ISSN 2186-7437

NII Shonan Meeting Report

No. 192

Augmented Software Visualization

Leonel Merino
Craig Anslow
Takashi Ishio
Alexandre Bergel

November 27–30, 2023

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

Augmented Software Visualization

Organizers:
Leonel Merino, Pontificia Universidad Católica de Chile, Chile
Craig Anslow, Victoria University of Wellington, New Zealand
Takashi Ishio, Future University Hakodate, Japan
Alexandre Bergel, RelationalAI, USA

November 27–30, 2023

Abstract

Visualization is "the use of computer-supported, interactive, visual representations of abstract data in order to amplify cognition." Software visualization is the use of interactive and visual representations of software data to support software development activities, as well as to explore systems with respect to the aspects of structure, behavior, and evolution [6]. In the last two decades, multiple studies have been published across main software engineering journals (e.g., TSE, IST, JSS, JSEP) and conferences (e.g., ICSE, FSE, ICSME, ASE, MSR), confirming software visualization as a relevant topic in software engineering research. Today, the field of software visualization is at the junction of two singular situations: (1) modern software systems are developed involving multiple sophisticated techniques and exhibit an increasing complexity, and (2) a variety of sensory technologies, such as Augmented Reality and Virtual Reality (AR/VR), eye-tracking, physiological sensors, are increasingly more available to both researchers and practitioners. These sensory technologies do not only represent opportunities to investigate visualizations suitable for complex systems but also to evaluate the impact of such visualizations on human cognition. We consider this Shonan meeting to be a unique and extraordinary tool for reflecting on these challenges. That is, the goal of the Shonan meeting is to reflect on the following challenges of software visualization: sensory augmentation, cognitive methods, and software complexity.

We collect the results of the discussions as recommendations for using sensory augmentation to support software development activities. In particular, we elaborate on the strengths and weaknesses of technologies, application domains, and implications for user evaluations. Finally, we discuss the possibilities of using cognitive methods in user evaluations. Concretely, we (1) identify the cognitive aspects that can play a role in user evaluations, (2) describe available data collection methods suitable for the analysis of those aspects, and (3) elaborate on how researchers can use these methods in practice. We also describe research opportunities in using visualization to deal with software complexity. Specifically, we outline project ideas of various sizes (e.g., paper project ideas, bachelor and master project ideas, and ideas for larger projects) that could be eventually funded and then realized by the community in the future.

Executive Summary

In the meeting, we organized the discussions around three main topics: sensory augmentation, cognitive methods, and software complexity.

Sensory Augmentation. In a rapidly evolving technological landscape, the traditional paradigms of software development are being reshaped by the emergence of innovative sensory augmentation tools. While developers have long relied on familiar interfaces like windows, icons, menus, and pointers (WIMP), the advent of advanced devices such as Augmented Reality and Virtual Reality (AR/VR) technology, alongside sophisticated physiological tracking capabilities embedded within everyday devices like smartphones and laptops, is revolutionizing how we perceive and interact with software visualizations. This evolution raises a crucial question. What is the true potential of sensory augmentation in enhancing our comprehension of complex software systems? By embracing these cutting-edge tools, we unlock new dimensions of understanding, empowering developers to explore novel perspectives and approaches in software development. Sensory augmentation can be an opportunity to achieve exceptional levels of understanding and productivity in software development processes, ranging from immersive AR/VR environments to detailed findings derived from physiological data. As we embark on this transformative journey, the integration of sensory augmentation promises to redefine the boundaries of possibility, catalyzing innovation, and propelling the field of software development.

Cognitive Methods. Within the field of software engineering, user assessments have typically concentrated on performance measures such as accuracy and completion time (i.e. time and errors). These metrics play a crucial role in determining task efficiency. However, this narrow scope has inadvertently overlooked a plethora of crucial cognitive aspects integral to user experience. While accuracy and completion time provide valuable insights into decision-making and problem-solving processes, they represent only a fraction of the cognitive landscape. Key facets such as cognitive load, learnability, emotional response, memory retention, and attention allocation have been largely marginalized in evaluation methodologies, despite their profound impact on user engagement and comprehension. Recognizing the interconnected nature of cognitive processes, it becomes evident that a holistic approach to software visualization evaluation is imperative for capturing the full spectrum of user experience [14]. Hence, the pivotal question arises: How can we integrate cognitive methods into software visualization evaluations to transcend conventional performance metrics and attain a comprehensive understanding of user interaction? By embracing cognitive methodologies, we not only refine our assessment criteria but also unlock deeper insights into the intricate dynamics between users and software visualizations. This paradigm shift towards a more inclusive evaluation framework promises to elevate the efficacy and relevance of software engineering practices, ushering in a new era of user-centric design and development.

Software Complexity. In the constantly changing world of technology, software systems have grown in complexity to levels never seen before, posing new challenges for humanity. The amalgamation of advancements in hardware, coupled with widespread accessibility to software, has propelled these systems into intricate and multifaceted entities. Today, software is crafted in a myriad of programming languages and deployed across diverse platforms, amplifying the intricacy of the technological ecosystem. In this milieu, the task of developing

software visualizations capable of effectively navigating this complexity poses a formidable challenge. The pressing inquiry emerges: How can visualizations rise to the occasion and effectively grapple with the escalating intricacies inherent in modern software systems? Addressing this query necessitates innovative approaches that transcend conventional paradigms, harnessing cutting-edge techniques to distill complex information into intuitive and actionable insights. From advanced data visualization methodologies to machine learning-driven analytics, the quest for solutions demands interdisciplinary collaboration and a relentless pursuit of novel strategies. By embracing this challenge with creativity and ingenuity, we can empower software visualizations to not only adapt to but also thrive amidst the burgeoning complexity of software systems, thereby charting a course towards a more comprehensible and navigable technological landscape.

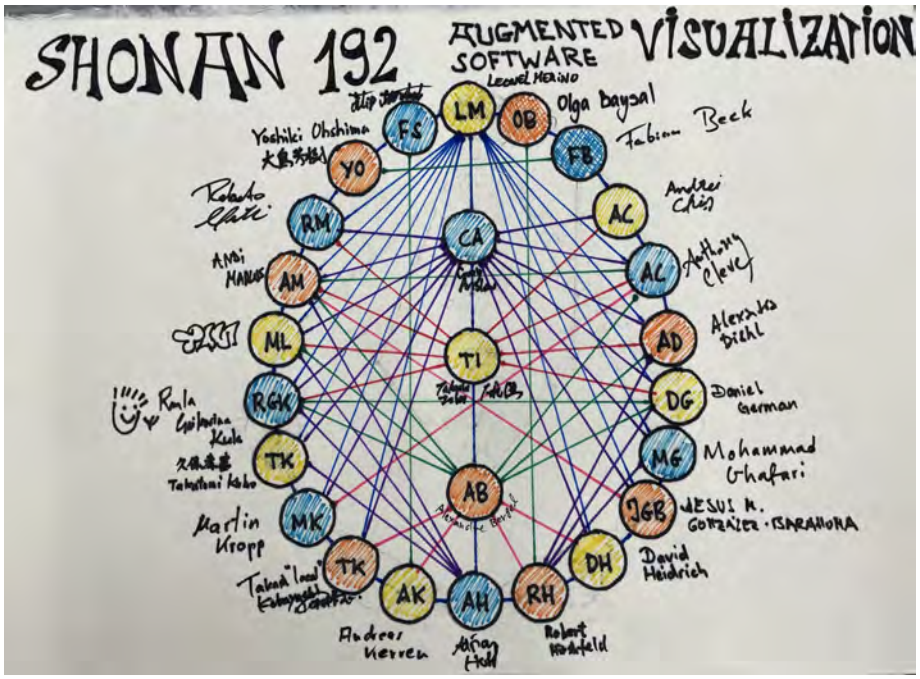

Figure 1: The poster of NII Shonan Meeting No.192 on "Augmented Software Visualization".

Overview of Talks

I. Showcases of Human-centered Visualization Research Relevant for Augmented SoftVis

Andreas Kerren, Linköping University

My talk will overview our research in information visualization, human-computer interaction, and immersive analytics which may be relevant for reaching the goals of this Shonan meeting. I will exemplify solutions that my research groups at Linnaeus and Linköping Universities have developed over the

past years. They cover areas such as explainable AI, Vis4ML, technologies for emotion-enhanced interaction, and collaborative immersive analytics systems. All together I hope that this talk can serve as a starting point for further discussions and the identification of a research agenda in augmented software visualization.

II. Babylonian-style Programming: Example-based, Live, Exploratory

Robert Hirschfeld, Hasso Plattner Institute, University of Potsdam

List of Participants

- Olga Baysal, Carleton University, Canada
- Fabian Beck, University of Bamberg, Germany
- Andrei Chis, Feenk (Bern), Switzerland
- Anthony Cleve, University of Namur, Belgium
- Alexandra Diehl, University of Zurich, Switzerland
- Daniel German, University of Victoria, Canada
- Mohammad Ghafari, Technical University of Clausthal, Germany
- Jesús González-Barahona, Universidad Rey Juan Carlos, Spain
- David Heidrich, DLR, Germany
- Robert Hirschfeld, Hasso Plattner Institute, University of Potsdam, Germany
- Adrian Hoff, IT University of Copenhagen, Denmark
- Andreas Kerren, Linköping University, Sweden
- Takashi Kobayashi, Tokyo Institute of Technology, Japan
- Martin Kropp, University of Applied Sciences and Arts Northwestern Switzerland, Switzerland
- Takatomi Kubo, Nara Institute of Science and Technology, Japan
- Raula Gaikovina Kula, Nara Institute of Science and Technology, Japan
- Michele Lanza, Software Institute – USI, Lugano, Switzerland
- Andrian Marcus, George Mason University, USA
- Roberto Minelli, Software Institute – USI, Lugano, Switzerland
- Yoshiki Ohshima, Croquet Corporation, Japan
- Filip Strömbäck, Linköping University, Sweden

Figure 2: The participants of NII Shonan Meeting No.192 on “Augmeting Software Visualization”.

Meeting Schedule

Check-in Day: November 26 (Sun)

- Welcome Banquet

Day1: November 27 (Mon)

- Welcome
- Keynote: "Showcases of Human-centered Visualization Research Relevant for Augmented SoftVis", Andreas Kerren
- Lightning Talks
- Group Photo Shoot
- Breakout Session 1
- Breakout Session 2
- Groups Presentations

Day2: November 28 (Tue)

- Program briefing
- Keynote: "Example-based Live, Exploratory Programming", Robert Hirschfeld
- Breakout Session 3
- Breakout Session 4
- Group Presentations
- Informal Demo Session

Day3: November 29 (Wed)

- Program briefing
- Breakout Session 5
- Breakout Session 6
- Excursion: Jomyoji and Hokokuji Temple
- Main Banquet

Day4: November 30 (Thu)

- Program briefing
- Breakout Session 7
- Conclusion/ Wrap-up

1 Plenary Discussion

The meeting started with a plenary session that identifies common problems that should be discussed in the software visualization community. We identified twelve problems as follows.

1. Large scale organizations (communication / requirements)
2. Private data preservation
3. Precision of data / software / languages / (reification)
4. Complexity of data / software / languages/ execution
5. Visualization paradigm / multi-scale
6. Expertise of people (developers and managers), knowledge
7. Design of software visualizations for different domains
8. Data collection of software design and user interaction
9. Live visualization feedback
10. Build tools for adoption
11. History of actions of making changes / provenance
12. Evaluation of tools

After the plenary session, participants formed three groups to discuss three topics: Sensory Augmentation, Cognitive Methods, and Software Complexity. Each participant freely chose one of the groups for each breakout session.

The following sections summarize the discussion results of the groups.

2 Sensory Augmentation

Alexandra Diehl, Adrian Hoff, Andreas Kerren, Andrian Marcus, **Craig Anslow**, Martin Kropp, Michele Lanza, and Takashi Kobayashi

2.1 Motivation

Our motivation is to provide a comprehensive overview of the currently available input and output (I/O) sensory technologies and to analyze their potential contributions to augmenting software visualization and enhancing the productivity of software development processes, particularly in terms of team collaboration. By investigating the strengths and weaknesses of various I/O devices, our objective is to understand how these technologies can be effectively applied across different domains and scenarios to streamline workflows, facilitate better communication among team members, and improve overall project results. Specifically, we examine how advanced I/O devices, such as eye-tracking systems [22], haptic feedback mechanisms [5], and immersive VR/AR interfaces [16], can be leveraged to create more interactive and intuitive software development environments. These technologies not only can improve the visualization of complex

software systems [21], but can also offer real-time feedback and adaptive interfaces that can dynamically adjust to user interactions, thus promoting more efficient and effective collaboration [8]. Our analysis includes case studies and practical examples to illustrate the diverse applications and benefits of integrating these I/O sensory technologies into software development practices. Our primary focus is on efficiency, collaboration, learning, and accessibility. We believe that these facets of software development can be greatly improved through the use of sensory augmentation.

Efficiency. The incorporation of advanced sensory technologies can improve developers' productivity by reducing the number of bugs, facilitating faster coding, and facilitating navigation throughout the development process. Sensory augmentation, such as haptic feedback, eye-tracking, and immersive AR/VR interfaces, provides real-time, intuitive interactions that streamline workflows and improve code comprehension. These technologies enable developers to identify and fix errors more quickly, enhance their coding speed by offering more responsive and context-aware tools, and navigate complex codebases with greater ease. Consequently, sensory augmentation leads to a more efficient and effective software development environment, ultimately boosting overall productivity.

Collaboration. The integration of advanced sensory technologies can enhance communication and support interdisciplinary collaboration, enabling activities like mob programming and effective feedback cycles. Sensory augmentation tools, such as real-time visual and auditory feedback systems, immersive AR/VR environments, and intuitive gesture-based interfaces, provide dynamic and responsive communication channels. These technologies help bridge gaps between team members from different disciplines, promote engagement, and ensure that feedback is seamlessly integrated into the development process. Consequently, sensory augmentation fosters a more collaborative, interactive, and productive teamwork environment.

Learning. The use of advanced sensory technologies, such as immersive game-like environments or 3D simulations, can greatly enrich the educational experience. Sensory augmentation provides interactive and engaging platforms where learners can visualize complex concepts in three dimensions and interact with educational content in real-time. This approach not only aids in comprehension and retention of information but also enables instructors to provide more effective guidance and feedback. These technologies create dynamic and immersive learning environments that cater to various learning styles, making education more effective and enjoyable.

Accessibility. Accessibility in software visualization can be greatly enhanced through sensory augmentation by supporting multi-channel interaction, making the technology more inclusive, versatile, and engaging. By integrating sensory technologies such as voice recognition, eye tracking, and screen readers with Braille displays, users with varying abilities can interact with software more effectively. These technologies provide alternative input and output methods that cater to different needs, ensuring that everyone can access and benefit from software visualization tools. Sensory augmentation thus creates a more inclusive environment, enabling diverse user groups to engage with and contribute to software development processes.

2.2 Outcomes

In our group discussions, using our collective expertise in software engineering, human-computer interaction (HCI), and emerging technologies, we identified key functionalities and properties of devices that are crucial to optimal performance and user experience. The functionalities and properties identified resulting from sensory technology / IO are listed in Tables 1 and 2, respectively. .

Table 1: Functionalities of sensory I/O devices.

Functionality	Description
Collaboration	Facilitates working together on tasks or projects, often in real-time
Context	Provides relevant information based on the user's current environment or situation
Coordination	Helps in organizing and aligning actions or schedules among team members
Fast Input	Allows for rapid data entry or command input to enhance user efficiency
Feedback to Users	Offers responses to user actions to confirm that the system has received and processed their input
Interaction History	Maintains a record of past interactions to improve user experience and personalization
Orientation and Navigation	Helps users determine their position and guides them to their desired destinations
Scale	TBD
Search	Enables users to find specific information or resources quickly and efficiently
Team Awareness	Provides insights into team members' activities, status, and progress
Team Communication	Supports the exchange of information and messages among team members
Visual Guidance	Uses visual cues to direct user attention and actions effectively

Together, we gathered to brainstorm the different sensory devices available. During our discussion of their characteristics, we compiled a list of properties and functionalities common to sensory I/O devices. We evaluated each one individually, assigning a ranking on a 3-point scale (1, 3, and 5). If a feature was not present, we assigned it a zero. The chosen value reflects our collective opinion on their impact, relevance, and application (a higher value indicates a more significant aspect). Once we agreed on a value, we proceeded to the next device. The results of the analysis of functionalities offered by sensory I/O devices is presented in Table 3. In summary, we note that features like fast input, scalability, and visual guidance are effectively supported by various modalities. However, most modalities offer minimal assistance with coordination and user feedback. This is an exception for XR and haptics, which provide significant support in these areas. When discussing the identified functionalities, we formulated the following research questions:

Table 2: Properties of sensory I/O devices.

Property	Description
Availability	The readiness of the device or system to be used when needed
Precision and Accuracy	The degree to which a device can measure or perform its intended function with minimal error
Scale	The extent to which a device or system can accommodate growth in terms of users, data, or complexity
Security and Privacy	Measures and protocols in place to protect user data and ensure secure interactions
User adoption	The rate at which users embrace and consistently utilize the device or system
User experience	The overall satisfaction and usability experienced by users when interacting with the device or system

1. What specific advancements in eye-tracking technology could enhance its precision and accuracy for detailed code analysis and debugging?
2. In what ways can haptic feedback systems be enhanced to deliver more refined and context-sensitive tactile responses in coding tools?
3. What strategies can be employed to ensure the scalability of immersive VR/AR interfaces in large-scale software development projects without compromising performance?

Table 3: Assessment of sensory I/O device functionalities.

Functionality	BCI	Bio.	Body Track.	Eye Track.	Tangibles	Haptics	Voice Rec.	XR
Collaboration	1	1	3	3	3	3	5	3
Context	1	1	3	5	3	5	5	5
Coordination	0	0	1	1	3	3	3	5
Fast input	3	5	3	5	5	5	5	5
Interaction History	1	1	3	5	3	5	3	5
Orientation	1	1	3	3	1	5	3	3
Scalability	5	3	3	5	3	3	1	5
Search	1	1	1	1	1	3	3	3
Team awareness	3	3	3	3	3	5	3	3
Team Communication	1	1	3	0	5	5	3	5
User Feedback	0	0	0	0	0	5	0	5
Visual guidance	3	3	3	5	5	5	5	5

Table 4 shows the evaluation outcomes of the sensory I/O devices' properties. It is noted that most modalities exhibit low to moderate levels of precision and accuracy, limiting their usefulness for more critical tasks. Except for voice recognition, all other modalities scale well, which is particularly important for software systems. Biometrics and body tracking are identified as requiring significant improvements in user privacy protection. Additionally, there's a clear link between user experience and the adoption rate, with better experiences

driving higher adoption. Consequently, we reflect the need for addressing the following research questions:

1. In what ways can real-time visual guidance provided by sensory I/O devices be optimized to support developers in navigating complex codebases more effectively?
2. How can sensory augmentation technologies be tailored to improve coordination and communication among interdisciplinary teams during collaborative programming sessions?
3. What role can adaptive interfaces play in enhancing user experience and adoption rates of sensory I/O technologies in software development environments?

Table 4: Assessment of sensory I/O device properties.

Property	**BCI**	**Bio.**	**Body Track.**	**Eye Track.**	**Tangibles**	**Haptics**	**Voice**	**Rec.**	**XR**
Availability	3	5	3	3	1	5	5		5
Precision & Accuracy	1	3	1	3	3	3	1		3
Scalability	5	3	3	5	3	3	1		5
Security & Privacy	3	1	1	3	5	3	3		3
User adoption	1	5	3	1	1	5	5		3
User experience	1	5	1	1	3	5	3		3

In conclusion, the integration of advanced I/O sensory technologies into software development holds significant promise to improve productivity, collaboration, learning, and accessibility. Using tools such as eye tracking systems, haptic feedback, and immersive VR/AR interfaces, developers can achieve more intuitive and interactive environments that streamline workflows and improve communication. The key results of our analysis indicate that features such as fast input, scalability, and visual guidance are effectively supported by various modalities, while coordination and user feedback are particularly well supported by XR and haptic technologies. However, many modalities exhibit low to moderate precision and accuracy, which limits their utility for critical tasks, and improvements are needed in user privacy protection for biometrics and body tracking. As these technologies continue to evolve, their adoption is likely to drive more efficient, effective, and inclusive software development practices, ultimately transforming how teams collaborate and innovate.

3 Cognitive Methods

David Heidrich, Daniel German, **Leonel Merino**, Olga Baysal, Raula Gaikovina Kula, Robert Hirschfeld, and Takatomi Kubo.

3.1 Motivation

Evaluations in software engineering have mainly concentrated on assessing user performance, typically by measuring task accuracy and completion time. Consequently, software visualization assessments have focused mainly on user performance analysis. Although metrics such as accuracy and completion time shed

light on cognitive aspects such as decision-making and problem-solving, there are numerous other cognitive factors that have been largely ignored in these assessments. Cognition, defined as "the mental action or process of acquiring knowledge and understanding through thought, experience, and the senses" [1], plays a crucial role in user evaluations. Various aspects of human cognition can be considered in user evaluations to provide researchers with a more thorough understanding of the effects of their methods. Examining these aspects can facilitate the comparison of results across different technologies. Although studies typically concentrate on a few cognitive aspects to keep their analyses manageable, comprehending multiple cognitive aspects simultaneously can help in developing theories that elucidate complex human factors phenomena. Moreover, due to the interconnected nature of cognitive aspects, it is crucial to integrate them into user evaluations, as high accuracy and low completion time may be influenced by various cognitive factors. Thus, the question arises: How can cognitive methods be incorporated into software visualization evaluations?

3.2 Outcomes

We start with the definition of cognition and then identify the main aspects of cognition that can play a role in software visualization. Next, we discuss the possibilities of embedding cognitive measurements into the IDE. We ended the discussion by listing research ideas for future projects to advance knowledge in this domain. Figure 3 illustrates a diagram during our dynamic discussion.

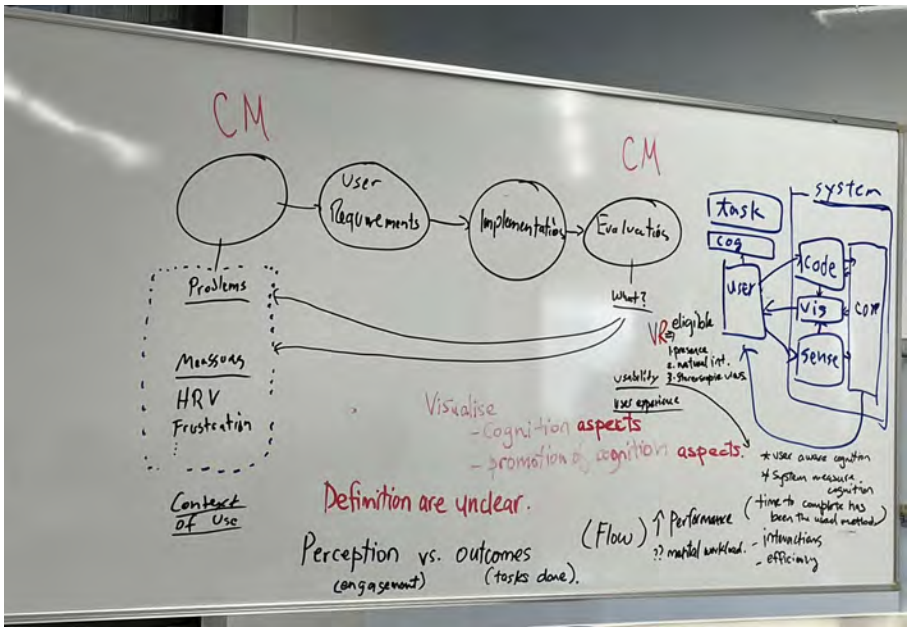

Figure 3: Day 2. Cognitive methods pipeline.

3.2.1 Cognition plays a fundamental role in visualization

Visualization is the use of computer-supported visual and interactive representations of abstract data to amplify cognition [4]. Therefore, understanding the role cognition plays in visualization is fundamental. A previous study [17] identified ten cognitive aspects analyzed in the evaluation of immersive technologies. Cognitive aspects were collected using evaluation methods as a proxy to identify aspects of human cognition involved in user evaluations. For instance, in evaluations employing the Self-Assessment Manikin [3] method, it can be inferred that researchers are exploring emotions. The list of identified cognitive aspects is presented in Table 5

Table 5: Cognitive aspects.

Aspect	Description
Attention	The process of selectively concentrating on an aspect while ignoring other information
Cognitive load	Relates to the mental load imposed by instructional parameters, e.g., task structure, the sequence of information given during an evaluation; and mental effort that refers to the capacity allocated by participants of a study to the instructional demands
Decision making	The process of identifying and choosing from several alternative possibilities
Emotion	A mental state relating to thoughts, feelings, behavior, and affects
Learnability	Capability of a system to enable users to learn how to use it, usually considered as an aspect of usability
Memory	Relates to the ability to encode, store, and retrieve information when needed
Motion sickness	A disturbance of the senses due to a difference between actual and expected motion
Perception	Relates to the interpretation of sensory information (e.g., visual, auditory, or haptic) to understand information of the environment
Presence	he feeling of having no mediation between oneself and the (virtual) environment, which promotes the psychological sensation of "being there"
Usability	The ease of use

We discuss these various cognitive aspects that we observe can have a great impact on the effectiveness of visualizations, and find that attention, emotion, learnability, memory, and perception are of particular interest for augmented software visualizations. Methods that are pertinent for evaluating these cognitive aspects include eye-tracking [25, 19, 12] for attention, the Positive and Negative Affect Schedule [24] for emotion, pre- and post-tests [2, 27] for learnability, cue and free recall [20] for memory, and the Two-Interval Forced-Choice method [26] for perception.

3.2.2 How embedding cognitive measurements in the IDE?

We observe that the use of biometric measures to assess cognitive aspects and the measurement and analysis of biometric data can be challenging. We notice the importance of frustration in the learning process and identify both strengths and areas for improvement in the visualization design process. It can be difficult to determine which cognitive aspects are most relevant for visualization. We also discussed how visualization can improve cognition and the need to define and quantify cognitive aspects. Moreover, we notice that the use of immersive experiences can boost the effectiveness and, eventually, the efficiency of visualizations. In particular, in the context of software engineering, efficiency is one of the most important aspects, as developers would like visualization to complement their daily tasks. In concrete, we identify three main biometric sources that could help embedding cognitive measures in the IDE.

EEG Biosensors that can be used to interpret brain activity. For that, the analysis of the EEG signal and delta, theta, low apha, high alpha, low beta, high beta and gamma waves can be key to the identification aspects of attention, calmness, mental effort, emotions, blink detection, and creativity. Such sensors can be used to assess cognitive load. Other existing physiological measures that have been examined to analyze, in particular, mental effort are pupil dilation [13], heart rate variability [18], event-related brain potentials [7], muscle tension [23], adrenaline level [9], skin temperature and galvanic skin response [10].

ECG Biosensors electrocardiograms are used to measure heart rate, temperature, and breathing, which can lead to the identification of symptoms of stress, fatigue, and particular moods. Objective data collection methods such as the electrocardiogram, the galvanic skin response, and the skin temperature [10] can help assess various aspects of presence, e.g., co-presence, spatial presence, social presence, social richness, closeness, or connectedness, which is a central characteristic of virtual and augmented reality environments.

Eye-tracking can help the analysis of gazes, fixation, areas of interest to evaluate attention, perception, and engagement. For instance, eye-tracking can be used to analyze when participants are distracted. Moreover, eye-tracking complemented by an analysis of head movements (e.g., orientation angle of participants heads) indicated when participants were distracted as well. Visual attention can also be analyzed based on subjective methods that consider, for instance, the assessment of engagement, through tailored questionnaires [11, 15].

3.2.3 Future Research Topics

Based on our discussion, we identified three research topics that can benefit from the incorporation of cognitive methods to improve software visualizations.

1. Opportunities for Enhanced Immersive Experience in Software Engineering. We notice the need for investigating the potential of immersive experiences in software development environments. To this end, we propose to address key research questions regarding opportunities, task identification,

developer tasks suitability, cognitive aspects, and adoption challenges. Consequently, we formulate the following questions:

1. What are the potential advantages that arise from the immersive experience?
2. How can we determine which tasks are suitable for the immersive experience? Should we transfer and modify existing tasks or find new ones?
3. Which developer tasks are suitable for an immersive experience? Is it necessary or desirable to have only one immersive experience for all development tasks? Can tasks such as code review, exploration, bug reviews, and authoring be done in a virtual or augmented development reality?
4. How significant are the various cognitive aspects in relation to immersive experience?
5. How can we implement these experiences in the developer workspace? What are the potential risks and benefits?

The objective of this project is to discover fresh possibilities and obstacles in the adoption of immersive experiences. It aims to provide recommendations for choosing tasks that are appropriate for immersive development environments. Additionally, it seeks to offer understanding into the cognitive factors that impact immersive development. The project intends to develop strategies for incorporating immersive experiences into developer workspaces.

This project can contribute to improving the productivity and innovation of developers by using immersive technologies. The findings would inform the design of immersive development tools and practices, shaping the future of software engineering.

2. Pitfalls and Recommendations for Visualization Evaluation. We also observe the opportunity to explore the impact of immersion on software engineering (SE) using insights from user studies. Address confounding factors, benefits, cognitive measurements, and mitigation strategies of interruptions. We formulate the following questions:

1. How do different levels of immersion impact developer productivity and well-being in software engineering tasks?
2. What are the primary confounders that influence the effectiveness of immersive experiences in software development environments?
3. What strategies can effectively mitigate interruptions such as motion sickness, fatigue, and battery power constraints during immersive software engineering experiences?

This research project aims to provide a comprehensive understanding of immersion's impact on software engineering (SE), addressing both its benefits and challenges. It will identify and analyze the confounding factors that influence immersion, shedding light on the complexities involved in creating immersive experiences for developers. In addition, the project will evaluate various cognitive measurement methods to assess their efficacy in capturing the cognitive

aspects of immersion in SE tasks. In addition, it will focus on developing strategies to mitigate interruptions during immersive experiences, considering factors such as motion sickness, fatigue, and battery power limitations. Finally, based on the findings, the project will provide recommendations for optimizing tool configurations to enhance the immersive SE environment, ultimately improving developer productivity and well-being in software development processes.

This research will improve SE practices by harnessing the benefits of immersion while addressing its challenges. The findings will inform the design of immersive tools and techniques, improving the productivity and well-being of the developers.

3. Guidelines for Software Engineering Cognition for Immersion. In this project, we plan to investigate cognition in SE through various methods, focusing on the contributions of quantitative and qualitative measures. Examine the factors that distinguish the evaluation of SE cognition from traditional methods, focusing on usability and user experience considerations. To this end, we define the following questions:

1. What are the current methods and their applications in studying cognition in SE, particularly regarding quantitative and qualitative measures?
2. How do the unique factors of SE cognition evaluation differ from traditional methods and how can usability and user experience be effectively integrated?
3. How can a combination of user feedback, biometrics, and tool metrics be used to gain insight into cognitive processes in SE, considering associated risks and benefits?

The expected results of this research endeavor encompass a multifaceted advancement in understanding cognition within the domain of Software Engineering (SE). Through an examination of quantitative and qualitative measures, the project aims to delineate their applicability, thereby providing information on effective methodologies for studying cognitive processes in SE contexts. Moreover, by identifying factors that differentiate SE cognition evaluation from traditional methods and emphasizing the integration of usability and user experience considerations, this research will contribute valuable guidelines for optimizing cognitive research practices in SE. Ultimately, the culmination of this work is expected to offer tangible recommendations for utilizing a combination of user feedback, biometrics, and tool metrics to enrich cognitive research methodologies within the SE domain, fostering innovation and efficiency in software development practices.

This research will advance our understanding of cognition in SE, providing methodological insight and practical recommendations to improve cognitive research methodologies in the field.

4. Enhancing the Use of Biometric Measures for Cognitive Assessment in Visualization Design We observe the need to investigate the application of biometric measures in the evaluation of cognitive aspects of visualization design, focusing on reliability, feasibility, and analysis challenges. Address

the need for education and focus on mental content while leveraging community strengths to bridge design gaps.

1. What biometric measures are the most reliable and feasible for evaluating cognitive aspects such as frustration, mental effort, and emotion in the visualization design?

2. How can challenges in accurately evaluating the autonomic nervous system be mitigated and how should the inclusion of vast mental contexts be limited?

3. What strategies can be used to effectively define cognitive aspects for visualization design, considering measurement challenges and the importance of education and mentoring?

The expected results of this project are multifaceted, with the aim of significantly enhancing the utilization of biometric measures for cognitive assessment in visualization design. First, through systematic investigation, the research will identify the most reliable and feasible biometric measures to assess cognitive aspects such as frustration, mental effort, and emotion. Additionally, the project seeks to address challenges related to accurately evaluating the autonomic nervous system and managing large mental contexts, providing valuable information about mitigating these obstacles. Furthermore, by emphasizing the importance of education and the focus on mental content, research will contribute to fostering a deeper understanding of how to utilize biometric measures effectively in visualization design. Lastly, the project aims to propose strategies for defining cognitive aspects in visualization design, considering measurement challenges and the role of education and mentoring, ultimately advancing the field by providing practical guidance for improving visualization design methodologies.

This project aims to improve visualization design by using biometric measures for cognitive assessment. By identifying reliable measures, overcoming obstacles, and prioritizing education, this project has the potential to enhance design choices and user experiences in various industries that rely heavily on efficient visualization tools. Ultimately, this can stimulate innovation and improve overall efficiency.

4 Software Complexity

Andrei Chis, Anthony Cleve, Fabian Beck, Filip Strömbäck, Jesús González-Barahona, Roberto Minelli, **Takashi Ishio**, Yoshiki Ohshima, Mohammad Ghafari

4.1 Motivation

Contemporary software systems are large and complex. To maintain those artifacts or make some decisions, software developers and managers would like to understand a particular view of a system for each task, without understanding the complex details irrelevant to the task. To overcome software complexity, software visualization techniques have been developed in the community.

The group discussed that there are several perspectives that developers need to focus on: educational, professional, performance, security, and privacy. We have to take care of those perspectives, while we cannot always think them through in detail. Visualizations have been separately proposed to visualize one of the aspects. However, they have not been integrated into a single environment.

4.2 Outcomes

As software visualizations are expected to be a part of software development environment, the Software Complexity group had discussions to identify characteristics that future IDEs should have. We listed ideas and then classified them into three groups.

1. IDEs should support various domains and demands.
2. IDEs should support smooth interaction.
3. IDEs should support artifact maintenance tasks.

The following subsections describe each of the characteristics in detail.

4.2.1 IDEs should support various domains and demands.

Software products are different depending on domains. However, software developers use the same IDE for different domains in general. Of course, general IDEs have been developed, but when needed, we should be able to customize or configure an IDE for a particular domain.

Domain-specific IDEs enable developers to manipulate domain-specific concepts, i.e., higher level abstractions than source code. For example, the node editor for Unreal Engine is a domain-specific IDE for interactive 3D graphics. MATLAB/Simulink is another example for mathematical models. While those IDEs are considered effective, making an IDE for each domain may be unrealistic due to the development cost. A possible direction is a meta IDE to create domain-specific IDEs.

While the current visualization techniques visualize structure, behavior, and metrics, properties like a certain level of security and privacy are not visualized. To enable developers to directly address such cross-cutting concerns, creating new domain-specific languages is another future direction.

4.2.2 IDEs should support smooth interaction.

Software developers have various tasks. Software visualizations should support not only individuals but also teams performing collaborative tasks. Software visualizations should also scale for large systems.

Investigation of multi-modal interfaces, including natural language text, speech recognition, eye tracking, AR, and VR, is an important direction. Those interfaces may change how software developers interact with an IDE and how the IDE visualizes information. This direction would require many exploratory studies, because the effectiveness of such interfaces is unclear. An interesting challenge is providing explanation (interpretation and diagnosis) along visualizations.

Our discussion listed several questions as follows.

- Does VR contribute new possibilities (e.g., for overviews) compared to 2D screens?
- Can natural language help?
- Can speech recognition etc. be used to provide new interactions together with VR?
- What could be an explanation?

4.2.3 IDEs should support artifact maintenance tasks.

Software developers produce many artifacts for a project including source code, requirements specification, design diagrams, bug reports, code review comments, and so on. While it is difficult to provide a clear definition, anything that developers continue to use during their project are included in artifacts. For example, database schema is an artifact, while data in a database owned by users is not an artifact. This tentative definition may be extended in the future.

Due to software complexity, software developers may have a huge number of artifacts. As software visualization techniques present a particular aspect of artifacts in a view, an interesting question has been posed in the discussion: *Can we program by visualization?*

Supporting such artifact maintenance would be possible if we have an *editable view* that automatically translates edit operations on a visualization into corresponding edit operations on the artifacts behind the visualization. The concept of such editable views is not quite new; for instance, editable views for database tables are available in recent database management systems. It is also known as bidirectional transformations.

A challenge for editable views is how to translate edit operations into artifacts because it is not obvious for abstract visualizations. Developing new visual languages to (partially) manipulate source code would be an interesting direction.

It should be noted that editable views are not always important. For example, software developers do not usually edit performance analysis reports visualized by profiling tools. Visualization developers should investigate effective use case scenarios of editable views. The group also listed some questions on use cases that IDEs should provide visualizations for:

- We can query data on a database, but we search code textually. How do we want to query and visualize our code?
- In architecture, we can view drawings at different levels of detail. How can the IDE help us to do this?
- Consider this scenario: “I have found a bug in a large codebase. How can the IDE help me find the location of the source code I should start looking at?”

Finally, bi-directional links are related to live visualization feedback. Some challenges were identified in the discussion.

- Visualization combining static and dynamic information (perhaps with characteristics based on behavior of "production system").
- Incremental visualization that is automatically updated for incrementally updated code and data.
- Visualization scaling for small systems, and up to large or even distributed systems (perhaps using simulations/predictions to help with handling scale of software quickly).

Conclusion

In this 4-day seminar, we discuss three main topics to enhance software visualization: sensory augmentation, cognitive methods, and software complexity.

Sensory technology. We review sensory technology and analyze how it can contribute to improving software visualization. We concluded that such technologies can help guide developers as they navigate a software system. They can be used for searching and as input methods, in general, for instance, to navigate history in a version control system. Integrating advanced I/O sensory technologies into software development shows great potential to improve productivity, collaboration, learning, and accessibility. Our analysis highlights that features such as fast input, scalability, and visual guidance are well supported by various modalities, while XR and haptic technologies excel in coordination and user feedback. However, many modalities need to improve the precision, accuracy, and protection of user privacy. As these technologies evolve, their adoption is expected to transform software development practices, making them more efficient, effective, and inclusive.Consequently, we observe the need to tackle the following research questions:

- How can advanced I/O sensory technologies be further optimized to improve precision and accuracy in software development tasks, particularly for critical applications?
- What are the most effective methods for enhancing user privacy protection in biometrics and body-tracking systems used in software development environments?
- How can sensory augmentation technologies be integrated into existing software development workflows to maximize their impact on team collaboration and overall productivity?

Cognitive methods. We also discussed that cognitive methods can be used to augment software visualization in several ways. They can be used to assess the interplay of multiple aspects of cognition. For example, we identify the main aspects of software visualization: attention, memory, perception, learnability, decision-making, and emotion. We concluded that the use of techniques such as EEG, ECG, and eye tracking can be used to embed cognitive measurements in the IDE. We list a set of open research questions that should be addressed in future work.

- Which developer tasks can benefit from an immersive experience? Is it necessary or preferable to have a single immersive experience for all development tasks?
- How can we incorporate these experiences into the developer workspace? How does the degree of immersion affect developer efficiency and well-being in software engineering tasks?
- What approaches can be employed to efficiently establish cognitive elements for the design of visualizations, taking into account the difficulties in measurement and the significance of education and guidance?

Software complexity. We discussed the need for augmenting visualizations by integrating a holistic view in development environments to deal with the fact that software systems are increasingly larger and more complex. Specifically, we conclude that IDEs should support various domains and demands, smoother interactions, and artifact maintenance tasks. Here is a list of open questions that emerged as critical to answer to make advances in that area:

- Does virtual reality (VR) offer novel opportunities (such as enhanced overviews) in comparison to traditional 2D screens?
- Is it possible to leverage natural language processing and technologies like speech recognition to enhance interactions in virtual reality?
- What are the methods available for querying and visualizing our code? How can the integrated development environment (IDE) assist us in accomplishing these tasks?
- What assistance does the IDE provide in identifying the specific location of the source code that I should begin examining?

References

[1] "Cognition". Lexico. https://www.lexico.com. Accessed on 2020-05-06.

[2] F. Bork, R. Barmaki, U. Eck, K. Yu, C. Sandor, and N. Navab. Empirical study of non-reversing magic mirrors for augmented reality anatomy learning. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 169–176, 2017.

[3] Margaret M. Bradley and Peter J. Lang. Measuring emotion: The self-assessment manikin and the semantic differential. *Journal of Behavior Therapy and Experimental Psychiatry*, 25(1):49–59, 1994.

[4] Stuart K Card, Jock Mackinlay, and Ben Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.

[5] Cléber G Corrêa, Márcio E Delamaro, Marcos L Chaim, and Fátima LS Nunes. Software testing automation of vr-based systems with haptic interfaces. *The Computer Journal*, 64(5):826–841, 2021.

[6] Stephan Diehl. *Software Visualization — Visualizing the Structure, Behaviour, and Evolution of Software*. Springer, 2007.

[7] Emanuel Donchin. Surprise!... surprise? *Psychophysiology*, 18(5):493–513, 1981.

[8] Anthony Elliott, Brian Peiris, and Chris Parnin. Virtual reality in software engineering: Affordances, applications, and challenges. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 2, pages 547–550. IEEE, 2015.

[9] Marianne Frankenhaeuser. Experimental approaches to the study of catecholamines and emotion. In *Proceedings of the Symposium on Parameters of Emotion*, page 684–685, 1975.

[10] M. Gandy, R. Catrambone, B. MacIntyre, C. Alvarez, E. Eiriksdottir, M. Hilimire, B. Davidson, and A. C. McLaughlin. Experiences with an AR evaluation test bed: Presence, performance, and physiological measurement. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136, 2010.

[11] M. Geronazzo, E. Sikström, J. Kleimola, F. Avanzini, A. de Götzen, and S. Serafin. The impact of an accurate vertical localization with HRTFs on short explorations of immersive virtual reality scenarios. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 90–97, 2018.

[12] A. Ibrahim, B. Huynh, J. Downey, T. Höllerer, D. Chun, and J. O'Donovan. ARbis pictus: A study of vocabulary learning with augmented reality. *Transactions on Visualization and Computer Graphics*, 24(11):2867–2874, 2018.

[13] Daniel Kahneman. *Attention and Effort*. Prentice-Hall, Englewood Cliffs, New Jersey, 1973.

[14] Heidi Lam, Enrico Bertini, Petra Isenberg, Catherine Plaisant, and Sheelagh Carpendale. Empirical studies in information visualization: Seven scenarios. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1520–1536, 2012.

[15] F. Lu, D. Yu, H. Liang, W. Chen, K. Papangelis, and N. M. Ali. Evaluating engagement level and analytical support of interactive visualizations in virtual reality environments. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 143–152, 2018.

[16] Leonel Merino, Mircea Lungu, and Christoph Seidl. Unleashing the potentials of immersive augmented reality for software engineering. In *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 517–521. IEEE, 2020.

[17] Leonel Merino, Magdalena Schwarzl, Matthias Kraus, Michael Sedlmair, Dieter Schmalstieg, and Daniel Weiskopf. Evaluating mixed and augmented reality: A systematic literature review (2009-2019). In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 438–451. IEEE, 2020.

[18] Gijsbertus Mulder. *The Heart of Mental Effort*. PhD thesis, University of Groningen, The Netherlands, 1980.

[19] J. Orlosky, T. Toyama, K. Kiyokawa, and D. Sonntag. ModulAR: Eye-controlled vision augmentations for head mounted displays. *Transactions on Visualization and Computer Graphics*, 21(11):1259–1268, 2015.

[20] C. Reichherzer, A. Cunningham, J. Walsh, M. Kohler, M. Billinghurst, and B. H. Thomas. Narrative and spatial memory for jury viewings in a reconstructed virtual environment. *Transactions on Visualization and Computer Graphics*, 24(11):2917–2926, 2018.

[21] Doreen Seider, Andreas Schreiber, Tobias Marquardt, and Marlene Brüggemann. Visualizing modules and dependencies of osgi-based applications. In *2016 IEEE working conference on software visualization (VISSOFT)*, pages 96–100. IEEE, 2016.

[22] Bonita Sharif and Jonathan I Maletic. An eye tracking study on camelcase and under_score identifier styles. In *2010 IEEE 18th International Conference on Program Comprehension*, pages 196–205. IEEE, 2010.

[23] A Van Boxtel and M Jessurun. Amplitude and bilateral coherency of facial and jaw-elevator EMG activity as an index of effort during a two-choice serial reaction task. *Psychophysiology*, 30(6):589–604, 1993.

[24] David Watson, Lee Anna Clark, and Auke Tellegen. Development and validation of brief measures of positive and negative affect: The PANAS scales. *Journal of Personality and Social Psychology*, 54:1063–1070, June 1988.

[25] C. A. Wiesner, M. Ruf, D. Sirim, and G. Klinker. 3D-FRC: Depiction of the future road course in the head-up-display. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 136–143, 2017.

[26] Yaffa Yeshurun, Marisa Carrasco, and Laurence T Maloney. Bias and sensitivity in two-interval forced choice procedures: Tests of the difference model. *Vision Research*, 48(17):1837–1851, 2008.

[27] J. Zhang, A. Ogan, T. Liu, Y. Sung, and K. Chang. The influence of using augmented reality on textbook support for learners of different learning styles. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 107–114, 2016.