December 2021

DELIVERING EARLY, CONTEXTUALIZED GUIDANCE REGARDING CLOUD CARBON FOOTPRINT

Trevor Smith

Carole Reynaud

Thierry Gruszka

Franck Bachet

Fabien Andrieux

DELIVERING EARLY, CONTEXTUALIZED GUIDANCE REGARDING CLOUD CARBON FOOTPRINT

AUTHORS:
Trevor Smith
Carole Reynaud
Thierry Gruszka
Franck Bachet
Fabien Andrieux

ABSTRACT

Across multiple industry segments, organizations that are deploying systems to the cloud are increasingly concerned with the carbon footprint of their activities. However, once a system has progressed through design and development and it is in production, its fundamental energy consumption profile is set. Consequently, a need exists for a way to deliver energy consumption awareness to engineers during the development process so that a lower carbon footprint may be established within a system, at the earliest possible stage. To address that need, techniques are presented herein that support a system for delivering such awareness to engineers during the development process. Aspects of the presented techniques encompass integrating with integrated development environments (IDEs) using industry-standard technology, interfacing with energy monitoring technology, and carrying out a statistical analysis involving correlation and regression between tokenized source code and granular workload-oriented carbon impact assessments.

DETAILED DESCRIPTION

Many organizations across multiple industry segments are deploying systems to the cloud and such organizations are increasingly concerned with the carbon footprint of their activities. Those concerns may be driven by a combination of factors including, public and regulatory pressure, a desire to seek a competitive advantage, the need to reduce costs, and the advantages that arise from an enhanced corporate social responsibility profile in attracting and retaining the best talent.

However, once a system has progressed through design and development and is in production, its fundamental energy consumption profile is set. From that point forward,

energy usage will only increase with adoption as the business grows and the system scales. It is very difficult to address the carbon impact of such systems retrospectively.

It is well-established within the software engineering industry that the largest impacts on system quality may be achieved by focusing on system elements as early as possible in the development lifecycle. The same philosophy needs to be applied to environmental impact. Thus, what is needed is a way to deliver energy consumption awareness to engineers during the development process so that a lower carbon footprint may be established within a system at the earliest possible stage.

Modern integrated development environments (IDEs) often provide some form of context-sensitive content assistance system. Such systems may be decomposed into several components.

Within a first component, a static source code analysis process produces a parsed, tokenized tree-based representation of the source code that is under development. Within a second component, a content assist agent reacts to navigation by the user within the IDE by looking up hovered over, highlighted, or clicked elements in the parsed token tree. With the relevant language construct thus identified, the content assist process may then resolve the construct against a specific content assist source, providing contextual information that is related to the construct. Such information may comprise, for example, documentation, parameter hints, or a source code snippet suggestion.

The approach that was described above has been standardized with the result that IDE plugins may be readily developed. An example of a standard in this space is the Microsoft Language Server Protocol (LSP) which is used in the popular Microsoft Visual Studio Code IDE software to provide pluggable content assistance across a variety of technologies.

Techniques are presented herein that support a system to deliver energy consumption awareness to engineers during the development process so that a lower carbon footprint may be established within a system at the earliest possible stage. Aspects of the techniques presented herein build upon the IDE foundation in order to provide carbon impact information in the form of a new content assist source that can be readily integrated with a modern IDE in the manner that was previously described.

Figure 1, below, depicts a number of interacting processes, separated by color and denoted by labels.

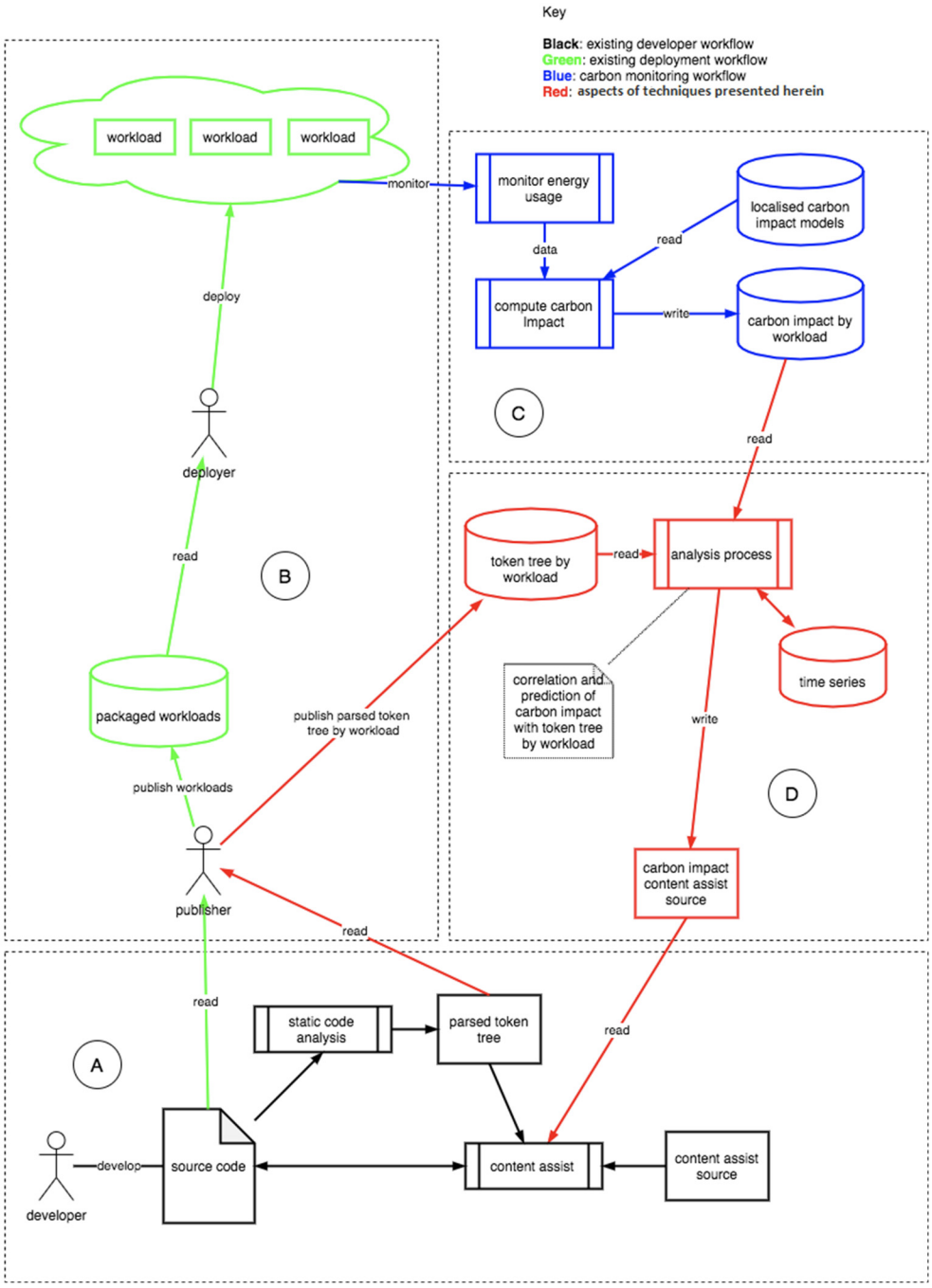

Figure 1: Illustrative Interacting Processes

As depicted in Figure 1, above, a first area (labelled 'A') portrays a developer who is interacting with a typical IDE. The source code that is under development undergoes a

static source code analysis which produces a parsed token tree. The developer is aided by a content assist process, which makes use of a content assist source. The developer can therefore expect to see, in a normal fashion, context-sensitive assistance within the IDE as the source code is being written.

Referring again to Figure 1, above, a second area (labelled 'B') shows a publisher consuming the developed source code, packaging it as a deployable workload, and adding it to a workload package store. In practice, the publisher role is usually automated in the form of a continuous integration (CI) and continuous delivery (CD) system (i.e., a CI/CD system).

Aspects of the techniques presented herein introduce two novel interactions for a CI/CD system. First, in addition to consuming the source code, the publisher also consumes the output of the static source code analysis (i.e., a parsed token tree). Second, the parsed token tree is packaged and added to a parsed token tree store with reference to the published workload package.

From the packaged workload store a deployer role may take workload packages and deploy them to a cloud platform in the usual way. Similar to the publisher role, such a process is often automated.

Returning to Figure 1, above, a third area (labelled 'C') shows an energy monitoring process that is observing the cloud-deployed workload and providing data to a process which estimates the carbon impact based on a model comprising the way in which localized energy use translates into carbon impact. It is important to note that different systems are available to perform such estimations. The output of the carbon impact assessment may then be added to a carbon impact store with reference to the workload. Under aspects of the techniques presented herein, the resolution of such an assessment is either at the level of the workload or at a more detailed level.

Referring once again to Figure 1, above, a fourth area (labelled 'D') portrays elements of a main component of the techniques presented herein. An analysis process accepts inputs from the parsed token tree store (that was populated by the publisher) and the carbon impact store (that was populated by the carbon impact computation process) and performs a statistical analysis consisting of correlating the parsed token tree with

carbon impact data and apportioning any impact to the constructs that are used within the workload.

There are a variety of well-known approaches for accomplishing the statistical analysis that was described above. As a first example, a Spearman's rank-order correlation provides a non-parametric correlation that may be used for non-linear relationships. As a second example, multiple regression may be used for determining the way in which a set of independent variables contribute to given dependent variable.

According to aspects of the techniques presented herein, in order to provide historical depth to the analysis process a time series store is provided. To such a store are added the results of each analysis that is carried out. The data in such a store may be used in future iterations of the analysis process, thus providing improved accuracy over time.

The output of the analysis process is a token tree that is contextualized with predicted carbon impact data which may be published to a carbon impact content assist source.

A carbon impact content assist source may be made available to the content assist process that is used by a developer's IDE through a suitable integration mechanism (such as described above). In this way, the developer is presented with contextualized carbon impact information while navigating the source code within the IDE (e.g., when highlighting, hovering over, or clicking on elements of the source code that is under development).

In summary, techniques have been presented that support a system to deliver energy consumption awareness to engineers during the development process so that a lower carbon footprint may be established within a system at the earliest possible stage. Aspects of the presented techniques encompass integrating with IDEs using industry-standard technology, interfacing with energy monitoring technology, and carrying out a statistical analysis involving correlation and regression between tokenized source code and granular workload-oriented carbon impact assessments.