

AI-driven creativity in software development using services information

Faisal Fahmi^{1,4}, Feng-Jian Wang², Hema Subramaniam³

¹Department of Library and Information Science, Faculty of Social and Political Sciences, Airlangga University, Surabaya, Indonesia

²Department of Computer Science, National Yang-Ming Chiao-Tung University, Hsinchu City, Taiwan

³Department of Software Engineering, Faculty of Computer Science and Information Technology, Universiti Malaya, Kuala Lumpur, Malaysia

⁴Center for Library and Information Studies, Airlangga University, Surabaya, Indonesia

Article Info

Article history:

Received Jun 12, 2024

Revised Aug 16, 2025

Accepted Sep 7, 2025

Keywords:

Creative software

Innovations

Requirement's engineering

Service registry

Software development

ABSTRACT

In competitive markets, organizations that do not innovate risk becoming outdated. At it is core, innovation depends on creativity, with creative outcomes typically characterized by novelty, usefulness, and surprisingness. In software development, creative solutions are often generated through brainstorming sessions. However, brainstorming is constrained by the knowledge of the participant and facilitator. In this paper, we present an artificial intelligence (AI)-based method to generate creative solutions in software by leveraging service information. The presented method includes two phases, where the first phase involves constructing creativity resources through text clustering (TF-IDF, K-medoid) and capability extraction (dependency parsing), and the second phase employs semantic similarity along with structured creativity techniques (exploration, transformation, and combination) to generate creative solutions in software. Besides, experimental results showed that the AI-based method achieved comparable creativity scores to traditional brainstorming with more limited time, demonstrating fast and strong performance in generating novel and useful solutions, although participants perceived some results as less surprising due to overlap with brainstorming outcomes.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Faisal Fahmi

Department of Library and Information Science, Faculty of Social and Political Sciences, Airlangga University
Dharmawangsa Dalam Street, Airlangga, Gubeng, Surabaya, Indonesia

Email: faisalfahmi@fisip.unair.ac.id

1. INTRODUCTION

In today's competitive market, creativity has become one of the most critical factors for product or service success. With increasing competition and changing customer preferences, businesses need to constantly innovate and introduce new ideas to remain relevant and attract consumers. Products or services that disrupt markets or transform industries are typically creative, i.e., novel, useful, and surprising. For example, Amazon and Netflix disrupted their industries, i.e., retail and entertainment, and at first launch exhibited the characteristics of creative software [1], [2]. Accordingly, systematic methods for enabling creative solutions in software targeting to develop a software that can drive the next market disruption are investigated.

Exploring creative solutions within software development is not an easy task as it involves thinking beyond the constraints of traditional programming and seeking innovative solutions to complex problems. This requires a deep understanding of user needs and expectations, as well as a willingness to challenge existing norms and conventions. Creative solutions in software are often discovered with brainstorming

session among stakeholders during the development phase. However, brainstorming session often depends on participants knowledge and facilitator(s) skill, and comes with drawbacks, such as time-consuming nature of process, evaluation apprehension when contributing ideas, groupthink effects, and limited scalability in distributed or large teams [3].

On the other hand, service-oriented architecture (SOA) provides a service registry containing structured information about the functionalities of existing services from the various domain [4], [5]. This structured service information enables systematic analysis, making it particularly suitable for artificial intelligence (AI)-based creativity discovery compared to unstructured sources, such as user feedback or market analysis. This paper presents a novel method to discover creative solutions in software by leveraging services information in a service registry. The presented method comprises two phases: i) construct creativity resources using AI techniques, and ii) discover creative solutions in software during development. The first phase introduces a new technique based on AI to construct knowledge as resources used for creativity discovery. The second phase generates creative solutions to improve creativity discovery by applying semantic similarity and structured creativity strategies (exploration, transformation, and combination).

The research objective is to develop and evaluate an AI-driven approach for generating creative solutions in software by leveraging service information. This approach aims to enhance the innovation process in software development by systematically supporting creative solution exploration across domains. The main contributions include an integrated method combining capability extraction, clustering, and semantic similarity, along with an experimental evaluation comparing the proposed approach to traditional brainstorming in improving the creativity process.

This paper is structured as follows: section 2 provides the background on creativity discovery. Section 3 details the proposed method for generating creative software solutions using service information. Section 4 reports the experimental results and findings. Lastly, section 5 offers the conclusion along with directions for future work.

2. CREATIVITY DISCOVERY

Creativity can be defined as the ability to produce solutions that are original (novel and unexpected) and appropriate (useful) for a particular problem [6], [7]. In software, creativity can be reflected in various elements, such as features, interfaces, data processing, or technologies applied [8]. These creative elements need to balance novelty, surprisingness, and usefulness, without compromising one for the other.

Creativity discovery can be described as an internal cognitive process of exploring, transforming, or combining ideas across different domains [9], [10]. This process usually shifts between two modes: divergent thinking, which creates many possible ideas, and convergent thinking, which filters and selects the best ones [11]. In software development, the concept often aligns with situated (S)-creativity, where solutions are new within the project context, but may not be historically new (H-creative) [12]. Divergent thinking can be significantly enhanced by drawing knowledge from other domains, such as applying analogies (exploration), loosening boundaries (transformation), and mixing ideas from multiple fields (combination) [10]–[12].

An example of divergent thinking using three different techniques are shown in Figure 1. In the example, developing a mobile application under hardware limitations for user authentication may initially suggest conventional options like passwords or biometrics. Using exploration, analogies from other domains, such as keyless entry in automotive systems, may appear. Transformation can involve extending or reinterpreting existing technologies, such as applying retina or behavioral scans, while combination creates new mixes of known techniques, such as combining fingerprint verification with key-based entry mechanisms.

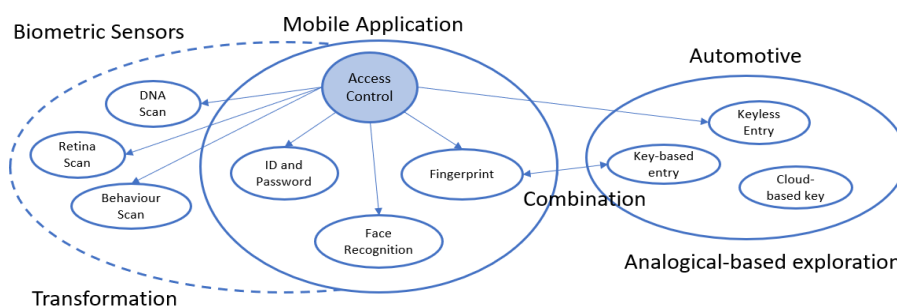


Figure 1. Example of exploration, transformation, and combination techniques for creativity

While brainstorming is commonly used to generate creative ideas by performing both divergent and convergent thinking, it has known drawbacks, such as evaluation apprehension, time-consuming, and limited scalability [3], [11]. Furthermore, the quality of outcomes depends heavily on the expertise of participants and the facilitator. Other divergent techniques, such as using semantic web or structured domain knowledge, may also be limited due to data gaps or incompleteness [13], [14]. This study introduces an AI-driven method designed to complement the divergent thinking phase by systematically extracting structured service information from service registries.

3. A METHOD TO GENERATE CREATIVE SOLUTIONS IN SOFTWARE

This section presents a method to generate creative solutions in software using AI-based techniques. The method contains two phases, where phase 1 applies natural language processing (NLP) and clustering techniques to transform unstructured service descriptions into structured creativity resources, and phase 2 uses semantic similarity analysis and structured creativity heuristics to discover and generate creative solutions. The creativity resources resulted at phase 1 is used to discover creative solution in software at phase 2, as shown in Figure 2. Furthermore, Figures 2 to 5 show the practical flow of tasks through the proposed method, from raw service descriptions through clustering and extraction to creative discovery, illustrating how each phase is applied step by step in practice.

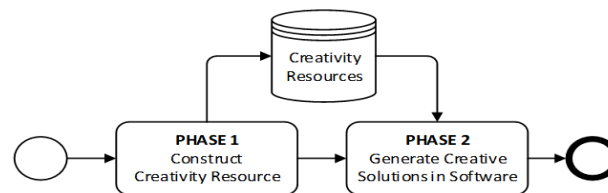


Figure 2. A method to generate creative solutions in software

3.1. Construct creativity resource

Software comprises of a set of functionalities as solutions to achieve the goals defined. The creative solutions in software can be obtained by discovering possible solutions with different functionalities from different domains that can achieve the same goal. These possible solutions may introduce creativity, because a solution from a specific domain may introduce novel, surprising, and useful properties when applied on the different domains.

Creativity resource contains a pre-processed collection of functionalities provided by existing service information from different domains. This service information is selected because it offers structured, machine-readable representations of functional capabilities that can be systematically analyzed and processed by the AI techniques in the proposed method. In contrast, alternative data sources, such as user feedback or market analyses, are typically unstructured, subjective, or qualitative, making them less suitable for automated, large-scale creative exploration. Prior to pre-processing, each service entered into the repository must provide at least three key elements: a unique name, a natural language description outlining its functionalities, and domain details specifying one primary domain along with any applicable secondary domains.

In this phase, there are two steps of pre-processing, as shown in Figure 3, that are applied to improve creativity discovery: i) service clustering and ii) service capability extraction, where service clustering is applied to reduce the search space, and service capability extraction is applied to derive functionalities provided by the services. The clustering processes apply AI-based text analysis techniques, including term frequency-inverse document frequency (TF-IDF) vectorization, cosine similarity, and K-medoid clustering, to organize service descriptions based on their semantic features. Additionally, dependency parsing with Stanford and Universal Dependencies (SUD) frameworks enables automated extraction of functional capabilities, creating structured data for later creativity discovery. Pre-processing also organizes services according to their domain, since S-creativity discovery methods rely on cross-domain knowledge to enhance creative outcomes. The resulting repository structure after pre-processing is illustrated in Figure 4, and the detailed pre-processing steps are described as follows.

3.1.1. Service clustering

To make the discovery process more efficient, the services are clustered within each domain since directly searching through the entire repository is time-consuming and resource-intensive. This clustering relies on TF-IDF, cosine similarity, and K-medoid algorithms, where TF-IDF is one of popular word

vectorization techniques that assesses how important a word is within a document collection [15], [16]. Cosine similarity then measures how closely related two documents are, regardless of their size, while K-medoid partitions the dataset into groups (clusters), each represented by a central point (medoid) with minimal average dissimilarity to other items in the group [17], [18].

For each domain, the clustering process includes the steps as follow:

- i) Pre-process the service descriptions by performing tokenization, lemmatization, and removing stop words, punctuation, and technical terms. Tokenization splits the text into meaningful words (tokens), while lemmatization transform words to their base form (e.g., “searching” becomes “search”). Common stop words (like “a,” “the,” “is”) and irrelevant technical terms (like “REST,” “HTTP”) are also removed.
- ii) Apply TF-IDF to convert each service description into a vector.
- iii) Apply cosine similarity to compute a dissimilarity matrix across services.
- iv) Select randomly k services as initial medoids.
- v) Refine iteratively the medoid set by selecting configurations that minimize overall dissimilarity.
- vi) Stop the iteration when the medoid set no longer changes.

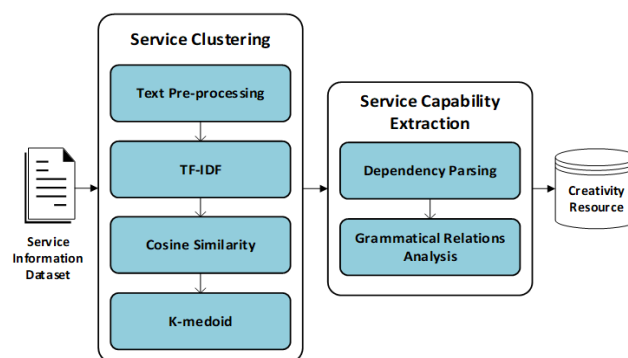


Figure 3. A method to construct creativity resource

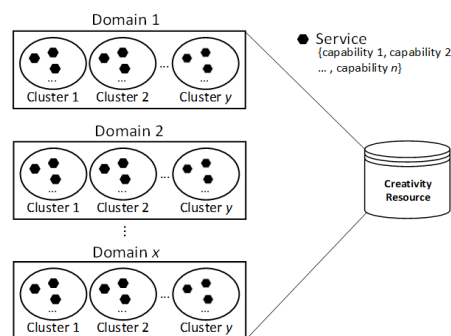


Figure 4. Structure of creativity resources

3.1.2. Service capability extraction

A service capability refers to a specific function that is provided by a service and can be accessed over a network. Typically, this function is represented as a verb (or verb phrase) paired with a noun (or noun phrase), where the verb describes the action, and the noun indicates the entity affected. For example, “submit order” pairs the action “submit” with the object “order”.

To improve the precision of the discovery process, each service is mapped to a set of capabilities that are extracted using the SUD parser [19], [20]. Services lacking extractable capabilities are excluded. The SUD parser identifies over 50 grammatical relations between words by performing tokenization, part-of-speech (POS) tagging (which labels each word according to its grammatical role, like noun or verb), and dependency parsing. A method based on SUD from [21] is adopted to extract service capabilities from a service description. The extraction procedure includes the following steps:

- i) Break down each service description into individual sentences.
- ii) Apply SUD parsing to each sentence to identify grammatical dependencies.

- iii) Identify service capabilities based on four key dependency relations: direct object (dobj), passive nominal subject (nsubj: pass), combinations of oblique nominal (obl) and nominal subject (nsubj), and combinations of conjunction (conj) and dobj.
- iv) Lemmatize the identified capabilities to ensure consistent base forms.
- v) Remove services that yield no extractable capabilities.

An example of service capability extraction for a sentence “The trader submits a new order” is “submits order,” that is extracted based on dobj relation. After lemmatization, this becomes the standard capability representation “submit order.”

3.2. Generate creative solutions in software during development

Software creativity can appear as creativity within specific features (as explained in section 2), which, during development, are represented within a workflow model. This phase consists of four main steps (steps 1~4), illustrated in Figure 5. Step 1 involves building the initial (non-creative) requirements specification as a workflow. Steps 2 and 3 introduce creativity into the workflow through divergent and convergent thinking, respectively, with step 3 focusing on the collective examination of creative outputs generated in step 2. Step 4 becomes necessary only if the workflow lacks creative elements, and steps 2 to 4 are repeated iteratively until all creative solutions accepted.

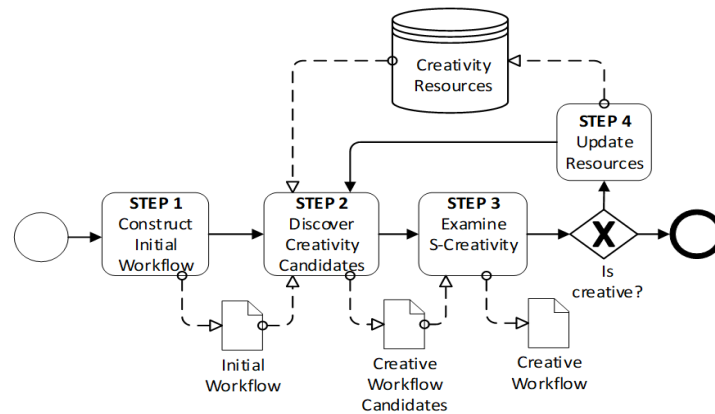


Figure 5. Generate creative solutions in software during development

Step 1: construct initial workflow. During this step, software engineers and stakeholders, i.e., clients, consumers, or domain experts, discuss each other to understand the application domain and construct requirement specifications with a workflow model. Similar to service capability, each task node in a workflow is described using a verb (or verb phrase) working on a noun (or noun phrase) [22].

Step 2: discover creativity candidates. Step 2 applies divergent thinking based on S-creative, described in Section 2, to discover candidates of creativity for the initial workflow. The creativity of workflow can be obtained by discovering different task nodes of the same goal from different domains. Since the formats of task node in initial workflow and service capability in creativity resource are the same, i.e., verb + noun, the alternative of the same goal for a task node can be discovered from the creative resource.

This step contains two main works: i) discover alternative of task nodes from service capabilities and ii) apply structured creativity techniques: exploration, transformation, and combination. These two works apply AI-based similarity calculations, where TF-IDF and Wu-Palmer semantic similarity are used to identify cross-domain analogies between task nodes and service capabilities. Through these automated matching and structured creativity techniques, the system can generate creative alternatives instantly that, in manual analysis, would be difficult or time-consuming to produce. To simplify the problem, the method in both works select at most three items (e.g., domains, and task nodes) for the calculation. The effectiveness of the number selected needs to be studied further. The details of the work are described as follows:

- a) Discover alternative of task nodes from service capabilities.
 - i) For each task node in the initial workflow
 - Calculate TF-IDF cosine similarity between task node and pre-processed service descriptions of cluster' medoid, where the highest similarity is defined as the similarity to the cluster.
 - Select three clusters (at most) with highest TF-IDF cosine similarity, each from different domains.
 - ii) Select three task nodes (at most) from each cluster selected.

- iii) For each task node selected, identify the service capability that is similar semantically to that of task node by calculating Wu-Palmer semantic similarity [23].
- iv) Select and sort the service capabilities with semantic similarity above a threshold, where the threshold studied in this paper is 0.5.
- b) Apply structured creativity techniques. For each task node selected and it is similar service capabilities:
 - i) Apply the exploration technique using analogical reasoning [24], where three distinct capabilities (at most) are selected from three separate clusters, prioritizing those with the highest semantic similarity, ensuring the selected capabilities and the original task node share no overlapping noun elements.
 - ii) Apply the transformation technique using boundary relaxation principles [25] by selecting three distinct capabilities (at most) with the lowest semantic similarity (i.e., the most unusual) from three different clusters.
 - iii) Apply the combination technique by selecting three distinct unfamiliar combinations [25] (at most) from three separate domains, ensuring that the selected capabilities and the task node differ syntactically in both verb and noun elements. These unfamiliar combinations are formed by interchanging the noun and verb components between the original capabilities and the task node, and are identified based on the fewest similar capabilities within the creativity resource.

Step 3: examine S-creativity. This step applies convergent thinking based on S-creative, where the creativity generated in step 2 is collectively reviewed. Participants from the software development process assess the creativity by assigning scores according to three main factors: novelty, surprisingness, and usefulness. Specifically, novelty encompasses originality and paradigm relatedness; surprisingness covers unexpectedness and unusualness; and usefulness includes relevance, feasibility, and completeness [26]. The review process identifies the workflow achieving the highest creativity score that surpasses the participant-defined threshold when replacing the original task node with selected service capabilities.

Step 4: update resources (if necessary). If the creativity generated for the workflow cannot pass the examination in step 3, the creativity resources need to be updated by adding necessary services. After updating, repeat steps 2 to 4 until the workflow passes the examination.

To provide a comprehensive evaluation, this research compares the proposed AI-driven creativity method with traditional brainstorming, assessing both feasibility and practical effectiveness relative to human-centered techniques. In addition, the AI-driven method is designed for easy integration into existing workflows, allowing developers to input tasks, review AI-generated suggestions, and select alternatives through a standalone interface or as a plugin in tools like requirements platforms or integrated development environments (IDEs). Section 4 provides a concrete example showing how the creativity resources were applied to generate and evaluate solutions in practice.

4. RESULTS OF THE EXPERIMENT

The feasibility of the method is assessed using a dataset from ProgrammableWeb [27], a leading service registry. It includes 21,569 atomic APIs across 427 different domains and 6,424 composite APIs across 323 different domains. For the experiments, 10 randomly selected domains with over 550 combined services were analyzed. The method was implemented in Python using NLP libraries and run on an Intel Core i7-7700 (3.60 GHz) with 16 GB RAM.

4.1. Results of creativity resources construction

The service clustering technique is developed with Spacy, Scikit-learn, and PyClustering libraries, where to simplify the problem, the number of clusters is calculated by the ceiling of the number of services divided by 300. The service capability extraction technique is developed with Stanza library. The results of creativity resources constructed are shown in Table 1, where the service capability extraction rate is 98.8%. This rate means that the technique can retain information from service description.

4.2. Results of creativity generation

The workflow derived in step 1 of the experiment is a workflow of order planning in a company, where the company want to estimate and evaluate the order before placing it, as shown in Figure 6. In step 2, "Create order" is selected as task node for creativity discovery because it has the higher similarity to medoid of clusters in different domains. The results of creativity discovery techniques are as follows: i) exploration: "Create application", "Create market", and "Create task", ii) transformation: "Support printing integration", "Include trade value", and "Offer uploading option", and iii) Combination: "Create service", "Create transaction", and "Offer order". After examination in step 3, the participants select "Offer order" to replace the "Create order", as shown in Figure 7. Rather than placing an order directly with a specific supplier based on their offered prices, the participants prefer to send an order offer to multiple suppliers aligned with the

expected price of the company. This replacement is considered creative, as it solves the same problem through a novel, surprising, and useful solution derived from a different domain.

Table 1. Experiment result for creativity resources

Domains	Services	Clusters	Services with capabilities
Financial	1,943	6	1,931
e-commerce	1,650	5	1,628
Social	1,900	6	1,863
Payment	1,073	4	1,072
Enterprise	901	3	897
Travel	959	3	936
Government	773	3	765
Transportation	591	2	577
Education	594	2	582
Banking	559	2	558
Total	10,943	36	10,809

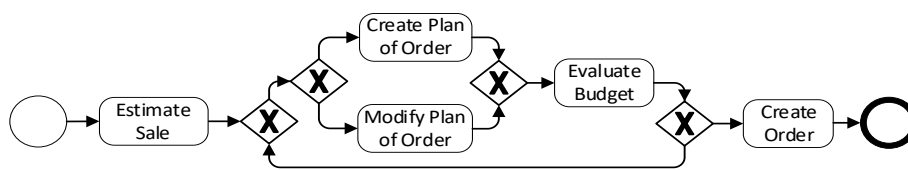


Figure 6. Initial workflow of order processing

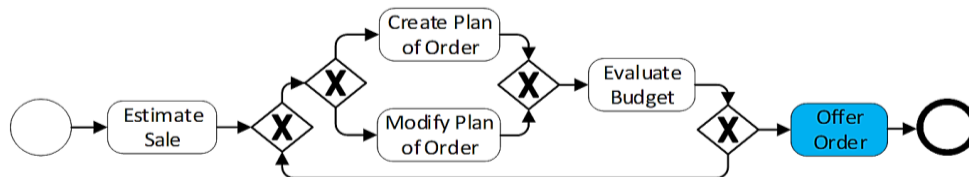


Figure 7. Workflow of order processing with creativity added

4.3. Quantitative comparison with brainstorming

To evaluate the feasibility and effectiveness of the proposed method, an experiment was conducted with seven participants, including software engineers and researchers. The experiment consisted of two sequential sessions: i) a traditional brainstorming session and ii) an AI-driven creativity session, both addressing the same workflow problem, i.e., order processing. Using the same participants across both sessions ensured a consistent comparison between human-centered ideation and AI-assisted creativity generation. It should be noted that since the brainstorming session was conducted first, its results can affect how surprising or novel the AI-generated solutions. Creativity was measured using three main dimensions: novelty, usefulness, and surprisingness, each divided into sub-properties described in step 3 of subsection 3.2.

In both sessions, participants provided quantitative assessments by scoring each solution on each sub-property, using a three-point scale (low, medium, high) guided by the standardized definitions. For example, paradigm relatedness was assessed as low when determining new uses for existing technology, medium when applying an established algorithm in a new category, and high when enabling technology transfer from one domain to another. In the brainstorming session, three rounds were conducted because only one idea was produced in the first and second rounds, indicating the time-consuming nature of brainstorming. From six resulted ideas, the top-selected ideas were “product dealing,” “compare product,” and “check shopping cart.”

The average total score for brainstorming-generated solutions was 14.86, slightly higher than the 13.57 average for the AI-driven solutions. Notably, the top brainstorming idea, “product dealing,” is conceptually similar to the AI-generated idea “offer order,” although “product dealing” may require a longer negotiation process with individual suppliers, while “offer order” directly sends offers to several suppliers simultaneously. Because the brainstorming session was conducted first, the similarity between these two ideas may have contributed to lower creativity scores for the AI-generated solution, as participants were already familiar with the general concept and thus perceived it as less novel or surprising.

On the other hand, completeness was the highest-scoring aspect in both approaches, with brainstorming scoring 2.71 and the AI-driven method scoring 2.29, indicating that both solution sets were

perceived as adequately solving the problem. The AI-driven solutions scored higher in originality (2.14 vs. 1.86), while brainstorming scored higher in paradigm relatedness (2.29 vs. 1.86) and unusualness (2.14 vs. 1.71). However, brainstorming required three iterative rounds to produce viable alternatives, whereas the AI-driven approach generated several creative solutions instantly. It should be noted that although multiple alternatives were generated, only ‘Offer order’ was selected, underscoring the essential role of human judgment in choosing the most suitable solution. The detailed creativity scores from the brainstorming and AI-driven sessions are summarized in Tables 2 and 3, respectively.

Table 2. Evaluation result of brainstorming-based creativity

Participants	Novelty		Usefulness			Surprisingness		Total
	Paradigm relatedness	Originality	Relevance	Feasibility	Completeness	Unexpectedness	Unusualness	
1	3	2	2	2	2	3	3	17
2	2	2	1	2	3	2	2	14
3	1	1	3	3	3	2	2	15
4	3	3	2	1	3	3	3	18
5	1	1	2	2	2	1	1	10
6	3	2	2	2	3	1	2	15
7	3	2	2	1	3	2	2	15
Average	2.29	1.86	2.00	1.86	2.71	2.00	2.14	14.86

Table 3. Evaluation result of AI-driven creativity

Participants	Novelty		Usefulness			Surprisingness		Total
	Paradigm relatedness	Originality	Relevance	Feasibility	Completeness	Unexpectedness	Unusualness	
1	1	1	1	1	2	1	1	8
2	2	1	1	1	2	2	2	11
3	2	3	3	3	3	3	3	20
4	2	3	2	2	3	3	3	18
5	1	1	2	2	2	1	1	10
6	2	3	2	1	3	2	1	14
7	3	3	2	2	1	2	1	14
Average	1.86	2.14	1.86	1.71	2.29	2.00	1.71	13.57

5. CONCLUSION

This paper presents a novel and effective AI-based method to generate creative service software, during development, by leveraging existing services information. The method includes two phases: i) construct creativity resource and ii) discover creativity, where the feasibility of the method in each phase is demonstrated in the experiments. The creativity resources are constructed by extracting and clustering service capabilities from a famous service registry using Spacy, Scikit-learn, and PyClustering libraries. The service capability extraction technique is developed with Stanza library. The experiment results of creativity resources show that service capability extraction rate is 98.8%. For the discover creativity phase, the experiment showed that the method can successfully add creative solutions to a workflow representing system features by generating multiple alternatives across domains and achieve better performance to traditional brainstorming in terms of originality. These findings support the feasibility and effectiveness of using AI-based techniques to enhance creativity discovery in software development. Beyond software development, the proposed approach may also be applied to other domains, such as product design, marketing, or business model innovation, where recombining existing components can produce novel and valuable solutions. Several limitations observed in this study are related to the dataset. The performance of the method may depend on the quality and completeness of service descriptions, which can affect discovery accuracy. Additionally, applying the method to smaller or less diverse datasets may reduce the variety and originality of generated solutions. Future work should address these limitations by enhancing robustness when working with limited or incomplete data. Future research may also focus on refining the AI techniques to increase solution diversity and quality, as well as extending the applicability of the method to later software development stages, such as prototyping and testing.

ACKNOWLEDGEMENTS

The authors would like to express their gratitude for the financial support provided by Airlangga University. Special thanks also go to the SATU Joint Research Scheme for facilitating collaboration and providing a platform for international research partnerships.

FUNDING INFORMATION

This research was funded by Airlangga University under Grant 1596/UN3.LPPM/PT.01.03/2023.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Faisal Fahmi	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓
Feng-Jian Wang	✓	✓		✓		✓			✓	✓		✓		
Hema Subramaniam				✓		✓				✓				

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author, [FF], upon reasonable request.




REFERENCES

- [1] V. Anindita, "Disruptive strategy in disruption era: does netflix disrupt the existing market?," *International Journal of Business and Technology Management*, vol. 3, no. 1, pp. 2682–7646, 2021.
- [2] S. Juteau, "Netflix: disrupting the entertainment market with digital technologies, time and again," *Journal of Information Technology Teaching Cases*, vol. 15, no. 1, pp. 163–166, May 2025, doi: 10.1177/20438869231226394.
- [3] P. C. Shih, G. Venolia, and G. M. Olson, "Brainstorming under constraints: why software developers brainstorm in groups," in *HCI 2011-25th BCS Conference on Human Computer Interaction*, Jul. 2011, pp. 74–83, doi: 10.14236/ewic/HCI2011.30.
- [4] T. Erl, *Service-oriented architecture: analysis and design for services and microservices*, New Jersey, United States: Prentice Hall Press, 2016.
- [5] F.-J. Wang and F. Fahmi, "Constructing a service software with microservices," in *2018 IEEE World Congress on Services (SERVICES)*, IEEE, Jul. 2018, pp. 43–44, doi: 10.1109/SERVICES.2018.00035.
- [6] R. J. Sternberg, "Handbook of creativity," *Choice Reviews Online*, vol. 36, no. 11, pp. 36-6571-36-6571, Jul. 1999, doi: 10.5860/CHOICE.36-6571.
- [7] M. A. Boden, *The creative mind: myths and mechanisms*. London, United Kingdom: Routledge, 2004, doi: 10.4324/9780203508527.
- [8] D. Jing and H. Yang, "Domain-specific 'ideation': real possibility or just another Utopia?," *Journal of Applied Science*, pp. 68–99, 2015.
- [9] N. Maiden, A. Gizikis, and S. Robertson, "Provoking creativity: imagine what your requirements could be like," *IEEE Software*, vol. 21, no. 5, pp. 68–75, Sep. 2004, doi: 10.1109/MS.2004.1331305.
- [10] N. Maiden, S. Jones, K. Karlsen, R. Neill, K. Zachos, and A. Milne, "Requirements engineering as creative problem solving: a research agenda for idea finding," in *2010 18th IEEE International Requirements Engineering Conference*, IEEE, Sep. 2010, pp. 57–66, doi: 10.1109/RE.2010.16.
- [11] N. Maiden and S. Robertson, "Integrating creativity into requirements processes: experiences with an air traffic management system," in *13th IEEE International Conference on Requirements Engineering (RE'05)*, IEEE, 2005, pp. 105–114, doi: 10.1109/RE.2005.34.
- [12] M. A. Boden, "Creativity and artificial intelligence," *Artificial Intelligence*, vol. 103, no. 1–2, pp. 347–356, Aug. 1998, doi: 10.1016/S0004-3702(98)00055-1.
- [13] F. Fahmi, P.-S. Huang, F.-J. Wang, and H. Yang, "Constructing a creative software with services," in *2021 IEEE International Conference on Services Computing (SCC)*, IEEE, Sep. 2021, pp. 134–144, doi: 10.1109/SCC53864.2021.00026.
- [14] P.-S. Huang, F. Fahmi, and F.-J. Wang, "A model to helping the construction of creative service-based software," in *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, IEEE, Jul. 2021, pp. 1235–1242, doi: 10.1109/COMPSAC51774.2021.00171.
- [15] S. Vajjala, B. Majumder, A. Gupta, and H. Surana, *Practical natural language processing: a comprehensive guide to building real-world NLP systems*. Sebastopol, United States: O'Reilly, 2020.
- [16] A. R. Lubis, M. K. M. Nasution, O. S. Sitompul, and E. M. Zamzami, "The feature extraction for classifying words on social media with the naïve bayes algorithm," *IAES International Journal of Artificial Intelligence*, vol. 11, no. 3, pp. 1041–1048, Sep. 2022, doi: 10.11591/ijai.v11.i3.pp1041-1048.
- [17] L. Kaufman and P. J. Rousseeuw, "Partitioning around medoids (program PAM)," *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, Hoboken, pp. 68–125, 2008, doi: 10.1002/9780470316801.ch2.




- [18] S. Jalal, D. K. Yadav, and C. S. Negi, "Web service discovery with incorporation of web services clustering," *International Journal of Computers and Applications*, vol. 45, no. 1, pp. 51–62, Jan. 2023, doi: 10.1080/1206212X.2019.1698131.
- [19] M. C. De Marneffe *et al.*, "Universal stanford dependencies: a cross-linguistic typology," in *Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC 2014*, 2014, pp. 4585–4592.
- [20] M.-C. de Marneffe, C. D. Manning, J. Nivre, and D. Zeman, "Universal dependencies," *Computational Linguistics*, vol. 47, no. 2, pp. 255–308, May 2021, doi: 10.1162/coli_a_00402.
- [21] B. Jiang, L. Ye, J. Wang, and Y. Wang, "A semantic-based approach to service clustering from service documents," in *2017 IEEE International Conference on Services Computing (SCC)*, IEEE, Jun. 2017, pp. 265–272, doi: 10.1109/SCC.2017.41.
- [22] B. Bruegge, *Object oriented software engineering using uml, patterns, and java*. New Jersey, United States: Prentice Hall, 2010.
- [23] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *32nd annual meeting on Association for Computational Linguistics*, 1994, pp. 133–138, doi: 10.3115/981732.981751.
- [24] N. Maiden, C. Ncube, and S. Robertson, "Can requirements be creative? experiences with an enhanced air space management system," in *29th International Conference on Software Engineering (ICSE'07)*, IEEE, May 2007, pp. 632–641, doi: 10.1109/ICSE.2007.24.
- [25] B. Giunta, C. Burnay, N. Maiden, and S. Faulkner, "Creativity triggers: extension and empirical evaluation of their effectiveness during requirements elicitation," *Journal of Systems and Software*, vol. 191, Sep. 2022, doi: 10.1016/j.jss.2022.111365.
- [26] D. Dean, J. Hender, T. Rodgers, and E. Santanen, "Identifying quality, novel, and creative ideas: constructs and scales for idea evaluation," *Journal of the Association for Information Systems*, vol. 7, no. 10, pp. 646–699, Oct. 2006, doi: 10.17705/1jais.00106.
- [27] M. Liu, Z. Tu, Y. Zhu, X. Xu, Z. Wang, and Q. Z. Sheng, "Data correction and evolution analysis of the programmableweb service ecosystem," *Journal of Systems and Software*, vol. 182, Dec. 2021, doi: 10.1016/j.jss.2021.111066.

BIOGRAPHIES OF AUTHORS






Faisal Fahmi    received the M.Sc. and Ph.D. degrees in Electronic Engineering and Computer Science from National Chiao-Tung University, Hsinchu City, Taiwan. He is currently an Assistant Professor in Airlangga University, Surabaya, Indonesia. His research interests include software engineering, microservices and service-oriented architecture, information development, and machine learning. He can be contacted at email: faisalfahmi@fisip.unair.ac.id.



Feng-Jian Wang    completed his Ph.D. program in Department of Electronic Engineering and Computer Science, Northwestern University. During his Ph.D. program, he worked on incremental analysis of data flow. He is currently a fulltime professor in Department of Computer Science in National Yang-Ming Chiao-Tung University, Taiwan. His research interests include software engineering, service-oriented architecture and workflow programming, cloud computation methodology, rich internet application, and object-oriented methodology. He can be contacted at email: fjwang@cs.nycu.edu.tw.



Hema Subramaniam    a senior lecturer in the Department of Software Engineering, Faculty of Computer Science and Information Technology at Universiti Malaya. She earned her Ph.D. in Software Engineering from Universiti Putra Malaysia in 2016, and prior to that, she obtained a Master of Computer Science (Software Engineering) from Universiti Selangor in 2010, as well as a B.Sc. in Information Technology from the same institution in 2006. Her research has primarily focused on software quality measurement and reusability, and she has actively engaged in investigating mental well-being analytics through social media insights and multi-model analysis, software solutions for learning disabilities. She can be contacted at email: hema@um.edu.my.