesion, precedentedness of the process to be modelled or the maturity of the modelling process.

Based on the four Ps of marketing [10] "Product, Price, Placement and Promotion" four categories of cost drivers for developing software do exist in software development [38]: Product, Personnel, Platform and Project. We provide the following example for these categories:

- Product: Cost driver here is the size of the process to be modelled
- Personnel: Modellers, analysts and programmers involved in the creation of a process model
- Platform: Time and size restrictions depending upon the BPMS or environment where the process models are to be deployed
- Project: Concurrent modelling at different locations

Those cost drivers are to be expected in business process modelling and we consider them in our model.

All three effort estimation methods described before (Function Point, Use Case Points, COCOMO II) are applicable (with adaptions) for the use during the modelling phase of the BPM life-cycle. These methods require - besides weighted input values - additional factors like quality, complexity and re-usability. Those factors can and should be stored during the application of the effort estimation method in order to increase the accuracy over time. Furthermore, it is possible with these methods to introduce new factors which is necessary for estimating BPM related efforts. The main disadvantage of adapting the Use Case Point method is the inexactness of the estimation and the lack of practical experience [34]. Due to the large interpretation range of what an actor is due to a clear definition of actors and the inexact mapping of transactions to activities during the execution of this method it is possible to receive skewed results[34]. As Use Case Points method is based on the Function Point Method for which more experience and literature is available we do not consider this method further in this work. Function Point Method is not suitable for a general effort estimation method for the modelling of processes as its factor list is not available for this domain as of the moment of writing. As we have not acquired real project / process data yet we issue the warning that results may vary and an inexactness is to be expected when applying it. Function Point also does not take the quality of the product to be developed into consideration [39]. We see a valid approach in combining Function Point method with COCOMO II for the derivation of LOC which are needed as input values for COCOMO II by using the resulting Unadjusted Function Points (UFP) from an intermediary step of Function Point. This combination is the key concept of this work.

C. BPM Function Point

Function Point derives the project's effort subject to its scope and difficulty[32]. The underlying method of Function Point is the determination of the so called Function Points by counting the input and output data, data objects, interfaces and queries and allocating points to them[34],[32]. Those five categories are functions of the software (to be implemented) and can be regarded as business transactions. For the estimation of a business process (model) this business transaction term is fitting. A process is also characterized by its input and output and whilst being executed a business process management system (BPMS) must store data and states associated with a process execution [1].

One activity in a process model correlates to one business transaction in the Function Point method. While counting the Function Points they are classified in one of three complexity classes ("Easy", "Average", "High"). Depending on the classification they are multiplied with weights ranging from 3 to 15. These factors need validation which we have not yet performed but are about to in a validation phase. As a baseline we choose the Function Point factors [36], [32]. Those factors have to be updated after a project is completed as the estimation effort increases in accuracy with completed projects and adjusted factors. The five steps for counting Function Points are applicable to our scenario as follows:

1) **Step 1**, Categorizing a requirement into one of the five groups.
2) **Step 2**, classify by "Easy", "Average", "High". At this stage only the complexity of separate modelling objects is to be classified and not the complexity of the process model as a whole. After completion of step 1 and 2 the result is a number of unadjusted Function Points (UFP) which can be combined with COCOMO II and be used as input variable when the determination of LOC is hard or impossible.
3) **Step 3** Adjusting the Function Points with scale factors. The list of factors is provided by the International Function Point Users Group (IFPUG [12]) for a variety of different programming languages. At the moment this work is performed no list for BPM or a modelling language is available. This is considered a restriction in using this method as of now. As we lay out real world data and exemplary projects are absent for this work we have not yet compiled such a factor list ourselves. We plan on composing a factor list in follow up work. Despite this restriction we describe the remaining steps to give a full overview and describe this method completely.
4) **Step 4** Calculation using the complexity adjustments described in Table III does fit into the BPM context. Other Methods like Mark II Function Point [40] expand the list of factors and adjust them. We propose an adjustment described in Tab.IV. In this step points are distributed according to the element and this is to be weighed/adjusted in total. The factors are associated with weighting factors and summarized, for the sum to reach a maximum of 60 points [34], [41].
5) **Step 5** calculates the adjusted Function Points from the previously collected unadjusted Function Points in order to extract the effort from the IBM effort table [42] in the last step. Another possibility is to derive a table with experienced values oneself and to update the original table which can be obtained from IFPUG after project completion. Updating and expanding this table with experienced values increases the accuracy of the effort estimation from project to project.

[12]http://www.ifpug.org

TABLE III
FUNCTION POINT CATEGORIES ADAPTED FROM [43]

Function-Point Category	Complexity of Components		
	Low	Average	High
External Inputs	3	4	6
External Inquiries	4	5	7
External Outputs	3	4	6
Internal logical files	7	10	15
External interface files	5	7	10

TABLE IV
FUNCTION POINT CATEGORIES MATCHING BPM FUNCTION POINTS

Function-Point Category	BPM Function Point	Complexity of Components		
		Low	Average	High
External Inputs	Inputs/Message	3	4	6
External Inquiries	Activities and Events	4	5	7
External Outputs	Outputs/Message	3	4	6
Internal logical files	Data-objects	7	10	15
External interface files	Pools and Lanes	5	7	10

TABLE V
FP ADAPTED FROM [41]

Function-Point Category	BPM Function Point
Quantity (see LOC)	Sum of Model Entities
Quality	Quality / Accuracy of Model
Complexity	Complexity of Model
Experience	Process / BPM Experience
Development Tools	modelling Tools
Programming Language	modelling Language

1) Summary of BPM Function Point: Due to the lack of appropriate conversion tables and missing project data it is currently not possible to implement steps three through five. We can not consider this method validated as of now due to these shortcomings. We plan on expanding research to fill this gap. Despite its shortcoming we consider this a promising approach to estimate the size of project models beforehand. Even without completing steps three to five, the user is enabled to collect or estimate unadjusted function points which can be used as input values for COCOMO II.

D. BPM COCOMO

Based on the fine granularity of the choice of factors and input variables for the estimation as well as the possibility to employ LOC or Function Points for input data COCOMO is the most promising method to be adapted as an effort

Fig. 2 BPM COCOMO method overview

estimation method for BPM. We present an adaption of COCOMO for the domain of BPM which we call BPM COCOMO. In this method the input data / size is derived analog to the Function Point method in order to have a virtual number of lines of code (LOC). The unadjusted Function Points as described before can be weighed better and more fine grained using COCOMO II. The extensive list of scale factors of the COCOMO II phase models "Early Design Model" and "Post-Architecture Model" (also named Stage 2 and Stage 3)[36] provide ample opportunity to adapt scale factors and cost drivers for BPM COCOMO. See figure 2 for an overview of the BPM COCOMO method. Starting with an idea or a process documentation of the process that is to be modelled one derives the number of entities to be modelled. Using this information a Business Model Point count is determined which then gets adjusted by scale factors and cost drivers.

1) Determination of Input Values: In order to state the LOC of the process model it is necessary for them to be derived first as they can not be estimated directly. For a direct classification it is necessary to enumerate the following objects

- activities
- events
- message flows
- control flows
- gateways
- data objects
- process participants

and then multiply those with the associated necessary LOC for their description (See Tab. II). This will yield the total LOC and is further used as an estimate [5],[6]. With this estimation it is not considered the fact that the modelling of the different object is of variable complexity. For example it is possible for an activity to have a multitude of control flows which would result in a higher number of LOC. Furthermore, not all objects are already known at the phase of modelling and therefore can not be taken into this estimation.

2) Derivation of Business Model Points (BMP): Table VI proposes an adaption of the complexity metrics for the domain of BPM. With this and the values from (Tab.IV) we derive the UFP. We count the objects and classify them accordingly to the complexity classes which then results in Business Model Points following the procedure from Function Point method.

3) Factor Adaption for BPM COCOMO: The effort in person months is calculated by (2):

$$PE = A \cdot Size^{E} \cdot \prod_{i=1}^{17} EMi \tag{2}$$

We keep the original value for A from Boehm[36] of 2.94. Due to the addition and omission of factors (see VII the values for SF and EF are changed. The values that are not changed and are only considered with a changed meaning are attributed with their original values. Additional Factors are added for the adaption to this domain.

The scale factor is calculated using equation (3):

$$E = B + 0.01 \cdot \sum_{i=1}^{6} SFi \tag{3}$$

Due to our changes the scale factor E lies now between 0.91 (unchanged to the original) and 1.312. The multiplication factor from the cost drivers is now between 0.0618 and 108.35 due to the changed and added factors. In contrast to the original COCOMO II we range from 0.0569 to 115.58 which is deemed plausible.

The description of BPM COCOMO is complete but lacking the explanation of the choosing and interpretation of adapted scale factors and cost drivers. We omit this because of the brevity of this paper.

III. Discussion

For this work no real world project data could be acquired. Proposals to companies for co-operations were made but came to no fruition. We think this is because of a variety of reasons: Enterprises do either have no such data available (process models and associated documentation), have no capacity for cooperation or regard such data as trade secrets. As no cooperation could be found an evaluation of this method has not yet been performed. We describe steps necessary for such an evaluation and plan to perform this in an academic setting. Referring to the determination of influencing factors and inputs in COCOMO II [38] we propose an empirical evaluation. Necessary for this are project data sets which include the proposed and actual effort. Projects from the following type should be considered for this evaluation.

- BPM introduction projects without existing documentation
- BPM introduction projects with existing documentation
- Business Process Re-engineering projects

We suggest research on the evaluation of:

1) Evaluation of the proposed method in general to see if it is valid and error free. If not then analyse if the core concept is suitable for adaption and improvement.
2) Evaluation of the selected scale factors.
3) Evaluation of the selected cost drivers.
4) Evaluation of the selected values for scale factors and cost drivers.

Dependent upon these results further influencing factors are possible to be determined which then can be added to the model.

A. Evaluation of the Proposed Method in General

For the evaluation of the proposed method in general [39], [43] evaluate the results of COCOMO II by employing

TABLE VI
OVERVIEW COMPLEXITY METRICS ADAPTED FOR BPM FROM [5]

Category	Input	Activities / Events	Outputs	Data Objects	Pools / Lanes
Easy	Easy to Conceive / In Sequence	Easy Flow	Easy to conceive / In Sequence	Easy application / Short Storage Duration	1-3 Lanes / Pools
Average	At different points in time	Average Flow / Large number of Joins / Large number of Tokens	At different points in time	Repeated access / Extended Storage Duration / Communication Intensive	3-10 Lanes / Pools
High	At different points in time, Dependencies	Complex Flow Complex Gateways	At different points in time / Dependencies	Generation of Events / Much Communication	$>$ 10 Lanes / Pools

TABLE VII
ADAPTION TABLE FOR BPM COCOMO WITH INPUT VALUES, SCALE FACTORS AND COST DRIVERS

	COCOMO II	BPM COCOMO	Abbreviation
Input Values	(K)LOC / Unadjusted Function Points	Business Model Points to LOC	
Scale Factors			
	Precedentedness	Precedentedness	PREC
	Development Flexibility	Development Flexibility	FLEX
	Architecture / Risk Resolution	Architecture / Risk Resolution	RESL
	Team Cohesion	Team Cohesion	TEAM
	Process Maturity (CMM)	BPM Process Maturity (BPMM)	PMAT
		Levels of Process Documentation	PDOC
Cost Drivers			
Product	Required Software Reliability	Process Correctness-Level	CORL
	Data Base Size	-	-
	Product Complexity	Process Model Complexity	CPLX
	Developed for Reusability	Level of Model Reuse	RUSE
	Documentation Match to Life-cycle Needs	Documentation Requirements	DOCU
		Process Confidentiality	CONF
Personnel	Analyst Capability	Analyst Capability	ACAP
	Programmer Capability	Modeller Capability	PCAP
	Personnel Continuity	Personnel Continuity	PCON
	Application Experience	BPM Experience	BPEX
	Platform Experience	Platform Experience	PLEX
	Language and Tool-set Experience	Language and Tool-set Experience	LTEX
Platform	Time Constraint	Time Constraint	TIME
	Storage Constraint	Storage Constraint	STOR
	Platform Volatility	Platform Volatility	PVOL
Project	Use of Software Tools	Use of Software Tools	TOOL
	Multisite Development	Multisite Development	SITE
	Required Development Schedule	Required Development Schedule	SCED

COCOMO-Abbreviation	**Scale Factor** Description	**Weighting** VL	L	N	H	VH	XH
PREC	Precedentedness Are there similarities to existing systems / models? Further factors: Precedetness of the process, description of project type	6,2	4,96	3,72	2,48	1,24	0
FLEX	Development Flexibility Further factors: Restriction due to modeling guidelines or best practises	5,07	4,05	3,04	2,03	1,01	0
RESL	Architecture/Risk Resolution Further factors: Quality of available process documentation	7,07	5,65	4,24	2,83	1,41	0
TEAM	Team Cohesion Further factors: Existances of a Chief Process Officer (CPO)	5,48	4,38	3,29	2,19	1,1	0
PMAT	BPM Process Maturity (BPMM)	7,8	6,24	4,68	3,12	1,56	0
PDOC	Levels of Process Documentation Further factors: Is a process documentation available? Is process documentation up to date? Is the process documentation suitable to derive input values from?	8,6	6,88	5,16	3,44	1,72	0

Fig. 3 BPM COCOMO Scale Factors (adapted from [34])

	EF_i	COCOMO-Abbreviation	Cost Driver	VL	L	N	H	VH	XH
Product	EF_1	CORL	Process Correctness-Level	0,82	0,92	1	1,1	1,26	-
	EF_2	CPLX	The complexity of the process model	0,73	0,87	1	1,17	1,34	1,74
	EF_3	RUSE	Levels of Model Reuse	-	-	1	1,07	1,15	1,24
	EF_4	DOCU	Documentation Requirements	-	-	1	1,1	1,23	-
	EF_5	CONF	Process Confidentiality	0,8	0,9	1	1,1	1,2	-
Personnel	EF_6	ACAP	Analyst Capability	1,42	1,19	1	0,85	0,71	-
	EF_7	PCAP	Modeller Capability	1,34	1,15	1	0,88	0,76	-
	EF_8	PCON	Personnel Continuity	1,29	1,12	1	0,9	0,81	-
	EF_9	BPEX	BPM Experience	1,22	1,1	1	0,88	0,81	-
	EF_{10}	PLEX	Platform Experience	1,19	1,09	1	0,91	0,85	-
	EF_{11}	TLEX	Language and Toolset Experience	1,2	1,09	1	0,91	0,84	-
Platform	EF_{12}	TIME	Time Constraint	-	-	1	1,11	1,29	1,63
	EF_{13}	STOR	Storage Constraint	-	-	1	1,05	1,17	1,46
	EF_{14}	PVOL	Platform Volatility	-	0,87	1	1,15	1,3	-
Project	EF_{15}	TOOL	Use of Software Tools	1,17	1,09	1	0,9	0,78	0,75
	EF_{16}	SITE	Multisite Development	1,22	1,09	1	0,93	0,86	0,8
	EF_{17}	SCED	Required Development Schedule	1,43	1,14	1	1	1	-

Fig. 4 BPM COCOMO Cost Drivers (adapted from [34])

performance metrics which can be applied to the evaluation of our method. They calculate a relative error (RE):

$$RE = (\text{estimated effort} - \text{actual effort})/\text{actual effort} \quad (4)$$

This yields a derivative of the estimated effort where small values mean small derivations from the estimation and validate the model.

B. Evaluation of Scale Factors

The selected scale factors are to be checked for their relevance for the estimation model and for plausibility. According to the results from the evaluation of the method in general scale factors are to be selected or deselected and their influence on the relative error should be observed.

C. Evaluation of Cost Drivers

Using the evaluation approach from scale factors it is to be observed what the selection or deselection of cost drivers changes in the relative error. Furthermore, the relevance of the cost drivers in the phase of modelling within the BPM life-cycle has to be checked, based on real world data. Evaluating the cost drivers for the modelling phase can lead to discovery of relevant cost drivers for the remaining phases implementation, execution and optimization.

D. Evaluation of Chosen Weights / Scale Factors

Besides the selection of relevant influencing factors the associated scale factor is of importance as deviations in both directions will effect large relative errors in the model. Wieschollek [44] states that modelling and process documentation are related 2:1 based upon his experience. This is neither scientifically quantified nor researched and can therefore neither confirmed nor denied. With large deviations of the estimations to the actual effort the method of counting Business Model Points (BMP) is to be re-evaluated and adapted.

IV. Conclusion

In this work the domain of business process modelling and associated domains are reviewed for methods and procedures to estimate cost or effort for process model creation. No method could be derived from the field of project or change management, as they do not estimate the effort directly but as a derivative from the total cost. The same holds true for the associated field of software engineering where [17] derives the cost from the total cost.

Especially in the software development domain no consideration of the modelling phase of the BPM life-cycle has occurred. There are authors that have researched effort estimation in business process management / business process modelling [13], [6], [14], [15] and [16] have examined agile and other widely adopted methods like Function Point or COCOMO. They base their effort estimation models on already existing models which only come into existing in later phases of the BPM life-cycle. Those effort estimation methods or models are to be placed in the technical level [1] or in the implementation phase of the BPM life-cycle or late. A clear distinction between functional and technical levels is blurred and those layers overflow [1]. This imprecise distinction can be regarded as a reason for the lack of proper effort estimations before.

A. Conclusion of Effort Estimation Methods

Out of the multitude of effort estimation methods from the domain of software development which would allow adaption for the use in the domain of BPM, COCOMO II is regarded the most adept one because it is well established and validated. Its fine grained description of factors makes it suitable for adaption. COCOMO has been continuously improved since its implementation in 1981. Individual adaptions of COCOMO like ADA COCOMO[45] or the counting of ObjectPoints [46] for object oriented software development prove that COCOMO is suitable for adaption. We could not identify an

adequate effort estimation method for the modelling phase of BPM which is why we adapt COCOMO in this work for this domain and create a proposed method we call BPM COCOMO. Furthermore, we introduce the effort necessary to create process documentation into the effort estimation method. Previously this effort was not considered. This process documentation effort is an additional scale factor that influences the effort of modelling a process model. Our proposed cost driver addition also influences the estimation. Our method provides the possibility to create effort estimations even without existing input values by deriving them using Business Model Points (BMP). BMP can be derived during the creation of process documentation. A statement of the quality of BPM COCOMO can only be made after the model has been verified and validated. Based upon the granularity of the model and the vastness of scale factors this model can be extended using empirical figures. Introducing and expanding empirical figures into the model it can be improved gradually in every execution step. Using this method academics and BPM professionals ([44]) are enabled to validate their expectations, experienced values and claims as well as improve and adapt their own methods.

REFERENCES

[1] J. Freund and B. Rücker, *Praxishandbuch BPMN 2.0*, 3rd ed. München: Hanser, 2012.
[2] J. Horan, Ed., *Schlüsselrolle CIO*, ser. CIO Studienreihe. IBM Institute for Business Value, 2011.
[3] S. Müller, "Studienarbeit 2451- situationsanalyse: Bpm in deutschland," Master's thesis, Universität Stuttgart, 2014.
[4] P. Posluschny, *Prozessmanagement*. Konstanz: UVK Verlagsgesellschaft mbH, 2012.
[5] V. Gruhn and R. Laue, "Komplexitätsmetriken für geschäftsprozessmodelle," in *Proceedings of the Modellierung 2006*, H. C. e. a. Mayr, Ed. Bonn: Gesellschaft für Informatik, 2006, pp. 289–292.
[6] K. Kluza and G. J. Nalepa, "Proposal of square metrics for measuring business process model complexity," in *Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on*. IEEE, 2012, pp. 919–922.
[7] F. Leymann, "Managing business processes via workflow technology," in *Tutorial at VLDB Conference*, Seattle, September 2011.
[8] B. Mutschler and M. Reichert, *Understanding the Costs of Business Process Management Technology*. Springer Berlin Heidelberg, 2013, pp. 157–194.
[9] A. Gadatsch, *Grundkurs Geschäftsprozess-Management*, 7th ed. Springer Vieweg, 2013.
[10] H. J. Schmelzer and W. Sesselmann, *Geschäftsprozessmanagement in der Praxis*, 6th ed. München: Hanser, 2008.
[11] M. Weske, *Business Process Management - Concepts, Languages, Architectures*, 2nd ed. Wiesbaden: Springer Berlin Heidelberg, 2012.
[12] T. Allweyer, *BPMS: Einführung in Business Process Management-Systeme*. BoD–Books on Demand, 2014.
[13] M. Baklizky, M. Fantinato, L. H. Thom, V. Sun, E. P. V. Prado, and P. Hung, "Business process points - a proposal to measure bpm projects," in *Proceedings of the 21st European Conference on Information Systems*. ECIS 2013 Completed Research. Paper 2., 2013. [Online]. Available: http://aisel.aisnet.org/ecis2013_cr/2
[14] B. Marin and J. Quinteros, "A cosmic measurment procedure for bpmn diagrams," *The 26th International Conference on Software Engineering and Knowledge Engineering*, 2014.
[15] S. Mishra and C. Kumar, "Estimating development size and effort of business process service-oriented architecture applications," in *Systems and Informatics (ICSAI), 2014 2nd International Conference on*. IEEE, 2014, pp. 1006–1011.
[16] E. Rolon, L. Sanchez, F. Garcia, F. Ruiz, M. Piattini, D. Caivano, and G. Visaggio, "Prediction models for bpmn usability and maintainability," in *Commerce and Enterprise Computing, 2009. CEC'09. IEEE Conference on*. IEEE, 2009, pp. 383–390.
[17] V. Nissen, M. Petsch, F. Termer, and M. Möhring, "A cost calculation model for determining the cost of business process modelling projects," *Ilmenauer Beiträge zur Wirtschaftsinformatik*, vol. 2013-01, April 2013.
[18] D. Çulha and A. Doğru, "Towards an agile methodology for business process development," in *S-BPM ONE-Scientific Research*. Springer, 2014, pp. 133–142.
[19] C. Thiemich and F. Puhlmann, *An Agile BPM Project Methodology*. Springer Berlin Heidelberg, 2013, vol. 8094, pp. 291–306. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40176-3_25
[20] T. Allweyer, "Das business process maturity model (bpmm) der omg," November 2007. [Online]. Available: http://www.kurze-prozesse.de/2007/11/08/das-business-process-maturity-model-bpmm-der-omg/
[21] D. M. Fisher, "The business process maturity model. a practical approach for identifying opportunities for optimization," *Business Process Trends*, vol. 9, no. 4, pp. 11–15, 2004.
[22] J. Cardoso, "Evaluating the process control-flow complexity measure," in *Web Services, 2005. ICWS 2005. Proceedings. 2005 IEEE International Conference on*. IEEE, 2005.
[23] S. Jakoubi and S. Tjoa, "A reference model for risk-aware business process management," in *Risks and Security of Internet and Systems (CRiSIS), 2009 Fourth International Conference on*. IEEE, 2009, pp. 82–89.
[24] H. Lhannaoui, M. I. Kabbaj, and Z. Bakkoury, "Towards an approach to improve business process models using risk management techniques," in *Intelligent Systems: Theories and Applications (SITA), 2013 8th International Conference on*. IEEE, 2013, pp. 1–8.
[25] R. Meziani and I. Saleh, "Towards a collaborative business process management methodology," in *Multimedia Computing and Systems (ICMCS), 2011 International Conference on*. IEEE, 2011, pp. 1–6.
[26] W. M. Van Der Aalst, A. H. Ter Hofstede, and M. Weske, "Business process management: A survey," in *Business Process Management*. Springer, 2003, pp. 1019–1019.
[27] A. Komus, *BPM Best Practice - Wie führende Unternehmen ihre Geschäftsprozesse managen*, 2011th ed. Berlin Heidelberg New York: Springer-Verlag, 2011.
[28] J. Mendling, H. A. Reijers, and W. M. van der Aalst, "Seven process modeling guidelines (7pmg)," *Information and Software Technology*, vol. 52, no. 2, pp. 127–136, 2010.
[29] M. Hinsch, *Die neue ISO 9001:2015 - Status, Neuerungen und Perspektiven* -, 1st ed. Berlin Heidelberg New York: Springer-Verlag, 2014.
[30] T. van Lessen, D. Lübke, and J. Nitzsche, *Geschäftsprozesse automatisieren mit BPEL*. Heidelberg: dpunkt Verlag, 2011. [Online]. Available: http://taval.de/publications/BOOK-2011-01
[31] C. Jones, *Estimating Software Costs : Bringing Realism to Estimating - Bringing Realism to Estimating*, 2nd ed. Madison: McGraw Hill Professional, 2007.
[32] T. Noth and M. Kretzschmar, *Aufwandschätzung von DV-Projekten: Darstellung u. Praxisvergleich d. wichtigsten Verfahren*, 2nd ed. Berlin: Springer, 1986.
[33] G. Karner, "Resource estimation for objectory projects," *Objective Systems SF AB*, vol. 17, 1993.
[34] S. Frohnhoff, "Use case points 3.0 : Implementierung einer use case bezogenen schätzmethode für das software-engineering betrieblicher informationssysteme," Ph.D. dissertation, Universität Paderborn, 2009.
[35] B. W. Boehm, *Software Engineering Economics*. New York: Prentice-Hall, 1981.
[36] B. Boehm and E. Harrowitz, *Software Cost Estimation with Cocomo II*. London: Prentice Hall, 2000.
[37] OMG, "Bpmn 2.0 specification." [Online]. Available: http://www.omg.org/spec/BPMN/2.0/
[38] B. W. Boehm, "Cocomo ii model definition manual," 2000. [Online]. Available: http://sunset.usc.edu/research/COCOMOII/Docs/modelman.pdf
[39] V. Khatibi, D. N. A. Jawawi, S. Z. M. Hashim, and E. Khatibi, "Neural networks for accurate estimation of software metrics," *IJACT: International Journal of Advancements in Computing Technology*, vol. 3, no. 10, pp. 54 – 66, 2011.
[40] C. Symons, "Function point analysis: difficulties and improvements," *Software Engineering, IEEE Transactions on*, vol. 14, no. 1, pp. 2–11, Jan 1988.
[41] H. W. Wieczorrek and P. Mertens, Eds., *Aufwandsschätzung in IT-Projekten*. Springer Berlin Heidelberg, 2007, pp. 205–223. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-48472-1_8
[42] IBM, *Die Function Point Methode: eine Schätzmethode für IS-Anwendungs-Projekte*, ser. IBM Form. IBM Deutschland

GmbH, 1985. [Online]. Available: https://books.google.de/books?id=dxzWPgAACAAJ
[43] V. Khatibi and D. N. A. Jawawi, "Software cost estimation methods: A review," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 4, no. 12, pp. 21–29, December 2011.
[44] M. Wieschollek, "Aufwandsschätzung für die prozessmodellierung." [Online]. Available: http://www.bpm-plus.de/2013/03/aufwandsschatzung-fur-die-prozessmodellierung/
[45] B. Boehm and W. Royce, "Ada cocomo and the ada process model," in *Proceedings. Third COCOMO Users Group Meeting, SEI*, 1987.
[46] B. W. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy, and R. Selby, "Cost models for future software life cycle processes: Cocomo 2.0," in *ANNALS OF SOFTWARE ENGINEERING*, 1995, pp. 57–94.