A.6 Usability Evaluation Questionnaire Results

This section contains the results of the usabilty questionnaire.

A.6.1 System Usability Scale

The following table contains the answers of the different questions for the SUS section of the questionnaire, where 1 means “Strongly Disagree” and 5 means “Strongly Agree”.

Participant	I think that I would like to use this system frequently.	I found the system unnecessarily complex.	I thought the system was easy to use.	I think that I would need the support of a technical person to be able to use this system.	I found the various functions in this system were well integrated.	I thought there was too much inconsistency in this system.	I would imagine that most people would learn to use this system very quickly.	I found the system very cumbersome to use.	I felt very confident using the system.	I needed to learn a lot of things before I could get going with this system.
1	3	2	4	2	4	2	2	4	2	2
2	4	1	4	1	4	2	5	2	4	1
3	4	2	4	3	4	2	3	2	4	2
4	4	2	3	1	4	3	3	2	5	1
5	3	2	4	4	4	3	4	2	3	4
6	3	2	4	3	4	2	4	2	4	5
7	4	2	4	2	4	3	4	2	4	2
8	4	3	4	1	4	1	3	2	4	4
9	4	3	3	2	5	2	5	2	4	2
10	4	3	3	3	4	1	4	4	3	3

A.6.2 Qualitative Questions

The following section contains the answers of the qualitative questions from the questionnaire.

"What aspects of the software did you find most useful or intuitive?"

1. UI to configure, Grafana (overall).

2. A clear win for me is the visualization of an experiment. That's such an extraordinary feature I fell in love with immediately. It provides instant feedback on the experiment configuration and the resulting execution timeline. That is a tremendous help to validate or analyze both the configuration and the results. The second aspect that provides a lot of benefit to me is the option to easily switch between a form view and a code editor for the configurations. The benefit of integrated code editing capabilities also extends to the implementation of the actual work, too. Thirdly, even though it wasn't required to use in the guided evaluation, the versioning feature allow for stress-free experimentation with variations of an experiment. Therefore, it's a capability I appreciate quite a bit.
3. The graph projecting the experiment over time is very helpful to get an idea of what will happen and when it will happen.
4. I liked the visualization of load and failures in one graph. It helps to understand what load and failure scenario you are about to execute. It creates a nice mental image of the defined configuration parameters.
5. Setting goals of the experiment.
6. I think the strength of the system is that you can configure a lot of details and get a lot of control over the experiment (which makes it not intuitive, but I do not think it has to be). Aside, I really like the diagram which allows me to see the load and timings of failures together.
7. The sections for creating the experiments. The 2-D plot visualizing the experiments.
8. Very well documented, documentation directly linked and accessible in the tool.
9. Visual UI over config files.
10. UI.

“"Were there any features or workflows that you found confusing or frustrating?”

1. Unclear how experiment config needs to be overwritten, no relative timestamps in Grafana, data from failures/chaos injection not visible in Grafana
2. A bit confusing to me, not much though, was how settings that could be attributed to the same configuration category (failures, load, work) were spread across the user interface. For example, the warm-up configuration is available only after clicking a hamburger icon button, while a cog icon also provides access to load settings. And that after having already specified the load test duration and the maximum arriving users per second in a form when creating a new experiment. I don't perceive it as an issue per se. Only if it were a purchasable off the shelf product, I'd wish for a cleaner user interface with dedicated sections that have little or no category overlap. Perhaps the choice to use a wizard guiding new users through their configuration of each individual aspect of an experiment would provide additional benefits to make such a product more approachable, too. Also, in some areas, the documentation did not match the actual user interface. Allowing for the documentation to open in a separate browser window/tab instead of hiding the user interface would be as helpful as opening links from the documentation in new windows/tabs by default (target="blankör something). Lastly, I found it not that intuitive to "Clear the config by creating a new failure set with a new failure configuration that has only null values for the target service".

3. Help text are overwhelming. Dedicated smaller help texts directly at the corresponding options might be easier to grasp. Overview help would be needed still. This should be no larger than a paragraph or two and should include a picture of the feature, because at the time of reading the help text, the UI is hidden.
4. 1. Definition of numbers across UI in different units. 2. Warm up integrated in a hideous way. 3. The extraction of load information from the Gatling definition (affects usability of the tool for other use cases and also understandability for Gatling-experienced users).
5. Failure Sets should be phases.
6. I could not find the button for setting the color for experiment goals. After inspecting the page, it turned out it has the height of 0 and is not displayed for me. Ignoring this (probably) implementation issue, it was frustrating to find the right place in the code for setting the pause in the abortedBuyProcessScenario. Some text highlighting or a search could be useful. I got also confused by the length of the experiment - I set it to 180, but the diagram ends at 205. I did not get at first that the load profile field in the configure load dialog is a dropdown. It took me some time to find the configure load and warm up dialogs. I did not expect them giving the icon.
7. Mostly the monitoring part. I wished to be able to directly query the logs. Or a more fine-grained control over graphs.
8. It took me about 5 min to find the “Configure Experiment Load”; it feels a bit unintuitive for failures to be persistent, even when their failure group is already done; there is too much information on the screen.
9. -
10. Durations, Pause is not clear

“Were there any features or workflows that your where missing?”

1. No live experiment execution data, syntax highlighting.
2. More input validation, and more pre-filled selectors to choose valid input from would be great. For example, service name selectors, Python module selectors, et cetera. Typos can happen and might be difficult to debug. As for the code editors, anything that helps writing the code properly could be useful, too. For example, code completion, syntax highlighting, linting. I know too little about the Chaos Toolkit and the MiSArch Experiment Configuration. From the evalutation experiment, I cannot tell right away what the breadth of supported infrastructure components is. Obviously, the more pieces of an environment are covered, the more versatile, thus, attractive the tool could become.
3. The definition of a fix “arriving users per second” should be an overlay over the recording of real usage, so you can define multiple overrides and still manage (e.g. delete) them later on – even after saving the experiment definition.
4. Experimental related configurations e.g., how many repetitions, what output analysis to conduct, are there variations, etc.
5. Start designing experiment from scratch.

6. I did miss some interaction on the diagram, in particular linking (e.g., clicking on the failure set brings me to its specification and the other way around). It would also be great to explicitly set the end of a failure and see the time it is active in the diagram.
7. I wished to see a visualization of the architecture. It would have been nice to specify the expected recovery times or degradation on architectural elements.
8. A general system overview would have been cool, visualizing which service is "attacked" during which time.
9. -
10. Dynamic UI to change the durations in the graph.

"Did the software support your understanding of chaos engineering concepts?"

1. Yes.
2. Yup, it did.
3. Yes.
4. Yes.
5. Yes.
6. Not that much, but I already knew a lot.
7. I would say yes, however, I wished the question would make it explicit what concepts specifically.
8. Yes.
9. -
10. Yes.

"Did the software support your understanding of load tests?"

1. Yes-ish.
2. Yup, it did that, too.
3. Yes.
4. Partially, the software makes a split of work and load concepts different from my own understanding. With work we intend to characterize the type of requests or work the system will handle whereas the load defines intensities, call frequencies, number of users and their behavior.
5. Yes.
6. To some extent, but I also had some previous knowledge here.
7. Same answer as chaos engineering question.
8. Yes.
9. Yes, the simple UI for configuring made learning much easier than a complex config file.

10. Yes.

"If you could improve one thing about the experience, what would it be?"

1. -
2. Making the UI a little more adaptable to my personal preference or focus in a given workflow step. For example, expanding the Work Configuration section to near-fullscreen when I work with the code in that section.
3. The dashboard shows an overwhelmingly rich set of metrics, which do follow a structure, however, this structure is not reflected visually. It would be great to have some visual distinction between ökänd "ko"metrics.
4. I would go in the direction of parsing YAML files as the only source of truth for devs and visualizing the experimental/workload/chaos information.
5. -
6. There is a lot of information and scrolling, particularly when you (first) open the experiment. Instead of calling elements Probe 1, Failure 1, etc. Give them a speaking name and minimize the area by default. That means follow the principle: overview first, details on demand.
7. The visualization of results. It would be nice to first verify whether or not the experiments worked as intended and second how it impacted the goals.
8. Organize the process as a documented workflow and split the screen into multiple that you edit after each other.
9. More consistency among UI elements, better Firefox support
10. -

"Do you have suggestions for features or improvements?"

1. See above.
2. If this were to become an off the shelf product, the integration of the results in the same product interface could be a neat touch. At least the basics. More inclined experts can always switch to their specialized tooling for in-depth analysis, but a basic results presentation and export functionality provided by the MiSArch Experiment Tool could be nice. Additionally, having a larger variety of integrations with different popular tooling for certain aspects, such as K6 for load testing, could be attractive for users that have already invested in it. In regards to the configurations, perhaps it could be more intuitive if all configurations that belong to the same area of an experiment were in a joint user interface section (e.g., chaos engineering failure configuration in one descriptor file). That brings me also to import and export features for scripts that could be desirable for larger experiments. The brilliant visualization of the experiment could be leveraged even more. For instance, making it interactive in the sense that one can drag and drop (pre-defined) configurations or move pieces around to influence timing settings and alike. ... and the aspects (e.g., a wizard) I mentioned in previous answers could make their way onto my wishlist.
3. Display of currently running and past tests on the start page for easy access to the test results.
4. 1. Experimental visualization, 2. goal visualization, 3. warmup visualization, etc.

5. -
6. 1. Add an option to open previous experiment executions (the Grafana board) again when I revisit the experiment 2. Add suggestions for the available service name (dropdown instead of text field). 3. In general, particularly for use cases in practice, more features for managing the experiments like an overview of experiments, real names, whether they were successful, etc.
7. I believe I addressed this on question number 5.
8. Grafana showing service independent statistics on top was a bit confusing, because changing the service did not change initially visible statistics; maybe just add a "service independent" "hint".
9. -
10. Backstack, Dynamic UI, duration setter, percentages instead of 0.5 etc.