# The Developer's Journey: A Storytelling Framework for Cooperative Learning in Software Engineering

Patrick Wolfschwenger[1][a], Mona Emara[2][b], Wolfgang Lumetsberger[3], Thomas Hatter[3], Barbara Sabitzer[1][c] and Zsolt Lavicza[1][d]

[1] *Johannes Kepler University, Linz, Austria*

[2] *Damanhour University, Damanhour, Egypt*

[3] *smartpoint IT Consulting (Bechtle), Austria*

Keywords: Continuing Professional Training and Development, Lifelong Learning, Cloud Computing Education, Narrative-based Learning, Project-based Learning, Instructional Design.

Abstract: An educational storytelling concept is presented, inspired by the archetypal story pattern common in ancient myths as well as modern day adventures. Software development is a complex profession demanding constant learning and improvement in a field changing almost daily. Apart from learning new technologies and writing computer code, much of a developer's time is spent on problem-solving and debugging – that is, detecting and correcting errors and bugs that cause a system to break or behave unexpectedly. The average developer regularly goes through a series of transformative steps to overcome intricate problems that often appear obscure and enigmatic at the beginning. The return with special knowledge to share with others is the final reward earned on the Developer's Journey. Under the premise that a good story can change our perception and offset the biases of our interpretation of reality, a didactic method has been designed for sharing and interpreting experiences in a cooperative learning environment. In the context of cloud computing education, its effects on problem-solving, motivation and perception are evaluated. We analyze transformative learning opportunities in connection with narratives and discuss its potentials and limitations in community-based learning.

## 1 INTRODUCTION

Stories are inherently used to organize and make sense of experiences and to share that understanding with others. Through stories, we bring structure to our memories and provide opportunity for learning from one another. When a story catches our attention and engages us, we are more likely to absorb its meaning compared to when the same message would be presented solely in facts and figures. Comprehending material presented in narrative form comes easy to us due to our deeply internalized understanding of how stories are told (Hazel, 2008).

In didactics, storytelling is used as a tool to create interest, provide a structure for remembering content, share information in a familiar and accessible form and create a more personal connection between learner and instructor. More empirical evidence about the effectiveness of this teaching strategy and resources on how to best use it for educational purposes are still required (Landrum et al., 2019).

In this paper, we introduce the Developer's Journey and evaluate its application in the professional development of cloud developers. We discuss the relationship between developers' problem-solving achievements and perceptions of the task flow and intrinsic motivation and draw conclusions whether fruitful learning experiences could be created so that transformative learning happens more regularly.

As the educational uses of cloud computing are on the rise, it becomes increasingly important to ensure the appropriate training and advisory for those concerned, including IT employees, instructors, learners and decision-makers (Wolfschwenger et al., 2020). The Developer's Journey was designed to give cloud developers the opportunity to exchange experiences and ideas as part of an in-company continuing educa-

[a] https://orcid.org/0000-0001-5325-0511
[b] https://orcid.org/0000-0002-9277-7073
[c] https://orcid.org/0000-0002-1304-6863
[d] https://orcid.org/0000-0002-3701-5068

Wolfschwenger, P., Emara, M., Lumetsberger, W., Hatter, T., Sabitzer, B. and Lavicza, Z.
The Developer's Journey: A Storytelling Framework for Cooperative Learning in Software Engineering.
DOI: 10.5220/0011064300003182
In *Proceedings of the 14th International Conference on Computer Supported Education (CSEDU 2022) - Volume 2*, pages 525-533
ISBN: 978-989-758-562-3; ISSN: 2184-5026

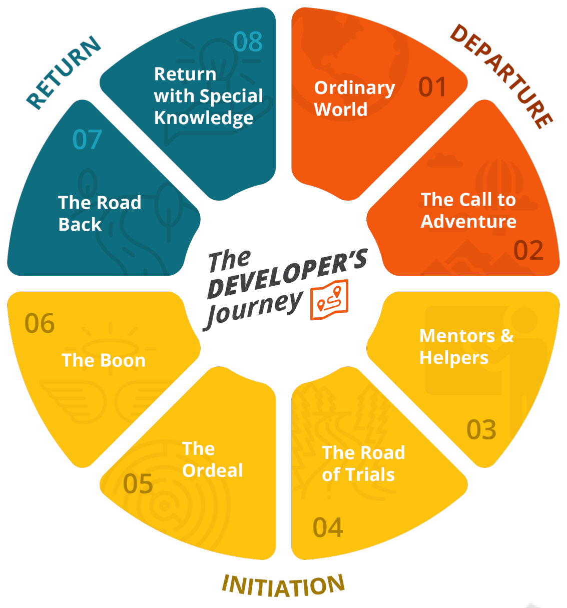


Figure 1: Illustration of the Developer's Journey.

tion program. In the spirit of community-based learning, the framework offers every participant the opportunity to spread, discuss and reflect upon requirements, typical problems and innovative approaches with the community in the form of a structured story rather than informal exchanges.

At first, background information on learning with narratives, transformative and collaborative learning is provided. Then, the Developer's Journey is delineated with reference to the individual phases of the software development process. Practical application of the concept is subsequently evaluated with quantitative and qualitative methods.

## 2 CONTEXT

Narratives are fundamental to our everyday lives. We all naturally use them to make meaning of our experiences and construct our individual identities (Fisher, 1984). Narrative-based learning (NL) means learning through stories, whether compiled or received, whether told or heard, whether written or read. On the one hand, narrators are learners who move from a cognitive understanding of a temporal sequence linked to their own experiences and combine many small pieces of information into a meaningful concept over time, which creates the possibility for reflection of underlying assumptions. On the other hand, receivers are learners who are drawn into an engaging experience and make sense of the narrated concept on a deeply human level.

By taking advantage of the inherent structure of narratives, NL environments provide learners with engaging worlds where they can actively participate in motivating story-based problem-solving activities (McQuiggan et al., 2008). NL frameworks serve as cognitive structures, help individuals to frame and understand their perceptions of the world and offer significant potential for supporting transformative learning processes (Bruner, 1991).

For transformative cognitive change to occur, a prior interpretation is needed to construe a new or revised interpretation of the meaning of one's experience (Mezirow, 1996). Cognitive transformation requires a learning process that includes both assimilation (adding new information into existing mental structures) and accommodation (changing base ideas in response to new information) (Corder et al., 1999). In NL, not only narrators but also recipients are actively engaged through critical reflection and discourse to question assumptions, expectations and context to achieve deeper meaning and new perspectives to guide their future actions.

Given the relationship between narrative and transformative learning, creating a NL framework for cooperative learning in software engineering holds much appeal. Niehaus et al. (2014) suggest that, to better support the creation of narrative-centered tools, creators need a flexible framework to integrate, catalog, select, and reuse narrative models. They specify model metadata to allow creators and practitioners to discover and understand models more easily. Prior work in computing education using narratives includes Guzdial and Tew (2006), who use an instructional design approach for motivating computing education. Landrum et al. (2019) give an overview of a variety of pedagogical storytelling approaches recently investigated and present suggestions and recommendations for implementing storytelling as a didactic approach.

While the motivational and pedagogical benefits of NL are compelling, it is important for researchers and educators to address how to maintain learners' motivation and maximizing learners' time-on-task as well as to address any adverse effects on problem-solving. A few studies have measured developers' intrinsic motivation, flow state, and knowledge construction. McQuiggan et al. (2008) investigated the effect of narrative on learning experiences and outcomes with eighth-grade middle school students and found motivational benefits of narrative-centered learning regarding self-efficacy, presence, interest, and perception of control. By drawing on the concept of narrative-centered learning, Kuusinen et al. (2016) have shown that intrinsic motivation and autotelic experiences are significant predictors of developers' user experiences and developers' needs are characterized by efficiency, informativeness, intuitiveness and flexibility. Mott et al. (2006) draw our attention to a critical need for orchestrating all of the events in the unfolding story to support appropriate levels of learners' motivation, flow state, and constructing knowledge for effective problem-solving in order to develop competences as they progress into more demanding academic contexts and ultimately the workforce.

Storytelling provides an opportunity to integrate narratives in academic courses with community engagement. Bareiss and Griss (2008) have proposed a story-centered curriculum as part of a software engineering program that prepares experienced developers for leadership roles in development teams. Within this framework, students confront realistic technical and organizational problems, including conflicting requirements, limited resources and challenges to team leadership. The system draws upon a learning-by-doing approach with the goal of producing a transformative experience for practicing software professionals.

In recent years, digital storytelling has achieved enormous upswing in teaching and research. Digital storytelling is the practice of combining narrative with digital content, including images, sound, and video (Vivitsou, 2018). In the tertiary sector, students reported interest for such teaching methods as part of a curriculum (Dia-Eddine, 2021). Also, K-12 has made positive experiences with digital storytelling. Kim and Li (2021) examine how digital storytelling facilitated students' reflection and learning in a project-based middle school capstone program.

## 3 DESIGN

The Developer's Journey is a synthesis of the various, well-known variations of the monomyth and the daily working life of a software developer. The concept of the Developer's Journey arose from considerations of applying the Hero's Journey as a didactic method, which is a common template of stories in narratology and comparative mythology that involves a hero who embarks on an adventure, is victorious in a decisive crisis, and returns home changed or transformed. Hero myth pattern studies were popularized by Joseph Campbell (2008), which sets out 17 distinct stages of a Hero's Journey. A range of variations evolved over time, such as Leeming (1998) and Vogler (2020), proposing adapted versions with different emphases. The stages are widely classified into three acts: Departure, Initiation and Return.

Since the problem-solving process of software developers is also reflected in patterns of the monomyth, an adapted version has been forged that serves as our template of the Developer's Journey. It was aligned with the daily routines of cloud developers in the consulting industry. Essentially, each project follows the scheme of preparation, execution and retrospective analogous to the three acts of the Hero's Journey. The structure is intended to support the organization of the experiences to be shared along with the purposeful introduction of content and complexity.

For the construction of the Developer's Journey, we have built on the work of Avraamidou and Osborne (2009), who explored the potential of narrative in science education and identified the seven necessary components of a narrative based on a meta-analysis:

- *Purpose* – Helps us understand the natural world
- *Events* – Contains a sequence of events connected to each other and involving a unified subject
- *Structure* – Has an identifiable structure around the sequence of events (beginning, middle, end)
- *Time* – Concerns the past
- *Agency* – Has actors or entities that cause and experience events
- *Narrator* – Has a teller who gives account of the story
- *Reader* – Is received by someone who needs to recognize the artifact as a narrative

The Developer's Journey was developed with those characteristics of a narrative in mind. Its purpose is the exchange of experiences in software development. It is built around a structured sequence of events connected to each other and has the developer as a unified subject and other persons and entities involved. Developers report from their past experiences, which may also incorporate an outlook for the future. The story is told by the developer and received by fellows in spoken form. That created the framework for the design of the Developer's Journey.

## 4 THE JOURNEY

The Developer's Journey, which is illustrated in Figure 1, centers on a software developer, who deviates from the definition of an as-is and to-be state, faces a series of tasks and challenges until reaching its climax, which is putting the developed solution into practice, and completes it with follow-up orders, maybe rework, and gained knowledge in hand. We define the eight steps as follows.

### 4.1 Departure

The developer starts his or her story in the *Ordinary World*, which represents the baseline situation before any action or operation is taken. At this stage, an overview of the different technologies and systems that play a role in the upcoming journey as well as their configurations at the outset is given.

It is followed by *The Call To Adventure*, which can be the order of a customer, a user story that needs to be implemented or a sudden system failures resulting from unforeseeable and unavoidable circumstances. The developer explains the purpose of the endeavor and clarifies the objectives and limitations of the work.

### 4.2 Initiation

The *Mentors and Helpers* supply the developer with tools and inspiration needed to accomplish the adventure. These are recurring characters over the course of the journey, such as project participants, colleagues or external individuals. While clients can be seen as the mentors who set the course of the journey, colleagues are the helpers who assist in finding a satisfying solution.

A range of smaller tasks needs to be completed on *The Road of Trials*, which leads to the achievement of the specified target. At this point, the narrative involves breaking down the problem into smaller pieces so that the audience can follow more easily. Since we know that active learning thrives on trial-and-error problem solving, both accurate and futile approaches should be pointed out. This prevents listeners who might find themselves in a similar situation from trying the same fruitless attempts.

*The Ordeal* refers to the release of the developed solution. When a software system implementation or update becomes operational, it becomes officially and formally available to users for productive use. The go-live is typically the most critical milestone in a development cycle, as it marks the culmination of weeks, months and sometimes even years of groundwork, design and development involving a company's internal and external stakeholders. The developer describes the execution of the go-live, release management as well as the methods applied (i.e. continuous delivery).

The final result of the development process represents *The Boon*. The developer demonstrates the working solution and practice examples in a hands-on way so that participants get a clear picture of the attainment. If practical, the audience can be directly involved.

### 4.3 Return

*The Road Back* depicts all the activities that happened after the release. The support phase begins when a system solution becomes operational and continues until it reaches the end of its useful life. The developer outlines change requests, alterations and feedback of clients received so far.

The *Return with Special Knowledge* is the final stage on the Developer's Journey. The developer gives a summary and discusses the lessons learned to generate additional value for the audience. It should be made clear to the colleagues in the audience on which topics the developer has become an expert and can henceforth provide assistance and support. The option to consult an expert of a certain topic may lead to additional motivation to find solutions to similar problems, or may at least promote discussions with that expert, which could improve the knowledge of the colleague in the future.

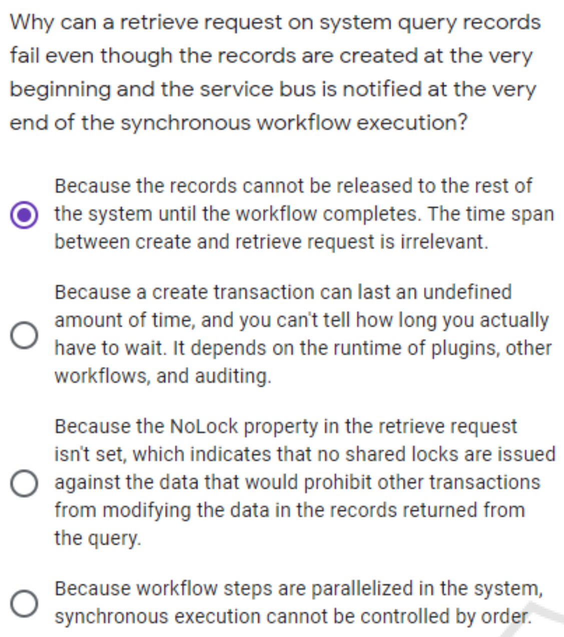
## 5 EVALUATION

### 5.1 Data Collection

The evaluation is based on two online surveys applied before and after the intervention and semi-structured interviews. The online surveys are designed to measure problem-solving skills before and after the intervention and include rating scale questions for self-assessment. The interviews were conducted retrospectively with five randomly selected participants to get in-depth feedback.

Self-assessment responses allow a classification of the experimental group and analysis of personal attitudes toward the intervention. Rating scale questions related to perceived competence and knowledge required, task value and achievement goal orientation serve as classification criteria for the experimental group. Additional rating scale questions were posed to reflect on intrinsic motivation, flow state, perceived learning success and comprehension of the method. The rating scale questions are in the five-point Likert scale format and listed in Table 1.

A one-group pretest-posttest design has been used to evaluate the learning outcome occurring as a result of intervention. The problem-solving items involve scenarios in which the respondent takes the role of a cloud developer that has to evaluate information relating to actual business cases and make design decisions. A correct answer represents the only possible implementation option. Figure 2 shows a sample item. Each of the two tests contained six multiple-


Why can a retrieve request on system query records fail even though the records are created at the very beginning and the service bus is notified at the very end of the synchronous workflow execution?

[✓ Because the records cannot be released to the rest of the system until the workflow completes. The time span between create and retrieve request is irrelevant.]

Because a create transaction can last an undefined amount of time, and you can't tell how long you actually have to wait. It depends on the runtime of plugins, other workflows, and auditing.

Because the NoLock property in the retrieve request isn't set, which indicates that no shared locks are issued against the data that would prohibit other transactions from modifying the data in the records returned from the query.

Because workflow steps are parallelized in the system, synchronous execution cannot be controlled by order.

Figure 2: Problem-solving sample item.

choice questions measuring problem-solving ability before and after the intervention. They were provided by those who designed and presented the stories. Comparison of the pretest and posttest results provides evidence of the effectiveness of the independent variable (the intervention). If the pretest and posttest scores differ significantly, the difference may be attributed to the intervention. But because there is no control group, this inference is uncertain, and the difference might be caused by extraneous variables. There is also no way of judging whether the process of pretesting influenced the results, since there is no baseline measurement against groups that remained completely untested.

Five semi-structured interviews with randomly selected participants were conducted after the intervention to obtain more in-depth feedback. Open-ended questions should be a trigger for the participants to comment freely and in as much detail as they like. Narrators were questioned about their experiences with the creation of their journeys. Recipients were asked how they assess the value and quality of the form of presentation. Thematic analysis was used to summarize and analyze the responses.

The target group consists of software engineers in the cloud computing domain. The Developer's Journey was tried out in an in-company continuous education program, which brings employees of different company sites together and gives them the possibility to exchange experiences and practices. The course as well as the surveys were carried out online due

Table 1: Self-assessment questionnaire.

| | |
|---|---|
| 1. | Software Development Task Value |
| 1.1 | I enjoy software development. |
| 1.2 | It's important to me to be someone being good at software development. |
| 1.3 | I am fascinated by the many possibilities that software development offers. |
| 1.4 | The exchange with my colleagues makes sense, because I can benefit from their experience. |
| 2. | Perceived Competence |
| 2.1 | I am an experienced software developer. |
| 2.2 | I can also complete difficult tasks in software development if I don't give up. |
| 2.3 | I am sure that I can master the skills taught in software development courses. |
| 3. | Achievement Goal Orientation |
| 3.1 | It is important to me that I constantly improve my skills in software development. |
| 3.2 | I want to expand my software development skills in this course. |
| 3.3 | It is important to me that I thoroughly understand the concepts discussed in the Meetup. |
| 4. | Procedural Knowledge (PK) Req. |
| 4.1 | For my work as a software developer, I need a lot of procedural knowledge (knowing-how). |
| 5. | Descriptive Knowledge (DK) Req. |
| 5.1 | For my work as a software developer, I need a lot of declarative knowledge (knowing-that). |
| 6. | Perceived Learning Success: PK |
| 6.1 | I feel like I've improved my procedural knowledge (knowing-how) in this lesson. |
| 7. | Perceived Learning Success: DK |
| 7.1 | I feel like I've improved my declarative knowledge (knowing-that) in this lesson. |
| 8. | Intrinsic Motivation |
| 8.1 | Today's session was entertaining. |
| 8.2 | I would attend such a session again. |
| 8.3 | I could imagine preparing a story in this form myself. |
| 8.4 | I have some ideas I would like to present. |
| 9. | Flow State |
| 9.1 | My attention was fully focused on the stories. |
| 9.2 | The stories were lively and captivating. |
| 9.3 | Time has passed quickly. |
| 10. | Comprehension of the method |
| 10.1 | The structure of the stories was clear and understandable. |
| 10.2 | I feel like I understood all the steps of the journey. |
| 10.3 | There were no ambiguities. |

to Covid-19 restrictions in Austria (Wolfschwenger et al., 2021). The course follows community-based learning principles, meaning that it is intended to integrate constructive community engagement with instruction and reflection to create reciprocal learning opportunities and mutually beneficial partnerships. A mix of didactic methods provide the building blocks for contributions to the course.

## 5.2 Data Analysis

### 5.2.1 Self-assessment

Figure 3 shows the average values for the logically grouped sections of the self-assessment inquiry. 73% of participants categorized themselves as experienced or very experienced software developers. Although being considered someone who is outstandingly good at software development is only moderately important, two thirds of all participants are confident that they can accomplish even the most difficult software development tasks if the necessary time and effort is put in.

The participants feel high interest, usefulness and importance of software development tasks, and it is important to them to continuously improve their development skills and understand the concepts discussed in the monthly meetings. The majority (68%) finds the exchange with colleagues very useful in order to benefit from each other's experiences and expertise.

When being asked what kind of knowledge they need for their work, the participants strongly agreed that a high level of knowledge about how to perform a specific skill or task ("know-how", procedural knowledge) is needed, knowledge of specific facts or propositions ("knowing-that", descriptive knowledge) is only moderately needed in their profession as software developers. In the associated open-ended part, it was added that quality awareness, logical thinking, creativity, depiction of complexity in a simple and readable manner, discipline and perseverance are important skills.

The posttest measured that the sessions were found enjoyable, and 72% of participants highly recommend taking part in such sessions again. The majority confirmed that they understood and could reproduce the structure of the method. The attitude towards preparing a story in the form of the Developer's Journey was positive, although the data shows a gap in terms of active contribution to the course. While about half of the listeners declared to have a variety of ideas, others were still uninspired at the time of inquiry. Participants were immersed in a moderately high feeling of energized focus and involvement during the sessions, but their attention was steadily directed towards the stories and the time passed quickly.

In terms of learning success, participants perceived an improvement in knowledge in both the procedural and declarative sphere. In particular, a greater increase was averagely seen in the declarative domain. Although the difference is only minimal, this is in contrast to the collected data on the knowledge required.

### 5.2.2 Problem-solving

In all of the three sessions, an increase in problem-solving ability could be determined from pretest to posttest. While the amount is significant in the first and second sessions, improvement is only moderate in the third session. Figure 4 illustrates average success in the competence enhancement check for each session before and after the intervention.

Looking at the individual responses in detail, it is noticeable that a lower level of motivation and flow state is related to a lower score on the problem-solving tasks. Less experienced participants also provided fewer correct answers in the tests. Since almost all participants considered the exchange with colleagues to be useful, no direct correlation between achievement goal orientation and better test results could be found.

The test items provided through the online questionnaire were similarly structured, corresponded in terms of complexity and relevance and were designed in such a way that participants were able to answer them following engagement in the sessions. The tests measured change in participants' skills needed for applying the acquired knowledge to comparable software development problem-solving tasks. The items were related to the stories presented and addressed the learning objectives that were agreed upon with the speakers in the preparation phase.

### 5.2.3 Interviews

Narrators reported that they sometimes had difficulty setting up their stories within the structure of the framework. Sequences in which essential parts of a typical software project are missing (e.g., projects without a software release or unfinished work) fit only partially into the framework.

Also, it is sometimes difficult to present the results of the work ("*The Boon*"), for example, when there are no test cases or no user interface. Still, they agreed that the framework helped them to prepare their stories in a way that was understandable and did not leave out important information.

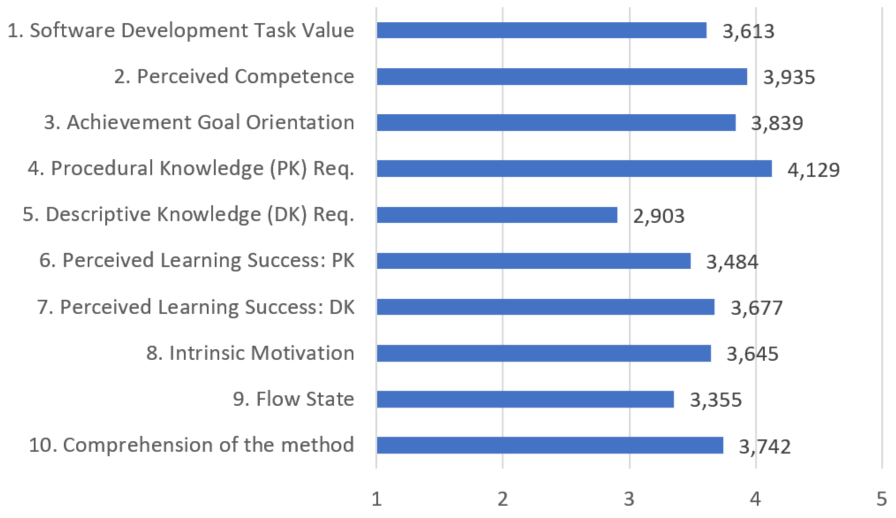


Figure 3: Arithmetic means of the self-assessment ratings (n=31).

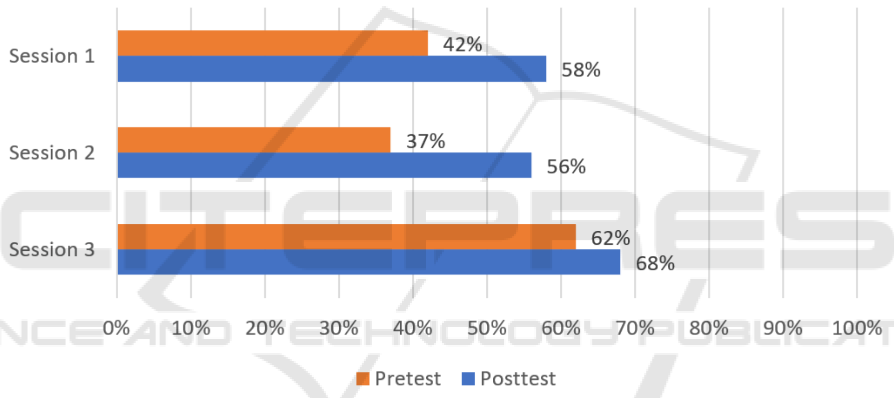


Figure 4: Average percentage of correct answers for each session in the problem-solving test (n=31).

Recipients stated that the consistent structure of the narratives made it easier for them to keep the thread in their eyes. The stories were always structured the same way, so they knew which parts contained the main information and were particularly worth listening to.

Partly, however, the unchanging structure also led to boredom. There should not be too many stories told in a row. Overall, they felt that they came to important information that they would otherwise never have learned. In informal conversations, workmates usually tend to mention only the most important aspects and don't pay attention to every detail. The Developer's Journey promoted awareness of the links between different aspects like software requirements, involved stakeholders, implementation procedures, releases and subsequent work.

## 6 DISCUSSION

The collected data indicates that the application of the Developer's Journey had a positive effect on participants' problem-solving ability. On the one hand, this is evident from the problem-solving measurement, where a significant improvement was achieved in two of three cases. On the other hand, the self-assessment questionnaire and interviews confirm that participants had the subjective impression of having enhanced their declarative and procedural knowledge.

The composition of skill levels of the experimental group for which the program was designed finds expression in the collected responses. We were dealing with a group of experienced software developers, and the averages of task value, achievement goal orientation and perceived competence are consistently in the upper range. It is likely that experienced software developers require different knowledge than beginners and the ratio of declarative and procedural knowledge growth varies from responses of unexperienced developers.

Our pretests and posttests were designed for an experienced group of software engineers with prior knowledge in application development. But also the few participants with a lower level of perceived competence were able to improve from pretest to posttest. This gives an indication that the method might be useful for groups of less experienced developers as well.

The high level of intrinsic motivation the results show implies that the participants engaged for intrinsic rewards rather than having concern for any potential external reward they might receive. It is important to rely on methods that trigger inner enthusiasm, as intrinsic motivation can affect human behavior more intensely than extrinsic motivation (Oudeyer and Kaplan, 2007). Using a method too often can become repetitive, reduce engagement and lead to a loss of motivation and adherence to the intervention (Singh, 2016). Used sparingly, the Developer's Journey provides an opportunity to mix things up and keep learners' attention.

Many studies show a connection between intrinsic motivation and flow (Rheinberg and Engeser, 2018). A prerequisite of attaining flow is voluntary engagement in a task that the individual finds enjoyable. When we perform an activity over a continuous period of time with a balance between the difficulty of the challenge and the skills of the individual, we are more likely to experience flow. The cohesive and guided process, which allows to focus on the story and make the common thread of a story visible, may support coming in this state. However, transitions and breaks between stories may disrupt the flow.

The overall high level of motivation indicates that the application of the Developer's Journey was well received by the participants and created a strong incentive for engagement. We saw improvement in problem-solving skills and also a subjective sensation of learning success. This combination makes us confident that the method has had positive effect on the participants.

## 7 CONCLUSION

We presented a storytelling concept based on NL principles for cooperative learning in software engineering. It was evaluated in the context of a post-secondary cloud computing education program for software engineers. The storytelling concept was inspired by the Hero's Journey, a common template of stories in narratology and comparative mythology and adapted as a didactic method tailored to the special needs of a software development continuous education program. It should help software developers to prepare and share their experiences in the form of instructive narratives, with the goal of activating transformative learning processes in others as well as themselves.

Many developers pursue the same objectives, follow the same development paths and solve similar problems without learning from each other's experiences. With the guidelines of the Developer's Journey, software developers should be enabled to prepare and present their lived experiences in a structured and effective manner. Sharing experiences with each other is a pillar for interconnected relations, mutual assistance and long-lasting partnerships. The evaluation of the Developer's Journey led to a positive outcome in terms of learning success and enthusiasm for the method. There is a positive correlation between the application of the framework and participants' problem-solving ability. It also seems to have a stimulating effect for the participants, which is reflected in the high level of intrinsic motivation and flow state.

In order to come to conclusions regarding the question if the flow state can be triggered by the use of this NL method, further research is needed measuring the flow state over a longer period of time. The recurring structure of the Developer's Journey provides a promising opportunity to design learning experiences so that transformative learning happens more regularly. The results indicate that a flow state could be reached.

In community-based learning, a narrative framework can be effectively used as a tool to create participatory, immersive and contextual experiences. While narrators are given a chance to structure and reflect upon overcome problems and achievements, listeners are drawn into a simulated adventure that helps them to learn from others' experiences. To facilitate the adoption and exploitation of NL in software engineering, further templates and guidelines are needed.

The Developer's Journey has proven to be a motivating and promising approach to the further development of cloud developers in community-based learning. There are still many challenges with regard to cloud adoption in the education sector (Wolfschwenger and Sabitzer, 2020). Different understandings and attitudes towards strategy, security, legal, ethical and other issues still exist. Especially in terms of awareness formation for groups concerned other than IT staff, the use of the Developer's Journey may also be a helpful tool.

## REFERENCES

Avraamidou, L. and Osborne, J. (2009). The role of narrative in communicating science. *International Journal of Science Education*, 31(12):1683–1707.

Bareiss, R. and Griss, M. (2008). A story-centered, learn-by-doing approach to software engineering education. *ACM SIGCSE Bulletin*, 40(1):221–225.

Bruner, J. (1991). The narrative construction of reality. *Critical Inquiry*, 18(1):1–21.

Campbell, J. (2008). *The hero with a thousand faces*, volume 17 of *Bollingen series*. New World Library, Novato, Calif., 3. ed. edition.

Corder, M., Horsburgh, M., and Melrose, M. (1999). Quality monitoring, innovation and transformative learning. *Journal of Further and Higher Education*, 23(1):101–108.

Dia-Eddine, K. (2021). Digital storytelling for tertiary education in the era of digitization. In Boussafi, K., Mathieu, J.-P., and Hatti, M., editors, *Social Innovation and Social Technology*, volume 162 of *Lecture Notes in Networks and Systems*, pages 16–49. Springer International Publishing, Cham.

Fisher, W. R. (1984). Narration as a human communication paradigm: The case of public moral argument. *Communication Monographs*, 51(1):1–22.

Guzdial, M. and Tew, A. E. (2006). Imagineering inauthentic legitimate peripheral participation: an instructional design approach for motivating computing education. In Anderson, R., Fincher, S. A., and Guzdial, M., editors, *Proceedings of the second international workshop on Computing education research*, pages 51–58, New York, NY. ACM.

Hazel, P. (2008). Toward a narrative pedagogy for interactive learning environments. *Interactive Learning Environments*, 16(3):199–213.

Kim, D. and Li, M. (2021). Digital storytelling: facilitating learning and identity development. *Journal of Computers in Education*, 8(1):33–61.

Kuusinen, K., Petrie, H., Fagerholm, F., and Mikkonen, T. (2016). Flow, intrinsic motivation, and developer experience in software engineering. In Sharp, H. and Hall, T., editors, *Agile processes in software engineering and extreme programming*, volume 251 of *Lecture Notes in Business Information Processing*, pages 104–117. Springer Open, Cahm, Sitzerland.

Landrum, R. E., Brakke, K., and McCarthy, M. A. (2019). The pedagogical power of storytelling. *Scholarship of Teaching and Learning in Psychology*, 5(3):247–253.

Leeming, D. A. (1998). *Mythology: The voyage of the hero*. Oxford University Press, New York, 3rd ed. edition.

McQuiggan, S. W., Rowe, J. P., Lee, S., and Lester, J. C. (2008). Story-based learning: The impact of narrative on learning experiences and outcomes. In Woolf, B. P., editor, *Intelligent tutoring systems*, volume 5091 of *Lecture Notes in Computer Science*, pages 530–539. Springer, Berlin.

Mezirow, J. (1996). Contemporary paradigms of learning. *Adult Education Quarterly*, 46(3):158–172.

Mott, B. W., McQuiggan, S. W., Lee, S., Lee, S. Y., and Lester, J. C. (2006). Narrative-centered environments for guided exploratory learning. In *Proceedings of the Agent Based Systems for Human Learning Workshop at the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (ABSHL-2006)*, Hakodate, Japan.

Niehaus, J., Romero, V., Koelle, D., Palmon, N., Bracken, B., Pfautz, J., Reilly, S. N., and Weyhrauch, P., editors (2014). *A Flexible Framework for the Creation of Narrative-Centered Tools: Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany*.

Oudeyer, P.-Y. and Kaplan, F. (2007). What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 1:6.

Rheinberg, F. and Engeser, S. (2018). Intrinsic motivation and flow. In Heckhausen, J. and Heckhausen, H., editors, *Motivation and Action*, pages 579–622. Springer International Publishing, Cham.

Singh, R. (2016). The impact of intrinsic and extrinsic motivators on employee engagement in information organizations. *Journal of Education for Library and Information Science*, 57(2):197–206.

Vivitsou, M. (2018). Digital storytelling in teaching and research. *SSRN Electronic Journal*.

Vogler, C. (2020). *The writer's journey: Mythic structure for writers*. Michael Wiese Productions, Studio City, CA, 4 edition.

Wolfschwenger, P., Albaner, B., Kastner-Hauler, O., and Sabitzer, B. (2020). The value of cloud-based learning environments for digital education. In *2020 IEEE Frontiers in Education Conference (FIE)*, pages 1–5, Piscataway, NJ. IEEE.

Wolfschwenger, P., Hinterplattner, S., Demarle-Meusel, H., Albaner, B., and Sabitzer, B. (2021). Learning under lockdown: The conditions in austria in a global context. In Csapó, B. and Uhomoibhi, J., editors, *Proceedings of the 13th International Conference on Computer Supported Education*, pages 648–656, Setúbal, Portugal. SCITEPRESS - Science and Technology Publications Lda.

Wolfschwenger, P. and Sabitzer, B. (2020). Effective integration of cloud services into educational organizations. In *Proceedings of EdMedia + Innovate Learning 2020*, pages 132–137, Waynesville, NC. Association for the Advancement of Computing in Education (AACE).