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

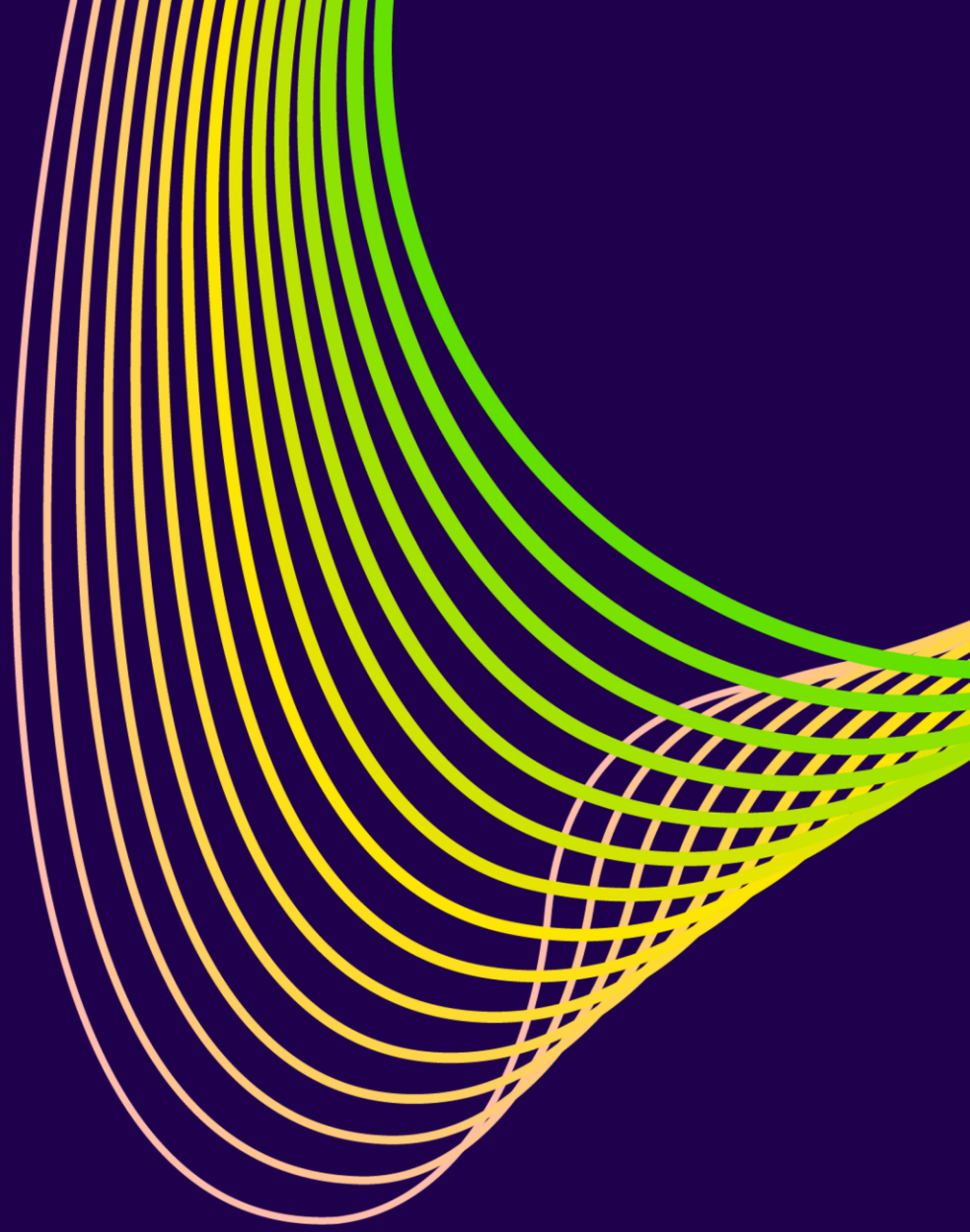
Expert GreenIT

Eric.Jochum@metafinanz.de

+ 49 89 3605316205

Green IT as a driver for future viability

Green IT – New thinking and acting



metafinanz

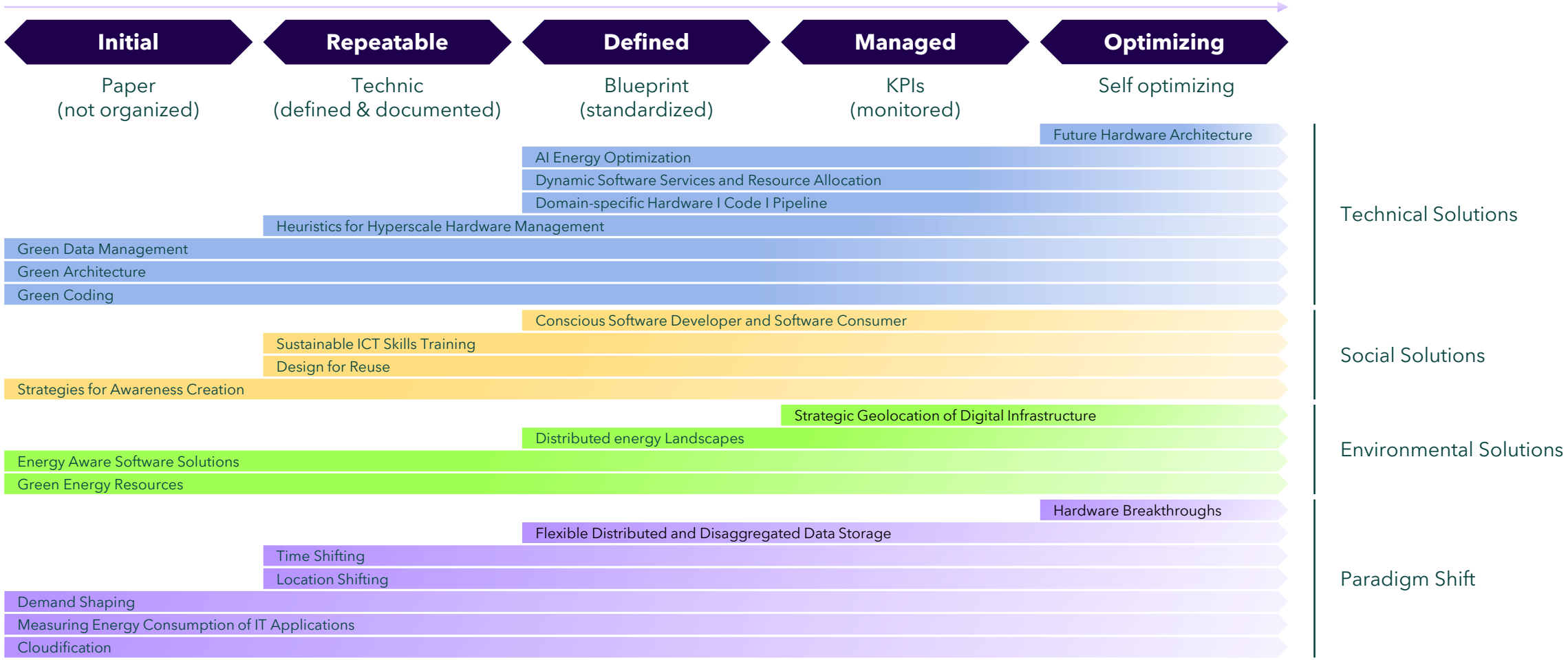
Green IT – New thinking and acting

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**!

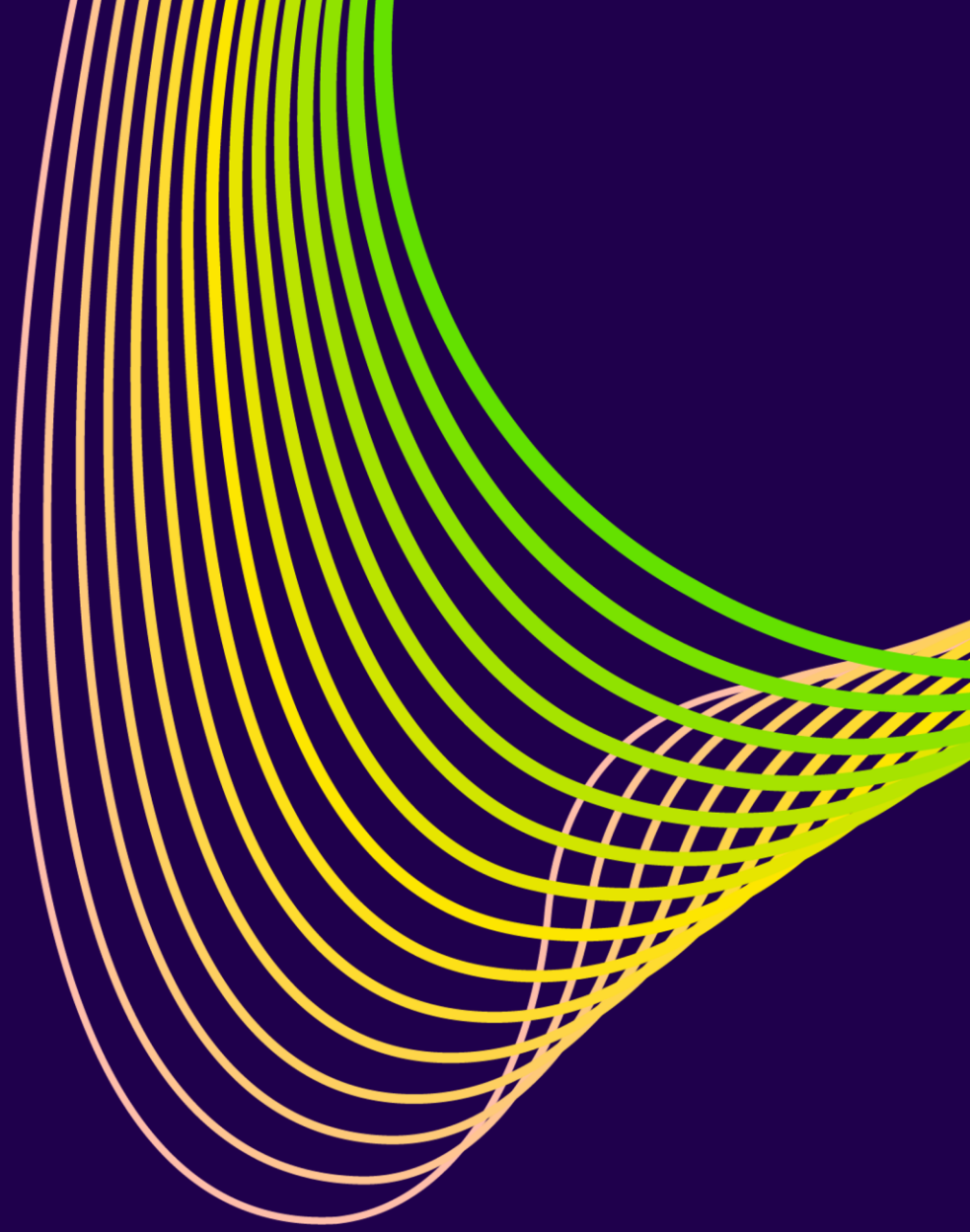
MMIGIT (Maturity Model Integrated Green IT) – Solutions Landscape Green IT



metafinanz

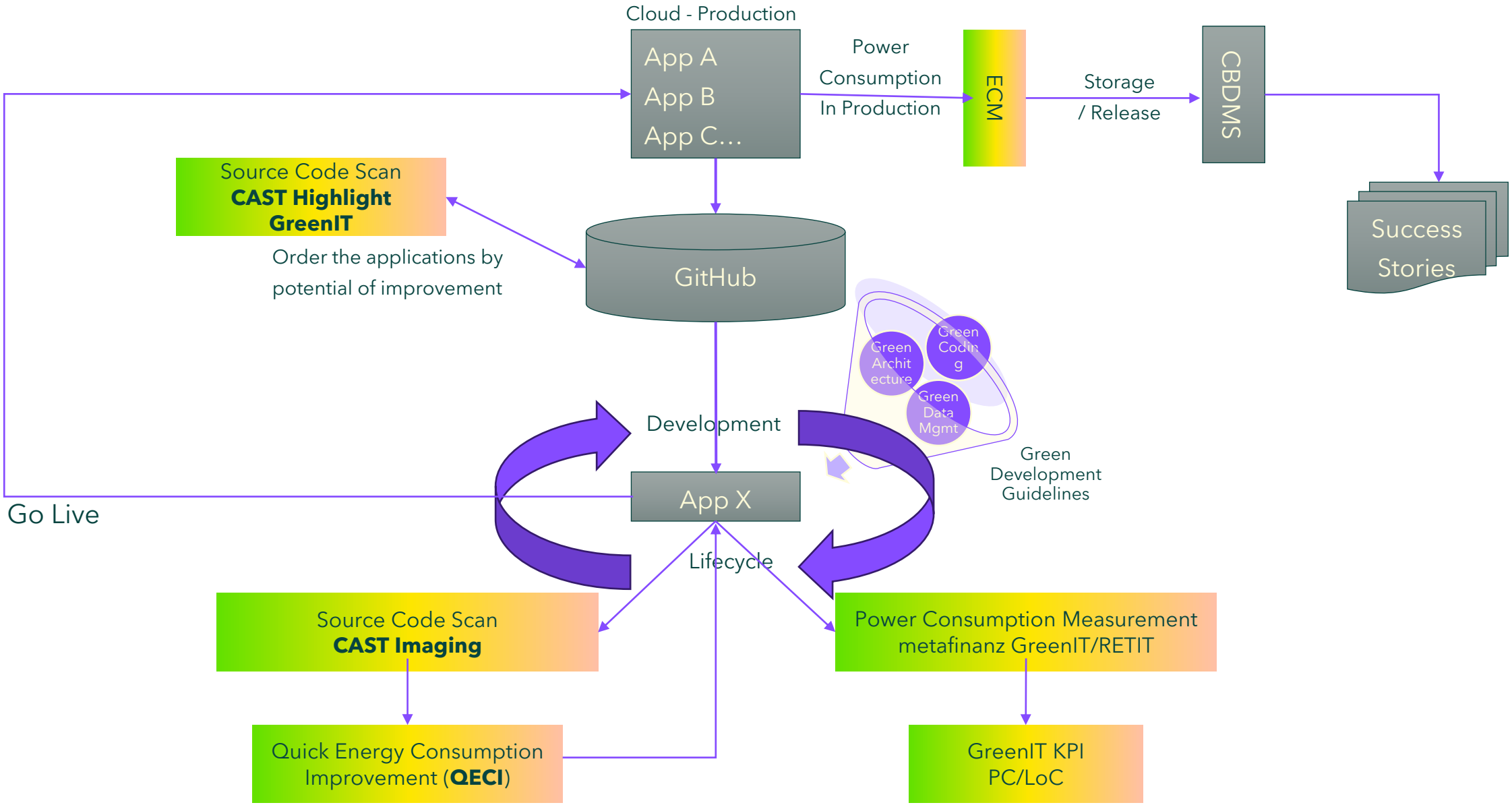
Green IT as a driver for future viability

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 specialist LLMs which are aware of the context

Show Case Using Generative AI in Code Optimization

Scan Report & QECI

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

RESULTS	Original		QECI			
API Call	**CPU [s]**	Energy [mWh]	**CPU [s]**	Energy [mWh]	**Savings [%]**	Absolute [mWh]
api_v1_private_catalog__id_	3,34	61,23	2,71	49,68	18,86	11,55
api_v1_private_catalog	2,99	54,82	2,53	46,38	15,38	8,43
api_v1_category_product__ProductId_	2,11	38,68	1,66	30,43	21,33	8,25
api_v1_category__id__manufacturer_	2,09	38,32	1,68	30,80	19,62	7,52
api_v1_category__friendlyUrl_	2,08	38,13	1,66	30,43	20,19	7,70
api_v1_category	1,39	25,48	1,17	21,45	15,83	4,03

https://github.com/marton-eifert/shopizer_showcase/compare/3.2.5-actual...3.2.5-actual-QECI-fix--ATTEMPT-1-only-avoid-nested-loops-1-of-6-a#diff-1e8015b8c0587993f78ad512df20fc9e4c33d1c860a4b73e46ae804f7f7ee28d

Results from QECI

37 ...p/src/main/java/com/salesmanager/shop/mapper/inventory/PersistableProductPriceMapper.java

```
@@ -8,6 +8,8 @@
 import java.util.List;
 import java.util.Optional;
 import java.util.Set;
+import java.util.HashMap;
+import java.util.Map;

 import org.apache.commons.collections4.CollectionUtils;
 import org.jsoup.helper.Validate;
@@ -156,6 +158,40 @@ private Set<ProductPriceDescription> getProductPriceDescriptions(ProductPrice pr
                         return Collections.emptySet();
                 }
                 Set<ProductPriceDescription> descs = new HashSet<ProductPriceDescription>();
+
+                /**********************************
+                 * CAST-Finding START #1 (2024-02-02 12:30:57.420270):
+                 * TITLE: Avoid nested loops
+                 * DESCRIPTION: This rule finds all loops containing nested loops.  Nested loops can be replaced by redesigning data with hashmap, or in some contexts, by using specialized high level API...  With hashmap: The literature abounds with documentation to reduce complexity of nested loops by using hashmap.  The principle is the following : having two sets of data, and two nested loops iterating over them. The complexity of a such algorithm is O(n^2). We can replace that by this process : - create an intermediate hashmap summarizing the non-null interaction between elements of both data set. This is a O(n) operation. - execute a loop over one of the data set, inside which the hash indexation to interact with the other data set is used. This is a O(n) operation.  two O(n) algorithms chained are always more efficient than a single O(n^2) algorithm.  Note : if the interaction between the two data sets is a full matrice, the optimization will not work because the O(n^2) complexity will be transferred in the hashmap creation. But it is not the main situation.  Didactic example in Perl technology: both functions do the same job. But the one using hashmap is the most efficient.  my $a = 10000; my $b = 10000; sub withNestedLoops() {     my $i=0;     my $res;     while ($i < $a) {         print STDERR "$i\n";         my $j=0;         while ($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
+                 **********************************/
```

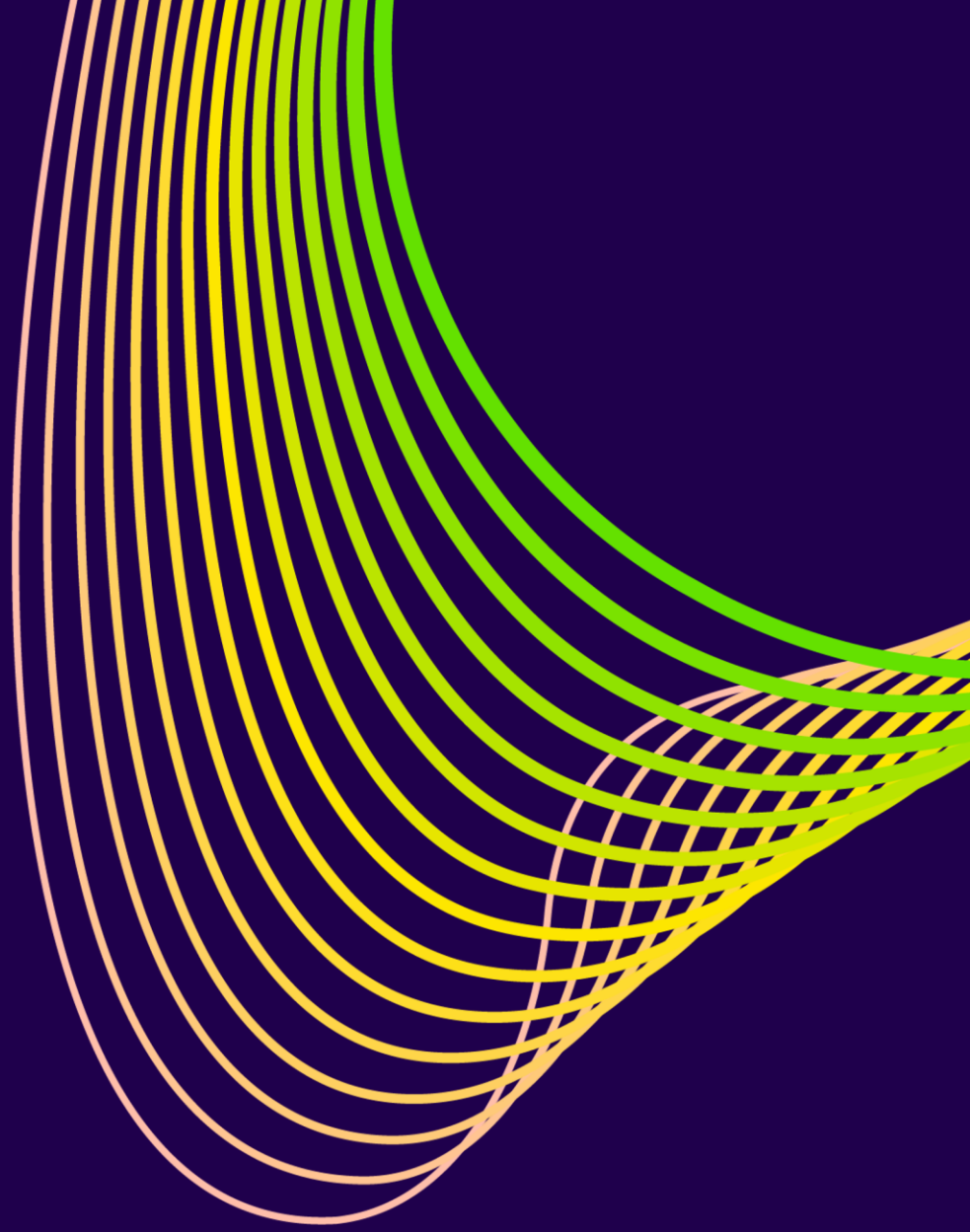
https://github.com/marton-eifert/shopizer_showcase/compare/3.2.5-actual...3.2.5-actual-QECI-fix--ATTEMPT-1-only-avoid-nested-loops-1-of-6-a#diff-1e8015b8c0587993f78ad512df20fc9e4c33d1c860a4b73e46ae804f7f7ee28d

Results from QECI

```
                                                                                 168 +                 *********************************/
                                                                                 169 +
                                                                                 170 +                 // QECI Fix
                                                                                 171 +
                                                                                 172 +                 // Create a HashMap to store the descriptions by their ID for quick lookup
                                                                                 173 +                 Map<Long, ProductPriceDescription> priceDescMap = new HashMap<>();
                                                                                 174 +                 for (ProductPriceDescription d : price.getDescriptions()) {
                                                                                 175 +                     priceDescMap.put(d.getId(), d);
                                                                                 176 +                 }
                                                                                 177 +
                                                                                 178 +                 for (com.salesmanager.shop.model.catalog.product.ProductPriceDescription desc :
                                                                                       descriptions) {
                                                                                 179 +                     // Check if the description ID is positive and exists in the priceDescMap
                                                                                 180 +                     if (isPositive(desc.getId()) && priceDescMap.containsKey(desc.getId())) {
                                                                                 181 +                         // Use the existing description from priceDescMap
                                                                                 182 +                         ProductPriceDescription existingDescription =
                                                                                       priceDescMap.get(desc.getId());
                                                                                 183 +                         existingDescription.setId(desc.getId());
                                                                                 184 +                         descs.add(existingDescription);
                                                                                 185 +                     } else {
                                                                                 186 +                         // Create a new description if not found in the priceDescMap
                                                                                 187 +                         ProductPriceDescription description = getDescription(desc);
                                                                                 188 +                         description.setProductPrice(price);
                                                                                 189 +                         descs.add(description);
                                                                                 190 +                     }
                                                                                 191 +                 }
                                                                                 192 +
                                                                                 193 +
                                                                                 194 +                 /*
159                  for (com.salesmanager.shop.model.catalog.product.ProductPriceDescription desc :   195                  for (com.salesmanager.shop.model.catalog.product.ProductPriceDescription desc :
       descriptions) {                                                                 descriptions) {
160                      ProductPriceDescription description = null;                 196                      ProductPriceDescription description = null;
161                      if (CollectionUtils.isNotEmpty(price.getDescriptions())) {  197                      if (CollectionUtils.isNotEmpty(price.getDescriptions())) {
@@ -169,6 +205,7 @@ private Set<ProductPriceDescription> getProductPriceDescriptions(ProductPrice pr
169                      description.setProductPrice(price);                         205                      description.setProductPrice(price);
170                      descs.add(description);                                     206                      descs.add(description);
171                  }                                                               207                  }
                                                                                 208 +                 */
172                  return descs;                                                   209                  return descs;
173              }                                                                   210              }
174                                                                                  211
```

Green IT as a driver for future viability

Energy Savings with AI?



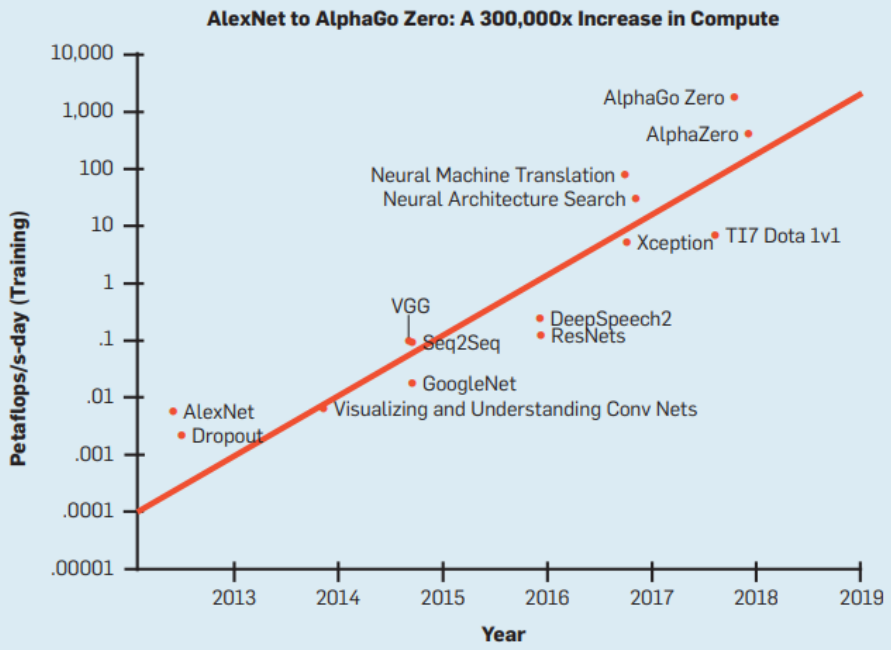
metafinanz

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.

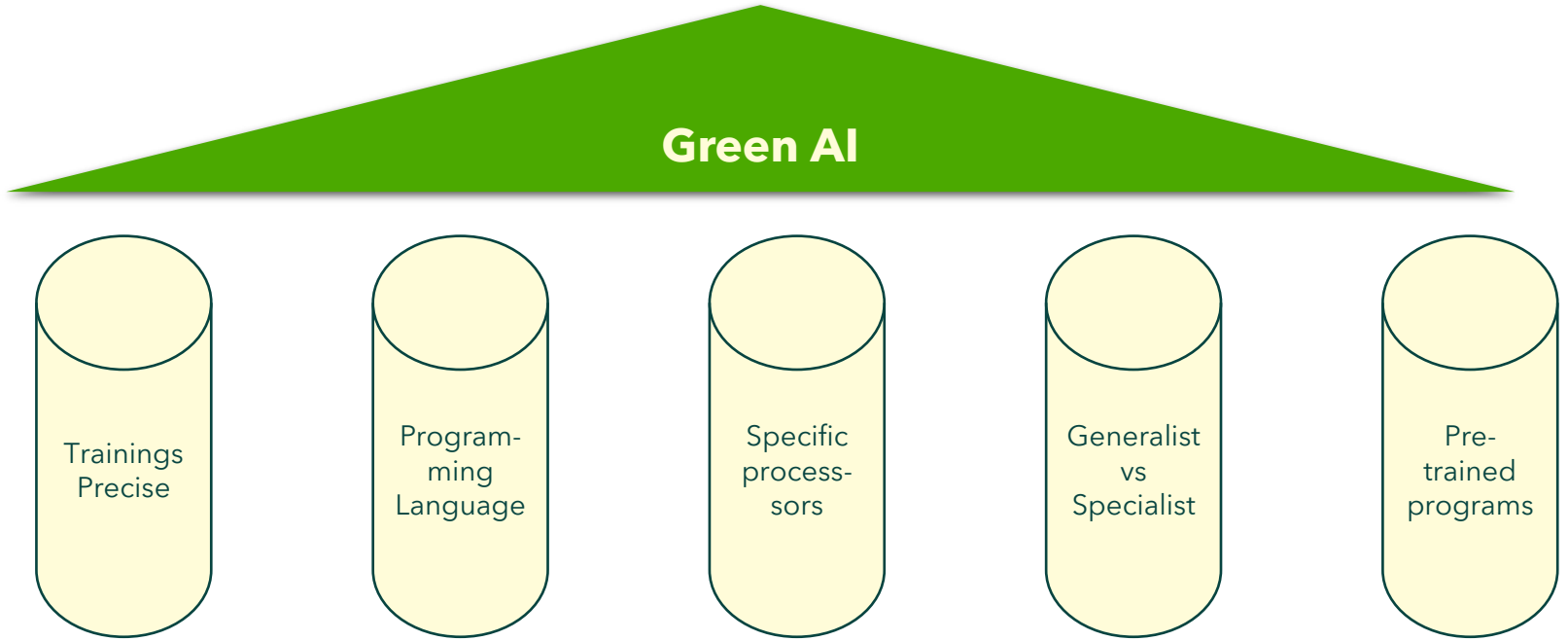
Figure 1. The amount of compute used to train deep learning models has increased 300,000x in six years. Figure taken from Amodei et al.[2]



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.



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