2023 IEEE 20th International Conference on Software Architecture Companion (ICSA-C) | 978-1-6654-6459-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICSA-C57050.2023.00045

Role-playing software architecture styles

Laura M. Castro
Centro de Investigación en TIC (CITIC)
Universidade da Coruña
A Coruña, Spain
lcastro@udc.es – https://orcid.org/0000-0002-3028-1523

***Abstract*—Software Architecture, from definition to maintenance and evolution, is a complex aspect of software development and, consequently, a challenging subject when it comes to teaching it, and learning it.**

Many research efforts have been devoted to designing teaching approaches, strategies and tools. Most of them, however, focus on the knowledge itself and the ways to convey it to students, rather than on the different learning styles of students themselves.

Teaching methods which predominantly rely on verbal and written communication, are very well aligned with some learning styles. However, students with learning styles that benefit more from physical activity or first-hand experience, need to defer to cognitive processes that are less natural to them.

In this work, we propose an innovative use of role-playing as teaching strategy for architecture models of reference (i.e. layered, pipe & filter, client-server, etc.). This role-playing of different software architectures, in which students play the part of specific components in the system, intends to complement other classical teaching materials, such as in-person or recorded lectures, lab assignments, or development projects.

Addressing all learning styles within a classroom is key to ensure that we favour and foster the students' different learning processes, and give everyone an even playfield in which to best develop their capabilities as Software Architects.

***Index Terms*—learning styles, role-playing, software architecture models**

I. Introduction

Teaching Software Architecture (SA) in a meaningful and effective way is difficult. Many reasons have been posed to explain this reality, although there seems to be some consensus around the idea that SA exposes students to concepts that have significantly greater scale than handled before, and which are not easy to reproduce within the classroom-and-academic-term context [13]. These concepts require the development of correspondingly increased abilities of abstraction, being also often the first time issues like performance or security are addressed [4]. Last but not least, there are non-technical skills at play, such as communication [7] and decision-making [2], that need to be fostered, too [3].

In this paper, we propose a SA teaching approach driven by the co-existence of different learning profiles, as defined by David Kolb [6], [8], in the classroom. Taking into account that traditional activities such as lectures and development projects are more aligned with some of those learning profiles, we contribute a set of role-playing activities designed to improve student's understanding of different software architectures characteristics, strengths and weaknesses, that target those learning profiles which are usually forgotten.

Supported by the Centro de Investigación de Galicia "CITIC", funded by Xunta de Galicia and the European Regional Development Fund (grant ED431G 2019/01.

II. Role-playing as a teaching tool

The use of participatory teaching methods features a wide range of options which are applicable in the context of computer science [5]: brainstorming, directed dialogues, small discussion groups, debates, panel discussions, or role-playing, amongst others. From this range of choices, for this work we have decided to use role-playing as a teaching tool.

Role-playing, sometimes referred to as *role-play simulation* in educational settings, is an experiential learning method in which learners are involved in a proposed scenario by representing an interacting part in it. The scenario is outlined by the teacher or professional, and while it must allow improvisation, it represents a safe and supportive environment where students will develop their own meaningful first-person experience.

Role-playing is widely acknowledged as a powerful technique across multiple avenues of training and education. There is previous experience in software engineering featuring role-playing, most often to simulate the conditions of an industrial environment [10], of software product lines [16], or for requirements engineering [15]. In these cases, students are asked to play the part of stakeholders, from clients to the different technical roles within the development team.

However, the way we have chosen to use role-playing to teach SA is not by assigning human roles to students, but by assigning them the roles of software components, as we will discuss in Section III. While "traditional" (i.e. human-focused) role-playing has been used before in the context of SA teaching [9] (as a trade-off analysis method), we have only found one previous similar approach to ours (i.e. software-focused): specifically, in the context of a programming course where students play the part of objects in order to grasp the concepts of object-orientation [1].

III. Role-playing architecture models

SA at University of A Coruña is a 6-ECTS (*European Credit Transfer and accumulation System* [12]) course for Software Engineering students. This translates in practice into 150 hours of work, typically spread amongst the 15 weeks of a term. Of the 10 weekly hours, 3 correspond to class and lab hours, and 7 to student autonomous work. In this course, students are exposed to architectural concepts and practices:

Authorized licensed use limited to: University of Canberra. Downloaded on March 17,2025 at 03:38:26 UTC from IEEE Xplore. Restrictions apply.

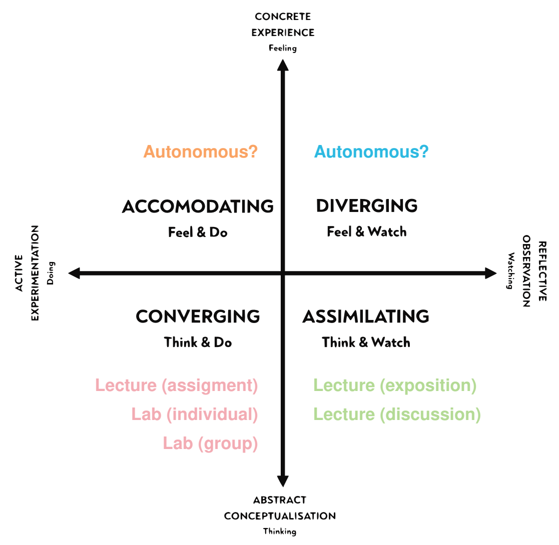

Fig. 1. SA course at UDC: link between methodologies/learning styles

from architectural patterns to non-functional requirements' analysis, the impact of the latter in the former, architectural representation and modelling, etc. The goal is to offer students a learning environment in which they can acquire the skills that allow them to carry out architectural tasks in the context of software development: component identification and characterisation, assignment of responsibilities, motivated election of communication and integration alternatives, and architectural evolution and maintenance.

The teaching methodologies that were being used in the SA course at UDC were clearly aligned with some of Kolb's learning styles (cfg. 1). Lectures, and the materials provided (bibliographic references, articles, videos, assignments) work well for people which more prominent learning traits are watching and thinking, i.e. those students that are reflective and take the most out of observation, and conceptualisation. Complementarily, practical work during lab sessions gave additional coverage to those most prominently guided by active experimentation. However, those with *accommodating* and *diverging* learning styles, in which experimentation or observation would most benefit from combination with concrete experience of the subject at hand, are entirely left up to self-manage their needs in their autonomous work.

Our proposal to address this situation was to role-play different architectures of reference, meaning that **scenarios were designed to provide students with a concrete experience of the structure, advantages and disadvantages of different architectural models**.

A. Role-playing a layered architecture

The instructions given for the first role-playing scenario are shown in Table I. Students were made to form systems with different number of layers, but the same functionality. The expected outcomes are first-hand experiences of:

- Layers reuse from one system to the next, increased when responsibilities are specific rather than broad.
- Limited impact of changes, affecting to single layers (or adjacent, at most).
- Request processing in two different moments at each layer (on the way *in*, on the way *out*, on both), and consequences (i.e., security).
- Structural limitations to performance, related to the number of layers.

B. Role-playing a pipe & filter architecture

The instructions given for role-playing a pipeline are shown in Table II (page 3). Groups of students were asked to define their own filters, and design their own pipeline. Expected outcomes were first-hand experience of:

- Filters reuse from one pipeline to the next, increased when responsibilities are specific rather than broad.
- Limited impact of changes, affecting to single filters (or adjacent, at most).
- Advantage to performance given by structural parallelism, increased by the possibility of duplicating filter stages when one is a bottleneck.
- Differentiated input and output points in the system.
- One-way path for requests and consequences (i.e. error handling).

C. Role-playing a client-server architecture

Table III (page 3) shows the instructions given for the client-server scenario, the first of the distributed architectural models. Students showcased the creation and evolution of several systems. The fist-hand experiences outcomes were:

- Independence of services, in terms of development, availability, etc.
- Robustness: to the ability to have a working system as soon as the first service, and the directory, are operational, service independence adds the ability to keep in operation even if one or several services are down.
- Directory component as single point of failure, that can be mitigated by having different directories (opens the door for offering tailored service catalogues to distinct client profiles).

D. Role-playing a leader-follower architecture

The instructions given for role-playing a leader-follower architecture are shown in Table IV. Students played the dynamic evolution during operation of several systems. Expected outcomes were first-hand experience of:

- Independence of workers in terms of availability.
- Robustness: ability to stay operational even if some workers go down.
- Scalability: ability to elastically react to demand by starting more workers.
- Resource consumption optimization: ability to elastically react to demand by stopping idle workers.
- Leader as single point of failure.

Authorized licensed use limited to: University of Canberra. Downloaded on March 17,2025 at 03:38:26 UTC from IEEE Xplore. Restrictions apply.

TABLE I
ROLES TO PLAY IN A LAYERED ARCHITECTURE[a]

Role: layers	Instructions
You are the input layer	Receive requests from the client (teacher) and pass them along after processing. Do not take another request until you have received the reply to the last. Give the reply back to the client.
You are an intermediate layer	Take requests only from your previous, pass them along after processing to your next. Do not take another request until you have received the reply to the last. Give the reply back only to your previous.
You are the last layer	Take requests only from your previous, and process it. Do not take requests until you are done. Give the reply back only to your previous.

[a] Due to space limitations, these role character description cards are not as detailed as the ones students' received, but instead some abstracted, summarized versions.

TABLE II
ROLES TO PLAY IN A PIPELINE ARCHITECTURE

Role: filters	Instructions
You are the input filter	Receive requests from the client (teacher) and pass them along after processing. Take a new request as soon as you have passed along the previous.
You are an intermediate filter	Take requests only from your previous, pass them along after processing to your next. Take new requests only after passing along the previous.
You are the last filter	Take requests only from your previous, process and reply to the expected output.

TABLE III
ROLES TO PLAY IN A CLIENT-SERVER ARCHITECTURE

Role: clients	Instructions
You are a client	Your job is to submit requests for the services offered by the directory. You may submit one or many requests at a time, to the same or different services, whenever you want.
Role: directory	
You are the directory	Your job is to receive the requests from the clients and pass them along to one of the instances of the corresponding service. As soon as you have passed a request along, you can take the next one right away.
Role: services	
You are a service	Your job is to receive client requests forwarded by the directory and produce a reply for them. Do not take requests from clients directly. Do not take a new request until you are done with processing the previous.

TABLE IV
ROLES TO PLAY IN A LEADER-FOLLOWER

Role: clients	Instructions
You are a client	Your job is to submit requests to the system, via the leader. You may submit one or many requests at a time, whenever you want.
Role: leader	
You are the leader	Your job is to receive the requests from the clients and pass them along to one of the workers. You have to decide how to choose such worker (randomly, round-robin, etc). You may create a new worker if you find that all of them are busy. As soon as you have passed a request along, you can take the next one right away. You may stop a number of workers if you find that most of them are often idle. You may send the request to several workers. You may split the request into several workers.
Role: worker	
You are a worker	Your job is to process leader' requests. Do not take requests from anyone else. Do not take a new request until you are done with processing the previous.

E. Role-playing a peer-to-peer architecture

The instructions given for the last role-playing scenario are shown in Table V. Part of the students were made to form one network of peers, while the rest were to perform as clients. Expected experiences-as-outcomes were:

- Need to embed both management logic and business logic into each peer (as opposed to the leader-follower architecture, in which these responsibilities are assigned to distinct components).
- Need to manage life of requests, to ensure clients get some reply even when the network is under heavy load.
- Difficulties of implementing security measures to differentiate malicious from faulty or unavailable peers.
- Absence of coordination (impact on scalability, self-load balance).
- No single point of failure: maximum availability.

IV. DISCUSSION

Literature shows that multi-role projects are an effective teaching strategy [14]. It also reveals that role-playing affects three areas that support student learning and engagement, namely: identified personalized learning, deepened content understandings, and enhanced collaboration skills [11].

However, empirical evaluation also reports that while many students feel that the infusion of role-playing aspects into

Authorized licensed use limited to: University of Canberra. Downloaded on March 17,2025 at 03:38:26 UTC from IEEE Xplore. Restrictions apply.

TABLE V
ROLES TO PLAY IN A PEER-TO-PEER ARCHITECTURE

Role: clients	Instructions
You are a client	Your job is to submit requests to the system, using any of the peers. You may submit one or many requests at a time, whenever you want.
Role: peer	
You are a peer	Your job is to receive the requests (which may come from other peers or from clients) and either handle them and produce a reply, or pass them along to one or several of your neighbours. If you produce a reply, give it back to whoever sent it to you. As soon as you have processed/passed a request along, you can take the next one right away. Periodically, you need to revisit your neighbourhood, look for new peers, dismiss others.

the courses supported their learning and engagement, some other students do not [11]. This is coherent with the fact that certain teaching activities resonate better with certain learning profiles, and may also explain our quantitative evaluation.

In any case, we consider this does not pose a threat to the validity of our efforts, since our role-playing activities aim precisely to provide an advantage to those learning profiles that benefit the less from the rest of teaching methodologies already present. There are a couple of factors that may influence how impactful this is when evaluating an initiative such as ours in the context of a whole class or course:

- On the one hand, given that experimental learning experiences that address all learning styles are not commonplace, it makes sense to assume that students, especially by the time they reach the university level, have already adapted to the lack of methods and activities that best resonate with them, in the case of the less-often addressed learning profiles.
- On the other hand, given that we do not know how the distribution of different learning profiles amongst students in general, and neither amongst our CS students at UDC, it may also be the case their numbers are not significant when considered as part of the whole class.

V. CONCLUSIONS

In this paper, we have faced the challenge of extending the teaching methodologies that were being applied in an undergraduate SA course. The goal was to take into account the different learning profiles that students have. In doing so, we aim to provide a more fair learning environment, where every individual has opportunities to connect with the learning experiences in the most meaningful and effective way to them.

Given that the teaching methods that were in place were found to be miss-aligned when it came to students with *accommodating* or *diverging* learning styles (following Kolb's nomenclature), the way we have addressed this challenge has been by incorporating role-playing in an innovative way. We have designed a set of scenarios where students play the part of components in different architectural models. Such role-playing game provides concrete experiences of the structure, advantages, and disadvantages of different architectural models.

REFERENCES

[1] Andrianoff, S.K., Levine, D.B.: Role playing in an object-oriented world. SIGCSE Bulletin **34**(1), 121–125 (2002). https://doi.org/10.1145/563517.563386

[2] Capilla, R., Zimmermann, O., Carrillo, C., Astudillo, H.: Teaching students software architecture decision making. In: Jansen, A., Malavolta, I., Muccini, H., Ozkaya, I., Zimmermann, O. (eds.) Proceedings of the European Conference on Software Architecture. pp. 231–246. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-58923-3_16

[3] Ferrari, R., Madhavji, N.H., Wilding, M.: The impact of non-technical factors on software architecture. In: Proceedings of the ICSE Workshop on Leadership and Management in Software Architecture. pp. 32–36 (2009). https://doi.org/10.1109/LMSA.2009.5074862

[4] Galster, M., Angelov, S.: What makes teaching software architecture difficult? In: Proceedings of the International Conference on Software Engineering. p. 356–359. Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2889160.2889187

[5] Jones, J.S.: Participatory teaching methods in computer science. In: Proceedings of the SIGCSE Technical Symposium on Computer Science Education. p. 155–160. Association for Computing Machinery, New York, NY, USA (1987). https://doi.org/10.1145/31820.31751

[6] Kolb, D.A.: The Kolb learning style inventory. Hay Resources Direct Boston, MA (2007)

[7] Lago, P., van Vliet, H.: Teaching a course on software architecture. In: Proceedings of the Conference on Software Engineering Education and Training. pp. 35–42 (2005). https://doi.org/10.1109/CSEET.2005.33

[8] McLeod, S.A.: Kolb - learning styles and experiential learning cycle. Tech. rep., Simply Psychology (2017), `https://www.simplypsychology.org/learning-kolb.html`

[9] Montenegro, C.H., Astudillo, H., Gómez Álvarez, M.C.: Atam-rpg: A role-playing game to teach architecture trade-off analysis method (atam). In: Proceedings of the Latin American Computer Conference. pp. 1–9 (2017). https://doi.org/10.1109/CLEI.2017.8226416

[10] Ohlsson, L., Johansson, C.: A practice driven approach to software engineering education. IEEE Transactions on Education **38**(3), 291–295 (1995). https://doi.org/10.1109/13.406508

[11] Toth, D., Kayler, M.: Integrating role-playing games into computer science courses as a pedagogical tool. In: Proceedings of the ACM Technical Symposium on Computer Science Education. p. 386–391. Association for Computing Machinery, New York, NY, USA (2015). https://doi.org/10.1145/2676723.2677236

[12] Union, P.O.: Ects users' guide. `https://education.ec.europa.eu/document/ects-users-guide-2015` (2015)

[13] Van Deursen, A., Aniche, M., Aué, J., Slag, R., De Jong, M., Nederlof, A., Bouwers, E.: A collaborative approach to teaching software architecture. In: Proceedings of the ACM SIGCSE Technical Symposium on Computer Science Education. p. 591–596. Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3017680.3017737

[14] Warin, B., Talbi, O., Kolski, C., Hoogstoel, F.: Multi-role project (mrp): A new project-based learning method for stem. IEEE Transactions on Education **59**(2), 137–146 (2016). https://doi.org/10.1109/TE.2015.2462809

[15] Zowghi, D., Paryani, S.: Teaching requirements engineering through role playing: lessons learnt. In: Proceedings of the IEEE International Requirements Engineering Conference. pp. 233–241 (2003). https://doi.org/10.1109/ICRE.2003.1232754

[16] Zuppiroli, S., Ciancarini, P., Gabbrielli, M.: A role-playing game for a software engineering lab: Developing a product line. In: Proceedings of the IEEE Conference on Software Engineering Education and Training. pp. 13–22 (2012). https://doi.org/10.1109/CSEET.2012.39

Authorized licensed use limited to: University of Canberra. Downloaded on March 17,2025 at 03:38:26 UTC from IEEE Xplore. Restrictions apply.