GREEN CODING – MEHR NACHHALTIGKEIT IN DER IT

In Zukunft wird es weitere Entwicklungen in Richtung Performanceverbesserung und damit auch in Richtung Nachhaltigkeit geben.

Der Klimawandel ist die zentrale Herausforderung des 21. Jahrhunderts. Unternehmen weltweit müssen ihren CO_2-Fußabdruck drastisch reduzieren, wobei sie verstärkt auf die Digitalisierung setzen. Allerdings stößt die Technik dahinter immense Treibhausgase aus, die verringert werden müssen. Um die IT nachhaltiger zu gestalten, wird ein neuer Ansatz verfolgt – Green Coding.

Green Coding bezieht sich auf den Energieverbrauch einer Software und hier insbesondere auf die Bereiche Softwarearchitektur, die konkreten Softwarekomponenten sowie die Betriebsplattform. Im Rahmen der Softwarearchitektur ist Green Coding eine weitere nichtfunktionale Anforderung analog zu Security oder Performance. Sie zielt darauf ab, den Energieverbrauch einer Softwarekomponente möglichst gering zu halten.

In der Praxis deckt sich das sehr oft mit Performanceanforderungen. Das heißt, häufig ist der performanteste Code auch der, der am wenigsten Energie verbraucht. Bei der Realisierung der Softwarekomponenten muss die Anforderung des geringen Energieverbrauchs messbar formuliert und während und nach der Entwicklung überprüft werden. Die Programmierer können bei der Optimierung weitgehend auf bekannte Maßnahmen zur Performancesteigerung zurückgreifen. Das Motto lautet hier: „Jede Codezeile hat das Potenzial, den Energieverbrauch zu senken."

Grüne Betriebsplattform

Die Betriebsplattform ist ein weiterer Bereich, in dem Green Coding eine große Rolle spielt. Der Betrieb in der Cloud ist in der Regel um ein Vielfaches energie- und damit CO_2-effizienter als in firmeneigenen Rechenzentren, weil virtuelle Server oder Container nur zum Bedarfszeitpunkt gestartet werden. Wird ein eigener lokaler Server betrieben, so sollte sich die Leistung der Hardware am tatsächlichen Bedarf ausrichten, der Server also nicht überdimensioniert sein.

Ist der Betrieb in der Cloud nicht möglich, dann sollte die Abwärme des Rechenzentrums in der kalten Jahreszeit bestenfalls auch als Heizquelle verwendet werden.

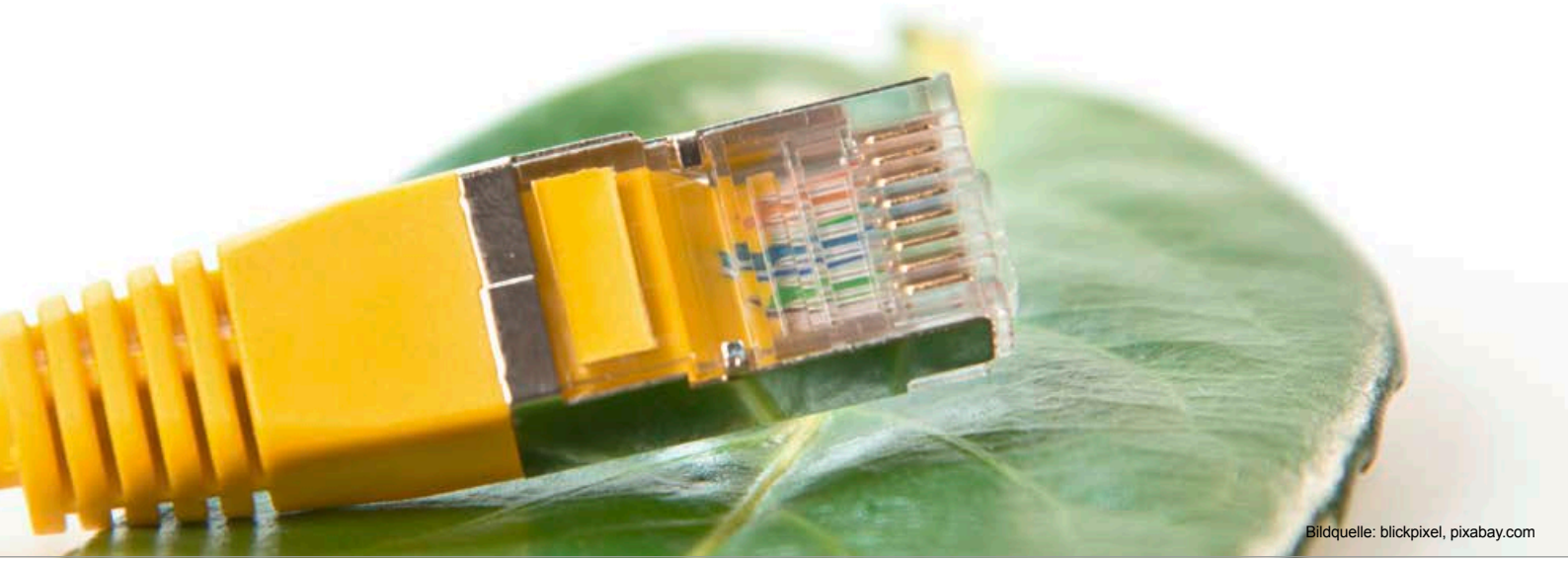
Laut wissenschaftlichem Dienst des Bundestages steigt der Energieverbrauch von Rechenzentren jährlich an. Der Anstieg sei dabei vor allem auf den steigenden Strombedarf der Server zurückzuführen. Obwohl der Anteil der Software im Gegensatz zum Beispiel zur Kühlung der Rechenzentren geringer ist, wird die Nachfrage nach energieeffizienter und ressourcenschonender Software in Zukunft steigen.

Auch die Anwender der Software haben Einfluss darauf, wann bestimmte Berechnungen durchgeführt werden. So können zum Beispiel unkritische Batchläufe um die Mittagszeit ausgeführt werden, wenn eine Nutzung von Solarenergie möglich ist.

```
import com.msg.powermeasurement.PowerMeasurementExtension;
import com.msg.powermeasurement.dto.SensorValue;
import com.msg.powermeasurement.dto.SensorValues;

@ExtendWith({PowerMeasurementExtension.class})
public class EnergyConsumptionTest {
    @SensorValues
    List<SensorValue> sensorValueList;

    @RepeatedTest(10)
    void calculateYtdPerformance() {
        BigDecimal sum = BigDecimal.ZERO;
        for (int i = 0; i < 1_000_000; i++) {
```


Bildquelle: blickpixel, pixabay.com

CHECKLISTE

Die wichtigsten Punkte für alle Stakeholder

- Nichtfunktionale Anforderungen zu Grüner IT und Green Coding sind definiert
- IT-Architekten und Entwickler sind geschult
- System skaliert dynamisch und fährt sich bei Leerlauf herunter
- Dynamische Inhalte und Echtzeitverarbeitung werden möglichst vermieden
- Komponenten mit dem höchsten Energieverbrauch sind identifiziert
- Green Coding Prinzipien und Entwicklungsmuster werden in folgenden Punkten berücksichtigt:
  - Effizienter Code mit optimalen Algorithmen
  - Vermeidung unnötiger Round-Trips
  - Caching
  - Optimiertes Datenvolumen
  - Komprimierte Netzwerkkommunikation
  - Datenbankindizes
  - Optimierte Ressourcennutzung durch optimierte Konfiguration
  - Zero Waste Code
  - Prinzip: „Jede Codezeile hat das Potenzial, den Energieverbrauch zu senken“
- Performance Engineering: Es ist klar, was, wann und wie gemessen wird
  - Was: Gesamtsystem, Prozess, Komponente, Methode
  - Wann: Unittest, Integrationstest, Betrieb
  - Wie: Open Hardware Monitor oder Smart Plugs, Frameworks
- Ressourcenverbrauch: Optimierung von CPU, RAM, Speicher, Netzwerk
- DevOps und Betrieb
  - DevOps Pipeline ist hinsichtlich Energieverbrauch optimiert
  - Containerbasierte Plattform mit intelligenter, dynamische Skalierung wird genutzt

Mögliche Entwicklungen

In Zukunft wird es weitere Entwicklungen in Richtung Performanceverbesserung und damit auch in Richtung Nachhaltigkeit geben. Die Vergangenheit zeigt, dass wir immer schneller immer komplexere Systeme entwickeln können und die Programme oft sogar noch performanter und mit weniger Energieverbrauch laufen als früher.

Beschäftigte in Fachabteilungen, IT-Architektur, Entwicklung und Betrieb können somit einen Beitrag zum Umweltschutz leisten. Welche konkreten Maßnahmen und Entwicklungsmuster für grünes Coding umgesetzt werden können, fasst die Checkliste zusammen.

Den Energieverbrauch von Software richtig messen

Um herauszufinden, an welchen Stellen im Sourcecode Energie eingespart werden kann, benötigt man Werkzeuge zum Messen des Energieverbrauchs. Tools, die auf Prozessebene den Energieverbrauch von Software messen, sind bereits zum Beispiel in Windows integriert.

Autor: Jakob Deiner
Jakob Deiner ist M. Sc. Informatik und verfügt über mehr als zehn Jahre Berufserfahrung. Als Lead IT Consultant bei msg GillardonBSM berät er Kunden in seinen Schwerpunktthemen IT-Architektur, objektorientiertes Design und containerbasierte Betriebsplattformen. Er ist außerdem Certified Professional for Software Architecture (iSAQB) – Foundation Level (CPSA-F) und Certified Scrum Master (CSM), Scrum Alliance.

Autor: Hans-Peter Keilhofer
Hans-Peter Keilhofer ist Dipl. Informatiker mit langjähriger Berufserfahrung und unterstützt als Executive IT Consultant Kunden in der Planung und Umsetzung von IT-Architekturen. Darüber hinaus hält er als Dozent an der Universität Passau regelmäßig die Vorlesung „Softwareentwicklung für Fortgeschrittene“ am Lehrstuhl für Wirtschaftsinformatik mit Schwerpunkt Internet- und Telekommunikationswirtschaft.

In einen Prozess „hineinsehen“ kann man mit diesen Tools nicht. Das bedeutet, dass keinerlei Aussage darüber getroffen werden kann, welche Komponente oder Funktion innerhalb des Prozesses am meisten Energie verbraucht. Exakt diese Information ist für Entwickler jedoch essenziell.

Wie soll gezielt hinsichtlich Energieeffizienz optimiert werden, wenn der genaue Ansatzpunkt unbekannt ist?

Der jPowerMonitor von msg

Das von msg entwickelte Tool jPowerMonitor überwacht innerhalb laufender Applikationen, welche Funktionen gerade aktiv sind und wie viel CPU-Zeit sie in Anspruch nehmen. Die zur Ermittlung des Energieverbrauchs zusätzlich notwendigen Messdaten, wie zum Beispiel den aktuellen Gesamtverbrauch der CPU in Watt, kann jPowerMonitor nicht selbst abgreifen.

Zu diesem Zweck ist eine Schnittstelle implementiert, welche Stand heute die beiden Windows-Tools Open Hardware Monitor sowie HWiNFO anbindet. Mittels CSV-Schnittstelle lassen sich durch Konfigurationsanpassungen auch Tools für Linux oder MacOS anbinden.

jPowerMonitor bietet darüber hinaus die Möglichkeit, auf die eigenen Funktionen in Java Programmen zu filtern. Manchmal kann es aber auch interessant sein, wie viel Energie die benutzten Frameworks benötigen. Dies ist gerade bei der Erstellung von Prototypen hilfreich, um die energieeffizienteste Bibliothek auszuwählen.

Außerdem ist eine Erweiterung des Testframeworks JUnit in jPowerMonitor integriert, mit der der Stromverbrauch in Komponenten- und Unittests gemessen werden kann. Damit lassen sich beispielsweise zwei Algorithmen, die das gleiche Problem adressieren, hinsichtlich ihres Energieverbrauchs vergleichen.

jPowerMonitor hilft Entwicklern, herauszufinden, an welchen Stellen des eigenen Programms der meiste Strom verbraucht wird. Dies gibt Entwicklern die Möglichkeit, den Code im Sinne von Green Coding zu optimieren und zur Reduzierung von CO_2-Emissionen beizutragen.

Aktuell ist das Tool exklusiv für Java verfügbar. Das Konzept von jPowerMonitor lässt sich jedoch auf weitere Programmiersprachen übertragen, die Introspektion beziehungsweise Reflection unterstützen, wenngleich die Library für die jeweilige Sprache neu zu implementieren ist.

	A	B	C	D	E	F	G
1	Zeitstempel	Thread	Funktion	Energie	Einheit	CO2 Verbrauch	Einheit
2	2022/09/16T09:28:20-830	main	group.msg.jpowermonitor.util.StressCpuExample.iAm100PercentParallel	2628.47832	J	0.35411	gCO2
3	2022/09/16T09:26:22-612	main	group.msg.jpowermonitor.util.StressCpuExample.iAm100Percent	1092.38193	J	0.14717	gCO2
4	2022/09/16T09:26:51-986	main	group.msg.jpowermonitor.util.StressCpuExample.iNeed50Percent	530.3873	J	0.07145	gCO2
5	2022/09/16T09:27:10-788	main	group.msg.jpowermonitor.util.StressCpuExample.iNeed30Percent	323.40486	J	0.04357	gCO2
6	2022/09/16T09:27:15-995	main	group.msg.jpowermonitor.util.StressCpuExample.iNeed10Percent	107.91608	J	0.01454	gCO2
7	2022/09/16T09:27:19-136	main	group.msg.jpowermonitor.util.StressCpuExample.iNeed5Percent	56.38952	J	0.0076	gCO2
8	2022/09/16T09:27:22-391	main	group.msg.jpowermonitor.util.StressCpuExample.iNeed2PercentToo	22.39065	J	0.00302	gCO2
9	2022/09/16T09:27:20-178	main	group.msg.jpowermonitor.util.StressCpuExample.iNeed2Percent	22.03581	J	0.00297	gCO2
10	2022/09/16T09:27:22-391	main	group.msg.jpowermonitor.util.StressCpuExample.iNeed1Percent	19.64748	J	0.00265	gCO2
11	2022/09/16T09:27:51-852	ResultsWriter-Thread	group.msg.jpowermonitor.agent.ResultsWriter.convertKiloWattHoursToCarbonDioxideGrar	0.00005	J	0	gCO2

jPowerMonitor Beispielausgabe für eine Benchmarkmessung

Bildquelle: msg GillardonBSM AG