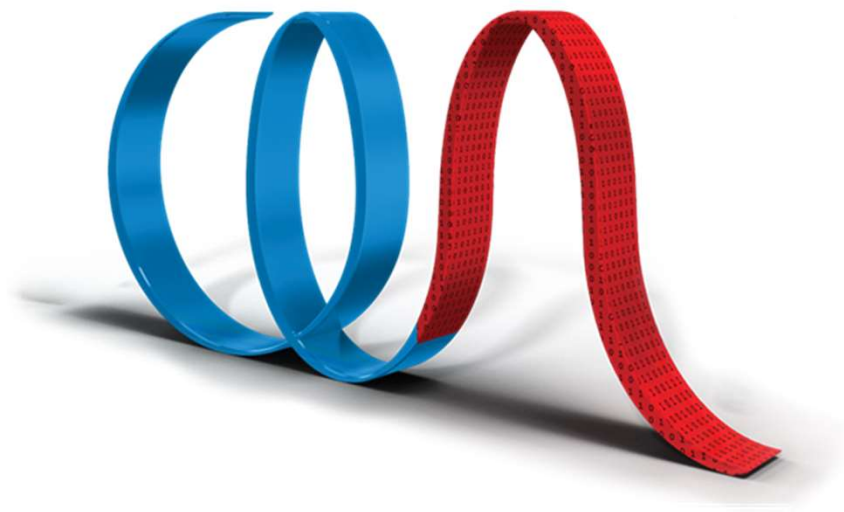


# Green Software – Architektur- und Source Code-Analysen im Hinblick auf Nachhaltigkeit



We lead from the emerging future!

Powered by  
**sieber & partners**    
digital business excellence **CAST**  
Software Intelligence

CNO - Panel

Bern, 30. Oktober 2023

# Agenda

- 14.00 Begrüssung / Vorstellung
- 14.15 Aktuelle Marktsituation Green Software
- 14.30 Gruppen - Austausch Teil 1
- 15.00 Pause
- 15.15 Mögliche Ansätze zur Analyse bestehender Softwarelösungen
  - Gesamtheitlicher Ansatz zum Vorgehen
  - Ansatz mit CAST zur Analyse der Software
  - Outcomes der Green Impact Analyse
- 15:45 Gruppen - Austausch Teil 2
- 16:15 Ausblick

## Begrüßung/Vorstellung

# Software Consulting sieber&partners

Wir analysieren, monitoren und verbessern die Code Qualität und technischen Schulden, SW- und IT-Architekturen, Security Risiken, Open Source Zustand, Cloud Readiness von Software und Softwareportfolios entlang des gesamten Lebenszyklus.



## Sicherheit

Wir prüfen Ihre Software auf Sicherheitslücken und um Security Risiken zu minimieren und begleiten Sie bei der Umsetzung der Massnahmen.



## Qualität

Wir zeigen Risiken auf, die aus der Qualität des Sourcecodes, problematischer Architektur oder ineffizienten Entwicklungsprozessen kommen.



## Innovation

Mit klaren und innovativen Handlungsempfehlungen helfen wir, neue Technologien einzusetzen und eine moderne Architektur zu gestalten.



## Performance

Wir analysieren die Team Produktivität entlang ihres Entwicklungsprozesses oder ihres Softwarelieferanten.



## Transparenz

Wir verschaffen Ihnen Einblicke in Ihre Software, in Ihren Entwicklungsprozess oder in die Leistungen Ihrer Lieferanten.



Dominik Strub-Tiedt



Faysal Cherradi



Stefan Stojkovski



Benno Hübscher

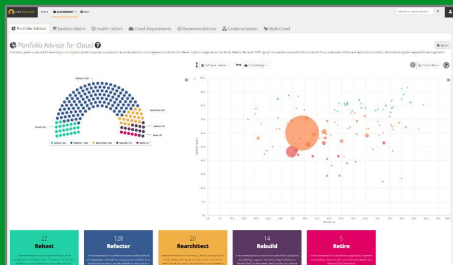
Unsere Technologiepartnerschaften:



# CAST Highlight: rapid insights across your entire portfolio

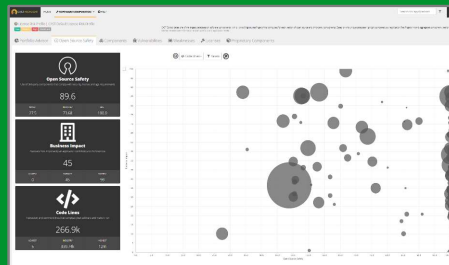
- Automated portfolio views, instant drilldowns, recommendations. Operational in a week.

## Cloud Readiness



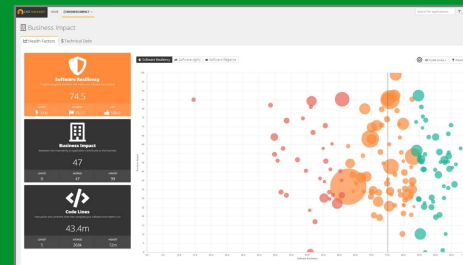
roadmaps, blockers, effort

## Software Composition



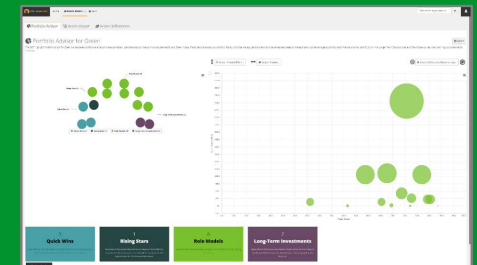
IP & security risks, SBOM

## Software Health



resiliency, agility, tech debt

## Green Impact



deficiencies, actions, trends



Objective & qualitative data



Non-intrusive



Fast & easy to use

## Kurze Vorstellungsrunde der Teilnehmenden

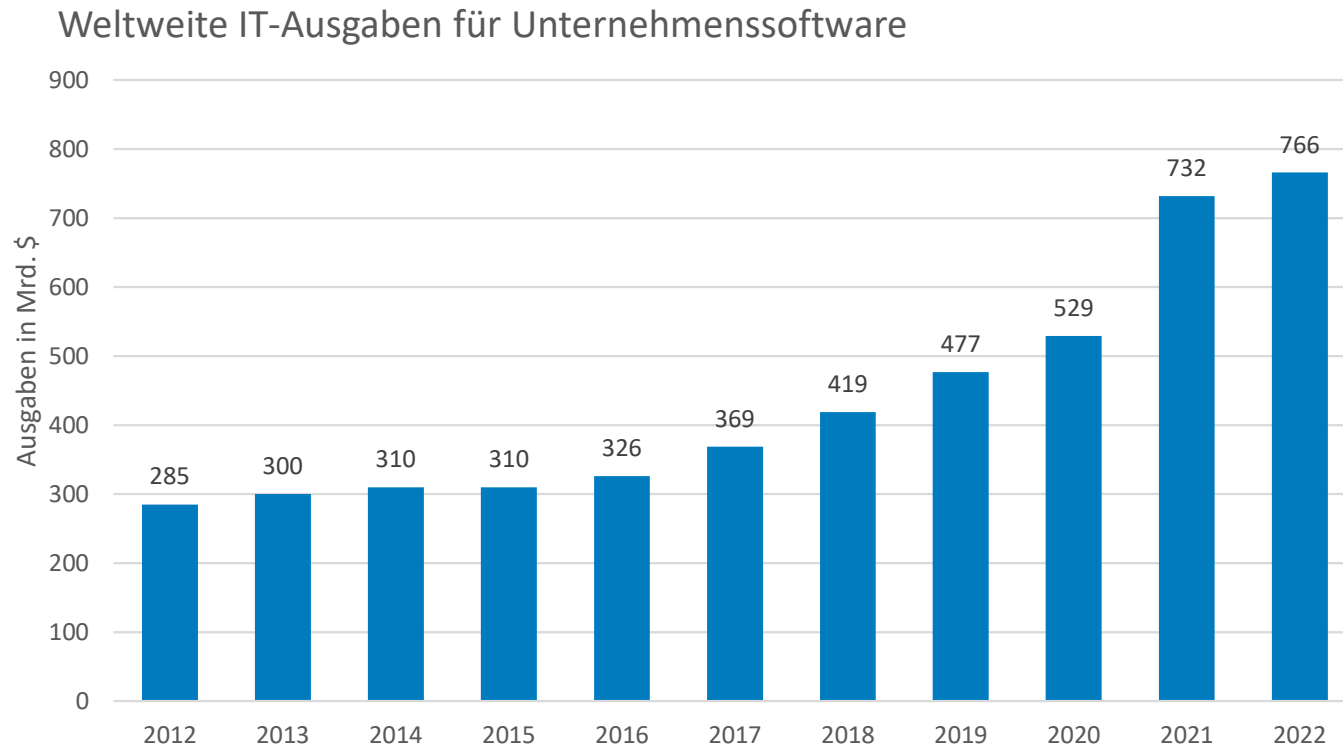
- Name...
- Firma/Funktion...
- Kenntnisse zu Green Software...



# Aktuelle Marktsituation Green Software

---

# Immer mehr IT-Ausgaben fließen in die Software



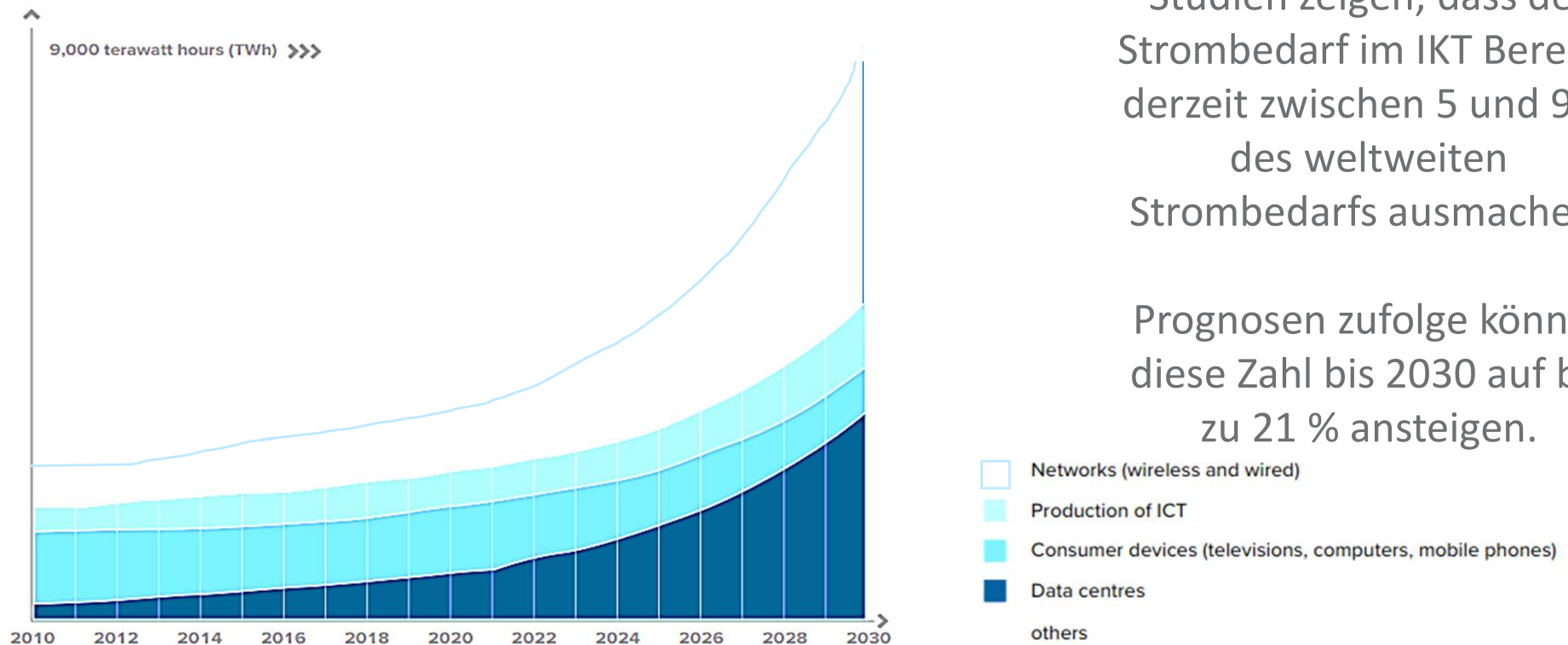
Massive Steigerung der Ausgaben für Unternehmenssoftware mit einer durchschnittlichen jährlichen Wachstumsrate von > 25% von 2012 bis 2022.

Source: [IT spending worldwide by segment 2012-2024](#) | Statista (ohne Forecast 2023 und 2024)



# Servers und Netzwerk sind die grössten Stromfresser

Prognose des Elektrizitätsbedarf von IKT



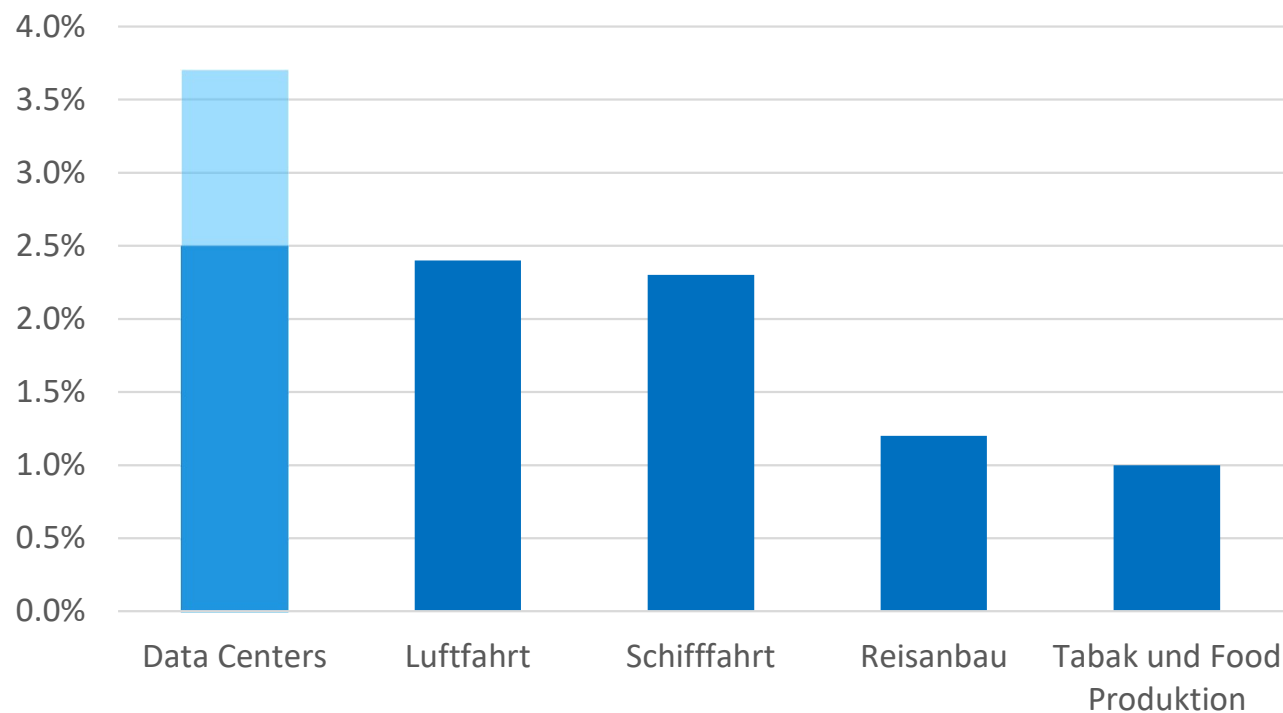
Source: <https://www.gft.com/dam/jcr:03015de2-79bd-4444-b62c-25a9e8cfc0f5/gft-thought-leadership-greencoding-en.pdf>

Studien zeigen, dass der Strombedarf im IKT Bereich derzeit zwischen 5 und 9 % des weltweiten Strombedarfs ausmachen.

Prognosen zufolge könnte diese Zahl bis 2030 auf bis zu 21 % ansteigen.

# Data Centers massiver Treiber des CO2 Ausstosses

Anteil ausgewählter Industrien am weltweiten CO2-Ausstoss



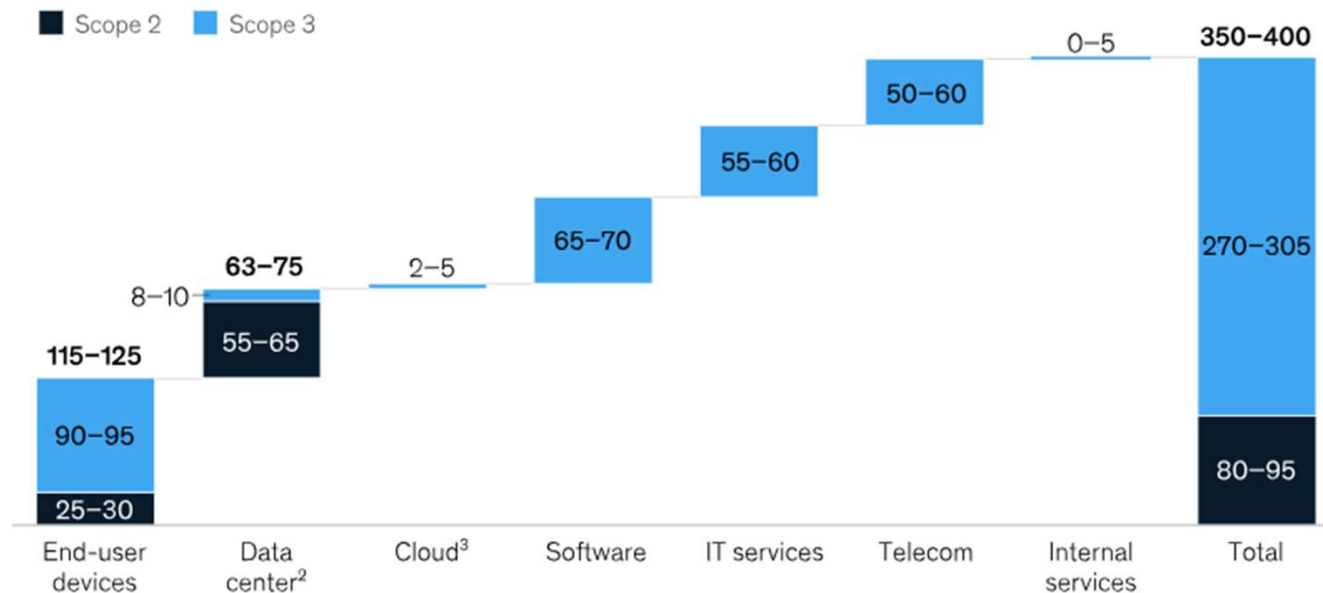
Anteil Data Centers zwischen 2.5% - 3.7%.

Studien zeigen, dass der IKT-Anteil am weltweiten CO2-Fussabdruck von 1.5% 2007 auf heute 4% gestiegen ist und er steuert bis 2040 auf bis 14% zu.

Source: <https://www.climatiq.io/blog/measure-greenhouse-gas-emissions-carbon-data-centres-cloud-computing>

# Software als Nummer 2 der indirekten Emissionsverursacher

Global enterprise technology emissions, Mt CO<sub>2</sub>e<sup>1</sup>



<sup>1</sup>Megatons of carbon dioxide equivalent gases.

<sup>2</sup>Includes emissions from on-premises data center and co-location.

<sup>3</sup>Infrastructure as a service (IaaS) only; software as a service (SaaS) and platform as a service (PaaS) spending accounted for in software category.

Source: [The green IT revolution: A blueprint for CIOs | McKinsey](#)

Gemäss Studie von McKinse erzeugen Endnutzengeräte - Laptops, Tablets, Smartphones und Drucker - weltweit 1,5 - 2,0-mal mehr Kohlenstoff als Rechenzentren.

## Technologie- und Nachhaltigkeitsstrategie gehen nicht Hand in Hand

*Laut dem Accenture-Bericht: "Uniting technology and sustainability,, haben nur 7 Prozent der Unternehmen ihre Technologiestrategien vollständig in ihre Nachhaltigkeitsstrategien integriert. Und nur 49 Prozent der CIOs sind Teil des Führungsteams, das Nachhaltigkeitsziele festlegt.*

Source: [Technology Sustainability Key to ESG Goals | Accenture](#)

# Warum braucht es nachhaltige Software?

- **Relevanz und Menge** von Software in Unternehmen und Organisationen steigt
- Softwareprojekte werden **immer komplexer**
- Es fließen immer **mehr Ressourcen** in Software
- **85% der CO2-Emissionen** durch Software können im Design und Entwicklung beeinflusst werden
- **Globaler Elektrizitätsbedarf** für die ICT wird auf **21% bis 2030** geschätzt

Quellen: Why the World needs Sustainable Software | Accenture ; gft-thought-leadership-greencoding-en.pdf | Whitepaper GfK



Foto: DestroLove - shutterstock.com

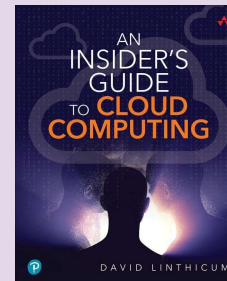
# Green Software als nächster logische Schritt

Die Entwicklung von **Green Software** ist für Unternehmen weltweit der nächste logische Schritt, um ihre Technologie nachhaltiger zu gestalten und neue gesetzliche Anforderungen zu erfüllen.



*"Green Code"  
Initiative  
launched by  
Salesforce in  
May 2023*

*"Salesforce Lightning apps are coded to run natively in browsers, resulting in up to 60% improvement in performance and a reduction in energy use." – [click here](#)*



*David Linthicum,  
Chief Cloud  
Strategy Officer  
at Deloitte  
Published in  
March 2023*

*"There are huge sustainability gains that can be made here (i.e., Green software development) that will also lower your cloud computing bill." – David Linthicum*

# Qualitätskriterien nach ISO 25010



Höhere Software Qualität dank...

- Beschleunigter Time-to-Market
- Einhaltung von Budgets und den Kosten
- Verringerung des Wartungsaufwandes
- Mehr Schutz dank Security
- Effiziente Erweiterbarkeit der Funktionalitäten
- ...

# Erweiterte Qualitätskriterien nach ISO 25010



Höhere Software Qualität dank...

- Beschleunigter Time-to-Market
- Einhaltung von Budgets und den Kosten
- Verringerung des Wartungsaufwandes
- Mehr Schutz dank Security
- Effiziente Erweiterbarkeit der Funktionalitäten
- **Steigerung der ökologischen Nachhaltigkeit**



## Gruppen Austausch Teil 1

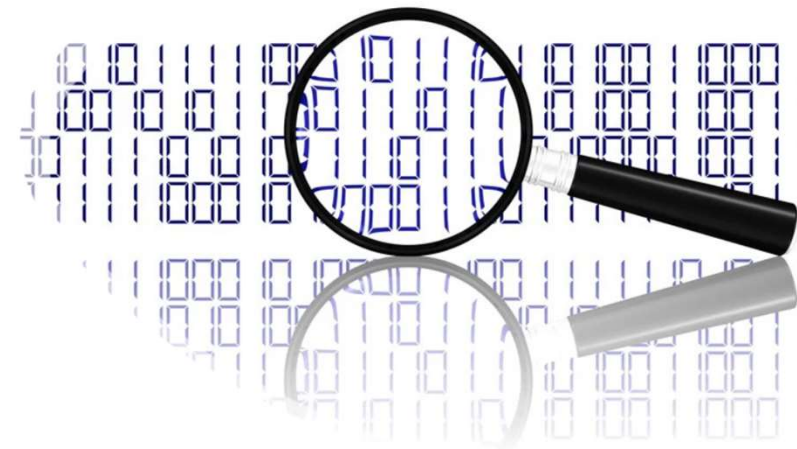
---

# Gruppen Austausch Teil 1

Was erwarten Sie von Green Software?

Aus der Sicht der...

- Auftraggeber / Fachbereich
- Betreiber / Infrastruktur
- Entwicklungsunternehmen / -abteilung
- Endnutzer / User



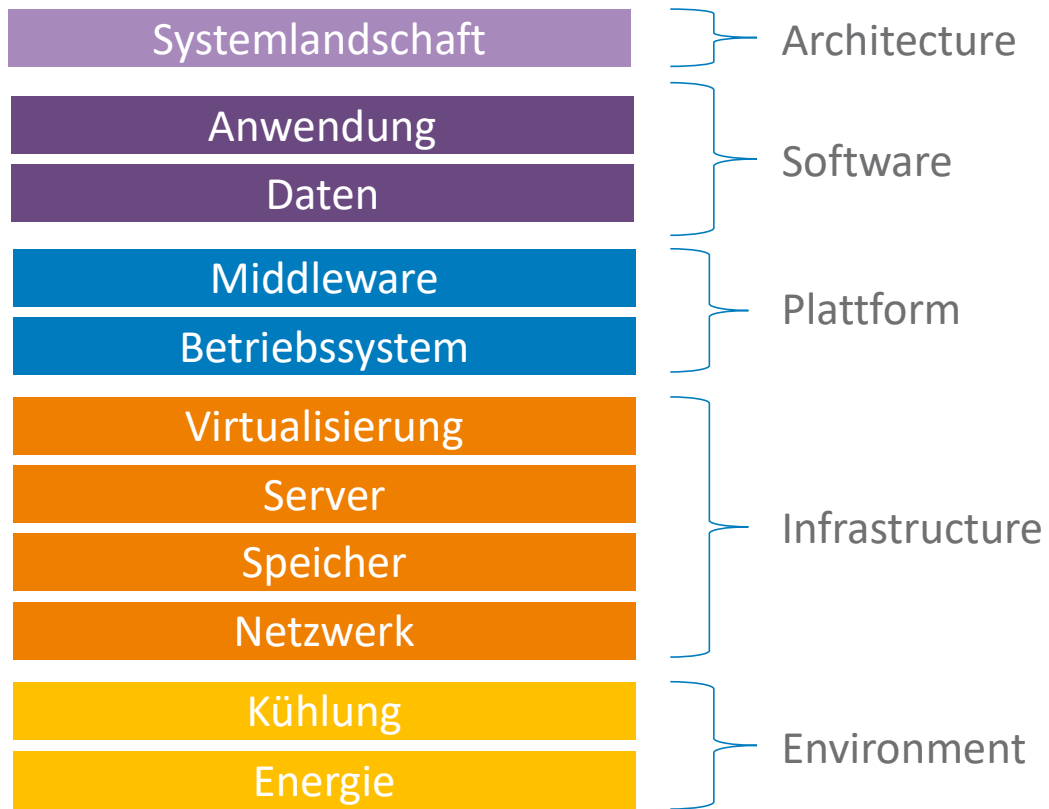
# Kaffeepause



## sieber&partners Ansatz Green Software

---

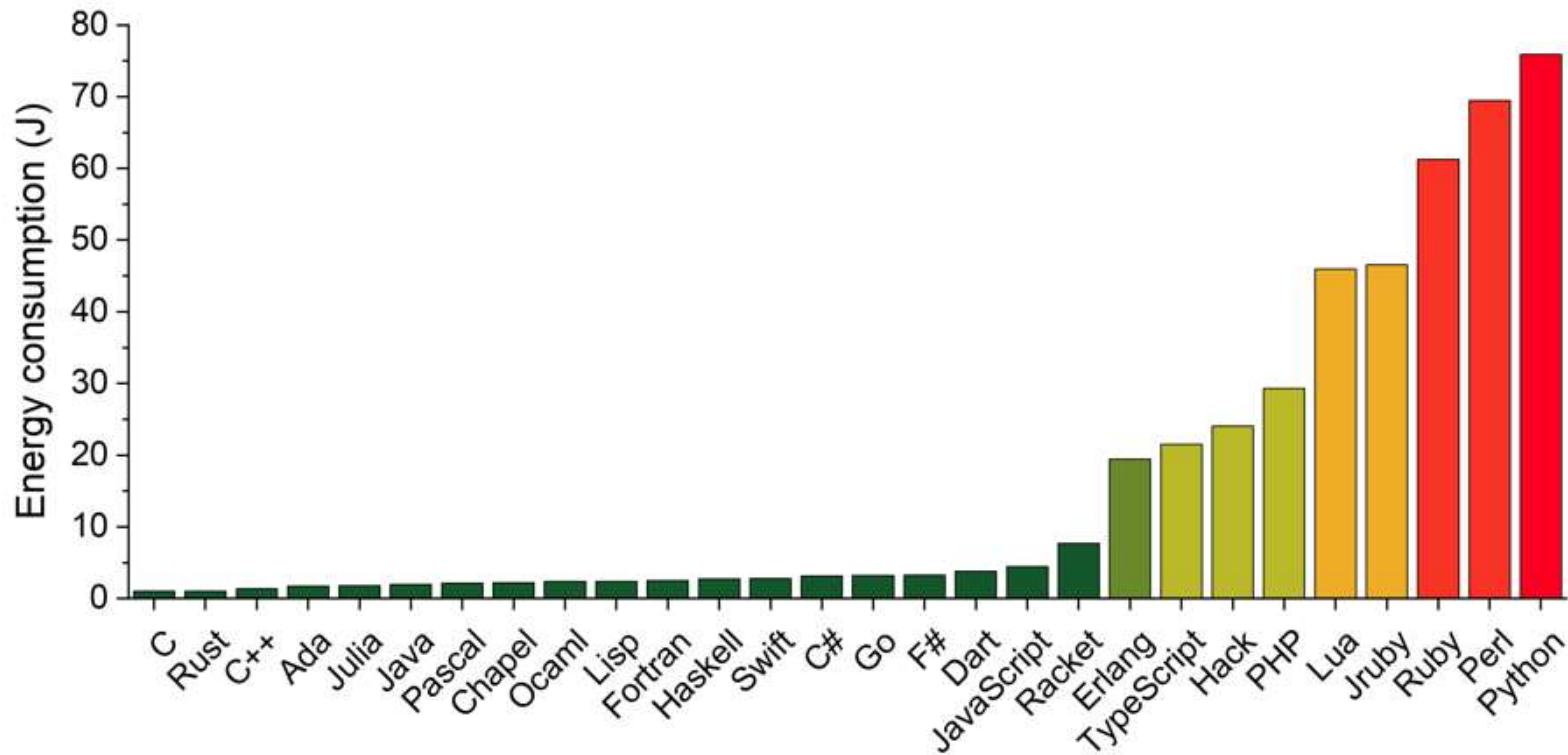
## Komponenten um Software zu betreiben



## sieber&partners Green-Model

Aspekte	Erklärung
<b>Algorithm</b>	Einsatz energieeffiziente-Algorithmen wo sinnvoll und möglich
<b>Technology</b>	Wahl der richtigen Technologien mit geringerem Energieverbrauch
<b>Library</b>	Nutzung ausschliesslich energiefreundlicher Bibliotheken
<b>Component Entaglement</b>	Mehr Kopplung bedeutet mehr Datenaustausch und mehr Energieverbrauch
<b>Network Communication</b>	Nutzung von Kommunikationsprotokollen, die energieeffizienter sind
<b>Data</b>	Nutzung von energieeffizientem Umgang mit Daten
<b>Development Process</b>	Einsparung von Ressourcen und effizientes Testen
<b>Environment</b>	Aktives Scheduling der notwendigen Laufzeitumgebungen Bewusste Wahl von energieeffizienten Datacenters
<b>Hardware life cycle</b>	Update-Politik der Hersteller hat sie nicht nur Einfluss auf den Energieverbrauch, sondern auch auf den Zeitpunkt einer Neubeschaffung.

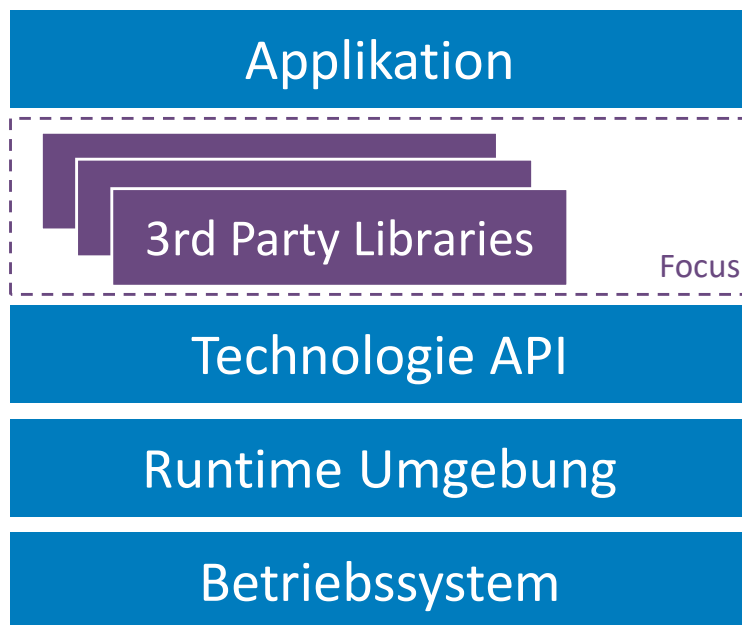
## Maschinen-nahe Sprachen verbrauchen weniger Energie



Source: <https://www.sciencedirect.com/science/article/abs/pii/S0167642321000022>

## Verwendung von energieeffizienten Bibliotheken von Drittanbietern

- Source Code Analyse der Third Party Libraries, um die Libraries zu nutzen, die weniger Ressourcen konsumieren

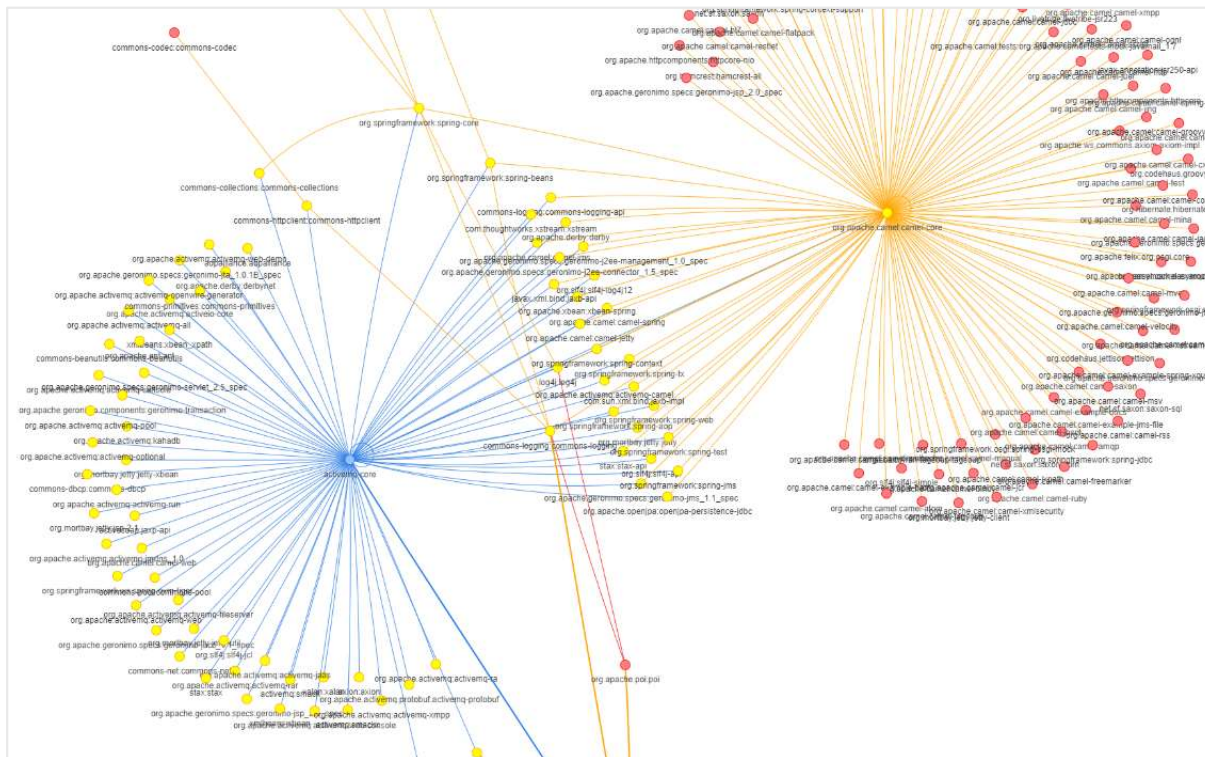


Source: Aufbau einer Applikation





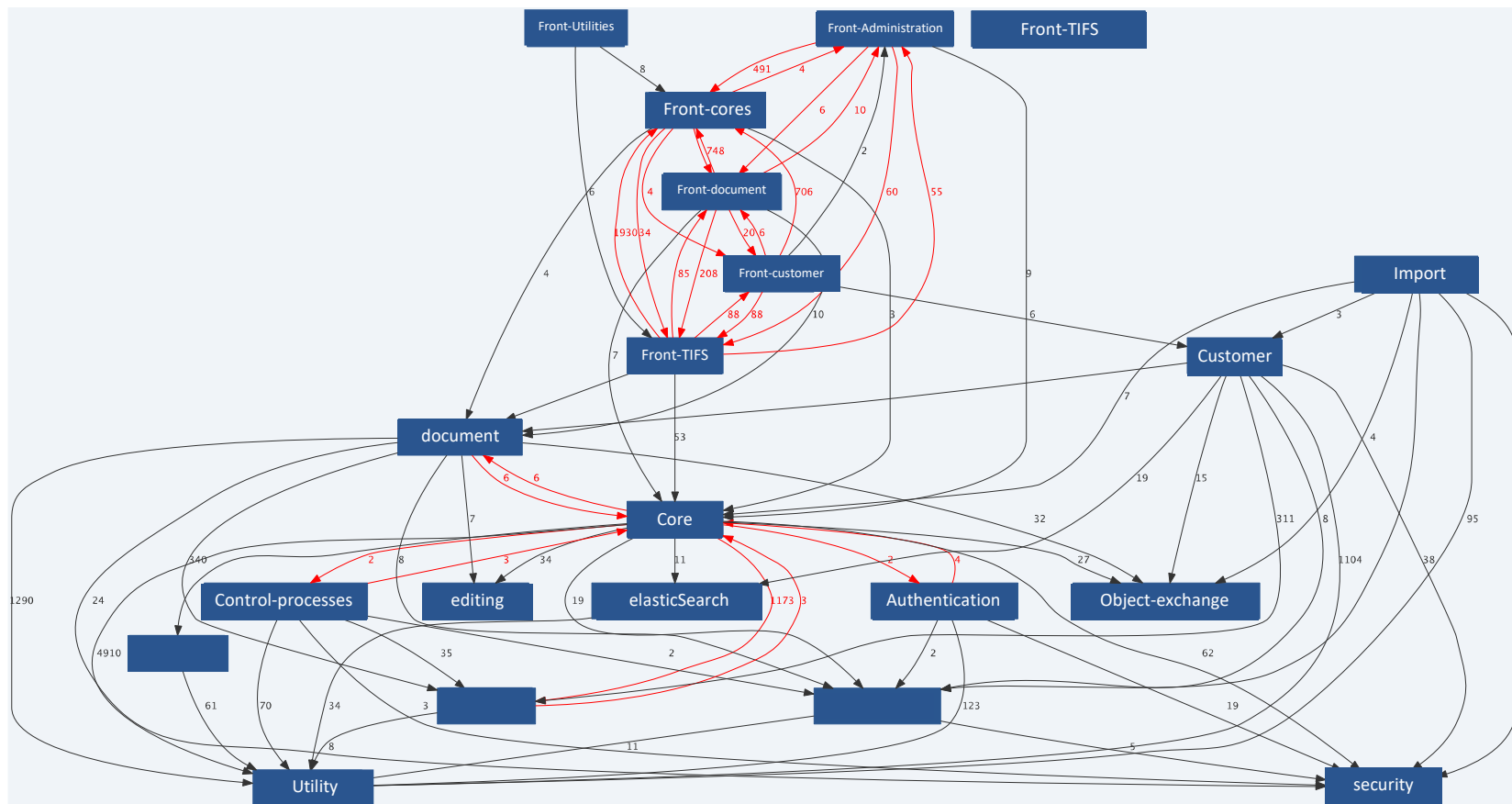
# Verringerung der Abhängigkeiten führt zu Energieeinsparungen



Source: Beispiel aus CAST Software

- Je mehr Abhängigkeiten es gibt, desto mehr Datenaustausch gibt es über die Kommunikationsnetzwerke

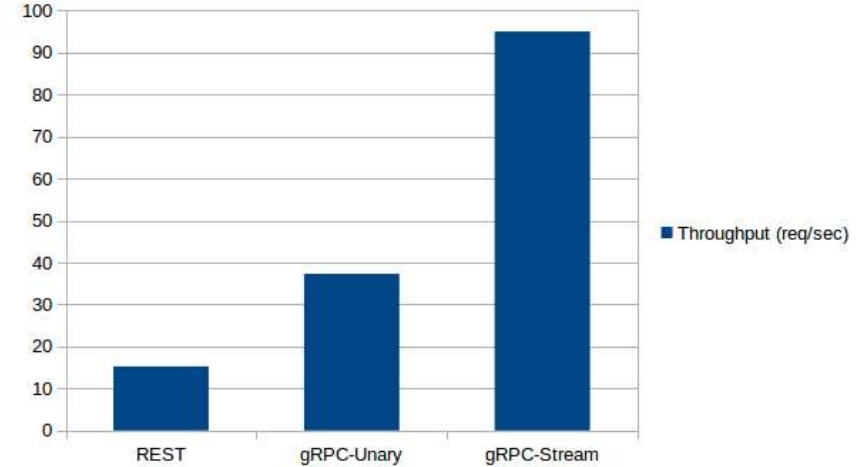
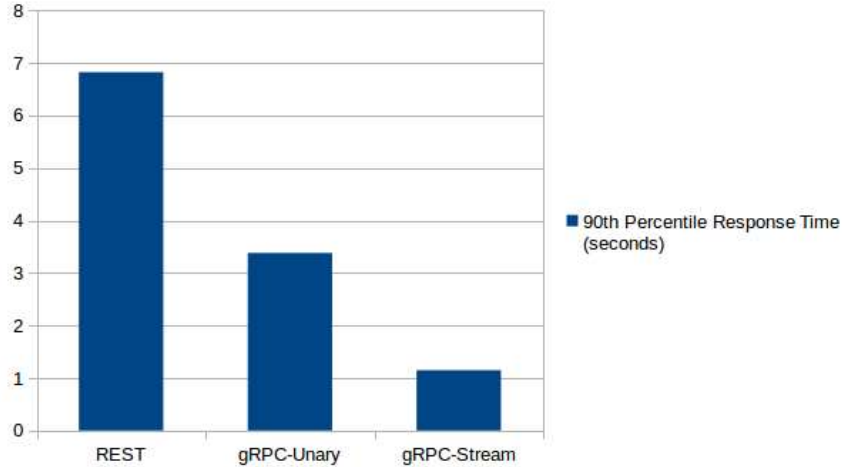
# Starke Kopplung führt zu mehr Datenaustausch zwischen den Services



Dependency Graph

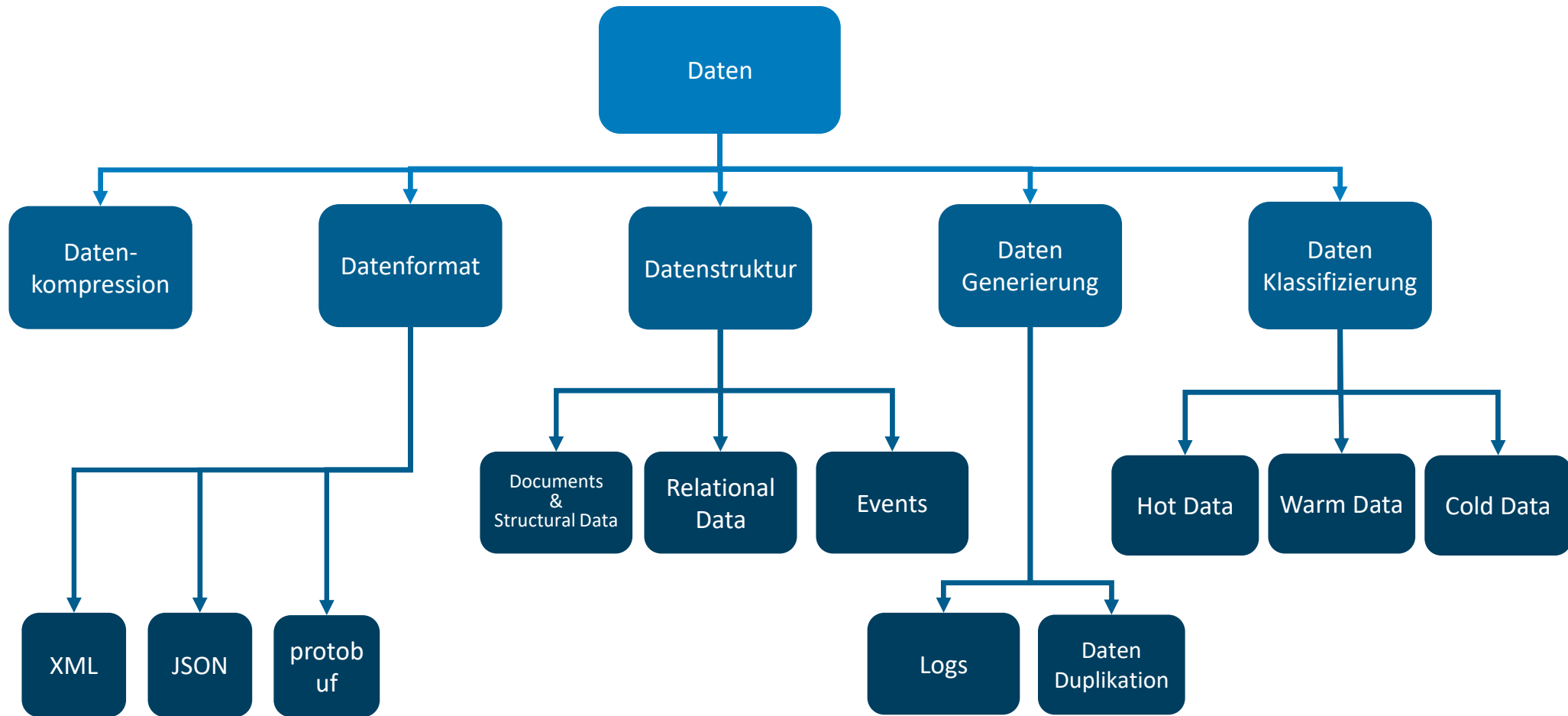
## Abhängigkeiten und ihre Auswirkungen auf die Netzwerkressourcen

- Die entsprechende Kommunikationsprotokolle helfen Energie zu sparen, z.B:



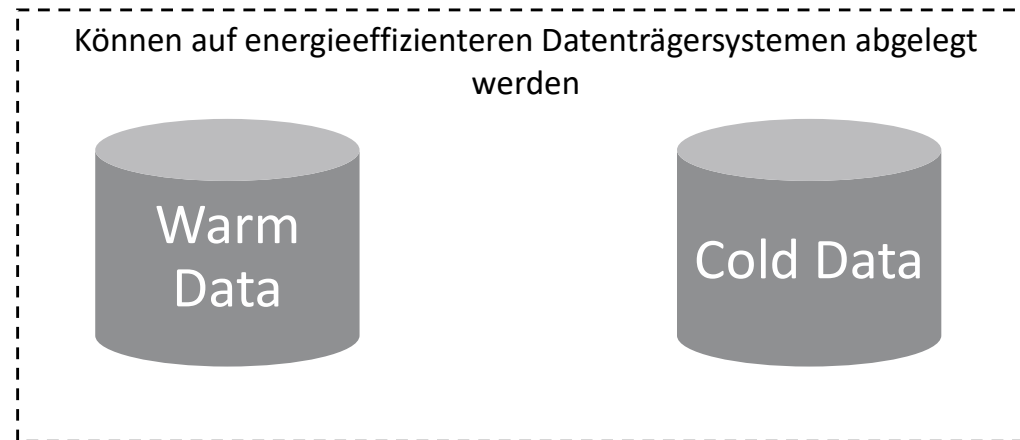
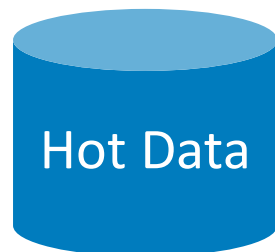
Source: <https://www.vinsguru.com/grpc-vs-rest-performance-comparison/>

# Ein angebrachter Umgang mit den Daten reduziert den Energieverbrauch



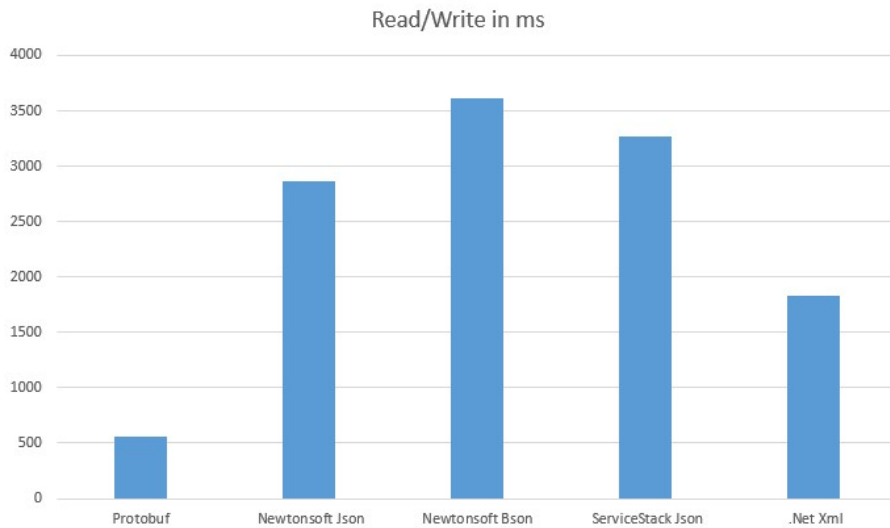
## Archivierungsprozess

- Nicht alle Daten müssen in den hochverfügbaren und performanten Datenablagen gespeichert werden

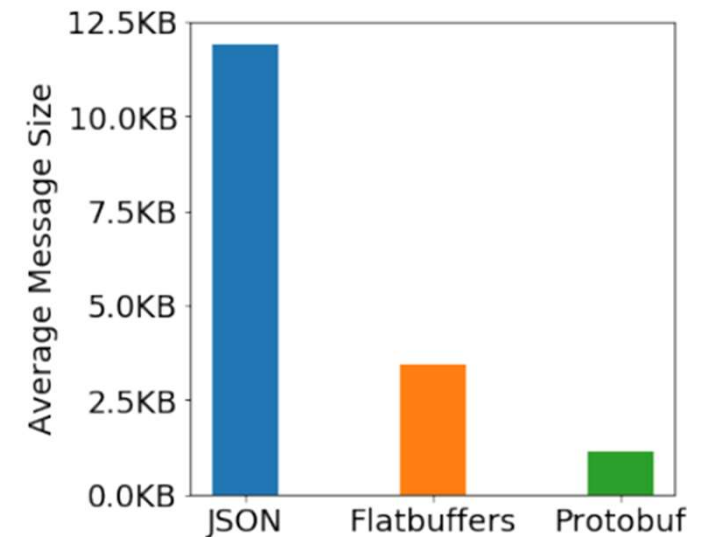


# Geeignete Datenformate können Daten schneller und effizienter zu übertragen

## Zeitverbrauch zur Serialisierung der Nachrichten



## Größen der serialisierten Nachrichten



Source: <https://damienbod.com/2014/01/09/comparing-protobuf-json-bson-xml-with-net-for-file-streams/>

averaged over all messages and 1000 runs/message.  
Source: <https://www.ida.liu.se/~nikca89/papers/networking20c.pdf>

## Energieeffizienter Entwicklungsprozess

- Testen
  - Der richtige Einsatz der Tests spart Ressourcen durch Vermeidung der Ausführung von unnötigen Transaktionen
  - Vermeidung von Doppeltest von Funktionalitäten
- Open Source Libraries
  - Im Entscheidungsprozess soll man sich für energieeffiziente Bibliotheken entscheiden
- Umgebung
  - Nur Umgebungen hochfahren, die man im Betrieb und in der Entwicklung braucht

## Architekturanalyse: Empfehlung für die Nachhaltigkeit

### Verbesserung der Architekturprinzipien:

- Infrastruktur auf Abruf (on-demand)
- Die Umgebungen müssen richtig dimensioniert werden
- Wahl der richtigen Technologie unter Berücksichtigung des Energieverbrauchs
- Daten:
  - Logging reduzieren
  - Klassifizierung von Daten nach Heiss, warm und Kalt
  - Nutzung von Binäre Datenformate wie protobuf oder mindesten die JSON-Komprimieren JSON
  - Caching von statischen Daten
- Nutzung von Binären Protokollen wie gprc
- Unnötigen Umgebungen müssen abgeschaltet werden
- Domain Driven Design: Zirkularitäten zwischen den einzelnen Komponenten aber auch Klassen
- Regeln der Micro-Service Architektur einhalten
- Nachhaltigkeits-Regeln sind architektonisch nicht eingehalten





# Green Software

*Powered by CAST Software Intelligence*

Diaa Shaheen  
Senior Solution Architect

Andreas Druzovic  
Senior Account Executive

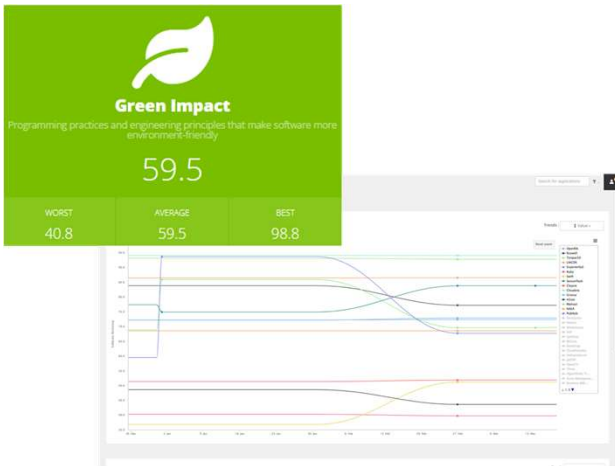
# How do you make your software greener?

- Now, there is a way to automatically analyze software source code for Green impact.
- **Software Intelligence\*** is required to understand how to reduce greenhouse gas emissions from the *software engineering* of applications.
- CAST **automatically** understands your source code and provides first-of-a-kind Green Software insights and sustainability recommendations.

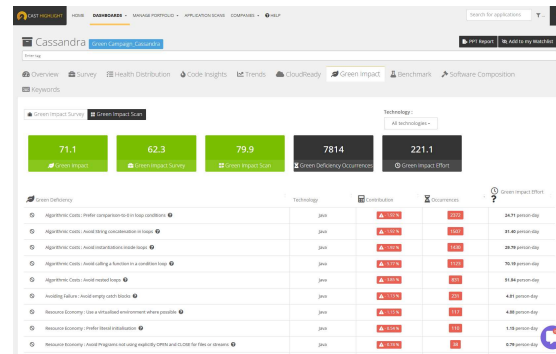


Example Green Deficiency code pattern:  
consuming unnecessary CPU cycles (requiring more power) by using SQL queries inside a loop

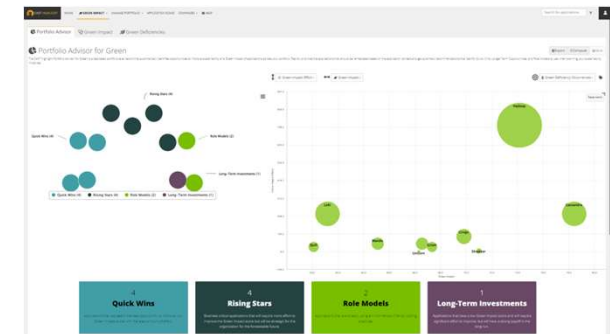
# What Green Software Insights are available?



**Green Impact Score** Automatically derived scores at both portfolio and application levels, with trends over time



**Green Deficiencies** Patterns found in the source code that contribute to excess use of resources/energy, with suggested remediation and effort estimates



**Automated Recommendations** Prioritized actions across all applications, including quick wins and actions with the biggest impact

# Green deficiency code patterns in detail

- Green deficiency code patterns are organized into a few categories:
  - Data access efficiency
  - Algorithmic costs
  - Resource economy
  - Avoiding failure

- Sample Green deficiency code patterns:

- Avoid using SQL queries inside a loop
  - Avoid indirect String concatenation inside loops
  - Avoid empty catch blocks
  - Avoid SQL queries using functions in the WHERE clause
  - Avoid using 'System.gc' and 'Runtime.gc'
  - Avoid "SELECT \*" queries

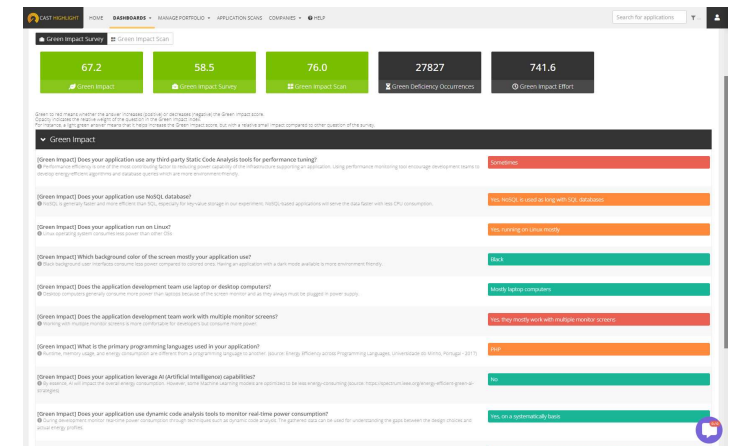
The screenshot displays the CAST application scan dashboard. At the top, there are navigation tabs for 'Green Impact Survey' and 'Green Impact Scan'. Below these are five summary cards: 'Green Impact' (67.2), 'Green Impact Survey' (58.5), 'Green Impact Scan' (76.0), 'Green Deficiency Occurrences' (27827), and 'Green Impact Effort' (741.6). The main section is a table of Green Deficiency patterns, filtered by 'All technologies'. The table has columns for the deficiency name, Technology, Contribution, Occurrences, and Green Impact Effort.

Green Deficiency	Technology	Contribution	Occurrences	Green Impact Effort
Algorithmic Costs - Avoid String concatenation in loops	Java	▲ 0.9%	8117	109.06 person-day
Algorithmic Costs - Prefer comparison-oid in loop conditions	Java	▲ 0.9%	2042	61.98 person-day
Algorithmic Costs - Avoid instantiations inside loops	Java	▲ 0.8%	4301	99.23 person-day
Algorithmic Costs - Avoid calling a function in a condition loop	Java	▲ 0.8%	2046	105.00 person-day
Algorithmic Costs - Avoid nested loops	Java	▲ 1.3%	3953	115.81 person-day
Resource Economy - Use a simulated environment where possible	Java	▲ 1.5%	3701	64.75 person-day
Avoiding Failure - Avoid empty catch blocks	Java	▲ 0.9%	1492	31.88 person-day
Resource Economy - Avoid OPEN/CLOSE inside loops	Java	▲ 0.8%	418	17.33 person-day
Resource Economy - Prefer literal initialization	Java	▲ 0.9%	817	4.29 person-day
Resource Economy - Avoid Programs not using explicitly OPEN and CLOSE for files or streams	Java	▲ 0.9%	280	5.44 person-day
Algorithmic Costs - Prefer comparison-oid in loop conditions	JavaScript	▲ 1.3%	239	3.49 person-day
Algorithmic Costs - Avoid String concatenation in loops	JavaScript	▲ 1.3%	56	1.17 person-day
Algorithmic Costs - Avoid nested loops	JavaScript	▲ 0.2%	44	3.79 person-day
Algorithmic Costs - Avoid calling a function in a condition loop	JavaScript	▲ 0.7%	17	1.06 person-day
Resource Economy - Avoid using System.gc and Runtime.gc	Java	▲ 0.9%	14	0.83 person-day
Resource Economy - Use a simulated environment where possible	JavaScript	▲ 0.3%	9	0.17 person-day
Algorithmic Costs - Avoid instantiations inside loops	JavaScript	▲ 0.9%	5	0.96 person-day
Avoiding Failure - Avoid empty catch blocks	JavaScript	▲ 0.2%	2	0.04 person-day

# Green Software application survey question examples

Additional context is overlaid on top of the automated code scan with a built-in survey:

- Does your application use dynamic code analysis tools to monitor real-time power consumption?
- Does your application use cache policy to minimize data exchange?
- Does your application leverage AI (Artificial Intelligence) capabilities?
- Does your application have data purge mechanism in place?
- Is your application running on your premises or in the Cloud?
- Does your application use NoSQL database?
- Does your application run on Linux?
- ...





# Green Deficiency Code Patterns and Effort Estimates

CAST HIGHLIGHT | HOME | DASHBOARDS | MANAGE PORTFOLIO | APPLICATION SCANS | COMPANIES | HELP

Search for applications

Cassandra | Green Campaign, Cassandra | PPT Report | Add to my Watchlist

Enter tag

Overview | Survey | Health Distribution | Code Insights | Trends | CloudReady | Green Impact | Benchmark | Software Composition

Keywords

Green Impact Survey | Green Impact Scan

Technology: All technologies -

71.1	62.3	79.9	7814	221.1
Green Impact	Green Impact Survey	Green Impact Scan	Green Deficiency Occurrences	Green Impact Effort

Green Deficiency

Green Deficiency	Technology	Contribution	Occurrences	Green Impact Effort
Algorithmic Costs: Prefer comparison-to-0 in loop conditions	java	-1.92%	2372	24.71 person-day
Algorithmic Costs: Avoid String concatenation in loops	java	-1.92%	1507	31.40 person-day
Algorithmic Costs: Avoid instantiations inside loops	java	-1.92%	1430	29.79 person-day
Algorithmic Costs: Avoid calling a function in a condition loop	java	-5.77%	1123	70.19 person-day
Algorithmic Costs: Avoid nested loops	java	-3.85%	831	51.94 person-day
Avoiding Failure: Avoid empty catch blocks	java	-1.13%	231	4.81 person-day
Resource Economy: Use a virtualised environment where possible	java	-1.15%	117	4.88 person-day
Resource Economy: Prefer literal initialisation	java	-0.54%	110	1.15 person-day
Resource Economy: Avoid Programs not using explicitly OPEN and CLOSE for files or streams	java	-0.74%	38	0.79 person-day

Algorithmic Costs

### Avoid String concatenation in loops

Avoiding concatenation in loop loops. Once string concatenation is used in a loop, it creates unnecessary temporary objects and results in slower runtime. It is a loop, instead using concatenation will save memory to add and join the rest after the loop terminates on the next iteration to concatenate.

Reference

Searched Code Pattern

```

1 concatenation operators considered are + and +
2 searched patterns are concatenation operators used with String string.
3
4 concatenation operators considered are + and +
5 searched patterns are concatenation operators used with String string.
6
7 concatenation operators considered are + and +
8 searched patterns are concatenation operators used with String string.
9
10 concatenation operators considered are + and +
11 searched patterns are concatenation operators used with String string.
12
13 concatenation operators considered are + and +
14 searched patterns are concatenation operators used with String string.
15
16 concatenation operators considered are + and +
17 searched patterns are concatenation operators used with String string.
18
19 concatenation operators considered are + and +
20 searched patterns are concatenation operators used with String string.
21
22 concatenation operators considered are + and +
23 searched patterns are concatenation operators used with String string.
24
25 concatenation operators considered are + and +
26 searched patterns are concatenation operators used with String string.
27
28 concatenation operators considered are + and +
29 searched patterns are concatenation operators used with String string.
30
31 concatenation operators considered are + and +
32 searched patterns are concatenation operators used with String string.
33
34 concatenation operators considered are + and +
35 searched patterns are concatenation operators used with String string.
36
37 concatenation operators considered are + and +
38 searched patterns are concatenation operators used with String string.
39
40 concatenation operators considered are + and +
41 searched patterns are concatenation operators used with String string.
42
43 concatenation operators considered are + and +
44 searched patterns are concatenation operators used with String string.
45
46 concatenation operators considered are + and +
47 searched patterns are concatenation operators used with String string.
48
49 concatenation operators considered are + and +
50 searched patterns are concatenation operators used with String string.
51
52 concatenation operators considered are + and +
53 searched patterns are concatenation operators used with String string.
54
55 concatenation operators considered are + and +
56 searched patterns are concatenation operators used with String string.
57
58 concatenation operators considered are + and +
59 searched patterns are concatenation operators used with String string.
60
61 concatenation operators considered are + and +
62 searched patterns are concatenation operators used with String string.
63
64 concatenation operators considered are + and +
65 searched patterns are concatenation operators used with String string.
66
67 concatenation operators considered are + and +
68 searched patterns are concatenation operators used with String string.
69
70 concatenation operators considered are + and +
71 searched patterns are concatenation operators used with String string.
72
73 concatenation operators considered are + and +
74 searched patterns are concatenation operators used with String string.
75
76 concatenation operators considered are + and +
77 searched patterns are concatenation operators used with String string.
78
79 concatenation operators considered are + and +
80 searched patterns are concatenation operators used with String string.
81
82 concatenation operators considered are + and +
83 searched patterns are concatenation operators used with String string.
84
85 concatenation operators considered are + and +
86 searched patterns are concatenation operators used with String string.
87
88 concatenation operators considered are + and +
89 searched patterns are concatenation operators used with String string.
90
91 concatenation operators considered are + and +
92 searched patterns are concatenation operators used with String string.
93
94 concatenation operators considered are + and +
95 searched patterns are concatenation operators used with String string.
96
97 concatenation operators considered are + and +
98 searched patterns are concatenation operators used with String string.
99
100 concatenation operators considered are + and +
101 searched patterns are concatenation operators used with String string.
102
103 concatenation operators considered are + and +
104 searched patterns are concatenation operators used with String string.
105
106 concatenation operators considered are + and +
107 searched patterns are concatenation operators used with String string.
108
109 concatenation operators considered are + and +
110 searched patterns are concatenation operators used with String string.
111
112 concatenation operators considered are + and +
113 searched patterns are concatenation operators used with String string.
114
115 concatenation operators considered are + and +
116 searched patterns are concatenation operators used with String string.
117
118 concatenation operators considered are + and +
119 searched patterns are concatenation operators used with String string.
120
121 concatenation operators considered are + and +
122 searched patterns are concatenation operators used with String string.
123
124 concatenation operators considered are + and +
125 searched patterns are concatenation operators used with String string.
126
127 concatenation operators considered are + and +
128 searched patterns are concatenation operators used with String string.
129
130 concatenation operators considered are + and +
131 searched patterns are concatenation operators used with String string.
132
133 concatenation operators considered are + and +
134 searched patterns are concatenation operators used with String string.
135
136 concatenation operators considered are + and +
137 searched patterns are concatenation operators used with String string.
138
139 concatenation operators considered are + and +
140 searched patterns are concatenation operators used with String string.
141
142 concatenation operators considered are + and +
143 searched patterns are concatenation operators used with String string.
144
145 concatenation operators considered are + and +
146 searched patterns are concatenation operators used with String string.
147
148 concatenation operators considered are + and +
149 searched patterns are concatenation operators used with String string.
150
151 concatenation operators considered are + and +
152 searched patterns are concatenation operators used with String string.
153
154 concatenation operators considered are + and +
155 searched patterns are concatenation operators used with String string.
156
157 concatenation operators considered are + and +
158 searched patterns are concatenation operators used with String string.
159
160 concatenation operators considered are + and +
161 searched patterns are concatenation operators used with String string.
162
163 concatenation operators considered are + and +
164 searched patterns are concatenation operators used with String string.
165
166 concatenation operators considered are + and +
167 searched patterns are concatenation operators used with String string.
168
169 concatenation operators considered are + and +
170 searched patterns are concatenation operators used with String string.
171
172 concatenation operators considered are + and +
173 searched patterns are concatenation operators used with String string.
174
175 concatenation operators considered are + and +
176 searched patterns are concatenation operators used with String string.
177
178 concatenation operators considered are + and +
179 searched patterns are concatenation operators used with String string.
180
181 concatenation operators considered are + and +
182 searched patterns are concatenation operators used with String string.
183
184 concatenation operators considered are + and +
185 searched patterns are concatenation operators used with String string.
186
187 concatenation operators considered are + and +
188 searched patterns are concatenation operators used with String string.
189
190 concatenation operators considered are + and +
191 searched patterns are concatenation operators used with String string.
192
193 concatenation operators considered are + and +
194 searched patterns are concatenation operators used with String string.
195
196 concatenation operators considered are + and +
197 searched patterns are concatenation operators used with String string.
198
199 concatenation operators considered are + and +
200 searched patterns are concatenation operators used with String string.
201
202 concatenation operators considered are + and +
203 searched patterns are concatenation operators used with String string.
204
205 concatenation operators considered are + and +
206 searched patterns are concatenation operators used with String string.
207
208 concatenation operators considered are + and +
209 searched patterns are concatenation operators used with String string.
210
211 concatenation operators considered are + and +
212 searched patterns are concatenation operators used with String string.
213
214 concatenation operators considered are + and +
215 searched patterns are concatenation operators used with String string.
216
217 concatenation operators considered are + and +
218 searched patterns are concatenation operators used with String string.
219
220 concatenation operators considered are + and +
221 searched patterns are concatenation operators used with String string.
222
223 concatenation operators considered are + and +
224 searched patterns are concatenation operators used with String string.
225
226 concatenation operators considered are + and +
227 searched patterns are concatenation operators used with String string.
228
229 concatenation operators considered are + and +
230 searched patterns are concatenation operators used with String string.
231
232 concatenation operators considered are + and +
233 searched patterns are concatenation operators used with String string.
234
235 concatenation operators considered are + and +
236 searched patterns are concatenation operators used with String string.
237
238 concatenation operators considered are + and +
239 searched patterns are concatenation operators used with String string.
240
241 concatenation operators considered are + and +
242 searched patterns are concatenation operators used with String string.
243
244 concatenation operators considered are + and +
245 searched patterns are concatenation operators used with String string.
246
247 concatenation operators considered are + and +
248 searched patterns are concatenation operators used with String string.
249
250 concatenation operators considered are + and +
251 searched patterns are concatenation operators used with String string.
252
253 concatenation operators considered are + and +
254 searched patterns are concatenation operators used with String string.
255
256 concatenation operators considered are + and +
257 searched patterns are concatenation operators used with String string.
258
259 concatenation operators considered are + and +
260 searched patterns are concatenation operators used with String string.
261
262 concatenation operators considered are + and +
263 searched patterns are concatenation operators used with String string.
264
265 concatenation operators considered are + and +
266 searched patterns are concatenation operators used with String string.
267
268 concatenation operators considered are + and +
269 searched patterns are concatenation operators used with String string.
270
271 concatenation operators considered are + and +
272 searched patterns are concatenation operators used with String string.
273
274 concatenation operators considered are + and +
275 searched patterns are concatenation operators used with String string.
276
277 concatenation operators considered are + and +
278 searched patterns are concatenation operators used with String string.
279
280 concatenation operators considered are + and +
281 searched patterns are concatenation operators used with String string.
282
283 concatenation operators considered are + and +
284 searched patterns are concatenation operators used with String string.
285
286 concatenation operators considered are + and +
287 searched patterns are concatenation operators used with String string.
288
289 concatenation operators considered are + and +
290 searched patterns are concatenation operators used with String string.
291
292 concatenation operators considered are + and +
293 searched patterns are concatenation operators used with String string.
294
295 concatenation operators considered are + and +
296 searched patterns are concatenation operators used with String string.
297
298 concatenation operators considered are + and +
299 searched patterns are concatenation operators used with String string.
300

```

Algorithmic Costs: Avoiding concatenation in loops	java/string	-2.76%	0	0.00 person-day
Algorithmic Costs: Prefer comparison-to-0 in loop conditions	java/string	-2.76%	0	0.00 person-day
Avoiding Failure: Avoid empty catch blocks	java/string	-2.76%	0	0.00 person-day

CAST HIGHLIGHT | HOME | DASHBOARDS | MANAGE PORTFOLIO | APPLICATION SCANS | COMPANIES | HELP

Search for applications

Cassandra | Green Campaign, Cassandra | PPT Report | Add to my Watchlist

Enter tag

Overview | Survey | Health Distribution | Code Insights | Trends | CloudReady | Green Impact | Benchmark | Software Composition | Keywords

Green Impact Survey | Green Impact Scan

Technology: All technologies -

71.1	62.3	79.9	7814	221.1
Green Impact	Green Impact Survey	Green Impact Scan	Green Deficiency Occurrences	Green Impact Effort

Green Deficiency

Green Impact

Green Impact: Does your application use any third party Static Code Analysis tools for performance testing? Not yet selected

Green Impact: Does your application use any third party Static Code Analysis tools for performance testing? Yes, only JetBrains

Green Impact: Does your application use any third party Static Code Analysis tools for performance testing? Yes, always running on our

Green Impact: Which background color of the screen mostly your application use? Dark

Green Impact: Does the application development team use laptop or desktop computer? Mostly laptop computers

Green Impact: Does the application development team work with multiple monitor screens? Yes, they mostly work with multiple monitor screens

Green Impact: What is the primary programming languages used in your application? CA

Green Impact: Does your application leverage AI (Artificial Intelligence) capabilities? Yes, the AI model is optimized for lower energy consumption

## Praxisbeispiel – wie kann die komplette Software mit CAST analysiert werden....



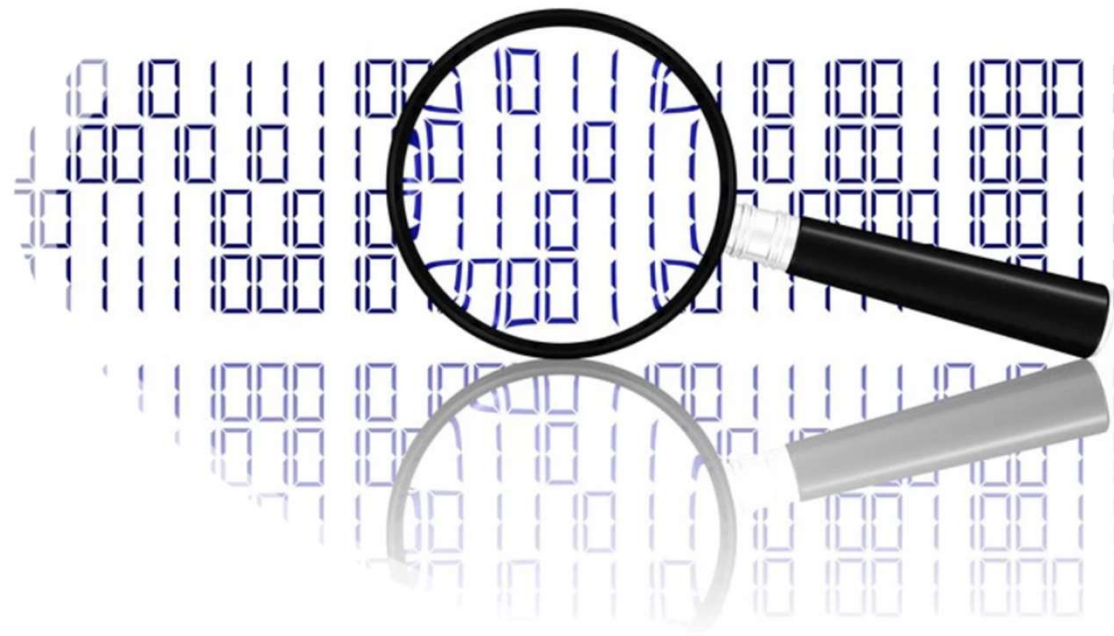


## Gruppen Austausch Teil 2

---

## Gruppen Austausch Teil 2

- Was sind aus Ihrer Sicht die nächsten Schritte in Richtung Green Software ?
- Wie kann man aus IT-Sicht das Business mit ins Boot nehmen?
- Wo sind allenfalls die Low-hanging-fruits bzw. mögliche Quick Wins?



## Zusammenfassung und Ausblick

---

# Zusammenfassung und Ausblick

Mittels einer Architektur- und Source Code Analyse von einer Software oder einem gesamten Softwareportfolio mit Fokus Nachhaltigkeit, kann...

- das mögliche „grüne Verbesserungspotential“ einfach identifiziert und verbessert werden
- von Beginn weg die Nachhaltigkeit eine aktive Rolle in der Überlegung einnehmen – „Sustainability by Design“
- die Software-Abteilung oder der Software Dienstleister einen Beitrag zu den Nachhaltigkeitszielen der Unternehmung leisten
- ein Beitrag zur globalen Klimalösung geleistet werden, mittels indirekter Reduktion von Treibhausgasen und Einsparungen von Energie

## Wie viel Strom braucht Ihre Software?

Dominik Strub-Tiedt,  
Faysal Cherradi & Pascal Sieber

Technologie spielt dabei eine entscheidende Rolle, denn skalierbare Geschäftsprozesse sind durch IT unterstützt. Dies kann einerseits zu Energieeinsparungen in der Abwicklung führen, andererseits steigt aber der Energieverbrauch in der IT enorm an. 2030 werden schätzungsweise 21 Prozent des globalen Stromverbrauchs durch IT konsumiert werden.<sup>1</sup> Climatiq geht davon aus, dass die Data Centers 2,5 Prozent bis 3,7 Prozent der globalen CO<sub>2</sub>-Emissionen verursachen. Die kommerzielle Luftfahrt verursacht im Vergleich dazu 2,4 Prozent (vgl. Abbildung 1). 85 Prozent dieser CO<sub>2</sub>-Emissionen können im Design und mit der Entwicklung der Software beeinflusst und damit reduziert werden. Dabei spielen die Architektur und Technologien entscheidende Rollen. Durch eine nachhaltige Gestaltung und Planung der Architektur können ökologische Prinzipien in

den gesamten Lebenszyklus von Applikationen und System-Landschaften appliziert werden, was die Einsparungen multipliziert.

### Kleine Komponenten brauchen weniger Strom

Eine effektive Strategie zur Verbesserung der Nachhaltigkeit der Software besteht in der Zerlegung von Softwarekomponenten in kleinere Stücke. Dieser Ansatz bietet den Vorteil, dass kleinere Stücke mit weniger Hardwareanforderungen auskommen können, was Ressourceneinsparungen und eine bessere Skalierbarkeit ermöglicht. Allerdings ist zu beachten, dass dies nicht immer der Fall ist, da einzelne Komponenten manchmal sehr hohe Hardwareanforderungen haben können.

Um den optimalen Schnitt der Lösung zu erreichen und klare Verantwortlichkeiten fest-

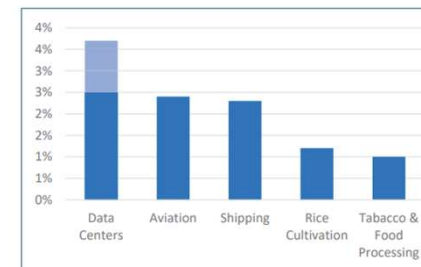
Der IT-Markt gilt angesichts des Wandels von der Industrie- zur Informationsgesellschaft als einer der dynamischsten Wachstumsmärkte weltweit. Die Globalisierung erhöht stetig den Wettbewerbsdruck, was Unternehmen dazu zwingt, effiziente Prozesse und Kostenstrukturen zu entwickeln. Nur so gelingt es ihnen, schnell genug auf die Marktveränderungen und Kundenbedürfnisse reagieren zu können.

zulegen, kann das Konzept des Domain Driven Design (DDD) angewendet werden. DDD legt den Fokus darauf, die Domäne und deren Fachlichkeit in den Mittelpunkt der Softwareentwicklung zu stellen. Durch die klare Abgrenzung der Domänen und die Definition wird eine bessere Modularisierung und Abstraktion erreicht. Durch die Zerlegung der Softwarekomponenten in kleinere Stücke und die Anwendung von DDD können Softwarearchitekten und -entwickler effizientere und wartbarere Lösungen entwickeln. Dies ermöglicht eine bessere Skalierbarkeit, verbesserte Wiederverwendbarkeit von Komponenten und eine klarere Zuweisung von Verantwortlichkeiten, was letztendlich zu einer besseren Gesamtlösung führt.

### Datenspeicher brauchen Strom, auch wenn die Daten nicht mehr gebraucht werden

Eine Software, die kaum technische Schulden aufweist, aber auf Informationsquellen mit hohen Datenspeicherkosten angewiesen ist, kann zu einem hohen Ressourcenverbrauch bei der Speicherung sowie beim Datenaustausch führen.

Eine nachhaltige Datenarchitektur beinhaltet eine effiziente Verwaltung von Daten. Diese umfasst die Identifizierung und Löschung nicht mehr benötigter, duplizierter oder veralteter Daten, um Speicherplatz freizugeben. Daten sollten auch entsprechend ihres Wertes und ihrer Aktualität priorisiert und entsprechend gespeichert werden. Durch eine effiziente Datenverwaltung können Speicherressourcen optimiert und



Greenhouse Gas Emissions in Data Centres<sup>1</sup>

# Ausblick Green Software



## Green Software Foundation

- Working Groups Standards, Policy, etc.
- Projects Green Software Maturity Matrix, etc.



## Green Software Design community

- Green Software Design Label



**Gut für mich.  
Gut für die Umwelt.**

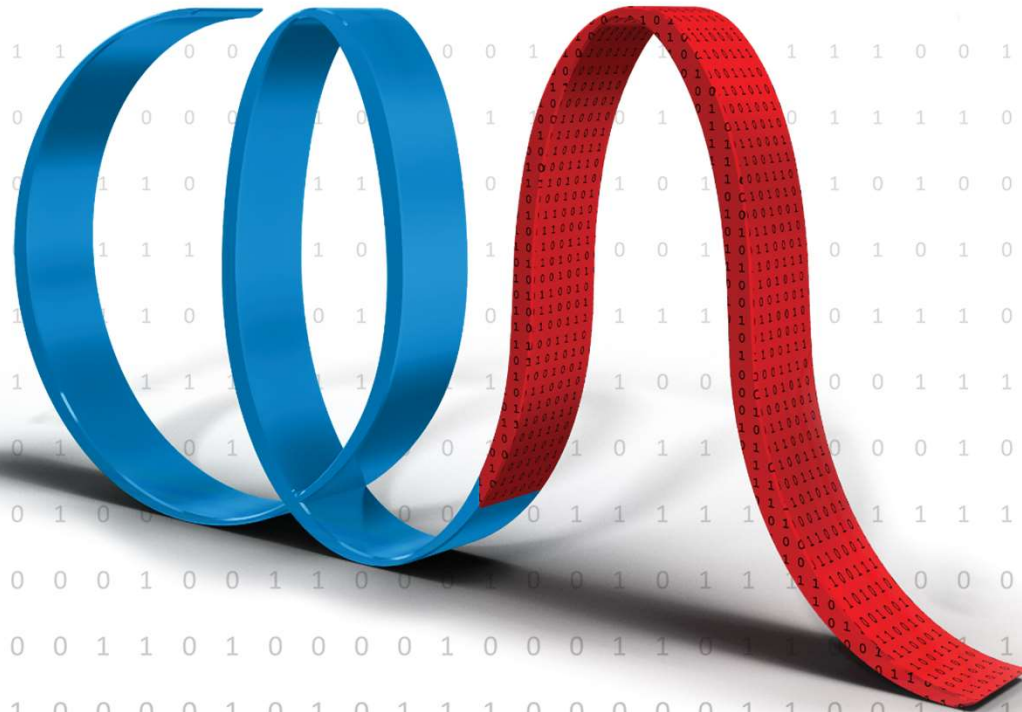
## Blauer Engel

- Label/Zertifizierung: Ressourcen- und energieeffiziente Softwareprodukte (DE-UZ 215)



## Swiss Software Industry Survey

- Ergebnisse „Software Sustainability“



**We lead from the emerging future!**



**Dominik Strub-Tiedt**

Transformation Consultant, Senior Manager  
Head Technology & Data

[dominik.strub-tiedt@sieberpartners.com](mailto:dominik.strub-tiedt@sieberpartners.com)

+41 79 866 01 17

Schwanengasse 1

3011 Bern

Usteristrasse 19

8001 Zürich



Bleiben wir  
in Kontakt!

**sieber & partners**



digital business excellence



**Benno Hübscher**

Senior Transformation Consultant

[Benno.Huebscher@sieberpartners.com](mailto:Benno.Huebscher@sieberpartners.com)

+41 31 566 93 00

Schwanengasse 1

3011 Bern

Usteristrasse 19

8001 Zürich



Bleiben wir  
in Kontakt!

**sieber & partners**



digital business excellence