User Preferences in Graphical and Textual Interfaces for Kubernetes

The Authors

August 19, 2021

1 Introduction

We seek to answer the following research questions and determine the validity of our hypotheses:

RQ1. How does tool preference vary by tool type, input type frequency, subtask, and perceived cognitive load?

H1. Preference correlates with input type frequency...

RQ2. How does perceived cognitive load vary by tool type, subtask, and input type switching frequency?

H2. Perceived cognitive load correlates with tool type...

2 Related Work

Effects of a hybrid interface and multimodal features for learning how to program [10, 6].

Benefits of a textual versus graphical interfaces for learning how to program [1].

Measuring perceived cognitive load using the NASA-TLX scale [7].

User evaluation methods and the impact of think-aloud on user testing [8, 2, 4].

Task analysis on the challenges of shell programming [5].

Optimizing visual and textual information and predicting user frustration in search user interfaces [9, 3].

motivate relevance of each and illuminate open research avenues that stem from their findings

3 Study Setup

We conducted a remotely moderated, within subjects user study with X participants who had an average of X years of programming experience. All participants belonged to the

Hybrid Cloud research group at IBM and regularly used Kubernetes in their work. Since the purpose of our study is not to learn the startup cost of using a tool, we recruited participants that had some familiarity with the tools.

Before the study sessions, participants filled out a background survey to provide information about their programming experience, use of Kubernetes, and familiarity with the two tools. After completing the study, participants filled out a follow up survey and ranked the tools based on their preference. In the follow up survey, participants were also asked to rate the tools based on their perceived cognitive load using the NASA-TLX scale. To give us a better understanding of preference, participants also rated the quality of each tool on a five-point Likert scale and were asked to give reasons for why they either liked or didn't like using each tool. Participants were also asked to share their recommendations for how each tool could be improved.

summarize the questions asked in the surveys in a more general/thematic way

During the study sessions, participants completed a Kubernetes task using the kubectl CLI and the OpenShift console. The order in which participants used the tools was counter balanced to mitigate learning bias between sessions, with half of the participants using the kubectl CLI first and the other half using the OpenShift console first []. We employed a classic "Think Aloud" user evaluation method during the sessions to avoid the effects of reactivity []. Video and audio from the sessions were recorded for later transcription and analysis.

3.1 Pilot Testing

We conducted pilot testing with two participants to identify any issues in our study design. One participant identified a misleading error in one of the task instructions so we rewrote it. Initially, we had only provided task hints for the kubectl CLI, and one participant realized that they could not apply these hints to the OpenShift console, so we also included task hints for the OpenShift console. We found that both participants were able to complete the task using each tool in under 20 minutes.

3.2 Kubernetes Task

The goal of the Kubernetes task is to create an application in a namespace, find the deployment pods that have the string "magic key" in their logs, delete the application, and then delete the namespace. This task was chosen for our study because it is simple enough to be accomplished by a Kubernetes novice in multiple steps or by an expert in fewer steps and it requires participants to perform search and navigation using each tool. Participants were given a GitHub repo with the application files and task instructions. If participants got stuck during the session, hints specific to each tool interface were provided in links below the task instructions. Participants were told to think aloud during the session and given a prompt to continue thinking aloud if they fell silent for more than 15 seconds.

4 Evaluation Methodology

4.1 Low Level User Interactions

A challenge we faced was in deciding what metrics to use to compare these tools. We sought to use a WhatPulse to measure physical user interaction frequencies and produce mouse movement heatmaps, however we observed errors in the numbers and decided not to use it []. Furthermore, to what extent do heatmaps let us compare between tools with different input methods? Heatmaps could be a good metric for A/B testing, however, they would be a poor metric for comparing different tools. We assume that these tools are already well designed and instead seek to study the limits that exist within the steady state of design and learning. These modalities will have certain limits and we seek to understand what are they as is to determine how we can improve them.

Wall clock time is easy to measure and is objective, however, across these tools isn't really comparable because of the user variance we observed in the participant workflows. For example, one participant used the "Topology" view to access the logs for each deployment's pods (shown in Figure X) while other participants used the list of pods found in the "Project Inventory" to access the logs (shown in Figure X). User preference is a better metric, however, it may be too subjective. For example, knowing people prefer a CLI for certain tasks doesn't help to improve the browser console for those tasks. Therefore, to support the qualitative data gathered in our surveys and study sessions, we also collected low level user interaction data, specifically user input types (keys, clicks, and mouse movements) and how frequently the participant used each user input type in completing the task on both tool interfaces.

The video recordings were manually analyzed to count how frequently participants used the keyboard to enter a command (k), how frequently participants clicked (c), and how frequently participants moved their mouse and/or scrolled to click (m). Participant input type frequencies were stored in tuples as (k, c, m). Rather than count characters to tally the keyboard input, we chose to count the number of total commands entered to normalize the input frequencies as we are neither counting the duration of the clicks nor the length of the mouse movements. In the OpenShift console, we counted commands as the number of times participants typed something followed by an enter or click to enter. Mouse movements were only noted when they were followed by a purposeful click (for example, moving the mouse to click on the browser back button), and otherwise random mouse movements were not counted in our analysis.

4.2 High Level User Interactions

As we observed in our study sessions, participants do not consider these inputs as having equal execution difficulty. For example, while half of our participants copy/pasted the pod names from the list of pods in the terminal to get the logs for each pod, the other half used tab completion to get the pod names and execute the command to get the logs. Therefore,

we also kept track of the specific commands used by participants, such as how frequently participants used copy/paste or tab completion. Table X provides a list of all the input measures we collected for analysis and their exact definitions.

Furthermore, Regular expressions were also used to create shortened descriptions of the patterns of input types used to complete the task with each tool. Table X provides a list of the regular expressions participants used to complete the task with each tool.

4.3 Interpretation

4.3.1 Exhaustion

In support of the qualitative data gathered from the surveys on participants' perceived cognitive load, we collect the user input frequencies as a measure of exhaustion due to the total number of interactions required to complete the task. These frequencies are shown for each participant in Tables X-X.

4.3.2 Dissonance

To further support the qualitative data gathered from the surveys on participants' perceived cognitive load, we kept track of the specific commands participants used as well as the inputs that make up those commands and patterns of switching between different inputs as a measure of dissonance due to the number of times the participant had to task switch to complete the task.

5 Results

P1	Copy/Paste	Keyboard	Click	Page Navigation
kubectl CLI	6	15	8	22
OpenShift console	0	2	77	80

P2	Copy/Paste	Keyboard	Click	Page Navigation
kubectl CLI	7	28	20	49
OpenShift console	0	13	145	172

P3	Copy/Paste	Keyboard	Click	Page Navigation
kubectl CLI	1	17	1	3
OpenShift console	4	5	142	192

P4	Copy/Paste	Keyboard	Click	Page Navigation
kubectl CLI	0	32	2	1
OpenShift console	4	14	133	137

P1	kubectl CLI	OpenShift console

P2	kubectl CLI	OpenShift console

P3	kubectl CLI	OpenShift console

P4	kubectl CLI	OpenShift console

6 Discussion

7 Future Work

8 Conclusion

References

[1] Dillon, E., Anderson, M., and Brown, M. Comparing mental models of novice programmers when using visual and command line environments. In *Proceedings of the 50th Annual Southeast Regional Conference* (2012), pp. 142–147.

[2] Ericsson, K. A., and Simon, H. A. *Protocol analysis: Verbal reports as data.* the MIT Press, 1984.

[3] Feild, H. A., Allan, J., and Jones, R. Predicting searcher frustration. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval* (2010), pp. 34–41.

[4] Fox, M. C., Ericsson, K. A., and Best, R. Do procedures for verbal reporting of thinking have to be reactive? a meta-analysis and recommendations for best reporting methods. *Psychological bulletin 137*, 2 (2011), 316.

[5] Gandhi, I., and Gandhi, A. Lightening the cognitive load of shell programming.

[6] Grafsgaard, J. F., Wiggins, J. B., Vail, A. K., Boyer, K. E., Wiebe, E. N., and Lester, J. C. The additive value of multimodal features for predicting engagement, frustration, and learning during tutoring. In *Proceedings of the 16th International Conference on Multimodal Interaction* (2014), pp. 42–49.

[7] Hart, S. G., and Staveland, L. E. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, vol. 52. Elsevier, 1988, pp. 139–183.

[8] Mcdonald, S., Cockton, G., and Irons, A. The impact of thinking-aloud on usability inspection. *Proceedings of the ACM on Human-Computer Interaction 4*, EICS (2020), 1–22.

[9] Treharne, K., Powers, D. M., and Leibbrandt, R. Optimising visual and textual in search user interfaces. In *Proceedings of the 24th Australian Computer-Human Interaction Conference* (2012), pp. 616–619.

[10] Unal, A., and Topu, F. B. Effects of teaching a computer programming language via hybrid interface on anxiety, cognitive load level and achievement of high school students. *Education and Information Technologies*, 1–19, year=2021, publisher=Springer.