RISC Software GmbH
www.risc-software.at

INSIGHTS²

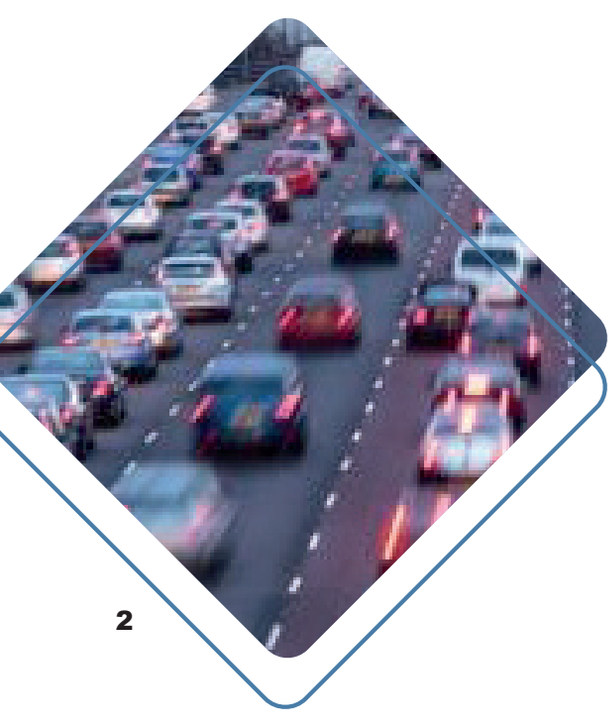
The technical magazine of RISC Software GmbH
on current research and development topics

Agile Software Development

Software Reengineering

Data Science and Prescriptive Analytics

Intelligent Transport Systems



CONTENT

Dear Reader,

I am pleased to present you the second issue of our collected technical papers. Once again, we would like to provide you with general introductions to topics and more in-depth insights into some of our areas of work and expertise. In this magazine, you will find the following thematic blocks:

- **Agile Software Development**: Agile process models focus primarily on the organizational side. Agile workflows and test-driven development used correctly provide rapid feedback and fast response cycles.
- **Data Science and Prescriptive Analytics:** The fusion of the two worlds of "classical optimization" and "learning systems" shows how models and data are combined to create solutions and how new insights are gained from them.
- **Software-Reengineering** – Get away from legacy systems!
- **Intelligent Transport Systems** – We need a mobility revolution ... an initiative by our (young) employees* for sustainable and self-determining mobility.

Much of what you read here is know-how that has been built up over many years and developed in research projects, coupled with the latest scientific methods that have been successfully implemented in customer projects. What sets us apart is our ability to work with our customers to solve their problems in such a way that, beyond digitization, we gain a decisive competitive advantage in the long term and often build up a good partnership over many years. The broad term "digital transformation" is on everyone's lips, but it must be carried out individually by each organization. We can support you on this path with our expertise and know-how in a variety of ways, but ultimately you have to take the path yourself.

A major topic currently occupying many industrial companies is re-globalization: This means that supply chains are changing in the direction of stability and redundancy and that production is being brought back to Europe. In combination with the omnipresent shortage of skilled workers, this is an enormous challenge that can only be met with increased automation ("digitization"). No matter where you stand - whether you want to digitize complex processes, bring old inventory software up to date, use your manufacturing data for forecasting or anywhere else - our team of experts will support you in the implementation of your R&D projects with diverse expertise!

Enjoy reading!

Wolfgang Freiseisen
CEO Software GmbH

Methods and tools for data preparation in the big data area

– DI Paul Heinzlreiter
Senior Data Engineer in the Unit Logistics Informatics

In recent years, the role of big data in numerous economic sectors such as the manufacturing industry, logistics or trade has become increasingly important. Using a wide variety of sensor systems, large amounts of data are collected that can subsequently be used to optimize machines or business processes. Methods from the fields of artificial intelligence, machine learning or statistics are often used here.

However, all these methods require a larger quantity of high-quality and valid data as a basis. In this context, data engineering is used to collect the raw data, cleanse it and merge it into an integrated database. While a previous article (the magazine INSIGHT #1) highlighted the general role and goals of data engineering, this article will focus on methods and proven tools as well as provide an exemplary insight into the algorithmic implementation of data engineering tasks.

Data stream and batch processing

If, for example, industrial sensor data is collected over time, large amounts of data do not accumulate per unit of time (e.g. every few seconds), but over months and years the stored data volumes often increase into the terabyte range. If data in this order of magnitude is to be processed, this can essentially be done using two different paradigms, described here for converting the data type of a table column:

- Batch Processing:

Here, all rows in a table are processed in parallel to convert one column.

- Data stream processing (Data Streaming):

Here, the rows of the table are read sequentially and the column conversion is performed per row.

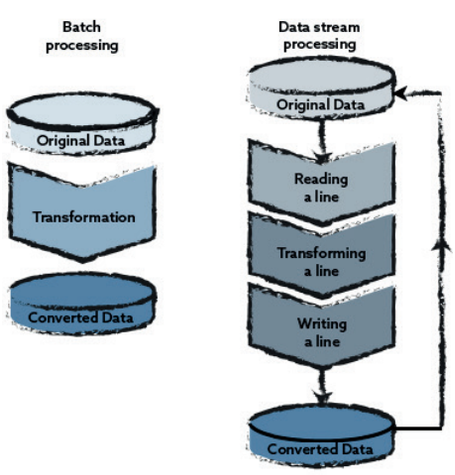


Figure 1: Comparison of batch and data stream processing

The main difference between the two data processing approaches is that in data streaming, the necessary data transformations - such as converting data fields to other data types - are performed directly on the currently supplied data set, whereas in batch processing, the data is first collected, and subsequently the data transformations are performed on the entirety of the data. Which approach is chosen depends on the data transformation requirements:

- If the transformation can be performed locally on the currently queried or received data, the use of the data streaming approach is often preferable, since it is usually a simple and local operation that can also be processed more quickly due to the smaller input data. A typical application of Data Streaming is the direct conversion of sensor data arriving distributed over time, as these can then be converted and stored individually.

- However, if the data transformation requires input data from the entire data already stored or if all data is already available, a batch approach is more suitable. Parallel processing of the data is also often easier to implement here, as this is directly supported by frameworks such as Apache Hadoop (through the Map-Reduce approach) or Apache Spark.

In general, the data obtained should be stored once in raw format in order to not lose any data that could still be needed as a basis for future analyses. Further processing of data stored in this way can then be done by batch processing or data streaming. In the second case, a data stream is generated again from the stored data by continuous reading. Conversely, a data stream can be stored continuously and thus serve as a starting point for batch processing.

Data stream processing: Apache NiFi

NiFi represents a tool for data stream processing, which makes it possible to connect data transformations in a graphical, web-based user interface to form a continuous data pipeline through which the source data flows and is transformed step by step. The strengths of Apache NiFi lie in the wide range of modules already available, which enable, for example, the reading and storing of numerous data formats. Due to the open source character of NiFi and the object-oriented structure of its modules, it is easy to develop your own modules and integrate them into data pipelines. Furthermore, NiFi also addresses issues such as the automated handling of different processing speeds of the modules.

Batch processing: Apache Hadoop

Hadoop is a software framework based on the fundamental principle of parallel data processing in a cluster environment. Within the distributed processing, each cluster computer takes over the processing of the data locally available there, which above all saves communication effort during the calculations. Hadoop distinguishes here between controller and responder services in the cluster, whereby the responder services take over the processing of the locally available data, while the controller services are responsible for the coordination of the cluster. Parts of the algorithms implemented in Hadoop were developed by Google and the concepts published in research papers, such as the Google File System, Map-Reduce and Google Bigtable. At Google, these solutions are used to operate the global search infrastructure, while the Hadoop project is an open-source implementation of these concepts.

At its core, a Hadoop system consists of a usually Linux-based cluster running the Hadoop File System (HDFS) and YARN as an implementation of the Map-Reduce algorithm. A Hadoop cluster with the HDFS and YARN services provides a solid technological basis for a wide variety of Big Data services such as BigTable databases like HBase - see below - or graph databases like JanusGraph, for example.

Hadoop File System (HDFS)

HDFS is an open source implementation of the Google Filesystem. Like other Hadoop subsystems, it consists of controller and responder components, in the case of HDFS Namenodes (controllers) and Datanodes (responders). While a Namenode stores where on the cluster the data for individual files is stored, the Datanodes handle the storage of the data blocks. Basically, HDFS is optimized for large files, the block size for storage is usually 128 megabytes. On the one hand, a file can consist of many individual blocks, on the other hand, the data blocks are replicated across multiple cluster nodes for redundancy and performance reasons. The access semantics of HDFS are different from the usual Posix semantics, since only data can be appended to HDFS files, but they cannot be edited. To create a new version of a file, it must be replaced. This can be done very effectively even for large files using the Map-Reduce algorithm described below.

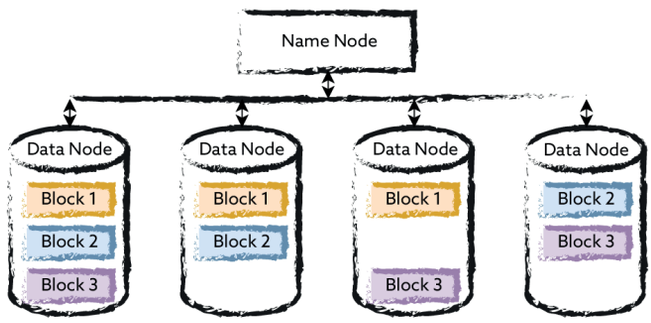


Figure 2: Block replication in the Hadoop file system (HDFS).

In the context of a Hadoop system, text files stored in CSV format, for example, can now be processed with Map-Reduce jobs, with the distribution of sub-jobs across the cluster based on the distribution of the HDFS file blocks being handled automatically by the Hadoop framework. In addition to plain text files, structured binary data such as ORC, Parquet or AVRO formats can also be processed directly by Hadoop. In addition, specific splitter classes for Map-Reduce can be implemented for new formats. Furthermore, as part of a Map-Reduce algorithm, it is possible without problems to perform only one map stage, for example, to add new columns to a CSV file.

Map-Reduce-Framework (YARN)

Based on the data distribution in HDFS shown above, a data-parallel batch job can now be executed by the YARN service, with each responder node processing the locally available data blocks. Conceptually, the execution follows the Map-Reduce algorithm. A classic application example for the Map-Reduce algorithm is the counting of words in text documents. Here, the map step emits a set of pairs of the form (word, number of occurrences in the line) per line. In the Shuffle step, these pairs of values are grouped according to the words, since they represent the key. In the final Reduce step, the word frequencies per word are summed. An exemplary execution could run as follows:

- The input text is divided into individual text lines. (Splitting)
- The map step, which is executed in parallel for each line individually, creates a pair of the word and the number 1 for each word in the line. (Mapping)
- The pairs are sorted by the words and combined into one list per word. (Shuffling)
- For each word, the number in the total text is determined by adding up the numbers. (Reducing)

While the Map step and the Reduce step must each be programmed out, the global Shuffle step is automatically taken over by the Map-Reduce framework. In practice, the implementation of the Map and Reduce steps requires, for example, the object-oriented overwriting of one Map and one Reduce method each, whose interfaces are already specified. This allows the focus to be placed on the transformation of a pair of values, while the framework subsequently takes care of the scaled execution on the cluster.

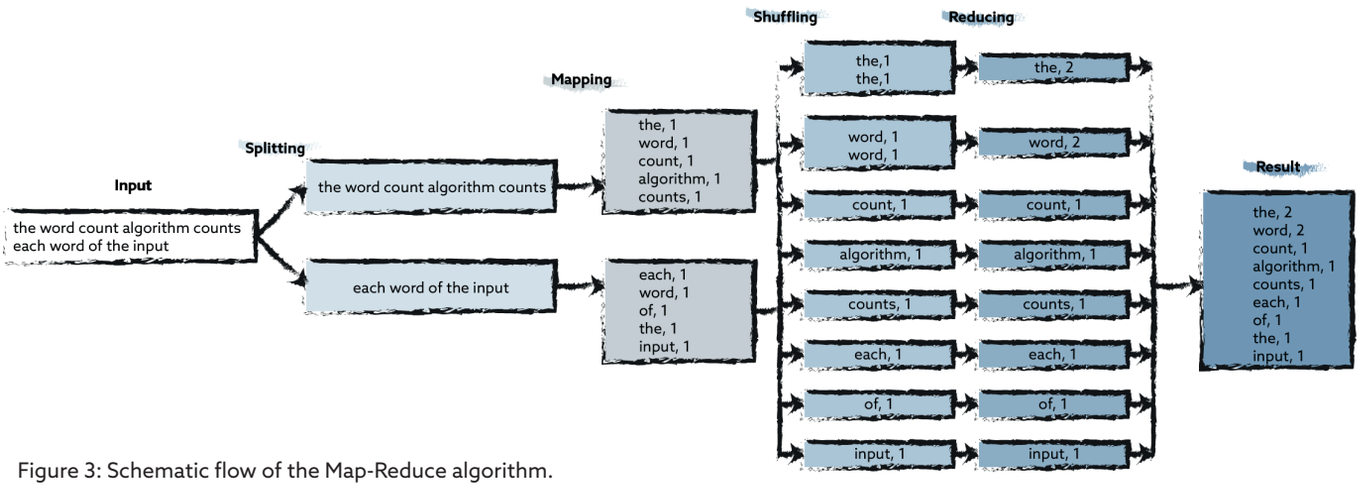


Figure 3: Schematic flow of the Map-Reduce algorithm.

Batch and data stream processing: Apache Spark

Spark is a flexible data processing layer that can be built on top of various infrastructures, such as Hadoop, and can be used for various data engineering and data science tasks. As a general data processing framework, Spark can perform data preprocessing tasks as well as machine learning tasks.

For example, Apache Spark can be installed on an existing Hadoop cluster and directly access the data stored there and process it in parallel. One approach to this is the Map-Reduce algorithm mentioned above, although Spark can also apply other flexible methods such as data filtering. Spark stores intermediate results as resilient distributed datasets (RDDs) in main memory, which avoids slow repetitive disk accesses - as is often the case with classic databases.

Key features of Spark include:

- Parallel batch processing, for example using the Map-Reduce algorithm.
- Support of SQL queries on arbitrary (e.g. in HDFS) stored data. To do this, you only need to interactively create a table that defines the data schema to be used and references the underlying data.
- Based on sequential processing of multiple RDDs, data stream processing can be performed.

Just like an underlying Hadoop cluster, a Spark installation can be made fit for processing larger amounts of data by a simple hardware upgrade.

Choosing the right tools for big data engineering

As can be seen from the application examples shown below (data transfer from a CSV file to an SQL database), different paths often lead to the same goal in the field of data engineering. Which methods should be used often depends on the specific requirements of the customer as well as their system environment:

- For example, if a Hadoop cluster is already in use or planned, it can already be integrated when designing a solution.
- Public cloud offerings such as Amazon AWS, for example, in turn offer alternatives to the open source solutions described above, which primarily simplify the operation of the solution, but can also lead to vendor lock-in.
- Other criteria for a technology decision are requirements for scalability and the planned integration of additional tools.
- Last but not least, open source solutions often offer cost advantages, as there are no licensing costs even for highly scalable solutions.

Application example: Processing of industrial sensor and log data

As part of the VPA4.0 research project, a data pipeline was set up for the pre-processing of production sensor data. This represents a good example of linking streaming and batch processing. Apache NiFi was used as a streaming solution to transmit the data directly from the project partner over the Internet in encrypted form before storing it locally on the Hadoop cluster. Further data processing was then performed using Spark in parallel on the Hadoop cluster and included the following steps:

- Unpacking the received data archives and removing unneeded files
- Preparation and storage of data as CSV files in HDFS
- Creating virtual tables based on CSV files enables further processing with SQL
- Data filtering and storage in optimized Parquet format for interactive SQL queries

Application example: Cleaning sensor data and storing it in an SQL database

This example includes sensor data collected on a heat engine. In the following example, negative values can be seen in the column power_dynamo, which were caused by a measurement inaccuracy. Rows with such values should now be filtered out as erroneous and

the cleaned data stored in a database.

```
timestamp;temperature_heater;temperature_boiler;pressure_boiler;rpm;power_dynamo;power_heating;valve_aperture;water_level
2019-06-14T12:43:00;39.404514;267.222229;1292.380981;0.000000;-0.019452;446.973846;0.0;21.00
2019-06-14T12:43:01;38.194447;268.361115;1292.380981;0.000000;-0.019452;446.973846;0.0;21.00
2019-06-14T12:43:02;38.194447;267.222229;1292.326538;0.000000;-0.019452;446.973846;0.0;21.00
2019-06-14T12:43:03;38.194447;267.222229;1292.380981;0.000000;-0.019452;446.973846;0.0;21.00
2019-06-14T12:43:04;39.404514;268.361115;1305.005615;0.000000;-0.019452;446.973846;0.0;21.00
2019-06-14T12:43:05;38.194447;267.222229;1292.380981;0.000000;-0.019452;446.973846;0.0;21.00
2019-06-14T12:43:06;38.194447;267.222229;1317.630371;0.000000;-0.019452;446.973846;0.0;21.00
2019-06-14T12:43:07;39.404514;267.222229;1305.005615;0.000000;-0.019452;446.973846;0.0;21.00
2019-06-14T12:43:08;38.194447;268.361115;1330.200562;0.000000;-0.019452;446.973846;0.0;21.00
...
```

Implementation in Spark:
In Spark, the data can be read in as a first step, converted to the correct data types and stored in a correctly typed dataframe. This represents a Spark standard data structure in which data is held in main memory. This can be implemented with a command in an interactive pyspark shell, which uses Python as the implementation language:

```
>>> df = spark.read.csv("file:///tmp/datacollection.csv", sep=';', header=True).selectExpr("cast(timestamp as timestamp)",
      "cast(temperature_heater as double)",
      "cast(temperature_boiler as double)",
      "cast(pressure_boiler as double)",
      "cast(rpm as double)",
      "cast(power_dynamo as double)",
      "cast(power_heating as double)",
      "cast(valve_aperture as double)").show()
+-------------------+------------------+------------------+---------------+---+------------+-------------+--------------+
|          timestamp|temperature_heater|temperature_boiler|pressure_boiler|rpm|power_dynamo|power_heating|valve_aperture|
+-------------------+------------------+------------------+---------------+---+------------+-------------+--------------+
|2019-06-14 12:43:00|         39.404514|        267.222229|    1292.380981|0.0|   -0.019452|   446.973846|           0.0|
|2019-06-14 12:43:01|         38.194447|        268.361115|    1292.380981|0.0|   -0.019452|   446.973846|           0.0|
|2019-06-14 12:43:02|         38.194447|        267.222229|    1292.326538|0.0|   -0.019452|   446.973846|           0.0|
```

With the following command the data can be stored directly in the SQL table EngineData:

```
>>> df = spark.read.csv("file:///tmp/datacollection.csv", sep=';', header=True).selectExpr("cast(timestamp as timestamp)",
      "cast(temperature_heater as double)",
      "cast(temperature_boiler as double)",
      "cast(pressure_boiler as double)",
      "cast(rpm as double)",
      "cast(power_dynamo as double)",
      "cast(power_heating as double)",
      "cast(valve_aperture as double)").createOrReplaceTempView("EngineData")
```

To filter out rows with incorrect values, queries can now be used based on the SQL table.
In this case negative power_dynamo values are filtered out:

```
>>> df_cleaned = spark.sql("select * from EngineData where power_dynamo >= 0")
```

A dataframe can be saved again as a CSV file after cleaning. The inclusion of the repartition function ensures that the result is saved in a file, even if the data frame was previously partitioned. This can be the result of parallel processing steps.

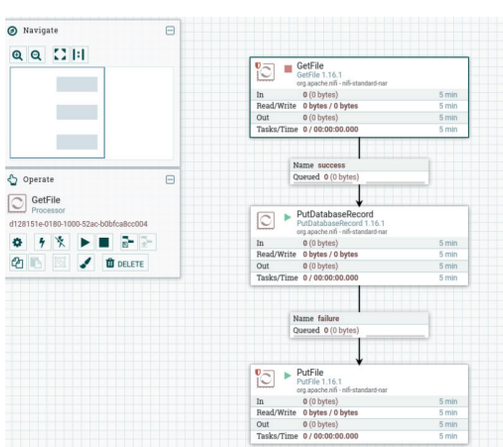
```
>>> df_cleaned.repartition(1).write.format('com.databricks.spark.csv').save("/tmp/cleaned.csv",header = 'true')
```

As an alternative, the dataframe can also be stored in a database via the JDBC API.
The following command saves the data in a SQLite database, for example:

```
>>> df_cleaned.write.jdbc(url="jdbc:sqlite:/tmp/engine.db", table="data", mode="overwrite")
```

Implementation in NiFi:
The example shown here again shows the reading of the CSV file with the heat engine data and its storage in a SQLite database. Here the CSV file is read in using the GetFile processor and converted into NiFi flowfiles. These are fed into a PutDatabaseRecord processor, which is configured to parse the CSV file correctly and access the database. Just like connecting the individual modules, their configuration is done interactively in the NiFi web interface.

The final PutFile processor is used to catch and store error conditions, such as incorrectly formatted lines in the input file. This allows error conditions to be easily traced in the saved text file. ◆



Agile software development using DevOps workflows

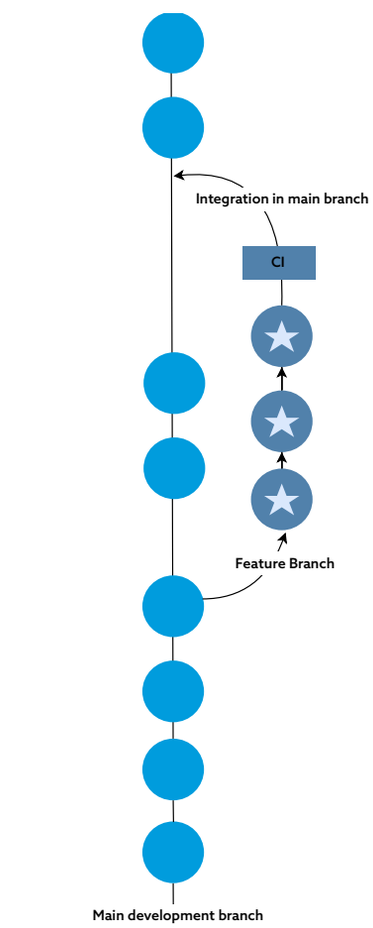
– Florian Haßler
Software Engineer in the Unit Domain-specific Applications

Developing and publishing software can be costly. Manual testing, integration and publishing steps take a lot of time and are prone to errors. Therefore, in the past, showstoppers often appeared very late in the course of the project and forced everyone involved to take a few steps back. To avoid this problem, a CI/CD DevOps workflow can be used.

Workflow

The developers select a ticket that has been recorded and finally specified by the customer. They start writing tests and subsequently cast the feature into code. All this happens far away from the active code base of the application in a separate code area. Before the feature is integrated into the active code base after all acceptance criteria have been met, it is put through its paces in an automated process. Only after it has been tested is it released to the test system and then to the production system. Why automated?

In the automation pipelines, tests can be defined once and thus the associated requirements can always be ensured without human intervention. Containerization additionally helps to make the tests reproducible. Publishing is also automated. A few minutes after implementation, a new feature is already visible on our customers' test system and available for user tests. Thanks to automated tests, programming errors rarely reach the test system. This means that customers do not have to deal with these errors and can concentrate on the content of the features. This article goes on to describe what the abbreviations CI and CD stand for at the developers of RISC Software GmbH. The four strategies presented here greatly simplify the everyday work for both developers and users.

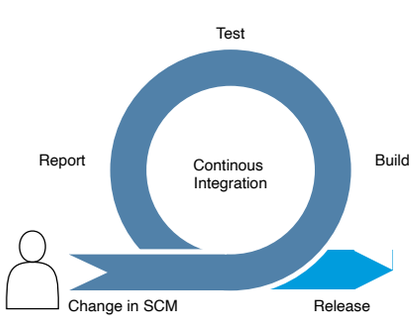


Experience with pipelines

There are several platforms for implementing such automation pipelines. At RISC Software GmbH, we have experience with Concourse, Jenkins and GitLab. Especially for the web development teams, we found that GitLab integrated best with their workflows.

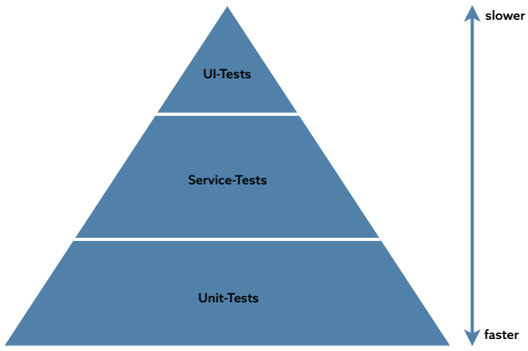
CI – Continuous Integration

One idea of Continuous Integration is to integrate new features as quickly and as often as possible in an automated way. This is about short feedback loops. They help developers to identify and fix problems before the context is changed (i.e. the next task is started). The key question is: Can the developed feature be integrated into the existing application without having a negative impact on existing functions?



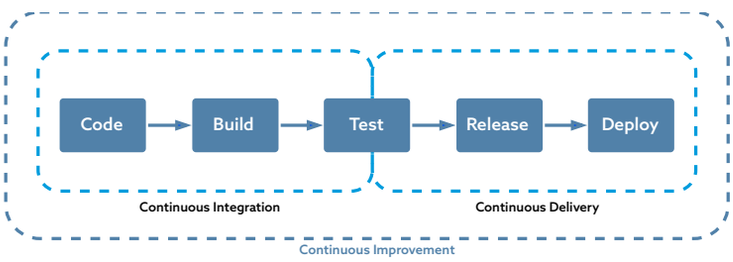
In simple terms, developers submit their changes to our source code management (SCM) system and the CI pipeline starts working. When finished, it notifies developers of success or failure. Depending on the type of project, different tools are used in the integration pipeline. It almost always starts with running test suites according to their position in the test pyramid (according to Mike Cohn). Unit tests form the basis of the test pyramid. The smallest possible behavioral units are tested. They ensure that the code works as expected. Integration tests are then used to ensure that the interaction between the various parts of the application or with external components (e.g. databases) also works. If a test fails, the test pipeline is aborted and the dev team receives a notification. Almost always, the last step in the RISC-Software-GmbH CI pipelines is a static code analysis by an appropriate tool. This helps to identify vulnerabilities and inconsistencies with common

development practices. The tool used at RISC Software GmbH contains many built-in rules for different programming languages. The result of the analysis is an evaluation of the technical quality of the source code. It also outputs the number of errors, vulnerabilities, test coverage, number of lines of code written and much more. The developers receive an automated evaluation of their code.



CD – Like: Continuous Delivery

Once the CI pipeline has been successfully run and the report from the code analysis tool is satisfactory, the continuous delivery pipeline is activated manually. It pursues the goal of executing the publishing process in an automated, fast and reliable manner. The artifacts built in the CI pipeline are collected and published to the test system. This publishing step can be simple, copying only source code to a server, or more complex. The target system and the location of the target system determine the complexity.



An advantage that is often mentioned in connection with Continuous Delivery is that the application can be published at any time. There is no time when there is code in the active code base of the version control system that is not functional. If a customer reports a problem, the team can respond promptly and release a new version. Another advantage is, of course, that classic releases are no longer necessary, but feature by feature can be tested and accepted directly by customers.

CD – Like: Continuous Deployment

Continuous Deployment takes the steps of Continuous Integration and Continuous Delivery one step further. Here, all changes that successfully pass through the CI pipeline are also released immediately. This process is fully automated and only a failed integration step prevents the changes from being transferred to the test and subsequent production system. This works if there is a high level of trust in the developers, the Continuous Integration and the Continuous Delivery process. Continuous Deployment has its place mainly in large-scale product development business; in classic project business, the necessary prerequisites are often not economically viable to implement.

CI – Like: Continuous Improvement

Figure 1 shows the notification that hit some dev teams the morning after the vulnerability in the Java logging library Log4j became known. The security checks of the libraries used run automatically every night to provide the applications we develop with patches as soon as security vulnerabilities become known. They also help to regularly update the software libraries used in applications developed by RISC Software GmbH, thus keeping them up-to-date. Improvements are implemented incrementally, evaluated, and then further improved. The result is leaner workflows and thus more time for the implementation of new features.

```
+-------------------------------+---------------+----------+-------------------+---------------+-------------------------------+
|            LIBRARY            | VULNERABILITY ID | SEVERITY | INSTALLED VERSION | FIXED VERSION |             TITLE             |
+-------------------------------+---------------+----------+-------------------+---------------+-------------------------------+
| org.apache.logging.log4j:log4j-api | CVE-2021-45046 | CRITICAL | 2.15.0            | 2.16.0        | log4j-core: DoS in log4j 2.x  |
|                               |               |          |                   |               | with thread context message   |
|                               |               |          |                   |               | pattern and context...        |
|                               |               |          |                   |               | --->avd.aquasec.com/nvd/cve-2021-45046 |
+-------------------------------+---------------+----------+-------------------+---------------+-------------------------------+
```

Figure 1: Notification after disclosure of the vulnerability in the Java logging library Log4j

Benefits

Through continuous integration, delivery, deployment, and improvement, RISC Software GmbH has gained the ability to release frequently without compromising quality. - The benefits of our DevOps workflow for customers:

- Minimized risk of error (and errors can be corrected more quickly)
- Shorter response times (customer feedback can be incorporated more quickly)
- Lower costs for manual testing (more features for the same amount of money)
- Progress in the development process is visible
- Security gaps are quickly detected and eliminated
- Components, frameworks and libraries are updated more efficiently ◆

Transformer models conquer Natural Language Processing

– Fabian Jetzinger, BSc
Data Scientist in the Unit Logistics Informatics

How to make the most of pre-trained models like Google T5

Artificial intelligence (AI) has become an integral part of our everyday lives. Every day we are in contact with numerous systems that are based on AI – even if we are not always aware of it. However, it becomes noticeable when machines communicate with us in our language, as is the case with voice assistants. Considerable progress has been made in AI-driven understanding of natural language in recent years, among other things through so-called Transformer models – including the T5 architecture developed by Google, which is presented in this article.

The field of Natural Language Processing (NLP) deals with the understanding of natural, human language. An introduction to the topic is provided by the specialist article in the first Insights magazine "OK Google: What is Natural Language Processing?

Since the machine construction of natural language understanding requires huge amounts of text data of several gigabytes (e.g., the entire text from Wikipedia), the training of NLP systems from scratch involves considerable costs (up to six-digit euro amounts or more) and time[1] until an acceptable quality of the results can be achieved. Therefore, numerous researchers as well as large companies like Google or Facebook make pre-trained language models publicly available. Since these so-called base models have already learned a basic understanding of language, other researchers can build on these models and adapt them for a specific project, extend them, and in turn share them with the NLP community.

This concept is called transfer learning. Here, models are trained with huge amounts of data to gain a general understanding of basic concepts (in this case, natural language) and then trained with more specific data to perform a concrete task. This not only allows reuse of models, but also reduces the size of the database needed for a concrete task.

When only limited data is available, zero-shot learning can be used. Here, a model is made to perform a task for which it has not been explicitly trained. One-shot and few-shot models also work according to this principle. Thus, models with no or only little data can be trimmed to a specific task. Unfortunately, these approaches do not completely eliminate the need to collect training data in very many cases: first, these zero-shot models often only serve as prototypes or baseline models, since they often cannot generalize sufficiently well (i.e., decreasing quality of results with new, unseen data) and are very susceptible to errors in the (training) data. Second, additional evaluation data is needed to quantify the quality of zero-shot models in the first place and to make an assessment of the value added by their use. One of the biggest breakthroughs in NLP in recent years has been the development of so-called transformer models. This is a special neural network architecture that uses so-called attention mechanisms and replaces the previously prevailing recurrent neural networks (RNN) with deep feed-forward networks. This novel architecture, developed in 2017, was able to largely eliminate the weaknesses of the previous models. Among other things, it enables a (better) understanding about the context of certain words and facilitates the performant handling of larger data sets. Probably the best known example of a group of Transformer models is BERT[2] (Bidirectional Encoder Representations from Transformers). The basic BERT models can be extended by customized extensions, which are trained on the desired task (e.g. classification of sentences on negative, neutral or positive sentiment). Systems based on the BERT architecture have achieved record-breaking results in numerous tasks since its publication in 2018 and are now an integral part of Google searches.

What is Google T5?

The T5 architecture[3] (Text-To-Text-Transfer-Transformer) developed by Google works very similarly to BERT, but has some differences. The eponymous Text-To-Text principle means that for T5 models, input and output consist of pure text data. This allows training the T5 models on arbitrary tasks without having to adapt the model structure itself to this task. Thus, any problem that can be formulated as text input to text output can be handled by T5. This includes, for example, classifying texts, summarizing a long text, or answering questions about the content of a text. Another unique feature of the T5 architecture is the ability to use a single model to solve several different tasks, as shown in Figure 1. For example, the pre-trained T5 models have already been trained on 17 different tasks. Among these, the task of answering questions about a given text is a particular strength of T5. If the model is provided with the entire Wikipedia article on the history of France, for example, it can successfully return the correct answer "Louis XIV" to the question "Who became King of France in 1643?".

T5 has already achieved impressive results in various areas, but as in all areas of artificial intelligence, it is not a solution to every problem (see also No-Free-Lunch-Theorem). A common approach to

[1] Sharir, Or, Barak Peleg, and Yoav Shoham. "The cost of training nlp models: A concise overview." arXiv preprint arXiv:2004.08900 (2020).

[2] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

determine the best solution to a problem is to test multiple model architectures for a task and then compare the results.

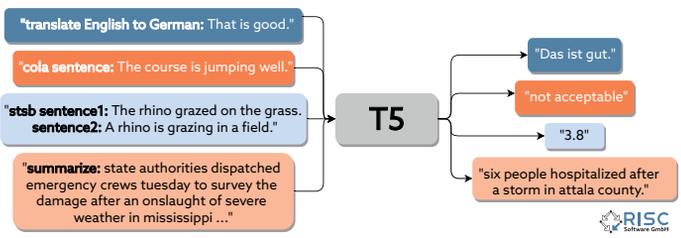
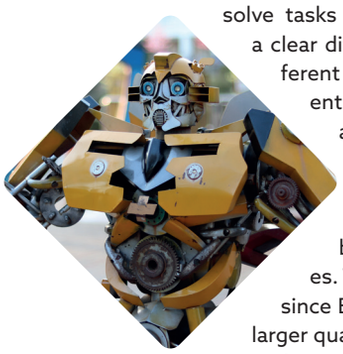


Figure 1: T5 model

Parlez-vous français? Oui!

An extension of the T5 models are the mT5 models[4], which have been trained on huge amounts of texts in a total of 108 different languages (as of 2022-01). This allows a single model to solve tasks in different languages. However, there is a clear difference in the quality of the results in different languages - the less a language is present in the training data, the worse the results achieved in it.



It is also particularly useful that languages for which less data is available can benefit from training data in other languages. This is especially helpful in data acquisition, since English-language texts are often available in larger quantities than texts in other languages. Thus, even if tasks are to be solved in only one language, mT5 models can still be helpful if there is not enough training data available in the target language. However, the results obtained tend to be worse than when training only on the target language with a similar amount of total data.

How do I use T5 for my use case?

The first step should be to clarify some general, important questions about the specific use case:

- In which languages is the problem to be solved?
- Where does the data needed for the training come from? Are they suitable for my use case?
- How can I check the quality of the model?
- Is test data available?
- What resources (computing power, time, financial resources, etc.) are available for training or fine-tuning?
- Which models are applicable for my use case?

For the potential use of T5, it must additionally be evaluated whether the use case can be formulated as a text-to-text task.

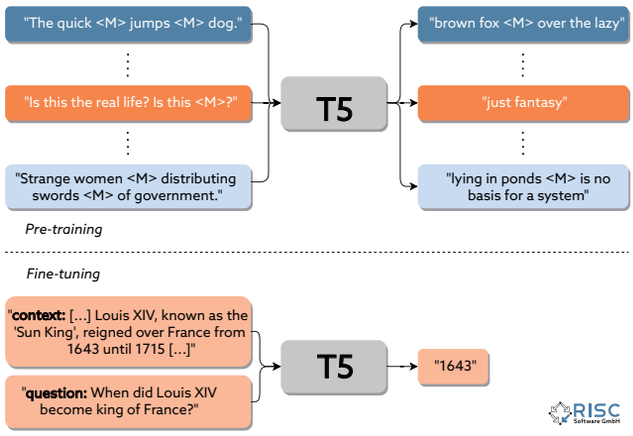


Figure 2: Pre-training and Fine-tuning

Next, a suitable pre-trained T5 model is selected. One of the best and most well-known sources for this is the Hugging Face platform, which provides a variety of pre-trained state-of-the-art models and also datasets as open-source solutions for the public. These are often available in a variety of sizes, and a balance must be struck between the computational power or time required and the quality of the model. While larger models generally provide better results, they also place significantly greater demands on resources, so the largest model is not always automatically the best choice.

Now it is time for the so-called fine-tuning. The pre-trained model (which at this point already has a basic understanding of language) is trained to solve a concrete task as shown in Figure 2. For this, training data is necessary, which shows the model which input should lead to which output. Not only the quantity, but also the quality of the data is crucial.

Challenges include the choice of the concrete base model, the acquisition and quality control of the training data, and the preparation of the data in a format that can be read by the model. Additional difficulties arise, for example, with longer text sequences, since Transformer models have a limit in terms of text length.

RISC Software GmbH has been researching the use of T5 models for some time. For example, a system for recognizing and assigning proper names in texts (Named Entity Recognition, or NER) could be extended and improved using T5. For this purpose, the already pre-trained ability of T5 to answer questions about a given text is used. Thus, the T5 model can answer questions such as “Which person is involved?“ or “Which company is involved?“ and thus assist the NER system.

Due to its design, the T5 architecture is suitable for a wide variety of different tasks (or combination of these) and has already been able to deliver impressive results in numerous application areas. It remains extremely exciting to see how these technologies will develop in the future and what successes can still be achieved with them. ◆

[3] Raffel, Colin, et al. “Exploring the limits of transfer learning with a unified text-to-text transformer.” arXiv preprint arXiv:1910.10683 (2019).

[4] Xue, Linting, et al. “mT5: A massively multilingual pre-trained text-to-text transformer.” arXiv preprint arXiv:2010.11934 (2020).

Technical debt and legacy systems

– DI (FH) Andreas Lettner
Head of Unit Domain-specific Applications and Head of Coaches

When do contracts encourage this development?

Wherever software is used, there is a risk that old systems will mutate into legacy systems. This means that they can no longer be further developed or maintained within reasonable costs and without foreseeable technical risk. So how can you prevent a system or legacy system from becoming a legacy system? Read the following article to learn how to identify legacy systems in time, how to remove them and ultimately how to avoid them.

How are legacy systems detected?

An essential quality criterion for software is long-term maintainability and expandability. By focusing on this criterion, it can be ensured that future extensions can be accurately classified in terms of costs, implementation time, cost-effectiveness and risk. Figure 1 uses the green line to show an optimal cost development in relation to the age of the software. An initial increase in costs is associated with investments in quality, which stabilize costs in the medium term and keep the software alive.

In contrast, the red line shows a possible development for a system where quality criteria are neglected. Each red dot shows a decision in development where a compromise was made between quality and cost or possibly implementation time. This creates something known as "technical debt" in the software. This technical debt leads to an increase in the complexity of future maintenance and enhancements, and therefore an increase in the cost of change, as well as the associated risk. From experience, technical debt quickly leads to parts of a system or the system as a whole becoming impossible to maintain or extend.

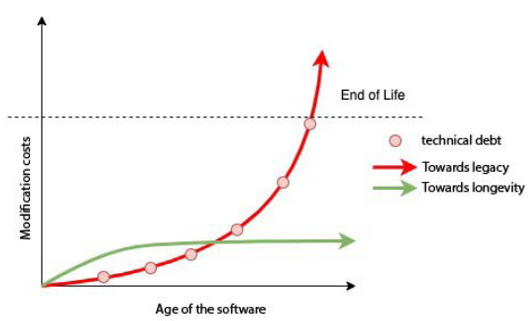


Figure 1: Cost of change

The evolution of a system towards a legacy system is generally easy to measure. Among other things, the following signs can be observed here:

- The costs and implementation times for changes increase over time. This can be observed particularly well for tasks with comparable content.
- There is a measurable increase in the error rate in the production system during or after version updates.
- The deadlines for go-live cannot be consistently met.
- The developers* show uncertainty in the estimates.
- Questions about feasibility are piling up.
- Product managers and developers are becoming reluctant to push for change.
- There is increased talk of workarounds.

How do you deal with an existing legacy system?

Technical debt will always arise in the course of software development. This is unavoidable. However, the processes in software development can be prioritized in such a way that technical debt can be reduced. A common mindset in the project team is necessary for this. A constant monitoring of the technical debts and the authorization of the developers to be able/allowed to remove these again are inevitable for this. Methods that are used here include code reviews, coding guidelines, pair programming, refactoring and test-driven development. Figure 2 shows the constant correction of technical debt through small adjustments.

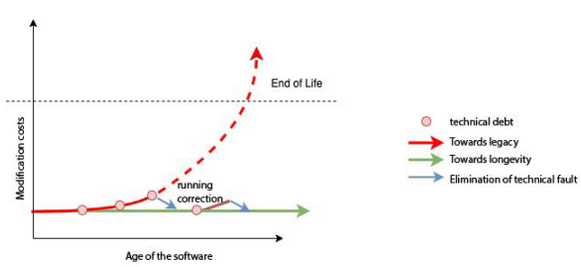


Figure 2: Constant quality management

Depending on how far along the software product is on its journey towards the "end of life", minor adjustments may no longer have any effect and major refactorings or reengineering may become necessary. In this case, it is helpful to bring reengineering specialists into the project team to gradually perform a consistent reengineering of the existing legacy systems. Figure 3 shows the late identification of technical debt. Here, countermeasures must be taken with significantly more effort and time in order to bring the system back onto the right path.

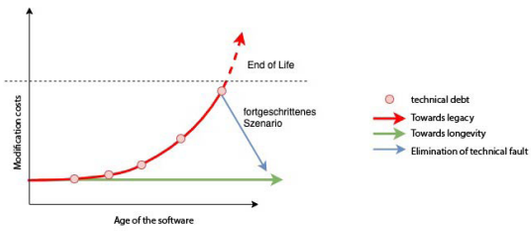


Figure 3: Reengineering, major refactoring

How to avoid a legacy system?

In addition to observing technical debt, the question arises as to how technical debt can be avoided. To do this, it is necessary to analyze how technical debt comes about. What are the causes and what can be done to mitigate these causes?

As stated in the previous section, developers are the ones responsible for the technical quality of a software product. Technical guilt is not integrated therefore deliberately into a system, but promoted by external factors. If one regards the temporal course of a software development, then the following connection can be determined: The blue line shows the linear project execution, i.e. the fulfillment of the planned project scope up to the delivery date at V1.0. The ideal line in reality should correspond to the green line. There are continuous adjustments in the implementation, which keeps the product development "on track" and thus ensures that a valuable, executable product is available on the planned date.

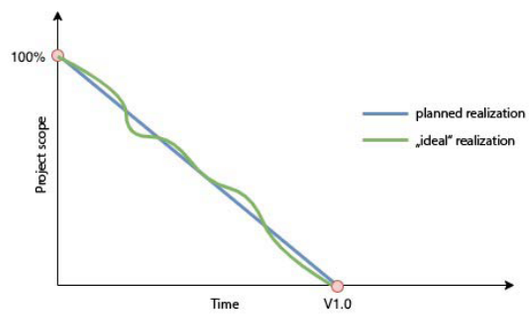


Figure 4: Development course

Figure 5 shows a picture which, however, is usually found in reality. A deviation from the plan is noticed. There can be various reasons for this. Possibly an initial estimate was not accurate, the team has lost efficiency, there are external influences such as sick leave, the complexity of the software has changed, the project scope has changed, etc.. The reasons for this can be many and varied. However, at this point it becomes interesting to see what options a project team has. The following primary decisions can be made considering the circumstances.

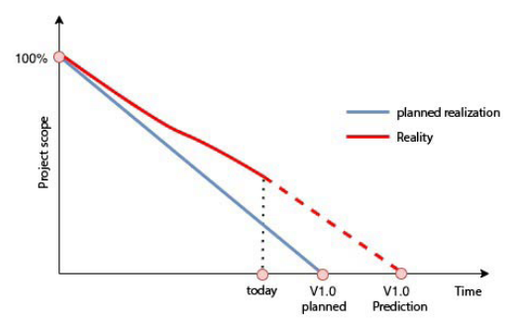


Figure 5: Reality

Option 1: The product managers postpone the completion date

The postponement of the completion date is done only if the people responsible for the product have the possibility to do so. Behind a postponement of the completion date are usually other people or systems that are dependent on it and more or less insist on the originally planned date. In addition, there is the disadvantage that not only a go-live and thus the (possibly economic) benefit is postponed in time, but also the project team has to be financed for a longer period of time. This leads to additional costs and later revenues. A direct, negative impact on the ROI is the consequence.

Option 2: The development speed is increased

The least resistance can be expected if the developers' working speed can be optimized to meet the completion date and the development costs. In fact, this is an option, although we need to distinguish between two different scenarios.

Option 2a: There are influencing factors, which disturb the team in the efficiency, which can be actually repaired. These measures work and are also those that take place during continuous process control and improvement. The scenario in Figure 4 assumes precisely this. The prerequisite for these measures, however, is that the deviations are not detected too late and are not too large. In Figure 7, it may already be too late for such a measure.

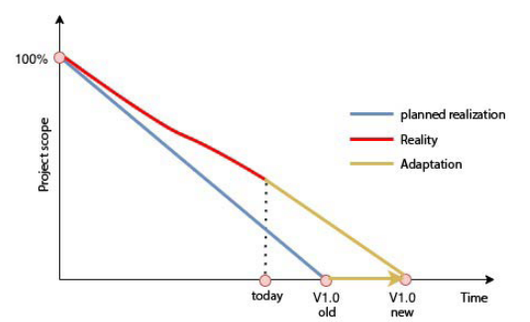


Figure 6: Move the release date

Option 2b: There are disruptive influencing factors, but these are not dealt with or it is too late for appropriate measures. In this case an increase of the development speed is reached only by the structure of pressure on the developers. Pressure usually leads to the fact that in the methods one saves, which do not have an obvious direct influence on the product range of a software. These are those, which are necessary for the preservation of the quality and thus it comes successively to a structure of technical debt. Since there is also a lack of time and capital for the reduction of the technical debts, this build-up progresses with appropriate speed if necessary.

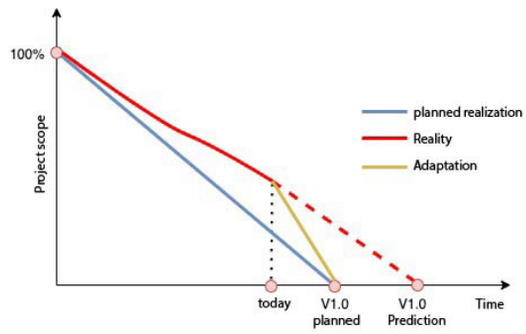


Figure 7: Increasing the speed of development

Option 3: Reduction of the project scope

The third option is to adapt the content of the software product. However, the product owner must have the appropriate authority to do this, and intensive negotiations between the product owner and the stakeholders may be required. Adaptations of this kind are facilitated by various measures:

- Variable project scope - variable content
- Ongoing prioritization of required functions
- Development of functions according to their priorities
- Continuous monitoring of speed, timeline, content, ...

In fact, such an adjustment in the project scope is possible at any time in the implementation. Detected in time, the changes are marginal; detected late, the changes are correspondingly more extensive. The following applies here: Provided that the product already contains the most valuable functions, a productive implementation with a reduced scope of functions should not be an obstacle. With appropriately well-trained and experienced product managers, this risk can be significantly minimized or even avoided.

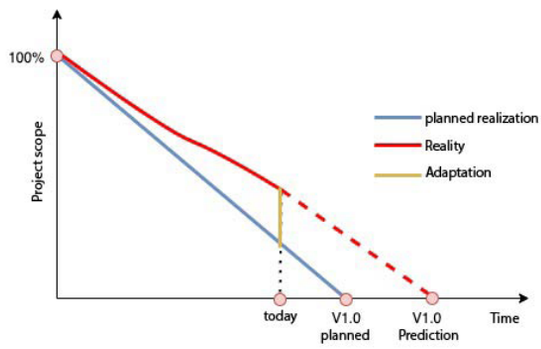


Figure 8: Adjustment of the project scope

Which option is the best?

As is so often the case, no general statement can be made here. Possibly one of the options makes sense on its own or a corresponding combination of options 1, 2a and 3 leads to a corresponding correction of the price. Only option 2b should be avoided, since exactly this ultimately develops a system in the direction of "End of Life".

Measures that ultimately prevent a legacy system are

- a strong and present leadership from an experienced and empowered product owner,
- the continuous focus on initiative and proactive improvement by a coach and
- deliberate technical quality assurance measures by the developers.

Why do you contractually bind yourself to technical debt?

Finally, an important influencing factor should be mentioned, which can limit options 1 - 3 accordingly: The contract. It is understandable that clients want to have contractual security with regard to what scope will be delivered in what time frame at what cost.

For this reason, contracts are often concluded with the contractors in which all three factors are fixed. However, this leaves the person responsible for the product with no room for maneuver:

- The release date is fixed - there must be no delay (option 1)
- The costs are fixed - no additional work may be carried out (option 1, option 2a)
- The scope is fixed - no features may be removed (option 3)

Thus, from a contractual point of view, usually only option 2b remains, namely increasing the pressure on the developers and the associated minimization of the quality of the product.

Instead, it is recommended that release dates and costs be fixed (e.g., by stabilizing the team) and that flexibility be created in the requirements for a product. Through the leadership of a product manager, clients can control the required functional scope of a software product up to the release date and thereby obtain transparent cost control. In addition, the product can be adapted to market needs during implementation and thus positioned in a target group-oriented manner. ◆


QSORT
YES
NO
YES
14
t[mid]
==
X
File Edit Search Run Compile
Procedure QSort(numbers : Array of
left : Integer; right
Var pivot, l_ptr, r_ptr : Integer;
Begin
l_ptr := left;
r_ptr := right;
pivot := numbers[left];
While (left < right) do
Begin
While ((numbers[right] >= pivot) AN
right := right - 1;
If (left <> right) then
Begin
numbers[left] := numbers[right];
left := left + 1;
End;
While ((numbers[left] <= pivot) AND
left := left + 1;
If (left <> right) then
Begin
55:1
F1 Help F2 Save F3 Open Alt+F9
COPY 2
DATA

Time series analysis - but correct!

– DI Dr. Alexander Maletzky
Researcher & Developer in the Unit Medical Informatics

How the choice of training data affects the practicality of models.

Time series data, for example, machine data in industry or vital signs in medicine, are nowadays an important data source for the analysis of complex systems. Modern analysis systems are mostly based on machine learning methods, i.e., learned prediction models, and draw on these data sources. However, for the development of practical models, the right choice of training data is a challenging task.

The problem: The right length of the sequences

Time series data are usually recorded automatically by sensors at regular intervals, and can be visualized as a line graph as shown in Figure 1. As explained in the technical paper Exploratory Data Analysis with Time Series (p.18), visual inspection of time series data is an important step in the data analysis workflow, which poses some difficulties. Even more challenging, however, is automatic time series analysis, where an AI model independently classifies time series, detects anomalies, or predicts the future course of a time series. Nowadays, models of this kind are mostly based on machine learning methods, i.e. they "learn" to make the right decisions independently based on training data. One of the main tasks of the developers of the models is - in addition to the selection of the appropriate model class and parameters - above all the selection of the training data. Time series are usually available as long sequences of measured values that extend over longer periods of time. Depending on the field of application, however, models should be able to make valid decisions already on the basis of comparatively short excerpts, and must therefore also be trained on such excerpts - and how these are selected has a strong influence on the practical suitability of the resulting models. On the one hand, the choice has to be made in such a way that no so-called sampling bias arises, i.e. the samples adequately reflect the different aspects of the time series (curve morphology, periodicity, trends, etc.). On the other hand, the trained models should perform correctly on those events that are of particular interest to the user. If these occur only rarely, they must be taken into account accordingly disproportionately in the model generation, which in turn can lead to a sampling bias.

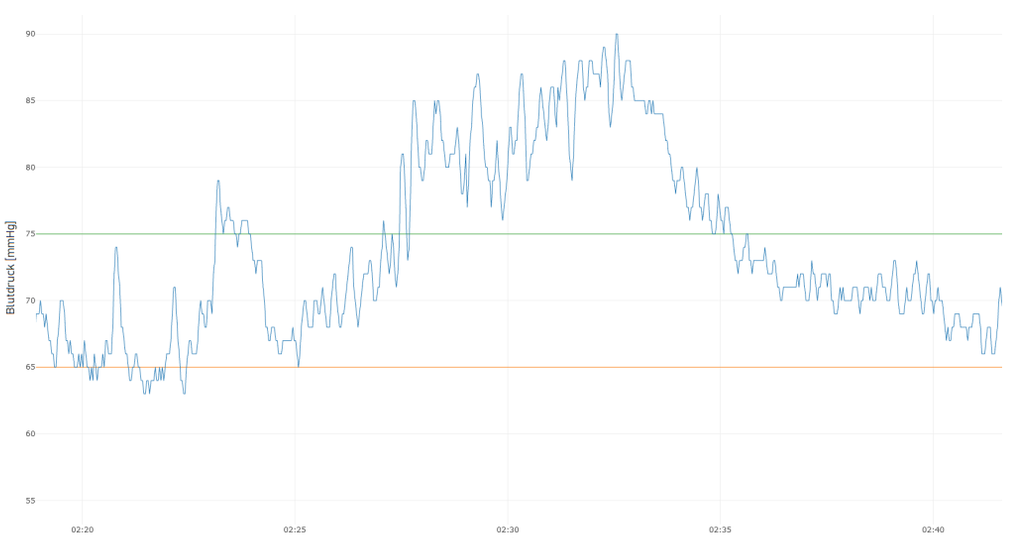


Figure 1: Mean arterial blood pressure (MAP) of an intensive care patient for 30 minutes, with one reading per second. The green line indicates the value above which the blood pressure is considered normal, and the orange line that value which represents a critical drop in blood pressure.

A concrete example from intensive care

In intensive care medicine, the condition of patients is continuously monitored to enable rapid intervention by nursing staff if necessary. Particular attention is paid to acute hypotensive episodes (AHEs), i.e., critical drops in blood pressure that can lead to irreparable damage. The prediction of future AHEs in the form of an early warning system, in order to be able to take countermeasures before they occur, is currently a highly regarded research topic in the field of artificial intelligence [1, 2].

Researchers from the Department of Medical Informatics at RISC Software GmbH are currently working on this issue in the MC³ project together with research partners from MedCampus III at Kepler University Hospital and the Institute for Machine Learning at JKU Linz.

Model development: sample selection for high classification accuracy

One possible strategy for selecting training samples is based on the time series of mean arterial blood pressure (MAP; see Figure 1): Whenever the MAP falls below the critical value of 65mmHg, a positively labeled sample is selected, i.e. a short observation window based on which the model should later be able to predict the upcoming drop and trigger an alarm. If instead the MAP remains constant above 75mmHg for a longer period of time, a negatively labeled sample is selected in it, i.e. here the model should not trigger an alarm. Figure 2 shows the sample selection schematically. In both cases, various time series data of the patient in the observation window, e.g. MAP, heart rate and oxygen saturation, serve as input data for the model.

Classification models trained on these training samples achieve high classification accuracy on the independent test set (which is generated according to the same scheme as the training set). Thus, nothing stands in the way of a practical application.

Use in practice: Where is the error?

Of course, the developed model was not immediately used in the hospital, but the practical application was first simulated in a test environment. This showed that the model triggers alarms almost continuously, even when there is no AHE in sight far and wide. Although the classification accuracy on the test set is very good, the model does not work in practice.

Error analysis: selection of training samples

What is the reason for the model's unsuitability for practice? As has been shown, the training samples contain only "extreme examples" that can be easily classified but cover only a small part of the spectrum of possibilities that occur in reality.

This is because the MAP usually changes only slowly, i.e., is usually significantly lower at the end of the observation window of positively labeled samples than of negatively labeled samples. The model ignores all time information and all other time series and only pays attention to the last available MAP value: If this is rather high, no alarm is triggered, otherwise it is. This does not work in practice, because the MAP then often moves in a "gray zone" that does not occur in the training samples.

How to do it better?

Sampling bias can be avoided by selecting training samples either randomly or regularly (e.g., every 10 minutes), regardless of MAP. However, such an approach brings other problems: on the one hand, classifying a sample into "positive" (MAP will fall below critical value) and "negative" (MAP will remain normal) is no longer so simple, because what to do if MAP remains above 65mmHg, but only just? It therefore makes more sense not to train a classification model but a regression model, for example to predict the exact MAP value 15 minutes later.

Another problem is the phenomenon discovered in the course of the failure analysis that the MAP usually drops only slowly. From a medical point of view, it is precisely those (rare) cases where the MAP drops rapidly that are interesting, because an early warning system only makes sense in such cases. In order to "sensitize" the model for such situations, they can be given higher importance during training. Researchers of the MC3 project are currently training prediction models for acute hypotensive episodes based on the new approach. If they prove to be practical, they could support nursing staff in intensive care units in the near future.

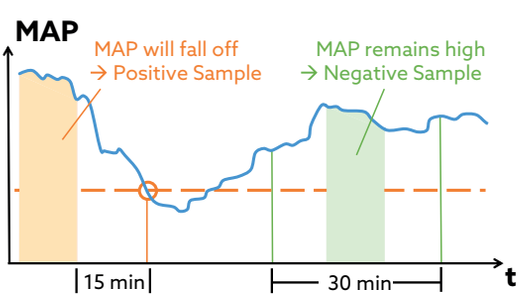


Figure 2: Schematic representation of sample selection as a function of mean arterial blood pressure (MAP).

Conclusion

As always in machine learning, a comprehensive understanding of the data and the use case is essential for the development of well-functioning, practical models - not only in the medical environment. Often, both domain knowledge and exploratory data analysis (see Exploratory Data Analysis with Time Series) are necessary to extract the necessary information, identify potential problems early on, and adjust model development accordingly. As explained, this includes in particular training samples, the correct choice of which plays a central role especially in the case of time series data. ◆

Exploratory data analysis with time series

– Dominik Falkner, MSc
Data Scientist in the Unit Logistics Informatics

Knowledge gain through data collection and data evaluation over time

Data science makes it possible to extract useful knowledge from data. Data is as diverse as people. Not only are they different in form - such as nominal, ordinal or cardinal - in order to derive knowledge from them, it is essential for many data to be recorded over a period of time. Therefore, data are often measured over a time course and so-called time series are created. A time series consists of a series of data points, which are sorted by a timestamp (e.g. of the form 1.1.2021 11:00:01). Time series can be found in a wide variety of industries, whether in industry (derived from manufacturing processes), medicine (ECGs), or the financial market (stock prices). Often time series are a central point for decision making, but bring some challenges for data analysis. The following article uses examples to show how exploratory data analysis can be designed with time series.

Data Science: Brave New World

Data Science is a science that has been established for decades, but for many companies it is still new, as they are only now recognizing a benefit for themselves from it. The hype surrounding this topic in recent years caused many industrial and commercial companies to introduce Data Science without established approaches. This led to a lot of data without annotations and the knowledge about it being scattered among selected experts. The biggest challenge is to know how data is generated, how systems react and how to interpret the data. This knowledge is distributed among both data scientists and experts in the application area and must first be brought together through intensive collaboration. The success of data science projects therefore depends heavily on the cooperation of the various knowledge carriers.

Time Series Exploration: What Catches the Eye Directly

Humans have a very good and fast visual perception, which allows them to understand relationships more quickly by looking at images than by reading raw numbers and texts. Visualizations are excellent for helping experts gain a comprehensive overview of complex data. Diagrams serve as a critical tool for explaining data properties. The pitfall, however, is that while well-designed visualizations are enormously helpful, naive attempts can often be misleading and quickly prove ineffective.

However, before the implementation of visualizations can begin, another aspect is crucial for time series: the so-called sampling rate. The sampling rate specifies how often the analog signal is sampled over the measurement period and therefore determines how precisely the process is observed. It is already determined when values are collected. It is usually specified by the subject matter experts, since the sampling rate is difficult to determine without prior knowledge. The sampling rate also influences the recording time and the data size: a very high sampling rate can lead to immense memory consumption, which makes both processing and data storage more difficult. If analyses already show useful results, it may make sense to reduce the sampling rate, thus saving memory and speeding up computations. For further processing by algorithms, it is furthermore often assumed that the sampling rate is the same between time series.

As can be seen in figures 1 and 2, there are different types of time series. The first graph shows a derivative of an electrocardiogram (ECG), which is used as a basis for defibrillators to decide whether to send an electrical impulse or not. The second graphic shows physical values extracted from a manufacturing process. Both processes occur repeatedly and can therefore be compared. Often these processes are also recorded continuously, resulting in a long combined time series. In this case, this must first be broken down into individual sequences, which are then superimposed. At this point, exploratory data analysis can be used to search for patterns and commonalities in the sequences without much effort. It is usually necessary to visualize the time series. The simplest form for this is the overlaid line chart, an example of which is shown in Figure 3. This representation allows several time series to be compared quickly and helps to maintain an overview. Depending on how many time series are displayed at the same time, it may be useful to highlight interesting data points (e.g.: the newest time series or those with special characteristics). Another possibility is to filter the data itself. Often there are environmental parameters that influence the process, such as outdoor temperature. This makes it easier to identify patterns. These visualizations are used by both data scientists and domain experts. The goal here is close collaboration between domain experts and data scientists so that as many insights as possible can be drawn from the data.

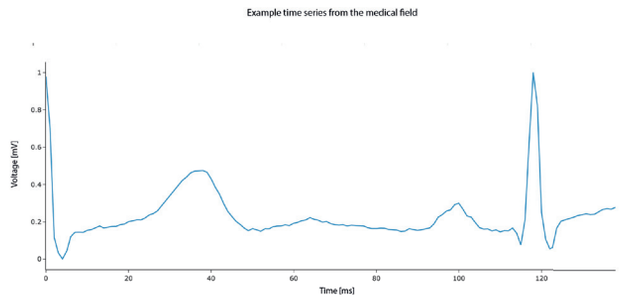


Figure 1: Example time series from the medical field showing a derivative of the electrocardiogram.

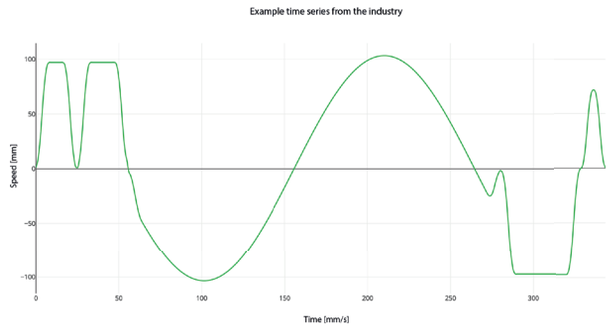


Figure 2: Example time series from industry showing the speed at which a machine moves.

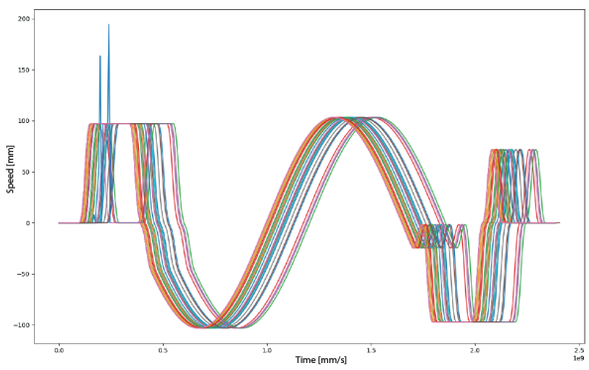
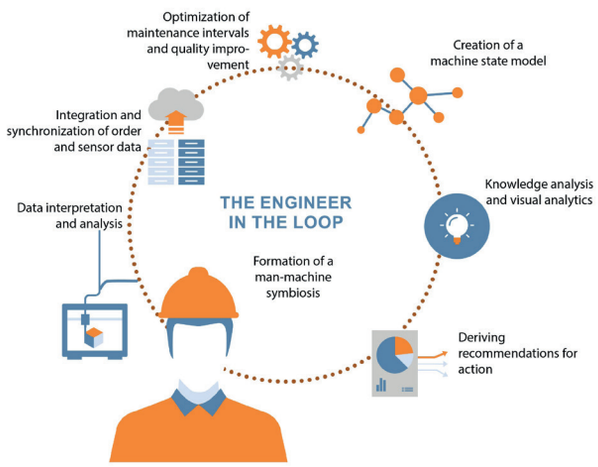


Figure 3: Superimposed line chart of several time series, which often serves as a starting point for initial analyses and provides a good overview.



Expert-in-the-loop: Collaboration is the most important thing

Algorithms alone cannot gain insights from data - it is even more difficult if the data is not annotated. Even experts often cannot solve problems on their own. The missing link between the two worlds is the concept of "expert-in-the-loop". This combines the information found by the algorithm and the knowledge and experience that experts have gathered over many years. The main idea is to improve or control the results of an algorithm. The results can then be used to enrich the analyses. The new information helps experts to find patterns and can show correlations that were previously difficult or impossible to see. In addition, experts can provide feedback on what the algorithm recognizes well and what it does not. On this basis, the algorithm can be adapted.

Another way to support subject matter experts in exploring data is to make the usability of the interface as interactive as possible. If the visualization of the data is only static and the experts cannot interact with it, the possibilities for visual acquisition are very limited. If, on the other hand, it is possible to influence the display options, the perspectives expand and with them the chance to find connections. The integration of domain knowledge works best if the data can be visualized in a way that can be interpreted by experts. On the one hand, it is necessary to create visualizations that express the different properties of the data, and on the other hand, there must be an interface that is easy to understand and allows domain knowledge to be incorporated.

Conclusion

Time series analysis is an excellent way to gain insights from data over time. Visualization in particular supports people in recognizing correlations and patterns. This is especially important so that subject matter experts can enrich the data with their domain knowledge and thus generate a great deal of added value from the data through their active participation in the data science process.

Another way to compare time series is time series clustering. This represents a method to automatically search data for patterns. Read more about this in one of our next technical papers. ◆

We need a mobility revolution!

– Sabrina Wagner, BSc
Software Developer in the Unit Logistics Informatics

Climate-friendly alternatives for the mobility of the future

Self-determined mobility is an integral part of everyday life and a convenience that people hardly want to do without nowadays. However, ever-increasing traffic is responsible for 30% of Austria's CO2 emissions [1]. A shift from undivided, motorized individual transport to climate-friendly alternatives is urgently needed. What (transport) means can we use to initiate the mobility revolution together and to change mobility habits towards environmentally friendly alternatives?

The problem: CO2 and inefficiency

More than 5 million passenger cars in Austria alone - equivalent to 0.6 cars per inhabitant regardless of age, driving ability or driver's license ownership - emit 17.13% of Austria's CO2 emissions [6] [2] [3]. However, in addition to the ever increasing number of passenger cars in Austria, the occupancy rate is also decreasing: in 2017, the average number of occupants in a vehicle was 1.15, which means that vehicles carry fewer and fewer people - usually even only one person - and still take up space and resources [4].

Public transport

The classic alternative to your own car is public transport, such as buses or trains. Especially in recent years, the public transport network has been greatly expanded to increase the attractiveness of this sustainable mode of transportation. In addition, various tariffs are offered for commuters and frequent travelers, such as annual passes and semester tickets. Since public transport in metropolitan areas can already be used very flexibly and is financially very attractive, it represents a real alternative to undivided private transport.

In rural regions, however, the range of services is not yet developed enough in many places to meet the requirements of flexible and convenient public transport. This flexible need for mobility is intensified by current developments such as flexible working hours and home offices, and is often not covered for all interest groups by public transport in rural areas, which is primarily designed for school transport.

Carpooling

Carpooling services, as well as carpooling in general, are an option to cover everyday and recurring routes in a cost-effective and quick way. To share the commute to and from work with colleagues, different providers offer ride-sharing services as part of the commute. These regular ride share services cover only part of the mobility needs, but in addition to the financial and environmental incentives, they can also reduce the need for a second car in the household. For businesses, corporate ride share programs also offer social and financial benefits, such as increased cohesion and reduced parking costs.

Shaped by the digital age, there is a wide range of ride-sharing services and apps available on the market, such as Carployee, Foahstmit, Ummadum or Hey Way. This (local) fragmentation of the various solutions leads to segregated groups of users and, as a consequence, to a smaller number of routes offered. Other providers, such as BlaBlaCar with its focus on long-distance routes, can be a supplement for routes not covered by public transport.

In addition to digital ride-sharing services, there are also offers for analog and spontaneous ride-sharing, such as the "Mitfahrbankerl" offered by the Freistadt energy district. They are intended to replace classic car hitchhiking and offer a flexible way to ride along.

Car-Sharing

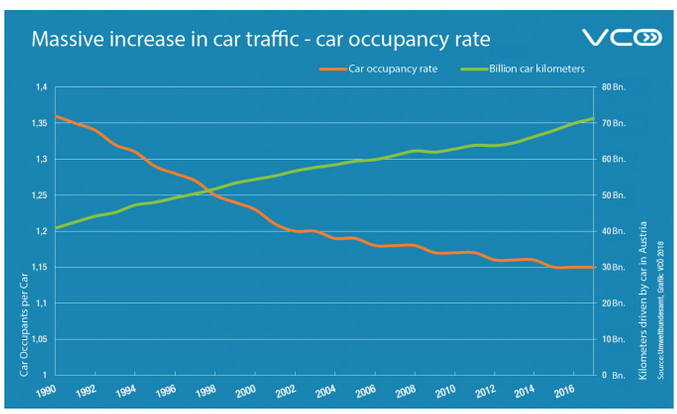
Almost exclusively, the question of environmental impact in the area of mobility considers fuel consumption and the emissions caused by it. However, in addition to the problem of low occupancy rates and the resulting high climate impact, there is another factor: vehicle production itself. A vehicle causes greenhouse gas emissions to the extent of several tens of thousands of kilometers driven already during production, without ever having been driven a kilometer [7].

Car-Sharing offers a good alternative to buying one's own vehicle, in that a passenger car can be rented flexibly for the required duration. Car-Sharing is an optimal complement to public transport, because when an individual vehicle is needed (e.g., to travel to leisure activities that are not open to the public), different types of vehicles can also be rented flexibly over time.

There is also a high degree of fragmentation among Car-Sharing providers and a confusing information situation for customers due to the different payment methods and availability. There are both private providers, such as on the Getaround platform, and public providers, such as ÖBB Rail&Drive, TIM or MühlFerdl. These also have different objectives, such as local supplement to public transport and/or replace-

[1] Umweltbundesamt, 2018a
[2] Umweltbundesamt, 2018b
[3] Statistik Austria, 2018a
[4] Trafico, Umweltbundesamt, VCÖ 2018
[5] Greenpeace, Städtische Mobilität nach Corona: Auto-Kollaps oder Fahrrad-Boom, 2020

ment of the second car or tourism-focused offers as a supplement to public travel. Basically, two types of Car-Sharing are distinguished: station-based and free-floating Car-Sharing. The difference is that in station-based car sharing, several fixed stations are defined by the provider where the vehicles can be both picked up and dropped off, whereas in free-floating car sharing, the vehicle can be parked anywhere within a zone. The station-based approach is most widespread only in some metropolitan areas is free-floating car sharing offered.



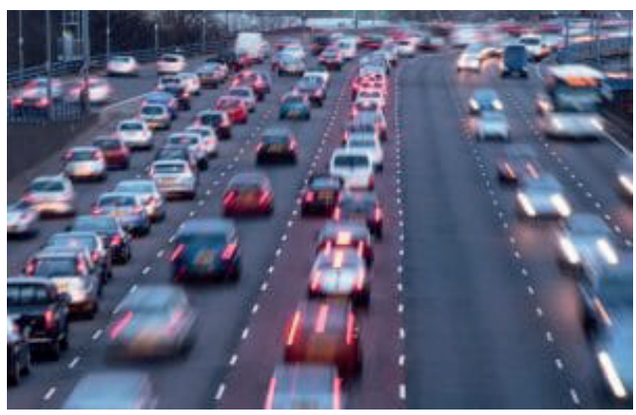
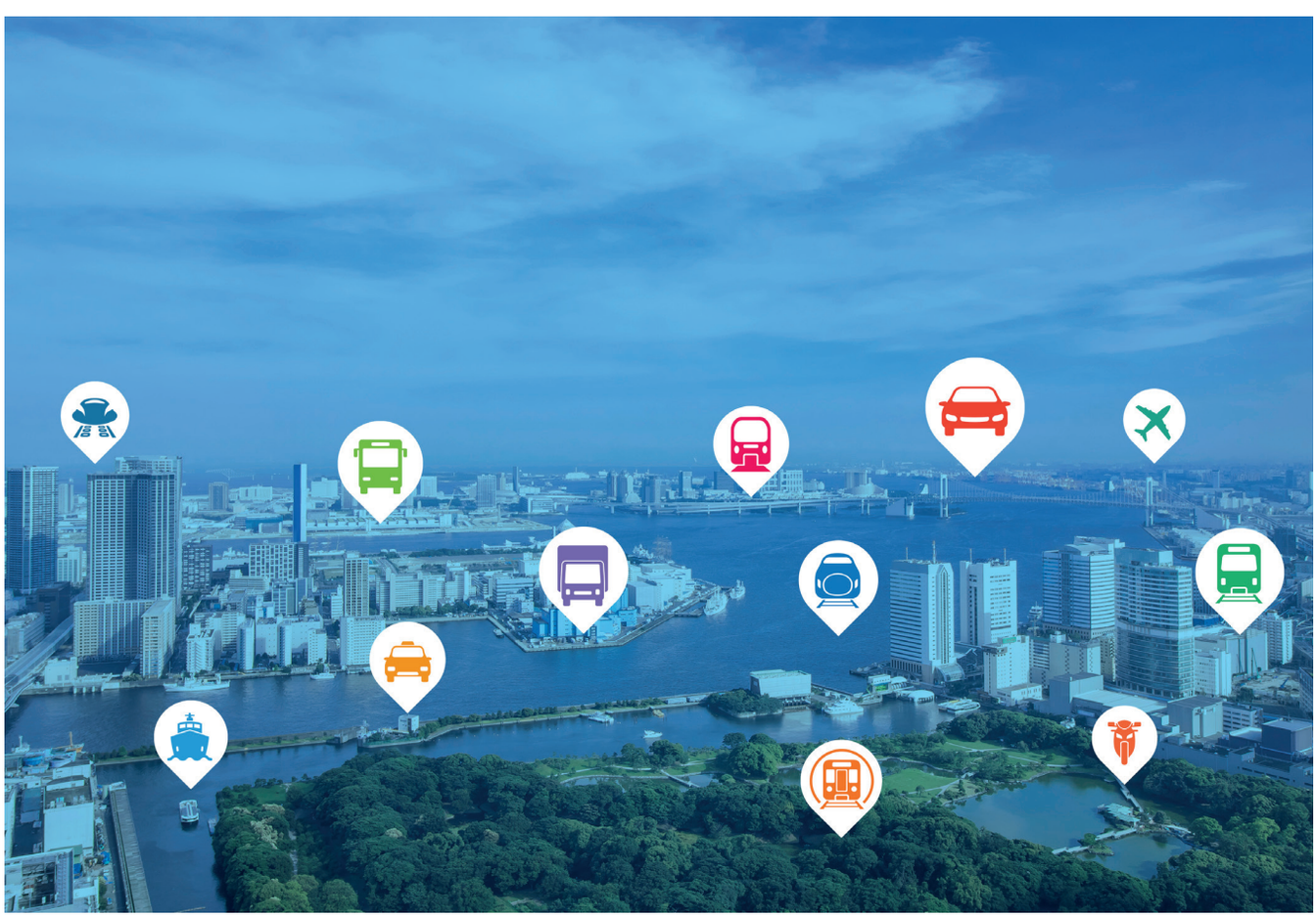


Figure (left) : Occupancy rate of cars over time and number of kilometers driven



Examples of research projects on sustainable mobility

RISC Software GmbH is strongly concerned with future-proof and sustainable mobility and is working on several projects with the aim of developing sustainable, inexpensive and uncomplicated solutions for the current challenges in individual transport.
to develop.

EVIS.AT

The research project EVIS.AT (Real-Time Traffic Information Road Austria) and the traffic data obtained from it serve as the basis for the other projects mentioned. In EVIS.AT several fleets (truck, car, a combination of truck and car, emergency vehicle, cab) were acquired for a data collection and route sections were equipped with sensor technology (VDL/counting loops, Bluetooth) for traffic counting. The aim of this project is to provide real-time traffic situation information for the region of Upper Austria. This traffic situation is also an integral part of the Austria-wide traffic situation of VAO.

LisiGo

The app LisiGo is the traffic jam and news portal of OÖNachrichten. It is used to avoid traffic jams on the way to and from work by calculating an optimal route for a user-defined route based on the current traffic situation. In addition, a forecast for a trip for the next 30 minutes can also be calculated to compare the traffic situation and find the optimal time for the trip. LisiGo thus contributes to the reduction of traffic jams or stop-and-go traffic and subsequently to the reduction of CO2 emissions.

DOMINO

The lead project DOMINO (Hub for Intermodal Mobility Services and Technologies) aims to enable the design of a sustainable, end-to-end and publicly accessible mobility management "Mobility as a Service" (MaaS) in Austria. Three different pilot regions (Upper Austria; Lower Austria, Salzburg) will be set up to develop, test and finally integrate new mobility services. The laboratory environment in the pilot region Upper Austria serves as a basis for the optimization of daily commuter traffic. This optimization will be achieved by increasing the occupancy rates of vehicles (through a ride-sharing exchange), promoting the switch to public transportation, reducing congestion (CO2 emissions) in the Linz metropolitan area, and attracting employers in the Linz metropolitan area by improving mobility services. In the course of the project, RISC Software GmbH is developing a ride-sharing pool that aims to link the various ride-sharing solutions in order to establish ride-sharing as an integral part of the mobility mix. ◆

[6] KFZ Bestand – statistik.at

[7] VOL (Vorarlberg Online)

Data engineering – the solid basis for effective data utilization

– DI Paul Heinzlreiter
Senior Data Engineer in the Unit Logistics Informatics

The path of data from sources to the integrated data lake.

Data engineering integrates data from a wide variety of sources and makes it usable effectively. This makes it a prerequisite for effective data analysis, machine learning and artificial intelligence, especially in the Big Data area.

In recent years, the topic of extracting information from big data has become increasingly important for more and more businesses in a wide range of economic sectors. Examples of this are historical sales data that can be used to optimize the product range of online stores and sensor data from a production line that can help to increase the quality of products or replace machine parts in good time as part of preventive maintenance. In addition to the direct use of an integrated database in operational practice, it is precisely the topicality of the topics of artificial intelligence (AI) and machine learning (ML) with the promise of being able to continuously optimize a production process, for example, that represents a strong motivation.

However, when the process of information acquisition is considered in its entirety, it quickly becomes clear that AI and ML represent only the proverbial tip of the iceberg. These methods require large amounts of consistent and complete data sets, especially for the steps of model training and model validation. Such data sets can be generated, for example, by sensor networks or by sensors in production.

The transfer, storage and processing of this data in order to make it effectively usable is the central task of data engineering. This is independent of whether the goal is company-wide and effective reporting, data science to improve the production process, or AI. A solid data basis is necessary in all cases. The integration of data into a common database can additionally form a reliable ground truth for a wide variety of use cases in the company: For effective day-to-day business, for strategic planning based on solid data and facts, or for model training in the field of AI.

Delimitation

At the top of the pyramid are the activities of Data Science, which are based on integrated and cleansed data sets. These can then be used to train ML models, for example. The levels colored in blue represent the data engineering activities, with the focus on the move, store and transform, explore levels. While the levels above with AI, deep learning, and ML are the domain of data scientists, activities such as data labeling and data aggregation are borderline areas that can be performed by data scientists or data engineers depending on the exact task and personnel availability.

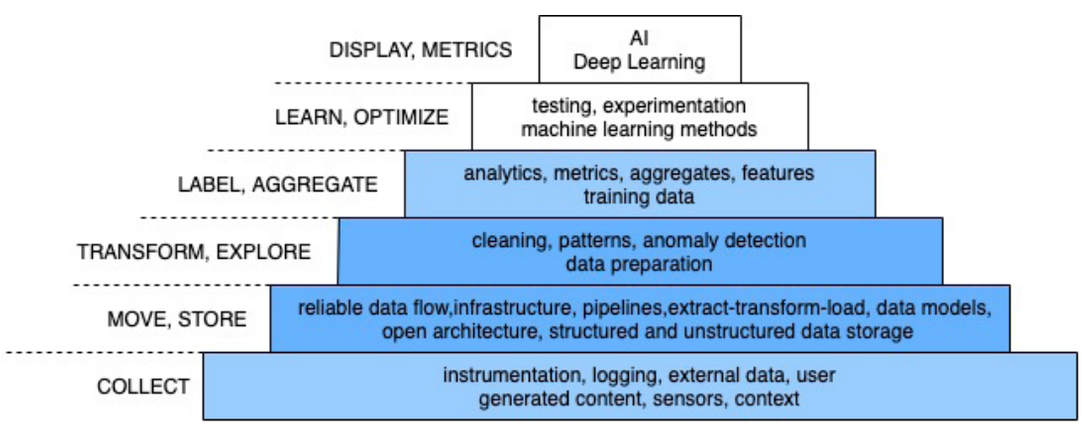


Figure 1: Data-Science Hierarchy-of-Needs [1]

[1] Blasch, Erik & Sung, James & Nguyen, Tao & Daniel, Chandra & Mason, Alisa. (2019). Artificial Intelligence Strategies for National Security and Safety Standards.
[2] Abraham Maslow: A Theory of Human Motivation. In Psychological Review. 1943, Vol. 50 #4, Seite 370–396.

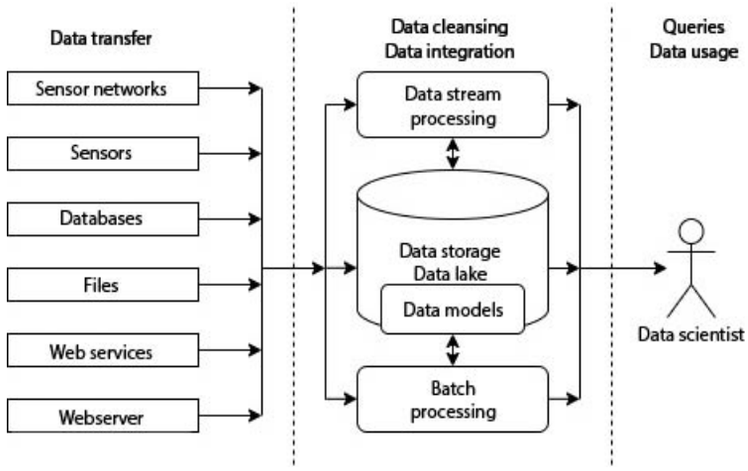


Figure 2: Data cleansing and integration

The activities of data collection at the base of the pyramid fall only partially within the scope of data engineering in that it usually takes the data at a defined interface - via files, external databases or a network protocol. This is also due to the fact that data engineering is a sub-field of computer science or software engineering, and thus does not usually deal with topics such as building or operating data collection hardware such as sensors.

Data cleansing and integration

As part of the data engineering process, the raw data is prepared over several steps after transfer and finally stored in the data store in a consistent and fully prepared form:

1. Data cleanup
2. Data integration
3. Data transformation

These forms of data transformations are carried out step by step and sequentially. The technical implementation can take the form of data stream processing - the consecutive processing of many small data packets - or batch processing for the entire data set simultaneously. An appropriately dimensioned data store - the data lake - makes it possible to persist the data in various states during its processing.

Data cleansing includes, for example, checking the read-in data lines for completeness and syntactic correctness. Data errors such as incorrect sensor values can also be detected by predefined rules in this step.

If these criteria are violated, the following options are available, depending on the application:

- Improve raw data quality: If the raw data can be subsequently delivered in improved quality, these replace the faulty data
- Discard data: Incorrect data can be discarded, for example, if the data set is to be used for training purposes in ML and sufficient correct data is available.
- Automatically correct errors during import: For example, if the data can be obtained from an additional data source, errors can be corrected during data integration.

In practice, discarding the incorrect data is the easiest solution to implement. However, if every single data point can have relevance for the planned evaluations, erroneous data must be corrected if possible. This case can occur, for example, during quality assessment in production, when the production data for a defective workpiece is incorrect due to a sensor error. The correction of data can either be done manually by domain experts, or the correct data can be supplied at a later point in time.

The involvement of domain experts is central here, because on the one hand they know the criteria for the correctness of the data, such as sensor values, and on the other hand they know how to deal with incorrect or incomplete data.

Data integration deals with the automated linking of data from different data sources. Depending on the application domains and the type of data, data linking can be done by different methods such as:

- Unique identifiers, analog to foreign keys in relational model
- Geographical or temporal proximity
- Domain-specific interrelationships such as sequences in manufacturing processes or in production lines

After the data cleansing and data integration steps, data engineers can provide a dataset suitable for further use by data scientists. The data transformation step mentioned above refers to ongoing adjustments to the data model to improve the performance of queries by Data Scientists.

Data storage and data modeling for Big Data

The cleaned and integrated data can be stored in a suitable data storage solution. In the application area of Industry 4.0, for example, data is generated continuously by sensors, which often leads to data volumes in the terabyte range within months. Such data volumes are often no longer manageable with a classic relational database. Although there are scalable databases available on the market that use the relational model, these are not an option for many implementation projects - especially in the SME sector - due to their high licensing costs.

As an alternative, horizontally scalable NoSQL systems are available, the term being an abbreviation for "Not only SQL". This term covers data stores that use non-relational data models. The property of horizontal scalability refers to the possibility of expanding such systems by integrating additional hardware for basically unlimited data volumes. Typical representatives of NoSQL systems are also often subject to liberal licensing models such as the Apache license and can thus also be used commercially without license costs. In addition, these systems do not place any special requirements on

the hardware used, which further reduces the acquisition costs of such systems. Thus, NoSQL systems such as Apache Hadoop and related technologies represent a cost-effective way of executing queries on data volumes in the terabyte range.

Particularly in the Big Data area, the selection of a suitable NoSQL database and a suitable data model is of central importance because both aspects are central to the performance of the overall system. This refers both to the input of data and to queries against the NoSQL system.

The selection of the technology to be used as well as the data model design is clearly driven by the system requirements:

- What data volumes and data rates need to be imported?
- Which queries and evaluations are to be performed with the data?
- What are the performance requirements for the queries? Is it a real-time system?

The central question is, for example, whether the system should only support fixed queries or - for example, using SQL - allow flexible queries.
In the context of technology selection, a distinction can be made, for example, as to whether data is always accessed via a known key or whether queries are also made on the values of other attributes. In the first case, a system with the semantics of a distributed hash map, such as Apache HBase, is suitable, while in the other case, for example, an in-memory analysis solution such as Apache Spark is suitable. If the use of the data is primarily aimed at the links between data, the use of a graph database should be considered.

In a Big Data system, data is stored denormalized for performance reasons, i.e., all data relevant to a query result should be stored together. The reason for this is that performing joins is very resource-intensive and time-consuming. Therefore, the planned queries are central to the design of the data model. For example, the attributes that mainly appear as parameters in the queries should be used as key attributes. This is also the reason why the data model often has to be extended when new queries are added to ensure their effective execution, and thus data engineering activities are continuously required even after the data has been introduced. ◆

Use Case 1: Corporate data integration

Data from different sources can be merged and used effectively in an integrated data model

Use Case 2: Data preparation for AI / ML

Data engineering methods can be used to provide a large amount of consistent and complete training data for AI and ML

Use Case 3: Transformation of the data model to improve data understanding

Data engineering can significantly increase data understanding by better adapting the data model to the use case. An example could be the introduction of a graph database.

Use Case 4: Improved (faster) data usage

Data engineering can help significantly speed up interactive queries by adapting the data storage and data model.

Agile & test-driven: Focus on the customer

– Yvonne Marneth, BSc
Software Developer in the Unit Domain-specific Applications

How test-driven-development benefits both customers and the for both customers and the development team

Agile methods are used to actively involve customers in the development process. At regular intervals, they get the opportunity to check the progress and direction of their product with their feedback. This results in a valuable product that can be delivered quickly. With the use of test-driven development, this core concept is also transferred to technical development.

What is Test-Driven Development?

Test-Driven-Development (TDD) is a process model in software development that is based on testing software automatically using various test methods. Traditionally, a feature is first implemented and then tested. Test-Driven-Development reverses this process: Here, a test is first written for a new feature and then the associated logic is implemented incrementally until the test runs successfully. Then the code is revised to improve it and make it more understandable. Finally, it is integrated into the code base, where it is checked against all existing tests. Only then is the feature considered successfully implemented.

This approach can seem very unintuitive and time-consuming at first. In fact, however, it brings many advantages - especially when integrated into an agile work mode - and can even make the development process more effective in the long term.

Quality improvement

Probably the most obvious consequence of Test-Driven Development is the higher code quality that results from the measures taken. This increase becomes measurable via various quality parameters such as the number of "code smells" (unpleasantness in the code), bugs, etc.. These parameters are also summarized under the term "technical debt". Also these measured values can be determined automatically and give an overview of the general condition of a software product. The goal of TDD here is to keep an eye on the technical debt and to keep it as low as possible. This results in significantly lower costs for changes and enhancements later on.

The feedback loop can be used to ensure that the code behaves correctly, but also to check that undesirable behavior does not occur. Developers are therefore not only concerned with the optimal case, but also with possible pitfalls that would otherwise only become apparent in actual user tests. The time-consuming testing phase with users can thus be shortened considerably, developers deal with the cases directly and do not first try to recreate and understand a reported error scenario. In addition, they handle bugs while the code is still "fresh" in their minds and do not have to think their way back into it.

Code reviews are a good way to give colleagues an insight into the written code during the revision phase. On the one hand, this prevents the formation of so-called "knowledge silos". This means that only a single team member has knowledge about a particular code component. This dependency is to be avoided. On the other hand the code quality increases, since constructive feedback is brought in. Code reviews can also be used to introduce new employees to a project or to distribute knowledge within the team. The process can also improve the division of labor and team dynamics in the long term.

If users still find further errors, existing tests support the reading of the code and thus the quick elimination of the error. New tests then immediately ensure that a specific error cannot creep in again in the future.

Customers have the advantage that the life expectancy of their software is increased and users experience fewer errors. This reduces the costs for support. However, the higher quality is only a partial goal of this process.

Test-Driven-Development in an agile context

In agile process models such as Scrum or Kanban, an iterative development cycle is depicted in contrast to traditional methods. This is designed to deliver a usable product to customers as quickly as possible and to obtain feedback as early as possible. In this way, problems can be addressed directly in the further development process and the direction can be corrected. Test-Driven-Development transfers this cyclical workflow to the technical development level. As shown in Figure 1, the cycle consists of continuously writing tests, implementing functions and revising the written code. Working in cycles selectively reduces the complexity of a task and helps maintain focus.

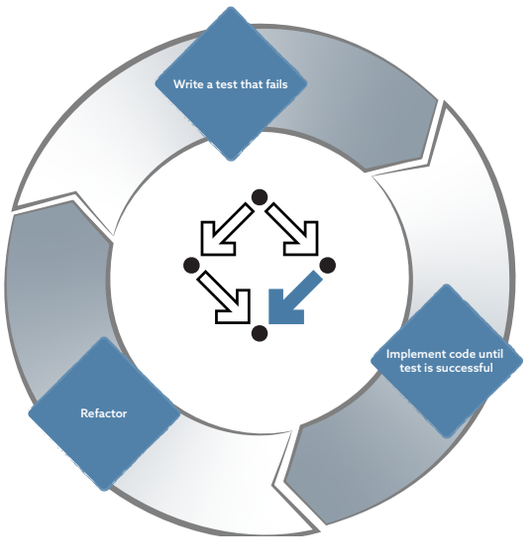


Figure 1: Test-Driven Development describes a cyclical development process in which tests are created for code even before it is implemented.

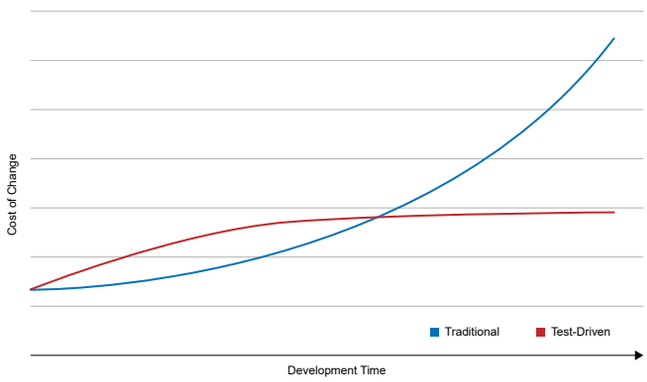


Figure 2: The goal of Test-Driven-Development is to sustainably reduce the Cost of Change compared to a traditional workflow.

In agile processes, customer requirements can change again and again. Therefore, flexibility and ease of change in the code base must be taken into account as early as the development stage. This, in turn, is only possible if the code base is lean and focused, which is ensured by constant revision. High technical debt means that changes can lead to regressions, which are sometimes time-consuming and expensive to fix, or even only become apparent on the production system and thus cause additional support costs. The goal of TDD is shown in Fig. 2.

Test coverage, successful test runs and reviews can help to formulate a meaningful "Definition of Done" in an agile environment and to verify its fulfillment objectively. A good Definition of Done is necessary to create clear expectations for the completion of a feature between the development team and the customer. It is usually not limited to the pure development time a developer needs until a feature is visible, but until it meets all mutually agreed requirements. These can be of a functional, qualitative and security nature. If there is no common understanding between the development team and the customer, functions that have actually been completed often have to be reworked. This can disrupt the workflow and ultimately lead to functions taking a very long time to implement.

Test-Driven Development builds on agile values and helps to exploit the full potential of this way of working. Initially, the cycle time in a team can increase while the developers learn to use the new methods. Over time, as new concepts become routine and experience is built up, the effort becomes negligible and pays off with a lower cost of change. The controlled increase in software quality is ultimately the only reliable method for accelerating the development process in the long term.

| Conclusion

Agile workflows and test-driven development are both designed to obtain and respond to feedback quickly. Agile process models focus primarily on the organizational side. To ensure that this can also function smoothly from a technical point of view, Test-Driven-Development introduces the agile values at the development level. As with the implementation of the agile framework, it is essential here to adapt Test-Driven-Development to the circumstances, the team, the project context, etc., so that it is effective and brings benefits for both customers and the development team. ◆

Data quality: From information flow to information content

– Sandra Wartner, MSc and Christina Hess, MSc
Data Scientists in the Unit Logistics Informatics

Why clean data (quality) management pays off

Making decisions is not always easy - especially when they are relevant to the fundamental direction of the company and can thus influence a far-reaching corporate structure. This makes it all the more important to know as many influencing factors as possible in the decision-making process, to quantify facts and to incorporate them directly (instead of making assumptions) in order to minimize potential risks and achieve continuous improvements in the corporate strategy. A possible quick-win for companies can be derived from corporate data: future-oriented data quality management - a process that unfortunately often receives far too little attention. Why the existence of large data streams is usually not enough, what decisive role the state of the stored data plays in data analysis and decision making, how to recognize good data quality and why this can also become important for your company, we explain here.

What is the cost of bad data?

When shopping at the supermarket, we look for the organic seal of approval and product regionality; when buying new clothing, the material should be made from renewable raw materials and under no circumstances produced by child labor; and the electricity provider is selected according to criteria such as cleanliness and transparency - because we know what influence our decisions can have. So why not stay true to the principle of quality over quantity when it comes to data management?

In the age of Big Data, floods of information are generated every second, often serving as the basis for business decisions. According to a study by MIT [1], making the wrong decisions can cost up to 25% of sales. In addition to the financial loss, unnecessarily high resource input or additional effort is required to correct the resulting errors and correct the data, and the proportion of satisfied customers as well as the trust in the value of the data decreases. Google is not the only company that has had to deal with the drastic consequences of errors in its data, for example with its Google Maps product [2]. Address information at the wrong location even led to a demolition company accidentally razing the wrong house to the ground; incorrectly lowered kilometer information to the navigation destination via non-existent roads left drivers stranded in the desert or sights suddenly appeared in the wrong places. Also NASA had to watch on 23.09.1999 how the Mars Climate Orbiter and with it more than 120 million $ burned up during the approach to Mars - the reason: a unit error [3]. Even if the effects of poor data quality may not be quite as far-reaching as those of the major players, the topic of data quality nevertheless affects every company.

From a business perspective, data quality is not an IT problem, but a business problem. This usually results from the fact that business professionals are not aware of the importance of data quality, or are not aware of it enough, and that data quality management is also successively weak or missing altogether. Linking data quality practices with business requirements helps to identify and eliminate the causes of quality problems, to reduce error rates and costs, and ultimately to make better decisions.

Despite the above criteria, it is not easy to describe good or bad data quality on the basis of more concrete characteristics, since data exists in a wide variety of structures that differ greatly in their properties. The collected data stock in the company is composed of different data depending on the degree of structuring:

- Structured data is information that follows a predefined format or structure (usually in tabular form) and may even be sorted systematically. This makes them particularly suitable for search queries for specific parts of information such as a specific date, postal code or name.
- Unstructured data, on the other hand, is present in a non-normalized, unidentifiable data structure, making it difficult to process and analyze. This includes, for example, images, audio files, videos or text.
- Semi-structured data follows a basic structure that includes both structured and unstructured data. A classic example of this is e-mails, for which, among other things, the sender, recipient, and subject must be specified in the message header, but the content of the message consists of arbitrary, unstructured text.

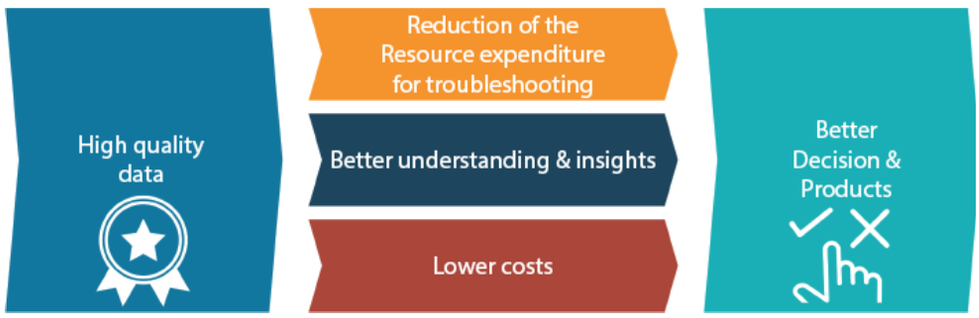


Figure 1: From data quality to sustainable decisions and products

Data basis and data quality - What's behind it?

There are already many definitions of the term data quality, but a general statement about it can only be made to a limited extent, since good data quality is usually defined on a domain-specific basis. A large data set alone (quantity) is no indication that the data is valuable. The decisive factor for the actual usefulness of the data in the company is above all whether it correctly reflects reality (quality) and whether the data is suitable for the intended use case.

There are various general approaches and guidelines for assessing the quality of data. Often, good data quality is understood very narrowly as the correctness of content, ignoring other important aspects such as trustworthiness, availability, or usability. For example, Cai and Zhu (2015) [4] define the data quality criteria Availability, Relevance, Usability, Reliability, and Presentation Quality shown in Figure 2. In the following, we discuss some relevant points for the implementation of data-driven projects based on these criteria.

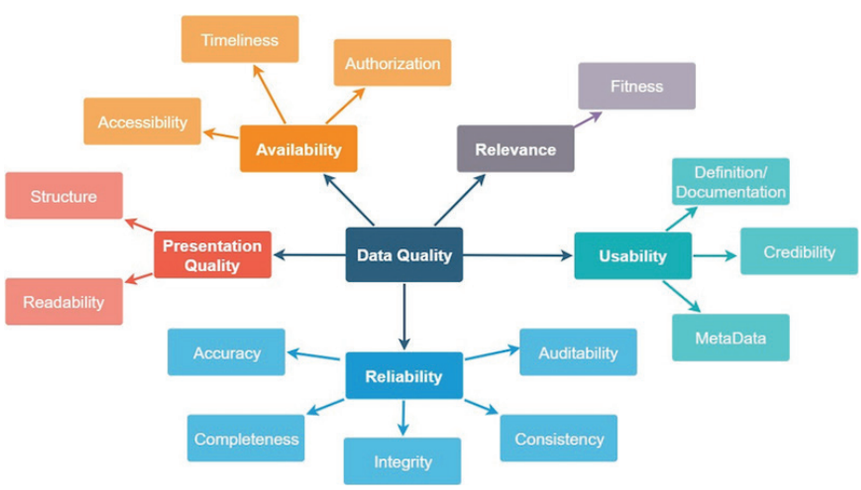


Figure 2: Data quality criteria according to Cai and Zhu (2015).

Relevance

In data analysis projects, it is particularly important at the beginning to consciously consider whether the data on which the project is to be based is suitable for the desired use case. In AI projects, it is therefore always important to be able to answer the following question with "yes": Is the information needed to answer the question even available in its entirety in the data? If even domain experts cannot identify this information in the data, how is an AI supposed to learn the connections? After all, machine learning algorithms only work with the data provided to them and cannot generate or utilize any information that is not contained in the data.

Usability

Metadata is often needed to interpret data correctly and thus to be able to use it at all. Examples of this are coding, origin, or even time stamps. The origin can also provide information about the credibility of the data, for example. If the source of the data is not very trustworthy or if the data originates from human input, it may be worthwhile to check it carefully. Depending on the content of the data, good documentation is also important - if codes are used, for example, it may be important to know what they stand for in order to interpret them, or additional information may be needed to be able to read date values, time stamps or similar correctly.

Reliability

Probably the most important aspect in the evaluation of data and its quality is how correct and reliable it is. The decisive factor is whether the data is comprehensible, whether it is complete and whether there are any contradictions in it - simply whether the information it contains is correct. If you already have the feeling that you do not trust your own data and its accuracy, you will not trust analysis results or AI models trained on them either. In this context, accuracy is often a decisive factor: for one question, roughly rounded values may be sufficient, while in another case accuracy to several decimal places is essential.

Presentation quality
Logically, data must be able to be "processed" not only by computers, but above all by people. In many cases, therefore, they must be well structured or be able to be processed so that they are also readable and understandable for us humans.

Availability
At the latest, if one wants to actively use certain amounts of data and, for example, integrate them into an (AI) system, it must also be clarified who may access this data and when, and how this access is (technically) enabled. Above all, this is an important factor for the trustworthiness of resulting AI systems, this is an important factor. For real-time systems, timeliness is also crucial. After all, if you want to process data in real time (e.g., for monitoring production facilities), then it must be possible to access it quickly and it must be reliable.

How do you recognize poor data quality and how can these data deficiencies arise in the first place?

Poor data quality is usually not as inconspicuous as one might think. A closer look quickly reveals a wide variety of deficiencies, depending on the degree to which the data is structured. Let's be honest - are you already familiar with some of the cases shown in Figure 3? If not, take the opportunity and start looking for these conflict generators, because you will most likely find some of them. These data can take many shapes in practical application and manifest themselves in various problematic issues such as image damage or legal consequences, among others (Figure 4 shows just a few of the negative effects). The costs of poor data quality can also be far-reaching, as we have already made clear in "What do bad data cost?". But how can such problems arise in the first place? The root causes often lie in the lack of data management responsibilities or data quality management in the first place, but technical challenges can also cause problems.

These errors often creep in over time. Particularly prone to errors are different data collection processes and, subsequently, the merging of data from a wide variety of systems or databases. Human input also produces errors (e.g. typing errors, confusion of input fields). Another problem lies in data aging: Especially when changes in data collection or recording take place (e.g. missing sensor data during machine changeover, lack of accuracy, too small/too large sampling rate, lack of know-how, changing requirements on the database), problems occur. Further risk factors are the often missing documentation and the faulty versioning of the data.

In practice, perfect data quality is usually a utopian notion, which is also shaped by many influencing factors that cannot be controlled or are difficult to control. However, this should not take away anyone's hope, because: Most of the time, even small measures achieve a big effect.

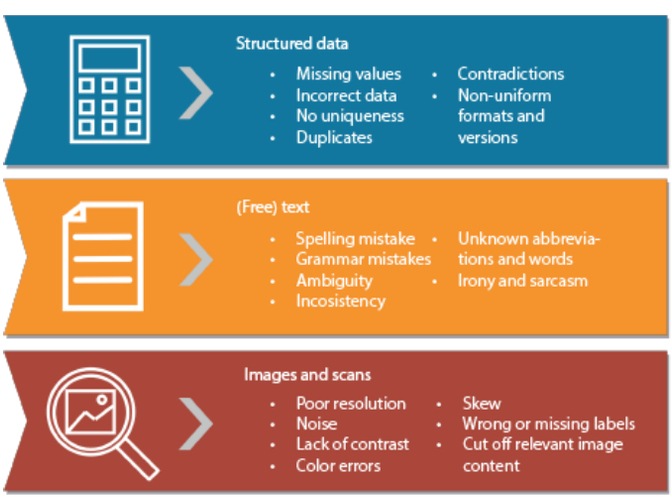


Figure 3: Examples of poor data quality by Degree of structuring of the data

Figure 4: Negative examples from practice

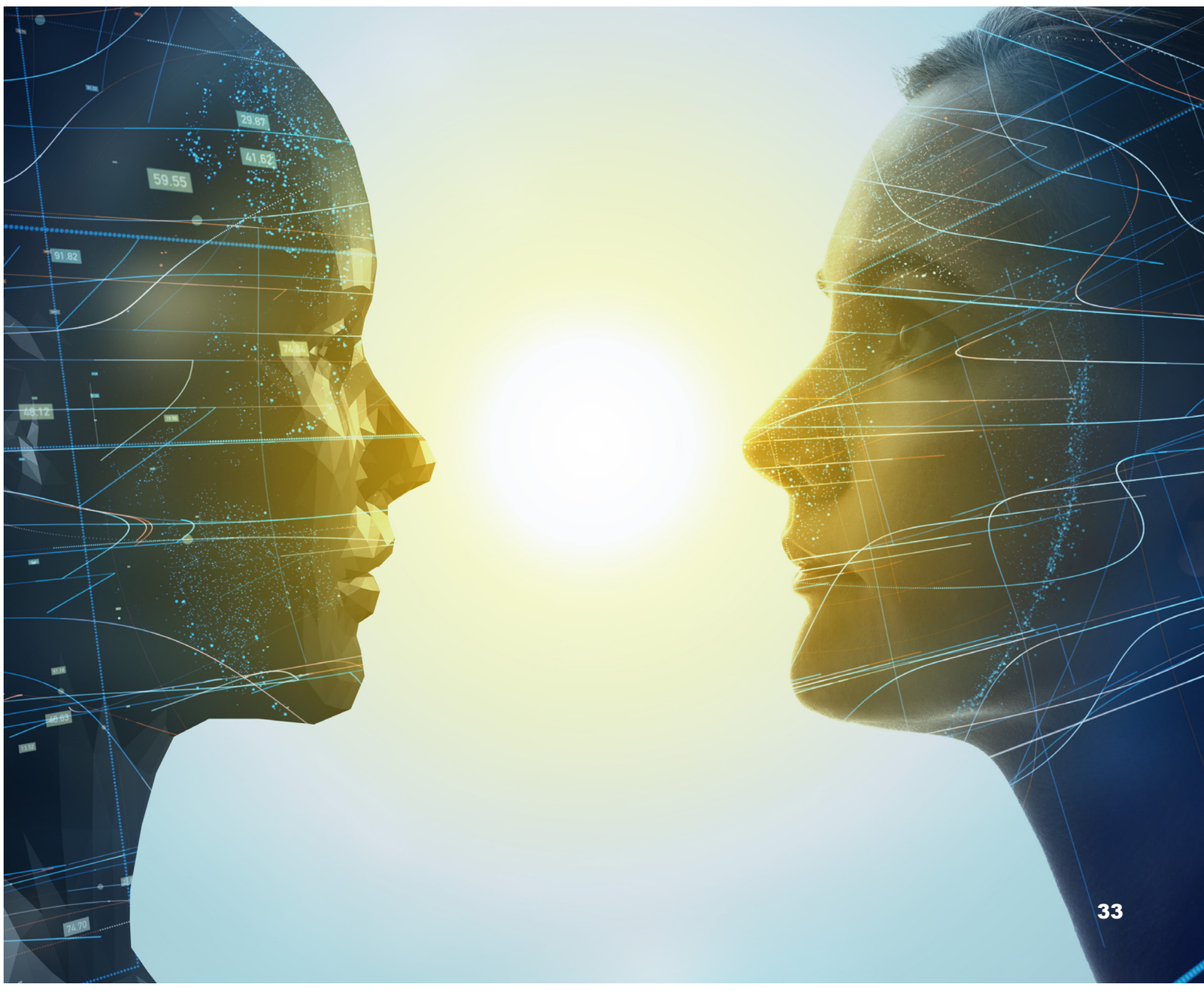
[1] https://sloanreview.mit.edu/article/seizing-opportunity-in-data-quality/
[2] https://www.googlewatchblog.de/2019/07/google-maps-fehler-katastrophen/
[3] http://edition.cnn.com/TECH/space/9909/30/mars.metric/
[4] Cai and Zhu (2015): The Challenges of Data Quality and Data Quality Assessment in the Big Data Era

Data as fuel for machine learning models

It is not only in classical data analysis that special attention should be paid to the data in order to obtain the maximum informative value of the results. Especially in the field of artificial intelligence (AI), the available database plays a decisive role and can help a project to a successful conclusion or condemn it to failure. By using AI - specifically machine learning (ML) - frequently repetitive processes can be intelligently automated. Examples are the search for similar data, the derivation of patterns or the detection of outliers or anomalies. It is also essential to have a good understanding of data (domain expertise) in order to identify potential influencing factors and to be able to control them. Well-known principles such as "decisions are no better than the data on which they're based" and the classic GIGO idea (garbage in, garbage out) clearly underline how essential the necessary database is for the learning process of ML models. Only if the data basis is representative and reflects reality as truthfully as possible can the model learn to generalize and successively make the right decisions.

Data quality as a success factor

Data quality should definitely be the top priority for data-based work. Furthermore, the awareness of the relevance of good data quality should be created or sharpened in order to be able to achieve positive effects throughout the company, to reduce costs and to be able to use freed-up resources more efficiently for the really important activities. Our conclusion: Good data quality management saves more than it costs, enables the use of new methods and technologies and helps to make sustainable decisions. ◆



Working with Fortran in 2020: Do's and Don'ts

– DI Dr. Christoph Hofer
Software Engineer in the Unit Industrial Software Applications

An etiquette guide for Fortran developers

Fortran is one of the oldest programming languages. The name is composed of "FORmel TRANslator". For many software developers Fortran is the archetype for an old, ponderous, limited and difficult to understand programming language, with which one would best have nothing to do. For the old versions of Fortran, this prejudice may indeed be true. However, Fortran has changed a lot in its long history, so in its "modern" variant (such as Fortran 2003) the language has a much worse reputation than it deserves. The typical use case for Fortran as a programming language is computationally intensive, numerical simulations such as weather forecasts, flow simulations or stability calculations.

From old to new

Fortran is considered as the first ever realized higher programming language and was developed in the years 1954 - 1957 by IBM (FORTRAN I). The scope of the language was still very limited, for example, there were only integers and reals (floating point numbers) as data types and no functions yet. In the following years new improved and more extensive Fortran versions were developed (FORTRAN II, FORTRAN III, FORTAN 66). The next big update Fortran got in 1977 (FORTRAN 77). Due to new features in the language, this version became very popular and thus quickly became "the" Fortran. Even today, when talking about Fortran code, mainly FORTRAN 77 code is meant, what also explains the many prejudices against the language. Since then, there have been several more updates to the language, bringing it up to modern programming concepts and standards. Major milestones in the development were the updates to Fortran 90 and Fortran 2003, which, in addition to the change of name (FORTRAN →Fortran) added common concepts such as free source file formats, modules, operator overloading, derived data types, pointers and object-oriented programming to the programming language. In addition to this, Fortran 95 and Fortran 2008 were each minor updates to the language. The latest version of the Fortran standard is Fortran 2018, although no compiler vendor yet supports all features.

Dos & Don'ts in Fortran

Due to the long development history of Fortran and in order to maintain compatibility with legacy code, there are numerous obsolete and sometimes obscure language features in current Fortran compilers. A comprehensive collection of good and bad coding practices is beyond the scope of this article. Nevertheless, we would like to present some common legacy features that can be found in legacy code, as well as selected opportunities offered by new Fortran standards. For a comprehensive list of dos and don'ts, please refer to [1].

Don't use common block and equivalent statement

In Fortran 77 and earlier, it was common for different variables to refer to the same memory address using common block and equivalent statement. These expressions were used to share information between subroutines or to reuse (expensive) memory for temporary variables. In the meantime, these expressions have been declared deprecated and should no longer be used. To share data between program parts, modules should be used and memory should be allocated and deallocated dynamically as needed.

Avoid using GOTO

No other expression is as rooted in Fortran as the GOTO. In old programs you can often find an excessive use of GOTOs, partly due to the lack of alternative constructs. Over the time different variants of the GOTO developed, like the computed GOTO statement or the assigned GOTO statement. Also there were variants for the handling of loops or IF-statements, which worked with jumps to corresponding labels. In modern Fortran code, all these variants of GOTOs should be avoided if possible, and replaced by IF and SELECT case (= switch). A notable exception to the need for GOTOs in Fortran code is error management, since exceptions do not exist in Fortran.

Avoid SAVE attributes

The SAVE attribute allows variables to retain their value in repeated function calls. Especially in conjunction with parallelization, this can lead to hard-to-find bugs and dataraces. The SAVE attribute can be safely used with variables that always have the same value on each function call to gain some performance. In all other cases it should be avoided. A particularly sneaky "feature" of Fortran is that all variables that are initialized the same when they are declared are automatically given an implicit SAVE attribute.

Use implicit none

A concept from old Fortran standards was that undeclared variables are automatically declared as REAL - except those starting with the letter i, j, k, l, m or n, which are declared as INTEGER. This concept is very error-prone, mainly because the compiler does not give an error message if undeclared variables are used, e.g. by a typo. Because of this concept the following joke about Fortran has become common: Fortran is the only language where "God is Real" applies. This automatic variable declaration can be disabled using IMPLICIT NONE for the current scope and it is "good practice" to implement it throughout the program code.

Make use of derived data types and classes

With Fortran 90, the language began to develop further in the direction of object-oriented programming. With this standard User-Defined Datatypes were introduced, which allowed for the first time to use reusable structures of logically related data. Also, the concept of generics was added, so that the same function name can be used with different types (but still the function must be programmed for each type). With the Fortran 2003 standard the object-oriented programming was again forced. At the latest since this time it should be tried to encapsulate data and logic in meaningful classes and to let program parts interact over well-defined interfaces.

Use the module system

Fortran 90 also introduced a new form of program organization, namely the module system. A module consists of a set of declarations of data, functions and function interfaces, which can then be used and made visible in other program parts. In addition, modules offer the possibility to restrict the access of the contained functions/data by means of PRIVATE/PUBLIC. Since the Fortran 2008 standard, there are submodules, which now allow the programmer to outsource the program code into a separate submodule. The necessity for this is on the one hand to avoid very large and unclear modules, to have the interface of the module clearly visible, on the other hand also to reduce the recompilation times.

Don't rely on short-circuit evaluation

Very many programming languages evaluate only the first expression in a logical combination of two expressions, if the result is already determined by this expression. This procedure is called short-circuit evaluation. In Fortran, however, it is up to the compiler whether short-circuit evaluation is used, i.e. in the Fortran standard this is not forbidden, but also not prescribed. A typical use case for the necessity of short-circuit evaluation would be the query on the right-hand side.

```
IF (PRESENT(x) .AND. x > 0 ) THEN
      do something with x
END IF
```

In Fortran there is the possibility of optional arguments, i.e. parameters of a function which do not have to be passed. With the help of the function PRESENT(x) it can be checked whether the parameter x was passed. In the example on the right, a query is made after the check whether x is greater than 0. If x is not passed, then by short-circuit evaluation the query x>0 would not be made any more, since already the first condition is not fulfilled. However, the program would crash at this point due to the possible failure of short-circuit evaluation. The correct notation is the splitting on two single conditions as shown in the example on the right.

```
IF (PRESENT(x)) THEN
     IF(x > 0 ) THEN
          do something with x
     END IF
END IF
```

Other typical cases are queries whether a pointer is assigned to a memory address or whether a mathematical operation is allowed with the input values (division, root). ◆

[1] S. Chapman, Fortran for scientists & engineers, McGraw-Hill Education, 4. Edition, 2017

Decision support for industry and business: Optimization has to be learned.

– DI Manuel Schlenkrich
Mathematical Optimization Engineer in the Unit Logistics Informatics

Artificial intelligence and optimization - the best of both worlds

“Learning by doing“, “Practice makes perfect“ or “You learn from mistakes“ - why these phrases are not only motivating for us humans, but also help mathematical algorithms to succeed, you will learn in this article. For complex problems in business and industry, optimization models and solution algorithms are used every day to make difficult decisions. In doing so, most decision makers are confronted with a highly dynamic environment, uncertain forecasts and countless variables. In order to deal with these challenges and to save enormous computational effort, methods are being developed that work with artificial intelligence. Classical optimization methods are combined with machine learning to meet the high demands - because one thing can be said: Optimization has to be learned.

Smart algorithms for complex tasks

Being successful means making good decisions. But when is a decision optimal and, above all, how do you find it? How many pairs of skis should a sporting goods manufacturer produce when their demand can only be estimated? How many shares of stock should an investor buy for her portfolio when the future stock price is highly uncertain? At which locations should COVID-19 test centers be set up to provide easy access to as many people as possible? As different as these applications may sound, they all have one thing in common: Optimal decisions are sought. Proven decision support tools are mathematical optimization models that break down real-world problems to their essential features. For these models, best possible decisions can then be found and applied to the real problems by using solution methods.

However, the demands of business and industry on these mathematical models are becoming ever higher, the problems increasingly complex, and aspects such as fluctuating parameters and dynamic environments ever more relevant. This development leads to the fact that classical exact solution methods would need hours, days or even weeks to calculate decisions under realistic conditions. At the same time, the availability of a large amount of data is increasing, mainly driven by technological progress and the advancing digitalization of industrial processes. Thus, a new trend deals with the fusion of methods of classical optimization with machine learning approaches to efficient data-driven solution methods. These methods are able to use artificial intelligence techniques to find patterns in problem properties, simplify complex relationships and learn promising solution behavior. The goal here is to combine the best of the worlds of optimization and machine learning to find good solutions in a reasonable amount of time, even for highly complex decision problems.

Optimization meets AI: combining the best of both worlds

For decision-making, a mathematical model is created as a representation of reality, in which an objective function has to be optimized while complying with various restrictions. There are two different approaches to find an optimal solution for a model. On the one hand, exact solution methods can be applied, which can also guarantee their global optimality if a solution is found. However, these solution methods usually require enormous computational effort and often do not provide a solution in a reasonable time, especially for large and realistic models. Besides the exact methods, there is also a group of heuristic solution methods. These are algorithms that have been tailored to concrete problems in order to find good solutions in a short time. However, no statement can be made whether the solution is a global optimum. The so-called metaheuristics are superordinate to the heuristic methods. These describe a general algorithmic procedure, which is applicable to a multiplicity of problem definitions.

Machine learning approaches can now be combined with both types of solution methods. For exact methods, this is mostly to reduce computational effort by having the artificial intelligence find promising regions of the solution space, determine efficient execution orders within the algorithm, or compute the objective function values faster. Heuristic methods are predominantly concerned with generating better solution qualities through the use of learning algorithms. In particular, tasks such as parameter tuning, algorithm selection or operator management are to be taken over by artificial intelligence.

How is learning done?

If one speaks of methods of optimization which contain a learning aspect, then basically two types of learning methods can be distinguished. These two types have their origin in two different motivations for applying such algorithms. In some areas, experts already have extensive theoretical or empirical knowledge about the decision environment, which could be represented by an exact optimization method. The users now want to use the learning algorithm to approximate this known knowledge and thus save considerable computational effort. The goal is to learn a behavioral rule or "policy" (π_{ml}) that imitates the decisions of the experts (π_{expert}) and thus achieves similar results. In this type of learning, the algorithm does not try to optimize the quality of the results, but minimizes the discrepancy between the decisions made and the expert*s demonstrations. However, it also happens that there is not enough information about the decision environment and new decision strategies should be developed. In this case, the learning algorithm is trained in a trial-and-error setting to maximize a so-called reward signal. In this type of learning, the algorithm independently explores the decision space and gains new information about the quality of decisions made with each exploration step.

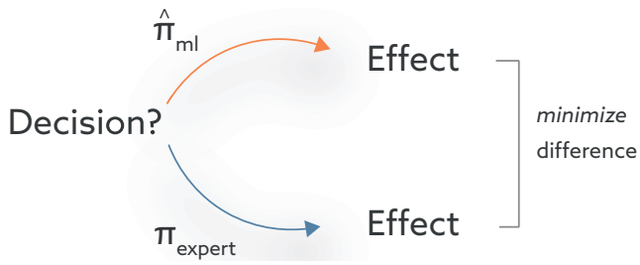


Figure 1: Learning by demonstration

Figure 2: Learning through experience

When to learn?

The methods presented can be divided not only in terms of how they learn, but also in terms of the structure in which the learning process is incorporated into the algorithm. Machine learning can be used in advance to configure the optimization method, for example by learning promising parameters or execution sequences within the algorithm. Another option is to alternate machine learning and the optimization method by iterative iteratively. In this case, the optimization algorithm continuously feeds information about the current solution quality to the learning part, while the learning part in turn derives and feeds back new promising solution domains from the information. In the third variant, the learning part replaces the actual optimization procedure and already determines a finished solution to a given problem. In this case, we also speak of end-to-end learning.

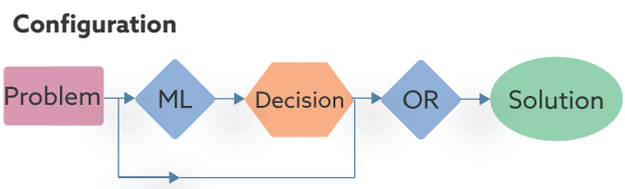


Figure 3: Algorithm configuration by ML at the beginning

Figure 5: End-to-end learning

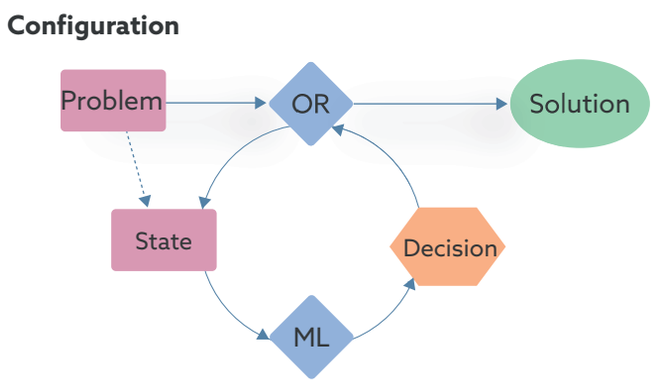


Figure 4: Iterative algorithm configuration by ML

What is learned?

After the question of how learning takes place has been clarified, the much more important question remains unanswered: What exactly is learned? There are different approaches to this question, depending on the properties of the properties of the optimization method used and problem-specific factors.

Parameter tuning through ML

Metaheuristics usually contain a number of parameters that have a significant impact on performance. Machine learning can be used to learn these parameters for a specific class of problem or for individual instances of a problem. Techniques of linear or logistic regression, neural networks or response surface methods are particularly used.

Target function evaluation by ML

For complex problems, evaluating the objective function requires a lot of computational effort. Machine learning can be used to create an approximation of the objective function and thus speed up the evaluation. Polynomial regression, neural networks or Markov fitness models are popular ML methods that can be used for this purpose.

Population and operator management through ML

Many metaheuristics (such as local search methods) use operators to generate new promising solutions starting from already generated solutions. In genetic algorithms, we also talk about populations that are modified by mutation and crossover operators. Often, the use of these operators is prescribed in advance by fixed rules based on the solution properties. However, these rules can also be continuously adapted and improved by machine learning. For example, inverse neural networks or classification algorithms from the field of symobolic learning methods are suitable for learning rules that do not repeat previous failed attempts and can explain why some operators are more suitable than others at this point.

Algorithm selection by ML

It may happen that a whole portfolio of different solution methods is available for the same problem class and one is interested in which of them provides the best performance. The algorithm selection problem describes exactly this situation, in which a solution method a is to be selected from the set of available methods A in such a way that the performance of a, applied to a problem x, is best possible among all methods in A. This selection problem is to be solved depending on the problem properties of the problem x. This selection problem is to be solved depending on the problem properties of problem x. Classification algorithms and neural networks are suitable to divide the available portfolio A into more or less promising methods based on the problem properties.

Determination of the execution order by ML

The branch-and-bound framework is a widely used exact solution method. The problem is broken down piece by piece into smaller subproblems ("branching") and can be represented in a tree structure ("branch and bound tree" / "search tree"), in which each node represents an incomplete solution of the overall problem. Lower bounds can then be computed for these nodes by relaxing the restrictions. At the same time, upper bounds can be found by heuristically solving the subproblems, and as soon as the lower bound is above the upper bound for a node in the search tree, the entire "branch" can be discarded, which in turn restricts the search space ("bounding"). The faster good upper bounds are found, the faster entire regions of the search space can be discarded, resulting in a significant performance improvement of the branch-and-bound algorithm. To obtain such upper bounds, various heuristics are used that attempt to generate an admissible good solution in each node. However, it is highly dependent on the particular problem and instance which existing heuristic yields the best results. It would be desirable to use the best heuristic in each case early and not waste time with worse heuristics beforehand. Typically, the execution order of heuristics is defined in advance, independent of the particular problem instance and incapable of responding to dynamic changes during the search run. A new approach improves the execution order of heuristics in a data-based manner and continuously adjusts it during the search run.

Summary

In an era where large amounts of data are collected in almost every process, it is obvious to take them into account in decision making and to support the optimization algorithms used for this purpose with learning components. The use of machine learning has the potential to reduce the computational cost of exact solution methods and improve the solution quality of heuristic methods. The fusion of the two worlds of "classical optimization" and "artificial intelligence" is in vogue, and one can be curious to see what results research in this interdisciplinary field can still achieve. In any case, one thing is certain: It is exciting to see how diverse the two approaches can be combined and how results can be improved as a result - learned is simply learned! ◆

RISC Software GmbH
www.risc-software.at

linkedin.com/company/risc-software-gmbh

twitter.com/RISC_Software

facebook.com/RISC.Software

xing.com/pages/riscsoftwaregmbh

Imprint
Publisher and
media owner:
RISC Software GmbH,
Softwarepark 32a, 4232 Hagenberg,
+43 7236 93028, office@risc-software.at
Responsible for the content: DI Wolfgang Freiseisen
Chief Editor: Mag. Cornelia Staub
Design and graphic layout: Melanie Laßlberger, MSc

Version: 1.0 | 16.06.2023
Image credits: RISC Software GmbH, iStock.com, Adobe Stock
if not stated otherwise