BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **20**, No 2
Sofia • 2020 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2020-0018

DevOps Project Management Tools for Sprint Planning, Estimation and Execution Maturity

Jayasri Angara[1], *Srinivas Prasad*[2], *Gutta Sridevi*[3]

[1]*Koneru Lakshmaiah Education Foundation, Vijayawada, AP, India*
[2]*GITAM, Visakhapatnam, AP, India*
[3]*Malla Reddy Institute of Technology, Secunderabad, TS, India*
E-mails: angara.jayasri@gmail.com srinivas_prasad@hotmail.com sridevi.gutta2012@gmail.com

***Abstract*: *The goal of DevOps is to cut down the project timelines, increase the productivity, and manage rapid development-deployment cycles without impacting business and quality. It requires efficient sprint management. The objective of this paper is to develop different sprint level project management tools for quick project level Go/No-Go decision making* (*using real-time projects data and machine learning*)*, sprint estimation technique* (*gamified-consensus based*)*, statistical understanding of overall project management maturity, project sentiment & perception. An attempt is made to device a model to calibrate the perception or the tone of a project culture using sentiment analysis.*

***Keywords*: *DevOps, Machine Learning* (*ML*)*, effort estimation, planning poker, sentimental analysis.*

1. Introduction

DevOps is a cross-disciplinary methodology emerged out of continuous development-test-deployment cycles. The goal of DevOps is to deliver the project quickly and with high reliability. It brings multiple tools, practices, and philosophies and change in the project execution culture. DevOps encourages to fail fast which in turns help in recovering fast [1]. The prime challenge for DevOps-Agile Project Manager is to manage continuous planning, continuous integration, continuous deployment and continuous monitoring processes between multiple sprints. Project timelines are short. It requires greater collaboration between multiple stakeholders [2]. In this setting, the Project Manager should able to manage projects without cost/effort overrun.

This paper proposes a predictability algorithm for quick Go/No-Go decision using Naive Bayes Classifier (machine learning probabilistic classifier). This algorithm can be executed at any stage of project execution, which helps to measure the project management maturity at any given point. However, it is ideal before the sprint begins.

Project Manager should able to make quick and right estimation by taking inputs from team members. Project Manager should also consider their previous judgment track record. We developed an effort estimation model using planning poker and scientific calculation of participant weights.

During the execution of the project, effective and real-time communication plays a major role. Proactively Project Manager should gauge the sentiment of project communication. Ineffective communication and negative sentiment in the project leads to huge risk for every dollar spent on a project [3]. Tight deadlines, overlapping functions between development teams, QA and operations teams, rapidly changing business priorities and customer expectations are few triggers for negative tone. Project tone has a profound impact on project success. A sentiment analysis based model is developed to appreciate the tone of the project sentiment/culture. This model helps to understand the tone of collaboration between various communication channels like project blog/chat messages, customer emails, audit or management reports, customer meetings, daily scrum/stand-up meetings, etc. We used illustrative datasets to develop this model.

This paper is structured as follows. Section 2 discusses the literature survey. Section 3 presents the methodology and results for the Go/No-Go decision, effort estimation model, statistical understanding of overall project management maturity and a model to measure the tone of the project culture. Section 4 presents threats to validity. Section 5 presents the conclusion.

2. Literature survey

The objective of the literature survey is to understand how the sprint cycle works, various tools needed for project manager during sprint phases to make the project success.

Sprint cycle goes through four stages: Sprint Planning, Sprint Execution, Sprint Review, and Sprint Retrospective. Sprint Planning deals with decisions related to planning the work for the subsequent sprint. Sprint Execution covers tactical implementation and development decisions. Sprint Review covers the decisions related to product/customer satisfaction, continuity of future sprints, etc. Sprint Retrospective covers the decisions related to the improvement of the sprint process in future sprints [4].

The success of a project in DevOps-Agile context depends on how well execution is managed and monitored on various factors. Many surveys and studies necessitate the need to develop the predictability algorithm for quick Go/No-Go decision in the context of agile and DevOps. We need to a develop feasibility assessment model based on the following parameters which will help the project manager during sprint planning. Among many factors, developer competency, developer seniority, and task dependency are the most critical factors for project success [5]. Many projects fail due to unskilled resources, lack of process understanding, not having customer experience, lack of cohesion among the team members, etc. The other parameters include Customer value generation [6], communication and collaboration between stakeholders [3, 7], positive customer perception, operations team confidence, organizational behavior, etc. DevOps is a socio-economic system. It solves more social issues than technical issues [8-10]. Unstable and unorganized requirements lead to project failures. The project should be able to manage such repulsive conditions [11]. Project success depends on funding availability and how well the requirements are phased. The right selection of automation tools (based on functionality/technical requirement) is key to project success [12, 13]. Domain understanding [13-15], usage of regression testing/smoke testing/load/ performance testing/ data-driven testing [16-18], application criticality [19], application stability, project management, and process maturity, etc. are few other parameters influencing the project success. It is important to have detailed planning before implementing any software project [20-22]. Hence, assessing the project health before sprint beginning is always crucial. Project health assessment is a three-level process: Screening-Level (Goals, Strategy, Organization SWOT Analysis), Readiness Level (resources, process, budget, etc.) and Complexity Level (Technical, Domain, Scope, and Dependencies) [23-25]. We need to consider all these levels in our assessment.

Repeated use of the Go/No-Go decision during the project execution gives a broad picture of project management maturity. Developer/Test/Operations team/ Customer communication and sentiment are levers to understand project mood and maturity [26]. Developing such tools is always useful to run the project smoothly [27]. DevOps-Agile cannot adopt traditional estimation methodology because requirements are not clear and frozen. It demands a highly intuitive-rationale-agreeable estimation methodology. There is a need for scientific, real-time and ground-level effort estimation technique during sprint planning [28]. An effective trust established through social communication between developers, testers, operations, and customer teams is a clear predictor for successful collaboration in the distributed project environment [29].

This literature survey stresses the need for Go/No-Go decision tool for the project manager during the planning stage [30]. The project manager should also be able to gauge how the communication is happening between various stakeholders like customers, team members, senior management, etc. It requires technology to help to calibrate sentiment of the project. Machine learning techniques come handy to develop such a sentiment analysis tool. During the planning stage, the project manager should be able to estimate accurately with wide consensus and also able to gauge his/her project management maturity. There should be some tools to handle these needs.

3. Methodology and results

Sprint is a basic iteration of an agile methodology. Sprint is a time-bound period with specific objectives. The goal of the sprint is to have flawless deliverables. Generally, sprints are of short timeframes, which may vary between one and three weeks. During sprint phases, project managers need special tools to avoid any delivery deviations. In this section, we present a few project management tools that can be used during respective sprint phases as shown in the Table 1 [4].

Table 1. Sprint stage vs. proposed project management tool

Sprint stage	Proposed project management tool
Sprint planning	Project feasibility decision algorithm
Sprint estimation	Effort estimation model using e-Planning poker and scientific calculation of participant weightages algorithm
Sprint review	Statistical view of project management maturity
Sprint retrospective	Project sentiment view

3.1. Project feasibility decision algorithm

Managers should ensure smooth execution of the project delivery while taking up the test automation. In this context, the manager should be able to check quick feasibility (Go/No-Go decision) to avoid cost and effort overruns. There should be a decision tool to facilitate the latter. The objective of this algorithm is to predict the feasibility of a (Go/No-Go) decision for automation of test of the project software. To develop this algorithm, we surveyed 36 test automation projects and created a dataset consisting of 21 independent factors (as identified as part of the literature survey) and their project outcomes (Success or Not). These values represent the perception of the project teams. Each factor value is categorized into High, Medium, or Low. Out of 36 records, 26 records (72%) are used as a training set and 10 records (28%) are used as testing set/prediction set. We have used Naive Bayes Classifier. This algorithm is preferred for text classification setting and when we have various classes. Naive Bayes Classifier is a classification algorithm built based on Bayes' Theorem. It assumes that every pair of features being classified is independent of each other. Another assumption is each feature makes an independent and equal contribution to the outcome. Bayes' Theorem is stated as the following equation: $P(A|B)=P(B|A)P(A)/P(B)$ where $P(A|B)$. Conditional Probability of occurrence of event A given the event B is true, $P(A)$ and $P(B)$: Probabilities of the occurrence of event A and B, respectively, $P(B|A)$: Probability of the occurrence of event B given the event A is true.

To generate these results, we used XLSTAT software to generate these tables when we input 36 records data. To fine-tune our classifier more robust with qualitative variables when classifying new observations, we applied a Laplace Smoothing value to 1. Once applied, the following tables are generated by the tool itself. The summary of statistics is presented in Tables 2-5. Table 2 presents the observed frequency distributions. Table 3 and Table 4 present the confusion matrices. Table 5 presents the classification of the prediction set.

Table 2. Training set dependent variable

Variable	Categories	Frequencies	%
Project success or not	No	11	42.3
	Yes	15	57.6

Table 3. Confusion matrix

Confusion matrix	Yes	No
Yes	15	0
No	0	11

Table 4. Confusion matrix (Leave-one-out error)

Confusion matrix	Yes	No
No	10	1
Yes	3	12

Table 5. Posterior probabilities

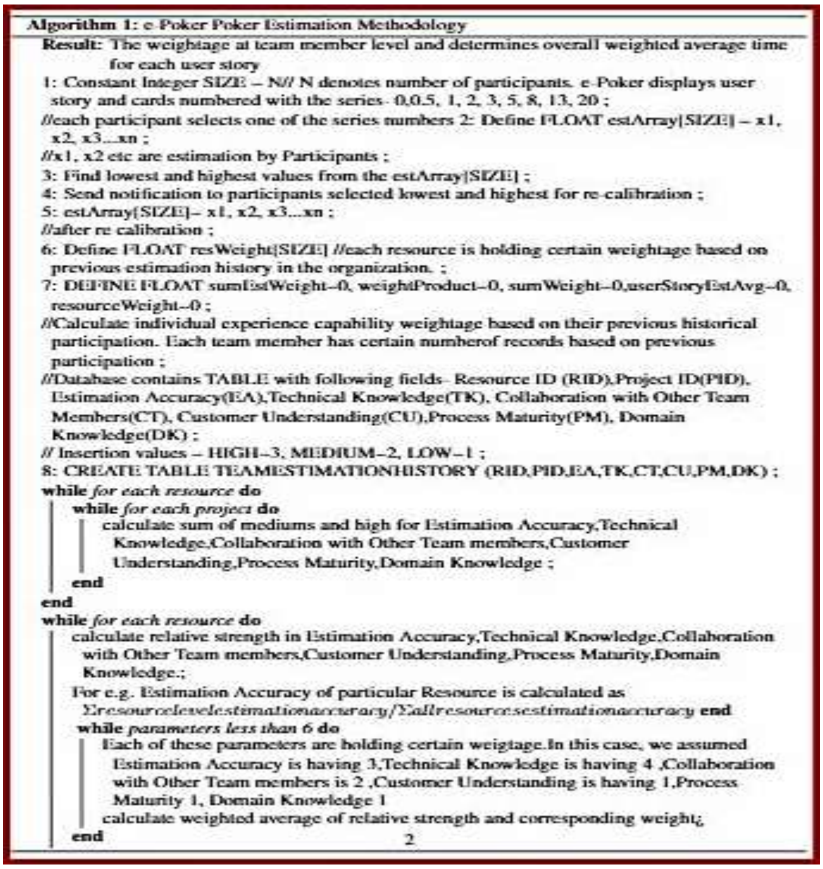
Predicted class	Yes	No
PredObs1	0.003	0.997
PredObs2	0.957	0.043
PredObs3	0.989	0.011
PredObs4	0.001	0.999
PredObs5	0.977	0.023
PredObs6	0.842	0.158
PredObs7	1.000	0.000
PredObs8	0.308	0.692
PredObs9	0.000	1.000
PredObs10	0.979	0.021

Table 5 presents the posterior probabilities. The global accuracy of the model is 1. As depicted in Table 5, predicted Class 6 and predicted Class 8 posterior probabilities are slightly weaker compared to other probabilities. If we want to predict any new project success, the algorithm can find the confidence level by entering respective project perception data. Currently, this data set contains 36 records only. This can be enhanced for better results. Having said that, Naive Bayes does not need much data. If data is providing a probabilistic relationship between attributes without connection to output variable, it works fine. It has a lower probability to overfit with training data compared to other similar algorithms when we have small sample sets.

3.2. e-Planning poker illustrative model

Effort estimation during the planning stage is the most difficult task due to the high volatility in customer requirements. Group estimation/high consensus estimation models for user stories is the most critical exercise in the context of Agile/DevOps. It is used for sprint planning and critical decisions. Group estimates are helpful and appear more rationale compared to individual estimations (subjective measures). However, sometimes estimations get biased with dominant personalities [31, 32]. Estimation should be relative rather absolute, which relieves pressure on the estimator and removes the emotional attachment. These estimations can be done quickly. A simplified progression similar to Fibonacci series of 0, 0.5, 1, 2, 3, 5, 8, 13, 20 is one best sequence for relative sizing If size goes up beyond 20 points, it is better to breakdown and keep the size below 20 points [33]. Team's historical throughput of estimated stories (called as team velocity) help inaccurate sizing and timelines.

Planning poker is one such group estimation technique, which is easily administered. Planning poker estimation is powerful, accurate, unbiased and manageable. As part of planning poker estimation, each team member picks up one user story, discusses the requirement and finally posts his/her estimated story points size. The values are compared. The lowest and highest values are given a chance for justification and re-estimation. The next important step is to determine the team member weightage based his/her previous estimation history in the organization. This will avoid bias in the estimation process. Participant weight is determined based on individual historical capability in terms of Estimation Accuracy, Technical Knowledge, Collaboration, Customer Orientation, Process Maturity, and Domain Knowledge. Each of these factors carries separate weight. The historical dataset is maintained at the team member level as well as at project level. The following Fig. 1 presents the algorithm that determines the weight at the team member level and calculates the overall weighted average story points for each user story.



```
Algorithm 1: e-Poker Poker Estimation Methodology
Result: The weightage at team member level and determines overall weighted average time
        for each user story
1: Constant Integer SIZE – N// N denotes number of participants. e-Poker displays user
  story and cards numbered with the series- 0,0.5, 1, 2, 3, 5, 8, 13, 20 ;
//each participant selects one of the series numbers 2: Define FLOAT estArray[SIZE] – x1,
  x2, x3...xn ;
//x1, x2 etc are estimation by Participants ;
3: Find lowest and highest values from the estArray[SIZE] ;
4: Send notification to participants selected lowest and highest for re-calibration ;
5: estArray[SIZE]– x1, x2, x3...xn ;
//after re calibration ;
6: Define FLOAT resWeight[SIZE] //each resource is holding certain weightage based on
  previous estimation history in the organization. ;
7: DEFINE FLOAT sumEstWeight–0, weightProduct–0, sumWeight–0,userStoryEstAvg–0,
  resourceWeight–0 ;
//Calculate individual experience capability weightage based on their previous historical
  participation. Each team member has certain numberof records based on previous
  participation ;
//Database contains TABLE with following fields- Resource ID (RID),Project ID(PID),
  Estimation Accuracy(EA),Technical Knowledge(TK), Collaboration with Other Team
  Members(CT), Customer Understanding(CU),Process Maturity(PM), Domain
  Knowledge(DK) ;
// Insertion values – HIGH–3, MEDIUM–2, LOW–1 ;
8: CREATE TABLE TEAMESTIMATIONHISTORY (RID,PID,EA,TK,CT,CU,PM,DK) ;
while for each resource do
    while for each project do
        calculate sum of mediums and high for Estimation Accuracy,Technical
          Knowledge,Collaboration with Other Team members,Customer
          Understanding,Process Maturity,Domain Knowledge ;
    end
end
while for each resource do
    calculate relative strength in Estimation Accuracy,Technical Knowledge,Collaboration
      with Other Team members,Customer Understanding,Process Maturity,Domain
      Knowledge.;
    For e.g. Estimation Accuracy of particular Resource is calculated as
      Σresourcelevelestimationaccuracy/Σallresourcesestimationaccuracy end
    while parameters less than 6 do
        Each of these parameters are holding certain weigtage.In this case, we assumed
          Estimation Accuracy is having 3,Technical Knowledge is having 4 ,Collaboration
          with Other Team members is 2 ,Customer Understanding is having 1,Process
          Maturity 1, Domain Knowledge 1
        calculate weighted average of relative strength and corresponding weight;
    end                                       2
```

Fig 1. e-Poker Estimation Algorithm

We assume an illustrative model to present e-Poker Estimation Algorithm. We have taken a team size of 6 as part of the illustrative data set. Each team member is evaluated against Resource ID, Project ID, Estimation Accuracy (EA), Technical Knowledge (TK), Collaboration within the Team (CT), Customer Understanding (CU), Process Maturity (PM), Domain Knowledge (DK) for every project. In this illustration, we assume Team Member 1 is having 10 historical records. Similarly, Team Member 2 is with five records, Team Member 3 with three records, Team Member 4 with two records, Team Member 5 with one record and Team Member 6 with 15 records. There are 8 fields in the table. These fields take numerical values (3-High, 2-Medium, 1-Low). The following sum values are calculated where field value ≥ 2 or 3 (Medium or high). The values are $\sum_{i=1}^{n} \mathrm{EA}_i = 41$, $i = 1, \ldots, n,$ where n is the total number of records, $\sum_{i=1}^{n} \mathrm{TK}_i = 51$, $\sum_{i=1}^{n} \mathrm{CT}_i = 35$, $\sum_{i=1}^{n} \mathrm{CU}_i = 64$, $\sum_{i=1}^{n} \mathrm{PM}_i = 28$, $\sum_{i=1}^{n} \mathrm{DK}_i = 36$. Similar exercise is carried at each team member level, which is presented in Table 6.

Table 6. Team No history – high values

Team No	EA	TK	CT	CU	PM	DK
TM1	17	20	13	24	12	14
TM2	9	9	6	13	6	8
TM3	5	7	4	8	4	4
TM4	2	5	4	5	2	2
TM5	2	3	2	3	2	2
TM6	6	7	6	11	2	6
Total	41	51	35	64	28	36

The relative performance values at the resource level (j is the resource number, e. g., $\sum_{i=1, j=1}^{n} \mathrm{EA}_{ij} / \sum_{i=1}^{n} \mathrm{EA}_i$) are presented in Table 7.

Table 7. Team No relative performance

Team No	EA	TK	CT	CU	PM	DK
TM1	0.4	0.39	0	0.4	0.43	0.39
TM2	0.2	0.18	0	0.2	0.21	0.22
TM3	0.1	0.14	0	0.1	0.14	0.11
TM4	0.04	0.1	0	0.1	0.07	0.06
TM5	0.04	0.06	0	0	0.07	0.06
TM6	0.1	0.14	0	0.2	0.07	0.17
Weights	3	4	2	1	1	1

Table 8 presents the story points estimated by each team member and their member weight which is calculated from Table 7. Member weight is calculated as sum product of (EA×weight + TK×weight + CT×weight + CU×weight + PM×weight + DK×weight). These values are presented in Table 8. Finally, the weighted average story points are calculated as sum-product of story points estimated × weight)/sum of weights as presented in Table 8. The weighted average is calculated as 5.3 sprint points for the given illustrative data.

Table 8. Team No vs estimate vs weightage

Team No	Story points estimated	Weightage
TM1	3	0.40
TM2	5	0.20
TM3	3	0.13
TM4	8	0.08
TM5	5	0.06
TM6	13	0.15

3.3. Statistical view of project management maturity

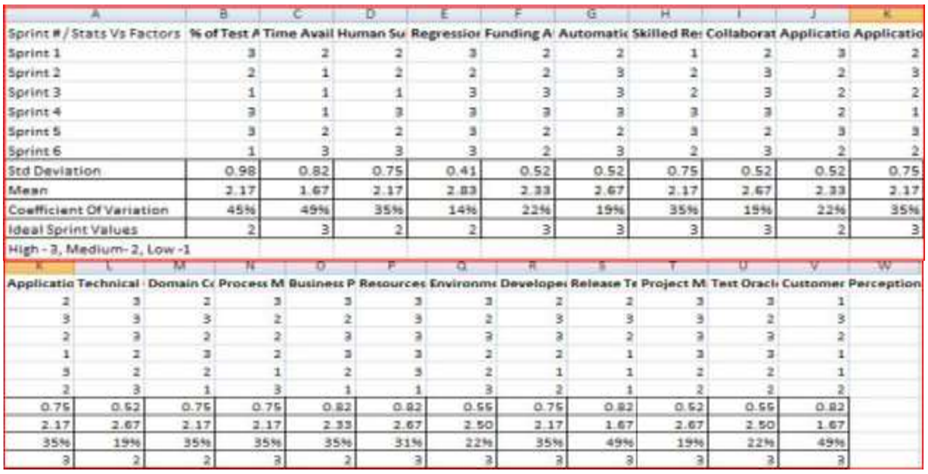
DevOps/Agile's goal is to provide flexibility and speed in the execution of the project. DevOps/Agile introduce short delivery cycles called sprints to felicitate these drivers. However, project management is a complex phenomenon and it requires high maturity. Real-Time communication, self-driven team, rapid adjustments, high collaboration are critical for project success. Project Manager should proactively identify various risks, issues and resolve them on a real-time basis. In this setting, assessment of project management maturity and project sentiment is critical for the manager. This section presents a model to derive a bird's-eye view of the project maturity and project culture/sentiment.

As mentioned in Section 3.1, we collected 21 project factors data related to three broad categories (Product/Application related, People related, Process / Other influencers) to derive project feasibility. This is generally done before the sprint begins. The same exercise should be continued for all sprints to know the sprint progress. We can compare these values with ideal sprint values to determine sprint planning. We assumed an illustrative dataset to explain this model. The key statistical parameters like Standard Deviation (SD), mean, Coefficient of Variation (CV) help us present the health of the sprints movement.

Coefficient of Variation (CV) is also known as a measure of relative variability. It is calculated as the ratio of the Standard Deviation (SD) to the mean (average), CV=(σ/μ)×100, where σ = SD, μ is mean. The following excel tool (as depicted in Table 9, Figs 2-6) helps in understanding the sprint movement. Project Manager can identify the factors deviating and make appropriate control mechanisms.

Table 9. Summary factor vs stats

Factor	SD	Mean	CV	Ideal sprint values
PAP	0.98	2.17	45%	2
TAA	0.82	1.67	49%	3
HS	0.75	2.17	35%	2
RTN	0.41	2.83	14%	2
FA	0.52	2.33	22%	3
ATA	0.52	2.67	19%	3
SRA	0.75	2.17	35%	3
CWT	0.52	2.67	19%	3
AC	0.52	2.33	22%	2
AS	0.75	2.17	35%	3
TC	0.52	2.67	19%	2
DC	0.75	2.17	35%	2
PM	0.75	2.17	35%	3
BP	0.82	2.33	35%	2
RCE	0.82	2.67	31%	3
EA	0.55	2.50	22%	3
DCL	0.75	2.17	35%	3
RCL	0.82	1.67	49%	3
PCL	0.52	2.67	19%	3
TOA	0.55	2.50	22%	3
CCL	0.82	1.67	49%	3



A	B	C	D	E	F	G	H	I	J	K
Sprint # / Stats Vs Factors	% of Test A	Time Avail	Human Su	Regressior	Funding A	Automatic	Skilled Re	Collaborat	Applicatio	Applicatio
Sprint 1	3	2	2	3	2	2	1	2	3	2
Sprint 2	2	1	2	2	2	3	2	3	2	3
Sprint 3	1	1	1	3	3	3	2	3	2	2
Sprint 4	3	1	3	3	3	3	3	3	2	1
Sprint 5	3	2	2	3	2	2	3	2	3	3
Sprint 6	1	3	3	3	2	3	2	3	2	2
Std Deviation	0.98	0.82	0.75	0.41	0.52	0.52	0.75	0.52	0.52	0.75
Mean	2.17	1.67	2.17	2.83	2.33	2.67	2.17	2.67	2.33	2.17
Coefficient Of Variation	45%	49%	35%	14%	22%	19%	35%	19%	22%	35%
Ideal Sprint Values	2	3	2	2	3	3	3	3	2	3
High - 3, Medium- 2, Low -1										

K	L	M	N	O	P	Q	R	S	T	U	V	W
Applicatio	Technical	Domain Cc	Process M	Business P	Resources	Environm	Developer	Release Te	Project M	Test Oracl	Customer	Perception
2	3	2	3	3	3	3	2	2	3	3	1	
3	3	3	2	2	3	2	3	3	3	2	3	
2	3	2	2	3	3	3	3	2	3	3	2	
1	2	3	2	3	3	2	2	1	3	3	1	
3	2	2	1	2	3	2	1	1	2	2	1	
2	3	1	3	1	1	3	2	1	2	2	2	
0.75	0.52	0.75	0.75	0.82	0.82	0.55	0.75	0.82	0.52	0.55	0.82	
2.17	2.67	2.17	2.17	2.33	2.67	2.50	2.17	1.67	2.67	2.50	1.67	
35%	19%	35%	35%	35%	31%	22%	35%	49%	19%	22%	49%	
3	2	2	3	2	3	3	3	3	3	3	3	

Fig. 2. Sprint planning tool

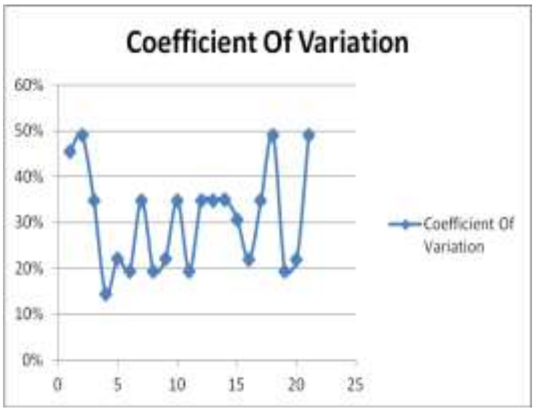


Fig. 3. Coefficient of variation of factors

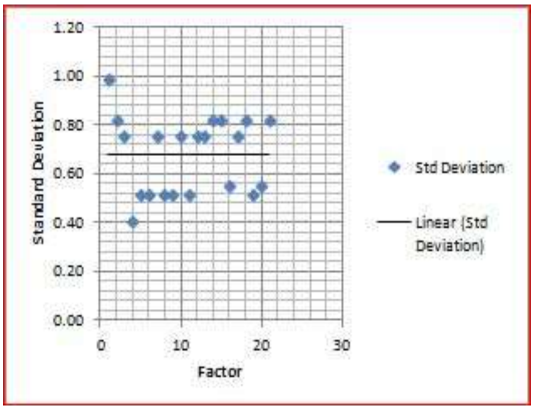


Fig. 4. Std deviation of factors

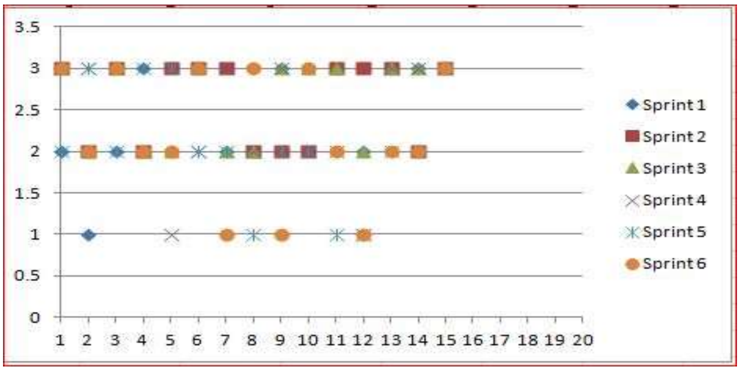


Fig. 5. Sprint fluctuations

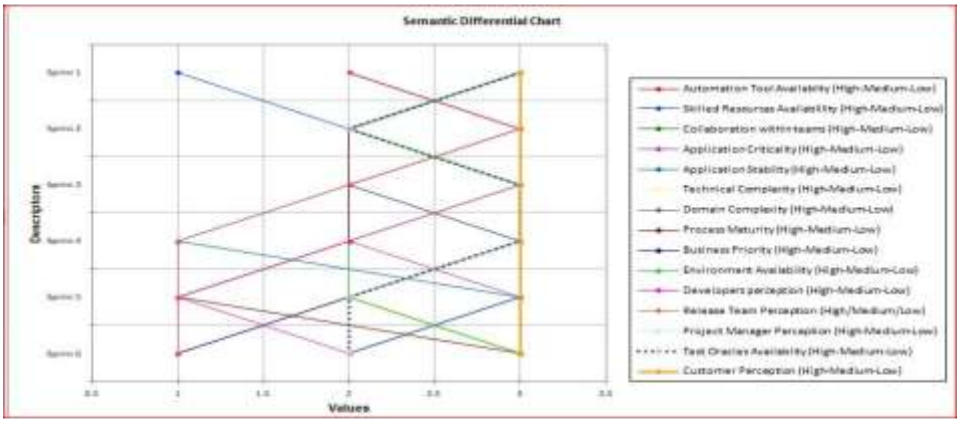


Fig. 6. Semantic view of sprint fluctuations

3.4. Project sentiment part of project management maturity

Ineffective communication, negative sentiment in the project become a high risk for every dollar spent [3]. Tight deadlines, overlapping functions between development, QA and Release teams, rapidly changing business priorities and customer expectations are few triggers for negative tone. Project tone has a profound impact on project success. It is desired to develop an algorithm to understand the tone of collaboration through various communication channels (e.g., project blog/chat messages, customer emails, audit or management reports, customer meetings, daily scrum/stand-up meetings). We have used sentiment analysis and illustrative datasets to develop an algorithm. In the advanced stage, this algorithm can be evolved to a conceptual level rather mere word-level sentimental analysis. In such scenarios, the focus would be on semantic analysis of text and derive conceptual information associated with NLO – natural language opinions. They are built based on the theme that semantically associated thoughts share common sentiment [34]. The following algorithm explains how to determine the project sentiment in the context of various sprints and multiple communication channels. Each project can determine their priorities and can customize the algorithm according to the project needs.

Step 1. Convert every possible communication (including voice files) between project stakeholders (customers, senior management, project members) into Text File.

Step 2. Segregate the Text Files into multiple folders (Customer, Senior Management, Project Level, etc.) {If possible for multiple sprints}.

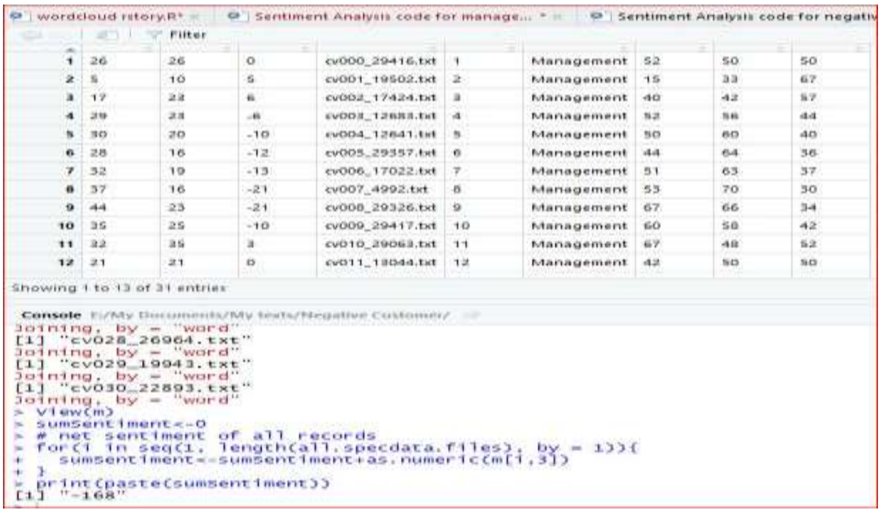
Step 3. Set weight to each of these files – General Weightage Rules{customize to the project requirement}:Customer> Senior Management>Project Manager > Test Manager/Architect > Team – Customer Mail > Customer Voice Call-Senior Management Audit > Customer Work-in-progress Audit.

Step 4. Get Sentiment value for each File { sentiment = positive-negative}.

Step 5. Calculate Net Sentiment {after applying weightages}.

Step 6. Plot sentiments.

We used 252 publicly available illustrative datasets using R Programming to develop this model as showed in Fig. 7.



wordcloud rstory.R*
Sentiment Analysis code for manage... *
Sentiment Analysis code for negativ
Filter

1	26	26	0	cv000_29416.txt	1	Management	52	50	50
2	5	10	5	cv001_19502.txt	2	Management	15	33	67
3	17	23	6	cv002_17424.txt	3	Management	40	42	57
4	29	23	-6	cv003_12683.txt	4	Management	52	56	44
5	30	20	-10	cv004_12641.txt	5	Management	50	60	40
6	28	16	-12	cv005_29357.txt	6	Management	44	64	36
7	32	19	-13	cv006_17022.txt	7	Management	51	63	37
8	37	16	-21	cv007_4992.txt	8	Management	53	70	30
9	44	23	-21	cv008_29326.txt	9	Management	67	66	34
10	35	25	-10	cv009_29417.txt	10	Management	60	58	42
11	32	35	3	cv010_29063.txt	11	Management	67	48	52
12	21	21	0	cv011_13044.txt	12	Management	42	50	50

Showing 1 to 13 of 31 entries

Console E:/My Documents/My texts/Negative Customer/

```
Joining, by = "word"
[1] "cv028_26964.txt"
Joining, by = "word"
[1] "cv029_19943.txt"
Joining, by = "word"
[1] "cv030_22893.txt"
Joining, by = "word"
> View(m)
> sumSentiment<-0
> # net sentiment of all records
> for(i in seq(1, length(all.specdata.files), by = 1)){
+   sumSentiment<-sumSentiment+as.numeric(m[i,3])
+ }
> print(paste(sumSentiment))
[1] "-168"
```

Fig. 7. Sentiment calculator function

Figs 8-9 depict the positive sentiment and negative sentiment. Similarly, project manager can get the word cloud, which provides cursory direction of the project written communication.

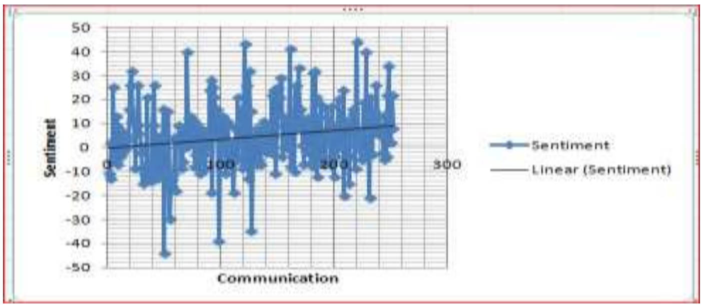


Fig. 8. Positive sentiment graph

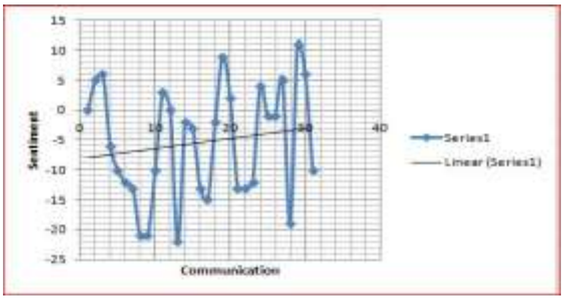


Fig. 9. Negative sentiment graph

4. Threats to validity

The success of the project depends on how well the factors associated with automation are derived and managed. As part of the dataset creation and feasibility, predictability algorithm development, authors interviewed project managers and delivery managers working in the industry and collected 36 real-time test automation projects perception data for the 21 identified factors. These projects are of varied size and from different domains. It is desired to increase the sample size so that feasibility accuracy is improved from the machine learning context. We used illustrative datasets for the case of Effort Estimation Model, Statistical View of Project Management Maturity and Project Sentiment View. However, we can collect real-time data for these three tools and customize the model suiting to project requirements. Data feeding is highly critical while calculating the project sentiment value. It is the project manager's discretion which data files to be fed to the system.

5. Conclusion

The changing traditional software development methodologies, traditional roles, distributed teams, dynamic customer requirements, high feedback loop, a large number of deployments, less documentation, multiple sprints, automation of DevOps and Agile setting, etc are making project complicated. These aspects are increasing project uncertainty. With the traditional heuristic assessment method, it is difficult to assess and predict the project outcomes. On top of these issues, project managers face unprecedented budget cuts. They are expected to deliver results precisely. In such critical circumstances, there is a need to develop predictability algorithms for project feasibility. The traditional estimation models like function point or use case points may not work in a DevOps/Agile context. There is a need for reliable, consensus-based effort & cost estimation methods in DevOps / agile context. It is also desired to develop metrics and dashboards to measure / monitor real-time communication and collaboration of the DevOps teams. Ultimately, the goal of these tools is to improve the productivity, decision precision, and professionalism that are the critical objectives of DevOps/Agile.

References

1. Soni, M. The CASE for DevOps in Insurance Industry-End to End Automation on Cloud with Build Pipeline. – In: Proc. of IEEE International Conference on Cloud Computing in Emerging Markets, Gandhinagar, India, 2015.
2. Fitzgerald, B., K.-J. Stol. Continuous Software Engineering and Beyond: Trends and Challenges. – In: Proc. of 1st International Workshop on Rapid Continuous Software Engineering (RCoSE), 2014, pp. 1-9.
3. The High Cost of Low Performance. The Essential Role of Communications (Accessed on 21 November 2017). **http://www.pmi.org/-/media/pmi/documents/public/pdf/learning/thought-leadership/pulse/the-essential-role-of-communications.pdf**
4. Drury, M., K. Conboy, K. Power. Decision Making in Agile Development: A Focus Group Study of Decisions and Obstacles. – In: Proc. of AGILE Conference (AGILE'11), 2011, pp. 39-47.
5. Alhazmi, A., S. Huang. A Decision Support System for Sprint Planning in Scrum Practice. – In: Proc. of IEEE SoutheastCon, Florida, USA, 4-7 April 2013, St. Petersburg, Rusia.
6. Sheshasaayee, A., V. Hannah. Identifying Bottlenecks in Agile Software Development Using Theory of Constraints Principles. – Indian Journal of Science and Technology, Vol. **8**, 2015, No 29.
7. Kurmann, R. Agile SPL-SCM Agile Software Product Line Configuration and Release Management. – In: Proc. of 1st International Workshop on Agile Product Line Engineering (APLE'06), Maryland, USA, 2006.
8. Ahmed, F., S. Bouktif, L. F. Capretz. Organizational Behavior & Software Product Line Engineering: An Empirical Study. – In: Proc. of International Conference Computer Systems and Applications, AICCSA, IEEE/ACS, Rabat, 2009, pp. 420-427.
9. Erich, F., A. Chintan, M. Daneva. DevOps Literature Review (Accessed on 24 October 2017). **http://www.utwente.nl/bms/iebis/staff/amrit/devopsreport.pdf**
10. Claps, G. G., R. B. Svensson, A. Aurum. On the Journey to Continuous Deployment: Technical and Social Challenges along the Way. – Information and Software Technology, Elsevier, Vol. **57**, 2015, pp. 21-31.
11. Angara, J., S. Prasad, S. Gutta. DevOps Product Line Engineering (DPLE): Where DevOps Meets Software Product Lines. – International Journal of Sciences and Research, Vol. **72**, 2016, No 11.
12. Test Automation Assessment and Implementation (Accessed on 24 October 2017). **https://xbosoft.com/software-qa-consulting-services/test-automation-assessment/**
13. Schur, M., A. Roth, A. Zeller. Mining Workflow Models from Web Applications. – IEEE Transactions on Software Engineering, Vol. **41**, 2015, No 12, p. 1.
14. Jumpstarting DevOps with Continuous Testing (Accessed on 24 October 2017). **https://www.cognizant.com/content/dam/Cognizant_Dotcom/whitepapers/Jumpstarting-DevOps-with-Continuous-Testing-codex1719.pdf**
15. Continuous Testing for IT Leaders (Accessed on 24 October 2017). **https://alm.parasoft.com/continuoustestingbook**
16. Cannizzo, F., R. Clutton, R. Ramesh. Pushing the Boundaries of Testing and Continuous Integration. – In: Proc. of Agile Conference (AGILE'08), Toronto, 2008, pp. 501-505.
17. Reich, C., B. Scharpf. Continuous Software Test Distributed Execution and Integrated into the Globus Toolkit. – In: 5th International Symposium on Parallel and Distributed Computing, Timisoara, 2006, pp. 185-190.
18. Saff, D., M. D. Ernst. Continuous Testing in Eclipse. – In: Proc. of 27th International Conference on Software Engineering (ICSE'05), St. Louis, 18-20 May 2005, pp. 668-669.

19. Muthusamy, T., K. Seetharaman. A New Effective Test Case Prioritization for Regression Testing Based on Prioritization Algorithm. – International Journal of Applied Information Systems, Vol. **6**, 2014, No 7, pp. 21-26.

20. Ramler, R., K. Wolfmaier. Economic Perspectives in Test Automation: Balancing Automated and Manual Testing with Opportunity Cost. – In: Proc. of International Workshop on Automation of Software Test, Shanghai, China, 23 May 2006.

21. Berner, S., R. Weber, R. K. Keller. Observations and Lessons Learned from Automated Testing. – In: Proc. of 27th International Conference on Software Engineering (ICSE'05), St. Louis, MO, USA, 2005. pp. 571-579.

22. Process for Scoping and Planning Test Automation Projects (Accessed on 24 October 2017). **https://www.stickyminds.com/sites/default/files/presentation/ file/2013/07STRER_W8.pdf**

23. A Process for Scoping and Planning Test Automation Projects. – In: Proc. of International Conference on Software Test Analysis and Review. **https://www.stickyminds.com/sites/default/files/presentation/file/2013/ 07STRER_W8.pdf**

24. Geetha Devasena, M. S., G. Gopu, M. L. Valarmathi. Automated and Optimized Software Test Suite Generation Technique for Structural Testing. – International Journal of Software Engineering and Knowledge Engineering, Vol. **26**, 2016, No 1, pp. 1-13.

25. Runeson, P., M. Host. Guidelines for Conducting and Reporting Case Study Research in Software Engineering. – Empirical Software Engineering, Vol. **14**, 2008, No 2, pp. 131-164.

26. Robinson, W. N., T. Deng, Z. Qi. Developer Behavior and Sentiment from Data Mining Open Source Repositories. – In: Proc. of 49th Hawaii International Conference on System Sciences (HICSS'16), Koloa, HI, 2016, pp. 3729-3738.

27. Calefato, F., F. Lanubile. Affective Trust as a Predictor of Successful Collaboration in Distributed Software Projects. – In: Proc. of 1st International Workshop on Emotional Awareness in Software Engineering (SEmotion), IEEE/ACM, Austin, Texas, 2016, pp. 3-5.

28. Muhammad, U., J. Börstler, K. Petersen. An Effort Estimation Taxonomy for Agile Software Development. – Int. J. Soft. Eng. Knowl. Eng., Vol. **27**, 2017, No 4, 641.

29. Borissova, D., I. Mustakerov, D. Korsemov. Business Intelligence System via Group Decision Making. – Cybernetics and Information Technologies, Vol. **16**, 2016, No 3, pp. 219-229.

30. Sgurev, V., S. Drangajov. Risk Estimation and Stochastic Control of Innovation Processes. – Cybernetics and Information Technologies, Vol. **14**, 2014, No 1, pp. 3-10.

31. Haugen, N. C. An Empirical Study of Using Planning Poker for User Story Estimation. – In: Proc. of Agile Conference, Minneapolis, MN, USA, 2006.

32. Zahraoui, H., A. Mohammed, J. Idrissi. Adjusting Story Points Calculation in Scrum Effort & Time Estimation. – In: Proc. of 10th International Conference on Intelligent Systems: Theories and Applications (SITA'15), Rabat, Morocco, 2015, pp. 20-21.

33. Atlassian the Secrets Behind Story Points and Agile Estimation (Accessed on 30 November 2017). **https://www.atlassian.com/agile/estimation**

34. An Introduction to Concept-Level Sentiment Analysis. Temasek Laboratories. National University of Singapore (Accessed on 21 November 2017). **http://www.sentic.net/intro-to-concept-level-sentiment-analysis.pdf**

Received: 26.05.2019; Second Version: 24.08.2019; Accepted: 23.03.2020