Intelligent Human Systems Integration (IHSI), Vol. 200, 2026, 797–806
https://doi.org/10.54941/ahfe1007139

Generating Comic Instructions for Self-Explaining Ambient Systems

Marvin Berger, Börge Kordts, and Andreas Schrader

University of Lübeck, Ratzeburger Allee 160, 23552 Lübeck, Germany

ABSTRACT

Dynamically connecting components to form Ambient Systems offers a variety of opportunities in various use cases. Particularly, the flexible integration of smart objects and applications allows for solutions tailored to the needs and daily tasks of the respective users. However, this approach also entails some challenges as the handling of such systems can be obfuscated due to the dynamic connection. Towards this end, self-explainability of involved components and dynamically generated instructions have been proposed to counteract this issue. In this paper, we present a novel comic instruction rendering engine that can generate user instructions based on the self-descriptions of all involved components. In a user study, we evaluated the effectiveness of these instructions. Results indicate that automatically generated comic instructions can effectively support the operation of Ambient Systems consisting of interconnected Smart Objects and applications. Interview feedback showed that the tutorials' consistent structure was positively received. Additionally, the instructions were considered clear and understandable.

Keywords: Smart object guidance, Self-reflection, Comic generation

INTRODUCTION

Ambient systems consisting of several smart objects integrated into the surroundings and applications executed in the environment allow for flexible adaptation to the needs of users and their daily tasks. For example, in smart homes, this development is primarily driven by the desire to improve quality of life and save energy.

However, a dynamic connection between smart objects—with different interaction modalities—in smart environments may obfuscate interaction options for users. This is because the user interfaces and their interaction options are no longer obvious due to their integration into everyday objects. Also, interaction interfaces for smart applications may be distributed over several objects. One possible countermeasure is to explain the interaction options to the users (Burmeister, 2018). Generating instructions for smart environments is one way of doing this.

In particular, the design of instructions for smart environments is challenging as technical details and functionalities must be communicated in an understandable way. In addition, it is important to present the details of the actions to be performed, such as gesture control, in a suitable manner.

Received November 19, 2025; Revised December 2, 2025; Accepted December 17, 2025; Available online February 1, 2026

However, comics have already been successfully used as a medium for conveying complex information in various fields such as data visualization (Boucher et al., 2023) and programming (Suh, 2023). They make it possible to simplify abstract concepts by combining graphic and textual content (Suh et al., 2020). In addition, they can be understood at a glance and are therefore particularly well suited for repeating or looking up specific passages.

There are already approaches to dynamically generate instructions for ensembles of connected smart objects in smart environments. An example of such a system is the *Ambient Reflection* framework (Burmeister, 2018). It can discover smart objects and applications in the environment and dynamically connect them for users, enabling meaningful control. In addition, instructions in various formats can be generated for these ensembles. However, the framework cannot provide instructions created in comic format.

This paper therefore examines the approach of automatically generating comic instructions based on self-descriptions of the involved components. Towards this end, we present a rendering engine for the Ambient Reflection framework that enables it to generate comic tutorials. Finally, we present a user study that evaluates how suitable the generated comic instructions are, particularly in terms of clarity and understandability.

This contribution addresses the following research question: To what extent are the generated comic instructions capable of explaining the operation of connected ensembles in smart environments to users?

RELATED WORK

Research on self-explainability in smart environments is limited and mainly addresses reasoning about system adaptations and their causes.

For example, self-explainability has been used to help users understand system logic via causal models or cause-effect chains derived from experimental and observational data (Fadiga et al., 2021). To address the limitations of technical explanations, Sadeghi et al. (2024) propose user-centric intelligible explanations combining algorithmic and contextual constructs with adaptable granularity, while King (2024) introduces self-aware smart objects that employ generative artificial intelligence for synthesizing actions and explanations.

Overall, current research predominantly explains system behavior and adaptation logic but pays little attention to user–system interactions or the dynamics of interconnected adaptive ensembles.

More closely related, the Ambient Reflection framework, developed by Burmeister (2018), enables the runtime generation of instructions for users in smart environments. Its main component is the Description Mediator, which discovers the potentially involved components, queries their self-descriptions, performs probabilistic ensemble methods to dynamically connect them, and merges these descriptions into an ensemble description. Finally, rendering engines are used to generate user instructions and tutorials in several formats based on the merged self-descriptions.

Components can use a library or a proxy to be integrated into the framework and to provide self-explainability. Both approaches handle network connections, messaging, the handling of self-descriptions, and the presentation of instructions to the user.

The foundation for the instruction generation is the Smart Object Description Language (SODL). It provides a formal, hierarchical description of Smart Objects and Ambient Applications and their interactions. Ambient Applications are defined as applications operating in smart environments that leverage smart objects (Kordts et al., 2025). The documentation of interactions in SODL is based on two concepts. First, the Hierarchical Task Analysis (HTA) which recursively decomposes complex tasks into executable sub-tasks (Stanton, 2006). And second, the Virtual Protocol Model (VPM), which provides a seven-layer model for describing the interaction between humans and computers, from the real-world goal to physical execution (Nielsen, 1986).

Interactions in SODL are structured across these levels, starting with the Goal Level (the objective, e.g., "Start the music"). The underlying levels are organized temporally. The Semantic Level describes output components and system reactions (e.g., "Turn on system"), while the Lexical Level specifies the state group (e.g., power state). The Alphabetic Level names the input component (e.g., gesture sensor) and the interaction primitive representing the smallest addressable element with a meaningful relation to the interaction (e.g., "Swipe left"). The Physical Level describes the actual movements (e.g., "Move your hand horizontally left"). SODL also allows graphical media to be linked to different levels to illustrate inputs and reactions.

Notably, none of the previously mentioned works focuses on the provision of comic instructions for smart environments.

Though, recent years have seen various approaches to automated comic generation. Manual or semi-automated approaches offer creative support but are labor-intensive and unsuitable for automatic generation. Moreover, several systems focus on converting existing visual media into comics, but they require visual input. Recently, AI-based generation has become popular. However, this approach requires significant manual intervention and iteration to produce coherent and meaningful results. They particularly tend to struggle with stylistic consistency across sequential images.

In the past, on one hand, rule-based generation systems like Strip This!\[1\] have been researched. On the other hand, serialization formats have been introduced to describe existing comics. One notable example is the Comic Book Markup Language (CBML) (Walsh, 2012).

More closely related, CodeToon is a semi-automatic system that translates programming code into comics to support learners (Suh et al., 2022). Furthermore, ComicQA provides technical support solutions in the form of Hyper-Comics that link to related resources (Sumi, 2017). It uses a modular structure but is reactive and assumes that users can describe their problems.

Another notable system in this area is Comic Chat (Kurlander et al., 1996), which converts online text chat into a comic representation in real-time using

[1]https://www.kesiev.com/stripthis/ (Visited on 20.11.2025).

a rule-based, modular system. The modular, adaptive, and rule-based concept of Comic Chat offers valuable insights into the comic generation process. Specifically, it demonstrates that a limited set of modules can be used for different scenes and that meaningful comics can be generated using a rule-based system.

While these related works show promising approaches, a gap remains regarding the fully automatic generation of instructions for dynamically connected smart environments.

Comic Generation using Ambient Reflection

When dealing with systems that are not self-explanatory by design, several details of user interaction must be explained, including how certain actions can be performed and how the system is expected to respond. Self-descriptions of the components involved are one way of achieving this. Hence, we have decided to expand the Ambient Reflection framework to enable the automatic creation of comic-based instructions for smart environments (see Figure 1).

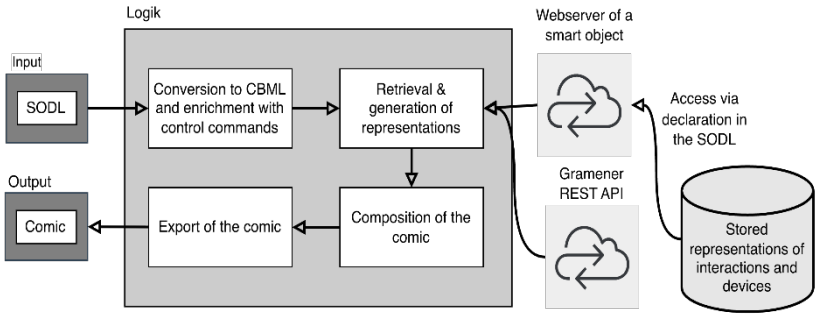

Figure 1: Conceptual design of the comic generator. Based on the ensemble's self-description (SODL), relevant information is compiled into natural language formulations and transformed into the comic descriptions (CBML). Graphics are then assembled into comic instructions, enriched with the formulations.

System Architecture and Implementation

We have implemented our comic generation system in a modular architecture. The core of the system is the *Comic Engine*, which we integrated into the Description Mediator. When ensembles are connected, the self-descriptions of all involved components are merged into a single description object and handed to the respective rendering engines. Our Comic Engine traverses this description object and compiles natural language formulations from the information and descriptions contained therein (e.g., compiling the task, input device and the output device into "Use the hand gesture controller to select an element from the menu"). They explain details of the interaction, such as how certain physical actions are to be performed or what the response of the connected remote station looks like. Formulations for several details of the interaction are then serialized using the JavaScript Object Notation

(JSON) format. The Comic Engine then invokes the *Comic Generator* as a sub-process by passing the structured JSON information and reading the process output.

The Comic Generator first transforms the provided JSON data into a Comic Book Markup Language (CBML) document. This transformation flattens the nested JSON structure derived from the SODL while preserving the essential level information (e.g., Goal, Task), resulting in a format suitable for generation.

Comic Generation Process

Next, the generation system accesses visual representations for devices and interactions, which are embedded within the self-descriptions of the respective objects. Textual content is similarly extracted from self-description and minimally adapted. For instance, the text description for the physical level is integrated into the speech bubble of the panel related to the alphabetic level of the Virtual Protocol Model. Dynamic elements, specifically characters and speech bubbles, are generated via REST API calls to Comicgen[2], a web-based, open-source comic creation software. It provides a selection of pre-made characters, emotions, poses, and angles that can be combined to generate personalized figures. The size of the speech bubbles is dynamically calculated based on the content.

The main logic resides in panel generation. To ensure structural consistency, the system uses distinct panel templates for the relevant levels of the Virtual Protocol Model (Goal, Task, Semantic, Alphabetic). These templates define specific zones within a panel for placing elements (i.e. the character, the device, and the speech bubble). To ensure a uniform aesthetic and prevent visual obstruction, objects are scaled to not exceed a maximum size. Collision avoidance is implemented using two techniques: predefined placement zones and a spiral search mechanism to resolve any remaining overlaps. This process calculates the element's bounding box and ensures placement within allowed areas. If an element collides or is out-of-bounds, a valid position is determined using the spiral search mechanism by expanding a search radius at fixed angle steps up to a maximum number of attempts.

Next, the individual panels are arranged sequentially based on the ascending IDs assigned during the transformation of the structured data into CBML. Additionally, panels are numbered to increase readability. Furthermore, a fallback mechanism ensures fault tolerance. If graphical resources are missing, the generator still produces the panel with the character and textual description, ensuring a usable output.

Ultimately, the generator exports the final instructions as a PDF document that can be sent to components in the smart environment which can present it (see an example in Figure 2).

[2]https://gramener.com/comicgen/v1 (Visited on 14.12.2025).

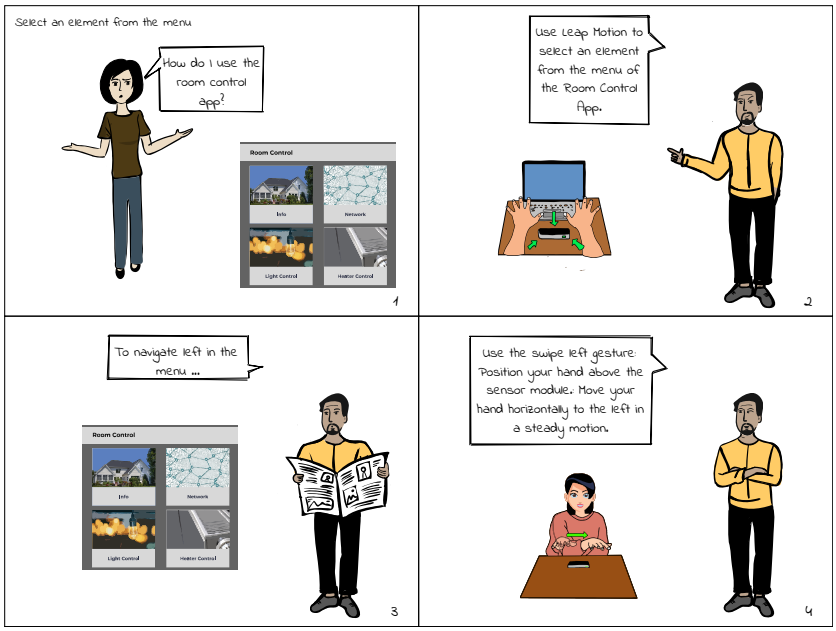

Figure 2: First four panels of a generated comic instruction.

User study

We conducted a user study to investigate the quality of the generated comic instructions. The study focused on the comprehensibility of the generated comic tutorials and the clarity of their textual and graphical elements. The emphasis was therefore placed on the quality of the instructions themselves, rather than on the system design or the usability of the ensembles being explained.

The investigation was conducted as a quasi-experimental mixed design study. The laboratory setup is visualized in Figure 3. The methodology included four practical tasks, as detailed in Table 1. Participants also completed three questionnaires. These questionnaires gathered sociodemographic data, including previous experience with smart devices. Additionally, we measured the Affinity for Technology Interaction (ATI) (Franke et al., 2019) and utilized a study-specific questionnaire to evaluate the instructions (Kordts et al., 2025). An accompanying observation was conducted throughout the tasks. This observation tracked how often instructions were consulted, whether tasks were completed successfully, and the total time taken. Errors were also logged, such as selecting the wrong devices, performing incorrect gestures, or using the wrong navigation path in the application. The study was concluded with a semi-structured interview. This interview aimed to identify any unclarities, preferences on where to display the instructions, whether all instructions were considered equally suited, and to gather further notes and comments.

The sample consisted of 13 participants, including eight employed professionals and five university students, with a mean age of 27 years (SD 3.4 years). While the participants exhibited a slightly above-average affinity for technology ($M = 3.93$, $SD = 1.33$, Cronbach's Alpha $= 0.97$), the majority

had no prior experience with the specific interaction devices used in the study. However, fundamental smart home concepts like voice control and smart lighting were familiar to most. The ATI was assessed using the questionnaire developed by Franke et al. (2019). According to the same study, the mean value for the German population is M = 3.6 (SD = 1.08).

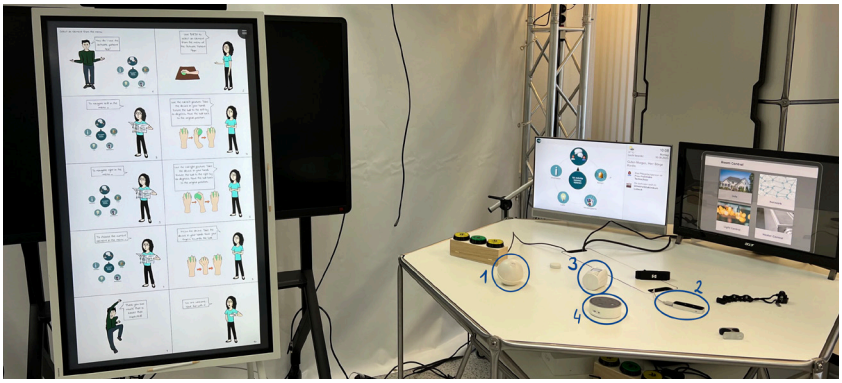

Figure 3: Laboratory setup during the evaluation of the comprehensibility and aspects of the automatically generated comic instructions. The image highlights the devices used, which were to be selected by the participants. 1 = BIRDY, 2 = leap motion, 3 = smart thermostat (FRITZ!DECT 301), 4 = amazon echo dot.

Table 1: The tasks and the required components during the participant observation. Note, that the instructions were always displayed on the same monitor.

Task	Involved Components	Description
1	Ball-Shaped Interaction Device (BIRDY), AAC Application	Read the comic instruction provided to find how to control the application. Identify the required devices (selection on a table). Use the instruction to switch on the light and then change the color of the light to blue.
2	Hand Gesture Controller (Leap Motion), Room-Control Application	Read the comic instruction provided to find how to control the application. Identify the required devices (selection on a table). Use the instruction to switch on the light and then change the color of the light to blue.
4	Hand Gesture Controller (Leap Motion), Room-Control Application, Smart Thermostat (FRITZ!DECT 301)	Read the comic instruction provided to find out how to change the temperature of the heater. Identify the required devices (selection on a table) and perform the necessary actions to set the temperature to "Comfort".
5	Voice Controller (Amazon Echo Dot), Smart lights	Read the comic instruction provided to find out how to control the lamp. Identify the required devices (selection on a table) and perform the necessary actions to switch the lamp on and off once.

During the study, participants were required to complete four tasks that addressed different forms of interaction. The first task involved gesture control using a ball-shaped interaction device (BIRDY) to control an application via tilting and pressing gestures. The second task focused on freehand gesture

control, requiring participants to operate a room control application using a hand gesture controller (Leap Motion). To assess system comprehension, the third task expanded upon the second ensemble by adding a smart thermostat, and participants had to explain the resulting system behavior. The final task centered on voice control, where a smart lamp was to be operated using a voice user interface (Amazon Echo Dot). For each task, participants first viewed the corresponding comic tutorial, then identified the correct device from a selection of nine and performed the required interaction. System responses were simulated by the researcher using the wizard-of-Oz method to ensure a fluid and natural interaction experience for the participants.

The results of the study were consistently positive. All 13 participants successfully completed every task, and in no instance an incorrect device was selected. Furthermore, all participants were able to explain the interplay between the devices and the resulting system reactions, demonstrating an understanding of the ensemble's functionality. The overall error rate was low; the most frequent errors occurred during gesture execution with the hand gesture controller in the second task, with an average of 0.77 errors per person. The time required to complete the tasks indicated a learning effect, with task duration decreasing over time, except for the more complex second task, which was the most time-consuming at an average of 132.9 seconds. The fourth task (voice control) was completed the fastest, at an average of 33 seconds.

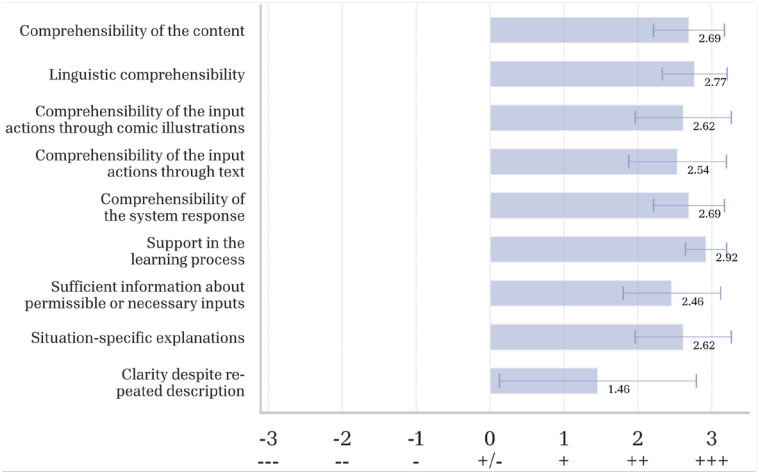

Figure 4: Average rating of the individual aspects of the generated comic instructions.

The comic tutorials themselves received very favorable ratings from the participants. Their length was judged to be appropriate (M = 0.46 on a scale from –3 'too short' to +3 'too long'). For the remaining items, agreement with the respective statements was expressed (on a scale from –3 for "strongly disagree" to +3 for "strongly agree"). Both linguistic comprehensibility (M = 2.77) and content comprehensibility (M = 2.69) achieved high scores. The participants found the input actions to be highly understandable through both the comic illustrations (M = 2.62) and the accompanying texts (M = 2.54). The highest rating was given for the "support in the learning process" (M = 2.92), highlighting the effectiveness of the comic format. The

only minor point of criticism was the "clarity despite repeated description of similar goals", which, with a score of M = 1.46, was still rated positively. The results are shown in Figure 4. Qualitative feedback from the interviews revealed that the consistent structure of the tutorials was explicitly rated positively. When asked about the ideal placement for such instructions, most participants favored access via QR codes on the smart objects, which would then display the tutorial on a personal smartphone or tablet.

DISCUSSION

Our findings indicate that automatically generated comic tutorials can effectively support the operation of ensembles within smart environments. Notably, the gesture execution errors observed with the Leap Motion device were attributed to some users skimming the instructions and overlooking panels with more detailed explanations. This suggests that future implementations could benefit from encouraging users to engage more completely with the entire tutorial. The combination of visual representation and text proved to be effective, and the textual instructions were reported to be especially helpful when resolving errors.

The evaluation methodology entails several limitations. The study exclusively involved young adults meaning influences of age, technological affinity, or educational background on the assessment cannot be precluded, although this group represents a significant user base. The selection of devices and interactions was restricted, limiting the generalizability of the findings to other distributed environments or more complex gestures. The use of a Wizard-of-Oz simulation lacked real system feedback, potentially distorting natural user behavior. Furthermore, the laboratory setting excluded real-world variables, limiting transferability to private contexts. Methodological constraints also include the sample size, recruitment strategy, and the qualitative interview format.

CONCLUSION

In this paper, we introduced a comic generation rendering engine for the Ambient Reflection framework. Based on the self-description of all the components involved, user instructions can be generated for dynamically connected ensembles. These ensembles may consist of smart objects and applications, which are executed within the smart environment.

Additionally, we conducted a user study to address our research question to what extent the generated comic instructions can explain the operation of ensembles in smart environments to users. The participants successfully completed all the tasks set. Quantitative results indicate that the instructions are clear, understandable and particularly support the learning process. In the interviews, participants positively commented on the structure of the comic instructions. The results indicate that our approach is suitable to generate comprehensible comic instructions—provided a good self-description of the components involved, including adequate depictions in comic optics. Nonetheless, the described limitations leave room for future studies.

Future work should address some shortcomings and expand the system's capabilities. Particularly, extended automated generation possibilities should be explored. The system currently requires that interaction representations are explicitly defined in the SODL. Future work could focus on tools assisting manufacturers in creating interaction depictions to advanced AI-based generators, though the latter involves a certain effort and risks regarding consistency.

Another focus should also be on output medium. This may require moving beyond static PDFs to interactive comics.

REFERENCES

Boucher, M., Bach, B., Stoiber, C., Wang, Z., & Aigner, W. (2023). Educational data comics: What can comics do for education in visualization? *2023 IEEE VIS Workshop on Visualization Education, Literacy, and Activities (eduVis)*, 34–40.

Burmeister, D. (2018). *Selbstreflexive Geräteverbünde in smarten Umgebungen* [PhD thesis]. University of Lübeck, Germany.

Fadiga, K., Houzé, E., Diaconescu, A., & Dessalles, J.-L. (2021, September). To do or not to do: Finding causal relations in smart homes. *2021 IEEE International Conference on Autonomic Computing and Self- Organizing Systems (ACSOS)*.

Franke, T., Attig, C., & Wessel, D. (2019). A Personal Resource for Technology Interaction: Development and Validation of the Affinity for Technology Interaction (ATI) Scale. *International Journal of Human–Computer Interaction*, *35*(6), 456–467.

King, E. (2024). *Continuous discovery and goal-oriented control of smart devices in mobile environments* [PhD thesis], The University of Texas at Austin, USA.

Kordts, B., Brandl, L. C., & Schrader, A. (2025). Managing self-explaining ambient applications. *Frontiers in the Internet of Things*, *Volume 4 - 2025*.

Kurlander, D., Skelly, T., & Salesin, D. (1996). Comic Chat. *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, 225–236.

Nielsen, J. (1986). A virtual protocol model for computer-human interaction. *International Journal of Man-Machine Studies*, *24*(3), 301–312.

Sadeghi, M., Herbold, L., Unterbusch, M., & Vogelsang, A. (2024). SmartEx: A Framework for Generating User-Centric Explanations in Smart Environments. *2024 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 106–113.

Stanton, N. A. (2006). Hierarchical task analysis: Developments, applications, and extensions. *Applied Ergonomics*, *37*(1), 55–79.

Suh, S. (2023). Cheat sheet for teaching programming with comics: Through the lens of concept-language-procedure framework. *arXiv Preprint arXiv:2306.00464*.

Suh, S., Lee, M., Xia, G., et al. (2020). Coding strip: A pedagogical tool for teaching and learning programming concepts through comics. *2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 1–10.

Suh, S., Zhao, J., & Law, E. (2022). CodeToon: Story Ideation, Auto Comic Generation, and Structure Mapping for Code-Driven Storytelling. *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, 1–16.

Sumi, Y. (2017). ComicQA: Contextual navigation aid by hyper-comic representation. *Proceedings of the 19th International Conference on Information Integration and Web-based Applications & Services*, 76–84.

Walsh, J. A. (2012). Comic Book Markup Language: An Introduction and Rationale. *Digital Humanities Quarterly*, *6*(1).