TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 18, No. 6, December 2020, pp. 2939~2947
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v18i6.13381

Adopted topic modeling for business process and software component conformity checking

Adhatus Solichah Ahmadiyah, Riyanarto Sarno, Fony Revindasari
Department of Informatics, Institut Teknologi Sepuluh Nopember, Indonesia

Article Info

Article history:

Received Jun 21, 2019
Revised Jun 20,2020
Accepted Jul 9, 2020

Keywords:

Business process
Document clustering
Software component
Topic modeling
Traceabilityv

ABSTRACT

Business processes and software components, especially class diagrams, have a firm connection. Considering software components support the business process in providing an excellent product and service. Besides, business process changes affect on software component design. One of them usually appears on the label or name of the software component or business process. Sometimes, a related business process and software component appears in the different label but the same meaning rather than using the same label. This situation is problematic when there are many changes to be made, in which the software component's modifying process becomes quite long. Therefore, the software maintainers should obtain an efficient procedure to shorten the modifying process. One solution is by using conformity checking, which helps the software maintainers know which software component is related to a specific business process. This paper compared two leading topic modeling techniques, namely probabilistic latent semantic analysis (PLSA) and latent Dirichlet allocation (LDA), to determine which one has a better performance for process traceability.

Corresponding Author:

Adhatus Solichah Ahmadiyah,
Department of Informatics,
Institut Teknologi Sepuluh November,
Kampus ITS Sukolilo, Raya ITS St., Surabaya, Jawa Timur 60111, (031) 5994251, Indonesia.
Email: adhatus@if.its.ac.id

1. INTRODUCTION

Each company or organization uses a business process that has a vital role in supporting performance and providing the best service and product [1]. This fact escalates the growth of researches in this field in aligning business processes and software systems [2-10]. At the same time, [11] detects fraudulence in a parallel business process. A business process is a bag of activities that interconnected with their tasks [12]. However, the company or organization can change one of the processes or all in the business process. Change in operation standard or supporting system is one of the reasons why the company changed its business process. Small changes in the business process have a significant impact on the operating standard or supporting system [13]. It happens because the business process and supporting system are interconnected.

In this research, the supporting system has a software component that contains classes in the form of a class diagram. Its class name and class function are used for reference. Meanwhile, the reference to the business process refers to its activity name. Those three references comprise a hint to identify the relationship of a specific software component to a particular business process. The straightforward way to detect the connection between them is by looking at the similarity between activity name in the business process and class or function name in the software component. In fact, business processes and software components often use a different name but have the same meaning [13, 14]. Although the business process and its software component use a different name, we can still explore the similarity between them using traceability. Traceability can help software maintainers understand which software component related to the business process. Clustering documents are obtainable to discover the traceability process [15].

In our previous studies, [16] and [17] were successfully used two leading information retrieval methods to retrieve traceability of business processes and software components. [16] used the probabilistic latent semantic analysis (PLSA) [18] method, while [17] used the latent Dirichlet allocation (LDA) [19] method. Even though each individual performed well, their usage for a more complicated case has not proved. In this study, we investigate and compare the performance of both mentioned methods to retrieve the traceability between software components and the business process in a more extensive case study.

2. LITERATURE STUDY

In [20], Aversano et al stated that traceability matrix aligned software components and business process. Earlier, Marcus and Maletic [21] conducted traceability between documentation and software source code using latent semantic indexing (LSI). Meanwhile, Pessiot et al proposed to use unsupervised dimensionality reduction methods in the document clustering [22]. The authors proposed the extension of probabilistic latent semantic analysis (PLSA) model in performing word and document clustering concurrently in a single aspect model. The researches as mentioned earlier need some clusterings and similarity measurements. In [23], Al-Anazi et al compared the performance of some clustering and similarity measurement methods. Some researchers employed latent Dirichlet allocation (LDA) to perform traceability on software engineering problems. As used by [19], LDA was employed to counter some IR problems. In [24], the author used LDA to mining the concepts in software implementation code. While in [25], LDA formed traceability links of the software documentation artifacts during the development process.

3. BUSINESS PROCESS AND SOFTWARE COMPONENT

We focused on aligning the operational integration view of the strategic alignment model (SAM) [26]. A business process is a group of activities which are interconnected to produce a product or service. While, software component consists of classes and each class contains functions. In a software development process, a sequence diagram describes the business process. Vice versa, each business process is supported by IT-driven software components to run the tasks [20]. Figure 1 illustrates the relationship between business processes and software components.

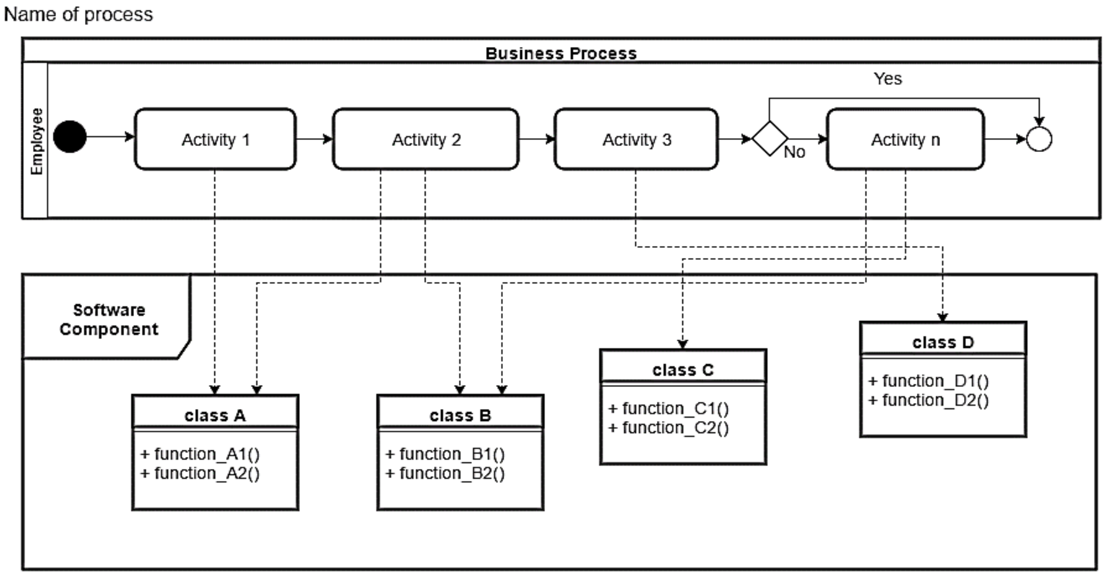

Figure 1. Relationship between business process and software component in general

Figure 1 explained that a process of business are composed of interconnected activities. Each activity runs under the order. As shown at the bottom of the business process block, a software component is associated with it, connected with the dash lines. Each activity runs following the order. At the bottom of the business process block, there is a software component that is associated with it, shown with the dash lines. The software component holds many classes to support its job. Although each class has a specific notion, it accessible for other business process activities. Each activity may be related to one class or more subject to its needs. Besides, the name used in business processes and software components are often different but have the same meaning.

As illustrated in Figure 2, the registration process consists of four consecutive activities starting from register activity to creating a new account activity. The registration activity carried out the enrollment process. The software components associated with this activity are "registrationForm" and "accountControl". The name on the different activities associated with the related name of the class. Although the names are different the meaning is the same. It is sometimes problematic because of differences in the name. Thus, the similarity is required to solve this problem.

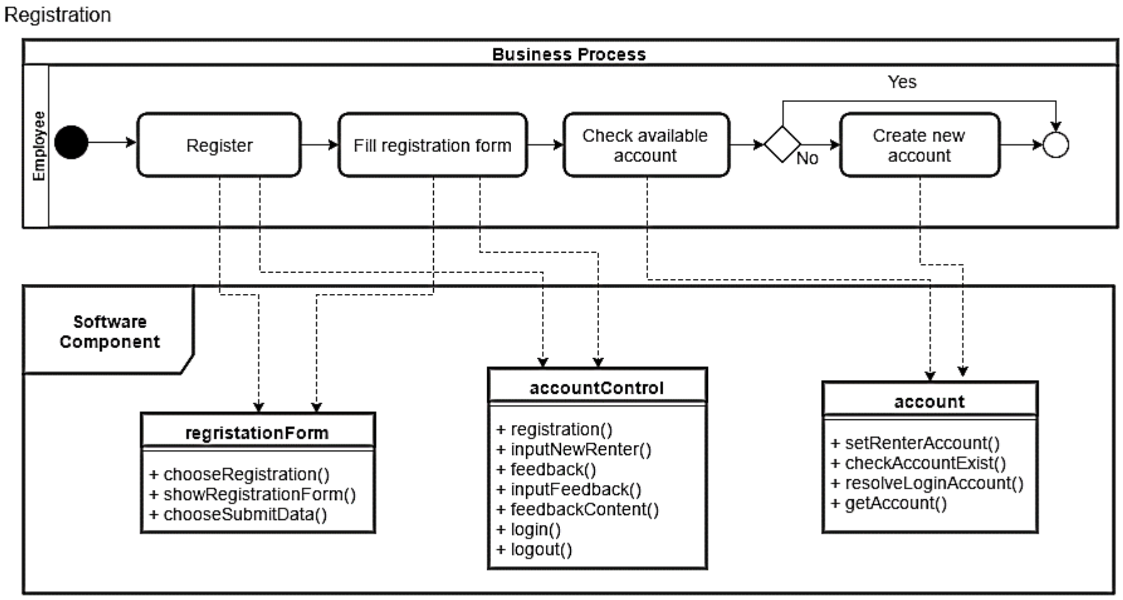

Figure 2. Business process and software component visual traceability

4. METHODOLOGY

This section describes the calculation process using PLSA and LDA and similarity process. The execution from the preprocessing phase to the comparison method phase can be seen in Figure 3. A sequence of activities are modeled as a business process description. Meanwhile, each method and functions are modeled as software component documentation. Each activity, method, and function are inserted into documents. Then, the documents are processed into the text preprocessing phase.

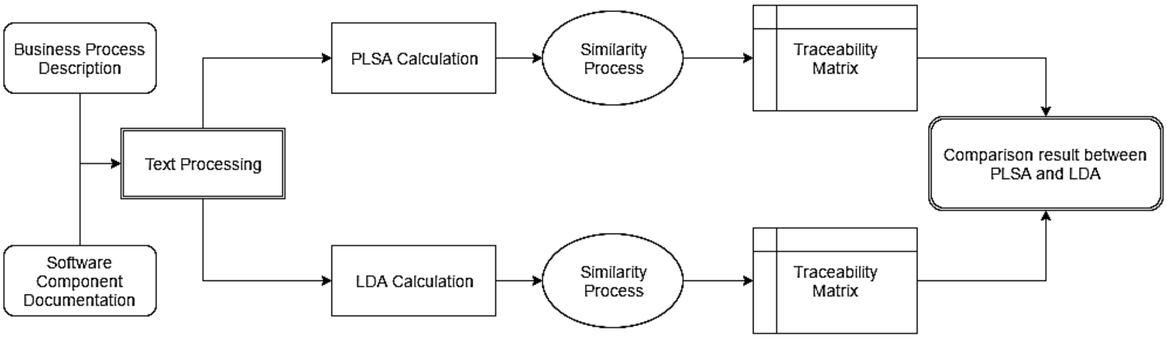

Figure 3. Comparison process phase

Text preprocessing phase operates tokenization, stopword removal, and stemming. Tokenization splits the sentence into words in a document called tokens [27]. Stopword removal eliminates trivial and meaningless words, for example, preposition and conjunction. Lastly, stemming removes prefix and form the basic word [28]. After the preprocessing phase, the next phase is the calculation process using PLSA and LDA. Every word or term in the documents has a probability value and grouped into certain topics. However, the similarity between documents and topics are unknown. Then, the process of similarity is required to determine similarity by using cosine similarity. Next step is the process traceability matrix. After all the process finish, the last phase is performing the comparison between the two methods, i.e., PLSA and LDA.

4.1. Probabilistic latent semantic analysis

Lexical and Semantic analysis are two common models to analyze a text document. Lexical analysis is textual analysis that focuses on the terms in a document. Meanwhile, semantic or contextual analysis is an analysis based on the meaning of words in a document. Each document has a bunch of words (terms) and keywords in which each keyword is a term that represents the document. The keyword in the PLSA method referred to the aspect model, a hidden variable to find the same pattern in each document. Since the words having the same frequency in the same document are grouped into one topic. Each text in the document is divided into words with more than one meaning and words that have the same meaning. Next, words with same frequency are calculated. For each word in the document with the same meaning, it has the same frequency value [29]. Probabilistic latent semantic analysis (PLSA) and latent semantic analysis (LSA) are dedicated to analyze the relationship between documents and terms or words. The difference is in how the calculations and processes. The process of the LSA involves calculating term frequency and formation matrix using singular value decomposition (SVD) calculation. However, in PLSA the relationship between documents and word are bridged by topics. The other difference is that PLSA give a probability value on documents, topics, and words.

The initial process in PLSA is giving random probability value on documents, topics, and words. In the PLSA method, the document (d) is given a probability value called the document probability P(d). Then, the topic is formed based on the probability of the previous document called probability topic of document P(z|d). After that, the word is formed based on the previous probability called probability word of topic P(w|z). The calculation word in the document using joint probability is described in (1).

$$P\left(d_i, w_j\right) = P(d_i)P\left(w_j, d_i\right),\ P\left(w_j, d_i\right) = \sum_{k=1}^{K} P\left(w_j \middle| z_k\right)P(z_k|d_i) \quad (1)$$

In PLSA, it involves an additional step namely expectation maximization (EM step). The probability of latent variables (topics) within the document and words is computed using this EM step. This calculation is done repeatedly until it reaches the number of iterations to optimize the fit of the data with a probabilistic model and find the estimated maximum likelihood parameter.

The EM step begins with the E step computation. It is used to find the probability of latent variables in the document and the word as shown in (2). Then, it is continued with the M step computation. M step updates the value of the parameter in E step as shown in (3) and (4). EM iteration process is carried out iteratively until it reaches the number of iterations to achieve optimal parameter values. The more optimal the parameter values, the more fit the data and probabilistic models.

$$P\left(z_k \middle| d_i, w_j\right) = \frac{P\left(W_j \middle| Z_k\right)P(z_k|d_i)}{\sum_{k=1}^{K} P(w_j|z_l)P(Z_l|d_i)} \quad (2)$$

$$P\left(w_j \middle| z_k\right) = \frac{\sum_{i=1}^{N} n\left(d_i \middle| W_j\right)P(z_k|d_i,w_j)}{\sum_{m=1}^{M}\sum_{i=1}^{N} n(d_i|w_m)P(Z_k|d_i, W_m)} \quad (3)$$

$$P(z_k|d_i) = \frac{\sum_{j=1}^{N} n\left(d_i \middle| W_j\right)P(z_k|d_i,w_j)}{n\ (d_i)} \quad (4)$$

The final probability value is then used to calculate its similarity. The similarity value represents the proximity of the business process and software components.

4.2. Latent Dirichlet allocation

Latent Dirichlet allocation or LDA is a probabilistic generative model. It works on a set of documents to determine topic structure contained. In LDA, we compute the probability distribution of words to form topics. Then, we classify multiple topics into documents. In LDA, the Dirichlet prior distribution extracts the probability distribution between documents and topics. Meanwhile, the Polynomial distribution derives the probability distribution between topics and words.

Figure 4 shows the generative model of LDA. The generation process of LDA is described as follows. First, it determines a topic distribution for the document (θ). Then it chooses a topic from the topic distribution for each word in the document (z). Then, it chooses a word using the determined topic-specific word distribution (Φ). α and β are Dirichlet parameter, estimated by user. As seen in the graphical model, W_i or word is the only observable variable, and the rest such as z, Φ, θ are hidden variables or usually called latent variables. Parameter estimation is required in LDA to approximate the posterior distribution of the hidden variable z (topic).

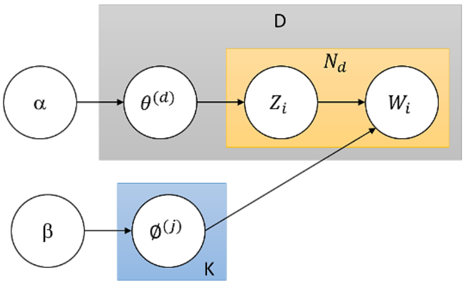

Figure 4. LDA generation model

One way to estimate the posterior distribution is by using Gibbs Sampling. Gibbs sampling is one of Markov chain Monte Carlo (MCMC) simulation that is a simple algorithm to estimate inference in high dimensional models like LDA and easy to implement. The generative algorithm for the LDA model using Gibbs Sampling is performed in two phases, i.e., initialization and Gibbs sampling, as follows:

4.2.1. Initialization phase

For every documents m in set D:
For every words w in document m:
- Draw sample topic z randomly ~ Mult(1/K).
- Increment $n_{-i,t}^{(w)}$, $n_{-i,t}^{(.)}$, $n_{-i,t}^{(m_i)}$, and $n_{-i}^{m_i}$.

4.2.2. Gibbs sampling phase

For each iteration do:
For all documents m in set D:
For all words w in document m:
- Cancel the current value of z in w.
- Decrement $n_{-i,t}^{(w)}$, $n_{-i,t}^{(.)}$, $n_{-i,t}^{(m_i)}$, and $n_{-i}^{m_i}$
- For topic j = 0 to K-1:
- Calculate $p(z_i = t|z_{-i}\,, w_i)$
- Draw new topic z ~ $p(z_i = t|z_{-i}\,, w_i)$
- Assign new topic z to word w
- Increment $n_{-i,t}^{(w)}$, $n_{-i,t}^{(.)}$, $n_{-i,t}^{(m_i)}$, and $n_{-i}^{m_i}$.

Next, (5) is used to approximate the posterior distribution $p(z_i = t|z_{-i}\,, w_i)$.

$$(z_i = t|z_{-i}\,, w_i) \propto \frac{n_{-i,j}^{(w)}+\beta}{n_{-i,t}^{(.)}+W\beta}\frac{n_{-i,t}^{(m_i)}+\alpha}{n_{-i}^{m_i}+T\alpha} \qquad (5)$$

$n_{-i,t}^{(w)}$ represents the number of word w_i put to topic t. while $n_{-i,t}^{(.)}$ represents the total number of word put to topic t. $n_{-i,t}^{(m_i)}$ is the number of the word put to topic t in document m_i and $n_{-i}^{m_i}$ is the total number of the word within document m_i. After the estimation process using Gibbs Sampling, the probability distribution of topics over documents, $\vartheta^{(d_i)}$, is calculated using (6) and the probability distribution of word over a topic for each word in the vocabulary, φ_j^{w}, is calculated using (7).

$$\vartheta^{(d_i)} = \frac{n_{-i,j}^{(d_i)}+\alpha}{n_{-i}^{d_i}+T\alpha} \qquad (6)$$

$$\varphi_j^w = \frac{n_{-i,j}^{(w)}+\beta}{n_{-i,j}^{(\cdot)}+W\beta} \tag{7}$$

4.3. Cosine similarity

Similarity measurement denotes how similar one document to the other. The measurement is described as distance and dimensions representing the features of the object. The similarity comes in the range of zero to one, each represents not similar and precisely same, respectively. The smaller the distance between the two documents, the higher the similarity between them. Vice versa, the greater the distance between the two documents, the lower the similarity between those documents [21]. There are common types of measurements for text mining: Euclidean distance, Manhattan distance [30], Jaccard similarity, and cosine similarity. Among those, cosine Similarity is popular because of its efficient evaluation. See (8) for details.

$$\cos(\theta) = \frac{A.B}{||A||||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{8}$$

After completing the cosine similarity calculation, the next step is to describe the traceability matrix. Traceability matrix values are retrieved and relevant value. Here, we use recall and precision measurements [20] as described in (9) and (10) to calculate the accuracy of the method. Recall value is obtained from retrieved and relevant data divided by the value that matches the relevant data. While the Precision value is obtained from retrieved and relevant data divided by the value in getting retrieved data.

$$Recall = \frac{\sum_i \#(Relevant_i \cap Retrieved_i)}{\sum_i Relevant_i}\ \% \tag{9}$$

$$Precision = \frac{\sum_i \#(Relevant_i \cap Retrieved_i)}{\sum_i Retrieved_i}\ \% \tag{10}$$

5. RESULTS AND ANALYSIS

In this research, the dataset carries 14 processes, 49 activities, and 29 classes. The dataset consists of a pair of sequence diagram and class diagram. In this research, we incorporate a medium scale dataset which is more comprehensive than the one used by [16] and [17]. The dataset was originally written in Indonesian then we translated it into English. Table 1 tabulates a list of business process activities. Each activity has "BP" identifier. While, each class on software components has "SC" identifier as shown in Figure 5. The identifiers promotes the traceability process and finding relevant value. PLSA and LDA methods generated the same final result in terms of precision and recall values. The result was obtained from a probabilistic value, furthermore the similarity was produced using cosine similarity. Since PLSA and LDA used a different calculation approach, in which the PLSA used expectation maximization for optimizing the probability value and the LDA used Gibbs Sampling, it affected the distribution of the document into topics.

Table 1. List of business process

Identifier	Name of activities	Identifier	Name of activities
BP 1	choose type field	BP 26	save transaction detail
BP 2	view schedule lits	BP 27	get payment and print payment
BP 3	choose schedule detail	BP 28	view report managed
BP 4	register	BP 29	choose field rental income report
BP 5	fill registration form	BP 30	choose field rental income report based transaction date
BP 6	check available account	BP 31	check detail field rental income report
BP 7	create new account	BP 32	print field rental income report
BP 8	reservation	BP 33	view reports managed
BP 9	check schedule certain date	BP 34	choose facility report
BP 10	choose desired schedule	BP 35	choose facility report based transaction date
BP 11	choose facility	BP 36	check detail facility report
BP 12	create new reservation	BP 37	print facility report
BP 13	choose my reservation	BP 38	view schedule rental price and facility list rule

5.1. Results

Similarity value is obtained from probabilistic value in the document to topic and probabilistic topic. The similarity was calculated using cosine similarity to each method used. The similarity value of PLSA and LDA are presented in Table 2 and Table 3, respectively. The ‘X’ and ‘O’ notations indicate the matches of the retrieval. ‘O’ means that the retrieved link is relevant and correctly retrieved. Alternatively, ‘X’ indicates that the retrieved link is not relevant. After constructing the traceability matrix, the next step is to calculate the value of precision and recall of the two methods. Before calculating precision and recall values, a similarity threshold was set. From our experiment, 0.6 and 0.8 are two top threshold values. Table 4 shows the topics used and the average precision and recall values on each topic using 0.6 threshold value. Meantime, 0.8 threshold compared the precision and recall values of each method as shown in Table 5.

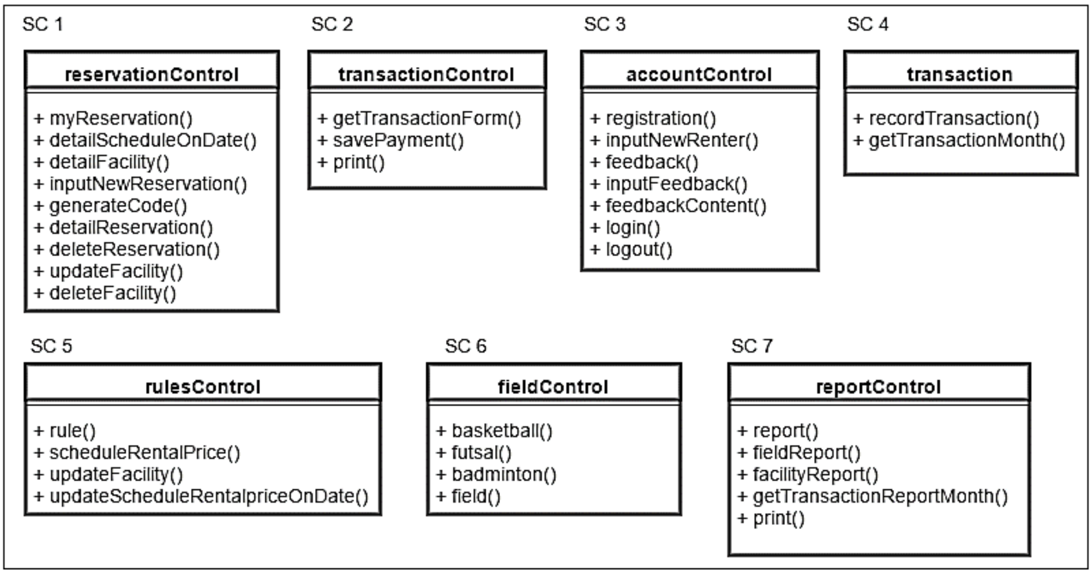

Figure 5. List of software components for data testing

Table 2. Traceability matrix obtained using PLSA

Business Process Activities	Software Component: Class Name	registrationForm			account		feedback		feedbackForm	
	Software Component: Function Name	Choose Registration	Show Registration Form	Choose Submit Data	Set Reter Account	Get Account	Set Feed Back	Get Feed Back	Show Feedback Form	Choose Feed Back
register		O	O	X	O	O				
fill registration form		O	O	O	X	X				
check available account		X	O	O	O	O				
create new account		O	X	X	O	O				
choose feedback							X	O	O	O
fill feedback form							O	O	O	O
view all feedback list							O	O	O	O
view feedback detail							O	X	O	O

Table 3. Traceability matrix obtained using LDA

Business Process Activities	Software Component: Class Name	RegistrationForm			Account		feedback		FeedbackForm	
	Software Component: Function Name	Choose Registration	Show Registration Form	Choose Submit Data	Set Reter Account	Get Account	Set Feed Back	Get Feed back	Show Feed back Form	Choose Feed Back
register		O	O	X	O	O				
fill registration form		X	X	O	O	O				
check available account		O	O	X	O	O				
create new account		O	O	X	O	O				
choose feedback							O	O	O	O
fill feedback form							O	O	O	O
view all feedback list							O	O	O	O
view feedback detail							X	O	O	O

Table 4. Model evaluation on threshold 0.6

Topic #	Avg. Precision of PLSA	Avg. Precision of LDA	Avg. Recall of PLSA	Avg. Recall of LDA
2	7.01	12.18	100.00	88.54
3	10.91	15.89	93.06	90.97
4	11.32	19.71	91.67	76.04
5	19.21	19.81	91.67	67.01
6	15.23	25.59	87.50	71.18
7	11.92	26.83	76.39	67.36
8	16.46	31.55	86.81	63.19
9	20.16	25.85	77.78	48.61
10	23.19	25.10	71.53	41.67
11	22.13	36.93	72.92	58.68

Table 5. Model evaluation on threshold 0.8

Topic #	Avg. Precision of PLSA	Avg. Precision of LDA	Avg. Recall of PLSA	Avg. Recall of LDA
2	6.81	12.08	100.00	86.46
3	11.32	17.79	78.82	87.15
4	12.83	22.23	82.99	64.58
5	19.22	23.89	87.85	55.21
6	16.62	33.09	77.08	47.57
7	13.57	27.42	65.63	45.14
8	18.82	30.28	77.43	47.92
9	20.79	24.76	73.61	37.85
10	24.43	25.17	56.25	30.56
11	25.62	42.19	64.93	44.44

5.2. Analysis

LDA and PLSA obtained a low value drawn from their final result. Many factors influence it. Specifically, datasets, total topics, and threshold. Problems occurred in class data and support functions. Classes are not specificly associated to a particular activity. For example, the activity "Register" should fit in with the class "registrationForm" but scores low on PLSA and LDA calculations. The low scores in the class "registrationForm" formed due to the class did not contain a special function to "register" alone but has another function that "reservation". If the class has a specific function, then the value obtained in class "registrationForm" is high.Other than dataset, the number of topics affected precision and recall values on PLSA and LDA methods. Although the comparison was performed on the same number of topics, the distribution of documents came on different topics.

The threshold was obtained from observation when testing. The 0.6 threshold value for the testing was higher than the one used in [16] and [17] because the result search query on the software component (retrieved relevant) was high, while the value that fits the relevant data was low. Thus, the need to increase the threshold value to be retrieved became less relevant and specific. The next threshold was raised to 0.8. The larger the threshold value, the more specific the retrieved relevant value. In contrast, the smaller the threshold value, the more relevant and wider the retrieved value. The precision using 0.6 and 0.8 thresholds in the LDA method is higher than the PLSA method due to specific relevant retrieved value obtained by LDA. However, the method PLSA gets higher recall value than the LDA method because the relevant software component value is more suitable to the activity in the business process.

6. CONCLUSION

In this paper, we performed traceability between business processes and software components. Specifically, for one problem caused by changing the name of an activity in the business process or a class name in the software component in which the name used in business processes and software components are different but have the same meaning. PLSA and LDA were adopted for performing traceability between business processes and software components. Both methods share the same underlying assumption, i.e., a collection of words comprise a topic, then a set of topics form a document. It led to handling the business processes and the software components as documents. Having optimized the probabilistic topic of the document, we calculated cosine similarity. Next, the traceability matrix was generated, and model evaluation was performed using precision and recall. Recall value on PLSA is higher than LDA for the relevant value because the software component is suitable for activity in the business process. Meanwhile, the precision value of LDA is higher than PLSA because the relevant retrieved value obtained by calculating the LDA is specific. The optimum result can be drawn when the dataset has a specific class based on the specific activity.

REFERENCES

[1] Tarhan A., Turetken O., Reijers H. A., "Business process maturity models: A systematic literature review," *Information and Software Technology*, vol. 75, pp. 122-134, July 2016.

[2] Samosir H., Siahaan D., "Generating requirement dependency graph based on class dependency," *IPTEK The Journal for Technology and Science*, vol. 29, no. 2, 2018.

[3] Luftman J., Brier T., "Achieving and sustaining business-IT alignment," *California management review*" vol. 42, no. 1, pp. 109-122, 1999.

[4] Ullah A., Lai R., "A systematic review of business and information technology alignment," *ACM Transactions on Management Information Systems (TMIS)*, vol. 4, no. 1, pp.1-30, October 2013.

[5] Habba M., *et al.*, "Alignment between business requirement, business process, and software system: A systematic literature review," *Journal of Engineering*, pp. 1-19, October 2019.

[6] Castellanos C., Correal D., "A Framework for alignment of data and processes architectures applied in a government institution," *Journal on Data Semantics*, vol. 2, no. 2-3, June 2013.

[7] Doumi K., Baina S., Baina K., "Strategic business and IT alignment: representation and evaluation," *Journal of Theoretical & Applied Information Technology*, vol. 47, no. 1, January 2013.

[8] Etien A., Rolland C., "Measuring the fitness relationship," *Requirements Engineering*, vol. 10, no. 3, pp. 184-197, August 2005.

[9] Kassahun A., Tekinerdogan B., "BITA*: Business-IT alignment framework of multiple collaborating organisations," *Information and Software Technology*, vol. 127, November 2020.

[10] Martinez A., *et al.*, "Incorporating technology in service-oriented business models: A case study," *Information Systems and E-Business Management*, vol. 2, no. 15, pp. 461-487, 2016.

[11] Darmawan H., *et al.*, "Anomaly detection based on control-flow pattern of parallel business processes," *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 16, no. 6, pp. 2808-2815, December 2018

[12] Von Rosing M., Von Scheel H., Scheer A. W., "The complete business process handbook: body of knowledge from process modeling to BPM, 2014.

[13] Sarno R., Pamungkas E. W., Sunaryono D., "Sarwosri business process composition based on meta models," *Proceeding of the Int. Semin. Intell. Technol. Its Appl. ISITIA*, pp. 315–318.

[14] Yan Z., Dijkman R., Grefen P., "Fast business process similarity search with feature-based similarity estimation," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, pp. 60-77, 2010.

[15] Dijkman R., *et al.*, "Similarity of business process models: metrics and evaluation," *Information System*, vol. 36, no. 2, pp. 498-516, April 2011.

[16] Revindasari F., Sarno R., Ahmadiyah A. S., "Traceability between business process and software component using probabilistic latent semantic analysis," *Proceedings of International Conference on Informatics and Computing, ICIC,* pp. 6-11, 2016.

[17] Baskara A. R, Sarno R., Ahmadiyah A. S., "Discovering traceability between business process and software component using latent dirichlet allocation," *Proceedings of International Conference on Informatics and Computing*, ICIC, Oct 2016.

[18] Hofmann T, "Unsupervised learning by probabilistic latent semantic analysis," *Machine Learning*, pp. 177-196, 2001.

[19] Blei D. M., Ng A. Y., Jordan M. I., "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993-1022, 2003.

[20] Aversano L., Grasso C., Tortorella M., "Managing the alignment between business processes and software systems," *Information Software Technology*, vol. 72, pp. 171-188, April 2016.

[21] Marcus A., Maletic J. I., "Recovering documentation-to-source-code traceability links using latent semantic indexing," *Proceedings 25th Int. Conf. Softw. Eng*, pp. 125-135, May 2003.

[22] Pessiot J. F., Kim Y. M., Amini M. R., Gallinari P., "Improving document clustering in a learned concept space" *Information Processing Management*, vol. 46, no. 2, pp. 180-192, March 2010.

[23] Al-Anazi S., *et al.*, "Finding similar documents using different clustering techniques," *Procedia Computer Science*, vol. 82, pp. 28-34, 2016.

[24] Thomas S. W., "Mining software repositories with topic models," *Proceedings - International Conference on Software Engineering*, May 2011.

[25] Asuncion H. U., Asuncion A. U., Taylor R. N., "Software traceability with topic modeling categories and subject descriptors," *Proceedings - International Conference on Software Engineering*, vol. 1, pp. 95-104, June 2010.

[26] Henderson J. C., Venkatraman H., "Strategic alignment: leveraging information technology for transforming organizations," *IBM Systems Journal*, vol. 32, no. 1, pp. 472-484, 1993.

[27] Verma T, "Tokenization and filtering process in rapidminer," *Int. J. Appl. Inf. Syst. Found. Comput. Sci*. FCS, vol. 7, no. 2, pp. 16-18, 2014.

[28] Ferilli S., Esposito F., Grieco D., "Automatic learning of linguistic resources for stopword removal and stemming from text," *Procedia Computer Science*, vol. 38, pp. 116-123, 2014.

[29] Hofmann T., "Unsupervised learning by probabilistic latent semantic analysis," *Machine Learning*, vol. 42, pp. 177-196, 2001.

[30] Joydeep AS, Strehl A., "Impact of similarity measures on web-page clustering," *Workshop of Artificial Intelligent for Web Search*, 2000.