Interaction Design for Binary Reverse Engineering in Virtual Reality

Dennis G. Brown, Julian C. Bauer, Kevan D. Baker, Luke J. Wittbrodt, Samuel Mulder
Computer Science and Software Engineering, Auburn University
Auburn, Alabama, US
dgb0028, jcb0209, kzb0154, ljw0024, szm0211@auburn.edu

ABSTRACT

Securing our networks and assuring information operations in cyberspace are vital activities for national security. Knowing precisely what a piece of software does can be critical to maintaining competitive advantage—for example, determining the capabilities of malware, or comprehending the undocumented logic in a legacy supply system. Binary executable programs are particularly difficult for humans to comprehend because the compilation process is a one-way transformation from context-rich source code to a highly-optimized binary program. Our central problem is that binary reverse engineering (RE) is a highly-specialized skill that requires extensive training and experience. Additionally, the RE process requires a human-in-the-loop because the compound uncertainties introduced in disassembling and decompiling a binary program prevent a fully-automated solution. Immersive virtual reality (VR) offers novel ways to visualize and spatially interact with the complex and expansive data produced in the binary RE process. It holds potential to amplify the effectiveness of both novices-in-training and experts. In tackling this problem, we follow a human-centered interaction design process of discovery, definition, development, and iterative refinement. In our discovery, we performed a thorough survey of the cognitive models of experts performing binary RE, related elements of cognitive theory, and the affordances in VR that leverage cognitive theory to improve human effectiveness. In the definition phase, we prioritized the identified affordances in VR into an initial set for the development phase, where we implemented a VR system providing an immersive spatial interface to data provided by industry-standard reverse engineering tools. With this baseline implementation, we began iterating based on qualitative feedback from practitioners with varying experience in binary RE. While the feedback is promising, especially in user organization of code and graphs in space, our goal is to build a system ready for a formal user study of effectiveness.

ABOUT THE AUTHORS

Dennis G. Brown is a PhD student in the Auburn University Department of Computer Science and Software Engineering under in a training program funded by the United States Air Force. He has authored or co-authored over 30 peer-reviewed publications in applied research of virtual reality for command and control as well as mobile, wearable augmented reality for dismounted forces. He leverages his career experience in VR, User Experience, and Software Engineering to study novel immersive interfaces designed to improve human understanding of complex software systems.

Julian C. Bauer is an Undergraduate student in the Auburn University Department of Computer Science and Software Engineering. He has over a year of experience working professionally in the Virtual Reality industry, where he designed human-centered immersive environments that encourage intuitive, democratic usage of VR. His research interests primarily focus on applied Virtual Reality with additional interest in Human-Computer Interaction, particularly in Human Centered Design.

Kevan D. Baker is a current PhD student who has achieved his B.S. and M.S. in Computer Science at Florida Polytechnic University and is furthering his education in the domain at Auburn University. Kevan additionally has background experience in the field of bioinformatics and creating educational tools to assist in learning about protein 3D structure prediction. He is currently researching in the domain of Binary Reverse Engineering, and in visualization techniques such as those in VR.

Luke J. Wittbrodt is an Undergraduate student in the Auburn University Department of Computer Science and Software Engineering. He has prior experience working in the Unity Game Engine, designing and developing projects that deal with complex immersive spaces. His research interests lie in UX and UI design.

Dr. Samuel Mulder is an Associate Research Professor at Auburn. Prior to entering academia, he spent 17 years doing research in cyber security, vulnerability assessment, and binary program analysis at Sandia National Labs. Research Interests: binary program analysis, adversarial analysis, reverse engineering, machine learning, AI, virtual reality, combinatorial optimization problems.

Interaction Design for Binary Reverse Engineering in Virtual Reality

Dennis G. Brown, Julian C. Bauer, Kevan D. Baker, Luke J. Wittbrodt, Samuel Mulder
Computer Science and Software Engineering, Auburn University
Auburn, Alabama, US
dgb0028, jcb0209, kzb0154, ljw0024, szm0211@auburn.edu

INTRODUCTION

Knowing precisely what a piece of software does can be critical to maintaining competitive advantage---for example, determining the capabilities of malware, or comprehending the undocumented logic in a legacy supply system. The practice of reverse engineering (RE) binary programs, also known as binary program understanding or binary program comprehension, is a critical activity because much software, including device firmware, is distributed only as binary code without source. Although we can scan this software for known vulnerabilities, we don't truly know what capabilities are in it, including those capabilities maliciously introduced somewhere in the supply chain, without reverse engineering.

Binary executable programs are particularly difficult for humans to comprehend for a few reasons. First, the compilation process is a one-way transformation from context-rich source code to a highly optimized binary program. Second, Rice's Theorem (Rice, 1954) implies that deciding whether a given binary program contains any non-trivial property is formally undecidable. Finally, programs are arguably the most complex things ever engineered by humans (Crockford, 2008). Many tools have been developed over time to expedite the process of binary RE, e.g., disassemblers, decompilers, profilers, and debuggers (Hex-Rays, 2023; US National Security Agency, 2023; pancake, 2023). Tools employing artificial intelligence and machine learning are showing promise (David, Alon, & Yahav, 2020; Maier, Gascon, Wressnegger, & Rieck, 2019), however, fundamental limitations, such as lacking a single ground truth disassembly of a given binary program (Li, Woo, & Jia, 2020), mean we are still far from full automation and the process depends on the expertise of a human to draw conclusions from disparate and conflicting sources of information. We have good reason to find ways to make humans more effective at binary RE.

The binary RE process taxes the cognitive abilities of practitioners in several ways to be explained later in this paper. Immersive virtual reality (VR) offers novel ways to visualize and spatially interact with complex and expansive data sets, offering a way to exploit the effects of embodied and external cognition (Wilson, 2002) from the physical realm to solve problems that are typically only conceptual. Applying VR to this problem domain has the potential to amplify the effectiveness of both novices-in-training and experts.

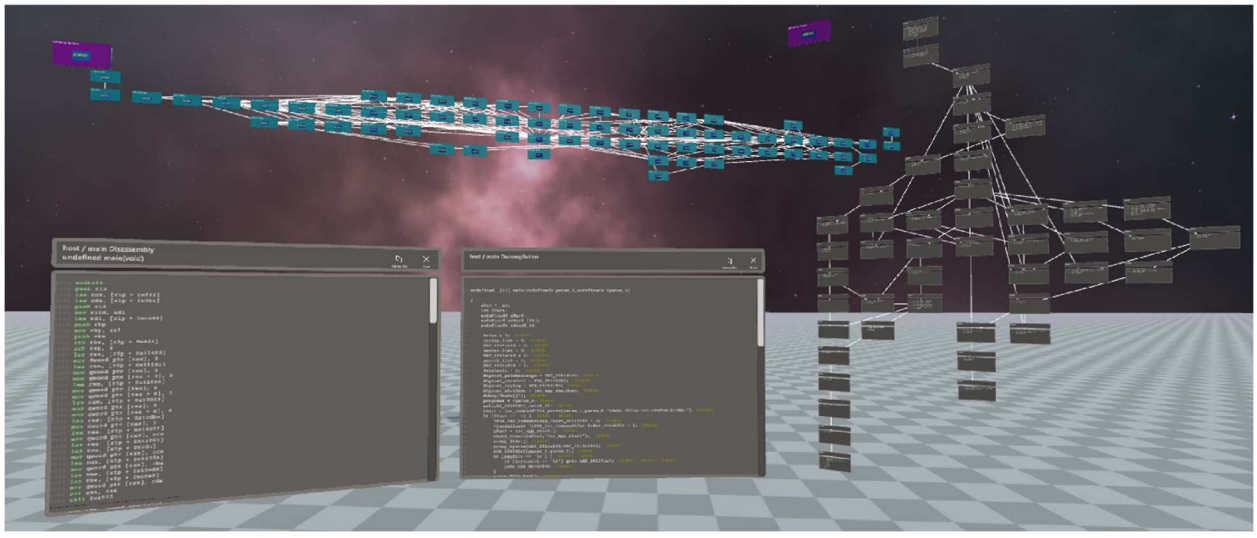

Figure 1: Exploring a binary program: function call graph (upper left) and the disassembly, decompilation, and control flow graph for a function (left to right under function call graph).

Because our work is inherently human-centric, we follow a human-centered interaction design framework. This approach is based on the British Design Council's Double Diamond (Design Council, 2005) (see Figure 2) and depends on collaboration with stakeholders and practitioners. In the left diamond, we start with divergent thinking to explore the issue then use the discovered information to converge on a well-defined instance of the issue. In the right diamond, we first develop a wide range of potential designs, then narrow down to an implemented solution. We then iterate the right-hand diamond as we continue to pursue this line of research.

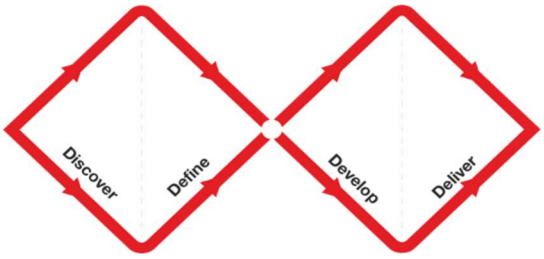

Figure 2: Double Diamond approach to delivering capability (Design Council, 2005).

In the remainder of this paper, we will discuss how we are applying this design thinking approach to our research phase-by-phase (discovery, definition, development, delivery), the feedback we received on the first iteration of our system, and the parameters for our upcoming formal user study.

DISCOVERY PHASE

Our approach to the discovery phase began with a cognitive task analysis (Militello & Hutton, 1998). We followed a cognitive systems engineering approach (Hollnagel & Woods, 2005), examining the joint cognitive system of human and machine performing binary RE. A comprehensive literature survey revealed salient concepts in three groups: (1) cognitive models of binary RE, (2) cognitive phenomena related to those elements, and (3) affordances in VR that affect those cognitive phenomena. While this paper summarizes our findings, our survey paper (Brown, Mulder, & Mulder, 2024) contains a full version. We begin with elements of the cognitive models of binary RE.

NOTE: In this paper we discuss affordances and in particular, affordances in VR. We use the term "affordance" to refer to a perceivable aspect of the environment that fosters interaction and enables external and embodied cognition. "Affordance in VR" refers to an environmental feature available in the VR experience. Most of the features we have implemented so far are intrinsically 2D, such as code listing windows, but they can be created, sized, oriented, and placed arbitrarily in the 3D space by the user to carry out a task. This approach builds a foundation for evaluating the effects of external and embodied cognition and is the subject of active research in other use cases (Lisle, Davidson, Gitre, North, & Bowman, 2021).

Literature Survey part 1: Cognitive Models of Binary RE

Cognitive models (also known as mental models) are internal representations humans use to understand and reason about the external world (Craik, 1943). Researchers have created cognitive models for source code RE and binary RE that capture the process of understanding code features and capabilities. We found several salient and common themes.

In the basic model of software RE, practitioners start by observing the program. They use short-term memory and existing knowledge of computing concepts and language syntax to build a multi-level internal semantic representation of the program. These are the essential elements of program comprehension (Shneiderman & Mayer, 1979). Later models of RE, particularly for binary RE, involve an iterative sensemaking or abductive reasoning process. Practitioners form, test, and update hypotheses within goals and plans (Bryant, Mills, Peterson, & Grimaila, 2012; Nyre-Yu, Butler, & Bolstad, 2022). They create increasingly complete hypotheses (Brooks, 1983; Weigand & Hartung, 2012; Dudenhofer, 2019; Votipka, Rabin, Micinski, Foster, & Mazurek, 2020) and test them through experimentation (Bryant et al., 2012; Sisco, Dudenhofer, & Bryant, 2017; Dudenhofer, 2019; Votipka et al., 2020). They adjust their problem framing based on observed results (Klein, Phillips, Rall, & Peluso, 2007; Bryant et al., 2012; Dudenhofer, 2019; Votipka et al., 2020).

The literature reveals other overarching characteristics of the binary RE process. Practitioners can become disoriented by recursions and execution paths (Zayour & Lethbridge, 2000). The process strains working memory (Shneiderman & Mayer, 1979; Zayour & Lethbridge, 2000) and often requires external memory aids (Détienne & Bott, 2001; Storey, 2005) and determining what to ignore (Mantovani, Aonzo, Fratantonio, & Balzarotti, 2022). Success depends on retrieving and generating declarative (factual) and procedural (patterns of interaction) knowledge (Bryant et al., 2012)

and referring to an overview of the binary executable (Votipka et al., 2020). Practitioners frequently translate code into a higher-level language (Sisco et al., 2017) and use beacons—items of interest chosen per the practitioner's judgment—to guide their work, with beacons in binary RE being more diverse than in source-code-based RE (Brooks, 1983; Dudenhofer, 2019; Votipka et al., 2020).

Literature Survey part 2: Cognitive Theory Applied to Binary RE

After identifying common elements of cognitive models of RE, we researched the cognitive phenomena that impact these model elements. We found examples in the literature that fit into three general categories.

External Cognition: Humans create and interact with external knowledge representations to reduce memory load, e.g., notes and reminders (Scaife & Rogers, 1996). They also use computational tools (e.g., calculators, or in our use case, various RE analysis tools) to make tasks easier (Preece, Rogers, & Sharp, 2019) and annotate, reorder, or restructure external representations of knowledge (Preece et al., 2019).

Embodied Cognition and Memory: Sensorimotor interaction within an environment stimulates the brain to process inputs from the body and external environment. Humans exploit our natural reliance on navigational and spatial cognition by building memory palaces to enhance memory; this works by associating data with specific locations or objects in an environment, (Ale, Sturdee, & Rubegni, 2022). Whole-body stimuli can also expedite memory storage and retrieval, such as associating data with particular motions or sensory inputs (Ale et al., 2022).

Cognitive Load Theory: This theory suggests humans have limited processing bandwidth, which can be optimized by balancing types of cognitive load. Intrinsic load is inherent to the task (the task would not be completed without this load); germane load connects current processing to long-term memory, and extraneous load arises from how tasks are presented and the means of interaction (Sweller, Van Merrienboer, & Paas, 2019). Methods to reduce extraneous load include studying solved problems (worked example effect), presenting information through multiple modalities (modality effect), and removing redundant information across modalities to ease conceptual reconciliation (Hollender, Hofmann, Deneke, & Schmitz, 2010).

Literature Survey part 3: Affordances in VR

We surveyed the literature to discover the affordances of immersive VR most closely associated with the cognitive phenomena described above, including prior work in applying VR to RE. We found affordances in three broad categories.

Use of Space: *Spatial semantics* provide semantic information about the sensemaking task through spatial organization to help the user understand the circumstances of the task (Andrews, Endert, & North, 2010). *Persistence* exploits the practitioner's spatial memory in the large 3D space when navigating and understanding code (Elliott, Peiris, & Parnin, 2015). User organization of visualizations in 3D space allows the user to express or record knowledge via positioning the data objects; users may need constraints to organization frameworks to fully take advantage of the space (Batch et al., 2020). Together, all three variations of exploiting 3D space have been shown to improve performance on sensemaking tasks (Lisle et al., 2021).

Information presentation: *Incremental formalism* evolves the visualization as one progresses in the task (Andrews, Endert, & North, 2010). *Cross-modal mapping* (coordinating multiple modes of interaction) enhances memory by exploiting multiple senses (Moloney, Spehar, Globa, & Wang, 2018). *Metaphors* to visualize programs leverage the participant's germane knowledge in other areas to more easily hold in information in memory for the RE task (Fittkau, Krause, & Hasselbring, 2015; Oberhauser & Lecon, 2017; Capece, Erra, Romano, & Scanniello, 2017; Averbukh et al., 2019; Romano, Capece, Erra, Scanniello, & Lanza, 2019; Hoff, Gerling, & Seidl, 2022; Weninger, Makor, & Mossenbock, 2020). *Signalling* directs users to specific types of information to increase understanding (Albus, Vogt, & Seufert, 2021; Mayer, 2005).

Other considerations: An *embodied assistant* (de Melo, Kim, Norouzi, Bruder, & Welch, 2020) provides a natural interface to guide progress. Incorporating common reverse engineering tools generates the information and meta-information most relevant to the task.

Stakeholder Analysis supporting Discovery

Our stakeholder analysis portion of the Discovery phase included three main elements: defining the "as-is" binary RE workflow; capturing the impacts of the "as-is" workflow in an empathy map; and a contextual inquiry to brainstorm about "to-be."

As-Is Workflow: We defined a notional practical workflow for binary static analysis supporting RE. It is broken into basic and advanced analysis and requires manually correlating the outputs of various tools. Basic static analysis includes steps such as reviewing the file header; identifying the type of file (architecture, format, etc.); reviewing strings (may need to be decrypted with another tool); reviewing symbols, function names, and library calls; and searching for embedded files or resources. Advanced static analysis includes disassembly and decompilation; code overview; reviewing the function call graph and function-level control flow graphs; pinpointing items of interest; adding comments or annotations; renaming functions (replacing auto-generated names with ones that have semantic meaning), disassembling at a micro scale to correct disassembly errors; and defining data structures. This workflow varies from practitioner to practitioner.

Empathy Map: Through observation and direct questions, we built an empathy map to understand how the current RE workflow affects practitioners. See Figure 3.

Contextual Inquiry: Brainstorming with RE practitioners, we captured their desires for a more intuitive experience with RE tools. They identified needs in several areas: identifying and tracking toward goals and hypotheses (designating them and tracking progress); enhancing working memory through external aids (e.g., a notepad); eliminating extraneous cognitive load so that working memory capacity can be used for the intrinsic problem instead of overhead workload (e.g., masking out what can be ignored; employing automation); identifying, marking, and tracking beacons; and providing an expedited way to get an overview of the complete binary program while still providing tools to drill down.

DEFINITION PHASE

The Discovery phase revealed a wide array of considerations. The purpose of the Definition phase in the Double Diamond design process is to systematically distill those findings into a realistic initial set of requirements. We designed a two-step process. In the first step, we formed three groups of elements from the three parts of the literature survey and added in elements from the stakeholder analysis. From these three groups of elements, we formed a set of "threads." Each thread traces across the three groups, linking a cognitive model element to a related cognitive

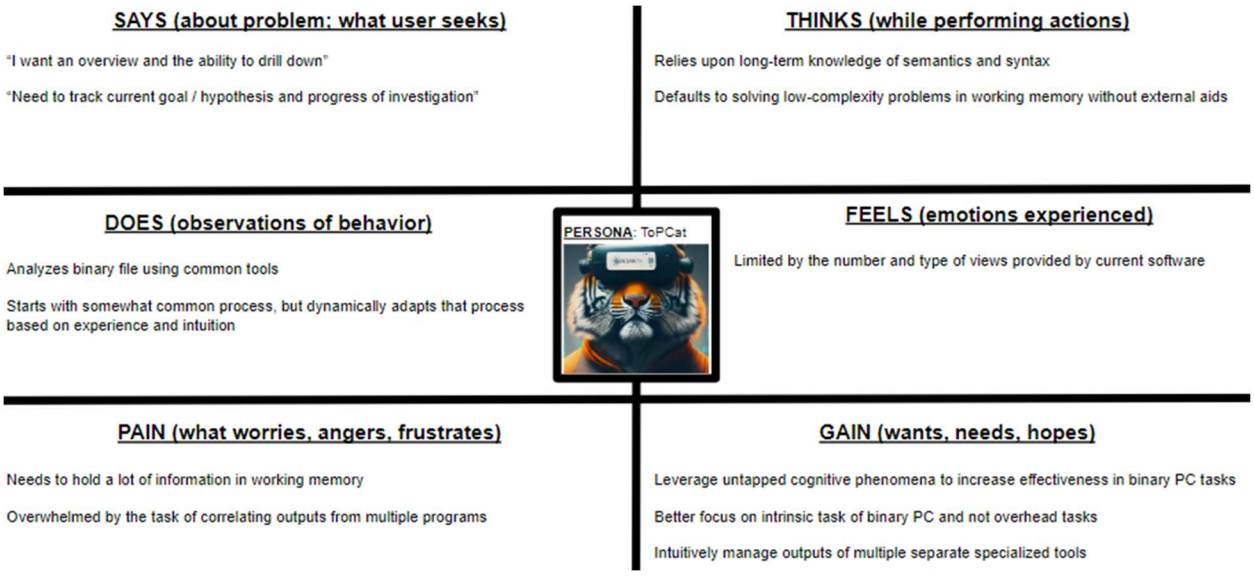

Figure 3: Empathy map.

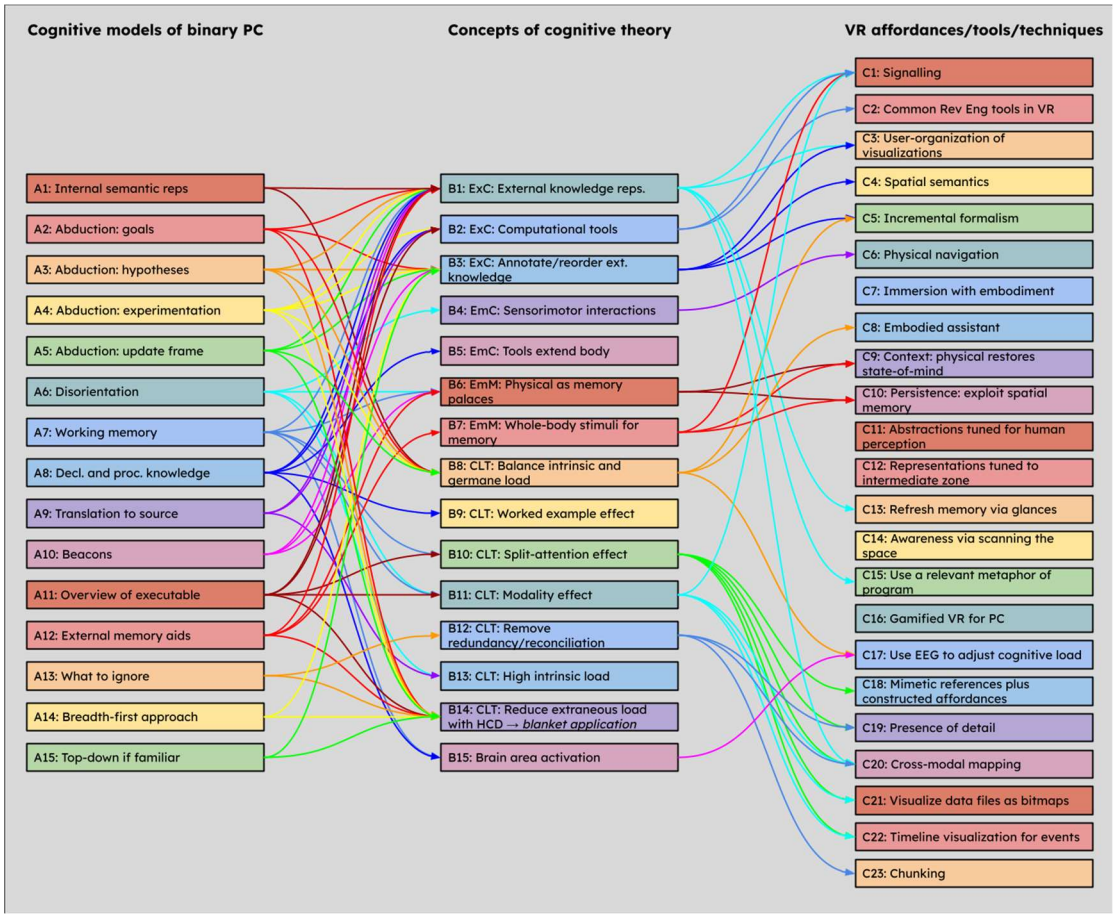

Figure 4: Threads connecting elements of (A) cognitive models of binary RE, (B) concepts of cognitive theory, and (C) VR affordances/tools/techniques.

phenomenon and then to a related affordance in VR. These relationships were decided subjectively based on how we interpreted the survey findings. These threads are depicted in Figure 4.

In the second step, we employed affinity mapping—a collaborative and subjective sorting process—to group the threads into three cohesive themes of highest relevance: enhancing abductive iteration, augmenting working memory, and supporting information organization and feature discovery. After binning the threads, we prioritized those that would provide the largest returns with the smallest development effort to facilitate early feedback. Figure 5 depicts the threads and affordances sorted into themes. The initial set of requirements for our system

This process yielded a set of affordances to implement in the first iteration of our system. This process did not define how they would be implemented; that will be addressed in the development phase described below. We aim to include these affordances (as highlighted on the right side of Figure 5): spatial semantics, incremental formalism, embodied assistant (stretch goal), persistence, metaphors, cross-model mapping (stretch goal), signalling, user organization of visualizations, notepad, and of course, incorporating common reverse engineering tools.

DEVELOPMENT → DELIVERY PHASE ITERATIONS

Once we defined the initial scope of work, we moved to the right-hand diamond of the double diamond approach encompassing development and delivery. We are executing multiple iterations of the "Develop → Deliver" diamond. These iterations implement new or updated features and are driven by user feedback. Their purpose is to deliver a system that is reliable to ensure valid and repeatable results, user-friendly to reduce extraneous friction points, and

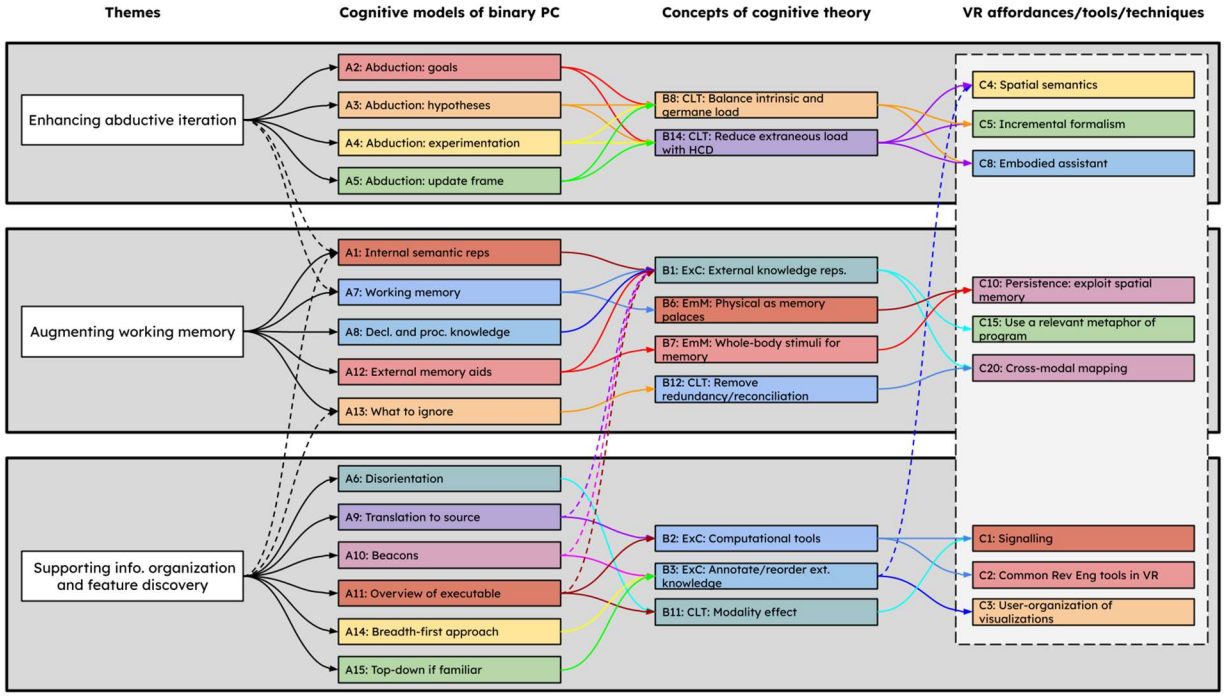

Figure 5: Primary themes for analysis with most closely-related elements; highlighted area indicates highest-priority affordances in VR.

relevant to the RE task. The first iteration encapsulates a large startup effort to build a "minimally viable product" (MVP) as a starting point for further minor iterations. These iterations are leading up to a formal user study to gauge effectiveness of our implemented affordances. Based on the outcome of that study, we will pivot and plan the next major portion of our research.

Iteration 1: MVP

Given our initial set of affordances, we defined a feature set that would implement, at least minimally, each affordance. This feature set is shown in Figure 6. The MVP employs two primary types of visualizations, graphs and slates. Each type of visualization can be created, repositioned, resized, and deleted in the 3D space at the discretion of the practitioner.

- *Graphs* visualize the elements and flows at two levels: *call graphs* display functions detected in a binary program and the flows between them, while *Control Flow Graphs (CFG)* display basic blocks within a function and flows between them. Each node of the call graph has a CFG nested within it (conceptually). The MVP can render the graphs via the hierarchical Sugiyama method (Healy & Nikolov, 2013) (as demonstrated in Figure 1) or with a force-directed method. The graph nodes in the 3D environment are responsive: users can select functions nodes for further analysis, and users can manipulate the visualization attributes of any node type. Visual callouts draw the user's attention to functions with identifiable capabilities.
- *Slates* display scrollable text fields containing the outputs of external tools, such as header information, strings, function disassemblies, and function decompilations. We additionally have a singleton slate that allows user input and functions as a notepad.

System Design and Architecture

We built a platform, Cognitive Binary Reverse Engineering (CogBRE), to support the implementation and testing of affordances in VR. We have three primary design objectives for this system:

- Leverage existing reverse engineering capabilities: There are many high-quality tools that offer means to control their functionality through automated means, such as command line scripting or application programming interfaces (APIs). We want to reuse these tools rather than implement redundant capabilities.

Theme	Affordance	MVP Implementation
Enhancing abductive iteration (hypothesis loop)	Spatial semantics	Hierarchical (Sugiyama) graphs of function calls and control flow
	Incremental formalism	User can manually explore Call Graph → CFG → Code
	Embodied assistant	*(not yet)*
Augmenting working memory	Persistence: exploit spatial memory	Environment provides basic tools for user-managed placement
	Use a relevant metaphor of program	Basic hierarchical (Sugiyama) graphs only
	Cross-modal mapping	*(not yet)*
Supporting information organization and feature discovery	Signalling	*(not yet)*
	Common Rev Eng tools	Exploit via chain of Nexus → Oxide → Ghidra
	User-organization of visualizations	Create, Move, Delete slates and graphs arbitrarily in 3D space

Figure 6: CogBRE MVP Feature Implementation.

- Support multiple VR, AR, MR clients: Although our first focus is on a fully immersive VR environment, we want to keep the system sufficiently modular to support other types of clients in the future, and to support multiple users.
- Be comfortable to use for extended periods: The process of binary RE can take hours. The system needs to minimize strain, fatigue, and simulator sickness as much as possible

In working toward these objectives, we designed a set of modular capabilities. Figure 7 shows the high-level architecture of the system. The remainder of this section will provide additional details of the major system components.

Oxide: Oxide is a flexible tool for performing analysis of binary programs (Mulder,2014). The design is modular, allowing the integration of multiple third-party tools along with custom analysis modules. For example, Oxide might pull disassembly from Ghidra (US National Security Agency, 2023), semantic features from Capa (FLARE Team, 2024), and a control-flow graph from angr (Shoshitaishvili et al., 2016). Since each of these tools handle loading differently, the Oxide modules map the address space into a common space based on file offsets. This allows us to easily compare and integrate results. For this project, we implemented a custom interface for Oxide called the Nexus that facilitates interactivity between the VR client and the Oxide backend.

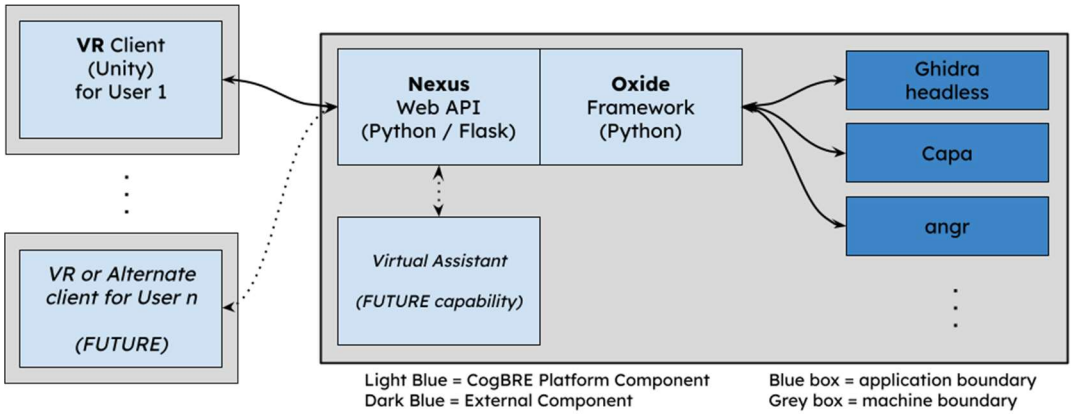

Figure 7: CogBRE System Architecture.

Nexus: The Nexus is the broker between the client systems and data sources. Implemented as a lightweight Python module, it provides a basic RESTful web API to clients using Flask 1.1. Its runtime instance hosts Oxide, which in turn connects with various data sources. Upon a client API call, it invokes appropriate method(s) in Oxide, processes the return values into a JSON payload if necessary, and returns that to the client. The web API design pattern allows a wide variety of clients to easily connect, and Nexus provides a rudimentary mechanism to track multiple users. This pattern also makes it easy to place the intense workload of reverse engineering tools on a different host than the also-intense workload of rendering the VR display.

VR Client: Of many potential client types, we first implemented a VR client, which is required to test our hypotheses about the affordances of an immersive environment. We provide more details on the design in several categories, as follows.

- Technology: The VR client is developed in Unity 2021.3.4f1 and runs on common desktop operating systems. We use Microsoft's Mixed Reality Toolkit 2 to provide many of the VR components. Our development targets SteamVR devices; in particular, we are using the HTC Vive Pro 2 head-mounted display due to its display resolution and field of view.
- Software Architecture: The architecture adopts the common pattern of a singleton GameManager that maintains references to instances of important objects in the environment and controls environmental conditions on update. New instances of virtual objects are created and destroyed based on user actions.
- Information Architecture: The VR application mirrors Oxide's hierarchy of collections (of files), files (primarily binary programs), functions, and basic blocks through its own hierarchy of those respective classes. Additionally, it provides access to all collections that are created in Oxide and all binary files that are imported into Oxide, via Nexus. As more information about a binary or a function is pulled via Nexus (for example, header information, disassembly, or decompilation), the corresponding VR object instances are progressively augmented.
- Performance Considerations: In an effort to remain performant and engaging for users, VR pulls data from Nexus "on demand" as the user requests it. We also fully leverage Oxide's intelligent caching capability, maintaining a high level of interactivity in the application.
- Quality-of-Life Considerations: Because a binary RE practitioner may spend an extended period of time in the VR application, we've taken steps in our design to increase presence and reduce cybersickness (Weech, Kenny, & Barnett-Cowan, 2019). The environment is at room scale (implied) and static except when objects are directly manipulated by the user, and for now, locomotion is only via physical user movement. Additionally, we maintain a responsive frame rate even during intensive operations or long-running transactions using Unity's Coroutines for enumerable tasks and C# Asynchronous Tasks for monolithic transactions to minimize the computation during each frame.

With these system components in place, we started gathering feedback on the system.

Feedback on Iteration 1 (MVP)

We held feedback sessions with five different individuals associated with our research group (four graduate students and one undergraduate). These individuals had moderate or higher experience with binary RE and no to moderate VR experience. After a brief demonstration of how the system and features work, we let the users explore the system freely. We observed their experience through the lens of empathetic questions: *What did they (users) do/say/think/feel?* Key observations include the following. Users found the system easier to use in person than they expected based on videos and images. They needed little to no guidance to use the head-mounted display (HMD) and controllers. Users praised the flexibility of the system compared to other reverse engineering tools, such as being able to analyze multiple binary files in the same environment. One user prone to VR-induced nausea did not experience it in this system. The participants took issue with illegibly small fonts; likely due to using a lower-resolution HMD than originally intended (this issue has been corrected).

After observing the participants' sessions, we asked direct questions to obtain explicit feedback, as summarized below:

- *Would you use this tool as-is when trying to comprehend a binary program?* This question yielded mixed results; most (but not all) users said no due to poor text legibility and lack of functionality.
- *What is better in the VR environment than tools and processes you usually use? What is worse?* Users responded that obtaining binary information was more intuitive and they appreciated the spatial real estate

for task management and comparison. However, they found severe limitations in having no ability to make freeform annotations and targeted transformations (e.g., function renaming, structure definition, and library call identification).

- *What are minor improvements you want to see in this tool?* Users wanted integration of abstractions into a unified graph, with a "find all instances" function and collapsible graphs to better manage the amount of displayed information (remove what does not matter in the moment).
- *With those minor improvements, would you use this tool when trying to comprehend a binary program?* Most users stated that they would use this program with the aforementioned minor improvements and that they would apply this technology in the context of solving very basic reverse engineering tasks.
- *What are major improvements you want to see?* Users provided a wide variety of ideas. Predominant themes were advanced integration with additional specialized tools and more robust comparisons between disassembly and decompilation across different functions.
- *With those major improvements, would you use this tool when trying to comprehend a binary program?* Users predominantly stated that they would use CogBRE under the stipulation of improvement.

Iteration 2

We are using the feedback from Iteration 1 to improve the design and development of the upcoming iterations of our system. We are currently implementing several features: a VR notepad (freeform text input) and text copy/paste between slates; importing and displaying capability findings from Capa as signals/beacons to draw attention to potentially important items; and basic usability and quality-of-life improvements for existing features. Other features under consideration are maintained in a backlog.

USER STUDY DESIGN

Once CogBRE is sufficiently reliable and user-friendly as determined through the iterative feedback and a pilot study, we will initiate a formal user study to measure the effectiveness of different configurations of the CogBRE and related RE tools. The purpose of this study is to discover quantifiable measures of effectiveness of using affordances in immersive VR in the process of performing binary reverse engineering tasks. The study protocol is under review with our Institutional Review Board and subject to change from what we present here.

Our hypotheses include: (1) Participants will have better task completion metrics with certain levels of affordances in VR. (2) Cognitive load will vary by level of affordances in VR. (3) Participants will rate certain affordances in VR (e.g., user organization, spatial memory, signalling) higher than others in terms of usefulness. (4) Participants who have better performance in the immersive environments will have used more of the VR space.

The study will involve three conditions or configurations: one traditional computer desktop environment and two immersive VR environments. Participants will execute one program understanding task (reverse engineering challenge) in one of these three environments, as this is a between-subjects design. The independent variable is the environment in use. Dependent variables are task performance metrics (e.g., accuracy and speed), system usability (survey), cognitive load (survey), immersive (survey), and feature rating (survey); the system will also log their actions in the environments. In the traditional approach, they will use reverse engineering tools in a common desktop windowed environment controlled by keyboard and mouse. In the VR environments, they will use an approved head-mounted stereoscopic display and two hand controllers. Each participant will attempt to solve a single reverse engineering challenge in one of the three environments. The reverse engineering challenge will be targeted to take up to 30 minutes to solve; participants can stop at any time. Complete sessions, from consent to performing the task to answering a survey, should take no longer than 60 minutes per participant.

We will recruit from the population of our school’s computer science and engineering students with an understanding of compilers and binary applications, who will typically be from the upper classes or graduate programs. The expected magnitude of the differences we expect to detect is medium-to-high, estimated at $d = 0.7$ based on Cohen's “Statistical power analysis for the behavioral sciences,” similar to other VR studies. We will analyze the results using Analysis of variance (ANOVA). Using statistical power = 0.8, significance level = 0.05, and $d = 0.7$ in a simple power analysis (executed via the Python module Statsmodels), our experiment needs 18 samples per condition. With 3 conditions we require 54 valid sessions, so we will recruit 70. Two similar studies of testing VR for program comprehension in three

conditions had numbers of participants in this same range (Romano, Capece, Erra, Scanniello, & Lanza, 2019; Hoff, Gerling, & Seidl, 2022).

We will collect data in several ways. Via surveys, we will employ the NASA Task Load Index (TLX) to gauge cognitive load (Hart & Staveland, 1988). Brooke's System Usability Scale will gauge user satisfaction with the system (Brooke, 1996). We will use applicable elements of the Presence Questionnaire to measure the participants' levels of immersion (Witmer, Jerome, & Singer, 2005). Our final survey asks participants to rate relative value of features in the environment using a Likert scale we developed. In addition to surveys, the system will transparently log command line activity in the desktop condition and the participant's telemetry (position and orientation of head and hands) and actions in the VR condition. Finally, we will record the participants' performance (speed and accuracy) on the reverse engineering challenge they performed. In our analysis, we will compare accuracy, speed, workload, system usability, and presence between environments using a statistical hypothesis test. Logged data will be analyzed for trends and patterns such as most frequently used commands or features, or hotspot activity locations in the virtual space.

CONCLUSION

Binary RE is a difficult but critical task that is not likely to be fully automated soon. We created a Design Thinking-based approach to building and evaluating an interactive system intended to improve the performance of reverse engineering practitioners using immersive virtual reality. Our discovery phase carried out a cognitive task analysis to understand the cognitive elements of binary RE, specifically those that show the greatest potential for improvement via embodied and external cognition. In our definition phase, we linked those findings to specific affordances in VR and narrowed them down to a core set of affordances to implement. We kicked off a series of development and delivery iterations, starting with an MVP, implementing the core affordances as features in VR that make the abstract and conceptual problem of binary RE more physical, where we can stimulate embodied and external cognition to augment traditional internal cognition. Our system is almost ready for a formal user study to measure effectiveness, for which we have developed a detailed protocol we hope to run in the next few months.

In the process of improving any human-in-the-loop process, assumptions at design time regarding how users solve problems and interact with tools are expedient, but those assumptions can stray from reality just enough to cause significant problems in deployment and usage. Our approach of starting with a low-level cognitive task analysis and building up to traceable affordances in VR provides a way to challenge and move past those assumptions. Although this paper focuses on a very specific use case and applications of VR specific to it, we believe this approach can be applied to a wide variety of training and simulation use cases.

REFERENCES

Albus, P., Vogt, A., & Seufert, T. (2021). Signaling in virtual reality influences learning outcome and cognitive load. *Computers & Education*, *166*, Article 104154.

Ale, M., Sturdee, M., & Rubegni, E. (2022). A systematic survey on embodied cognition: 11 years of research in child–computer interaction. *International Journal of Child-Computer Interaction, 33*, Article 100478.

Andrews, C., Endert, A., & North, C. (2010). Space to think: Large high-resolution displays for sensemaking. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10),* 55-64.

Averbukh, V., Averbukh, N., Vasev, P., Gvozdarev, I., Levchuk, G., Melkozerov, L., & Mikhaylov, I. (2019). Metaphors for software visualization systems based on virtual reality. In L. T. De Paolis & P. Bourdot (Eds.), *Augmented reality, virtual reality, and computer graphics* (pp. 60–70). Cham: Springer International Publishing.

Batch, A., Cunningham, A., Cordeil, M., Elmqvist, N., Dwyer, T., Thomas, B. H., & Marriott, K. (2020). There is no spoon: Evaluating performance, space use, and presence with expert domain users in immersive analytics. *IEEE Transactions on Visualization and Computer Graphics, 26* (1), 536-546.

Bragdon, A., Reiss, S. P., Zeleznik, R., Karumuri, S., Cheung, W., Kaplan, J., LaViola, J. J. (2010). Code bubbles: rethinking the user interface paradigm of integrated development environments. *Proceedings of the 2010 ACM/IEEE 32nd International Conference on Software Engineering, 1*, 455-464.

Brooke, J. (1996). SUS – a quick and dirty usability scale. In P.W. Jordan, B. Thomas, B.A. Weerdmeester, & I.L. McClelland (Eds.), *Usability Evaluation in Industry* (pp. 189-194). Taylor & Francis.

Brooks, R. (1983). Towards a theory of the comprehension of computer programs. *International Journal of Man-Machine Studies, 18* (6), 543-554.

Brown, D., Mulder, E., & Mulder, S. (2024). *Toward improving binary program comprehension via embodied immersion: A survey*. ArXiv. https://doi.org/10.48550/arXiv.2404.17051

Bryant, A., Mills, R., Peterson, G., & Grimaila, M. (2012, 01). Software reverse engineering as a sensemaking task. *Journal of Information Assurance and Security, 6*, 483–494.

Capece, N., Erra, U., Romano, S., & Scanniello, G. (2017). Visualising a software system as a city through virtual reality. In L. T. De Paolis, P. Bourdot, & A. Mongelli (Eds.), *Augmented reality, virtual reality, and computer graphics* (pp. 319–327). Cham: Springer International Publishing.

Craik, K. J. W. (1943). *The nature of explanation.* Cambridge: Cambridge University Press.

Crockford, D. (2008). *Javascript: The good parts*. O'Reilly Media, Inc.

David, Y., Alon, U., & Yahav, E. (2020). Neural reverse engineering of stripped binaries using augmented control flow graphs. *Proceedings of the ACM on Programming Languages, 4* (OOPSLA), 1–28.

de Melo, C. M., Kim, K., Norouzi, N., Bruder, G., & Welch, G. (2020). Reducing cognitive load and improving warfighter problem solving with intelligent virtual assistants. *Frontiers in Psychology, 11*.

DeLine, R., & Rowan, K. (2010, May). Code canvas: zooming towards better development environments. *Proceedings of the 2010 ACM/IEEE 32nd International Conference on Software Engineering, 1*, 207-210.

Design Council. (2024). *The Double Diamond.* https://www.designcouncil.org.uk/our-resources/the-double-diamond/

Detienne, F., & Bott, F. (2001). *Software design—cognitive aspects*. Berlin, Heidelberg: Springer-Verlag.

Dudenhofer, P. P. (2019). Modeling and automating the cyber reverse engineering cognitive process. *23rd Colloquium for Information Systems Security Education (CISSE)*. Las Vegas, NV, United States.

Elliott, A., Peiris, B., & Parnin, C. (2015). Virtual reality in software engineering: Affordances, applications, and challenges. *Proceedings of the 2015 ACM/IEEE 37th International Conference on Software Engineering, 2*, 547-550.

Fittkau, F., Krause, A., & Hasselbring, W. (2015). Exploring software cities in virtual reality. *Proceedings of the 2015 IEEE 3rd Working Conference on Software Visualization (VISSOFT)*, 130-134.

FLARE Team. (2024). *CAPA, a tool to identify capabilities in programs and sandbox traces*. https://github.com/mandiant/capa

Hart, S.G., & Staveland, L.E. (1988). Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In P.A. Hancock & N. Meshkati (Eds.), *Advances in Psychology* (pp. 139-183). North-Holland.

Hex-Rays. (2023). *IDA Pro binary code analysis tool.* https://hex-rays.com/ida-pro/.

Hoff, A., Gerling, L., & Seidl, C. (2022). Utilizing software architecture recovery to explore large-scale software systems in virtual reality. *Proceedings of the 2022 IEEE 10th Working Conference on Software Visualization (VISSOFT)*, 119-130.

Hollender, N., Hofmann, C., Deneke, M., & Schmitz, B. (2010). Integrating cognitive load theory and concepts of human–computer interaction. *Computers in Human Behavior, 26*, 1278-1288.

Hollnagel, E., & Woods, D. (2005). *Joint cognitive systems: Foundations of cognitive systems engineering*. CRC Press.

Klein, G., Phillips, J. K., Rall, E. L., & Peluso, D. A. (2007). A data-frame theory of sensemaking. In R. R. Hoffman (Ed.), *Expertise out of context: Proceedings of the sixth international conference on naturalistic decision making* (p. 113-155). Lawrence Erlbaum Associates Publishers.

Li, K., Woo, M., & Jia, L. (2020). On the generation of disassembly ground truth and the evaluation of disassemblers. *Proceedings of the 2020 ACM workshop on forming an ecosystem around software transformation*, 9–14.

Lisle, L., Davidson, K., Gitre, E. J., North, C., & Bowman, D. A. (2021). Sensemaking strategies with immersive space to think. *Proceedings of 2021 IEEE Virtual Reality and 3D User Interfaces (VR)*, 529-537.

Maier, A., Gascon, H., Wressnegger, C., & Rieck, K. (2019). TypeMiner: Recovering types in binary programs using machine learning. *Proceedings of the 2019 International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*.

Mantovani, A., Aonzo, S., Fratantonio, Y., & Balzarotti, D. (2022). RE-Mind: a First Look Inside the Mind of a Reverse Engineer. *Proceedings of the 31st USENIX Security Symposium (USENIX Security 22)*.

Mayer, R. E. (2005). Cognitive theory of multimedia learning. In R. Mayer (Ed.), *The Cambridge Handbook of Multimedia Learning* (p. 31–48). Cambridge University Press.

Militello, L. G., & Hutton, R. J. B. (1998). Applied cognitive task analysis (acta): a practitioner's toolkit for understanding cognitive task demands. *Ergonomics, 4*1 (11), 1618-1641.

Moloney, J., Spehar, B., Globa, A., & Wang, R. (2018). The affordance of virtual reality to enable the sensory representation of multi-dimensional data for immersive analytics: from experience to insight. *Journal of Big Data, 5* (1).

Mulder, S. A. (2014). *Cross-domain situational awareness in computing networks*. Sandia National Laboratories (SAND Report). https://doi.org/10.2172/1494613

Nyre-Yu, M., Butler, K., & Bolstad, C. (2022). A task analysis of static binary reverse engineering for security. *Proceedings of the 55th Hawaii International Conference on System Sciences (HICSS 2022)*, 1–10.

Oberhauser, R., & Lecon, C. (2017, Jan.). Gamified virtual reality for program code structure comprehension. *International Journal of Virtual Reality, 17* (2), 79–88.

pancake. (2023). *Radare2: Libre Reversing Framework for Unix Geeks*. https://www.radare.org/n/radare2.html

Preece, J., Rogers, Y., & Sharp, H. (2019). *Interaction design: Beyond Human-Computer Interaction (5th ed.)*. Hoboken, NJ: Wiley.

Rice, H. G. (1954). Classes of recursively enumerable sets and their decision problems. *Journal of Symbolic Logic, 19* (2), 121–122.

Romano, S., Capece, N., Erra, U., Scanniello, G., & Lanza, M. (2019). On the use of virtual reality in software visualization: The case of the city metaphor. *Information and Software Technology, 114*, 92-106.

Scaife, M., & Rogers, Y. (1996). External cognition: how do graphical representations work? *International Journal of Human-Computer Studies, 45* (2), 185-213.

Shneiderman, B., & Mayer, R. (1979, Jun 01). Syntactic/semantic interactions in programmer behavior: A model and experimental results. *International Journal of Computer & Information Sciences, 8* (3), 219-238.

Shoshitaishvili, Y., Wang, R., Salls, C., Stephens, N., Polino, M., Dutcher, A., Vigna, G. (2016). SoK: (State of) The Art of War: Offensive Techniques in Binary Analysis. *Proceedings of the IEEE 37th Symposium on Security and Privacy*.

Sisco, Z. D., Dudenhofer, P. P., & Bryant, A. R. (2017). Modeling information flow for an autonomous agent to support reverse engineering work. *The Journal of Defense Modeling and Simulation, 14* (3), 245–256.

Storey, M.-A. (2005). Theories, methods and tools in program comprehension: past, present and future. *Proceedings of the 13th International Workshop on Program Comprehension (IWPC'05)*, 181-191.

Sweller, J., Van Merrienboer, J. J. G., & Paas, F. (2019, 06). Cognitive architecture and instructional design: 20 years later. *Educational Psychology Review, 31*, 261-292.

US National Security Agency. (2023). *Ghidra software reverse engineering suite of tools*. https://ghidra-sre.org/

Votipka, D., Rabin, S., Micinski, K., Foster, J. S., & Mazurek, M. L. (2020, August). An observational investigation of reverse Engineers' processes. *Proceedings of the 29th USENIX Security Symposium (USENIX Security 20)*, 1875–1892.

Weech, S., Kenny, S., & Barnett-Cowan, M. (2019). Presence and cybersickness in virtual reality are negatively related: A review. *Frontiers in Psychology, 10*.

Weigand, K. A., & Hartung, R. (2012). Abduction's role in reverse engineering software. *Proceedings of the 2012 IEEE National Aerospace and Electronics Conference (NAECON)*, 57-62.

Weninger, M., Makor, L., & Mossenbock, H. (2020). Memory cities: Visualizing heap memory evolution using the software city metaphor. *Proceedings of the 2020 Working Conference on Software Visualization (VISSOFT)*, 110-121.

Wilson, M. (2002, December). Six views of embodied cognition. *Psychonomic Bulletin & Review, 9* (4), 625–636.

Witmer, B.G., Jerome, C.J., and Singer, M.J. (2005). The factor structure of the presence questionnaire. *Presence: Teleoperators and Virtual Environments, 14* (3), 298–312.

Zayour, I., & Lethbridge, T. C. (2000). A cognitive and user centric based approach for reverse engineering tool design. *Proceedings of the 2000 Conference of the Centre for Advanced Studies on Collaborative Research*, 16.