A Sequence-based Approach to Analysing and Representing Engineering Project Normality

Lei Shi[†], James A. Gopsill[‡], Linda Newnes[†], Steve Culley[†]
[†]*Dept. of Mechanical Engineering, University of Bath, UK*
[‡]*Dept. of Mechanical Engineering, University of Bristol, UK*

***Abstract* - Engineering projects are often highly complex, unique and safety critical, which can lead to the complex engineering processes and activity. To ensure the success of engineering projects, the projects often have to comply with stringent regulations and company processes. In addition, the increasing in-service lifespan of products has led to an increase in the number of re-design and maintenance projects. These are often run concurrently in a highly time-constrained and high-pressured environment, which has led to the monitoring of the sequence of engineering activity becoming difficult. This is because, the sequence of engineering activity is typically achieved through the ability of the project managers to use their knowledge, experience and constant contact with the engineers. However, the viability of the current method to manually generate and evaluate the activity plan is becoming an issue due to the increasing number and distributed nature of these projects.**

As regulatory and/or company process demands, the data relating to the project is often archived and thus, provides a wealth of potentially useful information that could be utilised in the management of current projects. Therefore, this research investigates the potential value provided by the automatic construction of past project activity sequences, and proposes analytical methods to represent the normality of project activity based on the extracted patterns from their sequences. The evaluation applies industrial data, and shows that the results generated by the proposed approach can accurately reflect the similarity and normality of the projects.

Keywords - Project Normality Identification; Sequence Construction; Sequence Analysis;

I. INTRODUCTION

Engineering projects are often highly complex, unique and safety critical, which can lead to the complex engineering processes requiring various resources and bring challenges on decision making. To ensure the success of the project products, the related project has to comply with stringent regulations (examples include the regulations for airworthiness and maritime vessels) and company processes before the work could commence.

Additionally, the in-service lifespan of products have been increased leading to projects, e.g. design or maintenance projects, being run concurrently in highly time-constrained and pressured environment [1]. The emerging trend of Product Service Systems (PSSs) whereby the cost of design and maintenance tends to be absorbed by company offering the PSS which has further compounded this effect [2] [3]. Thus, engineers are increasingly reacting to new developments across various design and maintenance projects and this is often at the loss of providing detailed project activity plans.

Therefore, to ensure the appropriate allocation of resources, compliance with regulations and company processes, and to effectively monitor project progress, project managers typically use their knowledge and experience of past projects alongside regular meetings with the engineers of the projects. Their knowledge and regular meetings provide insights into the sequence of engineering activity for each project and enable them to draw conclusions on the allocation of resources and potential issues with compliance. However, as the number of these highly complex, unique and safety critical projects are increasing, the viability of the current method to manually generate and evaluate the engineering activities is becoming an issue.

A number of studies within this context have already demonstrated the potential re-use value using past project knowledge to support current engineering projects [4] [5] [6]. Therefore, it is argued that the growing corpus of past projects could also be used to support the management of current projects. However, the corpus of past projects can be considerably large. Therefore, it is likely that the project managers would be unable to take full advantage of the knowledge that has been stored through the manual review of the corpus, and it is argued that it is impractical to do so given the time-constraints and priority of the projects. Therefore, an automated analysis of past project archives could have a greater potential for providing useful information to project managers and engineers. More specifically, the analysis could be used to generate sequences of engineering project activities and patterns, and the traces of these activities and patterns may provide dynamic indicators of project characteristics, e.g. changes of project processes, resources allocations, potential compliance issues, etc.

In order to explore the potential of using past project archives, this paper presents the automatic construction of sequences of engineering activity from a sample corpus of 236 projects. From this, the edit distance based sequence similarity measure has been proposed and used to identify typical patterns and anomalies in engineering activity. The paper then continues by presenting an approach to generating time-sliced sequences, and introducing the use of the patterns and

anomalies from the time-sliced sequences to support normality analysis and manage potential issues with project progress.

II. Related Work

Many engineering companies archive high volumes of past projects with large amount of project related data. Due to time constraints and resource limitation, the project decision makers, e.g., engineers and managers, are unable to review and understand all the past projects, thus comprehensively utilising the projects contained knowledge to support their decision making for running projects is unrealistically difficult. The remaining challenges include: (i) how to efficiently and effectively discover the essential knowledge from past projects, and (ii) how to design the reuse methodology to minimise the human interventions and time cost.

Recent research has shown that the application of data mining and machine learning is an effective way of processing large amount of data based on specified requirements whilst needing limited human effort [7] [8]. By using certain techniques, such as natural language processing, named entity recognition, pattern recognition, data classification and clustering, the essential information can be automatically identified and organised, which enables decision makers to browse, retrieve, and learn the featured projects from large collections of data. For example, the generation and visualisation of project profiles including project activity, people activity and project evolution, could assist decision makers to get an high-level overview of past projects [6]. Meanwhile, the patterns regarding file operations, sentiment and change of communication topics could facilitate the understanding on the detailed level of project planning and execution [9] [10] [11].

Time-related data is an important composition of project data, and its contained information can be used to indicate the changes of performance during the project execution. To extract useful patterns from time-related data, sequential pattern mining is required, which is an automatic approach that has been widely used in various fields, e.g., user behaviour detection, transaction data analysis, and DNA structure analysis [12] [13] [14]. In order to apply sequential pattern mining on the analysis of engineering project data, two factors need to be considered: (i) *data source*: as the data of real world projects could be heterogeneous and distributed, selecting appropriate data for the mining task is a challenge. In general, the selected data needs to contain explicit timestamps, or strong correlations with the time dimension, thus communication records, reports and log files as the typical time-related data are commonly used by the sequential pattern mining. Moreover, these types of data can be easily obtained from most projects, and their machine-readable formats enable the automatic mining process carried out over large scale dataset [15] [16], and (ii) *data representation*: in order to perform the pattern mining, the data needs to be represented in certain formats, i.e., a set of items and related timestamps. Items can have various definitions and is often determined by the attributes of the dataset and purposes of the analysis. For example, the data of customer purchase transactions could be represented by the purchased goods and time of the purchase actions [17].

In this research, the proposed approach focuses on two aspects, (i) creating a universal data representation for engineering projects; (ii) analysing the normality of projects based on the similarity between the data representations.

III. Proposed Approach

Representing engineering projects in a structured format is a challenging task due to the variety of activities that are often performed. Such activities include, generating product documentation, creating a technical report and performing a Computational Fluid Dynamics (CFD) simulation. In this approach, the projects are characterised by the output of their activities (i.e. the creation of documents, communication records and simulation models). This is applicable in this context as the companies often archive the output of their activities due to regulatory requirements. In addition, the majority of the output from the activities is digital and therefore contains time-related data such as created, modified, approved and sign-off dates. This provides the opportunity for sequences of activity to be produced.

Based on these factors, this paper takes a data-driven approach to analysing and representing engineering projects. The data required by this approach needs to contain the descriptions of project, and essential information about project planning, execution, problem solving, evaluation and feedback. Hence in this instance, technical reports and communication records are selected as the data for this research.

Figure 1 demonstrates the relations between the project and its respective activity, actions and data. As can be seen in this figure, a project is treated as an information container, which is composed of a set of activities, such as "*task planning*", "*information request*", "*problem solving*" and "*evaluation*" (a project needs to have at least one activity); each activity could have single or multiple action(s), e.g., "*information request*" may have actions "*sending emails*" and "*receiving damage reports*"; the data records project related information, and it can be used to identify the activities/actions; meanwhile, the data could be changed by the activities/actions directly or indirectly.

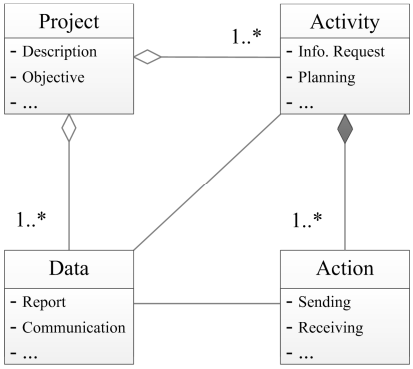


Fig. 1. Relations between the project components

A. Sequence Construction

Sequence construction is an automatic approach that aims to generate sequence of projects using time-related data. For a

given dataset, the construction requires two pre-processes, (i) corpus generation: generating a corpus of activity/action from the dataset; (ii) activity/action identification: identifying and extracting the contained activities and actions from each individual project.

In the corpus generation process, natural language processing and semantic techniques are applied to recursively analyse the entire dataset, and then a full list of contained activities/actions of the dataset is generated, which would then be converted to a corpus. In the activity/action identification process, the same analysis techniques and the generated corpus are applied. For each project, a list of activities/actions is generated, and then sorted in chronological order. This is used to construct the sequence of engineering activity for each project.

Using D, p_i and C_D to represent the dataset, project and corpus respectively, then $D = \{p_1, p_2, ..., p_i\}$ and $D \rightarrow C_D$. Let y_N^i denote an activity of p_i, then $y_N^i \in p_i$ and $y_N^i \in C_D$, $Y = \{y_1^i, y_2^i, ...y_N^i\}$, where Y is the *activity set* of p_i. Let $x_{M,N}^i$ denote an action of y_N^i, then $x_{M,N}^i \in y_N^i$ and $x_{M,N}^i \in C_D$, $X = \{x_{1,N}^i, x_{2,N}^i, ..., x_{M,N}^i\}$, where X is the *action set* of y_N^i.

In this paper, actions are the atomic components of the sequence. For different projects, weights can be applied to the various types of action. This enables certain actions to be emphasised or less considered according to the project characteristics. Consequently, the sequence of project is defined as,

$$s_i = \sum_{N=1}^{n} \sum_{M=1}^{m} \omega_j \cdot x_{M,N}^i \tag{1}$$

where ω_j is the weight of the action $x_{M,N}^i$, and $0 \le \omega_j \le 1$; m is the total number of actions of one activity, and n is the total number of activities of the project.

Figure 2 below shows an example of some project sequences that are generated from a real dataset. For simplicity, the displayed sequences have an identical length, which is equal to 15. Each row in the figure represents one project sequence with the project ID, and the entry of sequence Tx indicates the type of the action.

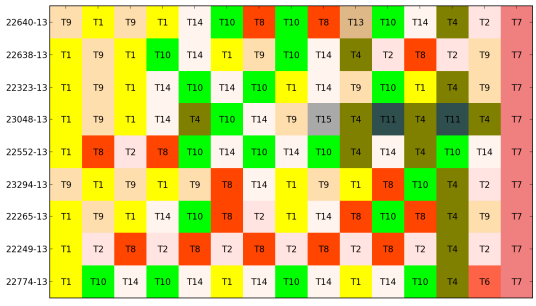


Fig. 2. Example of project sequences

B. *Similarity Measure*

The sequence construction process provides a consistent representation of engineering activity and enables the computation of a similarity measure. The proposed method for determining the similarity of sequences is based on the *edit distance*, which is a typical method that has been widely used in various fields including bioinformatics [18], machine translation [19] and information extraction [20]. Edit distance involves three types of operation in its calculation (i.e., insertion, deletion and substitution), and the distance between two sequences is defined as the minimum operation number of converting one sequence to another.

The calculation of the edit distance between s_i and s_j can be recursively defined as,

$$d(s_i, s_j) = \min \begin{cases} c(s_i, s_j) + d(s_{i-1}, s_{j-1}) \\ c(s_i, \varepsilon) + d(s_{i-1}, s_j) \\ c(\varepsilon, s_j) + d(s_i, s_{j-1}) \end{cases} \tag{2}$$

where c is cost function; $d(\varepsilon, \varepsilon) = 0$, and it means the edit distance between identical sequences is zero.

For given sequences s_i and s_j, let $|s_i|$ and $|s_j|$ denote the sequence length respectively, then the similarity between them can be calculated using the following equation,

$$sim(s_i, s_j) = \begin{cases} 1 - \dfrac{d(s_i, s_j)}{\min(|s_i|, |s_j|)}, & \text{if } k_1 \le d(s_i, s_j) \le k_2 \cdot \min(|s_i|, |s_j|) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where $\min(|s_i|, |s_j|)$ is the minimum length of s_i and s_j; k_1 and k_2 are thresholds, where $0 \le k_1 \le k_2 \cdot \min(|s_i|, |s_j|)$ and $0 \le k_2 \le 1$. The use of k is to eliminate the bias caused by sequence lengths with relatively larger variations. For example, when k_2 is 0.5, the sequence similarity of s_i and s_j will be considered as 0, if the edit distance between them is greater than the half-length of the shorter one.

Using the equation (2) and (3), the results of sequence similarity measure from part of the example in Figure 2 are shown in Table I. It can be seen that even if two projects contain common actions, their similarity can still be considered as 0, as long as the edit distance between them is greater than the threshold k_1, e.g., 22249 and 23048,

TABLE I. SEQUENCE SIMILARITIES

	22638	**23048**	**22552**	**22249**	**22265**
22638	-	0.35	0.29	0.21	0.43
23048	0.35	-	0.29	0	0.36
22552	0.29	0.29	-	0.29	0.21
22249	0.21	0	0.29	-	0.36
22265	0.43	0.36	0.21	0.36	-

C. Time-sliced Sequence

Projects could have different similarity values at different stages. For example, two projects could have a high similarity at their early stage, and then the similarity could become lower or even zero at the later stage. To measure this *temporal similarity*, the project sequences have been sliced into sub-sequences according to specified time intervals. The time interval, called *step interval*, is used to control the length of sub-sequences.

For a given *step interval*, the time-sliced sequences can be presented as,

$$s_i^t = \sum_{N=1}^{n} \sum_{M=1}^{m} (x_{M,N}^i)_t, t \in \tau \quad (4)$$

where $t = 0,1,\ldots$; τ represents one *step interval* or collection of intervals over the project timeline, where $\tau = \{t_i, i \in N\}$.

Given a collection of sequences S, with a specified *step interval* number n, the construction process of time-sliced sequences recursively uses equation (4), and the pseudo-code of the process is as follows (Table II):

TABLE II. PSEUDO-CODE OF GENERATING TIME-SLICED SEQUENCES

Algorithm: *generating time - sliced sequences*

input S, n
$length_{\max} \leftarrow 0$
for all s_i **in** S:
  comment: *find the maximum sequence length*
  if $|\, s_i \,| > length_{\max}$:
    $length_{\max} \leftarrow |\, s_i \,|$
  end if
end for
return $length_{\max}$
if $length_{\max} \geq n$:
  comment: *determin the sub - sequence length*
  $|\, interval \,| \leftarrow length_{\max} \,/\, n$
end if
for all s_i **in** S:
  $s_i^t \leftarrow \varnothing$
  $s_i^{'} \leftarrow s_i + [0] * (length_{\max} - |\, s_i \,|)$
  comment: *slice the sequences*
  for $j = 1,\ldots,n$:
    $s_i^j \leftarrow s_i^{'}[0 : j * |\, interva \,|]$
    $(s_i^t).append(s_i^j)$
  end for
  $(S^{'}).append(s_i^t)$
end for
return $S^{'}$

For example, if $n = 5$, the sequence will be sliced into 5 sub-sequences, and each one represents the involved actions within its related time interval, e.g., three actions (20% of the project progress). Figure 3 shows the results of sliced sequences using the example shown in Figure 2.

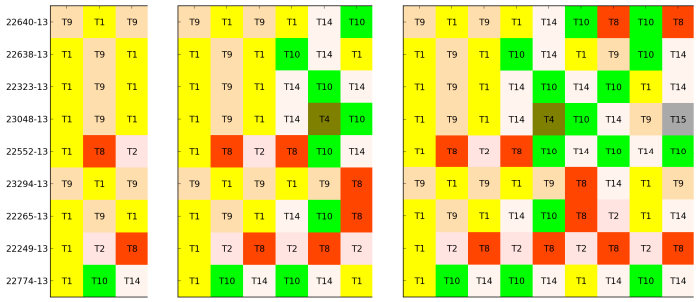


Fig. 3. Time-sliced sequences

The similarity between sub-sequences is calculated using equation (3). Table III, IV and V show the sequence similarity results regarding step interval 1-3 (20% project progress), 1-6 (40% project progress) and 1-9 (60% project progress) respectively.

TABLE III. STEP INTERVAL 1-3 (20% PROGRESS)

	22638	23048	22552	22249	22265
22638	-	1	0.33	0.33	0.50
23048	1	-	0.33	0.33	1
22552	0.33	0.33	-	0.33	0.33
22249	0.33	0.33	0.33	-	0.33
22265	0.50	1	0.33	0.33	-

TABLE IV. STEP INTERVAL 1-6 (40% PROGRESS)

	22638	23048	22552	22249	22265
22638	-	0.50	0.33	0	0.50
23048	0.50	-	0	0	0.67
22552	0.33	0	-	0.50	0.33
22249	0	0	0.50	-	0
22265	0.50	0.67	0.33	0	-

TABLE V. STEP INTERVAL 1-9 (60% PROGRESS)

	22638	23048	22552	22249	22265
22638	-	0.44	0.33	0	0.44
23048	0.44	-	0	0	0.67
22552	0.33	0	-	0.33	0.22
22249	0	0	0.33	-	0.22
22265	0.44	0.67	0.22	0.22	-

The results reveal the temporal sequence similarity between projects at different stages, e.g., the similarity between 22638 and 22265 equals 0.50 in both 20% and 40% progress, and then decreased to 0.44 in 60% progress; the similarity between 22249 and 22265 equals 0.33 in 20% process, which decreased to 0 in 40% progress, and then increased to 0.22 in 60% progress. The main reason is the project process and its included actions may need changes during the project execution in order to fulfil the changes of other factors such as time, environment and resource. In general, the degree of

similarity changes tends to decrease, and then become more stabilised along with the project's progressing.

D. Normality Analysis

Sequence similarity is the key element to measure the *project normality*. For example, a project has low similarity with others could indicate its process is more likely to be unique, which means its normality degree should also be low. Generally, the normality degree is considered in proportion to the similarity value. To analyse the normality, two strategies are defined here: (i) if a project is similar to the majority of projects, its normality degree will be considered as high; (ii) if a project is similar to a known "normal" project, its normality degree will be determined by the normality degree of the known project, and the similarity between the two projects.

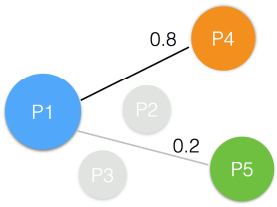
To evaluate the normality degree of multiple projects, a network-based visualisation, called *project normality graph*, is proposed. In this visualisation, a vertex presents a project, and the edge between vertices indicates the similarity between them is greater than 0. The size of a vertex is in proportion to its degree value, which is equal to the number of its connected vertices. The higher degree of a vertex indicates the higher volume of similar projects that the vertex indicated project would have. The weight of an edge is in proportion to the similarity value between its connected projects. The higher edge weight indicates the higher similarity value the edge-connected projects would have.

Using $G=(V,E)$ denote a normality graph, where V and E is the set of vertex and edge. Let v_i denote the vertex of project p_i, and e_j denote the edge of v_i, then the normality degree of p_i is defined as,

$$nor(p_i)=\omega_1\cdot\deg(v_i)+w_2\cdot\sum_{j=1,e_j\in E}^{\deg(v_i)}e_j \qquad (5)$$

where w_1 and w_2 are adjustable weights, and $w_1,w_2\geq 0$, $w_1+w_2=1$. For example, if $w_1=w_2$, the normality analysis will equally consider the degree of the project vertex in the graph and the cumulative weight of the vertex connected edges.

Figure 4 shows an example of the visualisation. Using equation (5), when $w_1=w_2$, P1 is a project with highest normality degree, i.e., $nor(p_1)=1.5$; P4 has higher normality degree than P5, as it has a higher similarity with P1 than P5, i.e., $nor(p_4)=0.9$ and $nor(p_5)=0.6$; P2 and P3 have the lowest normality degree, as they do not have any similar project connected, i.e., $nor(p_2)=nor(p_3)=0$. According to this example, the normality of individual project can be efficiently calculated by using the proposed approach.



Sequence Similarity

	P1	P4	P5
P1	1	0.8	0.2
P4	0.8	1	0
P5	0.2	0	1

Degree

P1	2
P2	0
P3	0
P4	1
P5	1

Fig. 4. Example of normality visualisation

IV. Experimental Study

In this experimental study, 236 in-service projects from a leading aerospace company have been used to investigate the potential of the proposed approach. The project data includes technical reports and communications records. These contain detailed information about the project, such as descriptions, objectives, processes, problems, solutions and evaluations. All the actions and their timestamps can be identified from the data by using natural language processing and semantic techniques.

The steps of this study include, (i) pre-analysis: generating action corpus from the dataset, and action list of each project; (ii) sequence construction: converting data to sequences using the corpus and action lists, (iii) sequence slicing: converting each sequence to 10 sub-sequences, and each interval represents 10% project progress, and (iv) normality analysis and representation: using the project normality graph to visualise and analyse the normality of projects.

To evaluate the project normality using the graph, *graph density* and *average degree* are applied,

$$density_G=\frac{2\cdot|E|}{|V|\cdot(|V|-1)}$$

$$deg_{ave}=\frac{2\cdot|E|}{|V|}$$

where $|E|$ and $|V|$ is the total number of edges and vertices of graph G. Graph density measures the edge quantity of the graph, which is used to identify the overall normality changes of the dataset. Average degree measures the number of connected vertices in the graph, and it is used to identify the normality changes between the projects.

Table VI shows the changes of graph density and average degree of the projects over the progress from 10% to 90%. It can be seen that the graph density and average degree decrease rapidly at the early stage, and then become more stable from the middle to end stage. The detailed visualisation regarding the changes is shown in Figure 5, 6 and 7.

TABLE VI. Average Degree and Graph Density in Different Steps

	10%	30%	50%	70%	90%
Graph Density	0.36	0.14	0.09	0.08	0.08
Ave. Degree	117.78	41.66	36.59	31.96	31.63

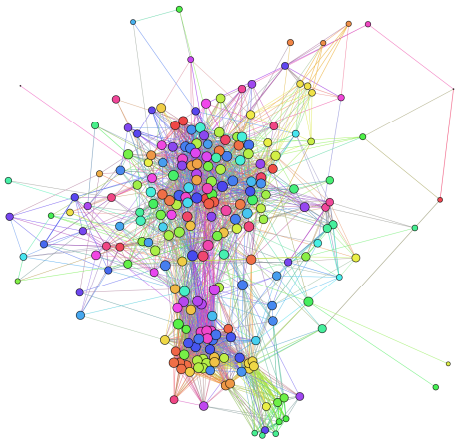


Fig. 5. Project Normality Graph, progress=10%

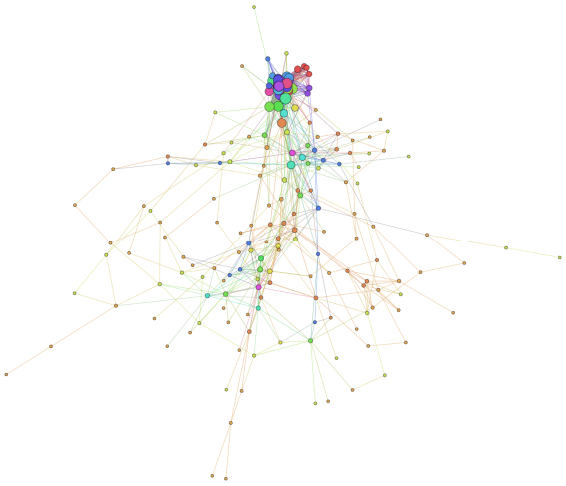


Fig. 6. Project Normality Graph, progress=30%

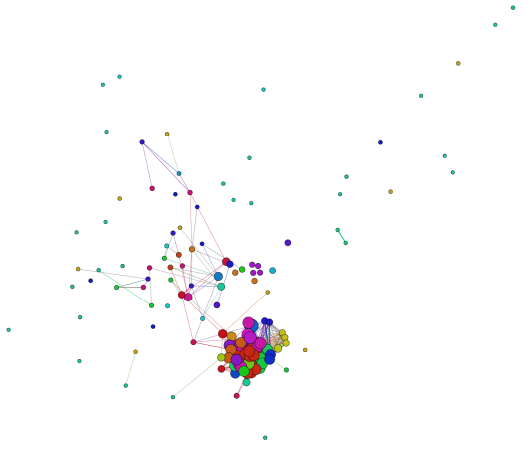


Fig. 7. Project Normality Graph, progress=50%

In Figure 5, most projects have relatively high normality degrees, i.e., the density of normality graph with 10% progress is 0.36, and the average degree of the dataset is 117.78 (see Table VI). According to this, one project could be similar to 49.91% of total projects in the dataset. The reason is most projects at their initial stage mainly aims to obtain information, thus the processes only involve limited types of actions, such as "*information request*", "*receiving damage report X*", "*conforming damage location Y*", etc. The involved actions are limited, means the process of projects at this stage could be less various, so that the normality degrees could be high.

However, some projects may still have low normality degree at the initial stage, which could be caused by the following reasons, (i) they are the projects that barely or never appeared in the past, thus their processes are likely to be different comparing with others, (ii) they are complex projects with relatively higher uncertainty on execution, thus their processes might be specifically designed or optimised at the early stage; (iii) their execution processes might be already effected by some unexpected factors, e.g., errors or equipment failures. By using the project normality graph, the normality changes caused by the listed reasons can be explicitly presented, thus the decision makers could be made aware of the changes and accordingly pay their attention to those relevant projects.

In Figure 6, the normality degree of projects is decreased, i.e., the density of normality graph is decreased to 0.14 (-61.11%), and the average degree of the dataset is decreased to 41.66 (-64.63%). After the initial stage, more types of actions could be involved in the processes, e.g., "*performing stress analysis*" and "*creating stress analysis report*". With different orders and combinations, the newly involved actions cause divergences on the structure of project sequences, and then decrease the normality degree between the projects.

In Figure7, the normality degree of certain projects is further decreased, i.e., the density of normality graph is decreased to 0.09 (-35.71%), and the average degree of the dataset is decreased to 36.59 (-12.17%). However, the trend of decrease becomes more gradual from the middle stage. According to Table VI, the normality degree is fairly stable from 50% project progress to the end, e.g., from progress 50% to 70%, the density of normality graph is decreased by 11.11%, and the average degree of the projects is decreased by 20.85%; the density of normality graph remains the same value, and the average degree of the projects is only decreased by 1.03%, from progress 70% to 90%. The rationale is that most projects do not massively change their processes and actions from the middle stage, as the project plan has been set at the earlier stage, therefore the projects should be steadily executed following the plan.

V. CONCLUSIONS

Engineering projects are often highly complex, unique and safety critical, and performed within an environment that is becoming increasingly distributed with many more project running concurrently. This leads to increased challenges on project managers and engineers in being able to effectively plan activity, allocate resources, and ensure compliance and monitor progress.

In order to provide effective indicators to support decision makers to understand the normality of engineering projects,

this paper proposed an automatic approach to construct engineering activity sequences for projects and determine its relative normality. The sequence of a project is constructed using time-related project data, and the similarity and normality between projects can then be measured using their sequence representations. To perform temporal normality analysis, the method of generating time-sliced sequence is proposed, which splits a single sequence to sub-sequences by using adjustable step intervals. Furthermore, a network-based visualisation is introduced, which is an accurate and understandable way to analyse and represent the changes of project normality over the project progress. The experimental study uses a collection of industrial data, and the result shows the proposed approach can identify and represent the normality.

Further work is being undertaken to increase the level of analysis and will include performing more micro analysis and improving the performance of real-time normality monitoring.

ACKNOWLEDGMENT

The research reported in this paper is funded by Engineering and Physical Sciences Research Council (EP/K014196/1). The authors would like to thank the industrial collaborators and their engineers for their input and support on this project.

REFERENCES

[1] S. Lee, Y.-S. Ma, G. Thimm, and J. Verstraeten, "Product lifecycle management in aviation maintenance, repair and overhaul," *Computers in Industry*, vol. 59, pp. 296-303, 2008.

[2] O. Mont, "Clarifying the concept of product–service system," *Journal of cleaner production*, vol. 10, pp. 237-245, 2002.

[3] J. A. Erkoyuncu, R. Roy, E. Shehab, and P. Wardle, "Uncertainty challenges in service cost estimation for product-service systems in the aerospace and defence industries," in *Proceedings of the 19th CIRP Design Conference–Competitive Design*, 2009.

[4] Y. Xie, S. J. Culley, and F. Weber, "Applying context to organize unstructured information in aerospace industry," in *Internal conference on engineering design (ICED' 11)*, Copenhagen, Demark, 2011.

[5] E. Carey, S. Culley, H. McAlpine, F. Weber, and Z. Xie, "Key issues in the take-up of knowledge management interventions in engineering design," in *DS 70: Proceedings of DESIGN 2012, the 12th International Design Conference*, Dubrovnik, Croatia, 2012.

[6] L. Shi, J. Gopsill, C. Snider, S. Jones, L. Newnes, and S. Culley, "Towards identifying patterns in engineering documents to aid project planning," presented at the *DESIGN 2014: 13th International Design Conference*, Dubrovnik, Croatia, 2014.

[7] M. Polczynski and A. Kochanski, "Knowledge Discovery and Analysis in Manufacturing," *Quality Engineering*, vol. 22, pp. 169-181, 2010.

[8] A. K. Choudhary, J. A. Harding, and M. K. Tiwari, "Data mining in manufacturing: a review based on the kind of knowledge," *Journal of Intelligent Manufacturing*, vol. 20, pp. 501-521, 2009/10/01 2009.

[9] S. Jones, S. Payne, B. Hicks, and L. Watts, "Visualization of Heterogeneous Text Data in Collaborative Engineering Projects," presented at *The 3rd IEEE Workshop on Interactive Visual Text Analytics*, Atlanta, 2013.

[10] J. A. Gopsill, S. J. Payne, and B. J. Hicks, "An Exploratory Study into Automated Real-Time Categorisation of Engineering E-Mail," in *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, 2013, pp. 4806-4811.

[11] B. Hicks, "The language of collaborative engineering projects," in *DS 75-6: Proceedings of the 19th International Conference on Engineering Design (ICED13), Design for Harmonies, Vol. 6: Design Information and Knowledge*, Seoul, Korea, 2013.

[12] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection for discrete sequences: A survey," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 24, pp. 823-839, 2012.

[13] H. K. Kwan and S. B. Arniker, "Numerical representation of DNA sequences," in *Electro/Information Technology, 2009. eit'09. IEEE International Conference on*, 2009, pp. 307-310.

[14] Y.-L. Chen, M.-H. Kuo, S.-Y. Wu, and K. Tang, "Discovering recency, frequency, and monetary (RFM) sequential patterns from customers' purchasing data," *Electronic Commerce Research and Applications*, vol. 8, pp. 241-251, 2009.

[15] L. Grace, V. Maheswari, and D. Nagamalai, "Analysis of web logs and web user in web mining," *arXiv preprint arXiv:1101.5668*, 2011.

[16] D. Perera, J. Kay, I. Koprinska, K. Yacef, and O. R. Zaïane, "Clustering and sequential pattern mining of online collaborative learning data," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 21, pp. 759-772, 2009.

[17] F. Masseglia, M. Teisseire, and P. Poncelet, "Sequential Pattern Mining," ed, 2009.

[18] D. Sokol, G. Benson, and J. Tojeira, "Tandem repeats over the edit distance," *Bioinformatics*, vol. 23, pp. e30-e35, 2007.

[19] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A study of translation edit rate with targeted human annotation," in *Proceedings of association for machine translation in the Americas*, 2006, pp. 223-231.

[20] Y. Kim, J. Park, T. Kim, and J. Choi, "Web information extraction by HTML tree edit distance matching," in *Convergence Information Technology, 2007. International Conference on*, 2007, pp. 2455-2460.