Journal of Engineering Sciences and Information Technology
Volume (4), Issue (4): 30 Dec 2020
P: 92 - 104

ISSN: 2522-3321

مجلة العلوم الهندسية وتكنولوجيا المعلومات
المجلد (4)، العدد (4): 30 ديسمبر 2020م
ص: 92 - 104

# Teaching Software Architecture Patterns Using ACME Language

**Tahani Elfatih Babeker**
Faculty of Computer Science and Information Technology || Sudan University for Science and Technology || Sudan
**Hany Ammar**
Lane Department of Computer Science and Electrical Engineering || West Virginia University || USA

**Abstract:** Software Architecture is one of the most important courses, in computer science discipline. It has many branches all of them aimed to prepare students to be architects on the industry. But actually, there is a gap between what the students find on the theoretical courses and what they find on the industry. On other words, the practical experience differs from academic theory. So the question is how to prepare students to join the industry?

Abstract nature of the software engineering courses as general and software architecture in a special manner, led to difficulties in understanding, this raises the second question, how to make these courses understandable?

All previous studies focusing on these problems either by changing course curricula or by using software tools.

This paper extension for the previous study [1] as we survey Architecture Description Languages (ADLs) and conclude that ACME is a general-purpose language and it may be suitable for use as a practical part for software architecture curricula. We aimed to design a framework use, ACME language, use it as practical part of the software architecture course and supporting on teaching, focus on architecture patterns, thus we use most common architecture patterns layer and Pipes-Filters, starting with a simple example and increase the complexity.

**Keywords:** Teaching Software Architecture، ACME Architecture Description Language، Software Architecture Education، Software Architecture Patterns

# تدريس أنماط معمارية البرمجيات باستخدام لغة أكمي

**تهاني الفاتح بابكر**
كلية علوم الحاسوب وتقانة المعلومات || جامعة السودان للعلوم والتكنولوجيا || السودان
**هاني عمار**
قسم علوم الحاسوب والهندسة الكهربائية || جامعة وست فرجينيا || أمريكا

**الملخص:** معمارية البرمجيات من المقررات المعتمدة في علوم الحاسب ولها العديد من الفروع والتي تهدف جميعها إلى تجهيز طالب علوم الحاسب للمشاركة في سوق العمل ولكن في الحقيقة هناك فرق بين ما يدرسه الطالب في الجانب النظري وما يجده عمليا في سوق العمل أو بصورة أخرى هناك فرق بين الدراسة الأكاديمية والخبرة العملية. إذن السؤال هو كيف يتم تجهيز الطالب حتى ينخرط في سوق العمل بدون معوقات؟

الطبيعة التجريدية لمقرري هندسة البرمجيات ومعمارية البرمجيات أدت إلى صعوبة في استيعابهم ومن هنا ظهر السؤال الثاني كيف يمكن أن نجعل تلك المقررات سهلة الفهم والاستيعاب؟

تعتبر هذه الورقة امتدادا للدراسة السابقة [1] والتي خلصنا فيها أن لغة أكمي تعتبر من اللغات ذات الأغراض العامة والتي يمكن أن تكون لغة مناسبة تستخدم كجزء عملي لمقرر معمارية البرمجيات.

قمنا بوضع إطار عمل فيه تستخدم لغة أكمي للمساعدة والدعم في تدريس الجزئية الخاصة بأنماط معمارية البرمجيات تم استخدامها في شرح نمطى الطبقات Layered والأنابيب و المرشحات Pipes- Filters ، ويعتبر هذين النمطين من أكثر أنماط معمارية البرمجيات استخداما. بدأً بمثال بسيط ومن ثم زيادة في تعقيد المثال بزيادة المتطلبات التي يجب إضافتها عل النمط المعين.

**الكلمات المفتاحية:** معمارية البرمجيات، لغة أكمي لتوصيف معمارية البرمجيات، تعليم معمارية البرمجيات، أنماط معمارية البرمجيات.

## 1. Introduction

Now days there are great depending on software in many aspects of the life, revolution of hardware and software is very clear, on the day activities of the human.

Increasing of the enterprises and its complexity causing focusing on software architecture and develop, so as to meet this demand, software engineering and software architecture get more attention to improve enterprises productivity and reduce its maintenance.

The challenge to cover all these requirements, is, how to prepare good architects? because actually increasing software enterprises not mean increasing graduates employment on the software architecture discipline, why? because graduates students does not meet industry demand. This causes due to main two reasons, the first one is the gap between academia and industry, and the second is the abstract nature of the software engineering and software architecture courses. There are many efforts to fix this problem, by developing new tools using to clarify and assist software architecture understanding and teaching or by using programming languages or curriculum reform.

Some universities reform and construct software architecture courses. New approaches and tools developed as helping tools in teaching software engineering and software architecture courses, all of them intended to build practical students and setup them to plug and play in the industry, more over give students more practice and experience in software architecture as general, they may have to do it in real systems from their environment to understand its requirements.

This paper focused in teaching software architecture as general and teaching software architecture patterns as special, and defining teaching by example approach, using real example as practical to assist students understanding and give them more practice, and teaching process be more attractive.

Teaching by example method, taught or educate students, what patterns is used and how to apply to certain project and evaluate. Explain this method using most common architecture patterns and ACME architecture description language.

The rest of this paper is organized as follows: In section 2, we present software architecture patterns and its categories. And related works are presented in section 3. In section 4 discussing the motivations about using ACME language, layered, and Pipes_Filters patterns. In section 5, tow case studies are introduced, follow by conclusion and recommendation in section

## 2. Software Architecture Patterns

The software architecture patterns provide solutions to certain problem family and the rational about this solution, it is not complete solution it is as blue print or skeleton or guide.

patterns main aspects are context, solution and problem, context is a given system software, the problem may be requirements needed to include or constraint must consider or properties has to be define for system software.

For example if the targeted system software require information hiding, an encapsulation, modifiability, performance or portability architect may have focused in on module structure patterns, on other hand when system software provide services or run concurrently with other systems or services, them may follow component connector structure, and can follow allocation structure when decide to migrate to other version of software or hardware, or when delivery efficiency requirement needed[2].

### 2.1 software Patterns categories

Main categories are controlled decomposing component, distributed system and interactive systems.

#### 2.1.1 Controlled Decomposing Component

It is separation of concern and control, or we can said this pattern category demand grouping of components, examples of these pattern are layer, pipe-filter and blackboard.

#### 2.1.2 Distributed System

Handling stream of data examples of these pattern is Broker, pipe-filter, micro kernel and client and sever

#### 2.1.3 Interactive Systems

Examples of these pattern are Model-View-Controller(MVC) and Presentation-Abstraction-Control(PAC).

## 3. Related Works

There are many approaches in teaching software architecture, from table (1) we find that:

1- Focused in education aspects (collaborative approach, community of learners developing tools).

2- Focused in software architecture curricula (reforming courses).

3- Other merge education by industry long real project with experts collaborative.

**Table (1) Summary of Related Works and Their Approaches**

| Source | Year | Method/Approach |
|---|---|---|
| [3] | 2012 | curriculum re form |
| [4] | 2014 | curriculum re form |

| Source | Year | Method/Approach |
|---|---|---|
| [5] | 2016 | develop tool |
| [6] | 2016 | curriculum re form |
| [7] | 2017 | develop tool |
| [9] | 2018 | develop tool |
| [8] | 2000 | develop tool |
| [10] | 2005 | programming language |
| [11] | 2018 | curriculum re form |
| [12] | 2002 | curriculum re form |
| [13] | 2018 | curriculum re form |
| [14] | 2011 | develop application |
| [15] | 2013 | curriculum re form |
| [16] | 2007 | develop tool |
| [17] | 2010 | develop tool |
| [18] | 2004 | develop tool |
| [19] | 2019 | curriculum re form |
| [20] | 2019 | community of learners |
| [21] | 2009 | community of learners |
| [22] | 2012 | teaching by example |

[3] AND [4] states that follow IEEE/ACM standard curricula for computer science students, is not enough, students need more practice in local system software. Changing in computer science curricula to prepare motivated, applicable and experts in software architecture students. However, there are international system software can be customized, to help students on practice, not just localized because localization may affect students experience in globalization.

Whether multiple examples having different layouts and different levels of detail can help students to create better models and use elements of UML in the right way? If the students could learn from examples? how many details to include in the student's designs in a way to make them more confident and optimize their model? [5] answer about these questions, software architecture can be as searchable operation, search on UML repository using class, attributes and design metrics, students can construct and improve their design models.

Using examples help students in

- increased the quality of their solution.
- makes them more confident in their models.
- helpful for them in order to create and improve their designs.
- understand the relationships between classes, naming of class diagram elements, roles and structure of the class diagram.

[6] depend on semiotic triangle software architecture, software description and software practice the question is how to design the concepts?. Contrast between abstract or fuzziness of software architecture concepts and practice nature of design, led to many difficulties in teaching software architecture, so educator have to design practical assignments, students have working in teams in long live projects example. However, the variation in software architecture definition does not introduce students confusion. Also they suggest building a repository of software architecture teaching material, including architecture exemplars that can be used for teaching architecture.

From other perspective teaching software architecture using games and programming languages, [7] The main challenge in teaching software architecture to undergraduates students, is the lack of their experience with medium and large real system software, introducing students to work in real systems related to their society and environment. So design a web-based role playing game (RPG) to support teaching of ATAM (Architecture Trade-off Analysis Method) to informatics students. Chilean national tsunami warning System (SNAM). As result students are strongly motivated and interested, this model also allow students to use and practice in UML Modeling, BPMN Modeling and Scrum Project. Also JavaBeans provides an excellent conceptual model, component architecture and platform for teaching software reuse in an upper-level software engineering course [8] [9].

[10]. Balancing between course content and new technology is needed, by updating software architecture courses to meet new technologies to put students in right track, achieving this by introducing students to work in real, long and complex systems, in teams using new technologies and professional tools.

Also [11] [12] course design and focus in practical topics, in addition[12]

using queuing model in software architecture design course, to clarify dependences between architecture design dissention and quality attributes using example of three styles monolithic, split frontend and backend and extracted services (productivity and resource utilization) so architecture design decision be more clear and understandable to students. However, students need to be more familiar with queuing model.

Most of studies intended teaching software architecture describe SWA as abstract, evidence and need more practice [13], to achieve deep understanding Develop teaching method by join study part with practical carriers, need real project to practice and software problem related to students environment, it is web instructional system or web application and apply data structure and software architecture knowledge and concepts. this method impact in two sides teachers and students, on teaching side, rich teaching content update their knowledge, because when using industry teaching with teaching theory make teaching process valuable also increase their interaction on student side increasing their practice which affect their understanding, interesting and remove barriers between academia and industry by using professional technologies.

The gap between undergraduates students and industry led to refine computer science and technology education, [14] redesigning education schema by dividing the academic four years to three years studying theory and academics, and the final year for practice using trend technologies, increasing engineering ability and application-oriented education over academic theory knowledge

To evaluate the quality of the model architects using a tool called Architecture Expert (RchE) [15], so students can used it to evaluate quality of the architected models, focusing in performance and modifiability qualities. These tool assist students to understanding qualities and how to apply and the impact of following different tactics. From instructors respective this tool help students to be more practical and deeply understanding.

An innovated software architecture teaching approach using (VR, AR ) [16] technologies, so software architecture be more practical, attractive and enhance teaching and learning. In the approach software architecture elements (components, connectors, architecture patterns) are created and documented, using Augmented Reality to capture 2D architecture and using Virtual Reality to view, interact and simulate.

How can we use modeling tools in teaching software architecture? [17] answer this by discussing types of modeling tools, and how they are success in teaching software architecture, and recommending to increase these tools to be help full for the students. How to make software architecture abstract concepts easy to comprehend? To answer these [18] design and improve software architecture curriculum, design the course that is relevant to the learner's environment with the practical skills.

Using game to get learning objectives [19], same as community of learners [20], students discussing about which best design and represent rational however this game have three learning objectives reasoning differences Reconsideration which led to students satisfying and appreciated their design decision.

Focusing in teaching software architecture patterns by applying two architecture using real trading application and evaluate the impact of applying each pattern so as to choose best design solution [21].However, security is not one of the quality attributes which is evaluate, as it a trading application also the comparison between two pattern has to be with same quality attributes.

Also scrum method of agile approach using in teaching software architecture [22].

## 4. Motivations

As we survey Architecture Description Language [1], we conclude ACME language is the most general purpose and it can be using in teaching purpose. We used in our case studies layered, pipes-filters and client server architecture patterns so as to cover most common patterns, Select ACME architecture description language to demonstrate.

Although Architecture Description Languages (ADLs) lack in architecture styles, analysis and description interaction, we find ACME has some features it provides a) structural frame work build on the architecture elements (components, connectors, ports, roles), and b) it's easy to read, write and understand by human so it become popular more over its maturity in describing architecture[23] [24], c) has predefined template for common patterns, d) allow define constrains, e) introduce acme studio which is an eclipse plug in for association pattern elements with architecture elements.

However, ACME weak in types because it has fix list of variable data types.

architecture patterns to be more understandable we ensure that

- referring to the main aspect of patterns we explain the problem and each pattern.
- understand the context and problem(constraints) in simple manner.
- increasing the complexity by adding new constrains.
- evaluate the solution according to specific pattern.

## 5. Modeling Patterns in ACME

To evaluate the suitability of ACME for modeling architecture patterns, we have select tow architecture patterns layered and pipe-filter to applying in tow applications address book and flooding alert, starting with simple cases and increasing complexity to some extent.

### 5.1 Address Book Case Study:

System that allows users to search for addresses, write new address or update.

Present simple Address book start with the list of the contacts then add search criteria, then add create, edit or delete functions to the application finally adding nonfunctional attributes such as security so system protect information from unauthorized access, usability the system should be easy to use and performance so the system should be able to process and full fill the user's request fast. So we have three stages to do so.

1- The application to display address book when user enter the application the contact list will display, figer.1

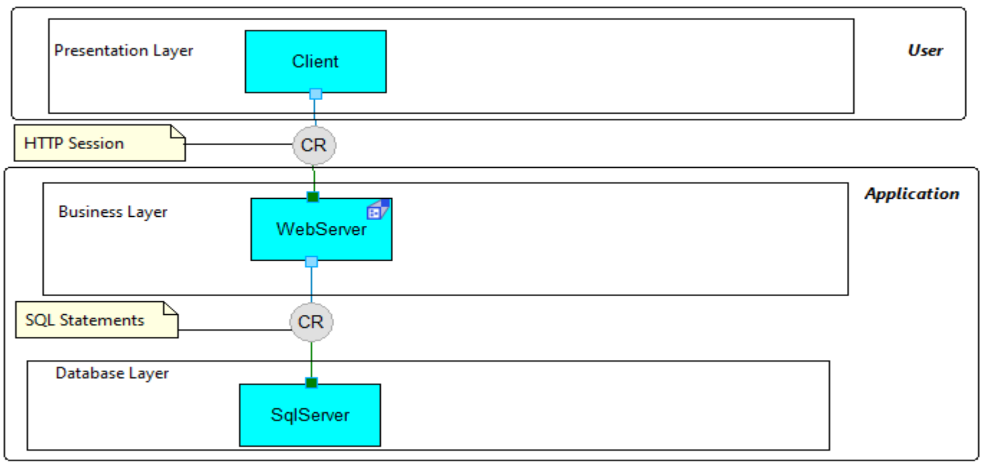

**Figure (1) Simple Address Book**

1- Application with search bottom, user enter search criteria name or telephone or address or combination of them, students have to add search bottom on GUI and change the SQL statement.
2- Application with search and update (delete, edit or create) adding delete, edit and create bottoms on GUI adding SQL statements for insert, update and delete, ADO or DAO components on business layer, figer.2

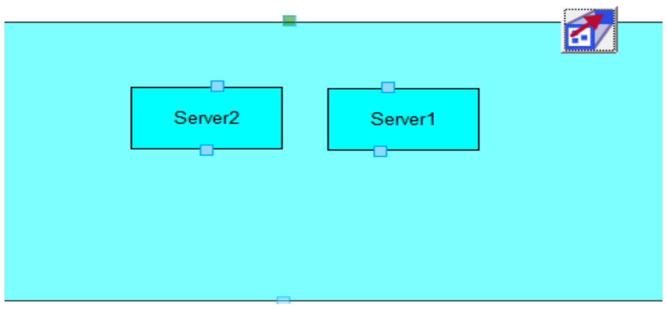

**Figure (2) Business Layer Structure**

Explaining this example gradually, with explaining the concepts of architecture patterns allow students to deeply understanding and be more creative more over architecture pattern be more tangible to some extent.

## 5.2 Flooding Alert System Case Study

As [3] mention, students have to architect system related to their environment, we find flooding alert system suitable, river Nile flooding is the yearly action, and most of the students have an idea about it, as previous example we start with simple application sensor and alarm, till alarming from expected amount of rains.

Description: To illustrate the Pipes-filters architecture pattern, we will study the example of a simplistic flooding system. We will assume that our platform has light alarm and three different location sensors; two of them that periodically sends data in integers for the current water level, a third sensor that raises an event in case of critical water level. Figer.3 illustrate our case study we will simulate the three sensors with a filters that will send update values at every dispatch, and will sometimes send the critical event. The first sensor will monitor the water level and raise warning message if certain conditions are met. And send it to the second sensor, which is also read its current water level, and sent it to the third one; if water level reached critical level send to signal to alarm led and then sends data to main server at every dispatch.

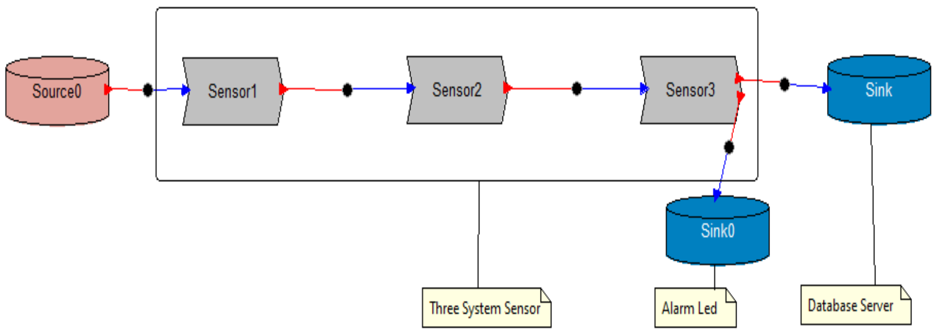

**Figure (3) Simple Flooding Alert System**

Then adding more complexity to the example; by adding new sensor to read the rain amount called rain sensor, when the rain amount reached critical level, sends it to the third sensor Figer.4.

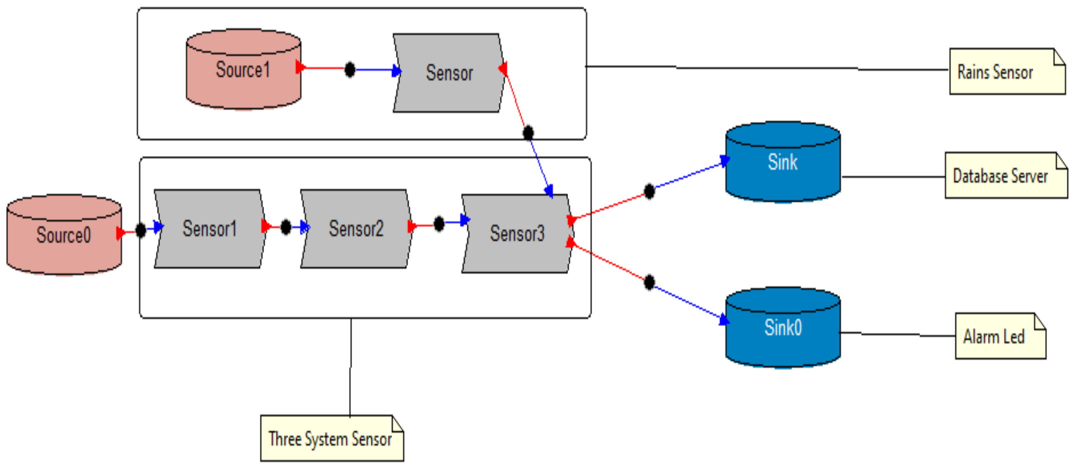

**Figure (4) Flooding Alert System with rains sensor**

## 6. Conclusion and Recommendation

In this paper we propose teaching by example model, to teach architecture pattern using Acme Studio editor for Acme language, illustrate case studies using two architecture patterns. Acme provide template for basic architecture elements components, connectors, system and configuration any violation of the constrains raise error or warning by acme studio, constraints are demonstrated as rules, there are two types of rules invariants and heuristic. Violation invariants rules raising errors and violation heuristic rules raising warnings.

Acme provide also template to express grouping of components which is suitable for designing layered architecture, also it's expressing filters and data flow, so it suitable for designing pipes-filters architecture pattern.

The overall affect about teaching by example is positive, so we intend to applying these issues by extending to other software architecture disciplines, and use Acme as lab work in software architecture course, also company training.

## References:

1- Erradi, "Applying and evaluating architectural patterns on a stock trading case study," 2012 Int. Conf. Inf. Technol. e-Services, ICITeS 2012, 2012.

2- W. Kamal and P. Avgeriou, "An Evaluation of ADLs on Modeling Patterns for Software Architecture," 4th Int. Work. Rapid Integr. Softw. Eng. Tech., 2007.

3- Karasneh, R. Jolak, and M. R. V. Chaudron, "Using examples for teaching software design: An experiment using a repository of UML class diagrams," Proc. - Asia-Pacific Softw. Eng. Conf. APSEC, vol. 2016-May, pp. 261–268, 2016.

4- Karasneh, R. Jolak, and M. R. V. Chaudron, "Using examples for teaching software design: An experiment using a repository of UML class diagrams," Proc. - Asia-Pacific Softw. Eng. Conf. APSEC, vol. 2016-May, pp. 261–268, 2016.

5- Zhang, B. Su, Y. Wang, and K. Liu, "Research of excellent software engineer education for application-oriented university," Proc. 8th Int. Conf. Comput. Sci. Educ. ICCSE 2013, no. June 2010, pp. 1107–1110, 2013.

6- Zhang, B. Su, Y. Wang, and K. Liu, "Research of excellent software engineer education for application-oriented university," Proc. 8th Int. Conf. Comput. Sci. Educ. ICCSE 2013, no. June 2010, pp. 1107–1110, 2013.

7- H. Montenegro, H. Astudillo, and M. C. G. Álvarez, "ATAM-RPG: A role-playing game to teach architecture trade-off analysis method (ATAM)," 2017 43rd Lat. Am. Comput. Conf. CLEI 2017, vol. 2017-Janua, pp. 1–9, 2017.

8- H. Montenegro, H. Astudillo, and M. C. G. Álvarez, "ATAM-RPG: A role-playing game to teach architecture trade-off analysis method (ATAM)," 2017 43rd Lat. Am. Comput. Conf. CLEI 2017, vol. 2017-Janua, pp. 1–9, 2017.

9- C. S. C. Rodrigues, "VisAr3D: An approach to software architecture teaching based on virtual and augmented reality, " Proc. - Int. Conf. Softw. Eng., vol. 2, pp. 351–352, 2010.

10- C. S. C. Rodrigues, "VisAr3D: An approach to software architecture teaching based on virtual and augmented reality," Proc. - Int. Conf. Softw. Eng., vol. 2, pp. 351–352, 2010.

11- Garlan and R. T. Monroe, "ACME: An Architecture Description Interchange Language Acme: An Architecture Description Interchange Language," 1997.

12- Simon, G. Vogel, and E. Plödereder, "Teaching Software Engineering with Ada 95," pp. 115–128, 2005.

13- D. Simon, G. Vogel, and E. Plödereder," Teaching Software Engineering with Ada 95," pp. 115–128, 2005.

14- Wedemann, "Scrum as a Method of Teaching Software Architecture," Proc. 3rd Eur. Conf. Softw. Eng. Educ. ZZZ - ECSEE'18, pp. 108–112, 2018.

15- J. D. McGregor, F. Bachman, L. Bass, P. Bianco, and M. Klein, "Using an architecture reasoning tool to teach software architecture," Softw. Eng. Educ. Conf. Proc., pp. 275–282, 2007.

16- J. D. McGregor, F. Bachman, L. Bass, P. Bianco, and M. Klein, "Using an architecture reasoning tool to teach software architecture," Softw. Eng. Educ. Conf. Proc., pp. 275–282, 2007.

17- J. Wang and Y.-A. Wang, "Teaching software reuse with JavaBeans," Front. Educ. Conf. 2000. FIE 2000. 30th Annu., vol. 1, p. T2C/7-T2C/8 vol.1, 2000.

18- J. Wang and Y.-A. Wang, "Teaching software reuse with JavaBeans," Front. Educ. Conf. 2000. FIE 2000. 30th Annu., vol. 1, p. T2C/7-T2C/8 vol.1, 2000.

19- K. Alfert and J. Schr, "Software Engineering Education Needs Adequate Modeling Tools £ 1 Introduction 2 Lightweight Modeling Tools," pp. 1–6, 2004.

20- K. Alfert and J. Schr, "Software Engineering Education Needs Adequate Modeling Tools £ 1 Introduction 2 Lightweight Modeling Tools," pp. 1–6, 2004.

21- L. Bass, P. Clements, and R. Kazman, "Software Architecture in Practice, Second Edition," Softw. Archit., p. 560, 2003.

22- L. Bass, P. Clements, and R. Kazman, "Software Architecture in Practice, Second Edition," Softw. Archit., p. 560, 2003.

23- M. Cao and Z. Cao, "Teaching data structures and software architecture while constructing curriculum platform, " ICCSE 2011 - 6th Int. Conf. Comput. Sci. Educ. Final Progr. Proc., no. Iccse, pp. 1433–1437, 2011.

24- M. Cao and Z. Cao, "Teaching data structures and software architecture while constructing curriculum platform, " ICCSE 2011 - 6th Int. Conf. Comput. Sci. Educ. Final Progr. Proc., no. Iccse, pp. 1433–1437, 2011.

25- M. Galster and S. Angelov, "What makes teaching software architecture difficult?" Proc. 38th Int. Conf. Softw. Eng. Companion - ICSE '16, pp. 356–359, 2016.

26- M. Galster and S. Angelov, "What makes teaching software architecture difficult?" Proc. 38th Int. Conf. Softw. Eng. Companion - ICSE '16, pp. 356–359, 2016.

27- O. E. Lieh and Y. Irawan, "Teaching adult learners on software architecture design skills," Proc. - Front. Educ. Conf. FIE, vol. 2018-Octob, pp. 1–9, 2019.

28- O. E. Lieh and Y. Irawan, "Teaching adult learners on software architecture design skills," Proc. - Front. Educ. Conf. FIE, vol. 2018-Octob, pp. 1–9, 2019.

29- R. C. D. Boer, R. Farenhorst, and H. Van Vliet, "A community of learners approach to software architecture education, " Proc. - 22nd Conf. Softw. Eng. Educ. Training, CSEET 2009, pp. 190–197, 2009.

30- R. C. D. Boer, R. Farenhorst, and H. Van Vliet, "A community of learners approach to software architecture education," Proc. - 22nd Conf. Softw. Eng. Educ. Training, CSEET 2009, pp. 190–197, 2009.

31- R. C. De Boer, P. Lago, R. Verdecchia, and P. Kruchten, "DecidArch V2: An Improved Game to Teach Architecture Design Decision Making," Proc. - 2019 IEEE Int. Conf. Softw. Archit. - Companion, ICSA-C 2019, pp. 153–157, 2019.

32- R. C. De Boer, P. Lago, R. Verdecchia, and P. Kruchten, "DecidArch V2: An Improved Game to Teach Architecture Design Decision Making," Proc. - 2019 IEEE Int. Conf. Softw. Archit. - Companion, ICSA-C 2019, pp. 153–157, 2019.

33- R. Osman and A. C. Dias-Neto, "Motivating by examples: An empirical study of teaching an introductory software engineering course in Brazil," Proc. - Int. Comput. Softw. Appl. Conf., pp. 245–250, 2014.

34- R. Osman and A. C. Dias-Neto, "Motivating by examples: An empirical study of teaching an introductory software engineering course in Brazil," Proc. - Int. Comput. Softw. Appl. Conf., pp. 245–250, 2014.

35- R. Osman, "Teaching software engineering in developing countries: A position paper," Proc. - Int. Comput. Softw. Appl. Conf., pp. 648–653, 2012.

36- R. Osman, "Teaching software engineering in developing countries: A position paper," Proc. - Int. Comput. Softw. Appl. Conf., pp. 648–653, 2012.

37- S. Heckman, K. T. Stolee, and C. Parnin, "10+ Years of Teaching Software Engineering with iTrust: The Good, the Bad, and the Ugly," Proc. 40th Int. Conf. Softw. Eng. Softw. Eng. Educ. Train. - ICSE-SEET '18, pp. 1–4, 2018.

38- S. Heckman, K. T. Stolee, and C. Parnin, "10+ Years of Teaching Software Engineering with iTrust: The Good, the Bad, and the Ugly," Proc. 40th Int. Conf. Softw. Eng. Softw. Eng. Educ. Train. - ICSE-SEET '18, pp. 1–4, 2018.

39- T. E. Babeker, "Architecture Description Language – Systematic Literature Review ركباب حتافلا ي ناهت, " vol. 55, pp. 55–68, 2019.

40- T. E. Babeker, "Architecture Description Language – Systematic Literature Review ركباب حتافلا ي ناهت, " vol. 55, pp. 55–68, 2019.

41- Liubchenko, "Queueing modeling in the course in software architecture design," 2018 IEEE 13th Int. Sci. Tech. Conf. Comput. Sci. Inf. Technol. CSIT 2018 - Proc., vol. 1, pp. 219–222, 2018.

42- Liubchenko, "Queueing modeling in the course in software architecture design," 2018 IEEE 13th Int. Sci. Tech. Conf. Comput. Sci. Inf. Technol. CSIT 2018 - Proc., vol. 1, pp. 219–222, 2018.

43- V. Kryuk, A. V. Pilyugin, A. A. Povzner, and I. N. Sachkov, "Features of formation of resistive states with abnormally small temperature coefficients of electrical resistance in the heterogeneous systems FeSi - FeSi2," Inzhenerno-Fizicheskii Zhurnal, vol. 75, no. 3, pp. 171–174, 2002.

44- V. V. Kryuk, A. V. Pilyugin, A. A. Povzner, and I. N. Sachkov, "Features of formation of resistive states with abnormally small temperature coefficients of electrical resistance in the heterogeneous systems FeSi - FeSi2," Inzhenerno- Fizicheskii Zhurnal, vol. 75, no. 3, pp. 171–174, 2002.