Measuring a Software Production Line with IFPUG-based Function Points

Volkan Halil Bagci, Umut Orcun Turgut, Ali Ciltik, Semih Cetin, Recep Ozcelik
Cybersoft Information Technologies R & D Center
Istanbul, Turkey
e-mail: {volkan.bagci, umut.turgut, ali.ciltik, semih.cetin,recep.ozcelik}@cs.com.tr

***Abstract* – Software Production Lines (SPLs) aim to manage cost-based activities for product delivery. Our company has been using SPL engineering for about 10 years and successfully implemented cost-controlled production cycles for SPLs during past two years, which are based on well-known Function Point (FP) approach supported by International Function Point User Group (IFPUG). Cost-based product delivery in SPLs requires the complete transformation of requirements gathering, cost estimation, time planning and productivity measuring steps. At the maturity level reached so far, every contributing part of the production line can be measured and cost-attached effectively and new targets can be set accordingly. Moreover, production bandwidth can be estimated precisely based on statistical productivity coefficients of every working team. This paper introduces our cost-controlled SPL approach, the achievements so far and our future plans for improvement.**

Keywords-Function Point; Software Measurement; Software Production Lines; Productivity Coefficient.

I. INTRODUCTION

Software is encountered in every part of our daily life nowadays. Consistent and cost-effective software products certainly make our life much easier. The consistency and cost-effectiveness of any product can be controlled by strict measurements and software is not an exception in that sense. In other words, software engineering is not an appropriate term unless the size, quality and productivity are measured accurately since unmeasured variables cannot be managed in any engineering discipline [1].

Software measurement enables the estimation of team productivity and improvement of existing processes based on recorded productivity metrics. Many researchers focus on new metrics to measure productivity [2] while others analyze software team productivity efforts and make empirical assessments for evaluating measurement efforts in software companies [3][4].

In particular to SPLs, the factors that accelerate and prevent team productivity can be statistically determined and exploited to the maximum extend for setting feasible targets. The approach explained in this paper has been used for the past two years in the banking SPL of our company and particularly implemented for a mid-scale bank in Turkey.

The rest of this paper is organized as follows: Section II discusses about the related works. Section III provides an overview of the organization and roles. Section IV describes the function point and its standard in the context of software evolution. Section V introduces the cost estimation process that is currently being held in our company. This section also describes the results obtained in the last period. Section VI discusses about the future work so as to improve the processes as a whole. Finally, Section VII concludes this paper.

II. RELATED WORKS

In the last decade many cost estimation models for software production lines have been proposed. Some representative proposals are: [5][6][7][8] and [9]. Poulin [5] presented a reuse metric and economics model that utilizes systematic reuse method. Poulin's model has two parameters: the relative cost of reuse (RCR) and the relative cost of writing for reuse (RCWR). Using these two parameters Poulin calculates the costs of product line development, thus provides extensive insight for the economics of software production lines.

Clements, McGregor, and Cohen [6] proposed the structured intuitive model for product line economics (SIMPLE) a general-purpose business model that supports the estimation of the costs and benefits in a product line development organization.

Lamine, Jilani, and Ghezala [7] proposed a new software cost estimation model for product line engineering that is based on integrated cost estimation model for reuse in general and Poulin's model of product line engineering. New tool supporting the model is described along with UML presentation.

Nóbrega, Almeida, and Meira [8] proposed integrated cost model for product line engineering (InCoME). As well as a new model is introduced along with its case study with results, the paper highlights important factors to acquire an effective model in terms of cost-benefit.

Nolan, and Abrahão [9] mentions about the experiences gained by using of estimation tools for the software product lines. It is clearly stated that a model is not only used for estimating cost and schedule but also for estimating and validating risks and opportunities. Future discussions about how a new cost model should be built are given for projects represented as number of Lines of Code (LOC).

 ISBN: 978-1-61208-367-4

III. MOTIVATION

Software Production Line is the adapted version of an industrial product line for developing software product families with the vision of managing cost and time-to-market concerns, which are based on structured reusability techniques [12]. The main supplier has its own SPL infrastructure so-called Aurora that is used for the production of different product families ranging from banking to insurance and tax administration to Enterprise Resource Planning (ERP) [10][11].

The customer bank decided to outsource the development and maintenance of its own banking software to the main supplier over its sister company the main contractor. In this setup, the main supplier is the main banking products supplier for many banks including the customer bank, and the main contractor company is the main contractor for customizing and maintaining the main supplier's banking products, particularly for the customer bank.

In order to provide high quality services to the customer bank, the main supplier and the main contractor decided to have a new unit called Product Management Department (PMD) in their joint organization chart. New organization chart including the PMD is given in Figure 1. PMD has a sub-unit so-called Product Improvement Group (PIG), which is responsible for inspecting and improving banking products (called product restructuring) using modern software engineering techniques as well as implementing corrective actions on existing modules (called product refactoring). Another sub-unit in PMD is Production Planning Group (PPG), which is responsible for cost-estimation of new inquiries, planning implementation tasks, and monitoring the production cycles. The contract between the customer bank and the main contractor is based on FP and inquiries are implemented with FP-based cost. FP-based cost anticipates the cost model based on software product functionality.

Pricing a single FP is not a trivial task in contractual terms since buyer and supplier do have different point of views. In case of the customer bank, a well-known international consulting group worked both with buyer and supplier teams to set the price for an FP, based on existing implementation costs and pricing models [13]. Working timesheets were examined, hourly and daily efforts were calculated and an average cost for an FP has been determined. Additionally, the FP-based cost estimation approach and related formula have been double-checked by the consulting group. The approach has been monitored for a while in real cases and finally approved both by the customer bank and the main contractor.

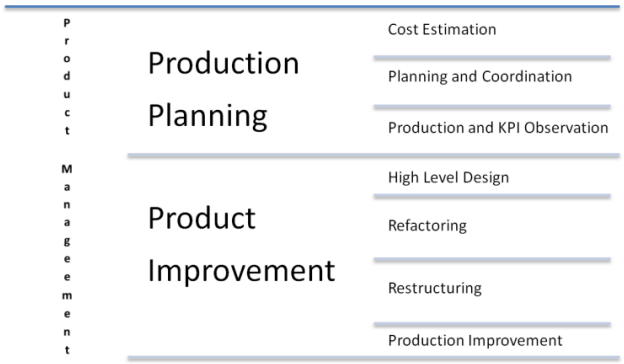

Figure 1. Organization schema of Product Management Department.

In this model, the customer bank Project Office (PO) only deals with the PPG as the single contact point of the main contractor and the main supplier. Once PO forwards inquiries, PPG prepares the Approximate Cost Form (ACF) for each inquiry, including the estimated starting and finishing dates of implementation. PO goes through each ACF and approves accordingly. The approval of ACF initiates the real planning of each inquiry with exact dates of implementation. PPG is also responsible for allocating necessary resources for the software development efforts. During the course of implementing every inquiry, PPG keeps certain Key Performance Indicators (KPI) to measure the effectiveness of every conveyor in the software production line. Using these SPL KPIs, PPG is expected to coordinate software development teams, business analysts, and test units throughout the lifecycle of a request.

IV. FUNCTION POINT

FP is a metric for measuring the functionality provided to the user of an information system. The concept was introduced by Albrecht in 1979 [14], and used widespread in the world as of today in a variety of 6 different standards, such as COSMIC FSM, FiSMA FSM, IFPUG FSM, MK II FPA, NESMA, and the automatic FP supported by Object Management Group (OMG) [15][16][17][18][19][20]. The OMG automatic FP standard is based on IFPUG approach in such a way that it determines functions, differentiates internal and external files, and calculates the FP accordingly.

IFPUG initiated the standardization of measuring software projects, which is accepted by the International Standards Organization (ISO) with most up-to-date version 4.3. As stated in IFPUG Counting Practices Manual (CPM) 4.3, FP is the unit of measurement to express the amount of business functionality [21]. IFPUG FP is calculated based on counting the factors, including internal and external information sources, external inputs, outputs, and queries. We particularly prefer to use IFPUG FP within other FP approaches as being the most widely used approach, being in line with banking domain, providing access to an extensive database of more than 5000 International Software Benchmarking Standards Group (ISBSG) project performance cases, having large volume of industrial data in management information systems, and enabling the official certification option [23][24].

 ISBN: 978-1-61208-367-4

V. COST ESTIMATION PROCESS

In this section of the study, cost estimation process is explained in detail while post process observations and outcomes are shared in later parts of the section.

Works that are being performed by the main contractor are handled via requests. Each request has a request type that might have impact on cost estimation as given in detail in Section A.

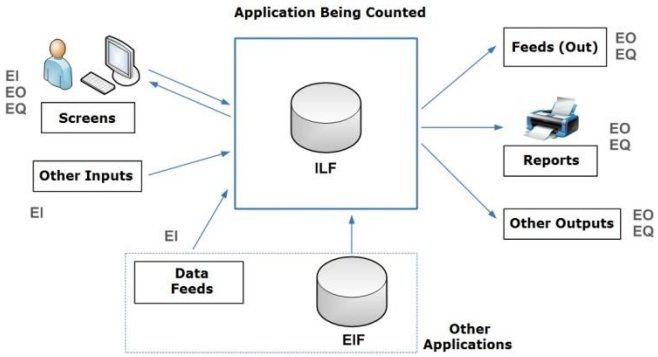

Figure 2. The view of a software application from the eyes of an FP practitioner [22].

Our FP practitioners examine requests to identify data and transaction functions using IFPUG FSM with similar view of a software application as shown in Figure 2. Once the initial examination of request is complete, project type and request size is decided as explained in section B and section C respectively. FP practitioners estimate cost of the request by using the process factor and calculation method as explained in section D.

After cost estimation is complete, planning and product phase starts as explained in Section E. Observations and importance of scope meeting are mentioned in Sections F and G respectively.

A. Request Type

Prior to our cost estimation process implemented, when a request is entered in the system, it is given a request type based on the expected application size and application type. In order to support these request types, general system characteristics (GSC) [21] are decided for these request types, making cost estimation balanced for a given type of request. Thus, there are three request types given in our system; which are project, improvement, and report.

Project and improvement types are both software applications that might involve brand new functionality and/or modifications over existing application. Main difference is the size of the application; for example, an estimated cost threshold of 62 FP or less is being used as improvement request type within our process. Any request that has estimated cost size of 62 FP is of project request type.

TABLE I. CALCULATED VAF VALUES FOR REQUEST TYPES.

Request Type	Total TDI Points	VAF (∑TDI * 0,01) + 0,65
Project	35	1.00
Improvement	35	1.00
Report	0	0.65

In our cost estimation process, request types can affect variable adjustment factor (VAF) as shown in (2), thus have impact on final cost estimation. VAF for project and improvement request types are set to 35 Total Degree of Influence Points (TDI), making VAF of these request types equal to 1.0.

$$VAF = (\sum TDI * 0.01) + 0.65 \text{ [21]} \quad (1)$$

Report is a special request type that addresses information retrieval using offline databases via quick third party development tools. VAF of report project type is calculated as 0.65 once all TDIs of the GSC are set to 0 due to the simple development efforts required for reports.

B. Request Requirements Category

Each request is represented by one or many requirements. These requirements can be identified as functional or non-functional ones. In our cost estimation process, while IFPUG FSM is used for functional requirements in terms of cost estimating, estimating cost of non-functional requirements handled using our non-functional point system. In order to cover a cost estimation process that would address requests with different possibility of requirement types, a request requirements category (RRC) is introduced as an element of decision node in our cost-estimation flow-chart, which is shown in Figure 3.

Based on the possible combination of the request requirement varieties, there are three RRCs as follows.

1) Functional RRC: Functional RRC addresses requirements that include only functional ones. Thus the cost of the request can be calculated according to IFPUG FSM v.4.3 standard. Whether the request has a functional component or not can be identified by examining the requirements of the request. If it has at least one function among Internal Logical File (ILF), External Interface File (EIF), External Input (EI), External Output (EO) or External Inquiry (EQ), then request may be processed as a functional RRC. Examples of functional projects are listed below.

- Data Migration (Customer data entrance, sending control signal)
- Data Transformation (Bank interest calculation, average temperature derivation)
- Data Storage (Customer order record, environment temperature record)

 ISBN: 978-1-61208-367-4

- Data Query (Listing current personnel, querying coordinate data)

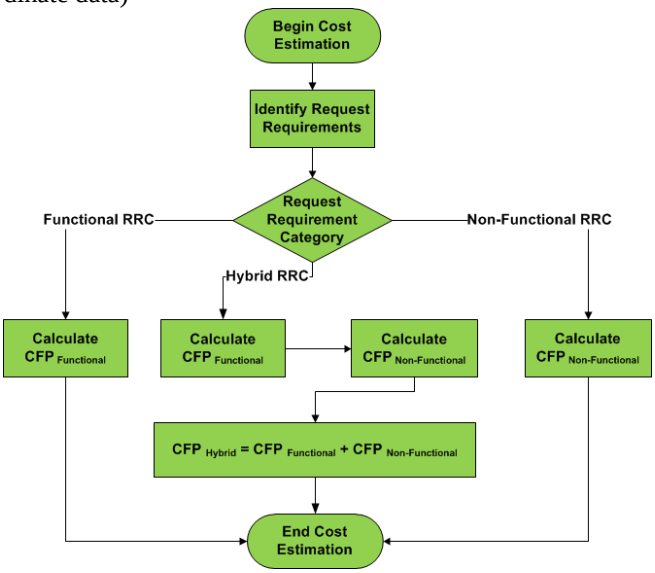

Figure 3. The cost estimation flow-chart.

2) Non-functional RRC: If the request does not contain a functional requirement, then the cost cannot be calculated using IFPUG FSM. The total cost of the request is calculated using non-functional point system by summing all of the separate FP costs of items, which are listed below in detail.

- Project Management, Coordination, Requirement Gathering
- Analysis, High Level Design, Quality Control
- Design, Software Development, Integration
- Functional Tests, Acceptance Tests, Technical Support Services
- Deployment, Fixed-Works

The costs of these types of requests are reckoned according to man/month data and used as NFP (Non-Function Point) in the system for setting connection with FP.

a) Fixed-Works: In order to decrease the operational cost of recurring non-functional requests, fixed-works list has been set up. Fixed-works list is a living document. Not only the FP practitioners, but also the planning experts and module managers do the relevant updates as NFP on that list.

3) Hybrid RRC: According to the standard of IFPUG FP calculation v.4.3, if cost price of a request can be executed, although it has non-functional requirements, this type of requests are called hybrid requests. The cost of these types of requests is calculated by summing the costs of both the functional and non-functional components.

C. Request Size

Requests that have a size below a certain threshold are classified as minor requests while the ones that are above the threshold are classified as major requests. Requests go through different states as shown in Table IV. Encompassing the period from request entry to the deployment, several output documents are created along these steps.

The aim of the request size classification is to have an efficient production line. As it can be seen in Figure 4 and 5, based on the request size, requests follow different path. With a few exceptions, minor requests usually get involved in a minor process pipeline, without passing through the analysis and design steps; thus, most of the documentation requirement is dropped off. On the other hand, major requests have to follow the big route, which is passing through quality processes and as a result, analysis and design documents are prepared in detail.

Current threshold in terms of FP is arranged to be just more than a single function, thus meaning if a request has more than a single function involved, it would be addressed as a major request. Based on IFPUG CPM 4.3 [21], minimum possible single function cost is 3 FPs; for example EI-Low and EQ-Low both have 3 FPs. Therefore, in agreement with the bank, it is decided to use 3 FPs as a threshold for request size classification.

D. Process Factor and Cost Calculation

Process Factor (PF) is the sum of total coefficients of all sub processes in the production line. It is used for reflecting the costs of all sub-processes to the total cost in minor and major requests. Besides distributing the total cost to the sub-processes, PF also calculates the partial cost when the job, which is being carried on the product line, is canceled. Maximum value for the PF can be 1.0. Table II details the PF values for some of the sub-processes and these are calculated according to their portions in the production period.

Equation (2) shows how the process factor is calculated.

$$PF = A + HLD + D + QC + DEV + AT + BT \qquad (2)$$

where A is the analysis process factor, HLD is the high level design process factor, D is the design process factor, QC is the quality control process factor, DEV is the development process factor, AT is the alpha test process factor and BT is the beta test process factor.

TABLE II. PROCESS FACTOR VALUES FOR EACH REQUEST TYPE.

Request Size / Request Type	A	HLD	D	QC	DEV	AT	BT	PF
Major Project	0.25	0.05	0.10	0.05	0.40	0.10	0.05	1.00
Major Improvement	0.25	0.05	0.10	0.05	0.40	0.10	0.05	1.00
Minor Improvement	0.00	0.05	0.00	0.02	0.40	0.10	0.05	0.62
Major Report	0.25	0.05	0.10	0.05	0.40	0.10	0.05	1.00
Minor Report	0.25	0.05	0.00	0.02	0.40	0.10	0.05	0.87

 ISBN: 978-1-61208-367-4

$$aFP = VAF * uFP \quad (3)$$

$$CFP = PF * aFP \quad (4)$$

As per definition given in [17] and shown in (3), adjusted function point (aFP) is calculated using VAF and unadjusted function point (uFP). Value of VAF can change based on the project type. As it can be seen from (4), PF has a direct consequence on the cost estimation. In (4), CFP is the cost in FPs, PF is the process factor and aFP is the regulated FP.

TABLE III. SAMPLE NET COSTS.

Request Size / Request Type	uFP	VAF Based on Request Type	aFP (VAF * uFP)	PF Based on Request Size	CFP (PF * aFP)
Minor Improvement	3.00	1.00	3.00	0.62	1.86
Major Improvement	32.00	1.00	32.00	1.00	32.00
Major Project	100.00	1.00	100.00	1.00	100.00
Minor Report	3.00	0.65	1.95	0.87	1.70
Major Report	6.00	0.65	3.90	1.00	3.90

With reference to the Table III, net CFP values for sample applications with distinct request types and request sizes are calculated according to distinct process factors. In the calculations, the threshold value is set to 3 FP.

E. Planning and Production Tracking

After estimating the cost of each request, planning experts decide on the deadline of the request, taking account of the characteristic of the request, source and integration status. Planned development time and the number of developers that are going to be assigned to the request are calculated according to the basic Constructive Cost Model (COCOMO) equations given in (7) and (8) [20]. In order to use these equations, reference values for planned development time and number of developers are calculated via (6) instead of (5). For this reason, instead of using code line of count parameter and COCOMO coefficients in (6), calculated effort value of product line is used and classical COCOMO equation is adapted to the (6) for our system. Since (5) is not being used directly, it does not have an effect on our productivity rates. c_b and d_b values are decided according to Boehm's semi-detached software project standards as stated in (7).

$$E = a_b (KLOC)^{b_b}, \; a_b = 3{,}0, \; b_b = 1{,}12 \quad [22] \quad (5)$$

$$E = \frac{CFP}{DM} \quad (6)$$

$$D = c_b E^{d_b}, \; c_b = 2{,}5, \; d_b = 0{,}35 \quad [22] \quad (7)$$

$$P = \frac{E}{D} \quad [22] \quad (8)$$

where E is the effort applied (person-months), KLOC is the estimated number of delivered lines of code for the project, a_b, b_b, c_b and d_b are COCOMO coefficients, CFP is the cost in FP, D is the development time in months, DM is the average work day count in a month (20 work days) and P is the count of required people.

TABLE IV. PRODUCTION LINE STATUS OF CORE BANKING UNIT.

	Production Line (FP)																				Amount
States	BKS		ÇEK		KYS		MEV		MUH		TAHSİS		TEM		TOTAL FP						#
01 Demand Admission	-	(0)	-	(0)	-	(0)	-	(0)	-	(0)	-	(0)	-	(0)	-	FP					8
02 Approximate Cost	-	(0)	-	(0)	-	(0)	-	(0)	-	(0)	-	(0)	-	(0)	-	FP					12
03 Approximate Cost Approval	0	(0)	0	(0)	297	(2)	0	(0)	0	(0)	0	(0)	0	(0)	297	FP					2
04 Planning	0	(0)	0	(0)	4	(2)	0	(0)	0	(0)	0	(0)	11	(1)	15	FP					3
05 Analysis	130	(4)	18	(3)	12	(4)	63	(5)	47	(1)	65	(3)	94	(6)	429	FP					26
06 High Level Design	0	(0)	0	(0)	24	(3)	0	(0)	0	(0)	19	(1)	0	(0)	43	FP					4
07 Quality Control 1	0	(0)	0	(0)	0	(0)	0	(0)	0	(0)	0	(0)	0	(0)	0	FP					0
08 Analysis Approval	6	(1)	0	(0)	115	(4)	0	(0)	15	(2)	114	(4)	121	(7)	371	FP					18
09 Design	4	(1)	40	(5)	7	(1)	83	(4)	0	(0)	0	(0)	48	(5)	182	FP					16
10 Final Cost	1	(1)	26	(1)	6	(1)	0	(0)	0	(0)	0	(0)	0	(0)	33	FP					3
11 Final Cost Approval	0	(0)	0	(0)	0	(0)	114	(1)	0	(0)	0	(0)	0	(0)	114	FP					1
12 Software Development	57	(4)	10	(3)	123	(3)	104	(9)	0	(0)	30	(3)	49	(8)	373	FP					30
13 Alpha Test	61	(2)	1	(1)	51	(4)	88	(6)	12	(2)	17	(1)	6	(1)	236	FP					17
14 Quality Control 2	0	(0)	0	(0)	7	(1)	23	(3)	0	(0)	0	(0)	0	(0)	30	FP					4
15 Quality Control	0	(0)	0	(0)	1	(1)	0	(0)	0	(0)	0	(0)	1	(1)	2	FP					2
16 Acceptance Test	20	(2)	4	(1)	49	(4)	52	(5)	20	(2)	61	(4)	10	(2)	216	FP					20
17 Acceptance Test Failed	0	(0)	0	(0)	0	(0)	0	(0)	0	(0)	0	(0)	0	(0)	0	FP					0
18 PO Control	0	(0)	10	(2)	0	(0)	12	(2)	0	(0)	0	(0)	1	(1)	23	FP					5
19 Deployment	218	(23)	87	(6)	76	(8)	285	(30)	6	(4)	88	(12)	93	(16)	853	FP					99
20 Production	0	(0)	0	(0)	1	(1)	0	(0)	0	(0)	0	(0)	0	(0)	1	FP					1
21 Completed	73	(20)	328	(61)	130	(30)	534	(83)	18	(11)	77	(17)	345	(64)	1.505	FP					286
22 Waiting	0	(0)	38	(1)	33	(1)	139	(3)	0	(0)	0	(0)	0	(0)	210	FP					5
23 Closed	4	(2)	0	(1)	18	(6)	0	(0)	2	(1)	0	(1)	34	(5)	58	FP					16

In order to obtain production line status data as show in Table IV, costs of requests are distributed among the request states. The production line status data enables us to track the current intensity of work load on each group and also to foresee the upcoming intensity of work load status of each group as well. By monitoring the product line data as shown in Table V, planned and completed work follow-up can be carried out. Using the statistical data gathered, resource planning and productivity performance analysis for each software module & team can be successfully accomplished. By taking goals and productivity coefficients into account, pre-detection actions for restructuring the problematic software modules can be put into practice in the future.

F. Cost and Planning Process Observations

In order to count functional size of any request, functional requirements are needed. In the beginning of the transition phase, it was hard to complete the cost estimation process because of lacking required information regarding the request requirement specifications. Therefore, to determine functional and non-functional requirements for estimating approximate costs for requests, meetings with the participation of module owners and FP practitioners are being held.

After calculating approximate costs of the requests, the requests are planned by putting them on the production line using available resources. Then as shown in Figure 4 and 5, analysis, high level design, and design steps are performed before the requests reach the final cost estimation step. On this step, final cost is reckoned using the analysis and design documents. Once the final cost estimation is complete and approved by PO, software development efforts may begin using the available resources.

 ISBN: 978-1-61208-367-4

TABLE V. THE MAIN CONTRACTOR'S PRODUCTION LINE PLANNING AND PRODUCTION TRACKING.

PLANNED		2012	January	February	March	April	May	June	July	Total
TB-II	Core Banking II	1.168	347	751	447	674	476	625	0	4.487,33
TB-I	Core Banking I	961	464	532	991	636	775	630	0	4.988,80
BA	Banking Infrastructure	209	252	390	613	513	409	353	0	2.738,66
DU	Support Systems	423	260	278	277	236	135	216	0	1.825,49
İZKD	Business Intelligence Support System	352	186	261	388	138	788	32	0	2.143,88
TB-III	Core Banking III	427	26	86	540	250	49	42	0	1.419,49
ÜY	Product Management	933	0	0	0	37	0	0	0	969,80
VY	Data Management	0	290	72	0	13	0	0	0	375,63
İZMİR	İzmir	10	0	0	0	220	7	0	0	236,06
TOTAL		4484	1824	2369	3256	2716	2638	1897	0	19.185,13

COMPLETED		2012	January	February	March	April	May	June	July	Total	Goal	%
TB-II	Core Banking II	113	346	592	375	524	141	100	0	2.189,24	7.100,00	30,83%
TB-I	Core Banking I	531	403	482	394	321	230	28	0	2.388,66	7.050,00	33,88%
BA	Banking Infrastructure	163	252	371	286	371	156	76	0	1.676,09	3.910,00	42,87%
DU	Support Systems	142	247	199	175	70	49	3	0	883,99	2.415,00	36,60%
İZKD	Business Intelligence Support System	34	159	217	110	116	95	6	0	736,89	1.945,00	37,89%
TB-III	Core Banking III	18	26	18	131	31	7	4	0	235,48	1.850,00	12,73%
ÜY	Product Management	0	0	0	0	0	0	0	0	0,00	909,00	0,00%
VY	Data Management	0	0	0	0	2	0	0	0	2,52	180,00	1,40%
İZMİR	İzmir	4	0	0	0	44	3	0	0	51,73	150,00	34,49%
TOTAL		1006	1432	1879	1472	1479	681	216	0	8.164,60	25.509,00	32,01%

As can be seen in Table V, in the new cost system, FP calculations in the transition period are less than the other periods. Total FP for January 2013 is 1824 and more requests are being loaded to the production line in the following months. The reasons behind the weak performance in the transition period are technical problems, personnel resistance fed from old habits and efforts spent for obtaining functional requirements. In the succeeding step of our process enhancement, one to one negotiations for functional requirement inference, which is examined in the transition period, are left and a new document, namely preliminary requirement analysis document, is organized with the contributions of business analysts and module owners in order to calculate true approximate cost.

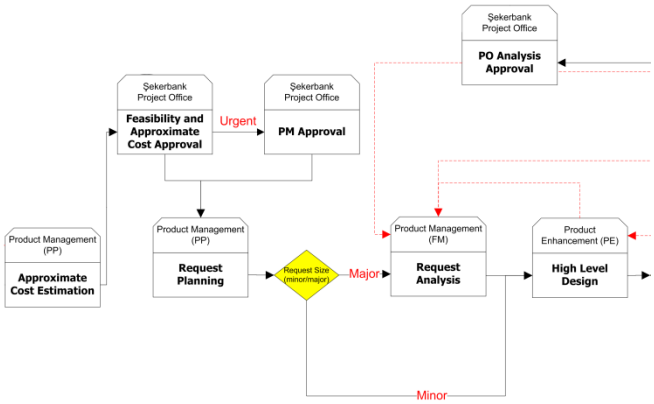

Figure 4. Software Production Process Pipeline Part 1.

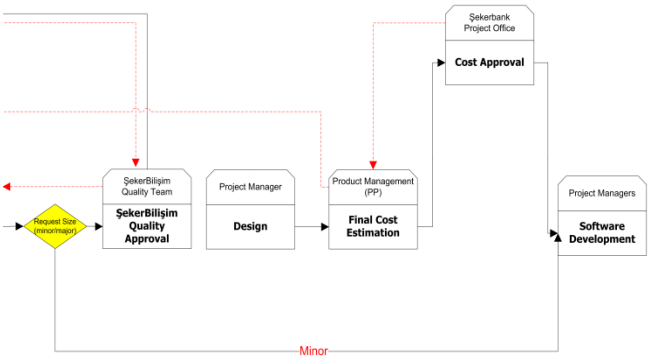

Figure 5. Software Production Process Pipeline Part 2.

G. Scoping Meetings

One of the ongoing improvement efforts is improving the efficiency of scoping meetings, which are performed at an intense pace. In order to perform more effective and more conscious monthly and quarterly plans, comprehensive requirement gathering activities are conducted for creation of a request pool, which consists of requests that have initial cost estimations. Project office, business unit, business analysts, module owners and production planning experts are participating in these activities. Utilizing the outcomes of the scoping meetings, due to assessing the situation of the production line from a wider perspective, long-term business targets will be identified and prioritized.

1) Observations: Difficulties in requirement gathering activities, especially requests that require integration of different modules, are noted and it is anticipated to cause inconveniences for accurate cost estimation efforts. However, in order to increase the efficiency of the software development efforts, we desire to minimize the participation of the relevant module's software engineers in requirement gathering activities. However, considering the lack of technical background of the business analysts at the moment, software engineers are still important assets for the scope meetings.

VI. FUTURE WORK

Because of historical reasons, software engineers have led scoping and requirements gathering activities. This role actually belongs to business analysts and we need to train them to increase their competence in business architecture. Accordingly, business analysts will contribute more in the scoping and requirements gathering meetings, so this affects efficiency of the Software Production Line, since software engineers will involve less in these meetings and activities, and focus only software development phase. Moreover, once problematic modules will be identified by observing productivity ratios, Product Improvement team will conduct necessary restructuring and re-factoring activities.

When it comes to cost estimation process, it is under continuous quality control, which let us fine-tune of productivity calculations. Establishing Software Product Line will be much easier after measuring all metrics of software production line, which is the next goal of the company.

VII. CONCLUSION

The main contractor and the main supplier have been capable of measuring several metrics related to the software production line via IFPUG functional size measurement

method. In the process of adaptation to the new system, a resistance fed from old habits is faced and new steps have been added to the production process to achieve the desired effect in requirement gathering activities.

Within the scope of adaptation of function points to the production line, arrangements are made on existing request types, request sizes, and project types. In order to estimate total cost for different request types with different request sizes on various project types, process factor is defined and is used as shown in the Table III. Simple COCOMO equations are adapted for FP and statistical data of the product line, hereby, are gathered. In consequence of available data, the condition of the production line can show the actual and planned works along with the accumulated workload on the business units. Making use of these indicators, production and resource planning can be made more efficiently and factors adversely affecting the process can be observed.

Scoping meetings are made in requirement of detailed information to accurately estimate cost of a request and new methods for the solution are actively being searched. Annual software development goals can be determined by productivity calculations that are based on FP for each team. By utilizing productivity factors, modules that have low productivity performance are identified. Once the identification process is complete, the identified modules are targeted for restructuring purposes to improve development productivity.

REFERENCES

[1] C. Jones, Applied software measurement: global analysis of productivity and quality. McGraw-Hill, 2008, ISBN 978-0-07-150244-3.

[2] M. Solla, A. Patel, C. Wills, "New metric for measuring programmer productivity," Proc. IEEE Symp. Computers and Informatics, IEEE, 2011, pp. 177-182.

[3] N. Ramasubbu , M. Cataldo , R. K. Balan , J. D. Herbsleb, "Configuring global software teams: a multi-company analysis of project productivity, quality, and profits," Proceedings of the 33rd International Conference on Software Engineering, USA, May 21-28, 2011, pp. 261-270.

[4] O. T. Pusatli, S. Misra, "Software Measurement Activities in Small and Medium Enterprises: an Empirical Assessment," Acta Polytechnica Hungarica, 8 (5) , 2011, pp. 21-42.

[5] J. S. Poulin, "The Economics of Software Product Lines," International Journal of Applied Software Technology 3, 1997, pp. 20–34.

[6] P. Clements, J. McGregor, S. Cohen, The Structured Intuitive Model for Product Line Economics (SIMPLE) (CMU/SEI-2005-TR-003). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005.

[7] S. B. Lamine, L. L. Jilani, H. H. B. Ghezala, "A Software Cost Estimation Model for a Product Line Engineering Approach: Supporting tool and UML Modeling," 3rd ACIS Conf. on Software Engineering Research, Management and Applications, 2005, pp. 383-390.

[8] J. P. Nóbrega, E. S. Almeida, S. R. L. Meira, "InCoME: Integrated Cost Model for Product Line Engineering," Proceedings of the 2008 34th Euromicro Conference Software Engineering and Advanced Applications, 2008, pp. 27-34.

[9] A. J. Nolan, S. Abrahão, "Dealing with Cost Estimation in Software Product Lines: Experiences and Future Directions," SPLC'10 Proceedings of the 14th international conference on Software product lines, 2010, pp. 121-135.

[10] N. I. Altintas, M. Surav, O. Keskin, and S. Cetin, "Aurora software product line," 2nd National Software Engineering Conference, Ankara, Turkey, Sep. 2005.

[11] N. I. Altintas, S. Cetin, A. H. Dogru, and H. Oguztuzun, "Modeling Product Line Software Assets Using Domain-Specific Kits," IEEE transactions on software engineering, vol. 38, Dec. 2012, pp. 1376-1402.

[12] D. E. Nye, America's Assembly Line. MIT Press, 2013, ISBN 978-0262018715.

[13] P. R. Hill, Practical software project estimation: a toolkit for estimating software development effort and duration. McGraw-Hill, 2010, ISBN 978-0-07-171791-5.

[14] A. J. Albrecht, "Measuring application development productivity," IBM Application Development Symposium, OCT. 1979, pp. 83-92.

[15] ISO/IEC 19761, 2003 COSMIC Method Measurement Manual v. 3.0.1.

[16] ISO/IEC 29881, 2008 Information technology Software and systems engineering FiSMA 1.1 functional size measurement method.

[17] ISO/IEC 20926, 2009 Software Engineering - IFPUG 4.3.1. Unadjusted FSM Method - Counting Practices Manual.

[18] ISO/IEC 20968, 2002 Software Engineering - Mk II Function Point Analysis - Counting Practices Manual.

[19] ISO/IEC 24570, 2005 Software Engineering - NESMA Functional Size Measurement Method v.2.1 - Definitions and counting guidelines for the application of Function Point Analysis.

[20] Automated Function Points (AFP) Version 1.0 - Beta 1, ptc/2013-02-01.

[21] The international function point users group: function point counting practices manual release 4.3.1., 2010, ISBN 978-0-9753783-4-2.

[22] Borland Conference, How to determine your application size using function points. [Online]. Available from http://conferences.embarcadero.com/article/32094.

[23] P. Morris, "Mapping the rules for IFPUG and COSMIC-FFP function size method," IFPUG Fall Conference, Scottsdale, Arizona, USA, 2003, pp. 299-319.

[24] Common Software Measurement International Consortium, A Comparison of the Key Differences between the IFPUG and COSMIC Functional Size Measurement Methods. [Online]. Available from: http://www.cosmicon.com/portal/public/IFPUG_COSMIC_Key_Comparison.pdf.

[25] B. W. Boehm, Software engineering economics. Englewood Cliffs, NJ:Prentice-Hall, 1981, ISBN 0-13-822122-7.

 ISBN: 978-1-61208-367-4