Improving Continuous Testing Infrastructure of Xsuite

Benchaphorn Chanprasertkul[1]
Advisor: Szymon Lopaciuk[2], Supervisor: Giovanni Iadarola[2]
[1] Faculty of Engineering, King Mongkut's University of Technology Thonburi, Bangkok, Thailand
[2] CERN, Geneva, Switzerland
benchanphorn234@gmail.com, szymon.lopaciuk@cern.ch, giovanni.iadarola@cern.ch

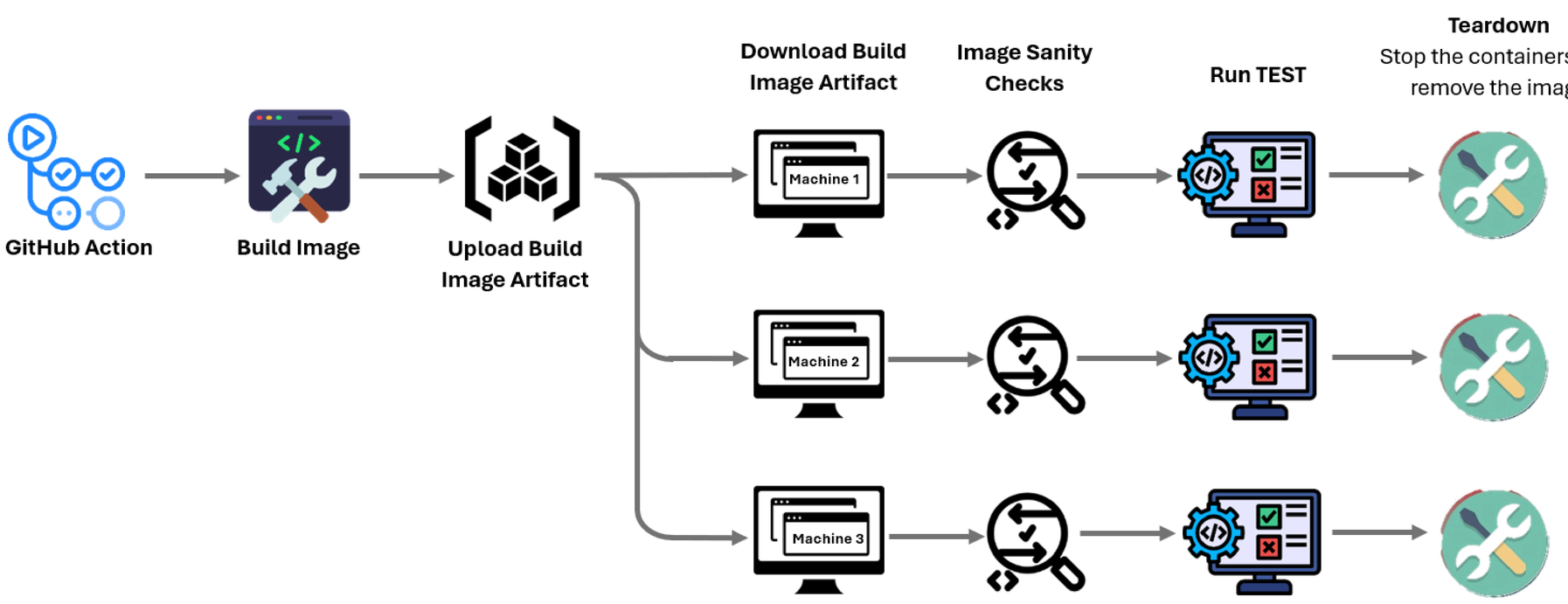
Introduction

Xsuite is a collection Python packages for the simulation of the beam dynamics in particle accelerators. It supports different computing platforms, in particular conventional CPUs and and Graphic Processing Units (GPUs).
To test whether new releases of our tool do not introduce bugs and regressions across all platforms (GPU and CPU), testing infrastructure is required. In this project, I optimized our approach by parallelizing the execution of test without having to rebuild the Docker image in every run. The Docker image we build contains all necessary dependencies, GPU drivers (if applicable), and the Xsuite packages in the versions we need to test. Previously, Docker images were built separately on each test runner machine, a process that was inefficient and resource-intensive. Now, we build Docker images once on a single machine and distribute these artifacts to other machines for testing. This not only enhances efficiency but also allows us to store images as artifacts that can be easily downloaded and reused for debugging purposes. We utilize CERN-hosted VMs with GPUs on OpenStack, emphasizing a scalable and flexible testing environment through rapidly configurable virtual machines. We run tests nightly, and the infrastructure also allows for tests to be triggered on demand, providing flexibility and timely feedback on the system's current state.

Tools

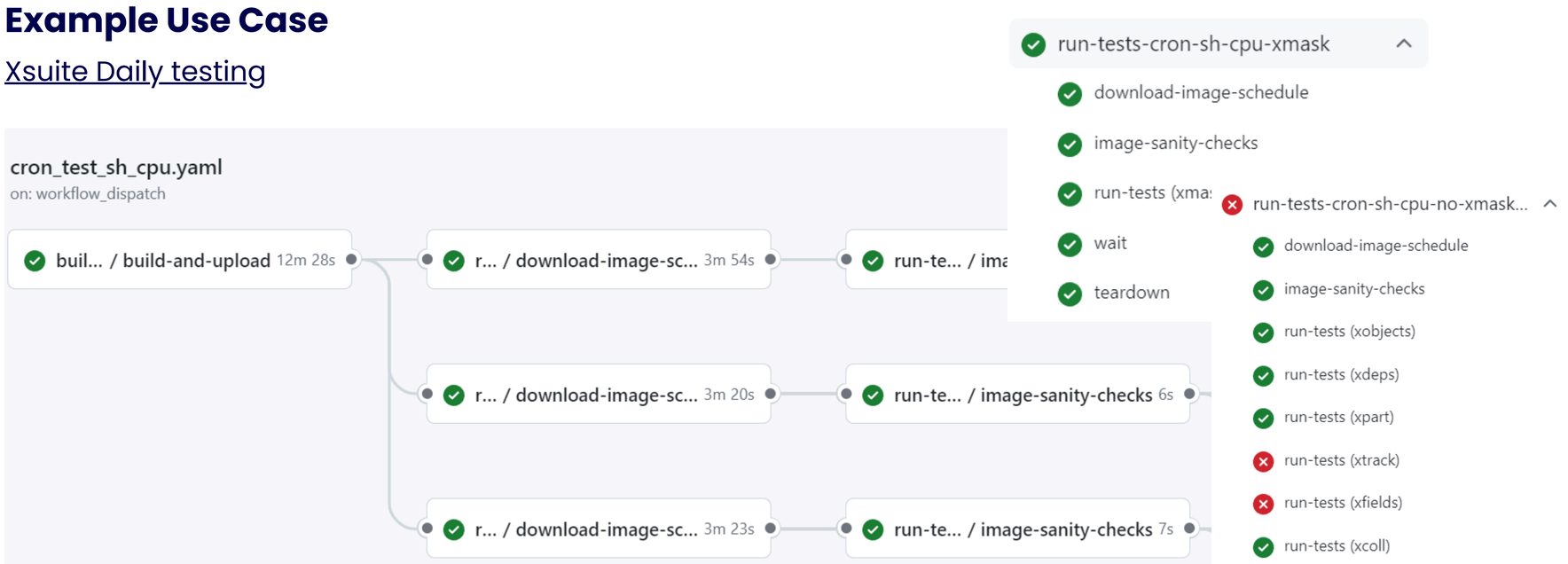
GitHub Action

Pipeline



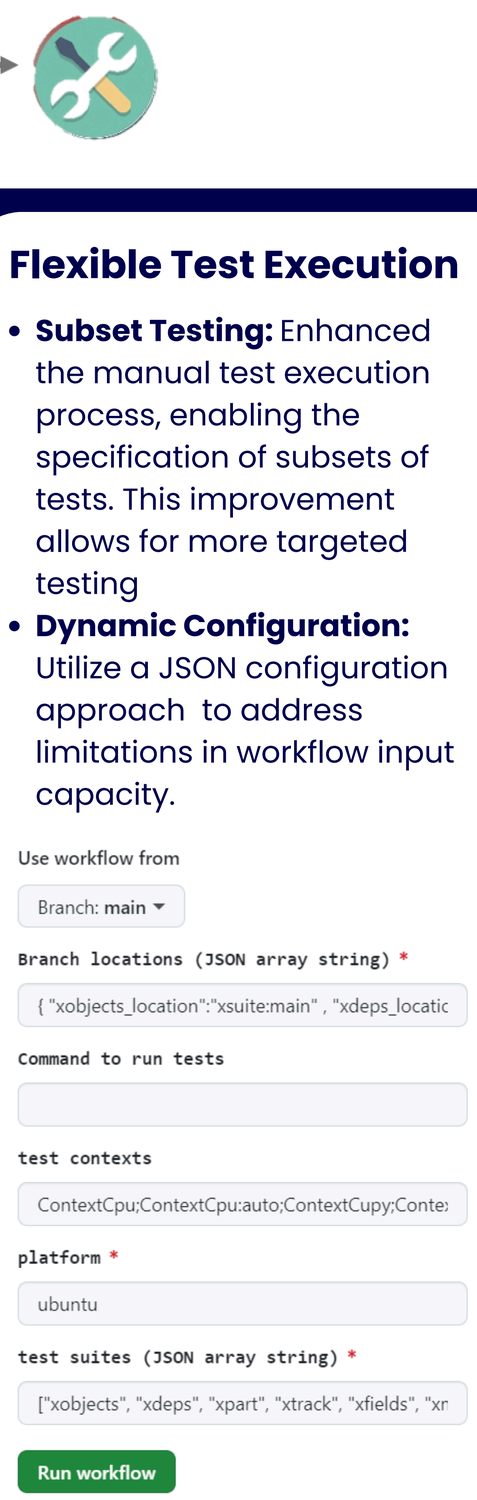
Example Use Case

Xsuite Daily testing



Flexible Test Execution

- **Subset Testing:** Enhanced the manual test execution process, enabling the specification of subsets of tests. This improvement allows for more targeted testing
- **Dynamic Configuration:** Utilize a JSON configuration approach to address limitations in workflow input capacity.



Conclusion

- **Enhanced Testing Efficiency:** Implementing GitHub Actions and Artifacts significantly optimized the testing process by reducing redundancy in building Docker images across multiple machines.
- **Increased Flexibility:** The ability to dynamically add Xsuite packages has made our testing process more adaptable

Limitations

- **Storage Limitations:** GitHub stores build logs and artifacts for 90 days and there is a limit of 500 artifacts

Future Work

- **Using CERN tools to build the image:** integration of CERN's GitLab CI/CD infrastructure into our processes.