On Architectural Diversity of Dynamic Adaptive Systems

Hui Song*, Amal Elgammal†, Vivek Nallur†, Franck Chauvel*, Franck Fleurey*, Siobhán Clarke†
*SINTEF ICT, Oslo, Norway. Email: first.last@sintef.no,
†Distributed Systems Group, Trinity College Dublin, Ireland. Email: first.last@scss.tcd.ie

***Abstract*—We introduce a novel concept of "architecture diversity" for adaptive systems and posit that increased diversity has an inverse correlation with adaptation costs. We propose an index to quantify diversity and a static method to estimate the adaptation cost, and conduct an initial experiment on an exemplar cloud-based system which reveals the posited correlation.**

I. Introduction

Dynamic adaptation is key feature in modern software-based systems. Adaptation approaches that are classified as *strong* [1] deal with high-cost and extensive-impact actions, such as replacing components and migrating resources. For such approaches, *adaptation cost*, such as the average time needed to replace components for one adaptation loop, is an important criterion to determine whether an adaptive approach is generically usable for the target system.

Adaptation cost can be lowered in different ways, such as reducing the frequency of changes, optimising adaptation algorithms, etc., These solutions require efforts in the implementation, design, or even requirement analysis phases. However, for the systems that have gone through these early development phases and come into the hands of system administrators, the question becomes: *How should the system be deployed such that it enables cheaper adaptations in the future?*

In order to answer this question, we study and observe the analogy between software engineering and ecology, which is the domain that investigates how biological systems are prepared to deal with their environmental changes. In ecology, the answer is *diversity*. A diverse system is a coalition of a large number of different species and a balanced distribution of individuals among the species. It is in general more adaptable to environmental changes, as opposed to a homogeneous one.

Diversity also exists in adaptive systems. At the architecture level, a diverse system is deployed with alternative and complementary types of components, with a good balance between the types. If the phenomenon of biological diversification is transferred to adaptive software systems, a diverse system should have positive influence on adaptation cost. To validate and exploit this hypothesis, we developed a novel constraint solving-based approach to statically estimate the minimal adaptation cost for given architectures, and provide a simple quantitative measurement of architectural diversity, based on a popular diversity index in ecology. Using these techniques, we conduct an initial experiment on a cloud-based Geographic Information System (GIS) named SMART-GH. The results reveal a positive correlation between *architecture diversity* of the deployed system and its *adaptation cost*.

The contribution of this approach is twofold: It provides a novel way to predicate and optimise the adaptation cost through a simple measurement based on diversity. It also suggests subsequent investigations on architecture diversity through the transfer of concepts and research methods from ecology to software engineering.

The rest of this paper is organized as follows: Section II introduces the GIS case study. Section III discusses adaptation cost and its quantification. The concept of architectural diversity is explicated in Section IV, which is followed by discussing the experimental setting and results. Finally, conclusions and future work is highlighted in Section VII.

II. Case Study Diversification

SMART-GH is a route planning application developed by Trinity College Dublin[1]. It searches routes on a city scale taking into consideration the sensor data such as pollution, noise, etc. The basic architecture of SMART-GH includes a set of *aggregators* which collect and process data from sensors, participatory sensing or public data sources and populate them into a Redis database. Sensor readings are linked to OpenStreetMap (OSM) maps for routing algorithms to return a smart route. SMART-GH components can be hosted on IaaS-based virtual machines, and a Redis database can also be rented from PaaS providers. For the sake of performance, each routing algorithm (for car, foot, shortest route, etc.) or aggregator (for noise, pollution, etc.) is implemented as an individual (and thus specifically optimized) component, wrapped as a docker image.[2] Consequently, when the context is changed, new components may need to be instantiated, or new connections be created. Therefore, SMART-GH is a typical architecture-based, *strong* adaptation system.

Fig. 1 illustrates three sample architectures of SMART-GH. Suppose, in the beginning all customers use SMART-GH to plan drive (car) routes based on real-time traffic data, and we have the deployment architecture as shown in Fig. 1(a), where three routing components are provided for load-balancing and backup purpose. From this architecture, adaptations to context changes are costly. If the user changes her routing preference to "foot considering pollution", we need to instantiate a

[1] https://github.com/DIVERSIFY-project/SMART-GH
[2] All the images can be found at https://hub.docker.com/u/songhui/

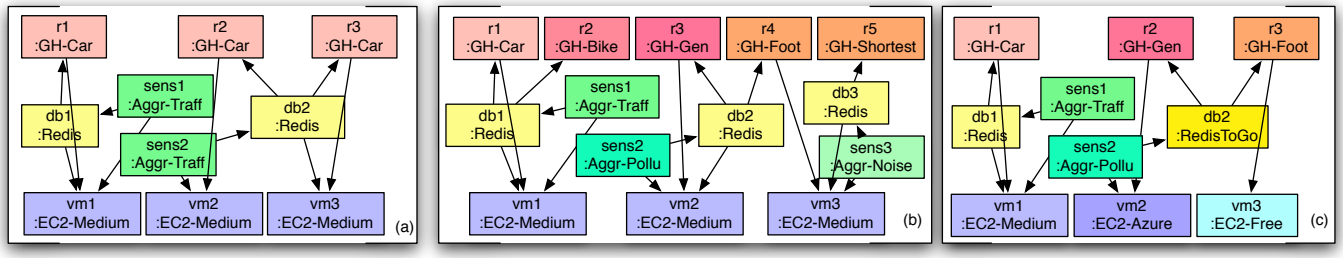

Fig. 1. Sample architectures of SMART-GH, where different colours illustrate different component types

new `GH-foot` component and an `Aggr-Pollu` component, which has a big cost in terms of the time spent on deployment and initialization. Moreover, if there are no available spaces in any existing VMs, a new one needs to be created, which adds a lot more cost. Since the architecture in Fig. 1(a) only fits user's preference combined with car routing and traffic status, there exist many possible changes for which we need to pay the cost of instantiating new components, and thus, on average, the architecture is quite costly in terms of adaptation.

A slightly diverse deployment in Fig. 1(b) may need less adaptation cost: If the changed user preference is a combination of the five routing options and the three data types as listed in the figure, there will be only trivial cost of re-connecting components. However, there are still possible changes which incur a high cost of adaptation. For example, if the user chooses to use free services, we need to initialize a free EC2 virtual machine and migrate the relevant components to it.

The last diversified architecture in Fig. 1(c) is more prepared to address the above changes. For a combination of the three routing options and two data types, there will be only the cost of re-connection. Although there are changes that need more cost in arch. (c) than arch. (b), e.g., biking considering noise, the cost is still only to deploy one or two components. However, for some other changes, such as to a free service, arch. (c) needs much less cost: no need to initialize VMs or databases, and no migration. Statistically speaking, it is highly probable that arch. (c) may need less cost on adaptation.

From the three sample architectures shown in Fig. 1, we can claim that *a more diverse system has lower expected adaptation cost*. To investigate this hypothesis, we now define *adaptation cost* and *architectural diversity* and discuss how to measure them in a quantitative way.

III. Adaptation Cost and Its Estimation

In this paper we focus on the adaptation cost on the execution phase. In particular, we define the cost of a single adaptation loop as the sum of costs for all the modifications that are made to the system. For each modification, we care about its cost at an abstract level using a relative value to synthetically reflect the concrete costs such as time spent, resource consumption. For example, from Figure 1(b), the user's preference might change to search least noisy "foot" routes on a free service level. The adaptation action consists of the following modifications: create a new EC2-Free virtual machine named vm4 (cost: 20), migrate r4 to vm4 (cost:4), and reconfigure it to obtain data from db3 (cost:1). The cost of this adaptation loop is 25. In our experiment, such relative cost numbers are obtained via a rough experiment of deploying SMART-GH on SINTEF's internal cloud platform.

Adaptation cost can be lowered by means of pruning unnecessary modifications or looking for cheaper alternatives. However, such improvement has a limitation. In other words, for a given adaptation loop, there exists a minimal cost, and any adaptation solution with a cost lower than it cannot satisfy the adaptation objectives. In this paper, we only measure such minimal cost, so that the estimation is not dependent on any concrete algorithm. For a given adaptation loop, we use a *static analysis approach* to estimate the minimal adaptation cost: It does not actually execute any adaptation algorithms, but theoretically computes the minimal cost.

We implement static estimation as a constraint solving approach [2] based on Satisfaction Modulo Theories (SMT) and soft constraints. We use constraint solving to seek for a new architecture that, together with the changed context, satisfies the adaptation objectives (constraints). Constraints are classified into:(i) *hard constraints*, that must be satisfied, and (ii) *soft constraints*, that may be violated when necessary. We use hard constraints to simulate adaptation objectives, and introduce a set of *soft* ones which attempts to force the new architecture to be the same as the original one before adaptation. If the adaptation modifies the system state, it will violate the corresponding soft constraint, and the cost is reflected by the weight of this violated constraint.

We use the sample in Fig. 1 to illustrate the approach. We first define an enumeration type on all existing components, as well as the potential ones that may be added after the adaptation. Configurations inside a component and relationships between components are represented by uninterpreted functions. Each component also has a function indicating whether it is used in the current configuration, and another function denoting its type. We define hard constraints (i.e., adaptation objectives) such as: Any routing components must be hosted by a virtual machine; A VM cannot host applications that exceed its capacity; If the user prefers "foot", the current main router[3] should not be the one for "car" or "bike"; If the

[3]At any particular time point, there is one and only one main router serving the user, others are candidates for upcoming users.

user requires free service, the main router should be hosted in a free VM, etc. Finally, we define a set of soft constraints corresponding to the current architecture. Each soft constraint has a *weight*, which is assigned according to the modification cost we discussed above. In order to estimate the minimal cost for a given adaptation loop, we feed all the variables, functions and constraints into a constraint solver, and obtain a solution that minimises the total weights of violated soft constraints. Details on constraint modelling and solving can be found in our previous paper [2].

IV. Architectural Diversity

The concept of diversity in ecology has three aspects: *variety* (the number of different species), *balance* (the distribution of individuals between these species) and *disparity* (how different the species are from one another). Since the difference between species are too fine-grained and hard to obtain, usually only variety and balance are considered for diversity measurement on big and complex systems. A widely used quantitative diversity measurement in ecology, combining variety and balance, is the Shannon index:

$$H = -\sum_{i=1}^{R} p_i \ln p_i, \texttt{where } p_i = \frac{n_i}{\sum_{i=1}^{R} n_i}$$

Here R is the number of species, and n_i is the number of individuals of the i-th species. Bigger R leads to bigger H, and so does more equally distributed n_i's. For a given R, H has an upper limit of $H_{max} = \ln R$, when all n_i's are the same. This also means that the Shannon index is neither normalized nor convergent: one can always add more species to increase the Shannon index, which is also true for diversity. We use Shannon index to quantify the diversity of system architecture: Each component type is mapped to a species, and each component instance is mapped into an individual. In Fig. 1(a), there are 4 species. The numbers of individuals are (3, 2, 2, 3), and $H_a = 1.37$. Similarly, for Fig. 1(b) $H_b = 2.17$, higher than H_a because of more species. For Fig. 1(c), $H_c = 2.30$. It has the same number of species, but the individuals are more equally distributed among the species.

V. Initial Experiments and Results

We conduct an experiment to investigate the correlation between architectural diversity and adaptation cost applied to SMART-GH system. The experiment starts from a pool of valid architectures, and performs changes on each of them; then it computes the minimal adaptation cost. Finally, we record the average minimal adaptation cost for each architecture, and observe the correlation between the cost and the diversity.

We generate a pool of 200 different deployments (architecture-context pairs) of SMART-GH, with the following requirements: 1) The architecture is consistent to the context, otherwise, the adaptation may need to pay extra cost to solve the inconsistency, which pollutes the result. 2) The architectures should cover a wide range of diversities, so that results are more representative. 3) There should not be duplicate architectures in order to avoid useless estimations.

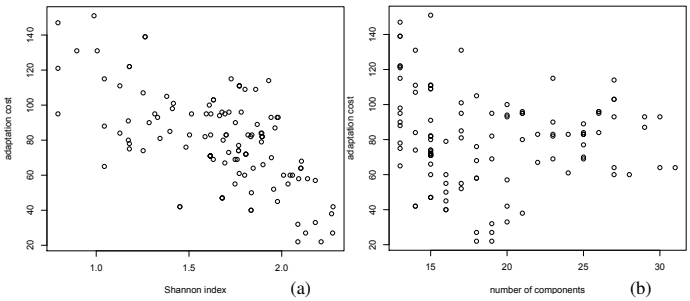

Fig. 2. Experiment result of correlation between diversity and adaptation cost

We initialize the pool with 5 minimal deployments, and then keep enlarging the pool by picking an existing deployment, mutating it, and putting back the new one to the pool (if there are no duplicated ones). The following three mutation rules are used: 1) *Adding a new component with the same type of an existing component*, which generates a less diverse architecture. 2) *Adding a new component with a random type*, which in general increases diversity, but may have an adverse effect when the architecture is already highly diverse. 3) *Picking two components of the same type, and forcing their types to be different* (by adding a temporal constraint), which increases diversity. 4) *Changing a context variable*. To ensure valid deployments, we do a round of constraint solving after each mutation to calibrate the architecture according to the context.

For each architecture, we generate 20 changes on either architecture or context, from the following three categories. 1) *Context change*: picking 1 to 3 context variables, and changing their values, 2) *Instance failure*: picking 1 to 3 components, and setting them to be not alive, 3) *Type failure*: picking a type, and setting all the components of this type to be not alive. After each change, we estimate the minimal adaptation cost to make the system consistent, and then record the average minimal cost for all the 20 changes.

Fig. 2(a) illustrates the results of this experiment. Each circle represents an architecture in the pool. The x-axis is the diversity measured by Shannon index, and the y-axis is the average adaptation cost from the 20 changes. The result shows an obvious correlation between diversity and adaptation cost, and as shown in Fig. 2(b), the improvement on adaptation cost is not caused by the increase of the architecture size (i.e., the number of components, which often increases along with diversity). The results reveal that *For the case of* SMART-GH *system, under the given types of changes, architectural diversity helps in decreasing the average cost of adaptation.*

It is also worth noting some interesting details. Firstly, the average cost of different architectures vary more significantly among low-diversity deployments (e.g., $H < 1.3$) than among high-diversity ones ($H > 2.0$). It is mainly caused by the imbalance between component types, i.e., some components fit more contexts than others. For low-diversity architectures, the cost is less predictable: The ones with "versatile" components are more prepared to changes than the one with only

"specialized" components. On the other hand, high-diversity architectures tend to consist of all types of components, and thus the difference between architectures are small. Secondly, the lowest average costs appears in the architectures with around 20 components, and they are also the most diverse ones. In fact, 19 is the smallest number of components with all the component types instantiated. After that, adding new components does not lower the cost further. Moreover, since more applications are hosted on VMs, it is more likely that some applications have to be migrated to adapt to some changes, and therefore, the average cost also increases.

VI. Related Work

Adaptation cost is an import aspect of self-adaptive software. In a recent survey, Salehie and Tahvildari [1] use *impact and cost* dimension in their taxonomy of research efforts for adaptive software, and here cost means the time and resource to execute an adaptation action. Villegas et al.'s measurement framework [3] also quantifies the *disturbance* to the system and the *settling time* to execute an adaptation action. However, there are not many attempts that we know of, which focus on cost control. From the perspective of adaptation objectives, Cheng et al. [4] provide an approach to relax the adaptation goals, in order to avoid unnecessary actions on small changes. From the perspective of adaptation algorithms, in our previous work [2], we select adaptation plans by considering both effect and cost of their modifications. This paper opens a new direction to control the adaptation cost by optimizing system architecture before adaptation.

Software diversity has been gaining interest recently. In the security domain, there is a big stream of research focusing on automated randomization of software artefacts, on the source code, binary code, or runtime level. The objective is to increase the difficulty for adversaries to attack the artefact [5]. There are also approaches to diversify network topology [6] or artefact dependencies [7] to reduce the global damage of attacks. In this work, we focus on architectural diversity emerging from the deployment of diverse artefacts, and investigate diversity in a new domain; the adaptation cost. The idea follows our previous publication [8], but that paper is focused on how to achieve more diverse architectures technically, rather than the measurement and exploitation of diversity.

VII. Conclusions and Future Work

This paper reports on our initial investigation on how architectural diversity of dynamic adaptive systems influences their adaptation cost. We use Shannon index as a simple quantitative measurement for architectural diversity, and provide a constraint solving-based approach to statically estimate the minimal adaptation cost on different architectures.An experiment on a sample cloud-based system shows a positive correlation between diversity and adaptation cost.

The conducted experiment is an initial attempt towards the investigation of architectural diversity. Future work involves the experimentations on different large-scale adaptive systems that involve more types of changes, which will enable us to draw general conclusions. In the current experiment, adaptation is simplified by means of satisfying constraints. Future experiments will also consider other kinds of adaptation using a multi-objective optimisation approach to combine both effect and cost of adaptation. Furthermore, we treated the cost of individual architecture modifications as isolated values, while in practice, performing one modification may influence the cost of subsequent ones. We will introduce more sophisticated cost models based on the observation of large-scale systems.

Besides providing a reference for administrators to deploy systems, the diversity can be also exploited in other ways to help control adaptation cost. At runtime, a proper diversity control could rejuvenate the system from being homogeneous after a long run of adaptation, and thus avoid the rising of adaptation cost. At design time, assessment on the potential maximal diversity can be an indicator to help decide the types and dependencies provided for a system.

We envision that adaptation cost is just one benefit of architectural diversity, but there are also others such as performance, resilience, security, etc. We also envision further application of concepts, theories and methodologies from ecology to software engineering. Software systems are no longer individual artefacts running in a closed environment. A system may contain thousands of components with complex interactions, which resembles an ecological ecosystem. Future work in this direction includes investigating diversity by combining software and social systems, introducing provision and extinction mechanism into software systems to achieve given properties, and imitating organisation of natural systems in software ones.

VIII. Acknowledgement

This work is partially supported by the EU FP7-ICT-2011-9 No. 600654 DIVERSIFY project

References

[1] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," *ACM TAAS*, vol. 4, no. 2, p. 14, 2009.

[2] H. Song, X. Zhang, N. Ferry, F. Chauvel, A. Solberg, and G. Huang, "Modelling adaptation policies as domain-specific constraints," in *MODELS*, 2014, pp. 269–285.

[3] N. M. Villegas, H. A. Müller, G. Tamura, L. Duchien, and R. Casallas, "A framework for evaluating quality-driven self-adaptive software systems," in *SASO*. ACM, 2011, pp. 80–89.

[4] B. H. Cheng, P. Sawyer, N. Bencomo, and J. Whittle, "A goal-based modeling approach to develop requirements of an adaptive system with environmental uncertainty," in *MODELS*, 2009, pp. 468–483.

[5] P. Larsen, A. Homescu, S. Brunthaler, and M. Franz, "Sok: Automated software diversity," in *Security and Privacy (SP), 2014 IEEE Symposium on*, 2014, pp. 276–291.

[6] J. Caballero, T. Kampouris, D. Song, and J. Wang, "Would diversity really increase the robustness of the routing infrastructure against software defects?" *Research Showcase Department of Computer Engineering, CMU*, p. 40, 2008.

[7] A. Gorbenko, V. Kharchenko, O. Tarasyuk, and A. Romanovsky, "Using diversity in cloud-based deployment environment to avoid intrusions," in *Workshop on Software Engineering for Resilient Systems*, 2011.

[8] S. Allier, O. Barais, B. Baudry, J. Bourcier, E. Daubert, F. Fleurey, M. Monperrus, H. Song, and M. Tricoire, "Multi-tier diversification in internet-based software applications," *IEEE Software*, 2015.