

User Agency in Runtime Malleable Software Under Change Aversion

XINQI ZHANG, Santa Clara University, USA

BRYAN MIN, University of California San Diego, USA

SHIZHAO (LAWRENCE) LIU, Santa Clara University, USA

KAI LUKOFF, Santa Clara University, USA



Fig. 1. From fixed UI variants to runtime UI. Traditional interfaces are designer-defined, while malleable and runtime systems expand the UI/UX space, enabling users and AI to co-shape interface structure and behavior through real-time interaction.

CCS Concepts: • **Human-centered computing**; • **Human computer interaction (HCI)**; • **Interactive systems and tools**;

Additional Key Words and Phrases: Generative UI, Runtime UI, User Agency, Change Aversion, Malleable Software

1 Introduction

Interactive interfaces have evolved from uniform, one-size-fits-all designs toward increasingly adaptive and personalized systems that respond to user intent and context [13, 20]. While these shifts aim to better support diverse needs, each step toward greater flexibility also introduces new interaction costs. Uniform interfaces emphasize consistency but often fail to accommodate varied contexts of use [15]. Adaptable interfaces increase user control but introduce cognitive and behavioral overhead, such as difficulty committing to changes [2, 17]. Adaptive interfaces automate adjustment through user modeling and machine learning, reducing manual effort but raising concerns about unpredictability and uneven performance [5, 8]. Across these paradigms, a central concern is user agency, the users’ ability to understand, influence, and control system behavior, which is critical to trust, learnability, and long-term adoption.

Recent advances in generative AI introduce a new paradigm: **runtime interfaces (runtime UI)**, in which interface structures can be generated dynamically from users’ real-time intent through mixed-initiative interaction. Enabled by generative UI (Gen UI), this supports prompt-driven end-user customization (analogous to “end-user vibe-coding”). [1, 7]. A simplified workflow is:

User Prompt → AI interprets intent → UI code is generated → Interface is rendered at runtime

In this model, users can request interface changes at runtime, shifting part of interface construction from designers to end users. This enables deeper personalization but also introduces reliability challenges. Because generative systems are probabilistic, they may produce inconsistent outputs, undermining learnability, which are critical for trust. [3, 15, 16].

From a user-adoption perspective, runtime interfaces create a central tension: they expand personalization and user agency while increasing the risk of disruptive change and user resistance. They also introduce new risks by shifting design-like responsibilities to users through prompt-based customization. This position paper focuses on malleable software on runtime and asks: How can runtime interface systems minimize change aversion while enabling deep personalization without turning users into designers?

2 Runtime UI Opportunities Under Change Aversion

Our position draws on prior literature on change aversion. Users often react negatively to interface redesigns simply because they introduce change, even when the redesign benefits newcomers [19]. Such reactions are commonly explained by status quo bias and loss aversion, as well as perceived loss of control and increased cognitive and attentional costs during relearning [2, 18].

These findings suggest that user agency depends not only on enabling control but also on limiting disruptive interface change and preserving stable interaction structures. For runtime UI, this implies a core constraint: because GenUI can produce frequent and large interface mutations, adaptation should be explicitly bounded and stabilized.

At the same time, runtime UI expands opportunities for user agency by enabling post-deployment interface tailoring [6, 22]. Historically limited by the coding gap [14], such tailoring is becoming more feasible through agentic coding tools. Following the three levels of end-user tailoring [12], runtime UI supports three forms of control:

Level 1: Customization. Runtime UI enables styling and layout customization (e.g., theme, font, color). For example, users can customize the theme of a video platform through a prompt such as: “Make the site use a Disney or Marvel theme.” Customization can improve comfort, increase perceived control and ownership, and support user agency more strongly than system-driven personalization [11, 21].

Level 2: Integration. Runtime UI supports user-defined functional connections across services and components, allowing users to reconfigure workflows without source-code access. This form of tailoring helps users better align tools with situated practices [4]. For example, a user could restructure a school resource management site (e.g., Canvas) with a prompt such as: “Move the financial section to the homepage because I will be a TA this quarter,” thereby reducing repetitive navigation steps.

Level 3: Extension. Runtime UI enables users to add new capabilities to a system, positioning them as co-designers and improving task fit and autonomy through domain-specific augmentation [6, 14]. For example, a user could add a time-management extension to a social media feed with a prompt like: “Create a panel that tracks the types of short videos I watch and alert me if more than half are meme-related.”

In summary, runtime UI can strengthen user agency by enabling direct, in-situ UI/UX modification, but it also increases the risk of destabilizing change. Because change aversion can undermine trust and adoption, runtime UI should bound adaptation through intelligible constraints that preserve familiarity.

3 Design Provocations: Customization Without Turning Users into Designers/Developers

Beyond change aversion, prompt-driven runtime UI introduces another practical challenge: how to support effective customization without turning users into designers or developers. Although users benefit from customization, they generally do not want to assume ongoing designer or developer responsibilities [9, 10]. Runtime mutations can range from lightweight personalization to functional restructuring. Without explicit guardrails, open-ended prompt changes may disrupt core interaction flows or violate product logic. To operationalize such guardrails in runtime UI, in addition to constraints derived from change aversion, it is also essential to explore how to protect *invariant features*—core functional elements and value-defining interaction pathways.

To guide discussion, we propose two provocations for Gen UI design:

- (1) How can prompt-driven Gen UI systems support user agency without transferring design burden to users?
- (2) What constraint and invariant patterns best reduce change aversion while still enabling meaningful runtime adaptation?

4 Acknowledgements

The authors used OpenAI’s ChatGPT (GPT-5) to assist with language editing, including grammar correction, phrasing refinement, and formatting in LaTeX. All conceptual content, arguments, and structure were developed by the authors. The use of AI assistance complied with ACM’s policy on the responsible use of generative AI tools in publications. The authors take full responsibility for the content of this work.

References

- [1] Saleema Amershi et al. 2019. Guidelines for human-AI interaction. In *CHI*. 1–13.
- [2] Brian P. Bailey and Joseph A. Konstan. 2006. On the need for attention-aware systems. In *CHI*. 11–20.
- [3] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots. In *FAccT*. 610–623.
- [4] Michael Bolin, Matthew Webber, Philip Rha, Tom Wilson, and Robert Miller. 2005. Automation and Customization of Rendered Web Pages. In *Proceedings of UIST*. 163–172.
- [5] Peter Brusilovsky and Mark Maybury. 2002. From adaptive hypermedia to the adaptive web. *Commun. ACM* 45, 5 (2002), 30–33.
- [6] Gerhard Fischer and Elisa Giaccardi. 2006. Meta-design: A Framework for the Future of End-User Development. *End User Development* (2006), 427–457.
- [7] Eric Horvitz. 1999. Principles of mixed-initiative user interfaces. In *CHI*. 159–166.
- [8] Anthony Jameson. 2003. Adaptive interfaces and agents. In *The Human-Computer Interaction Handbook*. 305–330.
- [9] Andrew Ko and Brad Myers. 2006. Barriers to end-user programming systems. In *VL/HCC*. 199–206.
- [10] Wendy E. Mackay. 1990. Patterns of sharing customizable software. *Proceedings of CSCW* (1990), 209–221.
- [11] Sampada Marathe and S. Shyam Sundar. 2011. What Drives Customization? Control or Identity?. In *Proceedings of CHI*. 781–790.
- [12] Anders Mørch. 1995. Three Levels of End-User Tailoring. *Computers and Design in Context* (1995), 51–76.
- [13] Brad A. Myers. 1998. A brief history of human-computer interaction technology. *interactions* 5, 2 (1998), 44–54.
- [14] Brad A. Myers, Andrew J. Ko, and Margaret M. Burnett. 2006. The Past, Present, and Future of End-User Programming. *Commun. ACM* 49, 9 (2006), 44–50.
- [15] Jakob Nielsen. 1994. *Usability Engineering*. Morgan Kaufmann.
- [16] Don Norman. 2013. *The Design of Everyday Things*. Basic Books.
- [17] Reinhard Oppermann. 1994. *Adaptive User Support*. Lawrence Erlbaum.
- [18] William Samuelson and Richard Zeckhauser. 1988. Status Quo Bias in Decision Making. *Journal of Risk and Uncertainty* 1, 1 (1988), 7–59. doi:10.1007/BF00055564
- [19] Aaron Sedley and Hendrik Müller. 2013. Minimizing change aversion for the Google Drive launch. In *CHI ’13 Extended Abstracts on Human Factors in Computing Systems (CHI EA ’13)*. ACM, New York, NY, USA, 2351–2354. doi:10.1145/2468356.2468767
- [20] Ben Shneiderman. 1983. Direct manipulation: A step beyond programming languages. *IEEE Computer* 16, 8 (1983), 57–69.
- [21] S. Shyam Sundar and Sampada S. Marathe. 2010. Personalization versus Customization: The Importance of Agency, Privacy, and Power Usage. *Human Communication Research* 36, 3 (2010), 298–322.
- [22] Randall Trigg and Susanne Bødker. 1994. From Implementation to Design: Tailoring and the Emergence of Systematization. In *Proceedings of CSCW*. 45–54.