Practical Approach to Manage the Optimum Size of a Software Maintenance Team

Mircea F. Marian Prodan and Adriana M. Petruta Prodan
University Politehnica Bucharest, Entrepreneurship, Business Engineering and Management Doctoral School, Romania
Email: {pro100mir, burduja_adriana}@yahoo.com

***Abstract*—Software maintenance has been and remains an important topic for all IT companies. All applications, no matter if they are based on Cloud, Mobile, Social, SAP or any other technology, they are all going through this stage. Therefore is essential for managers to ensure a low cost of the service, while also ensuring a high quality. A major contributor to the overall cost of maintenance is the cost with personnel. In many situations, the size of the team delivering the service is not properly determined, and as a result, the size of the team is either too big generating high costs, or is too small, the team not being able to cope with all requests ending with a long time to answer a request.**

***Index Terms*—software maintenance, team, optimum size, profitable, cost efficient, size estimation, model**

I. Introduction

Software maintenance has been and will long remain an important topic for IT companies. According with IEEE definition [1], software maintenance is "the modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment."

It is well known that the majority of the cost on software lifecycle is in the maintenance stage. [2]-[3].

Big IT companies like IBM, Accenture, Infosys, SAP, are offering such services to other big Fortune 500 companies.

One of the questions that every manager needs to answer is how big the size of the team should be. If the size of the team is too big, than the service probably will not be profitable since the costs will be high. On the other hand, if the size of the team is too small, the team will struggle with the volume of request and service provided will not by satisfactory for the clients. This problem is not new. It is the topic of various papers. Function Points, code size, tasks, have been used to estimate software maintenance and size of the team [4],-[6], just to mention some of the studies. Models like SLIM [5], COCOMO II [7], and various improvements to them are being used on large scale.

These models are used to estimate software maintenance teams before starting such service.

The more complex the software, the bigger number of non-conformities. In the same time, the number of non-conformities reported is proportionate to the number of users [8]. This indicates that models based solely on Function Points or size of code may not be accurate. Initial estimations need to be validated once the service started and reached a stable level.

Practical observations working as consultants in this field have conducted us to the conclusion that, most of the times, managers tend to not assess their team capacity to deliver the service and compare with the demand. In addition to this, whenever there are challenges in terms of not being able to deliver on time or increased backlog of tickets, the managers choose to increase size of the team and therefore also the cost of delivery. In this way, they fail to achieve a higher profitability of their accounts.

This approach is helpful for Service Delivery Managers to validate the size of their teams and make appropriate decisions to enhance profitability of the maintenance service.

II. Process

Maintenance process has the following steps:

- User raise a request
- Request is routed to the developer
- Developer analyze the request
- Solution identification
- Implementation of the solution
- Test solution
- Deliver solution
- Inform user

Requests are raised in a ticketing management application (TMA) and all related information are stored, like details, status, changes, dates, etc.

III. Concept

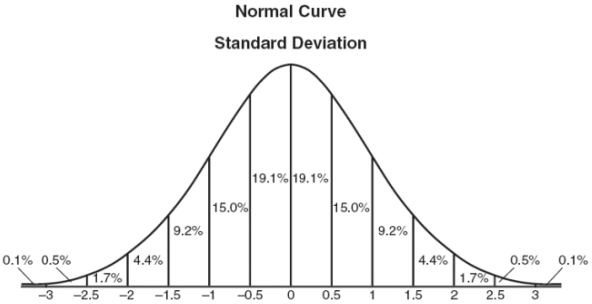

Figure 1. Normal distribution

Manuscript received July 13, 2014; revised September 17, 2014.

©2016 Engineering and Technology Publishing

doi: 10.12720/joams.4.1.30-34

Variation in a process is natural, and most of the times follows a normal distribution, values have the tendency to be centered on an average value (Fig. 1). Statistical methods have been used as early as beginning of the 20th century in manufacturing to manage processes. [9]

In Fig. 2 is shown a graph with variation of request from a maintenance process (with blue, named also demand). On the other side, variation of capacity to deliver doesn't have such variation (Fig. 2). This is defined as total time that team can dedicate to answer to various requests received. The difference between actual size of the team (red line) and optimum size (dotted line) can show an extra capacity or missing capacity.

Capacity do deliver is calculated based on the size of the team. Compared with the required capacity (demand) there will be cases where demand is higher than actual capacity, resulting in requests which will not be solved.

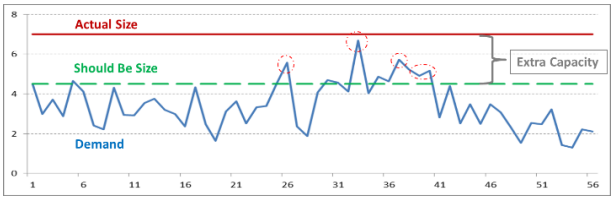

Figure 2. Concept of required capacity vs. team size

We have defined *backlog* as requests which cannot be solved in the same day and which will be solved in the following days, volume of unsolved requests will be added to the incoming volume in the next day.

On the other hand, there will be cases where actual capacity is higher than demand. The difference represents time which is used by the team members for trainings, other activities or just to relax.

Many managers are not performing studies to see if their team size is best to cope with demand and in the same time to ensure a high profitability.

IV. Study

The team under study was composed by 25 developers, covering 68 different applications. The period we conducted the study was one full year, from July 1st, 2012 to June 30th, 2013.

We didn't include in our study major enhancements, which are most of the times treated as software development projects.

To reach the optimum size of the team, we have followed next steps:

- Capture and analyze volume of incoming requests
- Understand effort required to solve requests
- Define full time equivalent (FTE)
- Understand demand and variation
- Identify optimum size

A. Capture and Analyze Volume

We have analyzed the incoming volume of requests for one year (July 2012 – June 2013). In average, there were 1890 tickets per month. Daily average volume was 81 tickets. Fig. 3 and Fig. 4 are shown the monthly variation and daily variation.

No pattern has been identified by looking at the monthly variation.

Looking at daily variation, we have observed:

- 2 peaks – one on Nov and one in Feb.
- Volume seems to drop significantly during May and June, as a result of Easter and summer holidays
- Low volume in the last week of the year due to Christmas holidays

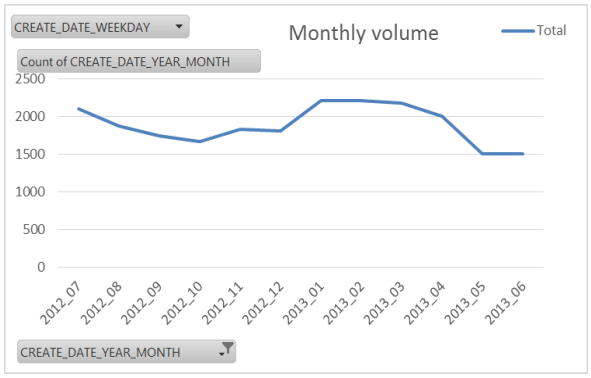

Figure 3. Monthly variation of demand

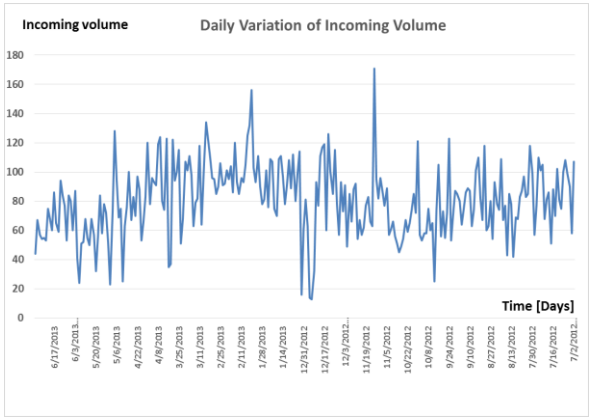

Figure 4. Daily variation of volume of tickets

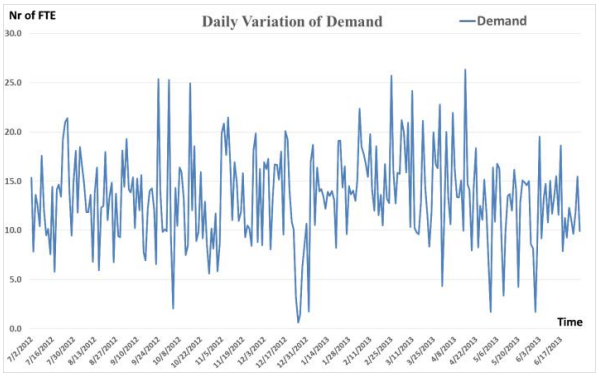

Figure 5. Daily variation of Capacity required to cope with Demand

B. Effort Required to Solve the Requests

Tracking the effort required to solve a request is not an easy task. Together with the team, we decided to track the effort as a range, rather than a value. The intervals are presented in Table II. This approach is not new, a similar approach has been used by Briand and Basili [10] to estimate software maintenance.

©2016 Engineering and Technology Publishing

TABLE I. EFFORT INTERVAL

Effort interval
0 - 15 min
15 - 60 min
60 - 120 min
120 -240 min
240 - 480 min

Furthermore, to approximate the effort required to solve each request, we have used a random function within the given range, simulating natural variation.

C. Identify Capacity

Another important part was to determine the capacity of the team to deliver. To calculate it, we started from the FTE definition given in the Business Dictionary [11]:
"The ratio of the total number of paid hours during a period (part time, full time, contracted) by the number of working hours in that period Mondays through Fridays."

The team was composed by 25 people, fully dedicated to this process. In total 25 FTEs. In the period 1 July 2012 – 30 June 2013 there are 260 working days, 21 vacation days for each team member, and 9 Public holidays. This mean that the total days allocated to perform work was 230 days, 1840 hours.

Not all team members have the same skills. Based on team member's observations, we took the following assumption:

One junior team member can deliver same service in double the time of a medium developer and a senior developer with 50% faster than a medium one (Table II).

TABLE II. SKILLS DISTRIBUTION INSIDE THE TEAM

Skills	Level of Productivity	Nr of team members
Junior	50%	10
Medium	100%	11
Senior	150%	4

Based on team member's skills, we identified an average level of productivity of 88% (weighted average).

We looked at the breaks each team member need to take. It is well known that a human cannot be productive if he is working without breaks. Most of employers allow their employees to have an unpaid lunch break of 30-60 minutes. On addition, smaller paid breaks are allowed. For example, in UK, according with law [12] an employer needs to allow for one 20 min break. In Canada [13], the Ministry of Labour recommends for computer work 5 minute break for every hour worked. We have used the Canadian recommendation and we allowed for 5 min breaks for every hour worked.

We calculated the team capacity using the following formula:

$$C = P \times \frac{DW}{W} \times (wh - br) \times AvP \qquad (1)$$

where:

C = Average individual available capacity dedicated for this process
P = Number of people in the team fully
DW = Number of days in a year dedicated to work (excluding holidays, medical leaves, trainings, etc)
W = Number of working days in a year (260 days)
wh = Working hours in a shift
br = Total Breaks time
AvP = average level of productivity of the team

In our case:

$$C = 25 \times \frac{230}{260} \times \left(8 - \frac{40}{60}\right) \times 88\% = 5.7\ hours \qquad (2)$$

Interpretation of the result: In average, for any full time developer in our team, in any given work day, he will be productive 5.7 hours. Team overall capacity is found by multiplying the individual average available capacity with the number of people in the team.

D. Understand Demand–Volume and Capacity

In Fig. 5 is shown the daily required capacity to cope with demand. Team size is 25 FTEs.

We have performed a hypothesis test to check that daily incoming volume is normal distributed. Our hypothesis was:

H0: data follow a normal distribution
H1: data do not follow a normal distribution

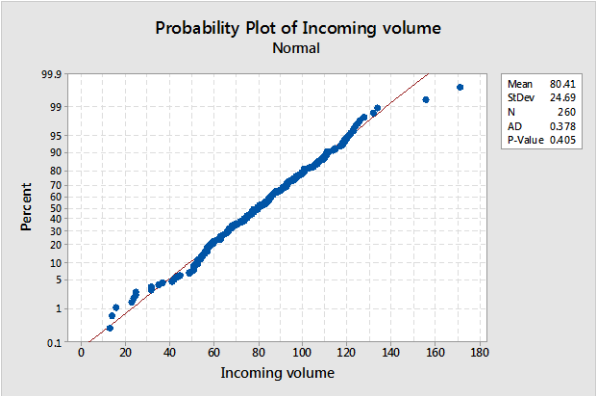

Figure 6. Test for normality

The results have ensured that data is normally distributed (P-value>0.05). Comparing the incoming volume (Fig. 3) and demand required (Fig. 4) we can easily observe that the peaks in volume doesn't translate into a peak if required demand. We explained this by the fact there were many easy request raised during the peaks.

We conducted a normality test for capacity and the results ensured that it is normally distributed, as well.

We have used IM-R chart to identify potential special causes (Fig. 7):

- Low volume of request during Christmas holidays
- 5 points very close to upper control limit (UCL), with one point over the limit
- 2 special cases where 4 out of 5 points more than 1 standard deviation from center line (test 6)
- 1 case where 9 points in a row on same side of center line (test 2)

We also realized that size of the team was very close to the upper control limit.

©2016 Engineering and Technology Publishing

E. Identify Optimum Size

To understand the optimum size of the team, we simulated the process with teams of various sizes, ranging from 25 (actual size) to 13 (average capacity required is 13.29).

We captured the following indicators which will allow us to decide on the optimum size of the team:

- NDDHC-Number of days in which required capacity is higher than team capacity. This will translate in accumulating backlog.
- NDDLC-Number of days in which required capacity is smaller than team capacity. No backlog will be accumulated during these days
- NDBS-Number of days in which backlog is solved in the next day. This means that if in day N will be a backlog of X, in the day N+1 the team will be able to answer a volume of requests equal with Volume in day N+1 +X
- APFC-Average percentage of free capacity. Free capacity is unused capacity, which usually the team members are using for trainings, documentation and other activities not related to solving requests
- AEC-Average extra capacity which would be required to solve the backlog (for the days with backlog)

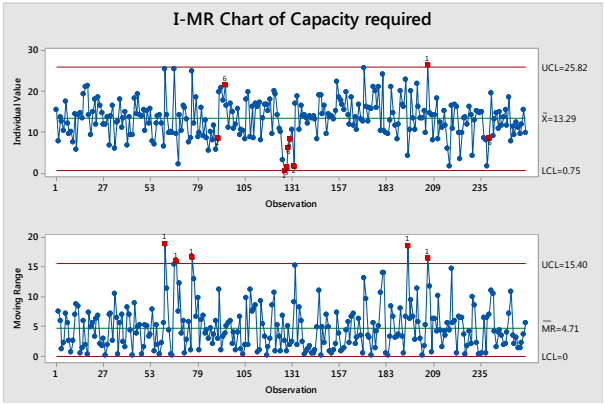

Figure 7. IM-R chart for capacity required

V. Results

The results are shown in Table III. Based on the results of the study, optimum size of the team is 21 FTEs. In these case, in 14 days the required capacity will be higher than the actual capacity of the team, however in 13 of the cases the team will be able to recover the backlog in the next day; required capacity to handle the backlog is 2.31 FTEs. Compared with the initial size of the team, this means an economy of 20% of the cost with personnel, just by performing this study and with no intervention on the process. This extra capacity will be used to perform more activities for the client and generate more revenue.

Even more economy could be achieved, however this is conditioned on having skilled and knowledgeable resources available for the days with higher demand. For example, with a size of 18 FTEs, in 17 days additional 3 knowledgeable resources will be needed.

TABLE III. Results of the Simulation

Team size [FTE]	NDDHC	NDDLC	NDBS	APFC	AEC
25	4	256	4	47%	0.75
24	6	254	6	45%	1.2
23	6	254	6	42%	2.3
22	8	252	8	40%	2.64
21	14	246	13	37%	2.31
20	18	242	13	34%	2.4
19	29	231	17	30%	2.47
18	40	220	23	26%	2.88
17	45	215	20	22%	3.74
16	68	192	28	17%	3.84
15	88	172	35	11%	4.35
14	115	145	40	5%	4.76
13	142	118	31	-2%	5.2

In Fig. 8 is shown daily variation of required capacity compared with optimum size of the team.

Empirical team size is similar with the size obtain using the following formula:

$$S = ArC \times 1.5 \times \sigma \quad (3)$$

where:

S = empirical size of the team

ArC = average required capacity (demand)

σ = standard deviation for daily demand

In our case S obtained in this way is 20.3 FTEs. According with normal distribution, the probability to have higher values of demand than S is 6.7% (1 day out of 15).

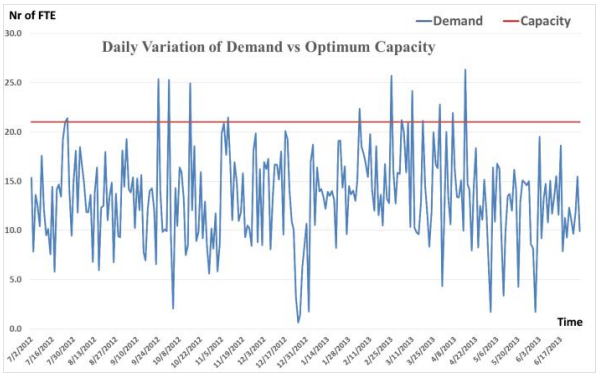

Figure 8. Demand vs optimum capacity

VI. Conclusions

There are several limitations to our study:

- Study has been performed on one team. Similar studies on other teams need to be performed in order to validate the approach.
- Another limitation is the data – incoming volume must follow a normal distribution.
- For simplicity, we assumed that capacity remains the same, while in reality things like medical leaves, attrition, group vacations, participation in training, are generating variation of team capacity.
- The model can be used after having history data which need to be analyzed

©2016 Engineering and Technology Publishing

- Simplistic approach of productivity of the developers

Nevertheless, this approach is simple, fast, can be useful to managers to make educated decisions regarding their team' size.

In the same time, we want to signal to maintenance managers that they have to know the optimum size for their particular service and they have now a practical tool to help them.

One missing piece in a TMA is tracking the effort. Almost all TMAs are able to offer information like turn around time, but very few have the possibility to track the effort. Attributed to W.E. Deming with or without real basis, "in God we trust, the rest bring data" emphasize on the principle that in order to improve something, you need to measure it. To optimize the effort, you need to measure it.

ACKNOWLEDGMENT

Research has been conducted with financial support from "Sectorial Operational Programme Human Resources Development 2007-2013," contract number POSDRU/159/1.5/S/132395 and 132397.

REFERENCES

[1] IEEE Computer Society, *Guide to the Software Engineering Body of Knowledge*, 2004 Ed. USA, California: IEEE Press, 2004, ch. 6, pp. 89.
[2] S. Schach, *Object-Oriented and Classical Software Engineering*, 8th Ed. McGraw-Hill, 2010, ch. 1.3, pp. 6-14.
[3] R. D. Banker and S. M. Datar, "Factors affecting software maintenance productivity: An exploratory study," in *Proc. 8th International Conference on Information Systems*, Pittsburgh, 1987, pp. 160-175.
[4] V. Nguyen, "Improved size and effort estimation models for software maintenance," PhD. dissertation, Faculty of the USC Graduate School University of Southern California, CA, 2010.
[5] L. Putnam and W. Myers, *Measures for Excellence: Reliable Software on Time, within Budget*, Prentice Hall PTR, 1991, pp. 234.
[6] H. M. Sneed, "Estimating the costs of software maintenance tasks," in *Proc. IEEE International Conference on Software Maintenance*, Opio (Nice), France, October 17-20, 1995, pp. 168-181.
[7] B. W. Boehm, E. Horowitz, R. Madachy, D. Reifer, B. K. Clark, B. Steece, A. W. Brown, S. Chulani, and C. Abts, *Software Cost Estimation with COCOMO II*, Prentice Hall, 2000, ch. 1.
[8] G. J. Holzmann, "Economics of software verification," in *Proc. PASTE '01 the 2001 ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering*, New York, ACM, June 1st, 2001, pp. 80-89.
[9] W. A. Shewhart, *Economic Control of Quality of Manufactured Product*, New York, D. Van Nostrand Company, Inc., 1931, pp. 7.
[10] L. C. Briand and V. R. Basili, "A classification procedure for an effective management of changes during the software maintenance process," in *Proc. Conference on Software Maintenance*, FL. Orlando, 1992, pp. 328-336.
[11] Business Dictionary. (June 2014). Full time equivalent definition. [Online]. Available: http://www.businessdictionary.com/ definition/full-time-equivalent- FTE.html
[12] UK government website. (June 2014). Rest breaks at work. Government Digital Service. [Online]. Available: https://www.gov.uk/rest-breaks-work
[13] (May 2005). Professional and Specialised Services of the Occupational Health and Safety Branch Ministry of Labour. Ontario, Canada. Health and Safety Guideline. [Online]. Available: http://www.labour.gov.on.ca/english/hs/pubs/gl_restbreaks.php

Mircea F. Marian Prodan has graduated from University Politehnica Bucharest, Faculty of Energetics, Romania, and has a Master degree in Sustainable Development from the same university. Currently is a PhD. student at University Politehnica Bucharest, Entrepreneurship, Business Engineering and Management Doctoral School, Romania.
He has 8 years of work experience as Lean Six Sigma consultant, working in various industries like manufacturing, banking, BPO, cadaster and IT. Currently leading Lean program in Europe for a big IT company. As a researcher at University Politehnica Bucharest, he has published a number of papers, "Case study of improving software applications maintenance service using Lean Six Sigma methodology," "Key success factors for a process improvement program in an organization," "Productivity in software maintenance multicultural teams." Mr. Prodan is a senior member of American Society for Quality, certified Six Sigma Black Belt and PMP certified by PMI.

Adriana M. Petruta Prodan has graduated from University Polytechnic Bucharest, Faculty of Energetics, Romania, and has a Master degree in Project Management at Bucharest University of Economic Studies-Economics Cybernetics, Statistics and Informatics. Currently is a PhD student at University Polytechnic Bucharest, Entrepreneurship, Business Engineering and Management Doctoral School, Romania. She has 5 years' work experience as Project Manager, working in IT industry. Currently managing an IT medium sized project. As a researcher at University Polytechnic Bucharest, she has published a number of papers: "Productivity in software maintenance multicultural teams," "Managing multicultural project teams," "Implementing small& medium IT projects in small& medium Enterprises". Mrs. Prodan is PMP certified by PMI.

©2016 Engineering and Technology Publishing