Published: 14 August 2025
Online AM: 27 January 2025
Accepted: 16 January 2025
Revised: 12 November 2024
Received: 05 September 2023

RESEARCH-ARTICLE

Introducing Phylogenetics in Search-based Software Engineering: Phylogenetics-aware SBSE

DANIEL BLASCO, San Jorge University, Zaragoza, Spain

ANTONIO IGLESIAS, San Jorge University, Zaragoza, Spain

JORGE ECHEVERRÍA, San Jorge University, Zaragoza, Spain

FRANCISCA PÉREZ, Polytechnic University of Valencia, Valencia, Valencia, Spain

CARLOS CETINA, Polytechnic University of Valencia, Valencia, Valencia, Spain

Introducing Phylogenetics in Search-based Software Engineering: Phylogenetics-aware SBSE

DANIEL BLASCO, SVIT Research Group, Universidad San Jorge, Villanueva de Gallego, Spain
ANTONIO IGLESIAS, SVIT Research Group, Universidad San Jorge, Villanueva de Gallego, Spain and PROS Research Center, VRAIN, Universitat Politècnica de València, Valencia, Spain
JORGE ECHEVERRÍA, SVIT Research Group, Universidad San Jorge, Villanueva de Gallego, Spain
FRANCISCA PÉREZ and CARLOS CETINA, PROS Research Center, VRAIN, Universitat Politècnica de València, Valencia, Spain

Phylogenetics studies the relationships, in terms of biological history and kinship, of a set of taxa (e.g., species). We argue that in Search-based Software Engineering (SBSE), the individuals of an evolutionary computation-driven population could be considered as taxa for which the leverage of Phylogenetic Inference might be beneficial. In this work, we present our Phylogenetics-aware SBSE approach. Our approach introduces a novel Phylogenetic Operation to promote results which are sufficiently aligned (in terms of lineage) with a certain reference given by the domain expert. Our approach is evaluated in two heterogeneous industrial case studies: Procedural Content Generation from Game Software Engineering, and Feature Location from Software Maintenance. The results are analyzed using quality-of-the-solution and acceptance-by-developers measurements. We performed a statistical analysis to determine whether the impact on the results is significant compared to baselines that do not leverage Phylogenetics. The results show that our approach significantly outperforms two baselines in both case studies. Furthermore, two focus groups confirmed the acceptance of our approach and stressed that solution acceptance may make the difference in industrial environments. Our work has the potential to motivate a new breed of research work on Phylogenetics awareness to produce better results in Software Engineering.

CCS Concepts: • **Software and its engineering → Search-based software engineering**; **Model-driven software engineering**;

Carlos Cetina is also with University College London.

This work was supported in part by the VARNETICA project (CNS2023-145422) and the Ministry of Economy and Competitiveness (MINECO) through the Spanish National R + D+i Plan and ERDF funds under the Project VARIATIVA under Grant PID2021-128695OB-I00, and in part by the Gobierno de Aragón (Spain) (Research Group T61_23R). Moreover, this work was partially supported by the Spanish Ministry of Science and Innovation under the Excellence Network AI4Software (Red2022-134647-T).

Authors' Contact Information: Daniel Blasco, SVIT Research Group, Universidad San Jorge, Villanueva de Gallego, Spain; e-mail: dblasco@usj.es; Antonio Iglesias, SVIT Research Group, Universidad San Jorge, Villanueva de Gallego, Spain and PROS Research Center, VRAIN, Universitat Politécnica de Valéncia, Valencia, Spain; e-mail: aiglesias@usj.es; Jorge Echeverría, SVIT Research Group, Universidad San Jorge, Villanueva de Gallego, Spain; e-mail: jecheverria@usj.es; Francisca Pérez (corresponding author), PROS Research Center, VRAIN, Universitat Politécnica de Valéncia, Valencia, Spain; e-mail: franciscaperez@upv.es; Carlos Cetina, PROS Research Center, VRAIN, Universitat Politécnica de Valéncia; e-mail: cetina@upv.es.

ACM 1557-7392/2025/8-ART193
https://doi.org/10.1145/3715002

Additional Key Words and Phrases: Phylogenetics, Search-based Software Engineering, Evolutionary Computation, Game Software Engineering, Procedural Content Generation, Software Maintenance, Feature Location, Model-Driven Engineering

ACM Reference format:
Daniel Blasco, Antonio Iglesias, Jorge Echeverría, Francisca Pérez, and Carlos Cetina. 2025. Introducing Phylogenetics in Search-based Software Engineering: Phylogenetics-aware SBSE. *ACM Trans. Softw. Eng. Methodol.* 34, 7, Article 193 (August 2025), 38 pages.
https://doi.org/10.1145/3715002

1 Introduction

Search-based Software Engineering (SBSE) [37, 38] addresses Software Engineering optimization problems and pursues the generation of near-optimal solution candidates for those problems by using an iterative search process. This process involves the evolution of solution candidates and finishes when a certain stop condition, which could depend on factors like time budgets or solution candidate evaluation, is met. First, SBSE approaches require the use of an encoded representation of both the problem studied and the solution candidates. Additionally, it is necessary to provide operations that allow the solution candidates to change or combine with each other in order to produce new solutions, like mutation and crossover [3, 25]. In the end, the solution candidates must be evaluated by means such as fitness functions.

In this work, we propose leveraging Phylogenetics in SBSE to achieve better results. In the context of Biological Systematics, which studies the relationships of living entities [71], Phylogenetics deals with the relationships regarding ancestry and lineages that exist for a group of taxa [12]. A possible example of a taxon would be a biological species. There are inference methods that produce a branching diagram called a Phylogenetic Tree for a given set of taxa. A Phylogenetic Tree hypothesizes about the relationships between those taxa in terms of kinship, lineage splitting, and genetic similarity.

Our approach leverages Phylogenetic inference, treating solution candidates as taxa in order to promote results that are sufficiently aligned with a certain reference given by the domain experts, due to the importance of their acceptance. Their approval is crucial in validating the final solution candidates proposed: the information encoded in those candidates and the metrics used to rank or evaluate them may be incomplete in comparison to the insight provided by a domain expert. Specifically, our approach promotes those solution candidates that share a significant portion of their lineage path with the lineage path for the reference given by the domain expert. In order to achieve this, our Phylogenetics-aware SBSE approach incorporates two new ingredients to SBSE: a Phylogenetic Input and a Phylogenetic Operation. The Phylogenetic Input includes the Reference Taxon and the Primeval Taxon. The Phylogenetic Operation measures the genetic distance between the solution candidates, which are used as taxa, and then uses that distance data to infer a Phylogenetic Tree that proposes a lineage hypothesis. Once the Phylogenetic Tree is built, the Phylogenetic Operation performs a lineage analysis to promote those solution candidates that are aligned with the Reference Taxon from a lineage point of view, which takes the Primeval Taxon as a starting point.

We have evaluated our Phylogenetics-aware SBSE approach and two baselines for each case study in two heterogeneous industrial case studies: **Procedural Content Generation (PCG)** [14] from **Game Software Engineering (GSE)** [7] provided by Kraken Empire;[1] and **Feature Location (FL)** [42] from **Software Maintenance (SM)** provided by **Construcciones y Auxiliar**

[1] 1https://www.krakenempire.com/

de Ferrocarriles (CAF),[2] which is a worldwide leader in train manufacturing that uses software models to generate the firmware that controls their trains. On the one hand, the presence of GSE is growing within the venues of software engineering research such as EMSE, SPLC, MODELS, FSE, ASE, or ICSE [74], and PCG is a topic of GSE that tackles a key challenge of video game development: content creation [14]. On the other hand, SM is a seminal challenge of software engineering, and FL can arguably be seen as one of the most frequent maintenance tasks undertaken by developers [26, 42, 61, 79] since SM involves the management of features (removing unwanted features, adding new features, improving existing functionalities).

We analyzed the results of each case study using quality and acceptance measurements. In the PCG case study, the quality measurements used are Video Game research measures [17] that are accepted in GSE research [14]: Completion, Duration, Uncertainty, Killer Moves, Permanence, and Lead Change. In the FL case study, the quality measurements are retrieval measures that are accepted in Software Engineering research: Recall, Precision, and F-measure [30, 48, 69]. In both case studies, the acceptance by eight domain experts in total is analyzed by means of the **Theory of Planned Behavior (TPB)** questionnaire [4]. TPB covers three dimensions: Attitude, Subjective norm, and Perceived behavioral control. This questionnaire suits the acceptance study best in the context of the case studies covered [64]. The results obtained by our approach and the baselines are then compared by means of statistical analysis (p-Values [9], $\hat{A}_{12}$ [78], ANOVA [77], Kruskal–Wallis [77], or Eta-squares [23, 76] depending on the data). Finally, we carried out a focus group interview with the domain experts of each case study.

The results show that our approach significantly outperforms the two baselines studied in the two industrial case studies. In the PCG case study, the overall quality measurement shows a small significant improvement over the baselines (up to 7% and 10%, respectively), whereas the acceptance measurements show a large significant improvement over the baselines (of 30.15% and 23.34%, respectively, considering the mean value). These results highlight that our approach is better aligned with the developers' expectations. When it comes to automatically generated content, acceptance might be the key to developers actually using the generated content in their video games. In the FL case study, our approach outperforms the baselines with a large difference in both the quality measurements (of 31.23% and 28.69% in F-measure for each baseline, respectively) and the acceptance measurements, where the mean improvement over the baselines reaches 20.14% and 31.14%, respectively. Two focus groups confirmed the acceptance of our Phylogenetics-aware SBSE approach.

In summary, we are using a SBSE approach with a "seeded" solution as starting point, with the two-fold goal of optimizing a given fitness function while still finding a solution that is not too different from the starting one. Our approach introduces a novel Phylogenetic Operation to achieve the latter by promoting results that are sufficiently aligned (in terms of lineage) with a certain reference given by the domain expert. The motivation is that the fitness function might not be expressive enough or properly represent all objectives that it needs to be optimized for. These "hard-to-encode" optimization goals are somehow present in the reference input, and therefore, one approach is to evolve individuals with the further goal of not being too different from the reference.

To the best of our knowledge, this is the first work that leverages Phylogenetics for SBSE. Our article claims that Phylogenetics awareness is beneficial in the context of SBSE, as this work shows in the case of PCG and FL. Specifically, we claim that:

—Phylogenetics-aware SBSE significantly improves the acceptance of the solutions produced in the cases of PCG and FL (large difference in both cases). Solution acceptance may make the

[2] www.caf.net/en

difference with regard to the use of results in industrial environments, as emphasized by the focus groups that we ran.
— Phylogenetics-aware SBSE also significantly improves the quality of the solutions produced in the cases of PCG and FL (small and large differences, respectively).
— Our work proposes a generic Phylogenetic Operation which works in two heterogeneous case studies. Since the only requirement for our Phylogenetic Operation is the measurement of the genetic distance between two individuals, the operation could be used in new domains in which it is possible to perform this calculation.
— In the context of SBSE, our Phylogenetic work could inspire other SBSE researchers to explore new Phylogenetic Operations. In addition, Phylogenetics might improve other parts of SBSE approaches: from the input (e.g., helping the domain expert to choose seeds in a Phylogenetic tree) to the output (e.g., presenting the solutions of the ranking in a Phylogenetic tree to assist with the decision making), including the fitness function (e.g., where lineage could be rethought to be one of the objectives).
— Apart from SBSE, Phylogenetics awareness has the potential to be useful in other Software Engineering areas. For example, in the area of **Software Product Lines (SPL)**, where commonalities and variability are exploited, Phylogenetics could improve SPL reengineering by utilizing lineages in order to formalize variability. Another possible application is Repository Mining since the evolution of the content of a repository could be reinterpreted as a Phylogenetic Tree. Indeed, our future work will explore these possibilities.

The structure of this article is organized as follows: Section 2 deals with related work. Section 3 gives background on Phylogenetics. Section 4 describes our leveraging of Phylogenetics in our Phylogenetics-aware SBSE approach. Section 5 presents the first case study: PCG in the context of GSE. Section 6 presents the second case study: FL in the context of SM. Section 7 deals with the evaluation, comparing our Phylogenetics-aware SBSE approach to baselines that do not make use of Phylogenetics. Section 8 presents the results obtained, and Section 9 addresses the discussion of those results. Section 10 describes the threats to validity. Finally, Section 11 includes our conclusions.

2 Related Work

This section is structured in terms of SBSE, PCG, and FL related work, taking into account both the main contribution of our work and the two case studies.

2.1 SBSE Related Works

In order to find research works that are related to Phylogenetics in SBSE, we performed a search in the science literature to find the answer to the following query: "Have Phylogenetics approaches been used in SBSE?" Our search string was "Phylogenetic" AND ("SBSE" OR "Search Based Software Engineering"). The inclusion criterion was (IC1) articles in the computer science area. The search was conducted in August 2023 in Scopus and WoS and included title, keywords, and abstract. The search returned one paper from Scopus and zero papers from WoS. After applying IC1, no paper remained. The results of the searches are available here: https://doi.org/10.5281/zenodo.13885161

However, the following works are relevant, even if they do not make use of Phylogenetics. The concept of population diversity has been used in SBSE search. Albunian [5] studied the effects of population diversity on search-based unit test generation by applying different diversity maintenance and control techniques. The author performed an experiment with a tool for automatic test generation (EvoSuite). He investigated the influence of five selection mechanisms and five configurations of fitness sharing on the diversity of the generated test suites. The results show that the roulette wheel selection leads to a greater increase in the population diversity. On the other hand,

the fitness sharing shows a clear increase in diversity. Menendez et al. [51] presented an automatic unit test generation tool (OutGen) that is able to detect bugs on C-written functions. The generation method used by the tool is driven by output diversity, which is semantic information. This approach improves the output uniqueness score of the test sets. The authors performed an experimental validation of the usefulness of their approach, which uses output sampling (in comparison to input sampling), and the results showed improvements in the mutation score and bug detection. The previous works take into account the study of diversity to improve SBSE, but our work follows a different research direction by leveraging lineage by means of Phylogenetics. Additionally, and even if it is not strictly related to Phylogenetics, there is a work by McMinn et al. [49], in which the concept of species is introduced in the context of SBSE. In that work, the notion of species is used to refer to sets of individuals or sub-populations. These sub-populations are evolved in the pursuit of separate objectives, for which sub-population-specific fitness functions, determined by species-specific desirable paths, are tailored. However, in our work, the concept of species is used to refer to a paradigmatic example of the role played by the individuals of a sole population in the process of Phylogenetic inference and study. In addition, there are works that are not related to Phylogenetics but study the use of pre-existing solution candidates as seeds, like the research by Rojas et al. [65] and Yoo and Harman [82]. However, those works focus on studying the effectiveness of seeds in the generation of unit tests for program code and test data production, respectively. Our work studies the outcomes of lineage-focused searches that use pre-existing candidates like the Primeval and Reference Taxa.

2.2 PCG from GSE Related Work

With the goal of finding recent works related to video game development with a Phylogenetics-based approach, we conducted a search of the recent literature. We looked for an answer to the following query: “Have Phylogenetic approaches been used to develop videogames?” We have used the following search string: “phylogenetic” AND (“game” OR “videogame”). The inclusion criterion was (IC1) articles in the computer science area. The search was performed in August 2023 on Scopus and WoS and included title, keywords, and abstract. The search returned 235 papers from Scopus and 609 papers from WoS. After applying IC1, only 33 papers remained. The full text reading revealed that none of these papers belong to the intersection of PCG and Phylogenetics.

However, the following works are relevant with regard to PCG. Melotti and Moraes [50] presented and implemented the **Deluged Novelty Search Local Competition (D-NSLC)** algorithm. That algorithm makes use of morphological niches to ensure solution diversity in PCG for a game. D-NSLC divides the population into different niches and seeks the best individuals in each of them while exploring the search space. They performed an experiment in a rogue-like videogame case study using four different setups. The results confirmed the benefits of introducing the Novelty Search: its introduction contributes significantly to the production of different individuals. Our work aspires to obtain videogame content that is not only new but that is also aligned with the vision of the developers and accepted for its inclusion in commercially released video games. For that reason, our work leverages the lineage of the candidate solutions.

The research presented by Preuss et al. [62] focuses on the study of quality and diversity when PCG is used to create game content. The authors performed an experiment where algorithms and distance measures were evaluated. The experiment was performed using a tool to generate game levels. They concluded that the Niching Evolutionary Algorithm 2 (NEA2) can balance quality and diversity well, but only when using a good distance function. Later, Gravina et al. [36] distinguished quality-diversity as a search strategy for search-based PCG. In comparison to our work, NEA2 makes use of distance matrices, which are also used to calculate genetic distances in the application of Phylogenetics in our work. However, this technique is used to perform diversity-ensuring clustering

tasks, while in our work, the use of Phylogenetics pursues the inference of a hypothesized graph-like structure that explains the lineage-like relationships between the solution candidates. Preuss et al. also discuss how the use of Novelty Search negatively influences the quality of the solutions. Conversely, our work does not decrease the quality of the solutions.

Our previous works address the use of SBSE for content generation [13, 14]. To this end, these works use an evolutionary algorithm that is guided by means of game simulations that use the content generated. We reuse the SBSE ingredients from these works: the individual encoding, the genetic operations, and the fitness function. In this work, these ingredients serve as a baseline which does not make use of Phylogenetics, which is what we propose here.

Finally, after the original submission of the present article to this journal, we conducted a study on the use of Phylogenetics for software product families [22]. That study proposes the creation of a phylogenetic tree based on the content of a commercial video game, with the leaves of that tree representing that content, whereas the inner nodes of the tree play the role of hypothetical ancestors. In that work, we argue that the hypothetical ancestors can serve as starting points for the creation of new content. The study suggests that hypothetical ancestors can inspire developers to create fresh content. In other words, developers visually inspect the phylogenetic tree, choose an ancestor (an inner node), and manually reconstruct it using their domain knowledge. The reconstruction of the ancestor is done manually by developers in contrast to the automatic content generation done by the SBSE approach in the present work. The present work explores how SBSE approaches can benefit from phylogenetics. It introduces a new phylogenetic operation that incorporates the concept of lineage, i.e., belonging to a path that connects two given references. In this work, there are no hypothetical ancestors, every node of the phylogenetic tree is a member of the population of the search approach. This approach automatically produces results that better align with decision makers' preferences. Both studies share the presence of an encoding, but the one used in the ancestors work is an ad-hoc encoding for video games and is based on game-specific content. In contrast, in this article, the encoding used is that of each specific SBSE approach that is meant to be improved by means of the phylogenetic operation proposed for SBSE.

2.3 FL from SM Related Work

In order to determine if a Phylogenetics approach has been used in FL, we searched for the answer to the question: "Have Phylogenetics approaches been used in FL?" In the search string, we used "Phylogenetic" AND ("Feature Location" OR "software product line" OR "SPL"). The inclusion criterion was (IC1) articles in the computer science area. The search was conducted in August 2023 in Scopus and WoS and included title, keywords, and abstract. The search returned 111 papers from Scopus and 135 papers from WoS. After applying IC1 only nine papers remained. The full text reading revealed that none of these papers belong to the intersection of FL and Phylogenetics.

Nevertheless, there are other works that also focus on improving FL. Poshyvanyk et al. [61] propose a FL technique named PROMESIR that combines **Latent Semantic Indexing (LSI)** over source code with a Scenario-based Probabilistic Ranking of events. Krüger et al. [43] present an exploratory case study on identifying and manually locating features in Marlin, which is a variant-rich open source embedded firmware. Another work [45] presents a FL technique named SITIR, which is a semi-automated technique for FL in source code. These works focus on source code and are not designed for software models like those from the case study of this article.

There are previous works by our SVIT research group that deal with FL in models using an evolutionary algorithm. In [31], a combination of Formal Concept Analysis and Latent Semantic Analysis is evaluated to guide the evolutionary algorithm. The work presented in [21] takes advantage of long-living software systems to address the FL challenge, using commonality and modifications through model retrospectives to promote model fragments that undergo fewer

modifications over time. In [32], the fitness function (the similarity to the feature description) is fixed and five search strategies are evaluated (Evolutionary Algorithm, Random Search, steepest Hill Climbing, Iterated Local Search with restarts, and a hybrid between Evolutionary Algorithm and Hill Climbing). The work in [59] presents an approach named Fragment Retrieval on Models that uses an evolutionary algorithm to search and retrieve the most relevant model fragments. In [47], a learning-to-rank approach is proposed to improve the fitness function to locate features in models. In [8], models at runtime are proposed to be used for increasing the information for FL. In [11], the study provides real measurements of location problems as a help to other researchers in the design of synthetic location problems. The work in [57] proposes that the human play the role of the fitness function of the evolutionary algorithm.

The FL strategy of the above works relies on the input query provided by a single engineer. In contrast, the work in [56] studies the impact that a number of engineers who collaborate to provide input information have on the quality of the solution during FL, and whether or not the inclusion of the engineers' confidence produces an improvement in the results. The work in [58] studies different criteria to assemble a team of software engineers that collaborate during FL. The goal of the work presented in [60] is to propose genetic operations that leverage the latent semantics that models hold rather than randomly generating new candidate solutions.

In contrast to the above works, the novelty of this work relies on leveraging Phylogenetics to promote those solution candidates that share a significant portion of their lineage path with the lineage path for a reference. To the best of our knowledge there is no previous scientific work that addresses Phylogenetics for FL. In addition, previous works do not evaluate the acceptance of the candidate solutions produced as this work does.

3 Phylogenetics Background

Phylogenetics consists in studying the existing relationships among taxa, hypothesizing their history with regard to biological evolution [12]. Taxa are groups of organisms that are associated to certain characteristics or attributes that determine their inclusion in such groups. For instance, a species is a taxon, but there are more high-level examples, like families or domains.

The relationship between two given taxa could be determined by measuring the genetic distance that separates them. For instance, in the case of species, genetic distance represents genetic divergence, which means that the distance between two specimens of the same taxon is zero. However, in other contexts such as neoplasia research it can represent mutation evolutionary distancing [72]. Genetic distance is commonly represented by a value in the interval [0.0, 1.0] [40] and can be calculated by means of various measures. There are measures, such as Cavalli-Sforza [20] or Nei's Standard [54], which assume that only mutations or genetic drift influence genetic differences. In the case of Cavalli-Sforza, the populations are represented in hyperspheres for which the distance units are denoted by the number of gene substitutions; Nei's Standard takes into account gene frequency divergence.

As an example of how diverse genetic distance measures can be, the top of Figure 1 shows a generic distance criterion based on the size of the intersection of the encoded taxa studied (T1, T2, T3, ...). For instance, Taxon 1 (T1) is encoded as A A A A A A A A A T. The example uses the widespread DNA nucleotides in a base-four encoding: Adenine (A), Cytosine (C), Guanine (G), and Thymine (T). The comparison of T1 and T2 evinces that they have five elements in common (5xA) out of the ten elements present in every taxon. Therefore, in this example, the distance between T1 and T2 is 0.5.

In our work, we use measures that calculate the differences between the genetic materials studied in a straightforward way without the assumptions described previously. These measures are a Euclidean genetic distance-based method and the Hamming Distance, for PCG and FL, respectively.

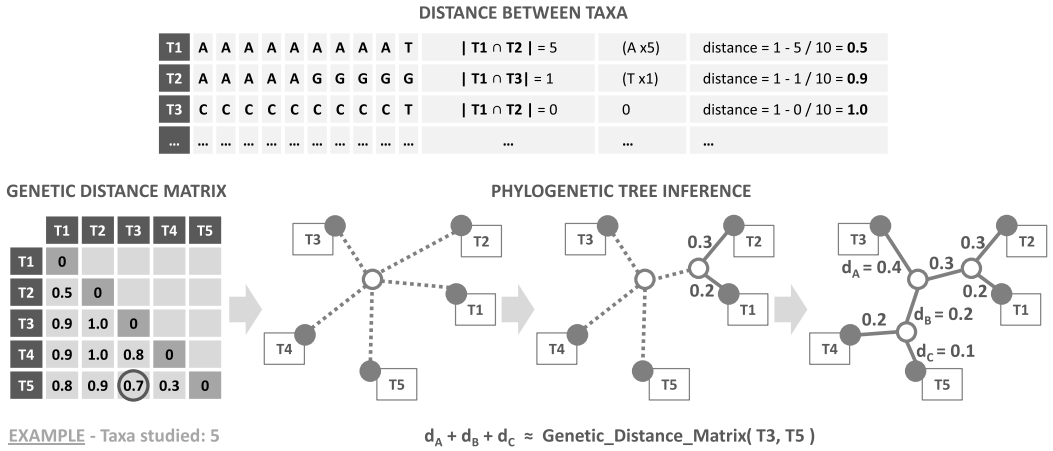


Fig. 1. Example of coherence between the genetic distance data and an inferred phylogenetic tree. Some of the values included in the distance matrix are omitted, due to the symmetric nature of the matrix. The example shows the neighbor-joining inference method and a generic genetic distance criterion based on intersection cardinality.

When two or more taxa are studied, it is common to use a distance matrix, which is a square matrix containing the distances for each pair of elements compared [80]. Therefore, the diagonal values of this matrix are always zero (see the Genetic Distance Matrix of Figure 1).

The Phylogenetics-based analysis of a given set or population of taxa results in a diagram known as a Phylogenetic Tree [12]. Usually, a Phylogenetic Tree inferred from a distance matrix will include the existing taxa studied as leaves. Additionally, it will include internal, non pre-existing nodes that define branches and the tree structure itself. These nodes are connected by edges having a distance value. For example, this internal structure is useful in determining the closeness between taxa or in identifying a point of inflection (i.e., an inner node) that marks the start of a certain branch or lineage.

The bottom of Figure 1 shows an example in which five taxa are studied. The Phylogenetic Tree that is inferred from the genetic distance data provides knowledge about relationships in terms of kinship, and it is coherent with the distance information from which it is built. The left part of Figure 1 shows a distance of 0.7 between T3 and T5. In our work, we used Neighbor-Joining. The Neighbor-Joining Method is a simple, fast, and widely accepted Phylogenetic Tree inference technique [53, 68]:

- —The algorithm starts assuming a star-like, unsolved tree topology with a distance matrix. Then, a Q matrix, which represents the goodness of joining any pair of taxa in a hypothetical new node, is calculated from that original distance matrix.
- —Next, a pair of taxa for which the distance is the lowest found in Q is selected.
- —A new node that connects the taxa of the pair is created and connected to the center of the star.
- —Once the distances from the node to all of the taxa are calculated, the new node takes the place of the two taxa of the pair in the original distance matrix, and the process starts over and continues until the length of every branch is known and the tree is resolved.

Then, once a Phylogenetic Tree is inferred, the path that connects those taxa includes the edges that are marked in the bottom section of Figure 1 as having distances of d_A, d_B, and d_C, respectively.

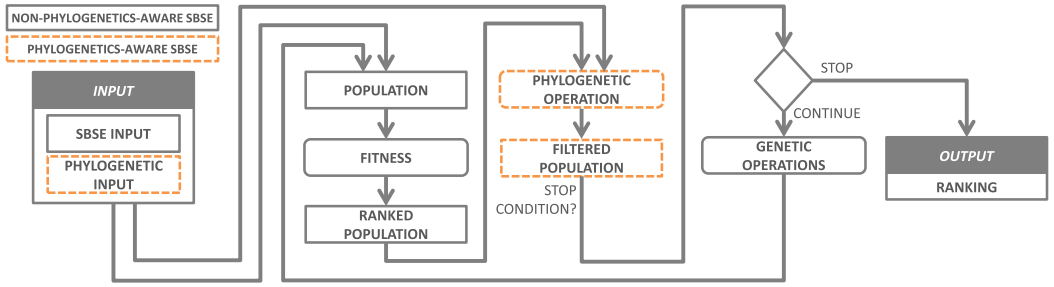


Fig. 2. Overview of phylogenetics-aware SBSE, with the elements of SBSE and the new elements of phylogenetics-aware SBSE.

The sum of those distances included in the path should be equivalent to the distance that appears in the distance matrix for the taxa studied. Hence, a Phylogenetic Tree keeps the relationships, in terms of genetic distance between the taxa, but it adds information (e.g., inner nodes of branches or lineages) with regard to relatedness for such taxa.

4 Leveraging Phylogenetics in the Context of SBSE

Figure 2 shows the ingredients of SBSE (solid lines). Only three key ingredients are needed to apply SBSE: (1) a representation (encoding) of the problem; (2) the definition of a fitness function; and (3) the definition of a set of operators. Then, candidate solutions (which are encoded following the representation chosen) are evaluated (by the fitness function) and evolved (by applying the operators) in an iterative process until optimal (or near-optimal) solutions to the problem are found.

Figure 2 also shows the novel ingredients that Phylogenetics-aware SBSE introduces to SBSE (dashed lines). Our approach promotes those solution candidates that to a certain extent share their lineage with a specific reference. This reference is provided by the domain expert (Phylogenetic Input). In order to ensure that the results promoted are aligned with that reference, our Phylogenetics-aware SBSE approach leverages Phylogenetic inference (Phylogenetic Operation).

4.1 Phylogenetic Input

In order to establish the reference lineage, our approach takes two examples as input. These examples become members of the population. In SBSE, it is common for approaches to take examples as inputs that are part of the population [38]. However, in our case these examples are considered as fixed members of the population because our approach keeps them in the population throughout all of the iterations. In other words, these two examples cannot be discarded from the population no matter what. In our Phylogenetics-aware SBSE, these two examples play the role of the ends of the reference lineage. Since the individuals of the population in our work are interpreted as taxa, we denote these examples as Reference Taxon and Primeval Taxon.

- —Reference Taxon: The Reference Taxon corresponds to a desirable reference for the developers. The Reference Taxon is the taxon that the solutions generated should resemble to a certain extent and that, at the same time, presents novelties.
- —Primeval Taxon: This taxon corresponds with the simplest example that is available with regard to the minimum desirable characteristics expected by the developers. During the lineage analysis of our approach, this primitive taxon is used as a starting point for the reference lineage.

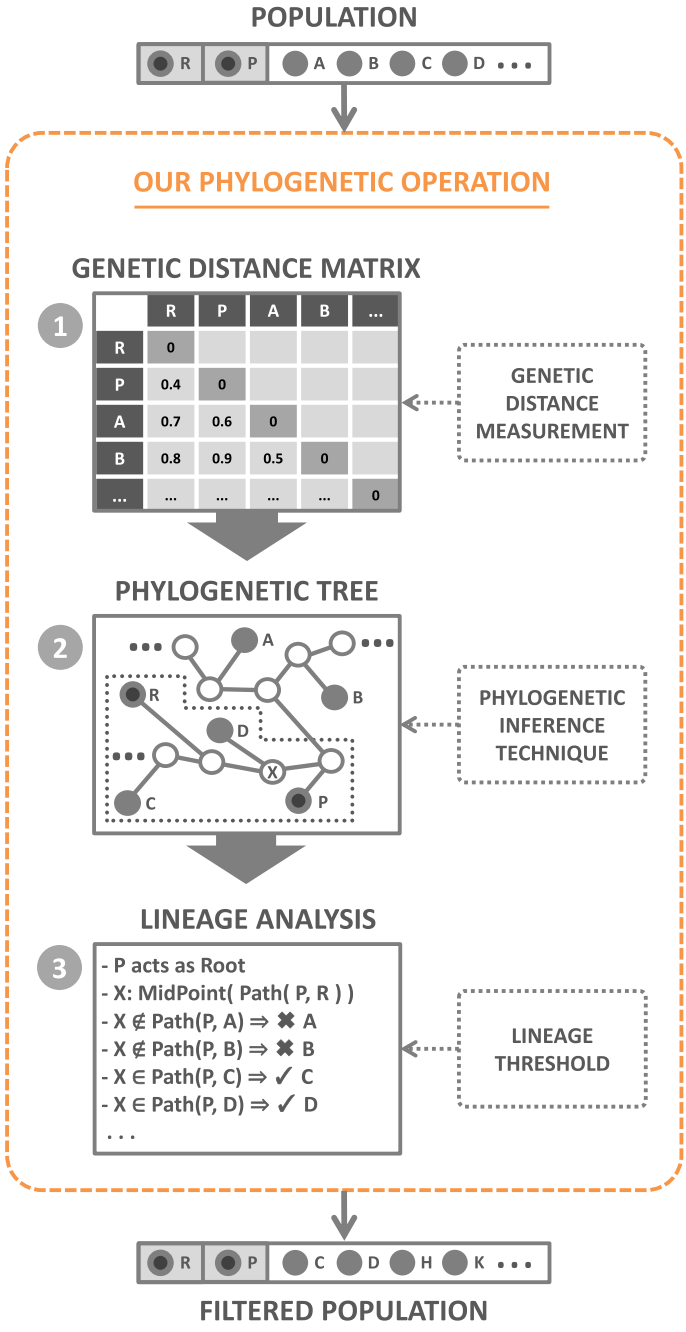


Fig. 3. Summary of the steps involved in the phylogenetic operation included in our phylogenetics-aware SBSE approach.

The top (population) and bottom (filtered population) of Figure 3 depict the Reference Taxon (R) and Primeval Taxon (P) as fixed members of the population by using a light-gray background.

4.2 Phylogenetic Operation

Given a population, our Phylogenetic Operation promotes those individuals that are sufficiently aligned with the reference by means of filtering out the individuals whose lineage is not sufficiently aligned to the lineage of the reference. To do so, our Phylogenetic Operation follows three steps: Genetic Distance Calculation, Phylogenetic Inference, and Lineage Analysis.

First, the Phylogenetic Operation calculates the Genetic Distance Matrix for the taxa represented by the individuals that are included in the population. The genetic distance value is obtained by using a measurement that compares the individuals of the population in pairs (see Step 1 of Figure 3). The distance between a taxon and itself should be 0.0, while two completely opposite individuals should produce a distance of 1.0. This genetic distance measurement is domain dependent, and the measurement must be provided in order to use our approach.

Then, our Phylogenetic Operation infers a Phylogenetic Tree. The construction of Phylogenetic Trees may be achieved by means of different methods, like Maximum Parsimony, Bayesian Inference, or Maximum likelihood. The focus of this work is not the investigation of the best Phylogenetic Tree construction method available but rather to leverage Phylogenetics to improve the results of SBSE. Our operation infers the Phylogenetic Tree by means of the Neighbor-Joining Method, using the Genetic Distance Matrix as input (see Step 2 of Figure 3). This method produces an unrooted tree. Our Phylogenetic Operation uses the Primeval Taxon as a root in terms of lineage. While rooted Phylogenetic Trees show ancestry relationships and evolutionary history information, the paths or relationships that are not explicit or directly related to chronology but mark interesting tendencies or drifts towards desirable results could remain hidden, since those relationships are only implicit. When the trees produced are unrooted, the paths produced focus on similarity, rather than on time-bound evolutionary routes. Therefore, in a tree like that, the path that connects the Primeval and the Reference taxa represents a latent progression (that is not necessarily linear or connected over time) of the genetic material of the taxa towards desirable characteristics. Figure 3 shows how the leaves in the inferred tree (marked in gray) represent the individuals of the population. In contrast, the white nodes do not represent existing individuals but rather provide the lineage structure of the tree by means of edges with distance values that are produced by the inference technique.

Finally, using the tree, our Phylogenetic Operation analyzes whether each individual of the population is sufficiently aligned with the lineage of the Reference Taxon. On the one hand, there is a lineage for each individual. This lineage consists in the path that exists between the individual itself and the Primeval Taxon. On the other hand, the reference lineage consists in the path between the Reference Taxon and the Primeval Taxon. In order to decide whether the two lineages (the lineage of the individual and the reference lineage) are sufficiently aligned, our approach uses a lineage threshold parameter.

The lineage threshold parameter allows being more strict or flexible with respect to how aligned the individual lineage and the reference lineage should be. This parameter enables our Phylogenetic Operation to be tuned.

Assuming that the parameter is set to the midpoint between the Reference Taxon and the Primeval Taxon, Figure 3 shows how Taxa A and B would be discarded, while Taxa C and D would be kept. Being an evolved relative does not only reflect genetic closeness; it also represents innovation as is the case of C and D (see Step 3 of Figure 3). Please note that the lineage threshold is represented by an X in the Phylogenetic Tree to ease the visual inspection of the lineages; however, the lineage threshold belongs to Step 3, not Step 2.

As a result, every individual included in the population is tagged depending on whether or not the similarity between its lineage and the reference lineage is sufficient. This Phylogenetic tag

allows for filtering the population in order to promote those individuals with a lineage that is close enough to the reference lineage.

For each new generation of the population, our Phylogenetic Operation updates the Genetic Distance Matrix, recalculates the Phylogenetic Tree, and reanalyzes the individual lineages with the reference lineage. In the case of the Genetic Distance Matrix, the update consists in removing or adding rows or columns, depending on the addition or removal of individuals. The Phylogenetic Tree, however, must be recalculated, since, for example, a small change in the population (like the inclusion of a new individual) does not necessarily imply the mere addition of a node in the tree or a slight change in a branch. It could mean a whole tree reconfiguration with respect to the tree obtained in the previous iteration of our approach. Therefore, the lineages must be analyzed again, using the updated tree.

5 Phylogenetics-aware SBSE for the Case of PCG

GSE [7] is a branch of Software Engineering that focuses on the development of video games. GSE work [14] tackles PCG. PCG helps with the automation or semi-automation of content generation for video games (levels, weapons, or items, to name a few).

Nowadays, most video games are developed by means of game engines. The term game engine refers to a development environment that integrates a graphics engine and a physics engine as well as a set of tools that wraps around them in order to accelerate development. The most popular ones are Unity [75] and Unreal Engine [34], but it is also possible for a studio to make its own specific engine.

Developers can create video game content directly using code (e.g., C++) or software models [83]. On the one hand, the code allows developers to have more control over the content. On the other hand, software models are much less bound to the underlying implementation and technology and raise the abstraction level using terms that are much closer to the problem domain.

Our industrial partner Kraken Empire provided us with the Kromaia video game.[3] Kromaia is a commercial 3D space shooter that was released on Playstation 4 and Steam. Each of the Kromaia levels involves flying from a starting point to a certain destination, and the player spaceship must reach the goal before being destroyed. The levels involve exploring floating structures, avoiding asteroids, and finding items along the route, which is protected by basic enemies. If the player manages to reach the destination, the boss corresponding to that level appears and must be defeated in order to complete the level.

In the Kromaia case study, we focus on the generation of a specific type of video game content: bosses. Video game bosses are powerful enemies that are typically confronted by the player at the end of the levels. Bosses are usually significantly stronger than the rest of the adversaries included in a video game. The bottom right part of Figure 4 shows one of the bosses of Kromaia.

For this case study, our approach includes the elements shown in Figure 2 as follows:

Input. The SBSE input is the set of final bosses that is included in the commercial version of Kromaia. These bosses are seeds for generating new bosses. The Phylogenetic Input includes the Reference Taxon and the Primeval Taxon. The Reference Taxon is the boss selected by the developers as being representative of what the new content should aim for, whereas The Primeval Taxon is the first boss ever created for the video game of the case study. Figure 4 shows examples of a possible Reference Taxon (BOSS-A) and Primeval Taxon (BOSS-B) for the case of PCG.

Population. An encoded boss from the video game of the PCG case study consists in a bi-dimensional matrix, as shown in the top right part of Figure 4: the columns correspond to the hulls or solid objects that are included in the body; the rows refer to both the presence/absence of parts

[3] https://youtu.be/EhsejJBp8Go

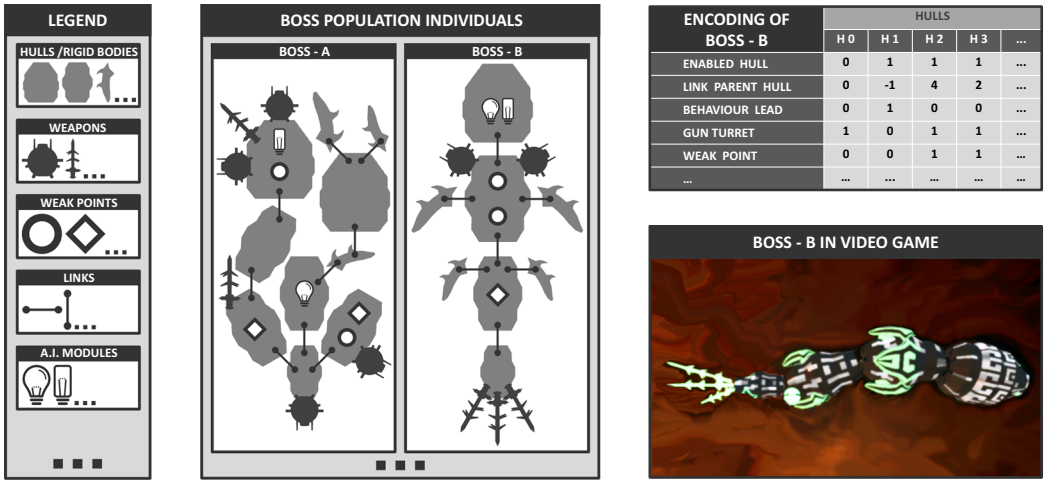


Fig. 4. Boss encoding example in the PCG case study.

with values of 1 and 0, respectively (e.g., weapons or weak points), as well as the relationships between them. For instance, links establish the anatomical hierarchy of the boss.

In addition to the Reference and Primeval Taxa, the initialization of the population involves the input set of final bosses and is complemented with a set of randomly generated encoded bosses.

Fitness. The fitness function used by the evolutionary algorithm included in our approach for this case study is based on the results of simulated combats between a human player (agent) and the boss being evaluated. The idea of including simulations within the fitness function has been successfully used in previous GSE works [14, 19], and, in fact, we use the same fitness as that used in those works. The value given to a boss by this fitness function depends on how close it is to optimal values, in terms of gameplay experience for the player given by the developers with regard to the following aspects: player victory ratio, health level difference between the contenders, and health level left in the case of the player once the combat is over.

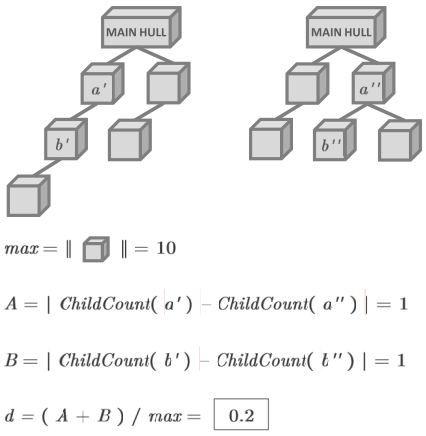
Genetic Distance Measurement for the Phylogenetic Operation. In the context of the PCG case study, the Phylogenetic Operation of our approach calculates the genetic distance as a measure of the anatomical divergence.

The boss anatomy is part of an encoded boss, as shown in the top right part of Figure 4. A boss anatomy consists of a set of enabled hulls, whose inclusion/absence is encoded with binary values, and a hierarchical structure for each hull defined by the index of the hull acting as its parent (or −1, if it has no parent). Algorithm 1 shows the method used to calculate the distance between two bosses. It works as a Euclidean distance-based measure that counts and accumulates differences with regard to the tree-like graphs shown by the anatomies formed by the hull hierarchy of any given pair of bosses. The algorithm consists of a heuristic technique that focuses on the main structural differences found. The method goes through the anatomies that are compared. It highlights the child count differences of those hulls, given a certain hierarchy level of depth, with the highest number of descendants (in the right part of Algorithm 1, a'/a'' for the first level, and b'/b'', for the second one), and obtains a value in the interval [0, 1]. First, using the root hulls of the anatomies compared as starting points, *max* stores the total sum of descendants found recursively for the two anatomies. This value is used in the final step of the algorithm in the quotient that calculates the result. Second, an iterative inspection is conducted, starting with the roots of each level of the anatomies, accumulating in d the child count differences found in the hulls with the most descendants (hence the most representative) of each level ($n1$ and $n2$). Third, once the leaves

Algorithm 1: Distance Algorithm for the Anatomy Graphs of Two Individuals in the PCG Case Study, with an Example

Require: $anatomy_1, anatomy_2$
Ensure: $d = Distance(anatomy_1, anatomy_2)$

$d \leftarrow 0$
$r_1 \leftarrow Root(anatomy_1)$
$r_2 \leftarrow Root(anatomy_2)$
$max \leftarrow DescendantCount(r_1) + DescendantCount(r_2)$
$n_1 \leftarrow r_1$
$n_2 \leftarrow r_2$
while $ChildCount(n_1) > 0$ **or** $ChildCount(n_2) > 0$ **do**
  $n_1 \leftarrow ChildWithMoreDescendants(n_1)$
  $n_2 \leftarrow ChildWithMoreDescendants(n_2)$
  $d \leftarrow d + | ChildCount(n_1) - ChildCount(n_2) |$
end while
$d \leftarrow d/max$



of the tree-like anatomy structures are reached, the number of differences taken into account is divided by *max* in order to calculate the result. Kromaia includes boss characters that resemble an octopus, a serpent, or a space station, among others. For instance, a horse-shaped boss would be closer to a dog-shaped one than an octopus.

Genetic Operations. The individuals of the population are used to create new ones by means of single-point crossover. This operation takes two encoded individuals and selects a random position that is used in those individuals. Then, a new individual is created that includes the genetic material encoded in those individuals, until that position and after that position, respectively. The designation of the individuals that are selected to be the parents of a new generation is done by giving those with a higher fitness value a good probability of being selected [3].

There is another Genetic Operation that may randomly affect (with a certain probability) a boss that is created through crossover: Mutation. Mutations involve the modification of values in encoded individuals. Mutations could improve an individual or make it worse, but they are also valuable as potential introductions to new, unexplored kinds of individuals.

6 Phylogenetics-aware SBSE for the Case of FL

Since our phylogenetics-aware SBSE approach has been designed to be generic and is not only applicable to video game content generation, we also apply our approach to a different software engineering task: FL in software models. The term "feature" refers to a specific functionality or characteristic of a product, and the goal of FL is to identify the elements that are associated with that specific functionality. To this end, it is essential [26, 42] for developers to automatically find the elements of the system's features, especially in industrial contexts where a vast amount of software is accumulated over the years.

The goal of FL in models is to identify the model fragment (i.e., set of model elements) that is associated with that specific functionality. The upper part of Figure 5 depicts a product model excerpt

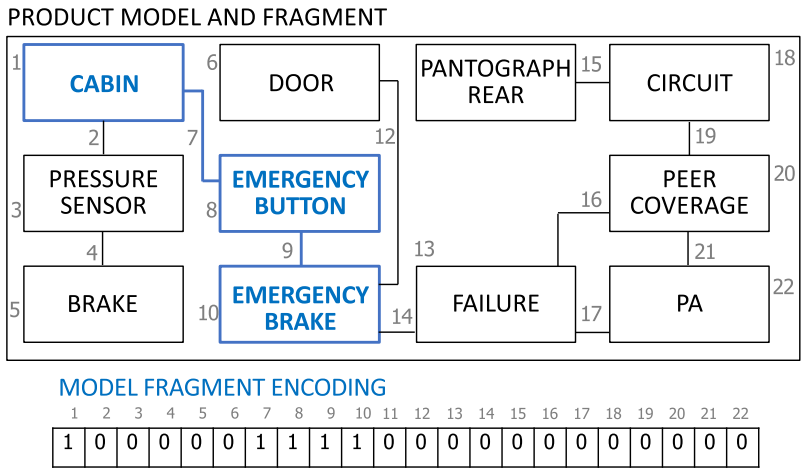


Fig. 5. Example of a product model, model fragment, and encoding in the FL case study.

that is taken from a real-world train of our industrial partner, CAF.[4] CAF is a worldwide leader in train manufacturing that uses models to generate the firmware that controls their trains. For the sake of understandability and legibility (and intellectual property rights concerns), the example in the figure is a simplified equipment-focused subset that uses the same graphical representation for all elements and omits properties of the model elements. The model of Figure 5 presents a braking system scenario where the brake or the emergency brake is activated due to different situations in the train such as the activation of a button in the cabin, and a failure in the circuit that collects energy from the rear pantograph. The figure also presents an example of model fragment (i.e., set of model elements) that implements the feature that comprises the activation of the emergency brake in case of activation of the corresponding button in the cabin.

To address FL in models, evolutionary algorithms have obtained good results [31, 32, 56, 59]. The execution of these approaches requires as input both a search query that describes the feature to be located in natural language and a set of software models where the feature must be located. Then, the evolutionary algorithm iterates over a population of model fragments (i.e., individuals) to generate new model fragments using genetic operations (e.g., the popular crossover and mutation). Each model fragment of the population is assessed by a fitness function (e.g., similitude of a model fragment to the input feature description). If the stop condition is met (usually a time slot, a fixed number of generations, or a trigger value of the fitness), the algorithm will stop returning the ranking of model fragments, which is sorted from the highest to the lowest fitness value.

Since companies often develop products that share a high degree of common functionalities [55], the Phylogenetic Operation of our Phylogenetics-aware SBSE approach can identify the possible lineage shared by the generated model fragments to keep or discard them. Thus, our approach could enhance the results of FL that are obtained using an evolutionary algorithm. In order to apply our Phylogenetics-aware SBSE approach in FL, we use the same elements that were described in Figure 2 as follows:

Input. The SBSE input is both the search query, which is the feature description, and the product models where the model fragment that realizes the feature description must be located. The Phylogenetic Input is: the Reference Taxon that is a model fragment, which is representative of the feature being located; and the Primeval Taxon that is a model fragment, which is the seed of the

[4] www.caf.net/en

feature being located. For example, the description of a feature being located in the context of our industrial partner CAF is: activation of the emergency brake in case of failure in the circuit of the rear pantograph. The Reference Taxon is the model fragment that comprises the following elements of Figure 5: 10, 13, 14, 17, and 22 (since these elements belong to a feature related to the emergency brake). The Primeval Taxon comprises the model element 10 of Figure 5 (since the model fragment of the feature being located may include the emergency brake).

Population. The individuals of the population are model fragments. A model fragment is encoded in a bit string to be easily manipulated. The bit string contains as many positions as elements in the parent product model. Each position in the string has two possible values: 0, if the element does not appear in the fragment; or 1, if the element does appear in the fragment. The lower part of Figure 5 shows the encoding of the model fragment highlighted in the upper part of the figure. Since the model fragment comprises model elements: 1, 7, 8, 9, and 10, only the corresponding positions in the encoding are set to "1."

The initial population of model fragments is generated from the input set of product models. To do this, model elements of a product model are randomly extracted and added to a collection of model fragments. We selected this random technique since it is commonly used in the SBMDE community [32].

Fitness. It assesses the relevance of each model fragment produced in relation to the provided search query (feature description). To do this, we apply LSI [39, 44] to analyze the relationships between each model fragment in the population and the search query. We selected LSI since this technique obtained the best results for FL tasks [81]. LSI constructs vector representations of the query and a corpus of text documents (model fragments in this FL context) by encoding them as a term-by-document co-occurrence matrix. Each row in the matrix corresponds to *terms*, and each column corresponds to *documents*, followed by the *query* in the last column. Each cell of the matrix holds the number of occurrences of a *term* inside a *document* or the *query*. Once the matrix is built, it is normalized and decomposed into a set of vectors using a matrix factorization technique called **Singular Value Decomposition (SVD)** [44]. With SVD, one vector is obtained for each *document* and for the *query*. Finally, the similarities between each *document* (model fragment) and the *query* are calculated as the cosine between their corresponding vectors. As a result, a value between 0 and 1 is obtained for each model fragment. A value closer to 1 denotes a higher degree of similarity between a model fragment and the query.

Genetic Distance Measurement of the Phylogenetic Operation. To calculate the genetic distance between two model fragments (where each model fragment is encoded using a bit string), we use the popular distance measure that also appears within information retrieval: the Hamming distance [16]. This measure obtains a value of 0 (when the model fragments compared are the same) or a value of n that represents the number of different model elements between the compared model fragments.

Genetic Operations. These operations are the same operations as those used in the case of video game content generation (the widespread single-point crossover and random mutation). In order to select those model fragments in which the genetic operations are applied (parents), the fitness value is taken into account. Model fragments with high fitness values will have higher probabilities of being chosen as parents for the next generation of the evolutionary algorithm. This method for selecting the parents (the wheel selection mechanism) is one of the most common choices [3].

7 Evaluation

This section explains the evaluation of our work: the research questions that we aim to answer, the evaluation process that is planned to answer the research questions, and the implementation

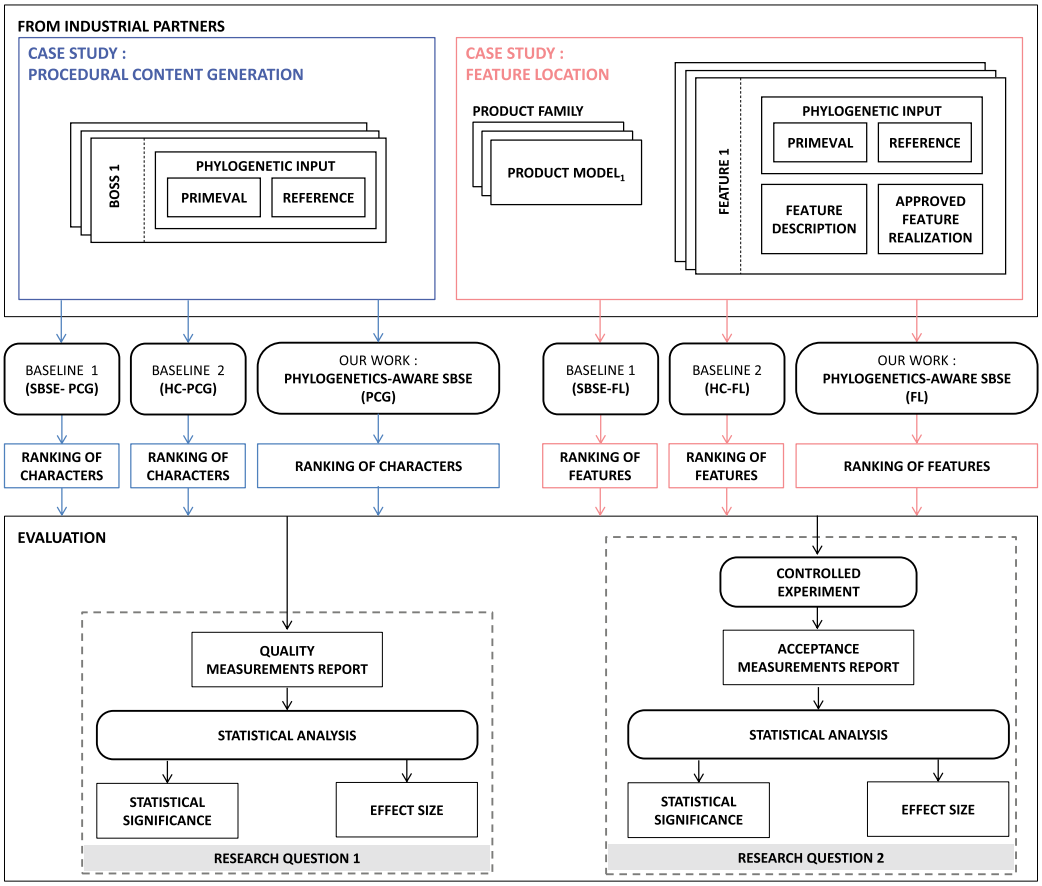


Fig. 6. The evaluation process followed in order to answer the research questions.

details. The evaluation process includes: two industrial case studies, their corresponding baselines, quality and acceptance measurements, and statistical analysis.

7.1 Research Questions

We aim to answer the following research questions:

RQ_1. How much is solution quality influenced using our Phylogenetics-aware SBSE approach compared to the baselines?

RQ_2. How much is the acceptance influenced using our Phylogenetics-aware SBSE approach compared to the baselines?

These research questions deal with the case studies of PCG and FL. In each of these two case studies, we consider both our Phylogenetics-aware SBSE approach and a baseline that uses SBSE without including Phylogenetics.

7.2 Planning and Execution

Figure 6 presents an overview of the process that is planned to answer the two research questions. The evaluation process takes the dataset provided by our industrial partners as input, as the upper part of the figure shows.

For the PCG case study, Kraken Empire provided a dataset that contains the five original final bosses included in the commercial version of Kromaia. The definition of a character of this nature in the video game of the PCG case study involves an average of 1,000 constituents or parts.

For the FL case study, CAF provided a dataset that contains 121 feature descriptions, the 23 product models where the model fragments should be located, and the approved feature realization (i.e., the model fragment that corresponds to each feature) that will be considered to be the ground truth (oracle). Each product model is composed of more than 1,200 model elements on average.

As the middle part of Figure 6 shows, the evaluation process not only includes the execution of our Phylogenetics-aware SBSE for each case study, but also the execution of two corresponding baselines to put the performance of our approach in perspective and to study the impact of the results. Baseline_1 uses the same evolutionary algorithm as our Phylogenetics-aware SBSE approach but without including the Phylogenetic Operation. These SBSE approaches without Phylogenetics have obtained the best results for PCG [14] and FL [28]. Even though the baselines do not include the Phylogenetic Operation, the individuals of the Reference Taxon and the Primeval Taxon are included in their populations to make a fair comparison, that is, both our Phylogenetics-aware SBSE and the baselines have the same seeds to start the initial population. Baseline_2 uses Hill Climbing, which is a technique that has been compared with evolutionary algorithms in the context of SBSE in the past [32]. Hill Climbing is a local search optimization algorithm that is simpler with regard to the operations used than an evolutionary algorithm [67]. Therefore, it can be used to measure the impact of a technique that does not need to manage an evolving, interacting solution candidate population. There are some relevant aspects that separate an evolutionary algorithm (used in our approach and Baseline_1) from Hill Climbing (present in Baseline_2):

— An evolutionary algorithm produces new solution candidates by mixing independent progenitors that are selected for their good genetic value. In contrast, in Hill Climbing, the current best solution candidate is tweaked in order to create a set of adjacent neighbors from which a new, better candidate is selected. If it is found, all of the other neighbors are discarded after that. In Baseline_2, we have included the steepest variant of Hill Climbing, which generates the whole encodable adjacent neighborhood of the current solution candidate in order to compare how that baseline and Baseline_1 explore the solution space in search of good results in terms of thoroughness and effectiveness.
— When Hill Climbing reaches local optima as possible candidates, other potentially interesting or better candidates cannot be reached. Since an evolutionary algorithm mixes the genetic material of the existing candidates in order to open new ways, it could possibly lead to weaker descendants or better and more interesting results because of the risk/opportunity duality given by the promotion of genetic diversity.

The lower part of Figure 6 presents the elements that are planned to answer the two research questions. RQ_1 entails the measurements to report the solution quality that is obtained by both our approach and the baselines. RQ_1 also includes a statistical analysis in order to provide quantitative evidence of the impact of the results and to show whether this impact is significant. RQ_2 involves a controlled experiment to obtain a report of acceptance measurements as well as a statistical analysis. These elements are described in detail in the following subsections.

7.3 Quality Measurements in the PCG Case Study

This subsection explains the quality measurements that were accepted by the research community in previous works [14, 17], which are used to measure the quality of the characters generated for the game used. In the context of this video game, these criteria are calculated taking into account

that the characters produced (bosses) are evaluated by means of a set of simulated duels between a human player and the bosses created.

Completion. In a game, the number of victories achieved by either the player or the boss should be greater than the number of draws. *Completion* is defined as the ratio of victory conclusions over the number of results (the sum of victories and draws):

$$Completion = \frac{Wins}{Duels}, \tag{1}$$

Duration. Duels with an inadequate duration (too long or too short) are not desirable. An optimal value for the duration of a game is relevant for the achievement of game experiences that are neither too difficult to finish nor too trivial. The duration of the game is one of the criteria presented in [6].

According to [14], the optimal time in a duel for a game like the case study is 10 minutes ($T_{Optimal}$), and the maximum accepted time was estimated at around 20 minutes ($2 \cdot T_{Optimal}$). Duels shorter than $T_{Optimal}$ could be too easy, and duels longer than the maximum accepted time make players lose interest. The criterion *Duration* is a measure of the average difference between the duration of each duel (T_d) and its optimal duration ($T_{Optimal}$):

$$Duration = clamp_{[0,1]}\left(1 - \frac{\sum_{d=1}^{Duels} \frac{|T_{Optimal} - T_d|}{T_{Optimal}}}{Duels}\right), \tag{2}$$

Uncertainty. A duel is more engaging for the player when the outcome remains uncertain for as long as possible. *Uncertainty* is maximized when the critical damage amount for the player or the boss (P_d and B_d, respectively) is reached right before the conclusion of the duel:

$$Uncertainty = clamp_{[0,1]}\left(1 - \frac{\sum_{d=1}^{Duels} \frac{T_d - min(P_d, E_d)}{T_d}}{Duels}\right), \tag{3}$$

Killer Moves. KillerMoves is greater when the number of killer moves (K_d by either the player or the boss is higher than the total amount of highlight moves (H_d) performed during a duel. In the video game of the PCG case study, a highlight move is a remarkable action that decreases the health of any of the contenders. In addition, a killer move is an action that causes the health gap between the contenders to reach 30%:

$$KillerMoves = clamp_{[0,1]}\left(1 - \frac{\sum_{d=1}^{Duels} \frac{K_d}{H_d}}{Duels}\right), \tag{4}$$

Permanence. Permanence is higher when the dominance given by significant moves is not quickly reverted. In the video game of the PCG case study, a low *Permanence* would imply that the advantage given by a highlight move or a killer move is cancelled by what the developers define as recovery moves (R_d), which would reduce an existing health level gap between the contenders:

$$Permanence = clamp_{[0,1]}\left(1 - \frac{\sum_{d=1}^{Duels} \frac{R_d}{H_d + K_d}}{Duels}\right), \tag{5}$$

Lead Change. In duels where there are no lead changes the player's interest decreases. In the context of the video game used in the PCG case study, the lead is designated by determining the contender with the highest health level. This criterion is measured by calculating the relation between the moves or actions that make the lead change during a duel (L_d) and the highlight and killer moves occurred.

$$LeadChange = clamp_{[0,1]}\left(\frac{\sum_{d=1}^{Duels} \frac{L_d}{H_d+K_d}}{Duels}\right), \tag{6}$$

7.4 Quality Measurements in the FL Case Study

For each execution of our approach and the corresponding baseline, a ranking of model fragments is obtained as a result of locating a feature. To measure the quality of each solution, we compare the first model fragment (i.e., the model fragment with the highest fitness value) against the oracle, which is considered to be the ground truth. Once the comparison is performed, a confusion matrix is calculated.

The confusion matrix is often used to describe the performance of a classification model on a set of data (the best solution) for which the true values are known (from the oracle). In this case study, a model fragment is obtained as a solution. Since the granularity is at the level of model elements, it is considered to be a classification of the presence or absence of each model element. The confusion matrix organizes the results of the comparison between the model fragment from the oracle and the solution into four categories of values: (1) **True-Positive (TP)** values, model elements that are present in the model fragments of both the solution and the oracle; (2) **False-Positive (FP)** values, model elements that are present in the solution but absent in the oracle; (3) True-Negative values, model elements that are absent in both the solution and the oracle; and (4) **False-Negative (FN)** values, model elements that are absent in the solution but present in the oracle.

From the values in the matrix, it is possible to extract measurements that evaluate the solution quality of our approach and the baseline. Specifically, we derive the following three measurements, which are widely accepted in the software engineering research community [30, 48, 69]: Recall, Precision, and F-measure.

Recall measures the number of elements of the oracle that are correctly retrieved by the proposed solution and is defined as:

$$Recall = \frac{TP}{TP + FN}, \tag{7}$$

Precision measures the number of elements from the solution that are correct according to the oracle and is defined as:

$$Precision = \frac{TP}{TP + FP}, \tag{8}$$

Finally, the *F-measure* corresponds to the harmonic mean of Precision and Recall:

$$F - measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}, \tag{9}$$

Recall values can range between 0% (which means that no single model element from the oracle is present in the model fragment that is obtained as a solution) to 100% (which means that all of the model elements from the oracle are present in the solution model fragment). Precision values can also range between 0% (which means that no model elements from the solution model fragment appear in the oracle) to 100% (which means that all of the model elements from the solution model

fragment appear in the oracle). A model fragment with values of 100% in both Precision and Recall implies that the model fragment is equal to the model fragment of the oracle.

7.5 Acceptance Measurement

This subsection presents the acceptance measure. In order to know the acceptance in the PCG case study, five expert video game developers compared the game bosses generated by our Phylogenetics-aware SBSE approach with bosses generated by the corresponding baseline. In the same way, in the case of FL, three expert software engineers compared the model fragments generated by our approach with the model fragments generated by the related baseline.

The TPB questionnaire [4] is used to compare the level of acceptance. The TPB questionnaire evaluates the most suitable dimensions for the acceptance study in these case studies [64]. The three dimensions are the following: (1) *Attitude* refers to one's degree of favorableness or unfavorableness towards using an artifact; (2) *Subjective norm* defines and measures the degree to which people think that others who are important to them think they should use an artifact; and (3) *Perceived behavioral control* refers to one's perceptions of constraints on the use of an artifact. The TPB questionnaire (available in the Appendix) is composed of 14 questions, and each question is a Likert scale with values between 1 and 7. The comparison processes were performed in the same way for both case studies, but with different artifacts and experts. The artifacts belonged to two types: The artifacts obtained by means of our approach, and those produced by the corresponding baseline. The comparison was performed as follows:

(1) First, artifacts of one type were shown to each expert. Then, the expert was asked to fill out a TPB questionnaire to assess the acceptance of that type of artifact. Next, artifacts of the other type were shown to the expert, and the corresponding TPB questionnaire was filled out. The two types of artifacts (obtained by means of the baseline for the case study and the corresponding version of our approach, respectively) were assigned to the expert randomly to avoid the learning effect.
(2) Finally, a focus group was carried out with the experts to know their opinion about the different types of artifacts.

In the PCG case study, the artifacts compared were video game bosses, and they were shown to game developers, whereas for the FL case study, the artifacts were model fragments, and the experts were software engineers.

7.6 Implementation Details

In order to implement the evolutionary algorithm, we have chosen the parameters that correspond to those settings that are commonly used in the literature [13, 14, 18, 56, 60, 70]. For example, we set the following parameters for the crossover and mutation operations: The top 10% of the individuals of the population combine to form pairs that act as parents by means of crossover, and new individuals may have their genetic material altered with a probability that is inversely proportional to the encoding size. As suggested by Arcuri and Fraser [10] and confirmed in Kotelysanskii and Kapfhammer [41], tuned parameters can outperform default values generally, but they are far from optimal in individual problem instances. Therefore, the focus of this article is not to tune the value of the lineage threshold to improve the performance of the algorithms when applied to a specific problem, but rather to compare the performance of the algorithms in terms of solution quality and acceptance. Thus, we set the parameter to the midpoint between the Reference Taxon and the Primeval Taxon as Section 4.1 described.

Moreover, we executed 30 independent runs (as suggested by Arcuri and Fraser [10]) to take into account the random variation of the evolutionary algorithm in both our Phylogenetics-aware SBSE

approach and the baselines for each boss or feature: 5 (bosses) × 3 (approach and two baselines) × 30 repetitions in the PCG case study, plus 121 (features) × 3 (approach and two baselines) × 30 repetitions in the FL case study, for a total of 11,340 independent runs.

In this article, we focus on the solution quality (i.e., obtaining a solution that is more similar to the expected solution) instead of algorithm speed (or search effort). After running some prior tests for our Phylogenetics-aware SBSE approach and the corresponding baselines in the two case studies in order to determine the time to converge (and adding a margin to ensure convergence), we allocated a fixed amount of wall-clock time (60 minutes) to stop the execution in the PCG case study since the tests showed that, after 40 minutes, no further improvements occurred, and an additional 20 minutes were added to ensure convergence, in the same manner as previous works [14]. Similarly, in the case of FL, we monitored the fitness of the best solution over several generations. There was no significant improvement from 47 seconds on average, so the algorithm can be considered converged. Hence, we allocated 80 seconds (adding a margin to ensure convergence as in previous FL works [32]) to stop the execution in the FL case study. For the PCG case study, Baseline_1 and Baseline_2 execute an average of 1,345 and 933 generations, respectively, and our approach is capable of executing an average of 347 generations. In the FL case study, Baseline_1 and Baseline_2 are capable of executing an average of 161,024 and 124,440 generations, respectively, whereas our approach is capable of executing an average of 10,355 generations. We performed the execution using an Intel Core i7-6700HQ processor (clock speeds 2.6 GHz and four cores) and 16 GB of RAM. The computer was running Windows 10 as the hosting Operative System and the Java(TM) SE Runtime Environment (build 1.8.0_331).

To implement the Phylogenetic Tree inference used by the Phylogenetic Operation, we used an implementation of T-Rex [15], which includes the Neighbor-Joining method, among others, and receives Genetic Distance Matrices as input.

In order to implement the evolutionary algorithm in the PCG case study, we used Eclipse [33] as the main IDE and the Java programming language. Additionally, the boss encoding data were obtained from SDML models (SDML is a Domain-Specific Language for shooting games [14]) provided by Kraken Empire for each of the bosses that are commercially released in the video game Kromaia.

To implement the evolutionary algorithm in the FL case study, we used the Eclipse Modeling Framework [73] to manipulate the models. We also used OpenNLP [1] and the English (Porter2) as the techniques to process the natural language of the models and feature descriptions. The fitness function (LSI) was implemented using the Efficient Java Matrix Library [2]. The genetic operations are built upon the Watchmaker Framework for Evolutionary Computation [27].

In the PCG case study, the developers of Kromaia provided the same boss definition files that are included with the commercial version of the product.

Our open source implementation of the approach, examples from the PCG dataset, and the CSV files used as input in all of the statistical analyses are available here: https://doi.org/10.5281/zenodo.13885161. The FL dataset is limited by confidentiality agreements that we have with the industrial partner. The trains of the dataset are currently operating and under maintenance contracts or will be released in the near future.

8 Results

This section presents the results corresponding to RQ_1 and RQ_2 for each case study.

8.1 Results in the PCG Case Study

For this case study, the results compare the video game bosses produced by the PCG version of our Phylogenetics-aware SBSE approach and the corresponding baselines. The boxplots in Figure 7

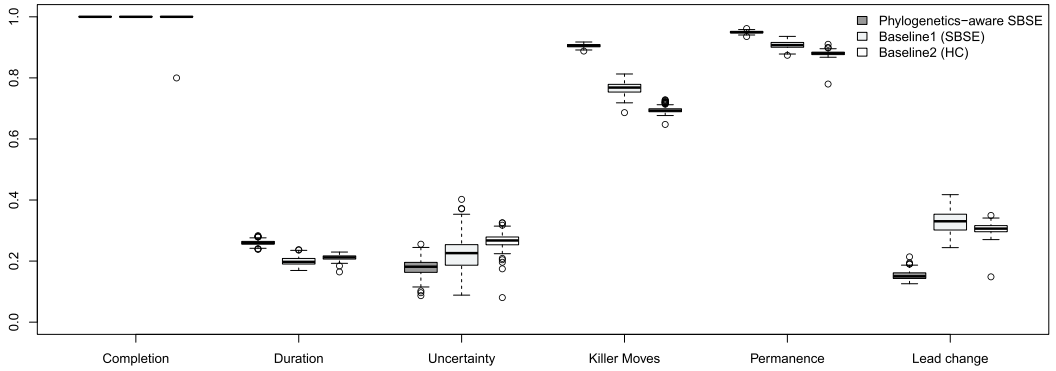


Fig. 7. Quality measurements achieved in the PCG case study by our approach and baselines.

depict the distribution of the mean results achieved by the five video game bosses produced in our Phylogenetics-aware SBSE approach and the two baselines for each quality measurement (completion, duration, uncertainty, killer moves, permanence, and lead change).

For each of the bosses, Table 1 shows cells with the mean values obtained for every measurement once all of the runs conclude in our approach and baselines. Besides the six quality measurements used, the table shows an additional column with the overall average result of those six measurements for each of the bosses in our approach and the baselines. The results show that the average values obtained for the bosses in four of the six quality measurements are equivalent or higher when our approach is applied, as highlighted in gray in Table 1. Therefore, taking into account the overall results present in Table 1, the use of Phylogenetics-aware SBSE does not prevent the approach from obtaining quality values that are comparable to those achieved by the baselines. Additionally, as shown by the results, there is a measurement (Completion) for which the values achieved are maximum for every approach and boss studied. Those values stem from the intrinsic dynamics of the duels in Kromaia: the effectiveness of the weapons used and the challenge for an average player to evade combat lead to duel completion maximization within acceptable time limits.

RQ_1 Answer (Quality). The overall quality measurement (seventh column in Table 1) shows an improvement that reaches 2% and 7% over $Baseline_1$ for the fourth and fifth bosses, respectively. For the rest of the bosses, it is 2% lower in average. In the case of $Baseline_2$ and the overall quality measurement, there is an improvement of 3%, 1%, and 10% for the first, fourth and fifth bosses, respectively. For the two other bosses, it is 4.5% lower in average. With regard to the six specific quality measures, there are three of them for which our approach always improves the average results obtained by the baselines: Those measures are highlighted in Table 1.

The results must be properly compared and analyzed using statistical methods in order to determine whether the differences between our Phylogenetics-aware SBSE approach and the baselines are significant. To do this, we aim to provide formal evidence by following the guidelines presented in [9].

A statistical test should then be run to assess whether there is enough empirical evidence to claim that there are differences between the baselines and our approach. It is accepted by the research community that a $p\text{-}Value$ under 0.05 implies statistical significance [9].

Our analysis requires the usage of non-parametric techniques since our data might not follow a normal distribution. There are several tests for analyzing this kind of data. However, the Quade test

Table 1. PCG Case Study: Mean Values and Standard Deviations Obtained for Quality Measurements

	Completion	Duration	Uncertainty	Killer Moves	Permanence	Lead Change	Overall
				Phylogenetics-aware SBSE (PCG)			
Boss 1	1 ± 0	0.18 ± 0.01	0.13 ± 0.05	0.93 ± 0.01	0.97 ± 0.01	0.11 ± 0.02	0.55 ± 0.01
Boss 2	1 ± 0	0.06 ± 0.01	0.13 ± 0.06	0.89 ± 0.01	0.95 ± 0.01	0.10 ± 0.03	0.52 ± 0.01
Boss 3	1 ± 0	0.24 ± 0.01	0.11 ± 0.05	0.96 ± 0.004	0.98 ± 0.003	0.10 ± 0.01	0.56 ± 0.01
Boss 4	1 ± 0	0.37 ± 0.03	0.13 ± 0.06	0.88 ± 0.02	0.93 ± 0.01	0.23 ± 0.05	0.59 ± 0.01
Boss 5	1 ± 0	0.46 ± 0.02	0.40 ± 0.08	0.87 ± 0.01	0.91 ± 0.01	0.23 ± 0.05	0.64 ± 0.02
				Baseline (PCG)			
Boss 1	1 ± 0	0.18 ± 0.02	0.28 ± 0.12	0.72 ± 0.05	0.89 ± 0.02	0.33 ± 0.05	0.57± 0.02
Boss 2	1 ± 0	0.18 ± 0.04	0.20 ± 0.12	0.81 ± 0.04	0.94 ± 0.03	0.33 ± 0.08	0.58 ± 0.02
Boss 3	1 ± 0	0.24 ± 0.04	0.22 ± 0.14	0.76 ± 0.04	0.90 ± 0.02	0.37 ± 0.09	0.58 ± 0.03
Boss 4	1 ± 0	0.17 ± 0.02	0.23 ± 0.07	0.80 ± 0.03	0.92 ± 0.01	0.28 ± 0.05	0.57 ± 0.02
Boss 5	1 ± 0	0.21 ± 0.02	0.21 ± 0.14	0.74 ± 0.06	0.90 ± 0.04	0.34 ± 0.10	0.57 ± 0.02
				Baseline (Hill Climbing)			
Boss 1	1 ± 0	0.2 ± 0.01	0.12 ± 0.02	0.65 ± 0.02	0.86 ± 0.01	0.29 ± 0.02	0.52 ± 0
Boss 2	1 ± 0	0.19 ± 0.01	0.46 ± 0.06	0.69 ± 0.02	0.89 ± 0.02	0.29 ± 0.03	0.59 ± 0.01
Boss 3	0.99 ± 0.07	0.23 ± 0.03	0.35 ± 0.07	0.68 ± 0.06	0.86 ± 0.06	0.33 ± 0.04	0.58 ± 0.04
Boss 4	1 ± 0	0.3 ± 0.02	0.3 ± 0.05	0.66 ± 0.02	0.86 ± 0.01	0.34 ± 0.03	0.58 ± 0.01
Boss 5	1 ± 0	0.14 ± 0.01	0.09 ± 0.05	0.78 ± 0.01	0.93 ± 0.01	0.28 ± 0.04	0.54 ± 0.01

The measurements for which the average boss results obtained with phylogenetics-aware SBSE are comparable or better than those produced by the baselines, are highlighted.

Table 2. PCG Case Study: Holm's Post Hoc p-Values and $\hat{A}_{12}$ Effect Size

	Completion	Duration	Uncertainty	Killer Moves	Permanence	Lead Change	Overall
				Holm's post hoc p-Values			
Phylogenetics-aware SBSE vs. Baseline$_1$ (SBSE)	–	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	3.5×10^{-5}
Phylogenetics-aware SBSE vs. Baseline$_2$ (HC)	0.32	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$
				$\hat{A}_{12}$ effect size			
Phylogenetics-aware SBSE vs. Baseline$_1$ (SBSE)	0.5	1	0.2323	1	1	0	0.6272
Phylogenetics-aware SBSE vs. Baseline$_2$ (HC)	0.5025	1	0.0137	1	1	0.0027	0.9801

is more powerful than other tests when working with real data [24, 35] and when the number of compared algorithms is low (no more than four or five algorithms). Moreover, the Quade test has also been used by previous SBSE approaches [46]. For each quality measurement, we record a Quade test *p-Value* by considering the mean results achieved by the five video game bosses produced in our Phylogenetics-aware SBSE approach and the two baselines. The Quade test p-Values are smaller than 0.05 for all of the quality measurements except Completion. Hence, we can state that there are differences between our approach and the two baselines for all of the quality measurements except for completion.

We also perform an additional Holm's post hoc analysis to individually compare our approach to each of the baselines, determining whether statistically significant differences exist. The upper part of Table 2 shows the p-Values of Holm's post hoc analysis after comparing the mean results achieved by the five video game bosses produced in our Phylogenetics-aware SBSE approach with each baseline for each quality measurement.

RQ_1 Answer (p-Values). The p-Values of Holm's post-hoc analysis for the pairwise comparison of our approach and each baseline in the overall is lower than the corresponding significance threshold value (0.05), so the difference is significant.

To determine how much the performance is influenced by using our Phylogenetics-aware SBSE approach, it is important to assess (through effect size measures) how much the solution obtained by our approach improves the quality of the solution obtained by the baseline (the magnitude of the improvement) in the case study. For non-parametric effect size measurements, we use Vargha and Delaney's $\hat{A}_{12}$ [78]. $\hat{A}_{12}$ measures the probability that running one approach yields higher values than running another approach. With the $\hat{A}_{12}$ statistic, the approaches are compared in pairs (A vs. B).

If the $\hat{A}_{12}$ statistic obtains a value greater than 0.5, the comparison will be in favor of A. If the $\hat{A}_{12}$ statistic obtains a value lower than 0.5, the comparison will be in favor of B and 1-$\hat{A}_{12}$ will be used to interpret the magnitude of effect. According to the guidelines for interpreting $\hat{A}_{12}$ values [78], the $\hat{A}_{12}$ value of 0.5 means that the two approaches are equivalent (no effect). The $\hat{A}_{12}$ value of 0.56 means a small effect in the magnitude of improvement, 0.64 means a medium effect, and 0.71 means a big effect. For example, a value of $\hat{A}_{12} = 0.67$ means that, in 67% of the runs, A would obtain better results than A and that the effect on the magnitude of improvement is medium.

The lower part of Table 2 shows the $\hat{A}_{12}$ values considering the mean results achieved by the five video game bosses produced in our Phylogenetics-aware SBSE approach and each baseline for all of the quality measurements. As the $\hat{A}_{12}$ values of the table show, our approach is equal or outperforms Baseline_1 (SBSE) in four quality measurements (Completion, Duration, Killer Moves, and Permanence). Specifically, in three of these quality measurements (Duration, Killer Moves, and Permanence), the $\hat{A}_{12}$ values (1) show a big effect on the magnitude of improvement (i.e., our approach outperforms Baseline_1 in 100% of the runs). This is observed in the same manner for those measurements in the case of Baseline_2 as well.

> *RQ_1 Answer ($\hat{A}_{12}$).* The $\hat{A}_{12}$ value in the overall results between Phylogenetics-aware SBSE and Baseline_1(SBSE) is 0.6272. This indicates that our approach (which includes Phylogenetics) obtains better results than Baseline_1(SBSE) in 62.72% of the runs. The $\hat{A}_{12}$ value in the overall results between Phylogenetics-aware SBSE and Baseline_2 (HC) is 0.9801. This indicates that our approach obtains better results in 98.01% of the runs.

In the context of RQ_2, the boxplots in Figure 8 depict the distribution of the results that are obtained from the five expert video game developers for each of the 14 questions of the TPB questionnaire in the PCG case study. The bottom part of the figure highlights the dimension to which each question belongs (e.g., Questions 1–5 belong to the *Attitude* dimension).

The value for Perceived Behavioral Control, Subjective Norm, and Attitude for each expert are represented by the mean score calculated for each respective measure. For example, the Attitude score for Expert 1 corresponds to the average of his responses to questions 1 through 5, as these questions specifically assess Attitude. These values are calculated in the same way for both case studies (Procedural Content Generator and FL).

Table 3 shows the mean values that are obtained by each expert using our approach and baselines considering each dimension of the TPB questionnaire: *Attitude* (Questions 1–5), *Subjective norm* (Questions 6–8), and *Perceived behavioral control* (Questions 9–14). Also, the bottom rows of the table show the mean values and standard deviations considering all of the experts as well as the mean values and percentages for acceptance.

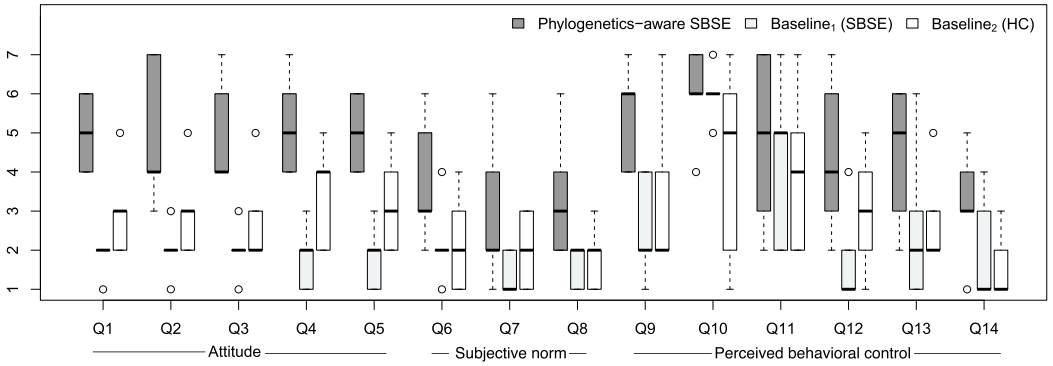


Fig. 8. Results of the TPB questionnaire in our approach and baselines in the PCG case study.

RQ_2 Answer (Acceptance). Our Phylogenetics-aware SBSE approach obtains an improvement over Baseline_1 and Baseline_2, respectively, of 30.15% and 23.34% considering the mean value of the three dimensions in our approach (4.38), in Baseline_1 (2.27), and in Baseline_2 (2.74).

In order to determine whether the differences are significant, we applied the steps shown in [77] to decide the statistical analysis strategy. First, we studied the normality of the samples. Following the recommendations of [52] for small sample size, we used the Shapiro–Wilk test. The test showed that the samples follow the normal distribution. Then, we applied parametric methods: an ANOVA test where *Attitude*, *Subjective norm*, and *Perceived behavioral control* are the dependent variables and the approach (Phylogenetics-aware SBSE, Baseline_1,or Baseline_2) is the independent variable.

RQ_2 Answer (ANOVA Test). The ANOVA test shows that the *p-Value* is greater than 0.05 for *Subjective norm* (0.067), *Perceived behavioral control* (0.089), and is less than 0.05 for *Attitude* (0.002), which means that there is significant difference for the Attitude dimension of acceptance. We conducted a post-hoc test to determine the significant difference between the three approaches for the Attitude dimension. We used the Tukey test as recommended by [77] when an ANOVA is applied for statistical analysis. The results indicate that there are significant differences (p-value less than 0.005), between Phylogenetics-aware SBSE and Baseline_1 ($p = 0.001$) and between Phylogenetics-aware SBSE and Baseline_2 ($p = 0.031$). In contrast, there is no significant difference between Baseline_1 and Baseline_2 ($p = 0.211$).

In order to know how much acceptance is influenced using our approach compared to the baselines, we calculate the effect size using Eta-squared. We use the Eta-squared value following the recommendations of [23] for the ANOVA test. The value that is obtained as a result of multiplying the Eta-squared value by 100 indicates the percentage of variance in the dependent variable explained by the independent variable [76]. The interpretation is the following: between 0.01 and 0.06 is a small effect; between 0.06 and 0.14 is a medium effect; and more than 0.14 is a large effect [63].

Table 3. PCG Case Study: Data and Means for Acceptance

	Phylogenetics-aware SBSE			**Baseline$_1$ (SBSE)**			**Baseline$_2$ (HC)**		
	AT	SN	PBC	AT	SN	PBC	AT	SN	PBC
Expert 1	6.20	6.00	5.00	2.60	2.67	3.17	2.60	1.66	1.50
Expert 2	4.00	4.00	4.33	2.00	2.00	3.83	2.40	1.33	3.00
Expert 3	4.40	2.33	4.50	1.80	1.67	2.83	3.40	3.33	4.00
Expert 4	6.60	2.67	6.00	1.00	1.00	3.50	5.00	1.00	5.33
Expert 5	4.00	2.00	3.67	2.00	2.00	2.67	2.00	2.66	2.00
Mean	5.04	3.40	4.70	1.88	1.73	3.20	3.08	1.99	3.17
Dev standard	1.26	1.64	0.87	0.58	0.64	0.48	1.19	0.97	1.54
Mean Acceptance	4.38 (62.57%)			2.27 (32.42%)			2.74 (39.23%)		

AT, attitude; PBC, perceived behavioral control; SN, subjective norm.

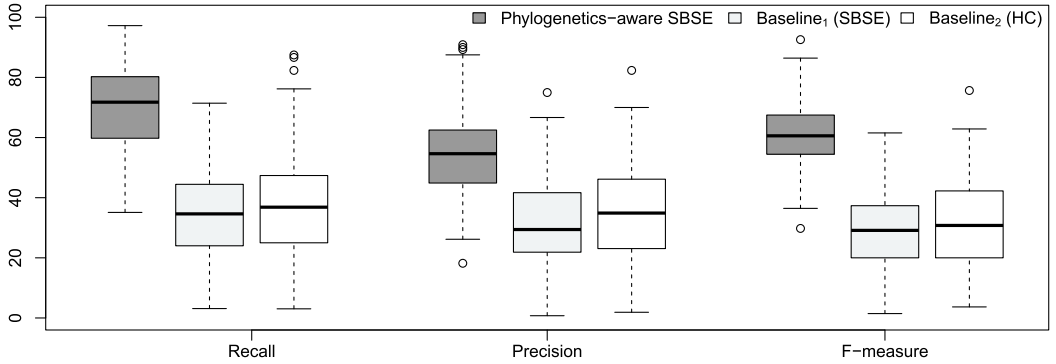


Fig. 9. Quality measurements achieved in the FL case study by our approach and baselines.

RQ_2 Answer (Eta-Squares). The Eta-squared values are 0.656 for *Attitude*, 0.332 for *Subjective norm*, and 0.363 for *Perceived behavioral control*. These results for the effect size show that the magnitude of the differences is large for the three dimensions of acceptance, in favor of Phylogenetics-aware SBSE.

8.2 Results in the FL Case Study

Figure 9 depicts the distribution of the FL results achieved in our Phylogenetics-aware SBSE approach and the baselines in boxplots for all of the quality measurements (Recall, Precision, and F-measure). The upper part of Table 4 shows the mean values and standard deviations of Recall, Precision, and F-measure. The best values for each measure obtained in our Phylogenetics-aware SBSE approach are highlighted in gray in the table.

RQ_1 Answer (Quality). The results reveal that our approach (using the Phylogenetic Operation in the FL case study) outperforms the baselines in the three quality measures (Recall, Precision, and F-measure). Our approach obtains 60.20% in F-measure, whereas Baseline$_1$ (SBSE) obtains 28.97%, which implies an improvement of up to 31.23% over Baseline$_1$. Baseline$_2$ (HC) obtains 31.51%, which means that the improvement over Baseline$_2$ reaches 28.69%.

Table 4. FL Case Study: For Each Measure, the Mean Results and Standard Deviations of the Features, Holm's Post Hoc p-Values and $\hat{A}_{12}$ Effect Size

	Recall	Precision	F-measure
	Mean ± (σ)		
Phylogenetics-aware SBSE (our approach)	70.69 ± 13.87	55.01 ± 14.72	60.20 ± 11.18
Baseline$_1$ (SBSE)	34.32 ± 14.43	30.96 ± 15.35	28.97 ± 12.43
Baseline$_2$ (HC)	36.69 ± 18.08	35.20 ± 17.21	31.51 ± 14.78
	Holm's post hoc p-Values		
Phylogenetics-aware SBSE vs. Baseline$_1$ (SBSE)	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$
Phylogenetics-aware SBSE vs. Baseline$_2$ (HC)	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$	$\ll 2 \times 10^{-16}$
	$\hat{A}_{12}$ effect size		
Phylogenetics-aware SBSE vs. Baseline$_1$ (SBSE)	0.9657	0.8718	0.9698
Phylogenetics-aware SBSE vs. Baseline$_2$ (HC)	0.9295	0.8060	0.9376

As described in the PCG case study, we perform the Quade test and the Holm's post hoc analysis. We record a Quade test *p-Value* by considering the results achieved in each quality measurement (recall, precision, F-measure) in our Phylogenetics-aware SBSE approach and in each of the two baselines. For all of the quality measurements, the Quade test *p-Value* is $\ll 2.2 \times 10^{-16}$ (smaller than the threshold value 0.05). Hence, we can state that there are differences between our approach and the two baselines in all of the quality measurements. The middle part of Table 4 shows the Holm's post hoc p-Values that are obtained as a result of comparing our approach with each baseline.

RQ$_1$ Answer (p-Values). All Holm's post hoc p-Values are lower than the corresponding significance threshold value (0.05) for all of the quality measurements when our approach is compared to Baseline$_1$ (SBSE) and to Baseline$_2$ (HC), so the differences are significant.

In order to know how much performance is influenced in terms of the solution quality of our approach over the baseline, we use the $\hat{A}_{12}$ statistic as used in the PCG case study. The lower part of Table 4 shows the $\hat{A}_{12}$ values that are obtained by comparing the results of our approach with each baseline in the three performance measures (recall, precision and F-measure).

RQ$_1$ Answer ($\hat{A}_{12}$). According to the $\hat{A}_{12}$ values, our approach outperforms the baselines with a large effect on the magnitude of improvement in all of the quality measurements. In 96.98% and 93.76% of the runs, our approach obtains better results in F-measure than Baseline$_1$ (SBSE) and Baseline$_2$ (HC), respectively.

With regard to RQ$_2$, the boxplots in Figure 10 depict the distribution of the results that are obtained from the three expert software engineers for each of the 14 questions of the TPB questionnaire in our approach, Baseline$_1$ (SBSE), and Baseline$_2$ (HC) of the FL case study.

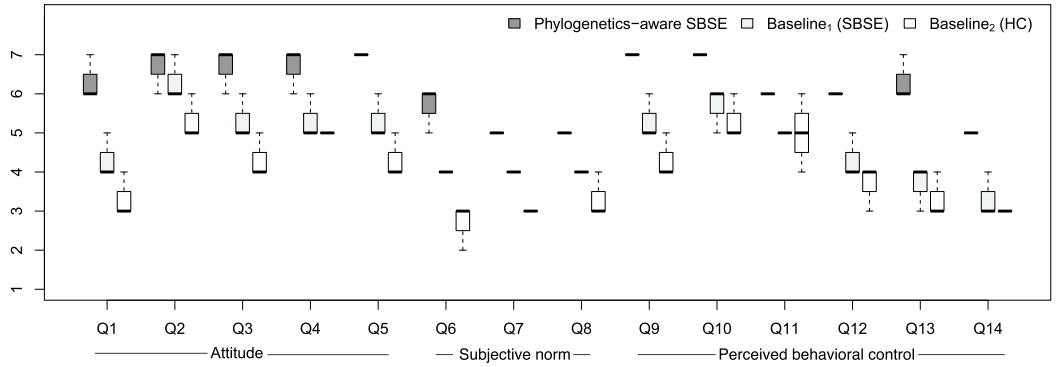


Fig. 10. Results of the TPB questionnaire in our approach, baseline$_1$ (SBSE) and baseline$_2$ (HC) in the FL case study.

Table 5. FL Case Study: Data and Means for Acceptance

	Phylogenetics-aware SBSE			**Baseline$_1$ (SBSE)**			**Baseline$_2$ (HC)**		
	AT	SN	PBC	AT	SN	PBC	AT	SN	PBC
Expert 1	6.80	5.33	6.33	5.80	4.00	4.33	4.80	2.66	4.00
Expert 2	7.00	5.33	6.16	5.20	4.00	4.50	4.20	3.33	4.16
Expert 3	6.20	5.00	6.16	5.00	4.00	4.83	4.40	3.00	4.16
Mean	6.66	5.22	6.21	5,33	4.00	4.55	4.47	2.99	4.11
Dev standard	0.41	0.19	0.01	0,42	0.00	0.25	0.30	0.33	0.09
Mean Acceptance	6.03 (86.14%)			4.62 (66.00%)			3.85 (55.00%)		

AT, attitude; PBC, perceived behavioral control; SN, subjective norm.

Table 5 shows the mean values that are obtained by each expert in our approach and baseline in the FL case study considering each dimension of the TPB questionnaire: Attitude (Questions 1–5), Subjective norm (Questions 6–8), and Perceived behavioral control (Questions 9–14). The bottom rows of the table show the mean values and standard deviations considering all of the experts as well as the mean values and percentages for acceptance.

RQ_2 Answer (Acceptance). Our phylogenetics-aware SBSE approach achieves an improvement over Baseline$_1$ and Baseline$_2$ of 20.14% and 31.14%, respectively, considering the mean value of the three dimensions in our approach (6.03), in Baseline$_1$ (4.62), and in Baseline$_2$ (3.85).

In order to determine whether the differences are significant, we applied the steps shown in [77] to decide the statistical analysis strategy. First, we studied the sample normality. Following the recommendations of [52] for small sample size, we employed the Shapiro–Wilk test. The test showed that the samples might not follow the normal distribution. Then, according to [77], we applied a non-parametric method: the Kruskal–Wallis test. In the Kruskal–Wallis test, the Attitude, Subjective norm, and Perceived behavioral control are the dependent variables and the approach (Phylogenetics-aware SBSE, baseline$_1$,or baseline$_2$) is the independent variable.

RQ_2 Answer (Kruskal–Wallis). The Kruskal–Wallis test shows that the *p-Value* is less than 0.05 for two dimensions of acceptance: *Attitude* (0.027) and *Perceived behavioral control* (0.026). On the other hand, for the *Subjective norm* dimension, the significance value was greater than 0.05 (0.054). Thus, there are significant differences between our approach and the baseline in two dimensions of acceptance. We conducted a post-hoc test to determine the significant difference between the three approaches for *Attitude* and *Perceived behavioral control* dimensions. We used Wilcoxon test for pairwise comparison as recommended by [77]. The results indicate that there are significant differences (p-value less than 0.05), between Phylogenetics-aware SBSE and $Baseline_2$ for *Perceived behavioral control* ($p = 0.020$) and *Attitude* (0.022). In contrast, there is no statistical significance among the other pairwise comparisons.

In order to know how much acceptance is influenced using our approach compared to the baselines, we calculate the effect size using Eta-squared. We use the Eta-squared value following the recommendations of [76] for the Kruskal–Wallis test.

RQ_2 Answer (Eta-Squares). The Eta-squared values are 0.876 for *Attitude*, 0.640 for *Subjective norm*, and 0.886 for *Perceived behavioral control* . These results for the effect size show that the magnitude of the differences are large for the three dimensions of acceptance, in favor of Phylogenetics-aware SBSE.

9 Discussion

Our results have confirmed that utilizing our Phylogenetic Operation pays off in SBSE for the PCG and FL cases. We analyzed the results in order to understand why Phylogenetics-aware SBSE is powerful enough to outperform the results of the baselines.

The results provided by our approach are closer to the results expected by the domain experts. This is because our Phylogenetic Operation mitigates problems that are present in the baselines. For instance, in the case of PCG, it could be thought at first that generating quality content is the key to success. However, in the context of video games, the alignment of the content with the vision of the developers is critical. The quality content that does not fit the rest of the content of the video game is not useful. If lineage is promoted, as our Phylogenetic Operation does, the results are more aligned with the vision of the developers. This could mean the difference that makes them include or discard the content in their video games.

Another example would be that, in the case of FL, both “PLC” and “COSMOS” are terms that refer to an on-board programmable controller. The baselines fail to establish these relations among the terms. Therefore, if the feature description does not include the same terms that are included in the models (vocabulary mismatch), the baselines will not include model elements with those terms even if they are part of the expected solution. In contrast, our approach includes model elements even if their terms do not appear in the feature description, thanks to the Phylogenetic Operation, which uses the Primeval and Reference Taxa. The same phenomenon occurs when the feature description omits relevant terms (tacit knowledge).

Furthermore, we ran a focus group to obtain feedback from the five domain experts in the field of GSE for the PCG case study. This focus group dealt with the following open questions: (1) What do you think of the bosses that are generated with the two approaches? and (2) Why would you choose the bosses that are generated with one approach over the other?

The domain experts of the PCG case study focus group stated that the Phylogenetics-aware SBSE is better than the baselines, even if they consider that, in some cases, neither approach produced optimal results. They even preferred those results that were produced by our Phylogenetics-aware SBSE approach with quality values that were lower than those generated by the baselines. In fact, they mentioned that such results, which had lower quality but were aligned with their expectations, could be used as secondary characters in video games, whereas the results produced by the baselines would not be used regardless of their quality. In order to improve the results provided by our Phylogenetics-aware SBSE, the experts discussed which elements they considered to be neglected in the bosses generated. This kind of discussion could be an opportunity for a refinement of the distance measurement; then, the approach could be iterated in the hope that the results improve.

The two baselines search for good results relying on a fitness criterion and quality measures that do not contain information regarding developer vision alignment through lineages, which our approach does by leveraging Phylogenetics. In addition, Baseline_2 does not promote genetic diversity, which leads to solution candidates that are too similar to the original content studied.

In addition, we also ran a focus group to acquire feedback from the three domain experts of our industrial partner in the FL case study. Specifically, the focus group was made up of the following open questions: (1) What do you think of the features that are located with the two approaches?; and (2) Why would you choose the features that are located with one approach over the other?

The domain experts stated that automatically obtaining model fragments from the feature description is beneficial to perform FL tasks. Although the domain experts acknowledge that all of the model fragments were incomplete and/or included mistakes, there was a consensus among the domain experts that the model fragments that were obtained by the Phylogenetics-aware SBSE approach were more complete and included more relevant model elements than those obtained by the baseline. Another aspect that the domain experts highlighted as an advantage of the approach is that it not only provided a feature description as input, but it also provided a Primeval and a Reference model fragment. These new inputs prevent the resulting model fragments from containing model elements that are not related to the feature being located just because these model elements share terms with the feature description.

For example, in order to locate a feature that is related to the emergency brake, the baseline includes the brake model element without the emergency properties in the model fragment that is obtained as a result because the term "Brake" is shared. This is not the case with our Phylogenetics-aware SBSE approach, since the Phylogenetic Operation of our approach discarded those model fragments with the "Brake" model element (and keeps model fragments with the "Emergency Brake" model element). For this reason, the domain experts mentioned that they would choose the model fragments that are located by means of the Phylogenetics-aware SBSE approach instead of the model fragments that are located using the baseline.

10 Threats to Validity

In this section, we describe the mitigated threats to validity or threats that we could not avoid. We use the classification of threats to validity for the case study research in software engineering of [66]. This classification distinguishes the following four aspects of validity:

Construct Validity. This aspect of validity reflects to what extent the operational measures that are studied really represent what the researcher has in mind and what is investigated according to the research questions. The Author bias threat appears if the researchers that define the artifacts can subjectively influence the obtainment of the results that they are looking for. To mitigate this threat regarding acceptance, the well-known TPB questionnaire was used. With regard to the quality measurements, we have used measurements used in previous works [14, 17, 30]. The Mono-operation bias threat appears when the experiment includes, for instance, a single treatment.

Our work was affected by this threat since we worked with only two case studies in two domains. The generalization of results to other contexts should be made with caution.

Internal Validity. This aspect of validity is of concern when causal relations are examined. When the researcher wants to know whether one factor affects a factor being investigated, there is a risk that the investigated factor is also affected by a third factor. The Selection threat appears when outcomes of the experiment may depend on the type of subjects. With regard to acceptance, the work was affected by this threat since all of the subjects were recruited based on their work experience. Work history or jobs performed by the subjects may also influence the results. The Instrumentation threat is the effect caused by the artifacts that are used for the execution of an experiment. With regard to the quality measurements, this threat was mitigated by using data sets from both the PCG case study and the FL case study, which were extracted from real contexts.

External Validity. This aspect of validity is concerned with to what extent it is possible to generalize the findings, and to what extent the findings are of interest to other people outside the investigated case. The Statistical Power threat appears when the data is not enough to generalize results. With regard to the acceptance study, the number of subjects is not sufficient to generalize results. However, it is important to note that the role of the subjects (software engineers and game developers) makes an interesting contribution in an area where most experiments are conducted with students [29]. The Influence of the Domain threat appears when the findings depend on a specific domain. This article is affected by this threat because it analyzes two specific examples from two domains. To increase generalization, it would be interesting to discuss other examples and other domains.

Reliability. This aspect is concerned with to what extent the data and the analysis are dependent on the specific researchers. Hypothetically, if another researcher conducted the same study later on, the result should be the same. The Data Collection threat appears when data collection is not done in the same way throughout the different sessions. This threat was minimized because the steps were identical in the comparison between the approaches, and it is a replicable process. The Completion Data threat appears when there is some missing data after the data collection process. This threat was mitigated because each of the data captures and data processing performed by one researcher were checked and validated by another researcher.

11 Conclusion

Phylogenetics belongs to the field of Biology and is intended to study the relationships between living or fossil organisms. Our research of the literature does not reveal past works that have applied Phylogenetics to the field of Software. This work harnesses Phylogenetics in order to make it beneficial for Software Engineering in SBSE cases like the PCG and FL subfields from our case studies. In fact, one of our contributions is a novel Phylogenetic Operation, which proposes the idea of lineage alignment to produce better results in SBSE. Our results show that, besides improving solution quality, it is possible to improve the acceptance by domain experts of the solutions generated. At least in the cases like PCG and FL, most past works focus on solution quality and neglect solution acceptance. Solution acceptance may make the difference regarding the use of results in industrial environments, as emphasized by the focus groups which we have run.

We think that this work provides opportunities for the Software Engineering research community. In the context of the SBSE community, our Phylogenetic Operation could be the first of many others to come. We are looking forward to seeing how a new catalog of Phylogenetic operations flourishes when the community continues the development of Phylogenetic ideas for their use in Software. In addition, our ideas could be developed and used in other stages of the process: in the input, helping the domain expert choose the seeds that the algorithm is fed with; in the output,

assisting with decision making; or in the definition of objectives in the fitness function. Besides that, we do not think that the potential is constrained to SBSE. Other Software Engineering research communities could take inspiration from this work in order to explore new ways of re-engineering SPL or Repository mining. We hope that this work brings the attention of the scientific community to what could be called Phylogenetics-aware Software Engineering, with the expectation of more benefits for Software Engineering.

References

[1] Apache OpenNLP. 2019. Toolkit for the processing of natural language text. Retrieved from https://opennlp.apache.org/

[2] Efficient Java Matrix Library. 2019. Retrieved from http://ejml.org/

[3] Michael Affenzeller, Stephan M. Winkler, Stefan Wagner, and Andreas Beham. 2009. *Genetic Algorithms and Genetic Programming – Modern Concepts and Practical Applications*. CRC Press. Retrieved from http://www.crcpress.com/product/isbn/9781584886297

[4] Icek Ajzen. 1991. The theory of planned behavior. *Organizational Behavior and Human Decision Processes* 50, 2 (1991), 179–211.

[5] Nasser M. Albunian. 2017. Diversity in search-based unit test suite generation. In *Search Based Software Engineering*. Tim Menzies and Justyna Petke (Eds.), Springer International Publishing, Cham, 183–189.

[6] Ingo Althöfer. 2003. *Computer-aided Game Inventing*. Technical Report. Friedrich Schiller Universität, Jena, Germany.

[7] Apostolos Ampatzoglou and Ioannis Stamelos. 2010. Software engineering research for computer games: A systematic review. *Information and Software Technology* 52, 9 (2010), 888–901. https://doi.org/10.1016/j.infsof.2010.05.004

[8] Lorena Arcega, Jaime Font, Øystein Haugen, and Carlos Cetina. 2015. Leveraging models at run-time to retrieve information for feature location. In *Proceedings of the 10th International Workshop on Models@Run.time Co-Located with the 18th International Conference on Model Driven Engineering Languages and Systems (MoDELS '15)*, 51–60.

[9] Andrea Arcuri and Lionel Briand. 2014. A hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering. *Software Testing, Verification and Reliability* 24, 3 (2014), 219–250. https://doi.org/10.1002/stvr.1486

[10] Andrea Arcuri and Gordon Fraser. 2013. Parameter tuning or default values? An empirical investigation in search-based software engineering. *Empirical Software Engineering* 18, 3 (2013), 594–623. https://doi.org/10.1007/s10664-013-9249-9

[11] Manuel Ballarín, Ana Cristina Marcén, Vicente Pelechano, and Carlos Cetina. 2018. Measures to report the location problem of model fragment location. In *Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS '18)*, 189–199. https://doi.org/10.1145/3239372.3239397

[12] D. A. Baum and S. D. Smith. 2012. *Tree Thinking: An Introduction to Phylogenetic Biology*. Macmillan Learning. Retrieved from https://books.google.es/books?id=zW_ApwAACAAJ

[13] Daniel Blasco, Jaime Font, Francisca Pérez, and Carlos Cetina. 2023. Procedural content improvement of game bosses with an evolutionary algorithm. *Multimedia Tools and Applications* 82, 7 (2023), 10277–10309. https://doi.org/10.1007/s11042-022-13674-6

[14] Daniel Blasco, Jaime Font, Mar Zamorano, and Carlos Cetina. 2021. An evolutionary approach for generating software models: The case of kromaia in game software engineering. *Journal of Systems and Software* 171 (2021), 110804. https://doi.org/10.1016/j.jss.2020.110804

[15] Alix Boc, Alpha Boubacar Diallo, and Vladimir Makarenkov. 2012. T-REX: A web server for inferring, validating and visualizing phylogenetic trees and networks. *Nucleic Acids Research* 40, Web Server issue (2012), W573–W579. https://doi.org/10.1093/nar/gks485. Retrieved from https://academic.oup.com/nar/article-pdf/40/W1/W573/4924823/gks485.pdf

[16] A. Bookstein and S. T. Klein. 1991. Compression of correlated bit-vectors. *Information Systems* 16, 4 (1991), 387–400. https://doi.org/10.1016/0306-4379(91)90030-D

[17] Cameron Browne and Frédéric Maire. 2010. Evolutionary game design. *IEEE Transactions on Computational Intelligence and AI in Games* 2, 1 (2010), 1–16. https://doi.org/10.1109/TCIAIG.2010.2041928

[18] Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *ACM Comput. Surv.* 44, 1 (2012), 1–50.

[19] Rodrigo Casamayor, Lorena Arcega, Francisca Pérez, and Carlos Cetina. 2022. Bug localization in game software engineering: Evolving simulations to locate bugs in software models of video games. In *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems (MODELS '22)*. Eugene Syriani, Houari A. Sahraoui, Nelly Bencomo, and Manuel Wimmer (Eds.), ACM, 356–366. https://doi.org/10.1145/3550355.3552440.

[20] Luigi L. Cavalli-Sforza and Anthony W. F. Edwards. 1967. Phylogenetic analysis. Models and estimation procedures. *American Journal of Human Genetics* 19, 3 (1967), 233.

[21] Carlos Cetina, Jaime Font, Lorena Arcega, and Francisca Pérez. 2017. Improving feature location in long-living model-based product families designed with sustainability goals. *Journal of Systems and Software* 134 (2017), 261–278.
[22] Jorge Chueca, Daniel Blasco, Carlos Cetina, and Jaime Font. 2024. Leveraging phylogenetics in software product families: The case of latent content generation in video games. In *Proceedings of the 28st International Systems and Software Product Line Conference (SPLC '24)*, 113–124.
[23] Jacob Cohen. 2013. *Statistical Power Analysis for the Behavioral Sciences*. Routledge.
[24] William Jay Conover. 1999. *Practical Nonparametric Statistics* (3. ed ed.). Wiley, New York, NY, USA
[25] Lawrence Davis. 1991. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York.
[26] Bogdan Dit, Meghan Revelle, Malcom Gethers, and Denys Poshyvanyk. 2013. Feature location in source code: A taxonomy and survey. *Journal of Software: Evolution and Process* 25, 1 (2013), 53–95. https://doi.org/10.1002/smr.567
[27] Daniel Dyer. 2016. The watchmaker framework for evolutionary computation (evolutionary/genetic algorithms for Java). Retrieved from http://watchmaker.uncommons.org/
[28] Jorge Echeverría, Jaime Font, Francisca Pérez, and Carlos Cetina. 2021. Comparison of search strategies for feature location in software models. *Journal of Systems and Software* 181, C (2021), 111037. https://doi.org/10.1016/j.jss.2021.111037
[29] Jorge Echeverría, Francisca Pérez, José Ignacio Panach, and Carlos Cetina. 2021. An empirical study of performance using clone & own and software product lines in an industrial context. *Information and Software Technology* 130 (2021), 106444.
[30] Davide Falessi, Giovanni Cantone, and Gerardo Canfora. 2011. Empirical principles and an industrial case study in retrieving equivalent requirements via natural language processing techniques. *IEEE Transactions on Software Engineering* 39, 1 (2011), 18–44.
[31] Jaime Font, Lorena Arcega, Øystein Haugen, and Carlos Cetina. 2016. Feature location in models through a genetic algorithm driven by information retrieval techniques. In *Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MODELS '16)*. ACM, 272–282. https://doi.org/10.1145/2976767.2976789
[32] J. Font, L. Arcega, Ø. Haugen, and C. Cetina. 2017. Achieving feature location in families of models through the use of search-based software engineering. *IEEE Transactions on Evolutionary Computation* 22, 3 (2017), 363–377. https://doi.org/10.1109/TEVC.2017.2751100
[33] Eclipse Foundation. 2015. Eclipse, Version Mars.1 Release (4.5.1). Retrieved from https://www.eclipse.org/
[34] Epic Games. 1998. Unreal Engine, Version 5.2. Retrieved from http://www.unrealengine.com/
[35] Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. 2010. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences* 180, 10 (2010), 2044–2064. https://doi.org/10.1016/j.ins.2009.12.010
[36] Daniele Gravina, Ahmed Khalifa, Antonios Liapis, Julian Togelius, and Georgios N. Yannakakis. 2019. Procedural content generation through quality diversity. In *Proceedings of the 2019 IEEE Conference on Games (CoG)*. IEEE. https://doi.org/10.1109/cig.2019.8848053
[37] Mark Harman, Yue Jia, and Yuanyuan Zhang. 2015. Achievements, open problems and challenges for search based software testing. In *Proceedings of the 2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST '15)*, 1–12. https://doi.org/10.1109/ICST.2015.7102580
[38] Mark Harman and Bryan F. Jones. 2001. Search-based software engineering. *Information and Software Technology* 43, 14 (2001), 833–839. https://doi.org/10.1016/S0950-5849(01)00189-6
[39] Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, 50–57.
[40] Kent Holsinger and Bruce Weir. 2009. Genetics in geographically structured populations: Defining, estimating and interpreting FST. *Nature Reviews. Genetics* 10, 9 (2009), 639–650. https://doi.org/10.1038/nrg2611
[41] Anton Kotelyanskii, and Gregory M. Kapfhammer. 2014. Parameter tuning for search-based test-data generation revisited: Support for previous results. In *Proceedings of the 2014 14th International Conference on Quality Software*, 79–84. https://doi.org/10.1109/QSIC.2014.43
[42] Jacob Krüger, Thorsten Berger, and Thomas Leich. 2019. Features and how to find them: A survey of manual feature location. In *Software Engineering for Variability Intensive Systems – Foundations and Applications*. Taylor & Francis Group, 153–172.
[43] Jacob Krüger, Wanzi Gu, Hui Shen, Mukelabai Mukelabai, Regina Hebig, and Thorsten Berger. 2018. Towards a better understanding of software features and their characteristics: A case study of Marlin. In *Proceedings of the 12th International Workshop on Variability Modelling of Software-Intensive Systems (VAMOS '18)*. ACM, New York, NY, 105–112.
[44] Thomas K. Landauer, Peter W. Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse Processes* 25, 2–3 (1998), 259–284.

[45] Dapeng Liu, Andrian Marcus, Denys Poshyvanyk, and Vaclav Rajlich. 2007. Feature location via information retrieval based filtering of a single scenario execution trace. In *Proceedings of the 22nd IEEE/ACM International Conference on Automated Software Engineering (ASE '07)*. ACM, 234–243.

[46] Roberto E. Lopez-Herrejon, Javier Ferrer, Francisco Chicano, Lukas Linsbauer, Alexander Egyed, and Enrique Alba. 2014. A Hitchhiker's guide to search-based software engineering for software product lines. arXiv:1406.2823. Retrieved from http://arxiv.org/abs/1406.2823

[47] Ana C. Marcén, Jaime Font, Óscar Pastor, and Carlos Cetina. 2017. Towards feature location in models through a learning to rank approach. In *Proceedings of the 21st International Systems and Software Product Line Conference – Volume B (SPLC '17)*, 57–64. https://doi.org/10.1145/3109729.3109734

[48] Andrian Marcus, Andrey Sergeyev, Vaclav Rajlich, and Jonathan I. Maletic. 2004. An information retrieval approach to concept location in source code. In *Proceedings of the 11th Working Conference on Reverse Engineering (WCRE '04)*. IEEE Computer Society, Washington, DC, USA, 214–223. Retrieved from http://dl.acm.org/citation.cfm?id=1038267.1039053

[49] Phil McMinn, Mark Harman, and David Binkley. 2006. The species per path approach to search-based test data generation. In *Proceedings of the 2006 International Symposium on Software Testing and Analysis*, ISSTA 2006, 13–24. DOI: https://doi.org/10.1145/1146238.1146241

[50] Alexandre Santos Melotti and Carlos Henrique Valerio de Moraes. 2019. Evolving roguelike dungeons with deluged novelty search local competition. *IEEE Transactions on Games* 11, 2 (2019), 173–182. https://doi.org/10.1109/TG.2018.2859424

[51] Hector D. Menendez, Michele Boreale, Daniele Gorla, and David Clark. 2022. Output sampling for output diversity in automatic unit test generation. *IEEE Transactions on Software Engineering* 48, 1 (2022), 295–308. https://doi.org/10.1109/TSE.2020.2987377

[52] Prabhaker Mishra, Chandra M. Pandey, Uttam Singh, Anshul Gupta, Chinmoy Sahu, and Amit Keshri. 2019. Descriptive statistics and normality tests for statistical data. *Annals of Cardiac Anaesthesia* 22, 1 (2019), 67–72.

[53] David W. Mount. 2004. *Bioinformatics – Sequence and Genome Analysis* (2. ed.). Cold Spring Harbor Laboratory Press.

[54] Masatoshi Nei. 1972. Genetic distance between populations. *The American Naturalist* 106, 949 (1972), 283–292.

[55] Hoan Anh Nguyen, Tung Thanh Nguyen, Nam H. Pham, Jafar Al-Kofahi, and Tien N. Nguyen. 2012. Clone management for evolving software. *IEEE Transactions on Software Engineering* 38, 5 (2012), 1008–1026. https://doi.org/doi.ieeecomputersociety.org/10.1109/TSE.2011.90

[56] Francisca Pérez, Jaime Font, Lorena Arcega, and Carlos Cetina. 2019. Collaborative feature location in models through automatic query expansion. *Automated Software Engineering* 26, 1 (2019), 161–202. https://doi.org/10.1007/s10515-019-00251-9

[57] Francisca Pérez, Jaime Font, Lorena Arcega, and Carlos Cetina. 2022. Empowering the human as the fitness function in search-based model-driven engineering. *IEEE Transactions on Software Engineering* 48, 11 (2022), 4553–4568. https://doi.org/10.1109/TSE.2021.3121253

[58] Francisca Pérez, Raúl Lapeña, Ana C. Marcén, and Carlos Cetina. 2022. How the quality of maintenance tasks is affected by criteria for selecting engineers for collaboration. *ACM Transactions on Software Engineering and Methodology* 32, 3 (2022), 1–22. https://doi.org/10.1145/3561384

[59] Francisca Pérez, Raúl Lapeña, Jaime Font, and Carlos Cetina. 2018. Fragment retrieval on models for model maintenance: Applying a multi-objective perspective to an industrial case study. *Information and Software Technology* 103 (2018), 188–201. https://doi.org/10.1016/j.infsof.2018.06.017

[60] Francisca Pérez, Tewfik Ziadi, and Carlos Cetina. 2022. Utilizing automatic query reformulations as genetic operations to improve feature location in software models. *IEEE Transactions on Software Engineering* 48, 2 (2022), 713–731. https://doi.org/10.1109/TSE.2020.3000520

[61] Denys Poshyvanyk, Yann-Gael Gueheneuc, Andrian Marcus, Giuliano Antoniol, and Vaclav Rajlich. 2007. Feature location using probabilistic ranking of methods based on execution scenarios and information retrieval. *IEEE Transactions on Software Engineering* 33, 6 (2007), 420–432. https://doi.org/10.1109/TSE.2007.1016

[62] Mike Preuss, Antonios Liapis, and Julian Togelius. 2014. Searching for good and diverse game levels. In *Proceedings of the 2014 IEEE Conference on Computational Intelligence and Games*, 1–8. https://doi.org/10.1109/CIG.2014.6932908

[63] Dragan Radovanović, Viktor Stoičkov, Aleksandar Ignjatović, Aaron T. Scanlan, Vladimir Jakovljević, and Emilija Stojanović. 2020. A comparison of cardiac structure and function between female powerlifters, fitness-oriented athletes, and sedentary controls. *Echocardiography (Mount Kisco, N.Y.)* 37, 10 (2020), 1566–1573.

[64] C. K. Riemenschneider, B. C. Hardgrave, and F. D. Davis. 2002. Explaining software developer acceptance of methodologies: A comparison of five theoretical models. *IEEE Transactions on Software Engineering* 28, 12 (2002), 1135–1145. https://doi.org/10.1109/TSE.2002.1158287

[65] JoséMiguel Rojas, Gordon Fraser, and Andrea Arcuri. 2016. Seeding strategies in search-based unit test generation. *Software Testing, Verification and Reliability* 26, 5 (2016), 366–401.

[66] Per Runeson and Martin Höst. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering* 14, 2 (2009), 131–164. https://doi.org/10.1007/s10664-008-9102-8

[67] Stuart Russell and Peter Norvig. 2010. *Artificial Intelligence: A Modern Approach* (3rd. ed.). Prentice Hall. Retrieved from https://www.bibsonomy.org/bibtex/20533b732950d1c5ab4ac12d4f32fe637/mialhoma
[68] N. Saitou and M. Nei. 1987. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution* 4, 4 (1987), 406–425. Retrieved from https://doi.org/10.1093/oxfordjournals.molbev.a040454
[69] Gerard Salton and Michael J. McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.
[70] A. S. Sayyad, J. Ingram, T. Menzies, and H. Ammar. 2013. Scalable product line configuration: A straw to break the camel's back. In *Proceedings of the 2013 IEEE/ACM 28th International Conference on Automated Software Engineering (ASE)*, 465–474. https://doi.org/10.1109/ASE.2013.6693104
[71] Randall T. Schuh and Andrew V. Z. Brower. 2009. *Biological Systematics: Principles and Applications*, (1 ed.), Cornell University Press. Retrieved from http://www.jstor.org/stable/10.7591/j.ctt7zbx6
[72] Russell Schwartz and Alejandro Schaffer. 2017. The evolution of tumour phylogenetics: Principles and practice. *Nature Reviews. Genetics* 18, 4 (2017), 213–229. https://doi.org/10.1038/nrg.2016.170
[73] David Steinberg, Frank Budinsky, Marcelo Paternostro, and Ed Merks. 2009. *EMF: Eclipse Modeling Framework 2.0* (2nd ed.). Addison-Wesley Professional.
[74] SVIT Research Group. 2023. Game Software Engineering Repository. Retrieved from http://www.gamesoftwareengineering.com
[75] Unity Technologies. 2005. Unity, Version 2022.2. Retrieved from https://unity.com/
[76] Maciej Tomczak and Ewa Tomczak. 2014. The need to report effect size estimates revisited. An overview of some recommended measures of effect size. *Trends in Sport Sciences* 1, 21, 19–25.
[77] Tassio Vale and Eduardo Santana Almeida. 2019. Experimenting with information retrieval methods in the recovery of feature-code SPL traces. *Empirical Software Engineering* 24, 3 (2019), 1328–1368. https://doi.org/10.1007/s10664-018-9652-3
[78] András Vargha, Harold D. Delaney, and Andras Vargha. 2000. A critique and improvement of the CL common language effect size statistics of McGraw and Wong. *Journal of Educational and Behavioral Statistics* 25, 2 (2000), 101–132. https://doi.org/10.3102/10769986025002101 Retrieved from http://jeb.sagepub.com/content/25/2/101.full.pdf+html
[79] Jinshui Wang, Xin Peng, Zhenchang Xing, and Wenyun Zhao. 2011. An exploratory study of feature location process: Distinct phases, recurring patterns, and elementary actions. In *Proceedings of the 27th Conference on Software Maintenance*. IEEE, 213–222. https://doi.org/10.1109/ICSM.2011.6080788
[80] Grady Weyenberg and Ruriko Yoshida. 2015. Chapter 12 Reconstructing the phylogeny: Computational methods. In *Algebraic and Discrete Mathematical Methods for Modern Biology*. Raina S. Robeva (Ed.), Academic Press, Boston, 293–319. https://doi.org/10.1016/B978-0-12-801213-0.00012-5.
[81] Stefan Winkler and Jens Pilgrim. 2010. A survey of traceability in requirements engineering and model-driven development. *Software & Systems Modeling* 9, 4 (2010), 529–565.
[82] Shin Yoo and Mark Harman. 2012. Test data regeneration: Generating new test data from existing test data. *Software Testing, Verification and Reliability* 22, 3 (2012), 171–201.
[83] Meng Zhu and Alf Inge Wang. 2019. Model-driven game development: A literature review. *ACM Computing Surveys* 52, 6 (2019), 1–32. https://doi.org/10.1145/3365000

Appendix

A TPB Questionnaire

This appendix shows the questions from the TPB questionnaire used for the PCG and FL case studies. Questions 1 to 5 assess Attitude, questions 6 to 8 assess Subjective norm, and questions 9 to 14 assess Perceived behavioral control. Each question is a Likert scale with values ranging from 1 to 7.

TPB Questionnaire for the PCG Case Study.

(1) The use of this type of bosses for the development of my video games would be:

(bad) 1 2 3 4 5 6 7 (good)

(2) The use of this type of bosses for the development of my video games would be:

(negligent) 1 2 3 4 5 6 7 (desirable)

(3) The use of this type of bosses for the development of my video games would be:

(unfavorable) 1 2 3 4 5 6 7 (favorable)

(4) The use of this type of bosses for the development of my video games would be:

(harmful) 1 2 3 4 5 6 7 (beneficial)

(5) The use of this type of bosses for the development of my video games would be:

(negative) 1 2 3 4 5 6 7 (positive)

(6) The video game developers I know would use these bosses in their video games

1 2 3 4 5 6 7

(7) The video game developers who are important to me would like me to use these bosses in my games

1 2 3 4 5 6 7

(8) The video game developers I value would prefer that I use these bosses in my games

1 2 3 4 5 6 7

(9) I would be able to use these bosses in my game successfully

1 2 3 4 5 6 7

(10) I have the resources, skill, and knowledge to use these bosses in my game

2 3 4 5 6 7

(11) Given the possibility to use these bosses in my game, it would be the easiest for me to use these bosses in my game 1

2 3 4 5 6 7

(12) If I have to develop a game with this type of bosses, I would not hesitate to use them

2 3 4 5 6 7

(13) I am confident in using this type of bosses to develop a game

2 3 4 5 6 7

(14) To develop this type of video games, I would use these bosses before any other

2 3 4 5 6 7

TPB Questionnaire for the FL Case Study.

(1) The use of this type of features automatically located for software maintenance would be:

(bad) 1 2 3 4 5 6 7 (good)

(2) The use of this type of features automatically located for software maintenance would be:

(negligent) 1 2 3 4 5 6 7 (desirable)

(3) The use of this type of features automatically located for software maintenance would be:

(unfavorable) 1 2 3 4 5 6 7 (favorable)

(4) The use of this type of features automatically located for software maintenance would be:

(harmful) 1 2 3 4 5 6 7 (beneficial)

(5) The use of this type of features automatically located for software maintenance would be:

(negative) 1 2 3 4 5 6 7 (positive)

(6) The engineers I know would use these model fragments in their corresponding maintenance tasks

2 3 4 5 6 7

(7) The engineers who are important to me would like me to use these model fragments during maintenance tasks

2 3 4 5 6 7

(8) The engineers who are important to me would like me to use these model fragments

2 3 4 5 6 7

(9) I would be able to use these model fragments successfully

2 3 4 5 6 7

(10) I have the resources, skill, and knowledge to use these model fragments during maintenance tasks

2 3 4 5 6 7

(11) Given the possibility to use these model fragments, it would be the easiest for me to use these model fragments in my maintenance tasks

2 3 4 5 6 7

(12) If I have to locate features that correspond to these model fragments, I would not hesitate to use them

2 3 4 5 6 7

(13) I am confident that I can use these model fragments to maintain their corresponding features in a product

2 3 4 5 6 7

(14) To locate features, I would use these model fragments before any other

2 3 4 5 6 7

Received 5 September 2023; revised 12 November 2024; accepted 16 January 2025