User Preferences in Graphical and Textual Interfaces for Kubernetes

The Authors

August 25, 2021

1 Introduction

A challenge we faced was in deciding what metrics to use to compare these tools. We sought to use 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 website design, however, they would be a less useful 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 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.

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 4 participants who had an average of 20 years of programming experience between them. All participants belonged to the Hybrid Cloud research group at IBM, regularly used Kubernetes in their work, and had prior experience using 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. Participants downloaded the necessary tools onto their machines (kubectl CLI, OpenShift Console) and either used or were given access to an OpenShift cluster to run Kubernetes. 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. To give us a better understanding of preference, participants also rated the quality of each tool 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.

During the study sessions, participants completed a Kubernetes task using the kubectl CLI and the OpenShift console. The goal of the Kubernetes task was 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. Participants were given a GitHub repo with the application files and task instructions. No time limit was set to complete the task. If participants got stuck while completing the task, hints specific to each tool interface were provided in hyperlinks below the task instructions. Half of the participants used the kubectl CLI first and the other half used the OpenShift console first. We employed a classic "Think Aloud" user evaluation method in which participants were

asked to think aloud while completing the task and given a prompt to continue if they fell silent for more than 15 seconds. Video and audio from the sessions were recorded using Webex for later transcription and analysis.

4 Evaluation Methodology

4.1 Low Level User Interactions

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 scrolled to click (M). In the terminal, we counted the total number of commands entered to determine how frequently participants used the keyboard. In the OpenShift console, we counted the number of times participants typed followed by an enter or click to enter, as the total number of commands entered to determine how frequently participants used the keyboard. In both interfaces, we counted how frequently participants clicked by counting the number of times participants clicked to copy/paste text. In the OpenShift console, we also counted the number of times participants clicked a button. In both interfaces, mouse movements were only noted when they were followed by a click to copy/paste or followed by a button click and we counted every time participants scrolled up or down.

4.2 High Level User Interactions

We kept track of the specific commands used by participants, such as how frequently participants used copy/paste or tab completion. Regular expressions were used to create shortened descriptions of the input type patterns used to complete the task with each tool. The regular expressions were composed using the letters that represent the input type (K, C, M). We composed the input type patterns used to complete the following subtasks for each tool interface: command execution, copy/paste, navigation, and tab completion.

4.3 Interpretation

4.3.1 Exhaustion

We collected participant 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 2–5. The total input frequencies for each tool interface are shown in Table 6.

Recommendations	kubectl CLI	OpenShift Console
P1	Cryptic syntax for many commands, hard to do easy things like switch contexts	More developer focused tools
P2		
P3	Seems to be doing what it is supposed to do. Minor problems can be super annoying though like this one: `https://github.com/kubernetes/kubernetes/issues/42552`	More scripting and automation? Plugins?
P4	Better documentation	NA (not using it)

Table 1: Participant recommendations for improving tools.

P1	Clicks	Commands	Mouse Movements
kubectl CLI	8	21	22
OpenShift Console	77	2	80

Table 2: P1 input frequencies.

4.3.2 Disruption

We kept track of the patterns of switching between different inputs (composed as regular expressions) as a measure of disruption due to the number of times the participant had to switch modalities (for example, between typing and mousing) to complete the task. Tables 7–10 provide a list of the regular expressions participants used to complete the task with each tool.

5 Results

All four participants ranked the kubctl CLI as the tool interface they preferred to use to complete the task.

P2	Clicks	Commands	Mouse Movements
kubectl CLI	20	35	49
OpenShift Console	145	13	172

Table 3: P2 input frequencies.

P3	Clicks	Commands	Mouse Movements
kubectl CLI	1	18	3
OpenShift Console	142	9	192

Table 4: P3 input frequencies.

P4	Clicks	Commands	Mouse Movements
kubectl CLI	2	32	1
OpenShift Console	133	18	137

Table 5: P4 input frequencies.

Total	Clicks	Commands	Mouse Movements
kubectl CLI	31	106	75
OpenShift Console	497	42	581

Table 6: Total input frequencies.

P1	kubectl CLI	OpenShift Console
Command Execution	KK	KMC
Copy/Paste	MMCMMKK	-
Navigation	M	MC
Tab Completion	-	-

Table 7: P1 regular expression patterns.

P2	kubectl CLI	OpenShift Console
Command Execution	KK	KMC
Copy/Paste		
Navigation		MC
Tab Completion	KKK	-

Table 8: P2 regular expression patterns.

P3	kubectl CLI	OpenShift Console
Command Execution	KK	
Copy/Paste		
Navigation		MC
Tab Completion		-

Table 9: P3 regular expression patterns.

P4	kubectl CLI	OpenShift Console
Command Execution	KK	
Copy/Paste		
Navigation		MC
Tab Completion		-

Table 10: P4 regular expression patterns.

6 Design Implications

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.