Application of The Clustering In Software Development Analysis

T Afanasieva[1] **and I Sibirev**[1]

[1] Information Systems Department, Ulyanovsk State Technical University, Russia,
tv.afanasjeva@gmail.com, ivan.sibirev@yandex.ru

Abstract. The paper describes the task of identifying of homogeneous groups of software projects with similar behavior of metrics of software development processes. This is necessary for analysis, understanding and improving the practice of software development, for planning, detection of numerical relationships between key metrics of the development process and the quality and cost of the project; it is also necessary to identify where efforts are made, where defects occur; to identify the impact of technological, process, and organizational aspects on the result. To solve this task the FBC-approach (Fuzzy Behavior Clustering) based on combination of three types of time series clustering is applied to group the software projects taking in account similar behavior of their development processes. Two experiments on the set of metrics from open projects repositories showed the availability of the proposed FBC-approach to identify homogeneous groups of software projects with similar behavior of key software metrics.

1. Introduction

The beginning of the 21st century is characterized by the entry of mankind into the "information" era, information becomes the main resource, tool and output of production. At the same time, the software engineering reaches industrial scales, software projects are becoming increasingly complex. According to the statistics of software projects success around the world, more than half of them are controversial or disastrous [13]. In the software development the version control systems, project- and issue-tracking systems (GIT, SVN JIRA, TFS, Bugzilla, Trac, Mantis, Redmine) are used to store and monitoring the processes. They operate with different terminology for describing the software development process and use repositories of metrics which are not always coincident with each other.
Recently, a lot of works have appeared where it is insistently stated about the need to analyze data from software development repositories. According to the work of A. Mockus [9], the objectives of such analysis are understanding and improving the software development practice, identifying numerical relationships between key metrics of the development process and the quality, cost estimating of the project; identification of where the main efforts are made, where the defects occur; revealing the influence of technological, process, organizational aspects on the result [9]. Therefore, it is necessary to create data mining tools that allow us to perform modeling, analysis, prediction of phenomena in software projects and create tools that improve the time, quality, and cost of software development according to the data from the repositories.

Among the tasks to be performed when analyzing data from a repository based on clustering, we will distinguish the following tasks:

- Determination of outliers and anomalies in the results and software design processes;
- Detection of similar patterns;

- Search for the causes of anomalies and how to remove them;
- Prediction of software defects (it was revealed in the work [17] that 20% of the code contains on average 80% of errors; identification of the corresponding blocks will facilitate work of testers);
- Preprocessing of data for obtaining more stable results when applying machine learning and data mining methods.

A review of the literature on the repository data analysis is given in [9]. Data from a software repository is also used to plan and coordinate the work, to discover what makes the changes difficult, to study which process control tools work and why, to check the release readiness criteria; to implement a process approach in assessing software quality, to search for people (personnel, customers, etc.); to detect defects, to predict risks from software changes, to find independently serviced pieces of code, etc. [9].

2. Related works

Clustering data allows us to split the objects of the software development process into homogeneous groups. In the work [5], the program code from the repository is subject to agglomerate hierarchical clustering in order to extract the component architecture of object-oriented systems. The coupling between classes is one of the metrics of object-oriented systems. Inheritance, composition, aggregation, and method calls are distinguished among the possible dependencies between object-oriented components. Applying to these dependencies the algorithm of agglomerate hierarchical clustering (the method of single connections using the measure of similarity as a metric) creates components in the object-oriented system.

The work of Chintakindi Srinivasa, Vangipuram Radhakrishnab and Dr.CVGuru Rao [14] describes the software components clustering from available repositories for efficient searching and for creating libraries. In this paper, a new similarity function is defined to calculate the similarity between any two software components, it is used for clustering software components using the "maximum capture" method. After formation of clusters, each cluster is identified by its word pattern calculated using fuzzy Gaussian membership functions. Uncontrolled clustering with Markov process templates or controlled clustering algorithms with templates that can be used to classify components in decision-making tasks, are possible.

Many works are devoted to the prediction of software errors based on data from a repository, reviews on this topic are given in [6], [12] and [16]. Most of the works involves the processing and analysis of program code using a variety of metrics based on historical data from previous versions of the software without clustering.

In the works of T. Zimmermann et al. [18], N. Nagappan et al. [10] there was investigated the chance of success for interproject prediction of defects, but the authors concluded that there is no any single set of metrics suitable for describing any projects, the models of one project history do not allow transfer to other projects. Upon that, defect prediction models could be accurate when they were obtained from similar projects (the similarity was not precisely defined). In this connection, the problem of "similarity" of projects arises. In the work [6], it was attempted to solve it by clustering vectors, the elements of which are correlations between the metrics of object-oriented programming (OO-metrics) and the number of defects.

In the work of James W. Tunnell [15], data from the repository are used to construct time series of problems that arise in the development of software, and that are subproblems of a certain parental problem. The sliding windows method, autoregression of models, their testing, model selection using a penalty for a residual error and the number of parameters are used. The selected models are used to predict program code errors.

Based on the results of a review of works in the field of clustering in the software development processes analysis, it can be concluded that the use of only static metrics does not fully allow analysis of the development process status. Existing approaches to the software development projects analysis do not provide for clustering by dynamic process metrics. For a deeper analysis of software

development, it is necessary to investigate metrics not only of the program code (static), but also the dynamic metrics of the software development processes (grouping of processes with similar dynamics, revealing the degree of proximity of processes, comparing groups of linguistic estimates reflecting qualitative metrics and processes are necessary).

Time series (TS) are often used to represent and study the dynamics of processes in complex systems, so the clustering of time series is relevant and in demand and is considered an important step in the analysis of processes that allows homogeneous groups of data to form.

A review of the time series clustering is given in [3], [7], [19], where the following are distinguished:

- Clustering of raw data (point-based, time-frequency metrics), upon that, the same length, scope, close regions of time series values are necessary; the problem becomes the significant effect of noises on the clustering result;
- Clustering based on density and based on a quantum grid, the limitation of which is the loss of individual features of objects, the loss of perception of time series as a chronological sequence, the nature of time series behavior (increase, decrease, etc.) is not taken into account;
- Clustering based on models or machine learning: statistical and artificial neural networks. They are characterized by dependence on idealization in the construction of a model, by the loss of individual features of objects when they are replaced by generalized metrics; moreover, there is no correction for a time series behavior, that are fuzzy in nature [3], [7] and [19].

Basic limitations of the considered above approaches for clustering of software metrics are the requirements of the same length and scope of the time series [7] and focusing on static data only. If the nature of time series behavior is not taken into account for clustering, then the results of clustering are likely to inherit this flaw. It is impossible to automatically detect time series similar to the accuracy of compression, stretching, and shifts of individual sections.

Therefore in [2] the FBC–approach combining the advantages of mention above approaches was proposed, moreover it includes clustering of time series by their fuzzy behavior.

In this paper, the application of FBC-approach (Fuzzy Behavior Clustering) [2] to group the software projects metrics in the form of time series is given. The FBC- approach allows us to identify new knowledge about the nature of software development processes based on data extracted from the software repository and, on this basis, to formulate solutions for their improvement. This knowledge is presented on three levels of the hierarchy of the time series model (general trends, trends, and fluctuations). The proposed approach makes it possible to cluster time series of different dimensions, time scale and span; and also, to receive clusters of time series of similar behavior. The use of fuzzy approach to cluster the behavior improves the noise immunity of the algorithm and increases its ergonomics.

3. Problem Statement

Let us consider the informational model of the software projects in the form

$$D = \{d_{ijt}\} \tag{1}$$

where i is a number of the project $(i = 1..m)$, j is a number of the key metric $(j = 1..n)$, t is a the time moment for the key metric $(t = 1..T)$.

The task is to analyze software projects for key metrics using model $D = \{d_{ijt}\}$ with a view to identify groups of similar projects Cl_k that is the clusters of software projects,

$k = 1..Q$. When $i = const$, $d_{i=const,i,t}$ is a set of time series characterizing temporal (dynamical) changes of project metrics. If $j = const$, then $d_{i,\,j=const,t}$ is the set of time series describing the temporal changes of the metric for different projects, such as, the number of commits, the number of branches, the number of developers, the average time between commits, the development time of the project,

etc. In the case $t = const$, $d_{i,j,t=const}$ are tabular data, with text, numeric and binary values, and static project metrics at a given time.

The problem is to group the software projects by temporal metrics (1) with similar structures and changes as well.

Then the clustering of software projects metrics is considered as the task of grouping of the time series

$d_{i,j=const,t}$, obtained from the repository $D = \{d_{ijt}\}$, and of deriving k clusters $C_k^{Dynamic}$ similar on time series behavior and on their density:

$$D \xrightarrow[j=const]{} C_k^{Dynamic} \quad (1)$$

In this paper similarity on time series behavior will be considered using the fuzzy sets in the form of general tendencies from the term set {"fall", "stability", "growth" and "fluctuation"}. Similarity on time series density will be determined based on Euclidean measure. Parameter k should be calculated in dependence of data. The task in the form (2) will be named as dynamic clustering (for short) in the paper further.

The task (2) is a machine learning task and the its results are needed to better understand of software development processes. The obtained clusters allow to extract information from repositories about groups of objects, such as tasks, projects and developers in temporal space. This knowledge is useful in software management to make decisions on improving development process in the future.

4. The software metrics clustering using time series

To cluster the software projects in the form (2), it is proposed to apply Fuzzy Behavior Clustering (FBC) approach described in the work [2]. It uses for clustering the time series representation at three levels of their hierarchy (general trends, trend component, fluctuation component) [8], the fuzzy trend concept; the fuzzy linguistic terms apparatus [1], the technique for extracting the main trend and the time series trend; F-transformation [11]. Besides that, it combines three time series clustering methods: point-based, feature-based and model-based.

The scheme of the proposed dynamic clustering in the form (2) is shown in figure 1. The input data are extracted from the software repository D. The input could be presented by: commit hash No, branch name, author's login commit, commit date. Then data are pre-processed to obtain a set of statistical project metrics from the repository: the number of commits, the number of branches, the number of developers, the average time between commits, the project development time, etc. With the help of data pre-processing, time series of these metrics are obtained for developers, years, months, days, hours. Then FBC-approach of time series of these metrics is then applied. As a result, k clusters $C_k^{Dynamic}$ are obtained being the outputs of dynamic clustering of software projects by temporal metrics.

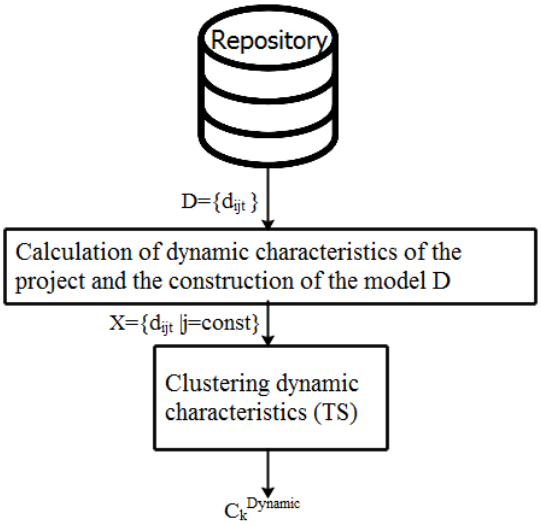

Figure 1. The proposed solution to the problem of clustering a three-dimensional structure of software development project process metrics $D=\{d_{ijt}\}$ from software repository, where in case of $i=const$ or $j=const$, d_{ijt} are time series of different dimensions, diapason, scope, and different trends of behavior.

Below we briefly recall the algorithm of FBC-approach of time series, proposed in the paper [2].

Let's $i=const$ or $j=const$ for $D=\{d_{ijt}\}$. Then we obtain the input data in the form of a set of time series which we denote $=\{X_s\}$, where $X_s=d_{i=s,j,t}$ $(s=1..m)$ or $X_s=d_{i,j=s,t}$ $(s=1..n)$. These time series could be of different dimension, diapason, scope and different trends of behavior. Output data of the FBC-approach are the clusters $C_k^{Dynamic}$ of time series X_s corresponding to software development process metrics.

Since FBC-approach uses the representation of the behavior of time series at three levels of the hierarchy (general trends, trend component, fluctuation components) [2], we define the set of linguistic terms of the main time series trend in the form GT= {"fall", "stability", "growth" and "fluctuation"}.

The adopted technique of FBC-approach to software metrics in respect to expression (2) includes the following steps.

First step. The transformation that puts time series X_s in line with the linguistic term of the general tendencies from the set of linguistic terms of the general time series trend GT= {"fall", "stability", "growth" and "fluctuation"}, according to the algorithm given in the work [1]:

$$X_s \longrightarrow gt_s,\ gt_s \in GT,\ X_s \in Y. \quad (2)$$

Second step. Clustering of the gt_s by the general trends using the equivalent in the form of a linguistic term of the general trend from the set GT= {"fall", "stability", "growth" and "fluctuation "}. In this case, the set of corresponding time series $Y=\{X_s\}$ is divided into subsets, or clusters of main trends:

$$Y=Y_{fall}\ \mathrm{U} Y_{growth}\ \mathrm{U} Y_{stab}\ \mathrm{U} Y_{fluct}. \quad (3)$$

Third step. The numerical clustering of time series X_s from the clusters of Y_{fall}, Y_{growth}, Y_{stab}, Y_{fluct} based on the transformation of each time series $X_s \in Y$ into the parameter vector Z_s.

A) Obtaining the parameter vector Z_s for each $X_s \in Y$. Grouping values of each initial time series X_s on $N1$ clusters. Then barycentres of clusters, which are considered as parameters of time series, are calculated. Then the time series of parameters Z_s is built from the barycentres sorted in chronological order. Upon that, the set $Z=\{Z_s\}$ of vectors composed of the obtained parameters is divided into

clusters of main trends

$$Z = Z_{fall} \cup Z_{growth} \cup Z_{stab} \cup Z_{fluct} . \quad (5)$$

corresponding to the original time series from the sets Y_{fall}, Y_{growth}, Y_{stab}, Y_{fluct} in accordance with the partition of the set Y to the subsets from step 2.

B) Clustering of the vectors Z_s from the sets Z_{fall}, Z_{growth}, Z_{stab}, Z_{fluct} to $N2$ clusters. Output is the clusters $C_k^{Dynamic}$ which elements are time series X_s, where $N2 \le k \le 4 * N2$. Note that the values of $N1$ and $N2$ are set by users. This setting is made due to the $N2$ parameter. The values of $N2 > 8$ correspond to the super-high intra-cluster similarity of time series requirement, up to full visual correspondence at the level of local trends. Values $N2 = 0,1,2,..,4$ – meet the requirement of low intra-cluster similarity of time series.

FBC-clustering algorithm has a high level of modularity, any numeric clustering methods can be used for 3rd step of clustering, including fuzzy ones. The proposed solution of the clustering problem for software development processes based on FBC-approach makes it possible to cluster software development processes for time series of different dimensions, scope and diapason; it also increases the information content of time series clustering in comparison with numerical methods, as it extracts knowledge about types of time series behavior.

5. Experiments on clustering the software development process metrics

The main purpose of the experimental study of the proposed approach is to show its usefulness for clustering the software development metrics using the FBC-approach [2]. The numeric clustering of time series in FBC-approach in experiments will be used in the form of hierarchical clustering [5]: Ward method, single-linkage method and centroid method. Euclidean distances between the centers of the numeric clustering were used.

Experiment 1. The goal of the Experiment 1 was to obtain clusters of software development processes based on FBC-approach of time series by metrics represented in the form of a number of commits for various projects and their versions from Git repositories to identify homogeneous groups of projects with similar dynamics, giving to groups of linguistic assessments used to plan and manage software development processes.

The experiment uses data from 142 time series of daily commits for various projects and their versions from Git repositories of 3 projects: MongoDB (10 years of development, [https://github.com/mongodb/mongo.git]); Libvideo - 3 years of development [https://github.com/i3arnon/libvideo.git]; ProjectUD - 10 days of development [https://github.com/Anton7393/ProjectUD.git].

Commits of the form (Hesh Commit No, the name of the branch, the author of the commit, the date of the commit, for example, “2d700d9 | 1.8 | Anton 7393 | 2016-07-08 13:36:01 +0400 | ^)” have been received from Git repositories. Based on that data, a set of static metrics for projects from the Git repository was obtained by preprocessing: the number of commits, the number of branches, the number of developers, the average time between commits, the time of project development, etc. With the help of pre-processing the data, time series were obtained for employees, years, months, days, hours. These time series extracted from the specified sources, have different lengths and different main trends.

Application of the first step of FBC-approach ensured separation of 142 time series of commits to different projects into 4 groups of time series according to their main trends: "chaos" with 63 time series, "fluctuation" with 32 time series, "fall" with 33 time series, "growth" with 14 time series. At the second step of FBC-approach, further decomposition of each group of time series into the number of clusters equal to one-third of the total number in the corresponding group was carried out. Clustering was carried out for normalized time series using the centroid method, the single connection method, and the Ward method. Figure 2 shows examples of clusters of time series obtained with the help of FBC-approach. The OX axis shows the normalized number of days, and the OY axis shows the normalized number of commits per day.

FBC-approach with the use of the Ward method gave within each cluster of trends from 3 to 4

clusters of non-unit size, examples of which are shown in figure 2, and several single clusters. Namely, with the general trends "fall" were formed 4 single clusters and 4 clusters with 7, 4, 3 and 15 time series (figure 2a). In the cluster of "growth" trends 3 clusters with 3, 4 and 6 time series were found (figure 2b). The cluster of "chaos" trends consists of 15 single clusters and 3 clusters with 7, 9 and 32 time series (figure 2c). The cluster of "fluctuation" trends involves 4 single clusters and 4 clusters with of 7, 6, 4 and 15 time series (figure 2d).

So, in terms of the time series of daily commits, the following knowledge about change of software projects in accordance to main tendency were obtained using proposed clustering with Ward method:

- 44% of the projects tends to be "chaos",
- 23% of projects has the trend "fluctuation",
- 23% of projects are characterized by the trend "fall",
- only 10% of projects are developed in direction of "growth".

When FBC-approach was used with the centroid method or the single-linkage method, one cluster collects most of the time series, a plurality of clusters of unit capacity is also obtained, and one or two clusters collect from 2 to 3 time series. As the number of clusters increases single clusters continue to be outlay from the total mass. That is, these methods are good for detecting abnormal time series far from the main group. Single clusters require management attention as being abnormal. They can be used to search for software projects with atypical development process.

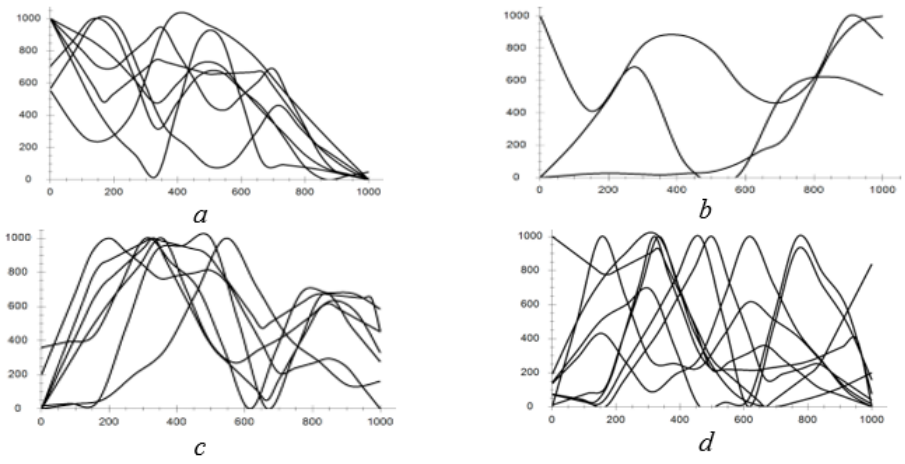

Figure. 2. Example of FBC-approach using the Ward method for numeric clustering. Examples of clusters with the main trend: a) "fall"; b) "growth"; c) "chaos"; d) "fluctuations". Axis OY is the normalized number of commits, axis OX is the normalized number of days.

Another findings is that FBC-approach with the use of the centroid method, the single-linkage and the Ward method formed different clusters of time series with intersections. It is customary to select the static clustering method for a researcher's task in the practice of applying methods of clustering. Moreover, the Ward method is used to obtain spherical embedded clusters. The centroid method is good for finding anomalies. In our experiment 1, the method of single-linkage gave better results if the task was to identify time series that are similar to accuracy of compression, stretching, shifts, and symmetry of sections.

Experiment 2. The goal of the Experiment 2 is to obtain clusters of software development processes based on FBC-clustering of time series by metrics represented by the number of commits of individual developer in software projects from mention above Git repositories. We hoped to identify homogeneous groups of developers with similar dynamics in the work, giving to project teams of linguistic estimates used for planning and management of software development processes.

The initial data of the experiment are: 34 developers, 34 time series, 11 clusters, the length of time series is different and is a term of participation in the projects. One value of time series is the number of commits per day (axis OY is the normalized number of commits, and axis OX is the normalized number of days).

Figure 3 shows some results of FBC-clustering with the single-linkage clustering. The result of the application of FBC-clustering with dividing time series by behavior and density allows us to extract the following knowledge:

- Five software developers decrease their activity. Two clusters C1 and C2 with a falling trend containing 2 and 3 time series of software project metrics (see figure 3a).
- Five software developers increase their activity. Two clusters with a general trend of "growth" (with 4 and 1 time series) were obtained (cluster C3 with 4 time series is depicted on the figure 3b).
- The activity of fifteen software developers characterized by non-regular fluctuations. Four clusters of the "chaos" trend for 9, 4, 1 and 1 time series (cluster 4 with 9 time series and cluster C5 with 4 time series are depicted on the figure 3c).
- Nine software developers characterized by regular fluctuations. Three clusters with the trend "fluctuation", including 7, 1 and 1 time series (figure 3d shows the cluster D6 with 7 time series).

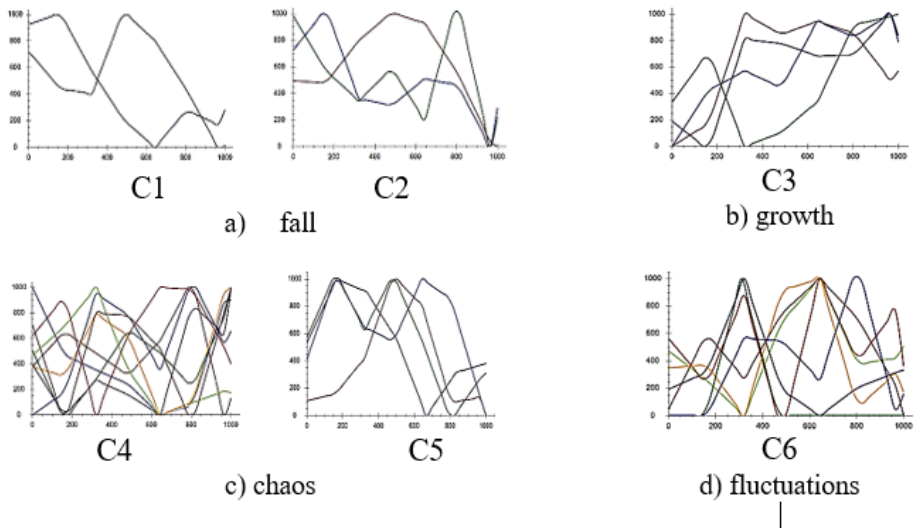

Figure. 3. Clusters of developers with dynamics a) fall, b) growth, c) chaos, d) fluctuation

By assigning to software developers the tasks suitable for this cluster, it is possible to stimulate professional growth if the developer's parameters are below the average ones for the cluster or decreasing. We can form a development vector to go to another cluster, if the parameters are higher than the average ones for the cluster. Clustering methods are used to search for anomalous objects (in this experiment, they are clusters of unit capacity), which, in particular, is suitable for "searching for talents".

Table 1 shows the inter-cluster distances (Euclidean distances between the centers of the clusters were used).

Table 1. Table of inter-cluster distances.

	C1	C2	C3	C4	C5	C6
C1	0	1284	984	983	1059	1030
C2	1284	0	755	770	1102	647

C3	984	755	0	808	707	698
C4	983	770	808	0	1148	736
C5	1059	1102	707	1148	0	860
C6	1030	647	698	736	860	0

Table 1 characterizes the mutual distance and scattering of cluster centers by data sets.

Inter-cluster distances are not homogeneous, what allows us to suggest, in the case of clusters, similarity of sets of control actions. The use of the first step of FBC-approach with division by general trends clusters made it possible to obtain clusters slightly different in their inter-cluster distances, but differing in the general trend of behavior.

Evaluation of the FBC-clustering results effectiveness was carried out using such indexes for clustering results evaluation as "The Ball-Hall index" and "Calinski-Harabasz index" [4]. The Ball-Hall index and the Calinski-Harabasz index turned out to be in 3 and 2 times better for FBC-approach, in particular, with the centroid method, than the corresponding point-wise clustering indexes with the centroid method for time series which are linearly interpolated to equal-dimensional series. This confirms the quality of clustering by the FBC-approach.

The FBC-approach application for the software projects analysis has shown the availability of the proposed grouping tools which allow us to identify on the basis of metrics from the software repository, homogeneous groups of software projects with similar behavior of key metrics. The FBC-approach can be used by software project managers, lead programmers, system administrators to improve the quality of project planning based on the extracted information.

6. Conclusion

In this paper the machine learning task of software project clustering is considered, described and solved using FBC-approach. The scheme and the adopted technique of FBC-approach are provided for clustering of temporal software metrics? Which could be of different length, behavior and diapason. The advantage of the FBC-approach is that it allows us to identify, on the basis of data from the repository, homogeneous groups with similar dynamics of key software metrics. The proposed approach showed their performance capabilities and efficiency in the clustering of software development processes according to the repository metrics presented by time series. The clustering of software development processes based on the repository metrics presented by time series describing the state and behavior of a system, will allow solving tasks on improving the practice of software development, planning, revealing numerical relationships of quality, cost of the project based on key metrics of the development process; identification of where efforts are made, where defects occur; revealing the influence of technological, process, and organizational aspects on the result. Grouping of dynamically changing key software development metrics allows us to identify groups of related processes, assign labels to classes, use results in order to search for anomalies and defects, and to build forecasts, what will be used in making managerial decisions.

The Future work will be focused on the study of using the domain knowledge in the form of domain ontology to produce the linguistic summarization of proposed machine learning results.

7. References

[1] Afanasieva T and Sapunkov A Selection of Time series Forecasting Model Using a Combination of Linguistic and Numerical Criteria. In Proc. of 2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT) 2016 pp 341-345

[2] Afanasieva T, Yarushkina N and Sibirev I Time Series Clustering using Numerical and Fuzzy Representations. In Proc. Of Joint 17th World Congress of lnternational Fuzzy Systems Association and 9th International Conference on Soft Computing and Intelligent Systems (IFSA-SCIS 2017), Otsu, Shiga, Japan, June 27-30, 2017. 978-1-5090-4917-2/17

[3] Bagnall A, Lines J, Bostrom A, Large J and Keogh E The Great Time Series Classification Bake Off: A Review and Experimental Evaluation of Recently Proposed Algorithms. Data Mining and Knowledge Discovery 2016 pp 1-55

[4] Bernard D Clustering Indices. University Paris Ouest. Lab Modal'X. April,2013, 34p.
[5] Budhkar, Sh., Gopal, Dr. A. Component identification from existing object oriented system using Hierarchical clustering. IOSR Journal of Engineering.Vol.2(5), May, 2012, pp. 1064-1068. – URL: http://www.iosrjen.org/Papers/vol2_issue5/X02510641068.pdf
[6] Jureczko, M., Madeyski, L. Towards identifying software project clusters with respect to defect prediction PROMISE. Wrocław University of Technology, Poland, 2010. – URL: http://madeyski.e-informatyka.pl/download/JureczkoMadeyski10f.pdf
[7] Liao, T.W. Clustering of time series data – a survey-Pattern recognition. Elsevier, 2005, pp. 1857-1874.
[8] Makridakis, S., Wheelwright, S.C., Hyndman, R.J. Forecasting methods and applications. John Wiley & Sons, Inc., 1998.
[9] Mockus, A. How to run empirical studies using project repositories. Avaya Labs, 2006. – URL: http://www.research.avayalabs.com /user/audris
[10] Nagappan, N., Ball, T., Zeller, A. Mining Metrics to Predict Component Failures. In Proceedings of the 28th International Conference on Software Engineering, Shanghai, China, May 20-28, 2006, ICSE'06. ACM Press New Your, NY, 2006, pp. 452-461. – URL: http://doi.acm.org/10.1145/ 1134285.1134349
[11] Novák, V., Perfilieva, I., Dvorak, A. Insight into Fuzzy Modeling. Wiley, 2016.
[12] Purao, S., Vaishnavi, V. K. Product metrics for object-oriented systems. ACM Computing Surveys 35, 2, June 2003, pp. 191-221. – URL: http://doi.acm.org/10.1145/ 857076.857090
[13] Rao, B. P., Seetharamaiah, P. Organizational Strategies and Social Interaction Influence in Software Development Effort Estimation.IOSR Journal of Computer Engineering (IOSR-JCE), Volume 16, Issue 2, Ver. XII, pp 29-40.
[14] Srinivasa, C., Radhakrishnab, V., Guru Rao, Dr.C.V. Clustering and Classification of Software Component for Efficient Component Retrieval and Building Component Reuse Libraries. In Proc. of the 2nd International Conference on Information Technology and Quantitative Management (ITQM), 2014, pp. 1044- 1050. – URL: https://pdfs.semanticscholar.org/67ac/2fc8979e84b4e57bac4ccab3af8e71f821ec.pdf
[15] Tunnell, J. W. Using Time Series Models for Defect Prediction in Software Release Planning. Central Washington University, Electronic Theses Student Scholarship and Creative Works, 2015. – URL: http://digitalcommons.cwu.edu/etd
[16] Wahyudin, D., Ramler, R., Biffl, S. A framework for Defect Prediction in Specific Software Project Contexts. In Proc. of the 3rd IFIP Central and East European Conference on Software Engineering Techniques (CEE-SET2008), Brno,Czech Republic.October13-15, 2008.
[17] Weyuker, E., Ostrand, T., Bell, R. 2008. Adapting a Fault Prediction Model to Allow Widespread Usage. In Proc. of the the International Workshop on Predictive Models in Software Engineering. PROMISE'08, Leipzig, Germany. May 12-13, 2008.
[18] Zimmermann, T., Nagappan, N., Gal,l H., Giger, E., Murphy, B. Cross-project Defect Prediction. In Proc. of the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE). Amsterdam, The Netherlands, August 24-28 2009, 2009, pp. 91-100.
[19] Zolhavarieh, S., Aghabozorgi, S., Ying, Wah Teh. A Review of Subsequence Time Series Clustering, In Scientific World Journal. Vol. 2014, Article ID 312521, 2014, 19 pp. –URL: shttp://dx.doi.org/10.1155/2014/312521.

Acknowledgments
The authors acknowledge that this paper was partially supported by the Russian Foundation of Basic Research, projects № 16-07-00535 and № 16-47-730715.