Green IT - MMIGIT

Green IT as a driver for future viability

metafinanz
technologie. kultur. netzwerke.

Earth, The Climate and Everything

EcoCompute Conference 2024
26.04.2024

metafinanz Contacts for GreenIT

Ghazal Aakel

Expert GreenIT

Ghazal.Aakel@metafinanz.de

+ 49 89 3605316444

Eric Jochum

Expert GreenIT

Eric.Jochum@metafinanz.de

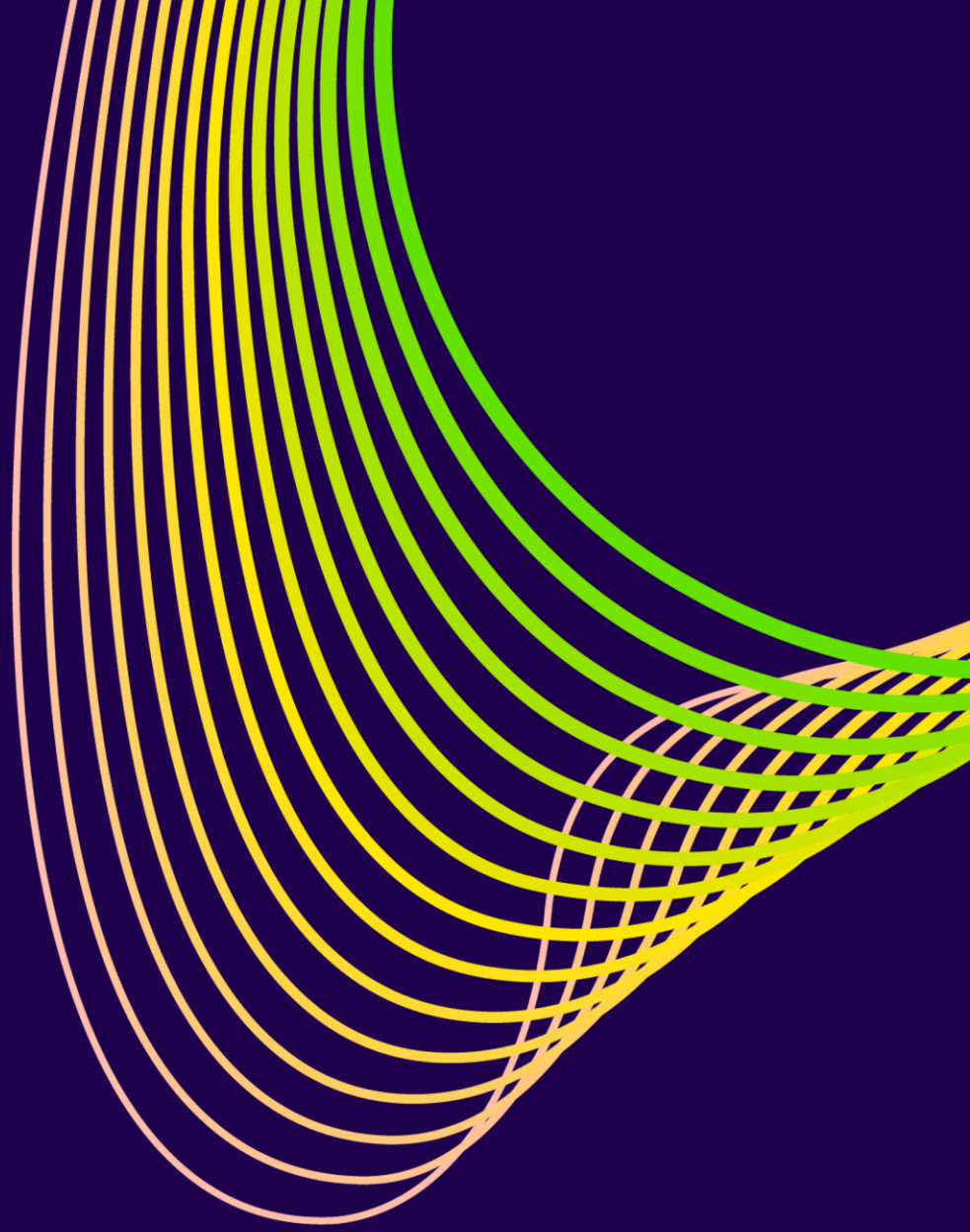
+ 49 89 3605316205

Green IT as a driver for future viability

Agenda

Green IT as a driver for future viability

Introduction

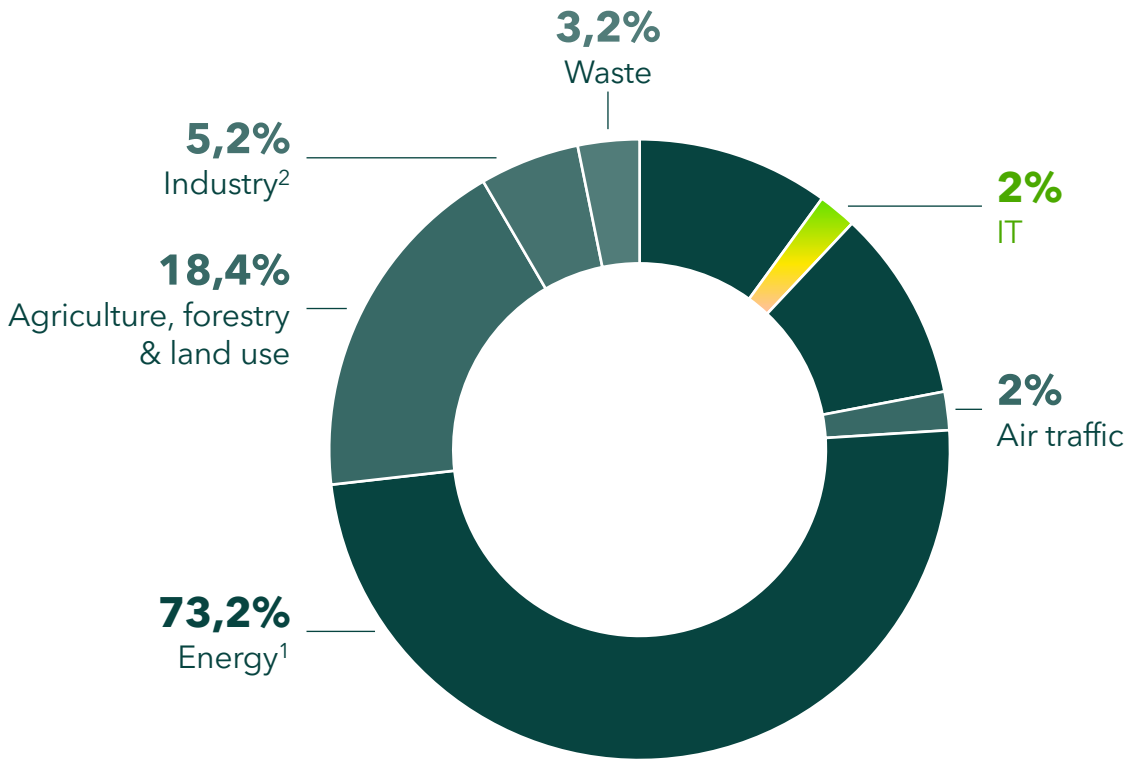


metafinanz

Increasing share of IT in GHG emissions

Global greenhouse gas emissions per sector in 2016 with a total of 49.4 billion tons of CO2 emissions.

The Federal Environment Agency assumes that the **share of IT could reach 26% by 2030 due to the massive increase in digitalization.**



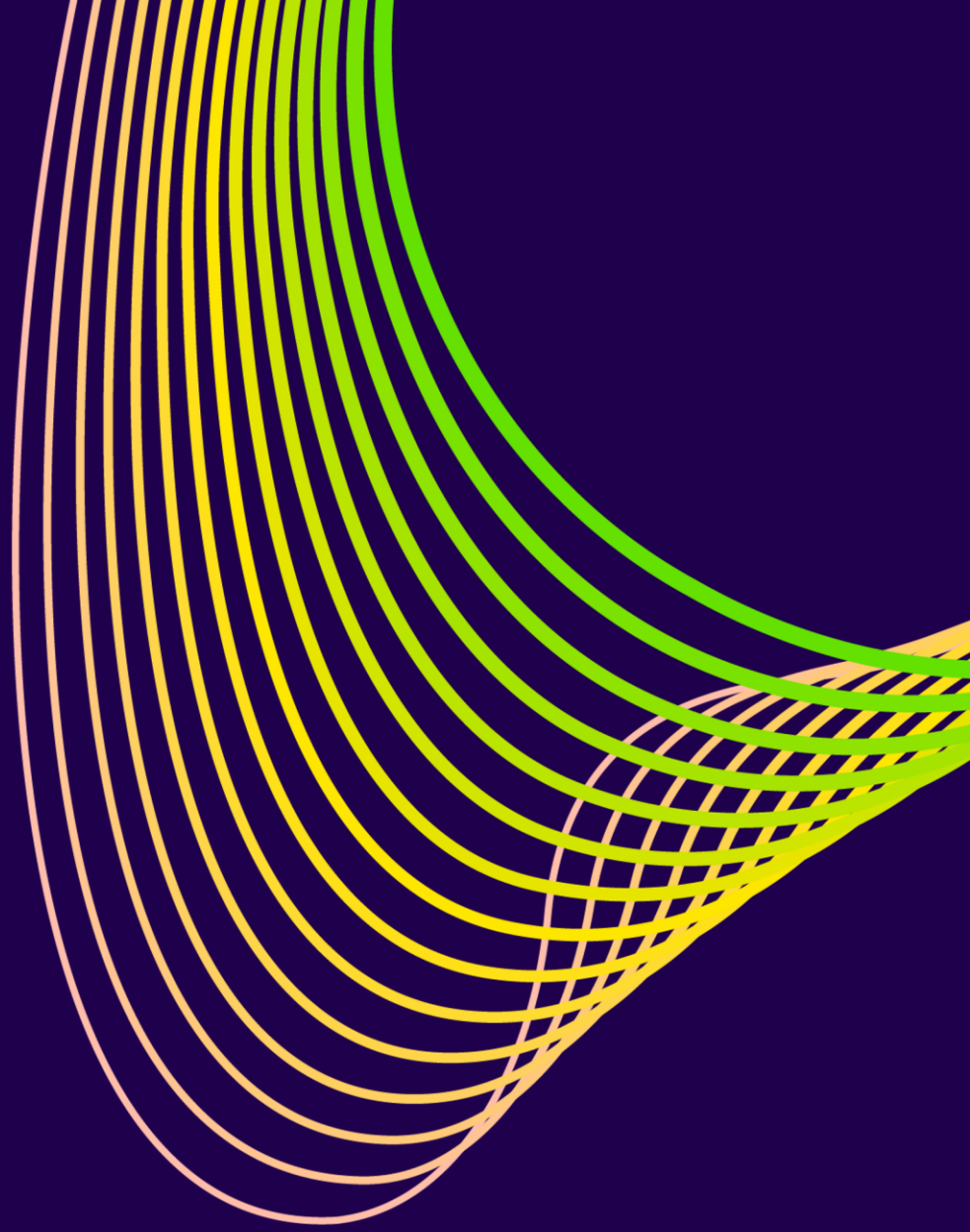
Souerces: Freitag, C. et al. (2021): Review The real climate and transformative impact of ICT: A critique of estimates, trends, and regulations: https://www.sciencedirect.com/science/article/pii/S2666389921001884, Emissionen je Sektor: https://ourworldindata.org/emissions-by-sector

1 Energy (= Power, Heat, Transport) | **2** Industry (= direct industriell processes)

metafinanz

Green IT as a driver for future viability

Energy Consumption – Awareness



metafinanz

CO_2 of data centers as the equivalent of moving cars

16 TWh power consumption in german data centers (2022)

→ 7.408.000.000 kg CO_2

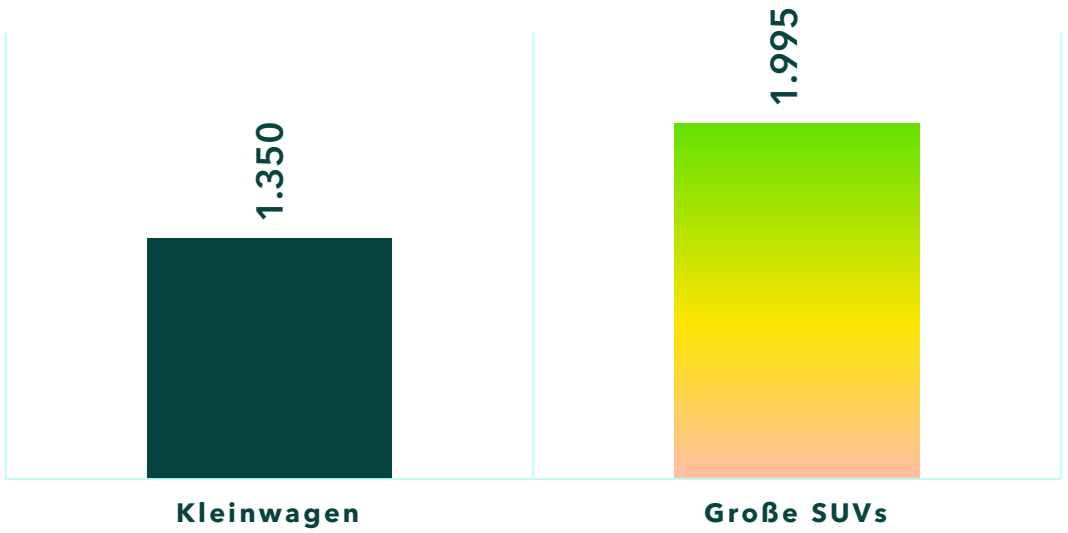
- ~ 5.487.407 Small car
- ~ 3.713.283 SUVs

Global Data Center Demand (600 TWh)

- With a German electricity mix (463g CO2/kWh):
  → 277.800.000.000 kg CO2
- ~ 205.800.000 Small Car
- ~ 104.160.000 SUVs

Co_2 emission of cars[1]

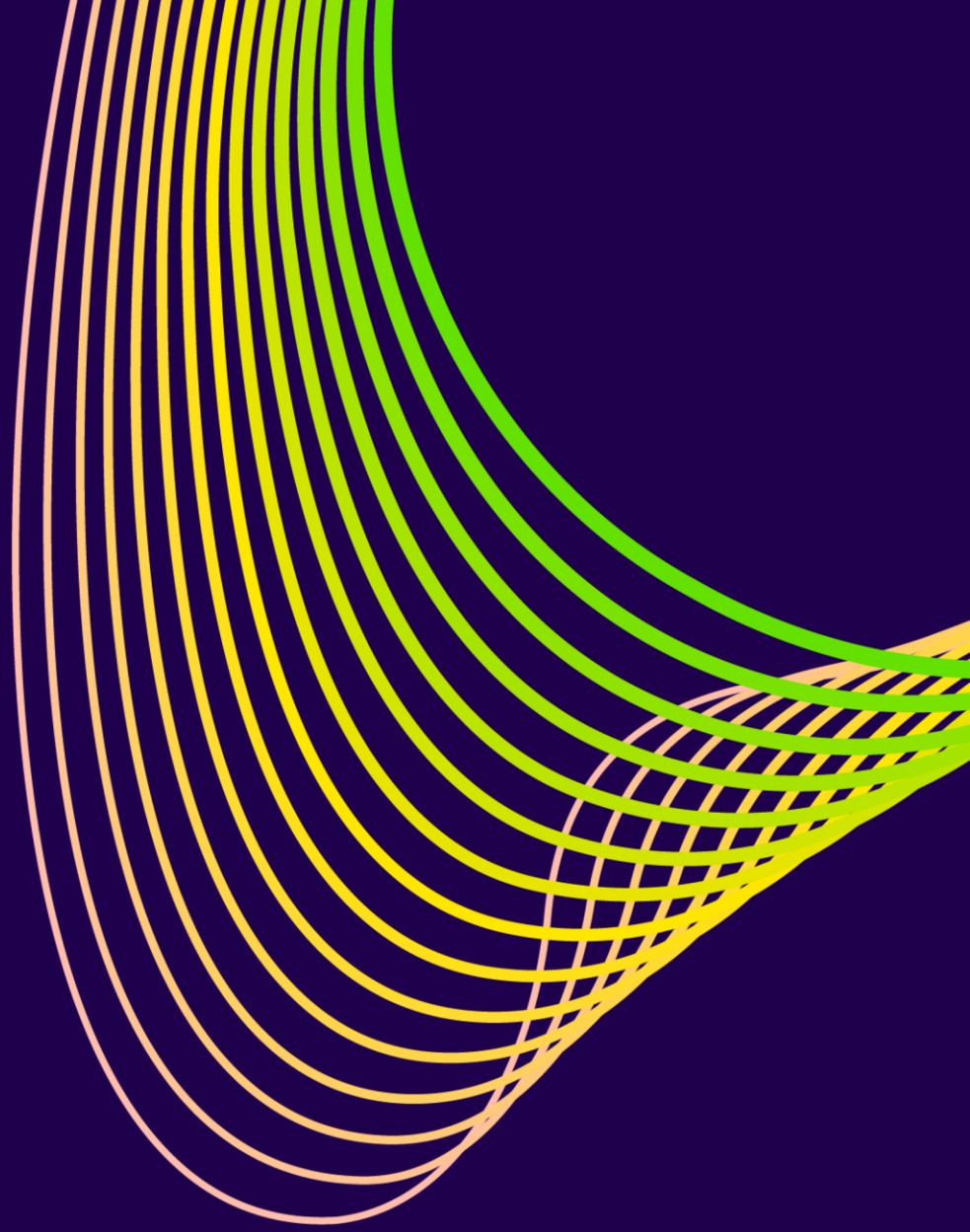
Estimated average CO2 emissions from passenger cars in Germany 2022 (in kg CO_2/Year with 15 tkm)



1 With an annual mileage of 15,000 km | Source: Statista Mobility Market Outlook

Green IT as a driver for future viability

Green IT – New thinking and acting



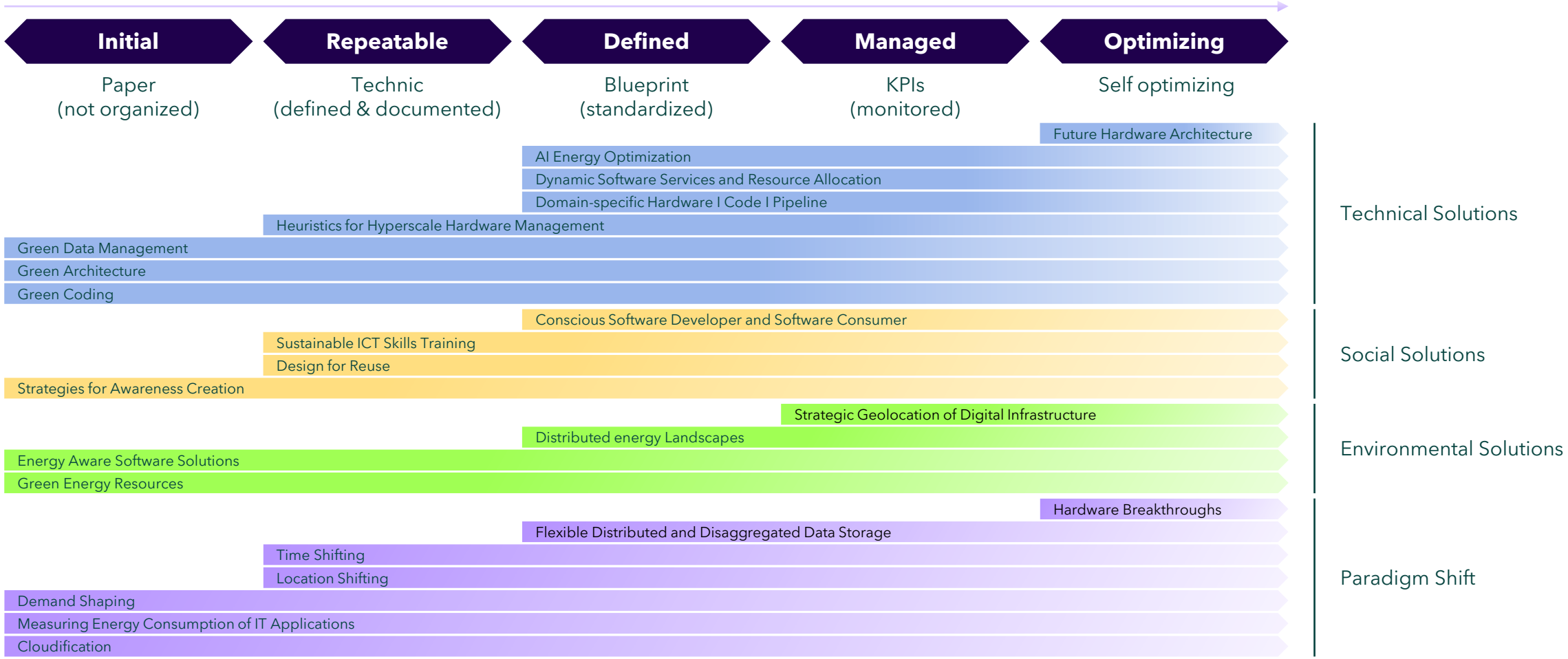
metafinanz

Digitalization

- Digitalization refers to the process of converting analogue information, data and processes into digital formats.
- Digitalization has far-reaching effects on society and the economy and is seen as a driver of innovation, increased efficiency and growth.
- This is not possible without energy!

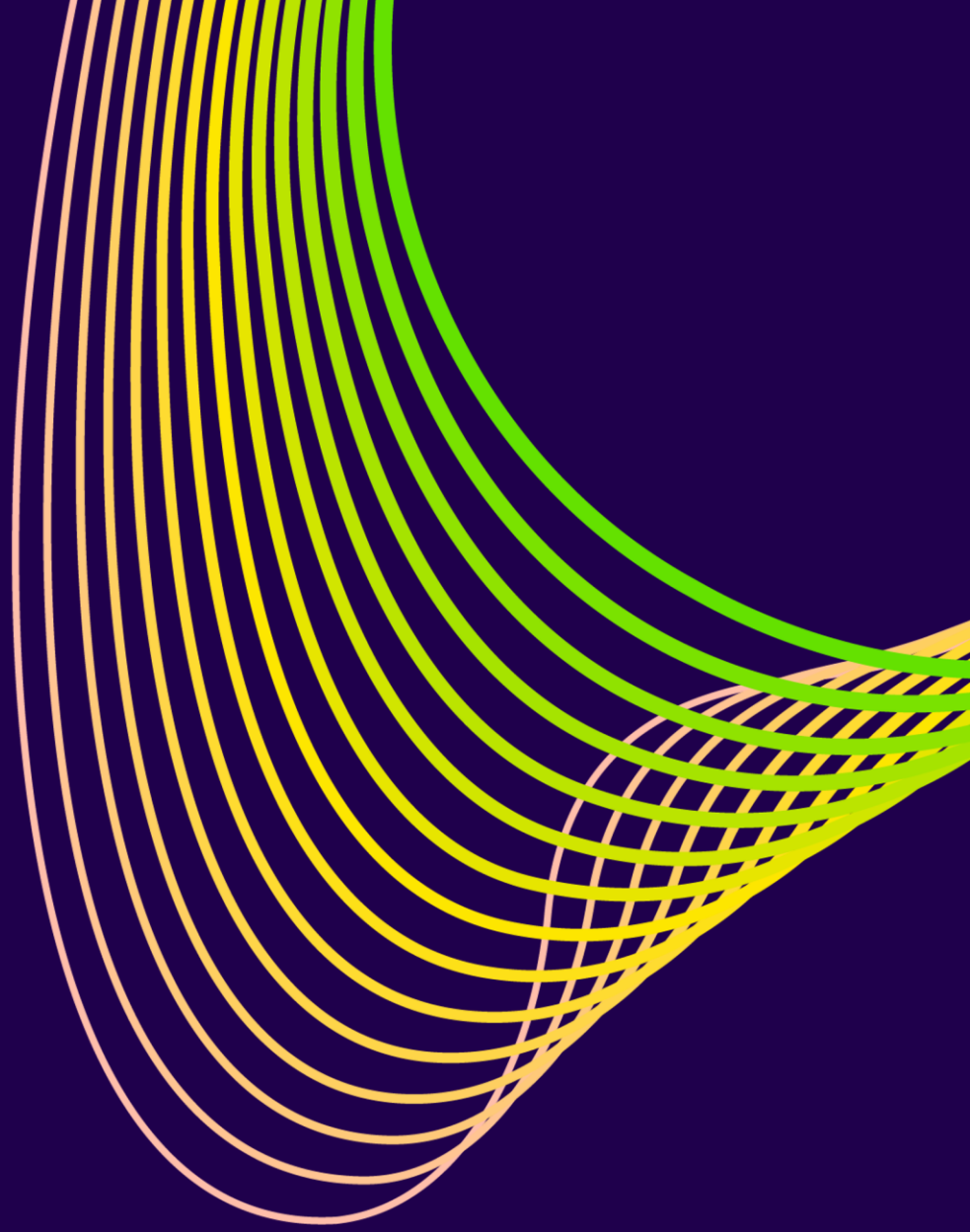
Green IT is the key to **sustainable digitalization** and **future viability**!

MMIGIT (Maturity Model Integrated Green IT) – Solutions Landscape Green IT



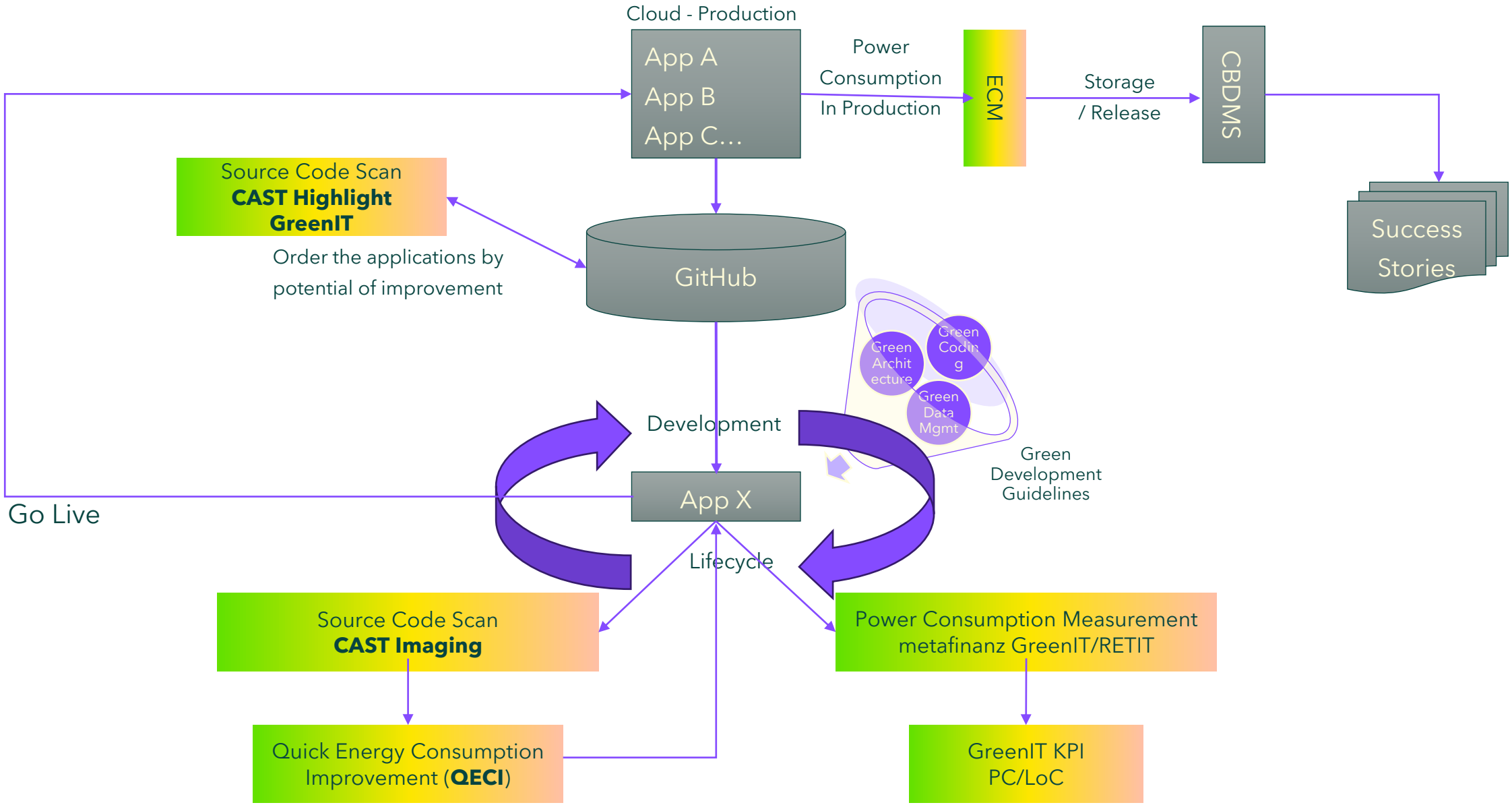
Green IT as a driver for future viability

Concrete Approaches



metafinanz

ALGIT: The Assembly Line of Green IT



metafinanz

Show Case Using Generative AI in Code Optimization

Model used: Generative Pre-trained Transformer

Setting the Context for the AI model to Optimize Code

- prompt the AI model to be familiar with the programming language and requiring its assistance.
- introduce the code to the AI model
- Instruct the model with Green Deficiencies rules (Guidelines) and best examples to improve its performance and accuracy.

The Goal is to use more accurate LLMs which are aware of the context

Program Size and CAST Highlight Findings	Numbers
Program Size (#LoC Lines of Code)	115.214
Number of Findings by CAST Highlight	233
Number of LoC / Finding	494 (every 494 LoC 1 Finding)

Findings Categories
Avoid instantiations inside loops
Avoid string concatenation in loops
Avoid nested loops
Prefer comparison-to-0 in loop conditions
Avoid calling a function in a condition loop
Avoid primitive type wrapper instantiation
Avoid Programs not using explicitly OPEN and CLOSE for files or streams

ALGIT – QECI – Example: Avoid Nested Loops

```
while ($j < $b) {  if ($i==$j) {  $res = $i*$j;  }  $j++;  }  $i++;  } }  sub withHashmap() {  my %hash = ();  my $j=0;  while ($j < $b) {  $hash{$j} = $i*$i;  $j++;  }  my $i = 0;  while ($i < $a) {  print STDERR "$i\n";  $res = $hash{i};  $i++;  } } # takes ~6 seconds withNestedLoops();  # takes ~1 seconds withHashmap();
 * STATUS: RESOLVED
 * CAST-Finding END #1
 **********/

// QECI Fix: Use HashMaps to avoid nested loops
Map<ProductAvailability, Set<ProductPrice>> availabilityMap = new HashMap<>();

// Populate the hashmap
Set<ProductAvailability> avails = product.getAvailabilities();
for (ProductAvailability as : avails) {
    Set<ProductPrice> availabilityPrices = as.getPrices();
    availabilityMap.put(as, new HashSet<>(availabilityPrices));
}

// Flatten the nested structure and iterate over the data set
avails.stream()
      .flatMap(as -> availabilityMap.get(as).stream())
      .forEach(ps -> System.out.println(ps.getProductPriceAmount().toString()));

/*
        Set<ProductAvailability> avails = product.getAvailabilities();
        for(ProductAvailability as : avails) {
                Set<ProductPrice> availabilityPrices = as.getPrices();
                for(ProductPrice ps : availabilityPrices) {
                    System.out.println(ps.getProductPriceAmount().toString());
                }
        }
-
        //check availability
        Set<ProductPrice> availabilityPrices = availability.getPrices();
        for(ProductPrice ps : availabilityPrices) {
@@ -265,4 +289,4 @@ public void testGetCustomShippingQuotesByWeight() throws ServiceException {
}
- }
+ */
+ }
```

- Results based on sum of 1000 iterations/API calls
- Reducing time complexity from O(n^2) to O(n)
- One enhancement affects 6 APIs
- → Reducing the energy conumption of an average of 42,78 mWh to 34,86 mWh
- → Energy Saving = 18,86%

ALGIT – QECI – Example: Avoid Primitive Type Wrapper

17 .../src/main/java/com/salesmanager/core/model/catalog/product/manufacturer/Manufacturer.java

```
@@ -52,8 +52,23 @@ public class Manufacturer extends SalesManagerEntity<Long, Manufacturer> impleme
        @Column(name = "MANUFACTURER_IMAGE")
        private String image;

-       @Column(name="SORT_ORDER")
-       private Integer order = new Integer(0);

        @ManyToOne(fetch = FetchType.EAGER)
        @JoinColumn(name="MERCHANT_ID", nullable=false)
```

```
        @Column(name = "MANUFACTURER_IMAGE")
        private String image;

+
+
+
+
+ /**********************************
+  * CAST-Finding START #1 (2024-02-02 12:30:38.874870):
+  * TITLE: Avoid primitive type wrapper instantiation
+  * DESCRIPTION: Literal values are built at compil time, and their value stored directly in the
variable. Literal strings also benefit from an internal mechanism of string pool, to prevent
useless duplication, according to the fact that literal string are immutable. On the contrary,
values created through wrapper type instantiation need systematically the creation of a new object
with many attributes and a life process to manage, and can lead to redondancies for identical
values.
+  * STATUS: RESOLVED
+  * CAST-Finding END #1
+  **********************************/
+
+

        @Column(name="SORT_ORDER")
+       // QECI Fix
+       private Integer order = 0;
+       // private Integer order = new Integer(0);

        @ManyToOne(fetch = FetchType.EAGER)
        @JoinColumn(name="MERCHANT_ID", nullable=false)
```

ALGIT – QECI – Example: Avoid String Concatination

```
185                                                                                                 204
186             }                                                                                   205             }
187                                                                                                 206
                @@ -255,17 +274,37 @@ public Transaction capture(MerchantStore store, Customer customer, Order order,
255                     com.braintreegateway.Transaction settledTransaction = result.getTarget();  274                     com.braintreegateway.Transaction settledTransaction = result.getTarget();
256                     trxId = settledTransaction.getId();                                         275                     trxId = settledTransaction.getId();
257             } else {                                                                            276             } else {
258 -               String errorString = "";                                                        277 +                   // QECI Fix: Use StringBuilder
259 -               for (ValidationError error : result.getErrors().getAllDeepValidationErrors()) { 278 +                   StringBuilder errorMsg = new StringBuilder();
260 -                  errorString += "Error: " + error.getCode() + ": " + error.getMessage() + "\n"; 279 +                   // String errorString = "";
261 -               }                                                                               280 +
                                                                                                    281 +                   errorMsg.append("Can't process Braintree refund ");
                                                                                                    282 +                   for (ValidationError error : result.getErrors().getAllDeepValidationErrors()) {
                                                                                                    283 +
                                                                                                    284 +                           /**********************************
                                                                                                    285 +                            * CAST-Finding START #2 (2024-02-02 12:30:42.222626):
                                                                                                    286 +                            * TITLE: Avoid string concatenation in loops
                                                                                                    287 +                            * DESCRIPTION: Avoid string concatenation inside loops.  Since strings
                                                                                                        are immutable, concatenation is a greedy operation. This creates unnecessary temporary objects and
                                                                                                        results in quadratic rather than linear running time. In a loop, instead using concatenation, add
                                                                                                        each substring to a list and join the list after the loop terminates (or, write each substring to
                                                                                                        a byte buffer).
                                                                                                    288 +                            * STATUS: RESOLVED
                                                                                                    289 +                            * CAST-Finding END #2
                                                                                                    290 +                            **********************************/
                                                                                                    291 +
                                                                                                    292 +                           // QECI Fix: Use StringBuilder
                                                                                                    293 +                           errorMsg.append("Error: ")
                                                                                                    294 +                                   .append(error.getCode())
                                                                                                    295 +                                   .append(": ")
                                                                                                    296 +                                   .append(error.getMessage())
                                                                                                    297 +                                   .append("\n");
                                                                                                    298 +                           // errorString += "Error: " + error.getCode() + ": " + error.getMessage()
                                                                                                        + "\n";
                                                                                                    299 +                   }
                                                                                                    300 +                   // QECI Fix: Use StringBuilder
                                                                                                    301 +                   String errorString = errorMsg.toString();
262                                                                                                 302
263 -                       IntegrationException te = new IntegrationException(                     303 +                   IntegrationException te = new IntegrationException(errorString);
264 -                                   "Can't process Braintree refund " + errorString);           304 +                   te.setExceptionType(IntegrationException.TRANSACTION_EXCEPTION);
265 -                       te.setExceptionType(IntegrationException.TRANSACTION_EXCEPTION);        305 +                   te.setMessageCode("message.payment.error");
266 -                       te.setMessageCode("message.payment.error");                             306 +                   te.setErrorCode(IntegrationException.TRANSACTION_EXCEPTION);
```

ALGIT – QECI – Example: Avoid Object Creation Inside Loops

40 ...a/com/salesmanager/core/business/modules/cms/product/aws/S3ProductContentFileManager.java

@@ -87,20 +87,52 @@ public List<OutputContentFile> getImages(String merchantStoreCode,

```
87        ListObjectsV2Request listObjectsRequest = new ListObjectsV2Request()
88            .withBucketName(bucketName).withPrefix(nodePath(merchantStoreCode));
89
90  -     List<OutputContentFile> files = null;
91        final AmazonS3 s3 = s3Client();
92        ListObjectsV2Result results = s3.listObjectsV2(listObjectsRequest);
93        List<S3ObjectSummary> objects = results.getObjectSummaries();
94        for (S3ObjectSummary os : objects) {
95  -       if (files == null) {
96  -         files = new ArrayList<OutputContentFile>();
97  -       }
98          String mimetype = URLConnection.guessContentTypeFromName(os.getKey());
99          if (!StringUtils.isBlank(mimetype)) {
100           S3Object o = s3.getObject(bucketName, os.getKey());
101           byte[] byteArray = IOUtils.toByteArray(o.getObjectContent());
```

```
87        ListObjectsV2Request listObjectsRequest = new ListObjectsV2Request()
88            .withBucketName(bucketName).withPrefix(nodePath(merchantStoreCode));
89
90  +       /*********************************
91  +        * CAST-Finding START #1 (2024-02-02 12:30:41.601250):
92  +        * TITLE: Avoid instantiations inside loops
93  +        * DESCRIPTION: Object instantiation uses memory allocation, that is a greedy operation.
    Doing an instantiation at each iteration could really hamper the performances and increase
    resource usage.  If the instantiated object is local to the loop, there is absolutely no need to
    instantiate it at each iteration : create it once outside the loop, and just change its value at
    each iteration. If the object is immutable, create if possible a mutable class. If the aim is to
    create a consolidated data structure, then, unless the need is to release the data case by case,
    it could be better to make a single global allocation outside the loop, and fill it with data
    inside the loop.
94  +        * STATUS: RESOLVED
95  +        * CAST-Finding END #1
96  +        *********************************/
97  +
98  +     // QECI Fix: Move instantiation outside loop
99  +     List<OutputContentFile> files = new ArrayList<OutputContentFile>();
100 +     // List<OutputContentFile> files = null;
101       final AmazonS3 s3 = s3Client();
102       ListObjectsV2Result results = s3.listObjectsV2(listObjectsRequest);
103       List<S3ObjectSummary> objects = results.getObjectSummaries();
104       for (S3ObjectSummary os : objects) {
105 +       // QECI Fix: Move instantiation outside loop
106 +       // if (files == null) {
107 +       //  files = new ArrayList<OutputContentFile>();
108 +       // }
109         String mimetype = URLConnection.guessContentTypeFromName(os.getKey());
110         if (!StringUtils.isBlank(mimetype)) {
111           S3Object o = s3.getObject(bucketName, os.getKey());
112           byte[] byteArray = IOUtils.toByteArray(o.getObjectContent());
```

metafinanz Contacts for GreenIT

Ghazal Aakel

Expert GreenIT

Ghazal.Aakel@metafinanz.de

+ 49 89 3605316444

Eric Jochum

Expert GreenIT

Eric.Jochum@metafinanz.de

+ 49 89 3605316205

Vielen Dank

metafinanz
technologie. kultur. netzwerke.

metafinanz Informationssysteme GmbH

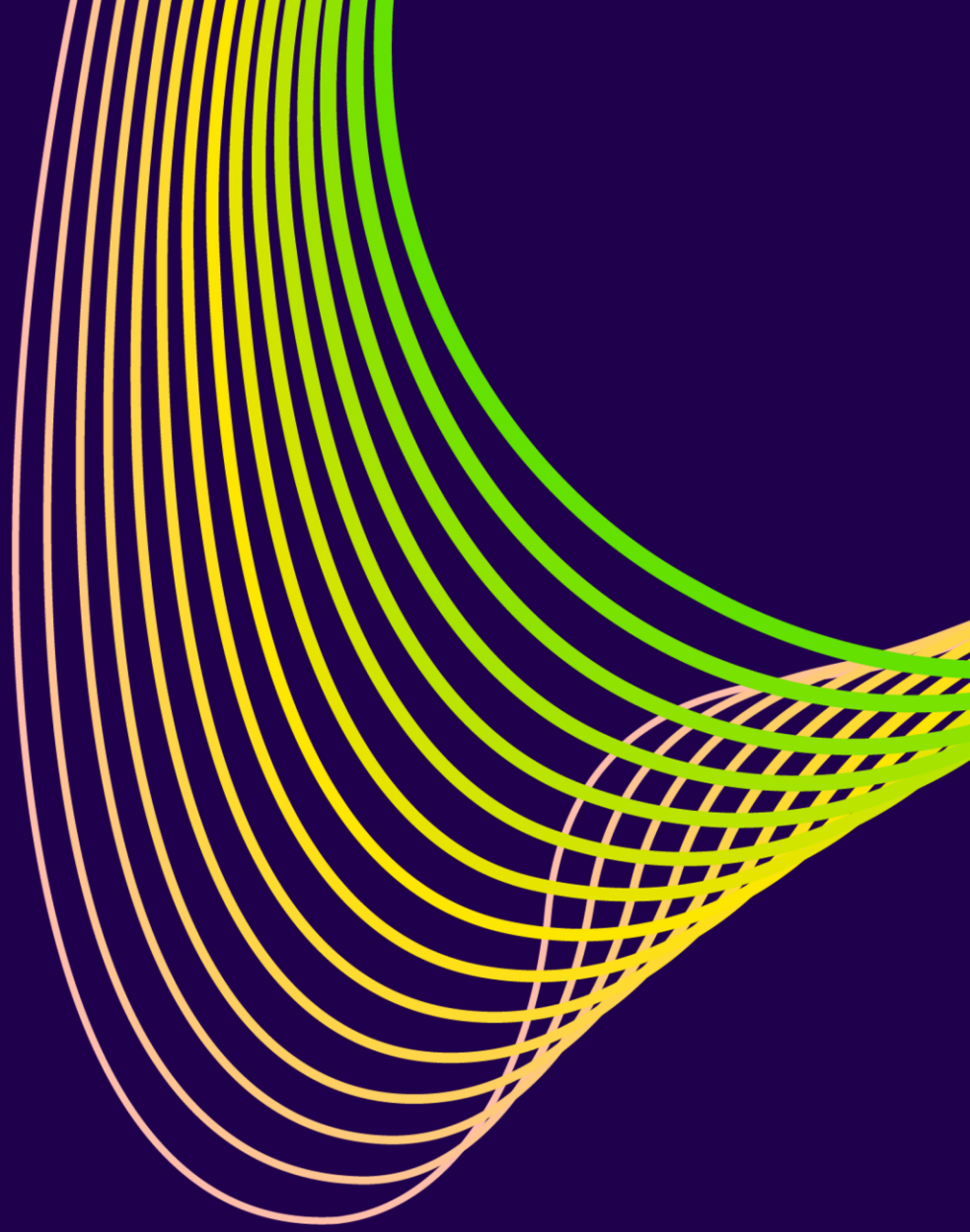
Leopoldstraße 146
80804 München

Wiesenhüttenplatz 25
60329 Frankfurt am Main

Theodor-Heuss-Str. 30
70174 Stuttgart

Green IT as a driver for future viability

Backup



Something has clearly gotton out of hand

Not to be used

Balanced

Independent regeneration of nature.

Unbalanced

Regeneration is no longer possible without human intervention due to air pollution.

The Objective

- Returning nature to self-regeneration through the fastest possible transition to the post-fossil age.
- Reduction of energy consumption, conversion to renewable energies in the economy, mobility and transportation.
- Education & awareness-raising, nature conservation & renaturation, biodiversity, sustainable agriculture & forestry, pollution prevention.

New IT technologies drive energy demand through mass use

Google/Cloud

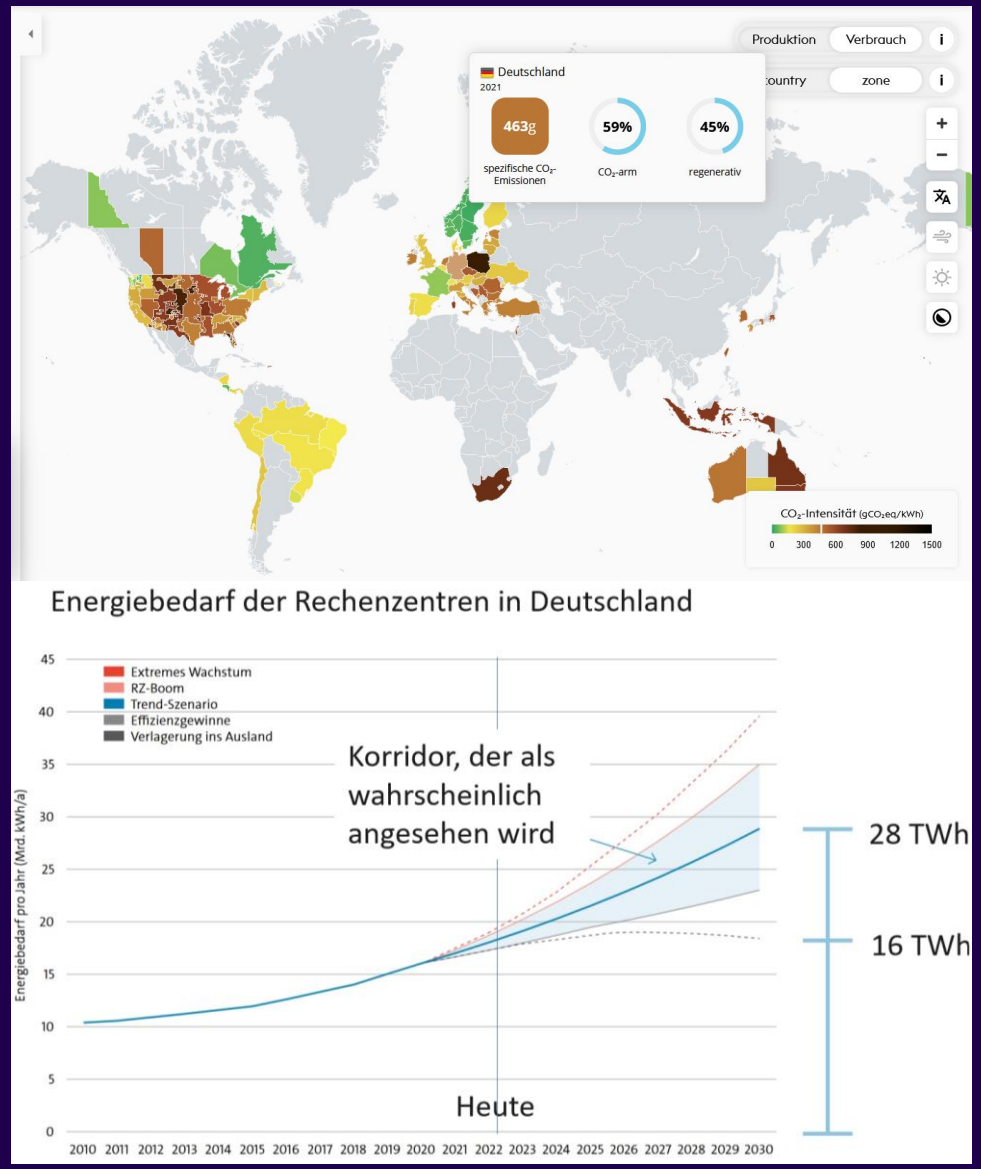
- Information search service on the Internet with functions such as Google Search, Gmail, Google Maps.
- Approx. 2.5 billion users per month (1/3 of the world's population).
- An estimated 5.6 billion search queries per day.

Bitcoin/Blockchain

- Cryptocurrency based on blockchain technology.
- Potential to revolutionize the financial system through speed, security & independence from traditional banks.
- Several million users carrying out thousands of transactions per day.

ChatGPT/AI

- AI model developed by OpenAI, based on the Generative Pretrained Transformer method.
- Around 100 million active users.
- Around 10 million queries per day.



Energy Consumption – Awareness

Data centers ~~hunger~~ for energy

Not to be used

- ~16 TWh power consumption in the data centers in Germany (2022) = 16.000.000.000 kWh
- ~463 g CO_2 pro kWh [1]

7.408.000 tons CO_2 only by german data centers.

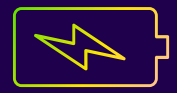
Sources: Own illustration from Electricitymaps (2023); Hintemann et al. (2021)

Covering electricity needs from sustainable sources

Not to be used

- Global energy consumption in 2019 amounted to around 170,000 terawatt hours (TWh).
- To generate this amount of energy with solar power, we would need around 680,000 km² of space, which is roughly equivalent to the area of Texas (< 1% of the Earth's land area).

IT has to become sustainable



Energy savings through more efficient hardware and technologies such as virtualization and cloud computing.

Conserving resources by recycling, reusing and extending the service life of IT devices.

Sustainable procurement of environmentally friendly IT products.

Reducing paper consumption through digital processes and working environments.

Concrete Approaches

Potential for optimization has been identified - MMIGIT leads the way to implementation

Cloud/Google

- **Data-Center optimization:** efficient cooling, reducing PUE
- **Server & Software – Green Code:** virtual server, searching algorithm
- **Reducing data transfer - Green Data Management:** Data Compressing, Content Delivery Networks
- **User Behavior:** Create awareness, use "green" search engines

Blockchain/Bitcoin

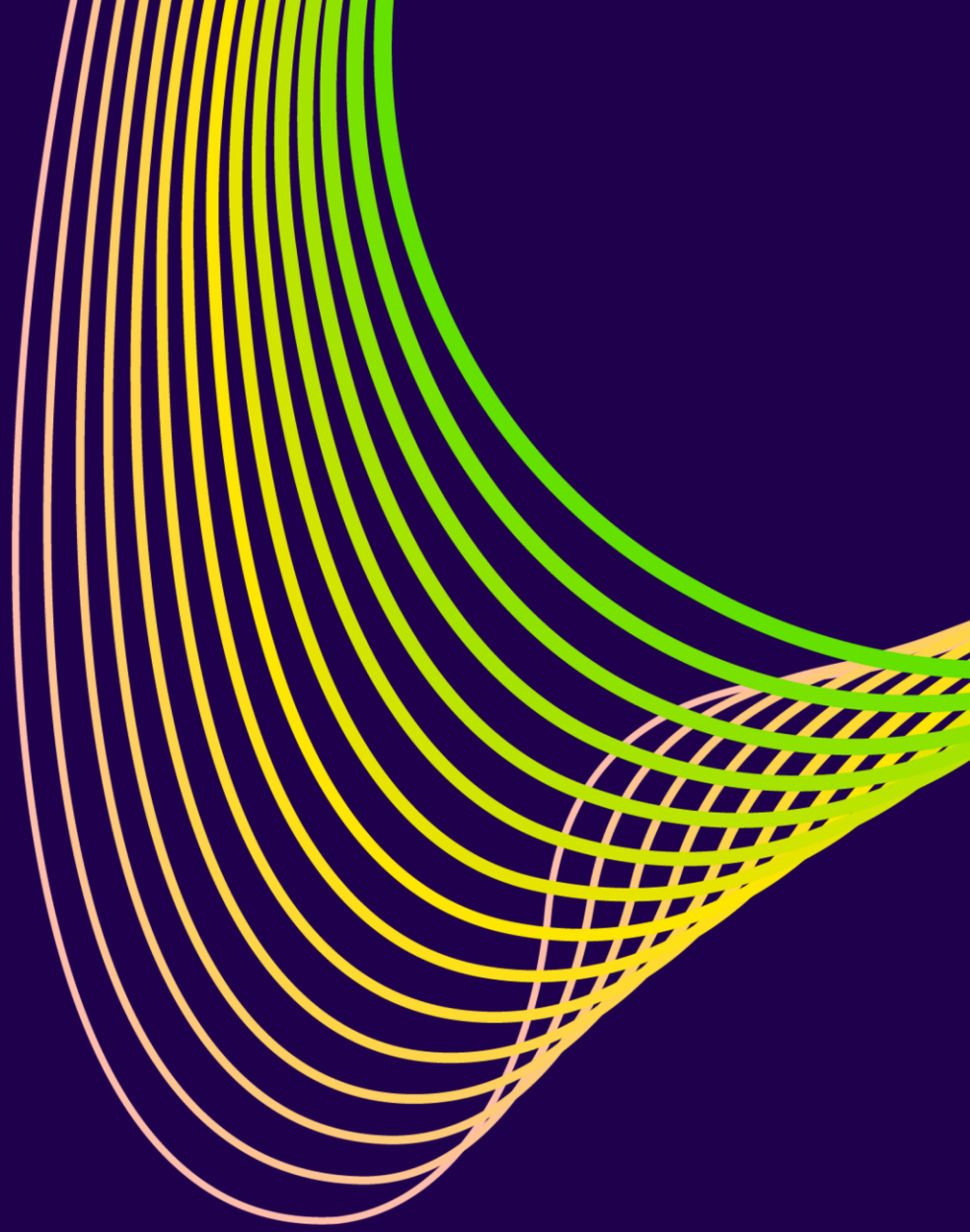
- **Alternative Procedures:** Proof-of-Stake (PoS) instead of Proof-of-Work (PoW)
- **Hardware optimization:** ASIC instead of von Neumann-Architectures
- **Reducing Requirements for Data Storing** (Green Data Management)
- **Awareness und Traing** Training of Miners, Green Coins

AI/ChatGPT

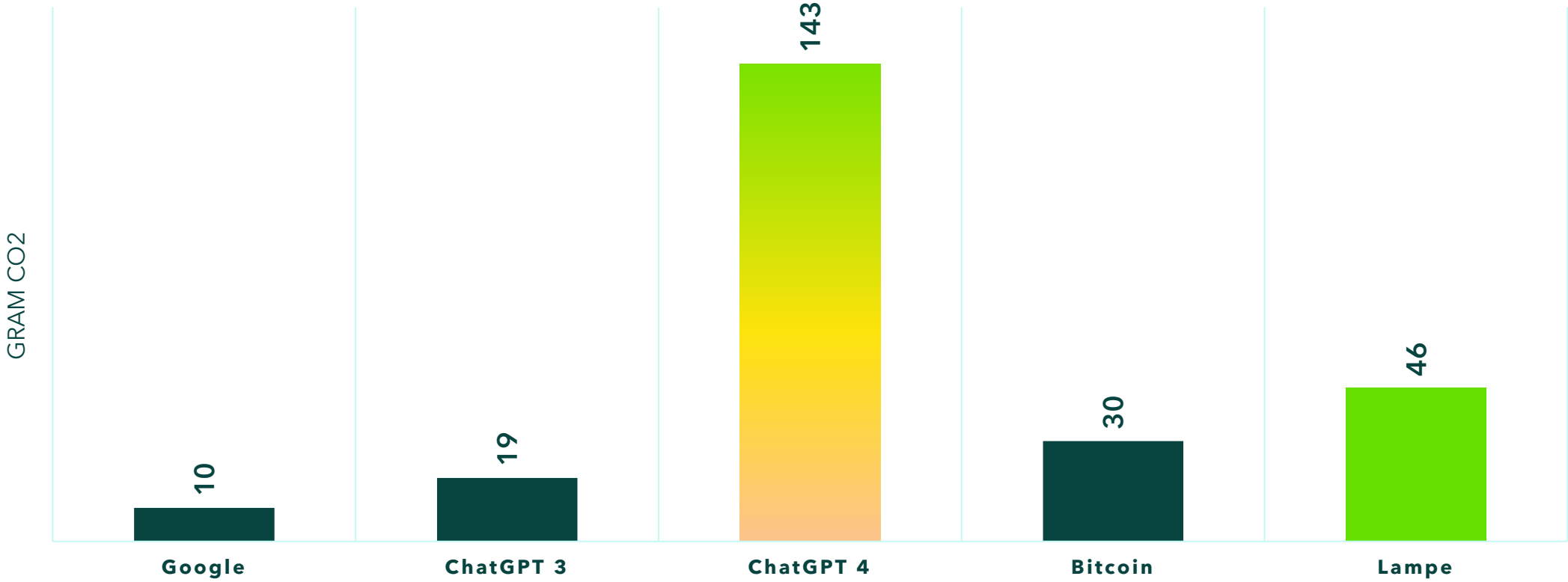
- **Model-Efficiency:** Optimization of decision trees.
- **Quantization :** Reduce calculation accuracy arithmetic.
- **Distillation :** Smaller Models
- **Efficient Training Methods**
- **Transfer-Learning:** Use of existing models.
- **Early Stopping:** Domain-specific depth of training
- **Use of specific chips: (ASICs)**

Green IT as a driver for future viability

Red AI vs Green AI



Overview Energy Consumption



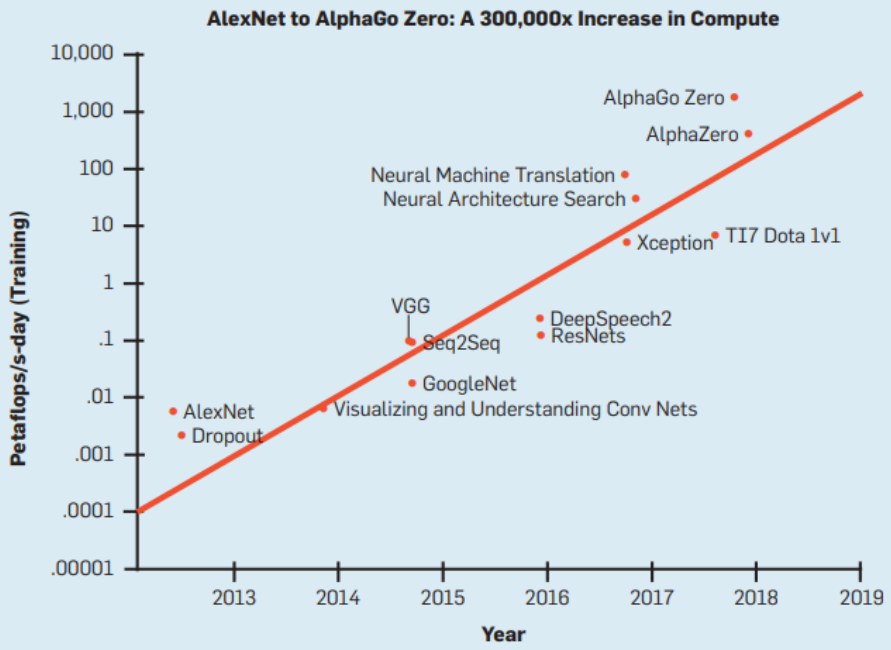
Annotation: Suppose a person is sitting at a desk and working intensively in the light of a 50-watt lamp. Within an hour, he makes 50 requests on Google, ChatGPT3 or ChatGPT4. At the end, he pays for a small purchase worth EUR 1 from his Bitcoin wallet.

Red AI vs Green AI

Red AI leads to a surprisingly large carbon footprint, and makes it difficult for academics, students, and researchers to engage in deep learning research.

The computational costs of state-of-the art AI research has increased 300,000x in recent years. This trend, denoted Red AI, stems from the AI community's focus on accuracy while paying attention to efficiency.

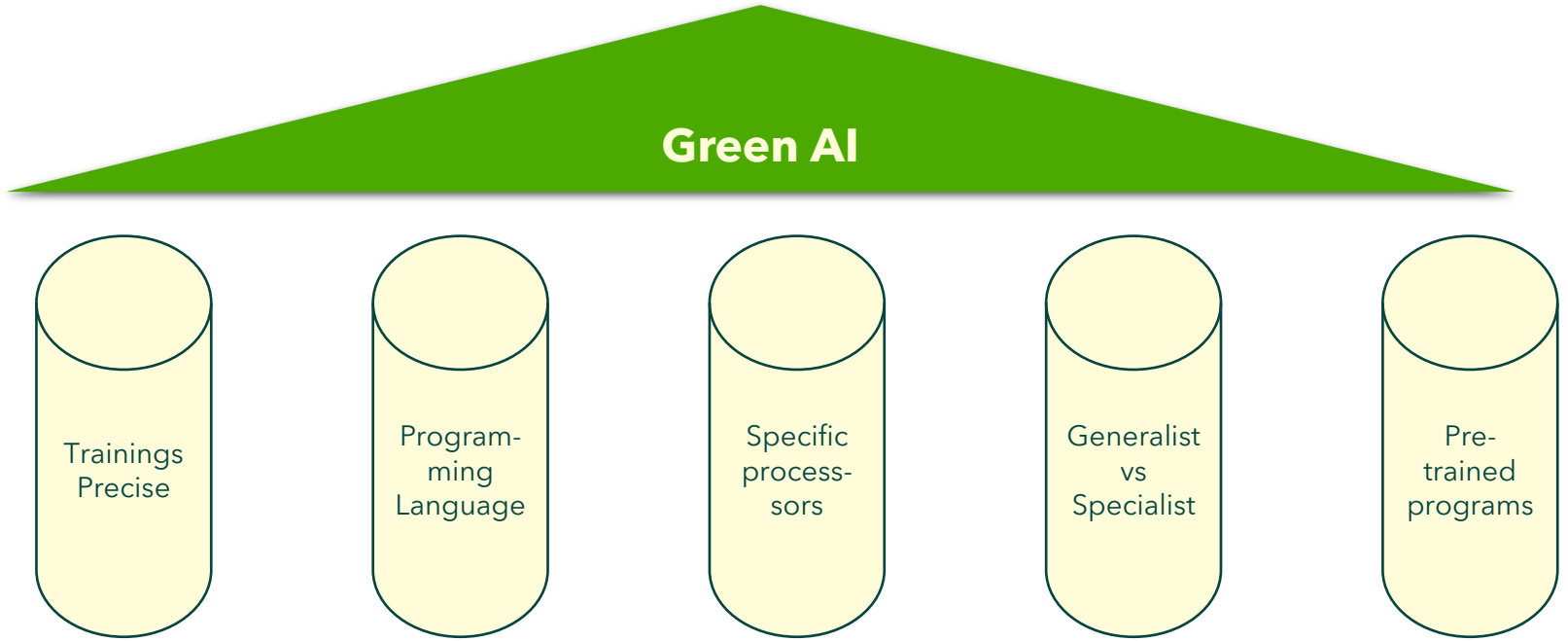
Figure 1. The amount of compute used to train deep learning models has increased 300,000x in six years. Figure taken from Amodei et al.[2]



Schmid, Thomas; Hildesheim, Wolfgang; Holoyad, Taras; Schumacher, Kinga, 2021. The AI Methods, Capabilities and Criticality Grid. A Three-Dimensional Classification Scheme for Artificial Intelligence Applications. KI - Künstliche Intelligenz 35 (3), S. 425–440 DOI: 10.1007/s13218-021-00736-4

Red AI vs Green AI

The term **Green AI** refers to AI research that yields novel results while taking into account the computational cost, encouraging a reduction in resources spent. Whereas Red AI has resulted in rapidly escalating computational (and thus carbon) costs, **Green AI** promotes approaches that have favorable performance/efficiency trade-offs.



IT is part of the problem and part of the solution.

It depends on the people what role IT will play in the future - solution or problem.