Autopilot: Enabling easy Benchmarking of Workload Energy Efficiency (Demo Paper)

Jóakim v. Kistowski
University of Würzburg
joakim.kistowski@uni-wuerzburg.de

Maximilian Deffner
University of Würzburg
maximilian.deffner@stud-mail.uni-wuerzburg.de

Jeremy A. Arnold
IBM Corporation
arnoldje@us.ibm.com

Klaus-Dieter Lange
Hewlett-Packard Enterprise
klaus.lange@hpe.com

John Beckett
Dell Inc.
john_beckett@dell.com

Samuel Kounev
University of Würzburg
samuel.kounev@uni-wuerzburg.de

ABSTRACT

Benchmarking of energy efficiency is important as it helps researchers, customers, and developers to evaluate and compare the energy efficiency of software and hardware solutions. Developing and deploying energy-efficiency benchmarking workloads are challenging tasks, as work must be able to be executed in a power measurement environment using an energy-efficiency measurement methodology. The existing SPEC Chauffeur Worklet Development Kit (WDK) enables the development and use of custom workloads (called worklets) within a standardized power measurement methodology. However, it features no integration in development environments, making building and deployment of workloads challenging. We address this challenge by proposing Autopilot, a plugin for the Eclipse IDE. Autopilot enables fast and easy building and deployment of a workload under development on a system for testing. It also enables benchmark execution directly from the development environment.

Keywords

SPEC, Workloads, Energy Efficiency, Load level, Deployment, Development

1. INTRODUCTION

Server energy-efficiency benchmarking helps researchers, customers, and developers to evaluate and compare the efficiency of hardware and software. Which in turn enables better decision making and leads to greater overall energy efficiency. However, benchmarks must meet a number of quality criteria in order to deliver usable results: They must be relevant, reproducible, fair, verifiable, and usable [3]. Consequently, developing energy-efficiency benchmarks is challenging. Energy-efficiency benchmarks, specifically, must meet these criteria and must also be able to be run in a power measurement environment. This includes a need for discrete power and temperature measurement equipment and stable environmental conditions.

ICPE'17 April 22-26, 2017, L'Aquila, Italy
© 2017 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-4404-3/17/04.
DOI: http://dx.doi.org/10.1145/3030207.3053667

Measurement device requirements and a general benchmarking methodology for energy efficiency measurements are specified in the SPEC Power and Performance Benchmarking Methodology [2]. The SPEC Chauffeur WDK [1] supports this methodology. It is a framework and development kit that manages benchmark configuration, calibration, execution, and data collection. It allows for the development of custom workloads (called worklets) to be executed and benchmarked. A developer using the Chauffeur WDK can focus on the specific benchmarking workload only, leaving other benchmarking considerations to the framework.

The Chauffeur WDK rapidly speeds up the development of energy-efficiency benchmarks. However, worklet building, deployment, and configuration remain a time-consuming process, as multiple executables must be deployed, configured, and started separately. Similarly, log files must be collected from all running instances for analysis of program behavior. To alleviate this, we introduce Autopilot. Autopilot is a plug-in for the Eclipse development environment that automates worklet building, deployment, and configuration. It also offers a graphical user interface that enables execution of benchmarks in development directly within the development environment. It displays measurement results to the developer and provides comparison features that allow for investigation of the impact recent changes in development have had on software energy efficiency.

We envision Autopilot's use in two scenarios: Autopilot's core scenario is its use for development of worklets for energy-efficiency benchmarking. In addition, it may also be used to test the energy efficiency of application software during its development.

2. USAGE SCENARIOS

Autopilot allows for quick and easy energy-efficiency testing of software during development. It features an automated building and deployment system, as well as graphical user interfaces for test execution, result interpretation and comparison directly in the Eclipse IDE. We envision two primary usage scenarios for this functionality:

1. **Development of Benchmarking Workloads**: Autopliot can be used to create and test benchmarking

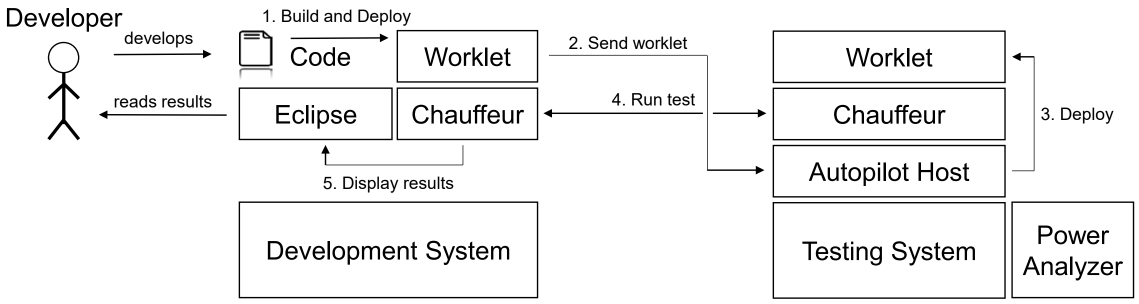

Figure 1: Autopilot workflow

worklets for the Chauffeur WDK. In contrast to traditional development for Chauffeur, Autopilot enables easier building, deployment, and testing. Worklets can be quickly tested and improved during development, ensuring that their power consumption characteristics behave as desired.

2. **Efficiency Testing of Software in Development**: Autopilot may also be used to test software or software components that are not specifically created to serve as energy efficiency benchmarks. As energy efficiency of software becomes a growing concern, its testing during development may help to improve the energy-efficiency of the final software.

3. AUTOPILOT WORKFLOW

Autopilot is intended to be used during the development phase of software creation. At each point in time at which the software under development is in an executable state Autopilot may be used to build, deploy and test the energy efficiency of the software under development. Autopilot also offers features enabling comparison of the performance, power consumption, and energy efficiency of test results. This way, a developer or tester can check the changes a recent change in the software has had on the energy efficiency.

The developer uses a graphical user interface in the Eclipse environment to initiate a test. This user interface is also used to configure the test. It prompts the user for few pieces of information required to execute the test, which includes the IP address of the testing system, test duration, and power analyzer information. Additionally, the user may choose to enter additional information, such as number of test load levels and warmup durations.

After initiating a test, Autopilot performs the following actions (as depicted in Fig. 1):

1. **Build and Deploy**: The project code is built and packaged into a Java .jar file using ANT scripts, as is common for Chauffeur WDK projects. It is also deployed on the developer system, which serves as experiment director for the test.
2. **Send worklet**: The packaged worklet is transferred to the testing system, where it is received by the Autopilot Host.
3. **Deploy**: The Autopilot Host deploys the worklet into the Chauffeur Host that runs on the testing system.
4. **Run test**: Chauffeur components on the development and testing systems are triggered to begin test execution, as configured by the user.
5. **Display results**: Results are forwarded from Chauffeur to Eclipse to be displayed to the user. Autopilot offers comparisons between different test runs, showing absolute and relative changes in power consumption, throughput, and energy efficiency between different versions of the software under test.

We expect tests to be configured to run for a short duration so that they may be run at a high frequency during development. This also means that development tests are usually not configured as full benchmark tests would be (i.e., they run at shorter durations, fewer load levels, shorter warm up phases). To support this scenario, Autopilot features a simple and advanced configuration mode. The simple mode features few configuration options and is intended to be used for quick testing during development. The advanced mode features more options and allows setting of interval durations and execution options that may be necessary for more complex or specialized workloads.

Of course, worklets developed using Autopilot are still normal worklets that can be used in any Chauffeur environment. Specifically, this means that the Jar file built by Autopilot, as well as the generated configuration files, can be moved to any other Chauffeur execution instance and executed from there.

4. CONCLUSIONS

This paper introduces Autopilot, an Eclipse plug-in that aides in development of energy-efficiency benchmarking worklets for the Chauffeur WDK framework. Autopilot automates the process of building and deploying a worklet in development for testing. It provides graphical user interface for test execution from the development environment and for result comparison.

Autopilot's measurement results can be used to improve energy efficiency of software under development. In addition, Autopilot simplifies development and testing of benchmarking loads, making testing and deploying easier than in conventional development.

5. REFERENCES

[1] SPEC Chauffeur WDK. http://spec.org/chauffeur-wdk/.

[2] SPEC Power and Performance Benchmark Methodology. http://spec.org/power/docs/SPEC-Power_and_Performance_Methodology.pdf.

[3] J. von Kistowski, J. A. Arnold, K. Huppler, K.-D. Lange, J. L. Henning, and P. Cao. How to Build a Benchmark. In *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering (ICPE 2015)*, ICPE '15, New York, NY, USA, February 2015. ACM.