Building Customer Capacity Through Organizational Patterns Improves the Development Team’s Understanding

Mohammad Ismail Khattab and Valentino Vranić
mohammad.khattab@stuba.sk, vranic@stuba.sk
Institute of Informatics, Information Systems and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava

The development team needs to work together with a customer who has a basic understanding of information technology. Customers need to be aware of the business processes and the importance of technology. A customer and the development team need to communicate and understand each other in the software project’s development. Software projects can be successfully built when the customer has sufficient capacity. The process of building an organization’s strength and sustainability is called capacity-building. Software projects are more effective and quality when capacity development takes place at the organizational level. Capacity directly affects work performance and business outcomes. Due to low customer capacity, we have found in practice and in the literature that *False Requirements*, *No Basic Technology Knowledge*, *No Domain Knowledge*, *Mismanaged Timeline*, *No Trust in Software Project*, and *Unproductive Discussion* are the main customer capacity challenges to the software project development team. We documented our experiences and five newly discovered patterns in government organizations to solve the mentioned challenges. We present this documented experience in the form of patterns in this paper to improve the customer capacity Before work begins on a software project or during the software development process. Some organizational patterns that reduce the mentioned challenges are also used along with newly discovered patterns. We have illustrated the sequences of the patterns and explained how we could make a pattern language from these patterns.

Categories and Subject Descriptors: [**Software and its engineering**] Patterns

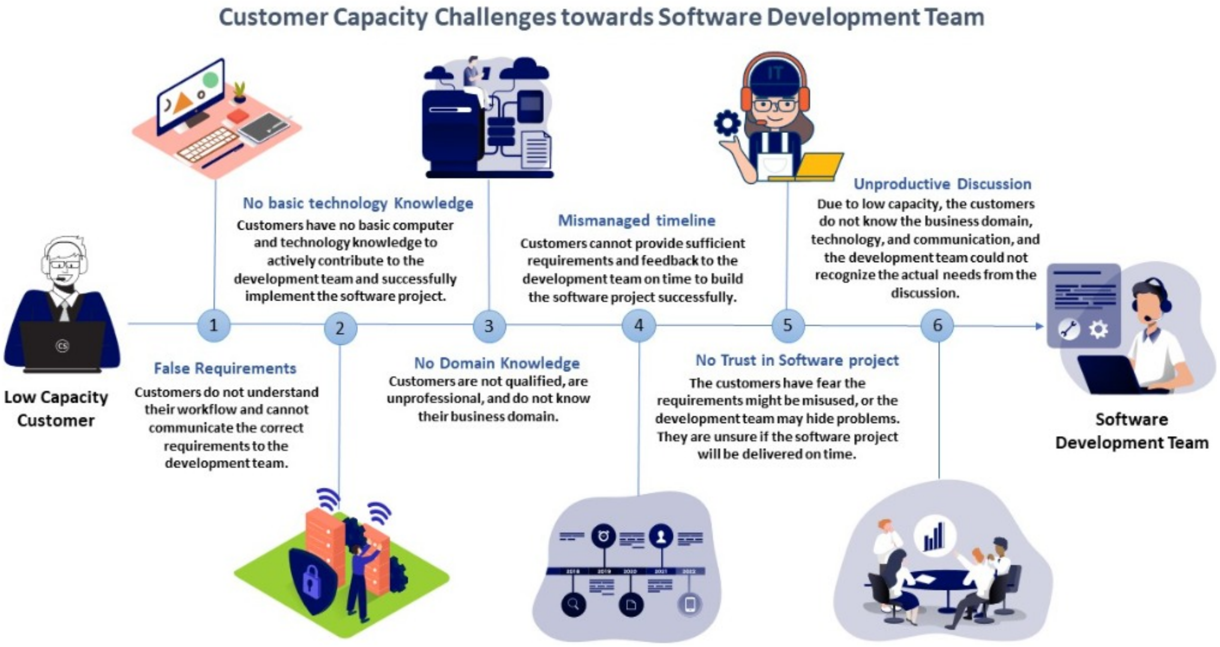
General Terms: Patterns
Additional Key Words and Phrases: Patterns, customer capacity building, organizational patterns, software projects, development team, customer

ACM Reference Format:

Khattab, M. I. and Vrani´c, V. 2022. Building Customer Capacity Through Organizational Patterns Improves the Development Team’s Understanding. HILLSIDE Proc. of Conf. on Pattern Lang. of Prog. 29 (October 2022), 12 pages.

1. INTRODUCTION

There is widespread recognition, both formally and informally, that capacity building considers a range of dimensions, from the knowledge and expertise of individuals to the capabilities of organizations. An organization with capacity is able to perform tasks effectively, efficiently, and sustainably [17]. Human resources play a crucial role in an organization, but developing them must be a priority regardless of their previous experience. The development team should communicate with a qualified customer during software project development [19]. There is a need for a customer who has a basic understanding of information technology and should work with the development team. It is not necessarily to train the customer to be an ICT master but to educate them to be qualified customers who can competently contribute to software project development.


Customer Capacity Challenges towards Software Development Team
Low Capacity Customer
Software Development Team
1
2
3
4
5
6

No basic technology Knowledge
Customers have no basic computer and technology knowledge to actively contribute to the development team and successfully implement the software project.

False Requirements
Customers do not understand their workflow and cannot communicate the correct requirements to the development team.

No Domain Knowledge
Customers are not qualified, are unprofessional, and do not know their business domain.

Mismanaged timeline
Customers cannot provide sufficient requirements and feedback to the development team on time to build the software project successfully.

No Trust in Software project
The customers have fear the requirements might be misused, or the development team may hide problems. They are unsure if the software project will be delivered on time.

Unproductive Discussion
Due to low capacity, the customers do not know the business domain, technology, and communication, and the development team could not recognize the actual needs from the discussion.

Fig. 1. Problem domain.

Training of employees or customers effectively improves the work performance [16]. Before work on a software project begins, the development team should thoroughly understand customers' level of profession and expertise. The development team needs customer knowledge and experience in the business domain. There are some government organizations where the customer capacity is very low, and they can't sufficiently describe the administrative requirements and provide them to the development team. However, this does not mean that the customers have no idea what they are talking about, but they have limited capacity in many situations. Customers with low capacity may not be able to articulate their requirements clearly to a development team and may not be able to explain their workflow in a way that makes sense to a team separated by language and culture. Such customers may not provide sufficient requirements and feedback to the development team on time to build the software project successfully. The development team develops the software project based on false requirements, as a result, the software project is no longer applicable in the organization.

In some developing countries, such as Afghanistan, there are customers who have no basic computer and technology knowledge to actively contribute to the development team and successfully implement the software project (see Figure 1). The development team visits the customers in their environment, but often these visits are not productive due to the low customer capacity and miscommunications. Miscommunication causes the software project to fall behind schedule. Low-capacity customers have fear the requirements might be misused, or the development team may hide problems [18]. They are unsure if the software project will be delivered on time. Due to low capacity, the customers do not know the business domain, technology, and communication, and the development team could not recognize the actual needs from the discussion. The low-capacity customers do not understand the technical terminologies used by the development team.

Sometimes, problem definition in the real world is difficult and very complex [6]. It is also possible for the development team's management to decide together with the customer how to build the customer's capacity before developing the software project. The team's management may explain the software's benefits and values during training to the customers.

The development team should share good and bad news throughout the project with customers to build trust [18]. On the other hand, the development team must consider the customer's suggestions and ideas to complete the software project. The development team needs to build customer capacity and communicate closely with a customer with sufficient expertise. Capacity building is improving a workforce's knowledge, attitudes, and skills to achieve short- and long-term organizational and personal goals [15]. Capacity building addresses all end-user inabilities while also developing the desired skills and attitudes to efficiently enable them to do acceptable activities [29]. Capacity building positively impacts end-user performance, and external factors are also considered [1]. In general, capacity development improves the effectiveness and quality of the software project at the organizational level. There is a need for customer training, and potential behavioral change focused not only on the use of new technologies and approaches but also on attitudes toward contracting and inclusion of customers and developers in project teams [14]. Initial results indicate that customers who receive training in requirements elicitation produce more results, measured in function points than those who do not receive training [10].

The organizational patterns of agile software development address patterns that can be used to improve the understanding of the development team and help in customer capacity building. Agile practices can be seen as patterns as the development team regularly observes them in their practical work. In the agile methodology, organizations, teams, and customers are expected to sense market changes rapidly, adapt capabilities, and create solutions to achieve goals in a rapidly changing environment [9]. To achieve better results and fully utilize an organizational pattern to solve the problems, we need to fully understand not only the pattern itself but also the pattern language. As it mentioned by [2, 20, 21]. We want to propose a pattern language for the mentioned challenge. In addition, pattern languages in general may not always reveal every reliable connection with other patterns [7]. Despite this, there are many hidden relationships between patterns described in the description and the areas of interest.

We documented our experiences with dozens of software projects already implemented in government organizations. Most developed software projects are implemented and functional in Afghan government organizations. We are encouraged to consider our own experiences and literature to discover some organizational patterns for building customer capacity through organizational patterns, which improve the development team's understanding. This paper focuses on the customer who uses custom software in government organizations.

The results of this study will motivate owners/policymakers to invest in capacity building to enhance their organizational performance/productivity. The results indicate that capacity building has a positive and significant impact on customers' performance during software project development.

The structure of the paper is as follows Section **??** brings the story behind the patterns of building customer capacity through organizational patterns that improve the development team's understanding. Section **??** explains the organizational patterns and newly discovered patterns. It also proposes a pattern language for the mentioned problem. Section **??** concludes the study and outlines the future work.

2. THE STORY BEHIND THE PATTERNS

In 2019, the Ministry of Education requested that the Education Management Information System (EMIS) develop an online teaching/learning application. In the application, which is now available,[1] students can access and follow their lessons even when not in class, with all lessons from elementary and high school covered.

The customers had very limited computer and technology knowledge. The development team started *Perform Need Assessments* to understand customer capacity. The customers provided some basic requirements and some documents that they thought would be enough, but the requirements were not helpful to the development team. Some customers didn't realize the significance of this software project and technology. We provided *Train Customers with Technical and Domain Expert*. We discussed the software project's purpose, objectives, outcomes, and impact during the training workshop with the customers. During the workshop, we created a pleasant

[1] www.maarif.af

environment with the customers, who were motivated and supported by the development team. In addition, we remind customers that they are the actual users and owners of the software project. This is because their contribution will ensure the system's success and potential benefits. During the training workshop, the development team worked with the customers and focuses on *Illustrate Standard Operating Procedure with Customer*.

The development team designed the prototypes and shared them for customers' satisfaction *Build Trust and Satisfy Customers by Results*. We listened to customers carefully and ensured them that the requirements would not be misused. The development team decided to build expert customer capacity with the development team *Engage Customers and Build Master Trainers*. These customers became master trainers to train the remaining employees in the organization. The development team built the outstanding capacity of the customers during the software project development. As a result, we delivered high-quality software to the organization on time.

We have used some existing organizational patterns, and some are newly discovered. These patterns are composed in hierarchical form, observed in the pattern sequence. In hierarchical pattern composition, the superior pattern holds the sub-ordinate patterns. In other words, each sub-ordinate pattern refines the previous one as was discussed by Vranić [25]. The use of patterns in the sequence reflects the strength of the patterns. The patterns sequence identifies the relationship with other patterns [13].

3. THE PATTERNS

In this paper, six organizational patterns (highlighted in green in Figure **??** were discovered for building customer capacity through organizational patterns. We have observed these patterns in practice during the last ten years of working experience in government organizations. In documenting and writing these patterns, iterative and creative approaches were used [5, 13, 18, 22, 25, 27, 26]. We used brainstorming, document analysis and expert collaboration to collect the most relevant and high-quality information, as is known in the literature [3, 4, 6, 8, 11].

3.1 Pattern Format

The newly discovered patterns are described in Coplien and Harrison's pattern format [5] in Sections 3.2–3.7 structured by using the most contradicting forces expressed in the *but* form proposed by Vranic' [12]. They are written in the following structure:

> <**Pattern Name**>
> **. . .** – Context of the pattern occurs.
>
> ✣✣✣ – The text in bold describes the actual problem as a conflict of the two most prominent contradicting forces.
>
> **Therefore** – The solution.
>
> ✣✣✣ – An optional part with resulting consequences upon applying the given pattern.
>
> **Description** optional description to explain the pattern.

3.2 Perform Need Assessment

. . . The development team has just contracted to develop a software project for the customer's organization. It's unclear how much domain knowledge and experience the customer has with technology.

✣✣✣

The customer can not understand how to communicate the correct requirements to the development team.

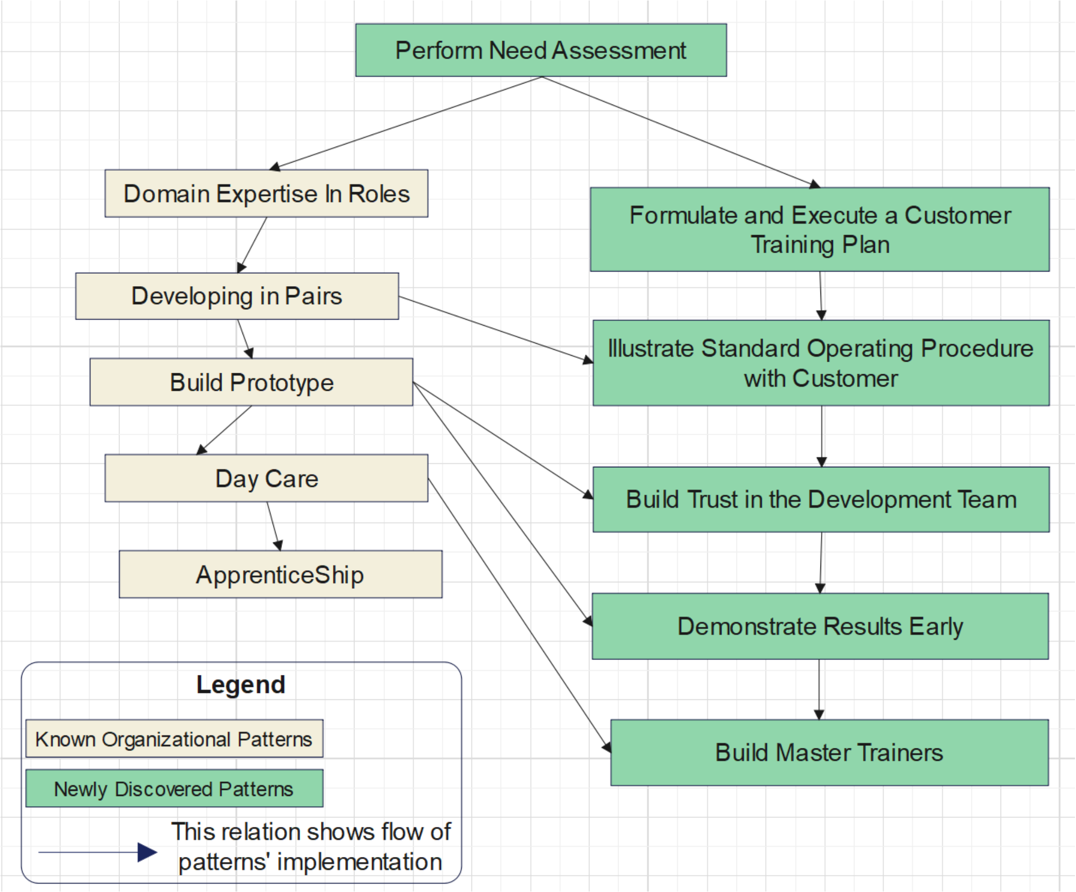

Perform Need Assessment
Domain Expertise In Roles
Formulate and Execute a Customer Training Plan
Developing in Pairs
Illustrate Standard Operating Procedure with Customer
Build Prototype
Day Care
Build Trust in the Development Team
ApprenticeShip
Demonstrate Results Early
Legend
Known Organizational Patterns
Newly Discovered Patterns
This relation shows flow of patterns' implementation
Build Master Trainers

Fig. 2. Building customer capacity through organizational patterns.

The management wants to implement the software project, but the development team wants to know how many employees have domain knowledge and how many are technology savvy. It is also important for the development team to know how many customers have basic computer literacy.

The management wants the software project delivered faster and of higher quality, but the development team wants to know which customers can share the right requirements at the right time.

The management wants to introduce specific staff to the development team for the development and implementation of the software project, but the development team wants to find out which staff members have communication and coordination skills in order to build the software project with their help on time and with good quality.

Therefore:

Perform a needs assessment workshop and create a comprehensive capacity-building plan based on the criteria set for the customers. The plan will identify how well the customer understands technology and business processes and who will be able to provide the team with accurate requirements within the specified time frame. This plan helps the development team in the implementation of the software project.

❖❖❖

We conducted a needs assessment in the organization before the development of the software project. This was done during a one-day workshop short questionnaire was created that included questions about software usage and information technology. As a first step, we asked the customer a few questions about their capacity. The development team will ensure that the customers understand the administrative processes and domain knowledge or not. The development team will understand whether the customers have basic computers and technology skills. The development team can determine which customer representatives are best suited to lead in the communication of requirements. The development team will also determine customer trust in software projects. The capacity building plan will help in delivering of software project successfully.

3.3 Formulate and Execute a Customer Training Plan

. . . According to the assessment, there are customers who do not know the workflow, have trouble using technology, or are unable to communicate with the development team.

❖❖❖

Due to customers' limited capacity, they cannot describe the organization's workflow to the development team and cannot share the correct requirements.

Customers are interested in implementing a software project in their organizations, but they cannot share the correct requirements with the development team due to capacity limitations. The development team build the software project based on false requirements.

The customers wish to describe the organization's workflow, but the development team does not understand it well due to the low capability of the customers.

Although the customer organization has human and physical resources, the concept may be feasible, but if the customer does not have the capacity to provide good requirements, the implementation may be flawed.

Therefore:

Train the customers in various workshops by domain and information technology experts. The workshops may conduct at different stages during the software project development.

The business analysts can work with the customer on capacity building. They can understand and has the experience of how to educate the customers. The business analyst further communicates with the development team.

The workshop will inform all customers about the importance of technology and the use of relevant information systems in public organizations. Customers will understand the types of requirements the development team needs for the software project. Customers will be able to share appropriate requirements with the project team. Such workshops may be conducted throughout a software development project to train customers about requirements elicitation, process simplification, software testing, and software implementation.

3.4 Illustrate Standard Operating Procedure with Customer

. . . The development team has trained the customer in requirements elicitation and technology awareness. Now the development team has to work with the customers to simplify and consolidate the administrative processes.

❖❖❖

In some organizations, administrative processes are complex and ambiguous. The customers are not able to properly explain their knowledge of a domain to a development team.

Administrative processes are complex and outdated, making them difficult for the development team to understand, but due to customers' low capacity, they are difficult to explain. If the customer fails to explain the processes correctly, the project may run over schedule and cost.

The customers' organization has defined the procedures and policies, but the processes they follow are outdated and not aligned with market demands.

The organization can re-engineer its processes using several methodologies and frameworks, but implementing these methodologies and frameworks is challenging due to low customer capacity.

Therefore:

To understand the overall flow of requirements, the development team needs to visualize the organization's entire administrative processes and work closely with the customer. It should be approved by the customer department and shared with the development team for the soft project to be developed using simplified processes.

From the experience and literature, simplifying and designing the administrative processes is extremely helpful for understanding software project requirements. A business process is a set of related activities that add value for the customer [24]. Customers understand the flow and may provide accurate requirements to the development team. The development team can use this pattern to simplify administrative processes and clarify requirements. It may also be used to design the business flow of a department.

✣✣✣

During the software project development, the development team can use business process management. Business process management has close relation with software development. It manages, control, and support routine processes [23]. It enables organizations to align business functions with customer needs. Sometimes the processes in an organization are simplified but outdated. These processes need to be re-engineered.

The development team and the customer may use a process re-engineering life cycle framework, which enhances administrative processes and strengthens the business. This framework consists of three significant steps [24]:

(1) Process assessment, which includes process identification and process analysis, and the decision is made based on the result of this assessment

(2) Process improvement, where the redesigning and execution of the processes are included

(3) Process change, where the management decides whether the proposed changes can be implemented or not

The development team may use this pattern to re-engineer and make administrative processes more standardized (see Figure 3).

The development team worked with the customer to simplify and re-engineer the administrative process. They can share the initial prototype with the customer. Now they need to satisfy customers by showing results.

3.5 Build Trust in the Development Team

. . . The development team has already started working with customers on product development. They need to establish a trusted environment with the customer.

✣✣✣

Customers are not sure that the desired system will solve their problems. They show resistance toward software project implementation, but the development team wants to build trust.

Customers show resistance to change [28] and have concerns that the development team may misuse the provided information, but the development team needs complete requirements from the customers. Some of the customers do not want to help the development team in the successful completion of the project [6]. They think their power will diminish and not keep their job safe, but the development team wants customer help and needs customer requirements to complete the software project.

Some customers believe that the new system may not bring about positive changes. They show resistance toward software projects, but the development team wants to build software projects based on submitted requirements, which may change the existing administrative processes.

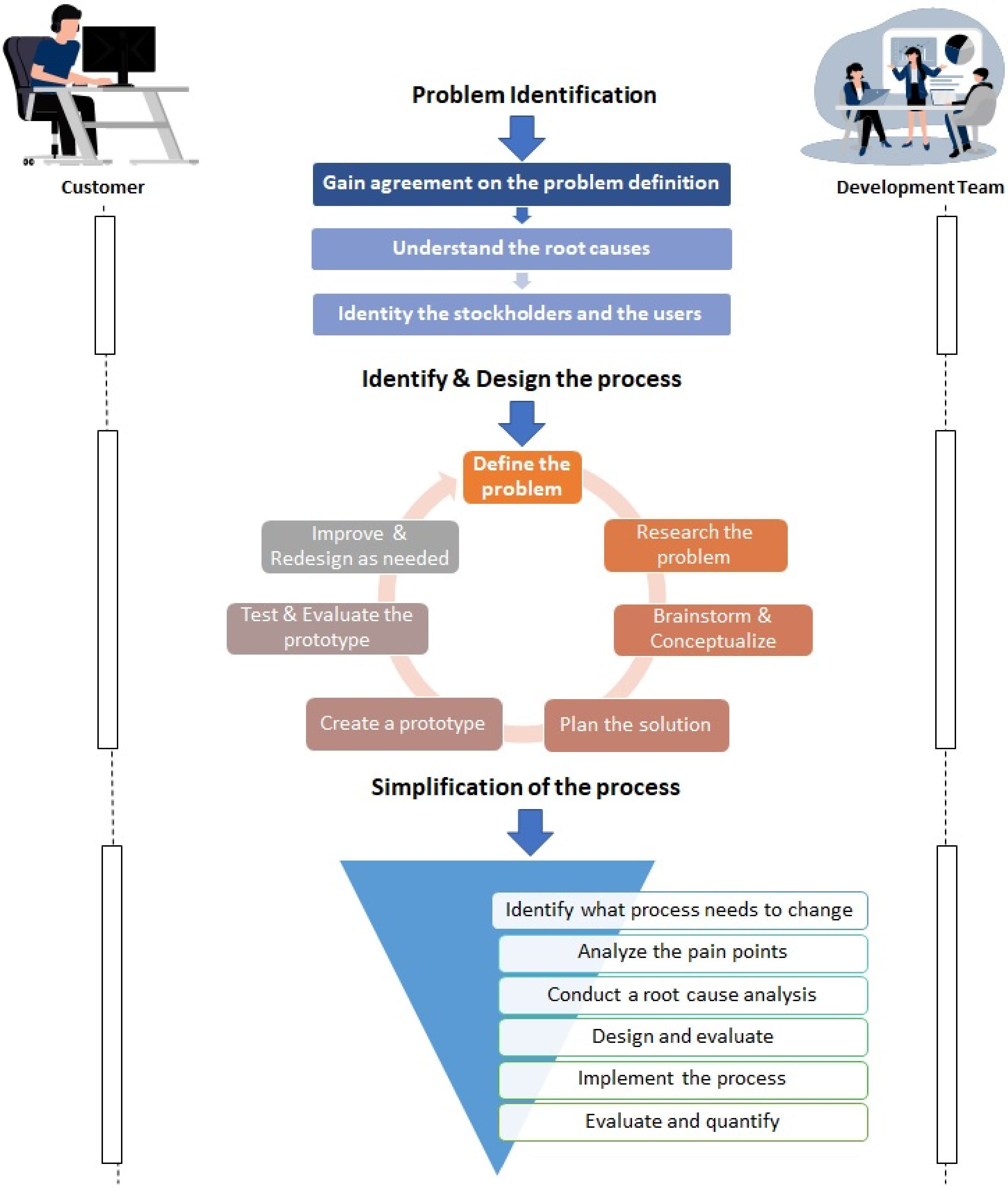

Customer
Development Team
Problem Identification
Gain agreement on the problem definition
Understand the root causes
Identity the stockholders and the users
Identify & Design the process
Define the problem
Research the problem
Brainstorm & Conceptualize
Plan the solution
Create a prototype
Test & Evaluate the prototype
Improve & Redesign as needed
Simplification of the process
Identify what process needs to change
Analyze the pain points
Conduct a root cause analysis
Design and evaluate
Implement the process
Evaluate and quantify

Fig. 3. Standard operating procedures between the development team and customer.

Therefore: The development team should design the essential part or module of the software projects. The developed module should be shared with the customers for their engagement and satisfaction before the entire software project development begins.

The developers must build a trusted environment among themselves and with the customer.

✣✣✣

The organizational pattern *Community Of Trust* [5] says that the technical team members should work with the customer in the development of the project plan. The project plan increases customer confidence in the software projects. That extends from *Size The Schedule* pattern. The customer must know how long takes the project development. Who is and what components are involved? How many resources are needed? Once trust is established, the development team should need to Demonstrate results early. The development team will demonstrate the results together with customers in the development environment.

3.6 Demonstrate Results Early

. . . Product development has already begun among the development team and the customers. The development team needs to establish or demonstrate the result to the customer.

✣✣✣

Customers show resistance to software project implementation, but the development team wants to demonstrate the results.

Customers are concerned that the new system will be difficult to use, but the development team should prepare system prototypes for the end-user to use the software projects.

Sometimes, customers are too busy with their routine work, but developers are free and willing to work with them. The customers think that working with a development team is a waste of time because they have not seen any results yet.

In some organizations, the users are not technology-oriented. They are concerned that they do not have enough capacity to run the new system or may the product fail, but the development team wants to build the software project, not build the end-user capacity.

Therefore: The development team should implement the essential part or module of the software projects. The developed module should be shared with the customers for their engagement and satisfaction before the entire software project development begins.

The developers should encourage the customer to use the developed software, and they should confirm to the customer that they work together for a common goal[18]. Customers want results and are satisfied with the results. When customers are satisfied, they may not show much resistance toward software development. In the initial phase of software development, if there is a lack of trust between the development team and the customer, the development team may use this pattern.

✣✣✣

The customer must know how long takes the project development. Who is and what components are involved? How many resources are needed? Once the result is established, the development team should create a training package. The development team will train the master trainers in the development environment.

3.7 Build Master Trainers

. . . Customer needs assessment is performed by the development team, and the capacity building plan is prepared based on this assessment. Both teams decided to train the trainers and the nominated staff in the development environment.

✣✣✣

It is difficult for the development team to train all users on a software project. In addition, it costs the development team and time-consuming.

Few employees have been trained in the development environment, but if this product is designed for the whole organization, the rest may not be able to use this product properly. The trained staff may train the rest of the employees, but they need to use or build the user manual.

Once the product is complete and the organization wants to train its employees, it may face budget cuts or higher product costs. It is also possible that the organization will not use the product and be harmed.

After the product is implemented in the customer environment, there may be some technical problems, but the trained staff should provide initial support, such as backup restoration and product configuration.

Therefore:

Build the capacity customers introduced in the development environment during software project development. These customers should become master trainers to train the remaining employees in the organization.

In government organizations, some of the users are not technology-oriented. They need to practice basic computer literacy and use the developed software. The developers should conduct training for end-users of the developed software. The developed software project should be localized in the end-user language to understand and learn quickly. The development team needs to prepare user guidelines for the end-users' better understanding. The development team applies this pattern to too many end-users in an organization where they cannot train all end-users. It will not cost much and on the other hand, time will not be wasted.

❖❖❖

Some organizational patterns focus on team training rather than customer training. *Day Care* pattern says experts need to train new persons or inexperienced, but these new persons are not customers. *Day Care* separates the progress team from the training team, but the training team does not give training to the customers. *Domain Experts In Roles* pattern is to answer the domain-specific requirements and design questions.

Both domain training and process training should provide to the development team, but domain training is more important than process training [5]. This pattern does not mention conducting training for the customers. This pattern emphasizes getting requirements from the customer and training the technical team in the domain.

The *Apprenticeship* pattern says every new employee should work in an apprenticeship program. Hire the inexperienced and turn them into experts [5]. This pattern does not contain too much end-user capacity building. It concentrates on newly hired employees.

4. CONCLUSION AND FUTURE WORK

Due to low customer capacity, we have found in practice and in the literature that *False Requirements*, *No Basic Technology Knowledge*, *No Domain Knowledge*, *Mismanaged Timeline*, *No Trust in Software Project*, and *Unproductive Discussion* are the main customer capacity challenges to the software project development. We documented our experiences and five newly discovered patterns in government organizations to solve the mentioned challenges. We present this documented experience in the form of patterns in this paper to improve the customer capacity Before work begins on a software project or during the software development process. Some organizational patterns that reduce the mentioned challenges are also used along with newly discovered patterns. We recommend that the development team apply the five new patterns in situations where the mentioned challenges exist. It is unnecessary to use these patterns when the mentioned challenges do not exist.

We will observe some new patterns that identify how the organizational patterns help in building the development team's capacity. We will focus more on organizational patterns to build a comprehensive pattern language for building the development team's capacity. There are still some organizational patterns to analyze and document in this area.

ACKNOWLEDGMENT

We would like to thank Kyle Brown for being our shepherd. Our sincere thanks also go to our writer's workshop group members: Michael Weiss, Fernando Pereira, Eduardo Guerra, Reinaldo R. Rosa, Hironori Washizaki, Foutse Khomh, and Yann-Gael Gueheneuc.

The work reported here was supported by the Scientific Grant Agency of Slovak Republic (VEGA) under grant No. VG 1/0759/19, by the Operational Programme Integrated Infrastructure for the project: Research in the SANET network and possibilities of its further use and development (ITMS code: 313011W988), co-funded by the European Regional Development Fund (ERDF), and by the Slovak Research and Development Agency under the contract No. APVV-15-0508.

REFERENCES

Tahir Ahmad, Faiza Farrukh, and Sana Nazir. Capacity building boost employees performance. *Industrial and Commercial Training*, 2015.

Christopher Alexander. *The Timeless Way of Building*. Oxford University Press, 1979.

Sebastián Alvarez, Kevin Duy, Mireya Zapata, Jorge Galarza, Danilo Martinez, and Carlos Puco. The communication between client-developer in the process of requirements elicitation for a software project. In *World Conference on Information Systems and Technologies*, pages 36–45. Springer, 2021.

Krumal J Bhavsar, Vrutik Shah, and Samir Gopalan. Process life cycle framework: A conceptual model and literature study of business process re-engineering for software engineering management. *CiiT International Journal of Software Engineering and Technology*, 11(6):096–100, 2019.

James O Coplien and Neil B Harrison. *Organizational patterns of agile software development*. Prentice-Hall, Inc., 2004.

Bill Davey and Kevin R Parker. Requirements elicitation problems: a literature analysis. *Issues in Informing Science and Information Technology*, 12:71–82, 2015.

Michael Falkenthal, Uwe Breitenbücher, and Frank Leymann. The nature of pattern languages. *cit. on*, page 14, 2018.

Daniel Fay, Neville Stanton, and Aaron Roberts. Streamlining experimental processes using bespoke software. 2018.

Daniel R Greening. Agile base patterns in the agile canon. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pages 5368–5377. IEEE, 2016.

Ville T Heikkilä, Daniela Damian, Casper Lassenius, and Maria Paasivaara. A mapping study on requirements engineering in agile software development. In *2015 41st Euromicro conference on software engineering and advanced applications*, pages 199–207. IEEE, 2015.

Linda D Hollebeek, David E Sprott, Tor W Andreassen, Carolyn Costley, Phil Klaus, Volker Kuppelwieser, Amela Karahasanovic, Takashi Taguchi, Jamid Ul Islam, and Raouf Ahmad Rather. Customer engagement in evolving technological environments: synopsis and guiding propositions. *European Journal of Marketing*, 2019.

Patrik Honíšek and Valentino Vranić. Mining drama patterns in dramatic situations. In *27th Conference on Pattern Languages of Programs, PLoP*, 2020.

Waheedullah Sulaiman Khail and Valentino Vranić. Reflecting pattern relationships in a pattern format. In *Proceedings of the 24th European Conference on Pattern Languages of Programs*, pages 1–5, 2019.

Ines Mergel. Agile innovation management in government: A research agenda. *Government Information Quarterly*, 33(3):516–523, 2016.

Patti Millar and Alison Doherty. Capacity building in nonprofit sport organizations: Development of a process model. *Sport management review*, 19(4):365–377, 2016.

Pramod Kumar Misra and Jitendra Mohanty. A review on training and leadership development: its effectiveness for enhancing employee performance in indian construction industry. In *IOP Conference Series: Materials Science and Engineering*, volume 1045, page 012020. IOP Publishing, 2021.

Chinazo Olisaeloka Okonkwo. Capacity building and employee performance in plastic manufacturing companies in anambra state. 2022.

Stacie Petter. Managing user expectations on software projects: Lessons from the trenches. *International Journal of Project Management*, 26(7):700–712, 2008.

Tarmo Robal, Deniss Ojastu, Ahto Kalja, and Hannu Jaakkola. Managing software engineering competences with domain ontology for customer and team profiling and training. In *2015 Portland International Conference on Management of Engineering and Technology (PICMET)*, pages 1369–1376. IEEE, 2015.

Niels Seidel. Empirical evaluation methods for pattern languages: sketches, classification, and network analysis. In *Proceedings of the 22nd European Conference on Pattern Languages of Programs*, pages 1–24, 2017.

Waheedullah Sulaiman Khail and Valentino Vranić. Treating pattern sublanguages as patterns with an application to organizational patterns. In *Proceedings of the 22Nd European Conference on Pattern Languages of Programs, EuroPLoP 2017*, Irsee, Germany, 2017. ACM.

Jeff Sutherland and James O Coplien. *A Scrum book: The spirit of the game*. Pragmatic Bookshelf, 2019.

Wil MP Van Der Aalst, Marcello La Rosa, and Flávia Maria Santoro. Business process management, 2016.

Carchiolo Vincenza, Catalano Giovanni, Malgeri Michele, Pellegrino Carlo, Platania Giulio, and Trapani Natalia. Bpm tools for asset management in renewable energy power plants. In *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 645–649. IEEE, 2019.

Valentino Vranić, Aleksandra Vranić, and Waheedullah Sulaiman Khail. Growing organizations with patterns: Lessons from drama. In *Proceedings of the European Conference on Pattern Languages of Programs 2020*, pages 1–11, 2020.

Shakirullah Waseeb, Waheedullah Sulaiman Khail, and Valentino Vranic. Establishing a pattern language for the organization of distributed software development. In *26th European Conference on Pattern Languages of Programs*, pages 1–9, 2021.

Shakirullah Waseeb, Waheedullah Sulaiman Khail, Haji Gul Wahaj, and Valentino Vranić. Extracting relations between organizational patterns using association mining. In *Proceedings of the European Conference on Pattern Languages of Programs 2020*, pages 1–9, 2020.

Krzysztof Wnuk, Richard Berntsson Svensson, and David Callele. The effect of stakeholder inertia on product line requirements. In *2012 Second IEEE International Workshop on Requirements Engineering for Systems, Services, and Systems-of-Systems (RESS)*, pages 34–37. IEEE, 2012.

Emmanuel Erastus Yamoah and Philip Maiyo. Capacity building and employee performance. *Canadian Social Science*, 9(3):42–45, 2013.