STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXII**, Number 1, 2017
DOI: 10.24193/subbi.2017.1.02

THIRD CASE STUDY FOR THE DYNAMIC MULTILEVEL COMPONENT SELECTION

ANDREEA VESCAN

Abstract. The architecture of a system changes after the deployment phase due to new requirements from the stakeholders. The software architect must make decisions about the selection of the right software components out of a range of choices to satisfy a set of requirements. This paper deals with the component selection problem with a multilevel system view in a dynamic environment.

To validate our approach we have used the case study method. Three different case studies were performed but only one is presented in the current paper. The research design was conducted using a research question, propositions and for interpreting the study's findings we have used the Wilcoxon signed ranks statistical test. The tests performed show the potential of evolutionary algorithms for the dynamic multilevel component selection problem.

1. Introduction

The problems of identification and selection the right software components out of a range of choices to satisfy a set of requirements have received considerable attention in the field of component-based software engineering during the last two decades [9, 10].

Identification of a software architecture for a given system may be achieved in two steps: (1) Component Identification and (2) Component Selection. Component Identification has the scope to partition functionalities of a given system into non-intersecting logical components to provide the starting points for designing the architecture. The aim of Component Selection methods is to find suitable components from a repository to satisfy a set of requirements under various constraints/criteria (i.e. cost, number of used components, etc.).

Received by the editors: January 25, 2017.

2010 *Mathematics Subject Classification.* 68N01, 68T20.

1998 *CR Categories and Descriptors.* D.2.2 [**Design Tools and Techniques**]: *Object-oriented design methods, Software libraries.*; I.2.8 [**Problem Solving, Control Methods, and Search**]: *Heuristic methods.*

Key words and phrases. Case study, Research design, Component selection, Dynamic, Multilevel, Multiobjective optimization.

This paper has focused on the component selection process, the goal being to provide the suitable existing components matching software requirements.

The **contribution** of this paper is **the use of the case study method and the research design** from the book of Yin [5] to validate our research proposal for the **Dynamic Multilevel Component Selection Problem** [2]. A research question and propositions are used to conduct research design. For interpreting the study's findings we used the Wilcoxon signed ranks statistical test. We have conducted three different experiments. The first one was reported in [1], and the second one in [2]. The current paper **reports the third experiment**. For each case study we have specified the component selection problem. After that, the experimental studies were followed for each considered case study: the two perspectives, changing requirements and changing component repository. Following the replication approach to multiple-case studies [5], each individual case study was finalized by an individual case report that will be next considered to be part of a summary report, i.e. a cross-case conclusion. Thus, in our case the results obtained are reported and conclusions about the potential of evolutionary algorithms for the dynamic multiobjective multilevel component selection problem are drawn.

The paper is organized as follows: Section 2 contains configuration and reconfiguration description problems, the description of the optimization process, and the proposed evolutionary-based algorithm approach. Section 3 presents the reasons for using case study method and the research design, the criteria used to interpret the findings of the results. The evaluation is presented in Section 4. In Section 5 we apply the approach to one example to validate the proposed approach. Some experiments are performed considering two dynamics: requirements changes over time and component repository varies over time. Section 6 introduces the current state of art regarding the component selection problem and analysis the differences compared with our present approach. We conclude our paper in Section 7.

2. Dynamic Multilevel Component Selection Problem

2.1. Component Systems, Configurations and Reconfigurations.

A component [23] is an independent software package that provides functionality via defined interfaces. The interface may be an export interface through which a component provides functionality to other components or an import interface through which a component gains services from other components.

A configuration [24] of a component system is described as the structural relationship between components, indicated by the layout of components and connectors. Reconfiguration means modifying the structure of a component system in terms of additions, deletion, and replacement of components and/or connectors. While the reconfiguration transforms the structural view of a component system, it changes the system's functionality and service specifications. From another aspect, a reconfiguration may consist of several individual updates or changes to components and/or connectors.

There are two type of components: *simple component* - is specified by the inports (the set of input variables/parameters), outports (the set of output variables/parameters) and a function (the computation function of the component) and *compound component* - is a group of connected components in which the output of a component is used as input by another component from the group. For details about the component model please refer to [25].

2.2. **Dynamic Multilevel Component Selection Problem Formulation.** An informal specification of the *configuration problem* is described in the following. It is needed to construct a final system specified by input data and output data. We can see the final system as a compound component and thus the input data becomes the required interfaces of the component and the output data becomes the provided interfaces, and in this context we have the required interfaces as provided and we need to provide the internal structure of the final compound component by offering the provided interfaces.

A formal definition of the *configuration problem* [25] (seen as a compound component) is as follows. Consider SR the set of final system requirements (the provided functionalities of the final compound component) as $SR = \{r_1, r_2, ..., r_n\}$ and SC the set of components (the repository) available for selection as $SC = \{c_1, c_2, ..., c_m\}$. Each component c_i can satisfy a subset of the requirements from SR (the provided functionalities) denoted $SP_{c_i} = \{p_{i_1}, p_{i_2}, ..., p_{i_k}\}$ and has a set of requirements denoted $SR_{c_i} = \{r_{i_1}, r_{i_2}, ..., r_{i_h}\}$. The goal is to find a set of components Sol in such a way that every requirement r_j ($j = \overline{1,n}$) from the set SR can be assigned a component c_i from Sol where r_j is in SP_{c_i} ($i = \overline{1,m}$), while minimizing the number of used components and the total cost of assembly. All the requirements of the selected components must be satisfied by the components in the solution. If a selected component is a compound component, the internal structure is also provided. All the levels of the system are constructed.

The *reconfiguration problem* [24] is defined similarly to the *configuration problem* but considering the dynamical changes of either requirements or component. Regarding the *reconfiguration problem* [15], the dynamics of the component selection problem can be viewed in two ways: the system requirements or the repository containing the components varies over time.

2.3. **Dynamic Multilevel Component Selection Optimisation Process.** Our approach starts by considering a set of components (the repository) available for selection and the specification of a final system (input and output). The optimisation process begins with the *Dynamic Multilevel Component Selection Problem Formulation* (see Figure 1 for details). The result of this step is the transformation of the final system specification as the set of required interfaces (and the set of provided interfaces).

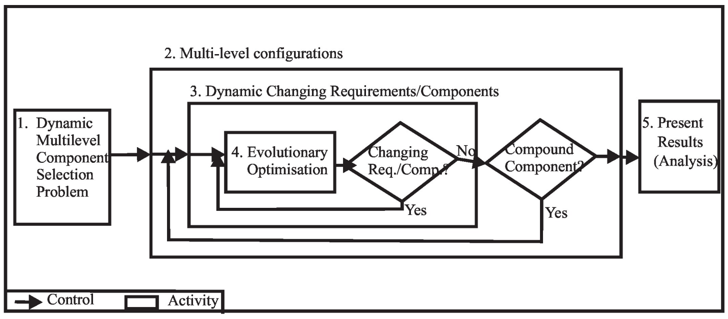
2. Multi-level configurations
3. Dynamic Changing Requirements/Components
1. Dynamic Multilevel Component Selection Problem
4. Evolutionary Optimisation
Changing Req./Comp.?
No
Yes
Compound Component?
Yes
5. Present Results (Analysis)
Control
Activity

FIGURE 1. Dynamic Multilevel Component Selection Optimisation Process.

In the second step, the construction of the multilevel configurations is done by applying the evolutionary optimisation algorithm (fourth step, in Figure 1) for each time steps (from the Dynamic Changing Requirements or Dynamic Changing Components step). The evolutionary optimisation algorithm is applied for each time steps (i.e. if there are still changing requirements or components) and for each compound component from each level. The solution with best fitness value is selected at each level. The fifth step presents the results. The problem is formulated as a multiple objective optimization problem having 5 objectives: the number of used components, the number of new requirements, the number of provided interfaces, the number of the initial requirements that are not in solution, and the cost of a component (group of components). All objectives are to be minimized.

Remark. Detailed information about the optimisation process can be found in paper [1]. Since the current paper focuses on the research design and case study method we give only the reference for the proposed approach [1] and second case study paper [2] such that the reader can find the information.

3. Case study method and research design

The book of Yin [5] provided us with a strategy of identifying the method for our research project, showing when to choose the case study method and how to do research design. Defining the *Research Questions* is the most important step to consider in a research study. In general, case studies are the preferred strategy when "how" or "why" questions are being posed.

Our Research Question: How and Why do Search-based Algorithms (in our case a Genetic Algorithm and a Random Search Algorithm) provide different results for the Dynamic Multilevel Component Selection Problem?

Another component of the research design are the *Propositions* that direct attention to something that should be examined within the scope of study. These propositions begin to tell you where to look for relevant evidence.

Our Proposition: The Search-based Algorithms (in our case a Genetic Algorithm and a Random Search Algorithm) provide different results for the Dynamic Multilevel Component Selection Problem because in the case of Genetic Algorithm the fitness of a solution is reevaluated.

Criteria for interpreting a study's findings represents the third component of the research design of a case study according to [5]. Statistical analysis offer some explicit criteria for such interpretation.

Our Criteria for interpreting the study is based on the Wilcoxon signed ranks statistical test that aims to detect significant differences between two sample means, that is, the behavior of the two algorithms. For more information regarding the Wilcoxon statistical test see Section 4.

After covering these components of research designs, the construction of a theory related to our topic of study will follow. *Our Theory development*: The case study will show why the Genetic Algorithm performs better than the Random Search Algorithm.

An issue related to case studies is referring to the generalization from a case study to theory. According to [5] statistical generalization is the common way when doing surveys, but in doing case studies the analytic generalization should be used. Multiple cases resemble multiple experiments and under these circumstances the mode of generalization is analytic. If two or more cases are shown to support the same theory, replication [5] may be claimed. The replication logic is analogous to that used in multiple experiments. Some of the replications might attempt to duplicate the exact conditions of the original experiment. Other replications might alter one or two experimental conditions considered unimportant to the original findings, to see whether the findings could still be duplicated. Only with such replications would the original finding be considered robust.

Our Replication strategy: The time steps for the dynamic changing requirements (Level 1) and for the dynamic changing components (Level 1, 2, and 3) were modified for Case study 2 [1] and for Case Study 3 [2]. Additional information relating to the replication strategy is presented in paper [2].

Thus, we have selected the case study method and conducted three case studies, the first one is a real case study for constructing a *Reservation System* (reported in [1]) and the last two of them are constructed using artificial data (the second experiment is reported in [2] and the current paper reports the third experiment). Each individual case study was finalized by an individual case report that will be next considered to be part of a summary report, i.e. a cross-case conclusion. Thus, in our case the results obtained are reported and conclusions about the potential of evolutionary algorithms for the dynamic multiobjective multilevel component selection problem are drawn.

4. Interpreting the study

When comparing [8] two algorithms, the best fitness values obtained by the searches concerned are an obvious indicator to how well the optimisation process performed. Inferential statistics may be applied to discern whether one set of experiments are significantly different in some aspect from another. Usually we wish to be in a position to make a claim that we have evidence that suggests that Algorithm A (Genetic Algorithm) is better than Algorithm B (Random Search). The Wilcoxon signed ranks test [6] is used for answering the following question: do two samples represent two different populations? It is a nonparametric procedure employed in hypothesis testing situations, involving a design with two samples. It is a pairwise test that aims to detect significant differences between two sample medians, that is, the behavior of two algorithms. The best fitness value (from the entire population) was used for comparing the two algorithms.

The Wilcoxon signed ranks test has two hypothesis:

(1) Null hypothesis H_0: The median difference is zero versus.
(2) Research hypothesis H_1: The median difference is not zero, $\alpha = 0.05$.

Steps of the Wilcoxon signed ranks test: Compute W_- and W_+, Check if $W_- + W_+ =$ n(n+1)/2, Select the test statistic (for the two tailed test the test statistic is the smaller of W_- and W_+), We must determine whether the observed test statistic W_t supports the H_0 or H_1, i.e. we determine a critical value of W_c such that if the observed value of W_t is less or equal to critical value W_c, we reject H_0 in favor to H_1.

Due to stochastic nature of optimisation algorithms, searches must be repeated several times in order to mitigate against the effect of random variation. How many runs do we need when we analyze and compare algorithms?

In many fields of science (i.e. medicine and behaviour science) a common rule of thumb [7] is to use at least $N = 30$ observations. We have also used in our evaluation 30 executions for each algorithm.

Remark. One of the main reasons why we have used Wilcoxon signed rank test is that the area of the study (i.e. comparing the best solutions obtained by the two algorithms) is better represented by the median, thus we compared the best chromosomes obtained by the two methods and not the mean of all the chromosomes for each execution. Another reason of using non-parametric test could be a very small sample size, but if the median better represents the centre of the distribution, the indication is to consider the nonparametric test even when we have a larger sample.

5. Experimental results

In this section, the proposed approach is evaluated and the results are reported. According to the book of Yin [5] and as stated in Section 3 we have conducted three different experiments. The first one was reported in [1], and the second one in [2]. The current paper reports the third experiment.

5.1. **Third case study. Component Selection Problem formulation.**

The third case study uses a set of 90 available components. The final system has one input data and two output data, the goal is to find a subset of the given components such that all the requirements are satisfied considering the optimisation criteria specified above. The set of requirements $SR = \{r_1,\ r_2\}$ (view as provided interfaces $\{p_1, p_2\}$) and the set of components $SC = \{c_0, c_1, c_2, c_3, c_4, c_5, c_6, ..., c_{90}\}$ are given. The final system has as input data (transformed in required interfaces) the set $\{r_3\}$.

Remark. Due to lack of space the component repository is not described in this paper but may be found at [16].

5.1.1. *Experimental studies - Case 1: Dynamically changing requirements.*

As in the case of the first case study, we consider two types of dynamics and, consequently two experiments corresponding to each of them: the requirements of the problem change over time, and the components available at a certain time step change.

The algorithm was run 30 times and the number of nondominated solutions and the number of distinct nondominated solutions were recorded for all situations. Also, the cost and the number of distinct used components in a solution were logged. Also, the best, worse and average fitness values were recorded for all situations.

TABLE 1. Wilcoxon statistical test for Case Study 1 - *changing requirements* experiment.

L-T	W_-	W_+	W_{test}	N	W_{critic}	H_0	H_1	p-value
L1-T0	-46	254	46	24	81	$\times$	✓	0.00298
L1-T1	-1	464	1	30	137	$\times$	✓	0.00000
L1-T2	-46	389	46	29	126	$\times$	✓	0.00020
L2-T0	-172	38	38	20	52	$\times$	✓	0.01242
L3-T0	-269.5	55.5	55.5	25	89	$\times$	✓	0.00560

Three different time steps are built using artificially generated data and the dynamics at each of these steps are: T0=The initial requirements, T1=Add one new requirement, T2=Add one new requirement.

Performed experiments.

The role of the performed test was to see if the number of iterations and the population size play a role in finding the Pareto solutions. The conclusions about the findings of this type of experiments are given in Section 5.1.3.

Multilevel configurations. The compound components from level 1 are constructed by applying the same algorithm but with different requirements and input data. For the Second Level of the system the set of required interfaces is $\{r_{11},\ r_{15}\}$ and the set of provided interfaces is $\{p_2\}$. For the Third Level of the system the set of required interfaces is $\{r_{15}\}$ and the set of provided interfaces is $\{p_{17},\ p_{18}\}$. The conclusions about the findings of this type of experiments are given in Section 5.1.3.

Remark. We have not presented the charts regarding the influence of population size or iteration number in finding the Pareto solutions, because we have concentrated our findings in comparing the algorithms using the Wilcoxon statistical test.

Wilcoxon statistical test.

In Section 4 we have described in details the Wilcoxon statistical test that we have used to compare our Genetic Algorithm with the Random Search Algorithm. In Table 1 the test results for the Case Study 1 - Dynamically Changing Requirements are shown.

The Wilcoxon statistical test (see Table 1) shows that we have statistically significant evidence at $\alpha = 0.05$ to show that the median is positive, i.e. the H_0 Null-Hypothesis is rejected in favor of H_1 for all levels and for all time steps.

5.1.2. *Experimental studies - Case 2: Dynamically changing components.* As in the first case study, the repository containing components changes over time.

This modification of the available components may be seen as an update of the COTS market, new components being available or other being withdrawn from the market.

Four different time steps are built using artificially generated data and the dynamics at each of these steps are: T0= The initial components, T1= Add two new components, T2= Remove one component and add one new component, T3= Remove one component.

Performed experiments.

The aim of the performed tests is the same as in the first case study: to see if the number of iterations and the population size play a role in finding the Pareto solutions. The conclusions about the findings of this type of experiments are given in Section 5.1.3.

Multilevel configurations. The compound components are next constructed by applying the same algorithm but with different requirements and input data. For the second level of the system the set of required interfaces is $\{r_{11},\ r_{15}\}$ and the set of provided interfaces is $\{p_2\}$. For the second level we have two time steps: T1= No modifications of the component repository, T2= Adding two new components.

For the third level of the system the set of required interfaces is $\{r_{15}\}$ and the set of provided interfaces is $\{p_{17},\ p_{18}\}$. The conclusions about the findings of this type of experiments are given in Section 5.1.3. For the third level we have three time steps: T1= No modifications of the component repository , T2= Adding two new components and removing two old components, T3=Removing three old components and adding one new component.

Remark. We have not presented the charts regarding the influence of population size or iteration number in finding the Pareto solutions, because we have concentrated our findings in comparing the algorithm using the Wilcoxon statistical test.

Wilcoxon statistical test.

In Section 4 we have described in details the Wilcoxon statistical test that we have use to compare our Genetic Algorithm with the Random Search Algorithm. In Table 2 we have the test results for the Case Study 2 - Dynamic Changing Components.

The Wilcoxon statistical test (see Table 2) shows that we have statistically significant evidence at $\alpha = 0.05$ to show that the median is positive, i.e. the H_0 Null-Hypothesis is rejected in favor of H_1 for all levels and for all time steps.

5.1.3. *Case Report 3.* The role of the conducted experiments had two directions: the first one was to see if the number of iterations and the population size play a role in finding the Pareto solutions. The second direction was

Table 2. Wilcoxon statistical test for Case Study 2 - *changing components* experiment.

L-T	W_-	W_+	W_{test}	N	W_{critic}	H_0	H_1	p-value
L1-T0	-110	325	110	29	126	$\times$	✓	0.02034
L1-T1	-53	298	53	26	98	$\times$	✓	0.00188
L1-T2	-65	286	65	26	98	$\times$	✓	0.01468
L1-T3	-56	350	56	28	116	$\times$	✓	0.00800
L2-T0	-152.5	39	39	19	46	$\times$	✓	0.02088
L2-T1	-197	56	56	22	65	$\times$	✓	0.02202
L3-T0	0	465	0	30	137	$\times$	✓	0.00000
L3-T1	0	78	0	12	13	$\times$	✓	0.00222
L3-T2	0	120	120	30	137	$\times$	✓	0.00064

to find out how and why do Search-based Algorithms (in our case a Genetic Algorithm and a Random Search Algorithm) provide different results for the Dynamic Multilevel Component Selection Problem.

According to the experiment presented in Section 5, subsection 5.1.1 and 5.1.2, outline of the results (regarding the influence of iterations number and population size in finding the Pareto solutions) are the same as in the first case study.

According to the Wilcoxon statistical test values presented in Section 5, subsection 5.1.1 and 5.1.2, the outline of the results (regarding the different results obtained by the Genetic Algorithm and by the Random Search Algorithm) is as follows:

- We have statistically significant evidence at $\alpha = 0.05$, to show that the median difference is positive, i.e. the Null-Hypothesis is rejected, in favor to H_1.

5.2. **Summary Report.** As stated in Section 3 to generalize from a case study to theory, the analytic generalization should be used. If two or more cases are shown to support the same theory, replication [5] may be claimed. Our replication strategy used time steps for the the dynamic changing requirements (Level 1) and for the dynamic changing components (Level 1, 2, and 3).

The time steps were modified according to Table 3: the number of time steps were modified for each case study, and also the number or requirements/components modified for each time step.

In first case study we have used 4 time steps for the changing requirements experiment and 3 for the second case study, respectively 5 time steps for

TABLE 3. Replication strategy - time steps for each conducted case study

Case study 1 [1]	Case study 2 [2]	Case study 3 (current paper)
Changing requirements		

T1=init req.
T2= +1R
T3= -1R and +1R
T4= +1R
T5=
	T1=init req.
	T2=+1R
	T3=+1R
	T4=
	T5=
		T1=init req.
		T2=+1R
		T3=+1R
		T4=-1R
		T5=+1R

Changing components		

Level 1
T1=init comp.
T2= +2C
T3= -1C
T4= -1C and +1C
T5= +3C
	Level 1
	T1=init comp.
	T2=+2C
	T3=-1C and +1C
	T4=-1C
	T5=
		Level 1
		T1=init comp.
		T2=+2C
		T3=
		T4=
		T5=

Level 2
T1=init comp.
T2= +2C
T3= +3C and -1C
T4=
	Level 2
	T1=init comp.
	T2=+2C
	T3=
	T4=
		Level 2
		T1=init comp.
		T2=+3C and -3C
		T3= +2C
		T4= +1C and -1C

Level 3
T1=init comp.
T2= +2C
T3= -2C
T4= +2C and -1C
	Level 3
	T1=init comp.
	T2=+2C and -2C
	T3= -3C and +1 C
	T4=
		Level 3
		T1=init comp.
		T2=+2C and -2C
		T3= +1C and -1C
		T4= +1C and -2C

the third case study. Regarding the changing components experiment we have used different time steps for all three levels of the case studies: 3 to 5 time steps for the first case study, 2 to 4 time steps for the second case study, and 2 to 4 time steps for the third case study. Also, the number of requirements/components to be changed were considered for variation: either adding or removing 1 to 3 elements. Thus, our replications altered one or two experimental conditions considered unimportant to the original findings, to see whether the findings could still be duplicated. With such replications [5] the original finding should be considered robust.

Each case's conclusions are next considered for the summary report.

For each individual case, the report ([1, 2] and current paper, Section 5.1) indicated that the proposition from Section 5 was demonstrated, i.e. "The Search-based Algorithms (in our case a Genetic Algorithm and a Random Search Algorithm) provided different results for the Dynamic Multilevel Component Selection Problem because in the case of Genetic Algorithm the fitness of a solution is reevaluated."

Regarding the research question from Section 5, i.e. "How and Why do Search-based Algorithms (in our case a Genetic Algorithm and a Random Search Algorithm) provide different results for the Dynamic Multilevel Component Selection Problem?", the conclusions sustained by the conducted case studies is that the Genetic Algorithm provides better results than the Random Search Algorithm for the Dynamic Multilevel Component Selection Problem and that we have statistically significant evidence at $\alpha = 0.05$.

6. Related work analysis and discussion

This section presents the current state of art regarding the component selection problem and analyzes the differences compared with our present approach. Component selection methods are traditionally done in an architecture-centric manner. One approach was proposed [20], where the authors present a method for simultaneously defining software architecture and selecting off-the-shelf components.

Another type of component selection approach is built around the relationship between requirements and components available for use. Paper [19] proposes a comparison between a Greedy algorithm and a Genetic Algorithm. The discussed problem considers a realistic case in which cost of components may be different. *In relation to existing component selection methods, our approach aims to achieve goals similar to* [18, 17]. The [18] approach considers selecting the component with the maximal number of provided operations. The algorithm in [17] considers all the components to be previous sorted according to their weight value. Then all components with the highest weight are included in the solution until the budget bound has been reached.

All the above approaches did not considered the multilevel structure of a component-based system. They all constructed the final solution as a one level system. Our previous research has studied the problem of multilevel component selection considering multilevel configuration [25]. The proposed evolutionary multiobjective approach provided a way of finding the "best" solution out of a set of solutions.

Various genetic algorithms representations were proposed in [26, 27]. The authors proposed an optimization model of software components selection for

CBSS development. The proposed methodology involves some subjective judgments from software development team, such as determination of the score of interactions and the function ratings. *We argue that our model differs by the fact that components interactions are computed automatically* based on required and provided component interface specification. Also, regarding the function ratings (that describe the degrees of functional contributions of the software components towards the software modules), *our approach discovers automatically the constituent components for each module of the final system.*

In [12] a hybrid approach for multi-attribute QoS optimization of component-based software systems has been proposed. The approach is able to exploit the approximated analytical Pareto front providing a larger number of solutions with a more accurate estimate of performance and availability metrics. *In relation to this existing approach, ours aims to achieve similar goals, being capable of obtaining multiple solutions in a single run and it can be scaled to any number of components and requirements.*

Another perspective refers to updating/adding/removing one/many requirements (components) from an already constructed system [24]. Our previous research regarding this perspective was proposed in [15]. How to deal efficiently with the design of systems that are able to evolve overtime and adapt to rapid changes of their requirements was investigated in [14]. They proposed some metrics definitions that are able to quantify and evaluate such software adaptability at the architectural level. *Our current approach considers dynamic modifications of the requirements of the final system, investigating different ways of modifying the requirements, by adding new requirements or deleting existing ones.*

A similar approach considering evolution of software architecture was proposed by [4]. It suggests the best actions to be taken according to a set of new requirements. They associate to the evolution of a new requirement a set of plans to be applied. The model select the best available evolution plan such that it minimizes the evolution cost under reliability and performance constraints. In relation to this approach, *our current approach also discovers the optimal solution minimizing the final cost when new requirements are needed, and it also considers the case that the component repository changes over time (that was not included in the [4] study.)*

In the course of experiments during the evaluation, a number of limitations of the dynamic multilevel component selection algorithm became apparent. First, it does not take into consideration many factors, which are effective in component selection, such as performance and reliability. However, it is possible to extend the algorithm to consider other factors. The quality factors (other than cost), mapped to some existing component-based metrics may be included in the fitness function of the genetic algorithm. Or the values of the metrics may assess the final obtained solutions, offering the architect a range of solutions with various measurements of a quality attribute or for many quality attributes.

The second limitation refers to the NP-complete problem of the component selection problem, therefore, like other existing methods, the proposed algorithm cannot guarantee to achieve an optimal solution (using the Pareto dominance principle it might be difficult to always find solutions which are better than the ones already found). However, we have introduced a supplementary condition for comparing two solutions which are nondominated among them (will prefer the one for which the aggregation of all of the objectives values lower).

7. Conclusion

The contribution of this paper is the use of the case study method and the research design from the book of Yin [5] to validate our research proposal for the **Dynamic Multilevel Component Selection Problem** [2].

We have conducted three different experiments. The first one was reported in [1], and the second one in [2]. The current paper **reports the third experiment**. Following the replication approach to multiple-case studies [5], each individual case study was finalized by an individual case report that will be next considered to be part of a summary report, i.e. a cross-case conclusions. Thus, in our case the results obtained are reported and conclusions about the potential of evolutionary algorithms for the dynamic multiobjective multilevel component selection problem are drawn.

The Wilcoxon statistical test was used to compare our Genetic Algorithm approach with a Random Search Algorithm: we have statistically significant evidence at $\alpha = 0.05$ to show that the median is positive, i.e. we obtain better results with our approach. The tests performed show the potential of evolutionary algorithms for this particular problem and for other similar ones.

Work-in-progress is devoted to improve the proposed approach in various ways. First, we are integrating the approach with the approach from [13]: to use fuzzy clustering to group similar components in order to select the best candidate.

Ongoing work focuses on the integration of metrics to asses the quality of the obtained solutions. We plan to integrate the evaluation of the final level configurations (an approach that is published in [3]) in the current approach by computing metrics values and assess and select the best solution architecture.

References

[1] A. Vescan, *An Evolutionary Multiobjective Approach for the Dynamic Multilevel Component Selection Problem*, The First International Workshop on Big Data Services and Computational Intelligence, in conjunction with ICSOC, 193–204, 2016.

[2] A. Vescan, *Case Study Method and Research Design for the Dynamic Multilevel Component Selection Problem*, The First International Workshop on Big Data Services and Computational Intelligence, in conjunction with ICSOC, 130–141, 2016.

[3] A. Vescan, C. Serban, *Multilevel component selection optimisation towards an optimal architecture*, Soft Computing Journal (accepted December 2016).

[4] V. Cortellessa and R. Mirandola and P. Potena *Managing the evolution of a software architecture at minimal cost under performance and reliability constraints*, Science of Computer Programming, no. 98, pp. 439–463, 2015.

[5] Robert K. Yin *Case Study Research: Design and Methods*, SAGE Publications, 2009.

[6] J. Derrac and S. Garcia and D. Molina and F. Herrera, *A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms*, Swarm and Evolutionary Computation, no. 1, pp. 3–18, 2011.

[7] Arcuri and L. Briand, *A practical guide for using statistical tests to assess randomized algorithms in software engineering*, The 33rd International Conference on Software Engineering, 1–10, 2011.

[8] M. Harman and P. McMinn and J. Teixeira de Souza and S. Yoo *Search Based Software Engineering: Techniques, Taxonomy, Tutorial*, Empirical Software Engineering and Verification, no. 7007, pp. 1–59, 2012.

[9] L. Iribarne and J.M. Troya and A. Vallecillo, *Selecting Software Components with Multiple Interfaces*, The 28th EUROMICRO Conference Component-Based Software Engineering, 26–32, 2002.

[10] Christoph Becker and Andreas Rauber *Improving component selection and monitoring with controlled experimentation and automated measurements*, Information and Software Technology, no. 6, pp. 641–655, 2010.

[11] M. A. Khan and S. Mahmood *A graph-based requirements clustering approach for component selection*, Adv. Eng. Software, no. 54, pp. 1–16, 2012.

[12] A. Martens and R.Mirandola and D. Ardagna and R. Reussner and H. Koziolek, *A Hybrid Approach for Multi-Attribute QoS Optimisation in Component Based Software Systems*, Proc. of the QoSA, 84–101, 2010.

[13] A. Vescan, C. Grosan, *A new Component Selection Algorithm Based on Metrics and Fuzzy Clustering Analysis*, Proceedings of the 4th International Conference on Hybrid Artificial Intelligence Systems, pp. 621–628, 2009.

[14] D. P. Palacin and R. Mirandola and J. Merseguer *Software Architecture Adaptability Metrics for QoS-based Self-Adaptation*, Proc. of the QoSA, pp. 171–176, 2011.

[15] A. Vescan .and Grosan, C. and Shengxiang Yang *A hybrid evolutionary multiobjective approach for the dynamic component selection problem* , Proc. of the 11th International Conference on Hybrid Intelligent Systems (HIS), pp. 714–721, 2011.

[16] A. Vescan and C. Serban *etails on case study for the dynamic multilevel component selection optimisation approach*, http://www.cs.ubbcluj.ro/˜avescan/?q=node/178, 2016.

[17] P. Baker and M. Harman and K. Steinhofel and A. Skaliotis *Search Based Approaches to Component Selection and Prioritization for the Next Release Problem*, Software Maintenance, The 22nd IEEE International Conference on, pp. 176–185, 2006.

[18] M. R. Fox and D. C. Brogan and P. F. Reynolds *Approximating Component Selection*, Software Maintenance, Proceedings of the 36th Conference on Winter Simulation, pp. 429–434, 2004.

[19] N. Haghpanah, S. Moaven, J. Habibi, M. Kargar, S. H. Yeganeh, *Approximation Algorithms for Software Component Selection Problem*, APSEC conference, pp. 159–166, 2007.

[20] E. Mancebo, A. Andrews, *A strategy for selecting multiple components*, SAC '05: Proceedings of the 2005 ACM symposium on Applied computing, pp. 1505–1510, 2005.

[21] A. Abraham and L. Jain and R. Goldberg *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, Springer Verlag, 2005.

[22] C. Grosan *A comparison of several evolutionary models and representations for multiobjective optimization*, ISE Book Series on Real Word Multi-Objective System Engineering, chapter 3, Nova Science, 2005.

[23] I. Crnkovic, M. Larsson, *Building Reliable Component-Based Software Systems*, Artech House publisher, 2002.

[24] L. Wei *QoS Assurance for Dynamic Reconfiguration of Component-Based Software Systems*, IEEE Transactions on Software Engineering, no. 38(3), pp. 658–676, 2012.

[25] A. Vescan and C. Grosan *Evolutionary multiobjective approach for multilevel component composition*, Studia Univ. Babes-Bolyai, Informatica,

no. LV(4), pp. 18–32, 2010.

[26] C.K. Kwong and L.F. Mu and J.F. Tang and X.G. Luo *Optimization of software components selection for component-based software system development*, Computers and Industrial Engineering, no. 58(1), pp. 618–624, 2010.

[27] P.C. Jhaa and V. Balib and S. Narulaa and M. Kalra *Optimal component selection based on cohesion and coupling for component based software system under build-or-buy scheme*, Journal of Computational Science, no. 5(2), pp. 233–242, 2014.

Department of Computer Science, Faculty of Mathematics and Computer Science, Babeş-Bolyai University, Cluj-Napoca, Romania

E-mail address: `avescan@cs.ubbcluj.ro`