Proceedings of the 52nd Hawaii International Conference on System Sciences | 2019

DecidArch: Playing Cards as Software Architects

Patricia Lago & Jia F. Cai
Vrije Universiteit Amsterdam
p.lago@vu.nl; j.f.cai@student.vu.nl

Remco C. de Boer
ArchiXL
rdeboer@archixl.nl

Philippe Kruchten
University of British Columbia, Canada
pbk@ece.ubc.ca

Roberto Verdecchia
Gran Sasso Science Institute & VU Amsterdam
roberto.verdecchia@gssi.it

Abstract

Teaching software architecture is a challenge because of the difficulty to expose students to actual meaningful design situations. Games can provide a useful illustration of the design decision making process, and teach students the power of team interaction for making sound decisions.

We introduce a game –DecidArch– developed to achieve three learning objectives: 1) create awareness about the rationale involved in design decision making, 2) enable appreciation of the reasoning behind candidate design decisions proposed by others, and 3) create awareness about interdependencies between design decisions.

The game has been played by 22 groups with a total of 83 players, all of them students of the VU software architecture course. We present some of the lessons learned, both from our observation and through participant survey. We conclude that the game well supports our three learning objectives, and we identify several improvement points for future game editions.

1. Introduction

Teaching *software architecture* to university students is particularly challenging as this is a topic that demands experience, technical competence, domain knowledge and software development skills - all characteristics that one acquires with time and practice rather than within the walls of a classroom.

Case studies, term-long projects, presentations of architectural designs by actual architects from industry, experiments with design tools, etc. can all contribute to some aspects of this learning. In this paper we present our attempt to devise a simple card game –DecidArch– to let players experience the concept of architectural design decision making.

DecidArch is not a computer game or a computer simulation. It is more akin to Monopoly or Risk, and can be played around a table in about an hour.

In this paper we describe our learning objectives around architecture decision making, the design of the game, and the validation experiment conducted.

2. Related Work

Over the past 15 years, there has been a lot of interest in games to support software engineering education, as testified by the series of ICSE workshops on the topic: GAS: Games and Software Engineering from 2011 onward e.g. [1]. In their extensive literature review [2], Kosa and colleagues show that the large majority (88%) of these games are digital, that is, they involve some rather encompassing simulation of a software development situation, and are mostly played by students against the computer (e.g. SimSE[1]).

To illustrate one situation or one concept, simpler board games or card games are often preferable, and there are numerous examples of successful ones by Taran at CMU [3] and the same group at UC Irvine [4]. "Bringing their own unique attributes, non-digital games may provide additional benefit. For instance, educators may utilize them, harnessing their social characteristics (e.g. face-to-face interaction) especially in teaching complex subjects that involve social aspects and that are hard to teach without simulating" [2]. Such games, where the students play against or with each other rather than against or with a computer, have been developed before, for instance, to illustrate architectural technical debt [5], to teach software project management [6], agile development [7], and process issues in software engineering [4]. To the best of our knowledge, there are two games that, similar to ours, target architecture design reasoning but have very different learning objectives: Schriek et al [8]

[1] https://goo.gl/3fVEey

URI: https://hdl.handle.net/10125/60220
ISBN: 978-0-9981331-2-6

specifically focus on design reasoning and their cards acts as triggers for the students to explore the design space by applying the reasoning techniques suggested by the selected card; Cervantes et al [9] focus on teaching the architecture design process by using the Attribute-Driven Design method [10]. While their cards are much more sophisticated than ours (including technologies, tactics and patterns), they are specific to big data systems. Also, the authors aim to lead the players to make "smart decisions". We rather do not enter the merit of the quality of the decisions themselves, but focus on making students appreciate the dynamics of architecture design, which entails collaboration and consensus, reasoning and among alternative options.

It must be noted that the literature already reports on courses designed to at least teach some aspects of such dynamics. Lago and van Vliet [11], for instance, focused on aspects of software architecture where communication is key, like defining views that frame stakeholder concerns, trade-off analysis and architecture assessment. De Boer et al. [12] emphasized collaboration and built a course that would let the students act as architects and experience the multiple and often conflicting variables influencing architecture design decisions. The game we present here, DecidArch, differs by zooming into one single course topic, namely *architecture design decision making*, and leveraging the examples cited above by emphasizing collaboration (the students play our game as teams of architects) and bringing together various skills and competences (by applying theory into practice).

3. Study Design

DecidArch is a game to be played by students in the context of our software architecture course. There, the students attend the traditional lectures explaining the theory related to architecture design decision making; in parallel they work in teams on a practical project proposed by one of our industrial partners. At the mid-point of their project, we ask them to play our game. In this way, they already have some practical experience in Software Architecture Design Decision Making (SADDM), so that they can transfer better what they learn during the game onto their own project.

Our study addressed the following main Research Question (RQ): *Can we teach SADDM through a card game?*. To answer this main research question, we defined four sub-questions, which we addressed in four subsequent study phases as illustrated in Fig. 3: Phases 1-3 focused on building the game, and Phase 4 on applying the game in our software architecture course. In particular:

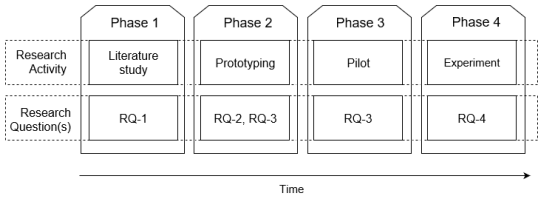

Figure 1. Study design: activities and research questions.

Phase-1 By studying the literature, we identified topics, a.k.a. knowledge elements [13], that are needed in SADDM and that we want to include in the game. The corresponding research sub-question is **RQ1** - What topics are included in the reasoning process when making architecture design decisions?

Phase-2 By using the results of Phase-1, we designed a first prototype of the game addressing two research sub-questions: **RQ2** - What are the target learning objectives? and **RQ3** - How will we design the card game? We used the literature and related works to identify our learning objectives.

Phase-3 We addressed **RQ3** by evaluating our game in a pilot where we asked practicing software architects to play the game prototype and to provide feedback on the learning objectives. We used these pilot results to improve the game and create a revised version of DecidArch.

Phase-4 Finally, we used the improved version of DecidArch to address our last research sub-question: **RQ4** - How do we evaluate the game on its ability to teach the learning objectives? We conducted an experiment with the Master students of the software architecture course at our university. As a result of Phase-4, we were able to draw conclusions on our main research question, RQ.

4. Study Execution

4.1. Phase-1: The Selected SADDM Topics

The literature provides a quite rich and mature body of knowledge in SADDM, which informed our answer to RQ1. As our game is targeting

novice software architects (i.e. university students) in time-bounded sessions, it must be simple and self-contained, yet cover the dynamics and key topics at the basis of design reasoning. Therefore, starting from the ISO/IEC/IEEE 40201:2011 standard [14] on *Software architecture representation*, and the seminal work of Babar et al. [15], we adopted the topics illustrated in Fig. 2. In particular:

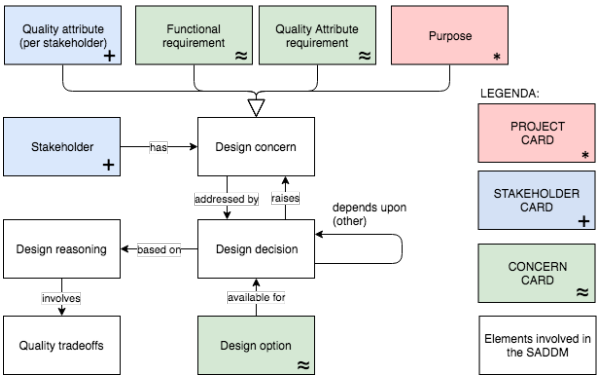

Figure 2. SADDM topics and their mapping on game cards

Purpose. As pointed out by Tang & van Vliet [16], a system has business goals, and a higher-level purpose that helps framing the context of the decisions by stating what the system is for. In [14], the system purpose is a key stakeholder concern. In our game, we mapped it on a specific type of card (Project Card) because it provides the overarching context for the whole game session.

Stakeholders. Depending on their specific concerns, each stakeholder can be concerned with a different list of Quality Attributes (QAs), or they may assign different priorities to different QAs.

Functional requirements. Among the various design concerns influencing the design decision making, we need to provide functional requirements to reason about the behavior of the designed system, and its quality. They "state what the system must do, and how it must behave or react to runtime stimuli" [17].

Quality Attribute requirements. A QA is a "measurable or testable property of a system that is used to indicate how well the system [will] satisfy the needs of its stakeholders" [17]. QA requirements, in turn, state the conditions that a system is expected to satisfy in terms of quality concerns [18,19].

Design options. Given a system stakeholders and design concerns, SADDM is the process of making design decisions and justifying them [16, 20]: each design decision is chosen among alternative available design options, as based on a certain rationale (or design reasoning), which involves quality trade-offs (as each design option offers a viable solution but yields different quality properties, i.e. strengths and weaknesses). The incremental nature of this process is captured in Fig. 2 by the cyclic relations between Design concern and Design decision: a concern is addressed by a certain decision, which when made, can raise new concerns. Further, by looking at software architecture as a set of design decisions [21], each decision also depends upon other decisions.

4.2. Phase-2: The Card Game Prototype

The literature study[2] carried out in Phase-1 allowed us to gain synthetic understanding of the key factors in SADDM. Based on such understanding, we formulated the game learning objectives and designed the prototype of the card game that realizes them. Similarly, we also included a gaming objective addressing the user experience (UX). Below we describe first the learning/gaming objectives and the game prototype, and then how the game elements address the objectives.

4.2.1. The Learning Objectives. We formulated the following target learning objectives that also provide our answer to RQ2:

- *LO1-Reasoning: Create awareness about the rationale involved in making trade-offs and choosing design options among alternatives.* Explicitly or implicitly, design decisions are chosen among alternative design options. This choice depends on the rationale behind the decision. The literature shows that design space exploration and making explicit the reasoning and rationale helps in making better-informed design decisions [20, 22]. As a valuable co-product, it also offers software architecture documentation for future change. Accordingly, the game should make the students aware of the topics playing a role

[2]We performed a simple literature review starting from pilot studies suggested by three experts followed by a round of snowballing.

in design reasoning, and that the underlying rationale involves making trade-offs when choosing design options among alternatives. This should improve their competence in design reasoning.

- *LO2-Differences: Enable the appreciation of the design decisions proposed by others.* Typically, SADDM is a collaborative effort. However, different architects may make different design decisions due to for example biases, differences in experience, technical competence or even design reasoning. Similarly, because of such differences, the students can have different ideas that result in different design decisions, too. If our game makes the students aware of these differences, and makes them appreciate that diversity contributes to a high-quality software architecture design [23], this should improve their competences in collaborating as a team by discussing their design ideas.
- *LO3-Reconsideration: Create awareness about the dynamics of software architecture design.* Design decisions may change and this may result in inconsistencies with other design decisions they depend upon. Therefore upon changing a decision, the architect should reconsider the other design decisions to check if inconsistencies have arisen, and resolve them. The later such inconsistencies are addressed, the more expensive is their resolution. If our game makes the students aware that changing design decisions may result in changes to other decisions, this should improve their competence in design reasoning.

4.2.2. The Gaming Objective. From a purely UX perspective, the game should be fun to play [24, 25] so that the players are engaged (essential for teamwork) yet challenged (i.e. the game should be easy to understand and easy to play so that the attention of the payers is fully on decision making). Also, they should be rewarded for the effort, i.e. be scored on the quality of the produced architecture design decisions. We called this objective *Playability*.

4.2.3. How the game works. We designed DecidArch with a deck of playing cards, which cover the topics identified in Section 4.1. Examples of playing cards types in the game are shown in Fig. 3.

The Project card describes the name and purpose

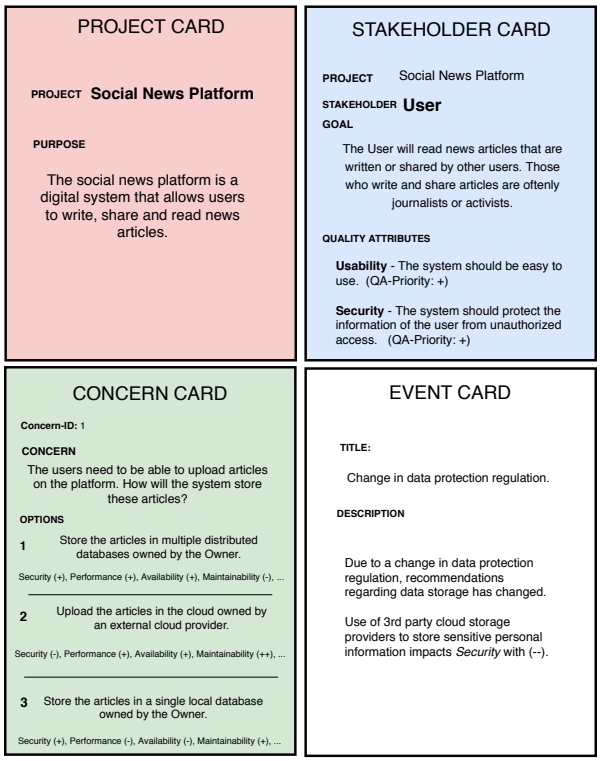

Figure 3. Examples of cards.

of the system whose software architecture is the game focus. This provides the decision making process its context for a complete game session.

A Stakeholder card describes the role of a certain stakeholder in terms of his or her role name, goal, and the list of QAs the stakeholder is concerned with. Each QA in the list is described by a short definition and the level of importance (or QA-priority) for the specific stakeholder.

A Concern card describes a concern that needs to be addressed with a design option. It can be a functional requirement or a QA requirement. The card also includes an initial list of alternative design options; each option is illustrated by a short description and an associated simplified representation of its *quality impact*, i.e. the list of QAs each with a logical value from "++", "+", to "−" and "−−", which, respectively, indicate if the design option would contribute to that QA very positively, positively, negatively or very negatively. In addition, a neutral value "0" stands for "no significant impact".

Concern cards are the key game design element: when one is drawn, the players must address that concern, and make a design decision by choosing one among the possible design options. Such a choice entails a line of reasoning: using the quality impacts of each alternative, therefore making trade-offs.

Event cards describe events especially designed to disrupt the current set of design decisions, and put the players in a new, changed situation where the design decisions made so far must be reconsidered and some of them possibly changed. Event cards are introduced to address LO3-Reconsideration.

The playing cards have been populated with the contents from the student projects proposed by our industrial partners in the software architecture Master course at the Vrije Universiteit Amsterdam, editions 2015 (9 project reports) and 2016 (14 project reports). For details about how the data has been extracted, we refer the reader to the online material [26].

Further, Fig. 4 shows how the game is played. In the *Preparation* phase (Fig. 4.(a)), the players sit at the table as a team of software architects that collaborate to design a software system architecture for a project and its stakeholders. The game session is set up by a Project card and the Stakeholder cards corresponding to that project[3]. In addition, the players prepare two shuffled stacks of cards (face down), one of Concern cards and one of Event cards.

Finally, each player gets a *Decision Preparation Template* where, for each design concern drawn during the game, he or she will incrementally record the design option s/he would personally choose to address that concern. Each team also gets a shared *Decision Taking Template* where, for each design concern, the design option decided upon by the team of architects is recorded. In the respective templates, the players (and the team) also briefly indicate the rationale for the proposed (resp. decided-upon) design decision.

As illustrated in Fig. 4.(b), playing the game consists of multiple rounds. For each round each player, in turn, (i) draws a Concern card; (ii) individually records his/her suggested option for that concern along with his/her rationale; (iii) followed by a team discussion of all suggestions, rationales and trade-offs; and (iv) a team decision recorded on the Decision Taking Template. At each turn the players need to consider during their SADDM the decisions made until that point. When a round is completed (i.e. all players drew a concern card), (v) an Event card is drawn: this card describes some change for the project. Depending on the change, the team has to collaboratively reconsider the addressed Concern cards to make sure that the change is properly covered. The changes due to the drawn Event cards last for the entire game.

The team of players carries out a number of rounds. The game finishes when either (i) all Concern cards have been addressed, or (ii) the allotted time limit (e.g. 30 minutes) is expired. In both cases, the players get a last chance to revise their set of design decisions, given the list of concerns and the list of events occurred during the game.

Each game has a reward: winning the game! In our case, we have been looking for a scoring mechanism that would be simple to calculate, and make DecidArch more fun by rewarding the players for doing well, and at the same time by encouraging them to produce the best possible design within the allotted time. The scoring elements we have been looking at, are: (i) address all concerns and events, yet (ii) satisfy all project stakeholders, and (iii) produce a set of design decisions with the best quality impact for the target Quality Attributes. The details of the scoring mechanism are provided in the online material [26].

4.2.4. How the Game addresses the Learning/Gaming Objectives. The elements of the game and the way it works are especially conceived to accommodate the target objectives, as explained in the following.

- In a game session, the Project card and Stakeholder cards provide the needed context for the players to carry out SADDM. The way game rounds are organized should facilitate key aspects of the reasoning process, hence **LO1-Reasoning**. In particular, the Concern cards with their list of design options prompt each player to consciously explore the design space, reason about the associated quality impacts and make QA trade-offs (that also consider the QA-priority in the Stakeholder cards). When players were not satisfied with the suggested design options on the card, they could also propose their own. This realizes the SADDM process illustrated by the white elements in Fig. 2. In addition, the use of the two templates help make the design decision

[3]The number of Stakeholder cards may vary depending on the drawn Project card.

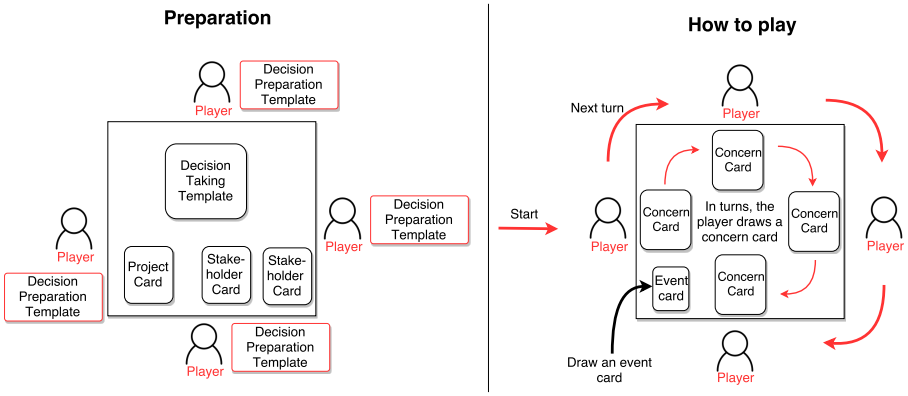

Figure 4. (a) Game preparation and (b) How to play the game.

and rationale explicit at both individual and team level.

- To address **LO2-Differences**, we separated SADDM in two steps: whenever a Concern card is drawn, first each player selects the design option s/he wants to propose and writes (in the individual Decision Preparation Template) his/her rationale; then all proposals are discussed in the team where the players can make their case for their own proposal but also appreciate the reasoning of the other players for alternative options. In this way, the collaborative nature of SADDM is enforced, as well as the appreciation of different perspectives and rationales. The use of a shared Decision Taking template also facilitate team building, hence helping the players overcome individual competitiveness.
- As change is a fact of life, effective design decision making *must* include continuous reconsideration of past design decisions. While the game already enforces changes due to the incremental iterative decisions embedded by multiple rounds and individual turns, these changes might not challenge past decisions enough. To ensure effectively addressing **LO3-Reconsideration**, Event cards introduce unforeseen changes placing players in disruptive situations where, e.g., decisions become invalid, or new concerns impact the whole design. Also, thanks to the templates (that enforce making decisions and rationales explicit), checking past decisions against new events becomes easier. This creates further awareness of the silent added-value of expliciting architectural knowledge.
- To address **Playability**, we designed the cards to be as self-explanatory as possible, and the game rules to be short and easy to grasp. Also, we devised a mechanism to score the decisions made by the team of players to balance simplicity (e.g. easy scoring to satisfy the stakeholders' needs and QA trade-offs) and significance (e.g. decisions' consistency and satisfaction of all stakeholders).

4.3. Phase-3: The Pilot

We piloted the game prototype with 7 software architects from the firm ArchiXL[4]. After a brief introduction to the game, the architects split in two teams and played. Thereafter, we carried out a discussion session in which we presented the learning objectives and gathered the architects' feedback on the extent to which the game addressed these objectives, and any additional comments. For research purposes, the whole pilot was video-recorded.

We collected valuable suggestions for improvement, which we used to revise the game. To address LO1-Reasoning, the participants suggested to *introduce the templates* (absent in the prototype) that would help making explicit the rationale behind a design decision and use it later to justify it in the team discussion. Not to overload the game dynamics, simplicity was mentioned as key. To keep rationale short and simple, the templates offer a bounded space to write it down.

LO2-Differences was considered insufficiently covered. To correct this weakness, the participants suggested to introduce *two different types of templates* that would put explicit emphasis on individual contributions (e.g. for players to voice

[4] www.archixl.nl/en

their own design ideas in spite of the perceived "time pressure") and teamwork.

Finally, in the discussion around LO3-Reconsideration the participants observed that the events foreseen in the prototype (representing incidents like a fire in a data center) had an insufficient impact on the players' past design decisions, hence not putting enough pressure for reconsideration. To address this comment, we introduced *new Event cards* especially engineered to either influence the quality impact of the design options (e.g. a new privacy law that forces a higher level of security), or change the quality demands of some stakeholders.

5. Experiment Results

5.1. Experiment context

The experiment has been conducted as part of our 2017-2018 Software Architecture master course. The target audience for this course is computer science and information science students. Participation in the experiment was compulsory for students enrolled in the course. The students received the game manual beforehand, and were asked to prepare by studying the game rules. At the start of the experiment, instructors briefly reiterated the rules. It was made clear that, while participation was mandatory, performance in the game would not impact the course grade. Instructors were present to clarify any questions that arose during the game.

A total of 83 master students, organized in 22 groups, played the game. When asked about their professional experience, 74 students indicated they possessed some experience in software engineering - either as software developer, software engineer and/or software architect - while 19 indicated they had none.

5.2. Learning Objective 1: Reasoning

Fig. 5 shows the responses to survey statements about Learning Objective 1: *create awareness in the students about the rationale involved in making trade-offs and choosing design options among alternatives*. The chart shows that, by and large, the game supports those aspects that we consider indicative for achieving this learning objective.

Almost all students indicated they reflected on how a design option could impact stakeholder concerns (statement S1.1), and that they had to consider quality attribute trade-offs (statement S1.2). To a somewhat lesser extent, but still in majority, students indicated they were motivated by the game's templates to document the reasons behind the chosen design options (statement S1.3) and that the quality impacts on the game's concern cards motivated them to explicitly reason about these trade-offs (statement S1.4).

Most noticeable are the relatively large number of students who indicated they felt neutral towards statement S1.3, as well as those who felt neutral towards or disagreeing with statement S1.4. Regarding the use of templates (statement S1.3), remarks from these students indicate that they would have preferred less writing and more interaction. Regarding the quality impacts (statement S1.4), students remarked that they did not agree with the impacts that were provided on the cards, that they "had their own beliefs", or already knew enough about the design options provided.

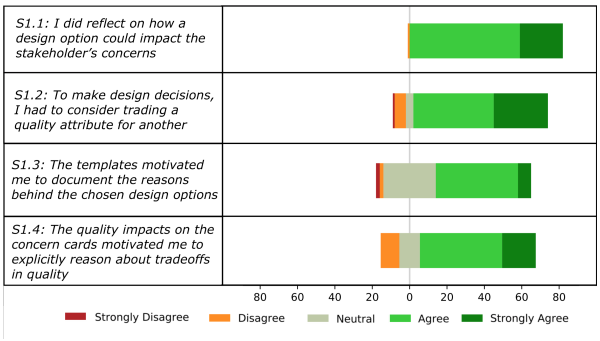

Figure 5. Diverging stacked bar chart for survey statements related to LO1-Reasoning

Is it interesting to note that the introduction of the templates had been prompted by the feedback from the pilot. Whether or not the use of templates to structure discussions is valued more by experienced architects than by novices is not something we've investigated further, but there appears to be a difference in this respect between the two player populations. Perhaps some further fine-tuning is needed to find a sweet spot between recording in-game design decisions and discussing them. It is well-known from knowledge management studies that 'recording what you know' (e.g. design rationale, code documentation, and knowledge in general) is not something people tend to do without a very clear reason to do so [27].

5.3. Learning Objective 2: Appreciating Other Decisions

Fig. 6 shows the responses to survey statements about Learning Objective 2: *enable the appreciation*

of the design decisions proposed by others. The overall impression from the diagram is that the game supports aspects relevant to LO2.

Almost all students indicated that, while playing the game, they discussed design ideas with the other players (statement S2.1). For most student teams, the differences between players in the team led to discussions about different design ideas (statement S2.2). However, about half of the students were neutral to or disagreeing with the statement that the decision preparation template gave an opportunity to contribute new design ideas (statement S2.3).

The comments that students left for statement S2.3 point to three reasons for the lower scores: (1) the templates are good for traceability, but not so much for creativity (for which direct discussion was deemed more suitable, cf. LO1); (2) the use of the template is costly (write down your own idea, discuss, write down the consensus); and (3) some teams did not feel the need to introduce any new design ideas, since the options provided on the cards were already sufficient.

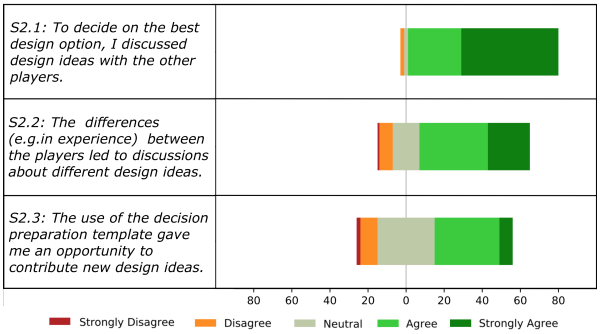

Figure 6. Diverging stacked bar chart for survey statements related to LO2-Difference

The idea behind the decision preparation template had been that individual team members could, in relative quiet, first think about their own position and arguments before 'defending' that position in a group discussion. Apparently, not all players did (or needed) that.

5.4. Learning Objective 3: Awareness of the Dynamics of Design

Fig. 7 shows the responses to survey statements about Learning Objective 3: *create awareness about the dynamics of software architecture design.* The diagram shows that the game's support for aspects related to LO3 is not as high as for the other two learning objectives. The root cause can probably be found in the responses to statement S3.3: "During the game, we had to change past design decisions". Responses to this statement were predominantly negative. The dynamics we intended to happen in the game did apparently not always occur.

Students remarked that they 'got lucky', '(by accident) did everything right', that 'event cards did not have an impact', that in the allotted time they 'reached only one event card', or 'none at all', even that they 'didn't know [changing decisions] was possible'. Some others, however, had to change 'two out of the four decisions we made before' and others even had to 'change completely all design decisions.

We conclude that the support for LO3 is present in the game, but depends too much on chance. In a next iteration of the game, the impact and frequency of events should be reconsidered. In short, players should run into enough events with enough (potential) impact so that the chance they have to revisit earlier design decisions increases.

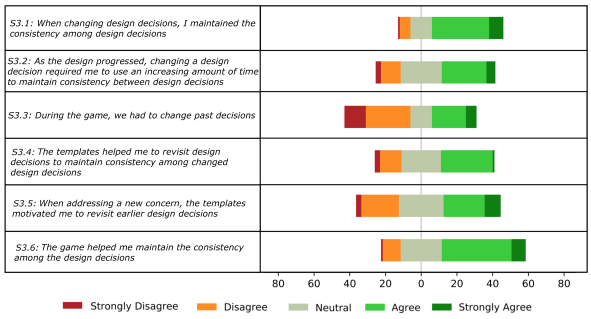

Figure 7. Diverging stacked bar chart for survey statements related to LO3-Reconsideration

5.5. Gaming Objective: Playability

Fig. 8 shows the responses to survey statements about the playability of the game. The majority of students who played the game agree that the game was fun (statement S4.1), that the cards were easy to understand (statement S4.2), the rules were clear (statement S4.3) and the scoring sheet was clear (statement S4.4). Overall, playability of the game is sufficient although some remarks indicate that the scoring sheet could be improved. An additional suggestion was to explicitly incorporate the 'consistency'-aspect (cf. LO3) in the scoring.

6. Threats to Validity

In this section we present potential threats to validity of our study results and our strategies to mitigate them.

Internal Validity refers to the ability to draw

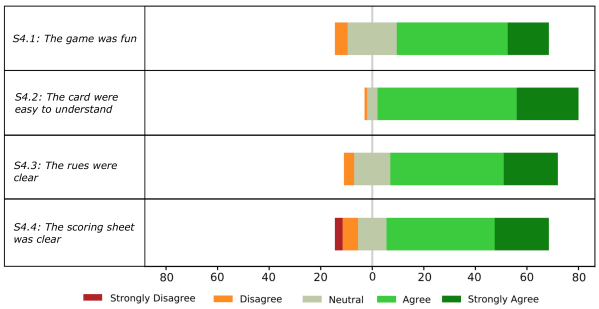

Figure 8. Diverging stacked bar chart for survey statements related to Playability

correct conclusions from the experiment and the collected data. To avoid threats to internal validity we designed *a priori* a survey. The survey covered the envisioned learning objectives through a list of statements (reported in Figs. 5-7). To avoid biases introduced by the adoption of the Likert scale to evaluate the statements, each statement was accompanied by an open question, in which participants could add additional information on their scaled responses.

An additional threat to internal validity is resulting from the background of the respondents. As they are students enrolled in a university course, the answers could be biased due to their concerns regarding potential relations between their answers and their academic performance. To mitigate this threat, before the survey was filled in, students were assured that their answers would not affect in any way their course grade and it would be purely used for research purposes.

External Validity refers to the ability to generalize the results of the experiment. In our case, we want to generalize the findings to students learning software architecture. This threat, therefore, mainly relates to the possible homogeneity of the respondents (all students of the Software Architecture course at our university). However, this threat is mitigated by the multidisciplinary and multicultural nature of our student population: about 50% is international, and the students were attending three different programs (CS, SE and IS) and eight different study tracks, ranging from *Software Engineering and Green IT* and *Business Information Systems* to *High Performance Computing* and *Theoretical Computer Science*. The students also differed in terms of professional experience, as reported in Section 5.1.

Construct validity refers to potential threats related to the extent to which the data collected is suited to answer our research questions. A potential threat to construct validity lies in the data source we adopted. In fact, it is crucial that the survey respondents are representative of the typical card game users. The demographics of the survey were all students of a university course focusing on Software Architecture. As the teaching objective is specific to this topic, we deem the vast majority of respondents as being interested in the topic and hence representative players.

An additional threat to construct validity is constituted by the environment in which the game was conducted. In fact, the game was carried out under the supervision of instructors who explained the game and intervened if clarifications were asked for. This could potentially lead to a different interaction between players playing with and without supervision.

Conclusion validity refers to the ability to draw correct conclusions from analyzing the gathered data. A first threat to conclusion validity lies in the statements adopted to evaluate our learning objectives. In order to mitigate this threat, we defined the survey *a priory*. The survey was then jointly discussed by all researchers. This was required in order to ensure that the survey covered all aspects of the game that were needed to comprehensively answer the research questions.

Another potential threat lies in the quality of the answers provided by the survey respondents. To mitigate this threat, each Likert scale response was accompanied with an open answer. Regarding further threats resulting from potential biased interpretations of the survey results, the collected data was analyzed by all authors. When necessary, results were jointly discusses in order to align the interpretation of the data and provide an objective analysis.

7. Conclusions

DecidArch is a board game that is inexpensive and easy to introduce in the classroom to teach

university students about the concept of software architecture design decision making: examining tradeoffs and compromises among stakeholders' demands, and major quality attributes, in the face of some moderate uncertainty.

The answer to our main research question: *Can we teach SADDM through a card game?* is a: *yes*. DecidArch well supports our three learning objectives, although the support for learning objective 3 (Reconsideration) currently depends too much on chance. Besides addressing this aspect through higher impact and frequency of the event cards, a future improvement we envisage for the game is a more balanced use of the decision preparation templates. These changes do not change the nature nor the design of the game, and are only tweaking the game's behavioral parameters.

The game DecidArch is available along with the experiment replication data [26].

References

[1] *Proc. 5th International Workshop on Games and Software Engineering (GAS)*, ACM, 2016.

[2] M. Kosa, M. Yilmaz, R. V. O'Connor, and P. M. Clarke, "Software engineering education and games: A systematic literature review," *Journal of Universal Computer Science*, vol. 22, no. 12, pp. 1558–1574, 2016.

[3] G. Taran, "Using games in software engineering education to teach risk management," in *20th Conference on Software Engineering Education Training*, pp. 211–220, 2007.

[4] A. Baker, E. Oh Navarro, and A. van der Hoek, "An experimental card game for teaching software engineering processes," *Journal of Systems and Software*, vol. 75, no. 1-2, pp. 3–16, 2005.

[5] T. Grant, "Dice of debt game." www.agilealliance.org/dice-of-debt-game. Accessed on: 2018-03-13.

[6] P. Kruchten and J. King, "Mission to Mars: An agile release planning game," in *24th Conference on Software Engineering Education and Training (CSEET)*, pp. 552–552, 2011.

[7] Agile42, "Kanban pizza game." www.agile42.com/en/training/kanban-pizza-game. Accessed: 2018-03-13.

[8] C. Schriek, J.-M. E. van der Werf, A. Tang, and F. Bex, "Software architecture design reasoning: A card game to help novice designers," in *Software Architecture*, LNCS, pp. 22–38, Springer, 2016.

[9] H. Cervantes, S. Haziyev, O. Hrytsay, and R. Kazman, "Smart decisions: An architectural design game," in *Proceedings of the 38th International Conference on Software Engineering Companion (SEET Track)*, ICSE, pp. 327–335, ACM, 2016.

[10] R. Wojcik, F. Bachmann, L. Bass, P. Clements, P. Merson, R. Nord, and B. Wood, "Attribute-driven design (ADD), version 2.0," tech. rep., 2006. CMU/SEI-2006-TR-023.

[11] P. Lago and H. van Vliet, "Teaching a course on software architecture," in *18th Conference on Software Engineering Education Training (CSEET)*, pp. 35–42, 2005.

[12] R. de Boer, R. Farenhorst, and H. van Vliet, "A community of learners approach to software architecture education," in *Conf. on Software Engineering Education & Training*, 2009.

[13] M. A. Babar and P. Lago, "Design decisions and design rationale in software architecture," *Journal of Systems and Software*, vol. 82, no. 8, pp. 1195–1197, 2009.

[14] International Organization for Standardization (ISO), "Systems and software engineering – Architecture description," standard, 2011. ISO/IEC/IEEE 42010:2011.

[15] M. A. Babar, T. Dingsoyr, P. Lago, and H. van Vliet, eds., *Software Architecture Knowledge Management - Theory and Practice*. Springer, 2009.

[16] A. Tang and H. van Vliet, "Software architecture design reasoning," in *Software Architecture Knowledge Management: Theory and Practice*, pp. 155–174, Springer, 2009.

[17] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*. SEI Series in Software Engineering, Pearson Education, Inc., 3 ed., 2012.

[18] E. Hochmuller, "Towards the proper integration of extra-functional requirements," *Australasian Journal of Information Systems*, vol. 6, no. 2, 1999.

[19] L. Chung and J. C. S. do Prado Leite, "On non-functional requirements in software engineering," in *Conceptual Modeling: Foundations and Applications*, pp. 363–379, Springer, 2009.

[20] M. Razavian, A. Tang, R. Capilla, and P. Lago, "In two minds: how reflections influence software design thinking," *Journal of Software: Evolution and Process*, vol. 28, no. 6, pp. 394–426, 2016.

[21] A. Jansen and J. Bosch, "Software architecture as a set of architectural design decisions," in *Working IEEE/IFIP Conf. on Software Architecture*, 2005.

[22] A. Tang and M. F. Lau, "Software architecture review by association," *Journal of systems and software*, vol. 88, no. 2, pp. 87–101, 2014.

[23] M. Razavian and P. Lago, "Feminine expertise in architecting teams," *IEEE Software*, vol. 33, no. 4, pp. 64–71, 2015.

[24] A. Iosup and D. Epema, "An experience report on using gamification in technical higher education," in *Technical Symposium on Computer Science Education*, SIGCSE, pp. 27–32, ACM, 2014.

[25] M.-B. Ibáñez, A. Di-Serio, and C. Delgado-Kloos, "Gamification for engaging computer science students in learning activities: A case study," *IEEE Transactions on Learning Technologies*, vol. 7, no. 3, pp. 291–301, 2014.

[26] J. F. Cai, P. Lago, R. C. de Boer, P. Kruchten, and R. Verdecchia, "DecidArch Replication Package," https://github.com/DecidArch/HICCS2018.

[27] J. F. Hoorn, R. Farenhorst, P. Lago, and H. van Vliet, "The lonesome architect," *The Journal of systems and software*, vol. 84, no. 9, 2011.