Journal of Architectural Engineering Technology
ISSN: 2168-9717
Mini Review Open Access

A Novel Approach for Code Design Line of Business Engineering

Chris Jones*

Department of Transportation and Energy Engineering Dr.-Friedrichs-Ring 2A, 08056 Zwickau, Germany

Abstract

A large software package exists in several forms, as totally variants targeting different business desires and users. This type of systems is provided as a collection of "independent" product and not as a "single-whole". Developers use ad-hoc mechanisms to manage variability. We have a tendency to defend a vision of software package development wherever we have a tendency to contemplate associate SPL design ranging from that the design of every variant is derived before its implementation. Indeed, every derived variant will have its own life. during this paper, we have a tendency to propose a unique approach for software package design line (SAPL) Engineering. It consists generic method for sick associate SAPL model that may be a line of "software architectures" from large-sized variants. Forward-engineering method that uses the recovered SAPL to derive new tailor-made software package design variants the approach is foremost experimented on 13 Eclipse variants to make a brand new SAPL. Then, associate intensive analysis is conducted mistreatment associate existing benchmark that is additionally supported Eclipse IDE. Our results showed that we will accurately reconstruct such associate SAPL and derive effectively pertinent variants. Our study provides insights that sick SAPL so account software package architectures offers smart documentation to grasp the software package before dynamical it.

Keywords: Systematic literature review; Content analysis; Construction engineering and management; Architecture; Architecture deigning

Introduction

Software Product Line Engineering (SPLE) aims to enhance employ by focusing not on the event of one product however on a family of connected product. The systems during a product Line (SPL) approach area unit developed from a typical set of assets during a prescribed manner, in distinction to being developed one by one, from scratch, or in associate ad-hoc manner. This production economy makes the product line approach engaging. SPLE considers the existence of one design model describing all the variants that implement completely different software package product of one line. The quality of this "single" design model is that it includes what's refereed as a variability model (also known as feature model), during which variability and commonality area unit expressly such mistreatment high level characteristics of the questionable options these area unit then mapped to parts, that area unit organized per the known options. Specific software package variants is derived (generated) by selecting from the feature model a collection of desired options, then SPL tools opt for and assemble the acceptable parts mapped to the chosen options [1,2].

During recent years, multiple approaches are projected addressing SPL implementation, or product derivation However, there area unit several software package systems that exist as many "independent" software package variants and not as a "single whole". Indeed, giant element software package systems exist in several forms, as totally software package variants targeting different business desires and users. as an example, day like Eclipse exist as many variants targeting completely different varieties of software package engineers (Martinez et al., 2018). These software package variants typically use ad-hoc mechanisms to manage variability and that they don't take complete advantages from the SPLE framework. For developers of latest software package variants that area unit engineered upon existing ones, the presence of one model describing the design of the full system with a certain specification of commonality and variability is of nice interest. Indeed, this permits to visualize the common a part of the full, on prime of that new practicality is engineered, additionally to the various options they'll use.

In this work, we have a tendency to defend a vision of software package development wherever we have a tendency to contemplate associate SPL design ranging from that the software package design of every software package variant is derived. Indeed, every derived software package variant will have its own life. This life is regulated by evolution desires whose origin typically depends on the context that is restricted to every software package. From the purpose of read of the accountable of the software package maintenance, the design may be a crucial unit for 2 reasons perceive the software package before creating changes thereon, and appraise changes created on the software package to stay its documentation compliant with its implementation.

However, matters wherever the software package variants don't have their own/proper design raises issues throughout the upkeep stage of a software package on the 2 points mentioned above concerning a generic design to grasp a given software package may be a terribly troublesome task. Knowing that comprehension is that the costliest activity throughout maintenance this can generate tidy further costs modifying a generic design, to require into consideration the modifications created on one in all its software package product, may be a task that's not solely troublesome and error prone, however conjointly with unpredictable consequences on the opposite software package product. Our vision is that the various software package variants is created from a similar SPL, however should have their standalone software package architectures to be ready to evolve severally and while not constraints. However, it's unremarkably familiar that having the software package design of a system is healthier than addressing its ASCII text file [3-6].

***Corresponding author:** Chris Jones, Department of Transportation and Energy Engineering Dr.-Friedrichs-Ring 2A, 08056 Zwickau, Germany, E-mail: chrisjones.22@gmail.com

Received: 29-Sep-2022, Manuscript No. jaet-22-76459; **Editor assigned:** 03-Oct-2022, PreQC No. jaet-22-76459 (PQ); **Reviewed:** 10-Oct-2022, QC No. jaet-22-76459; **Revised:** 17-Oct-2022, Manuscript No. jaet-22-76459 (R); **Published:** 27-Oct-2022, DOI: 10.4172/2168-9717.1000302

Citation: Jones C (2022) A Novel Approach for Code Design Line of Business Engineering. J Archit Eng Tech 11: 302.

Our approach for finding the 2 issues mentioned higher than is that the merchandise line should 1st turn out the software package design of a product, before its corresponding software package unit. This paper considers the challenge of analyzing the ASCII text file and therefore the software package designs of existing variants of component-based software package systems to reverse-engineer a software package architecture to any or all the present software package variants. we have a tendency to decision this created design a software package design line (SAPL) that represents the distinctive software package design that supports the product line and customary to any or all the software package variant members of the SPL.

Discussion

Most of existing SPL extractive approaches focuses solely on ASCII text file. They in the main recover feature models from the ASCII text file and maintain traceability links between every feature and its associated code fragments. In our case, we have a tendency to recover SAPL as well as a special reasonably feature models, wherever options area unit associated with design fragments. Additionally, the obtained SAPL permits thereby to derive a software package design for a given product instead of solely showing traceability links. Besides, within the literature, and to our greatest data, there area unit few works that mix during a complete method the advantages of software package design recovery techniques with SPL extractive approaches. Such works were analyzed and mentioned during a mapping study wherever the authors state that it's unclear however software package design techniques that are principally developed for one system is used effectively in associate SPL context.

In this work, we have a tendency to propose a unique approach for software package design line Engineering. the method of our approach was at first introduced in our previous work that is well extended during this paper according 2 main dimensions a lot of elaborated and extended specification of the 2 steps [7]. Specially, we have a tendency to describe the SAPL-Forward Engineering step during a new complete manner, and a brand new larger experimentation. This approach consists of a whole method that aims to take advantage of the advantages of software package design recovery techniques for single systems within the context of SPL. The projected approach consists of 2 methods a process for SAPL-reverse-engineering that extends the BUT4Reuse framework that is taken into account jointly of the foremost effective ways for SPL-reverse-engineering.

This framework was projected as a generic and protrusive framework for SPL reverse-engineering. For sanctionative extensibility, BUT4Reuse depends on adapters for the various unit varieties. These adapters area unit enforced because the main parts of the framework. Many adapters covering a good vary of unit varieties area unit already accessible. During this work, we have a tendency to follow the extensibility mechanisms of the BUT4Reuse Framework to implement a brand new adapter for SAPL reverse-engineering from giant component-based software package systems from a set of their existing variants.

The created SAPL architectures area unit of nice interest since they permit to visualize the variability points within the software package variants furthermore as maintain the dependency between these variants forward engineering method that uses the recovered SAPL to derive new tailor-made software package design variants. Many configurations is created ranging from this SAPL. They represent associate thorough enumeration of all the doable valid configurations. During this method, the discovered constraints from the bottom-up method area unit accustomed derive valid and consistent variants. Thus, we have a tendency to followed the extensibility mechanisms of the to develop a software package design musician that permits to pick out ranging from the SAPL a collection of desired options (a doable configuration) that meet a given set of user necessities and derive the software package design of the new variant [8,9].

The approach is foremost experimented on 13 Eclipse IDE variants to make a brand new SAPL. Then, associate intensive analysis is conducted mistreatment associate existing benchmark that is additionally supported Eclipse IDE. We have a tendency to engineered the design model of Eclipse IDE SPL and derive new software package design variants. The results of the experiments showed that our approach will effectively reconstruct such associate SAPL and derive valid and pertinent variants. one in all the insights that may be provided supported our study is that sick SAPL is of nice interest since it permits to derive the software package architectures of latest variants before their implementations. This can be a very important activity in software package maintenance and evolution since it offers smart documentation to grasp the product before dynamical it [10].

Conclusion

The remaining of the paper is organized as follows. In Section two, we have a tendency to expose background material concerning product Line Engineering and therefore the extractive adoption of SPLs. we have a tendency to conjointly introduce associate example that is a running example for illustrating our proposals. Section three presents a general image of the projected approach. In Section four, we have a tendency to expose our SAPL-Reverse Engineering method, the projected SAPL Met model for Component-Based software package Variants, and its internal representation for the OSGi systems. Section five describes our SAPL-Forward Engineering method.

Recovering design models of large-sized software package product is a very important activity in software package maintenance and evolution. These design models supply an honest documentation to grasp the product before dynamical it. For big software package product with many software package variants, these models area unit of nice interest since they permit conjointly to visualize the common and variable options between software package variants. SPL Reverse Engineering (SPL-RE) processes change to recover models.

Acknowledgement

None

Conflict of Interest

None

References

1. Zhuqing W, Ping Z, Yunsong L, Longwei L,Yongsheng Z (2020) Four-dimensional bioprinting: Current developments and applications in bone tissue engineering. Acta Biomater 101: 26-42.
2. Coppens O (2021) Nature-Inspired Chemical Engineering for Process Intensification. Annu Rev Chem Biomol Eng 12: 187-215.
3. Moataz A, Felix A, Michael B, Rebeca DE, Fabian K, et al. (2022) Visualization for Architecture, Engineering, and Construction: Shaping the Future of Our Built World. IEEE Comput Graph Appl 42: 10-20.
4. Chen J, Kan W, Yi L, Chuck Z, Ben W (2021) Application of textile technology in tissue engineering: A review. Acta Biomater 128: 60-76.
5. Cong G, Peng X, Chao Y, Xiulai C, Liming L (2019) Genetic Circuit-Assisted Smart Microbial Engineering. Trends Microbiol 27: 1011-1024.

6. Stefany L, Fernanda C B, Cristiane , Tatiana D (2021) Effect of an ergonomic intervention involving workstation adjustments on musculoskeletal pain in office workers-a randomized controlled clinical trial. Ind Health 59: 78-85.

7. Rob H, Sean B, Justine H, Jeffrey K, Burton S (2022) Integrating Engineering With Nature® strategies and landscape architecture techniques into the Sabine-to-Galveston Coastal Storm Risk Management Project. Integr Environ Assess Manag 18: 63-73.

8. Samuel G, Srikanthan R, Azadeh M, Ali T, Iris VR, et al. (2021) Extrusion-based 3D (Bio)Printed Tissue Engineering Scaffolds: Process-Structure-Quality Relationships. ACS Biomater Sci Eng 7: 4694-4717.

9. Ringer W (2014) Monitoring trends in civil engineering and their effect on indoor radon. Radiat Prot Dosimetry 160: 38-42.

10. Srubar W (2021) Engineered Living Materials: Taxonomies and Emerging Trends. Trends Biotechnol 39: 574-583.