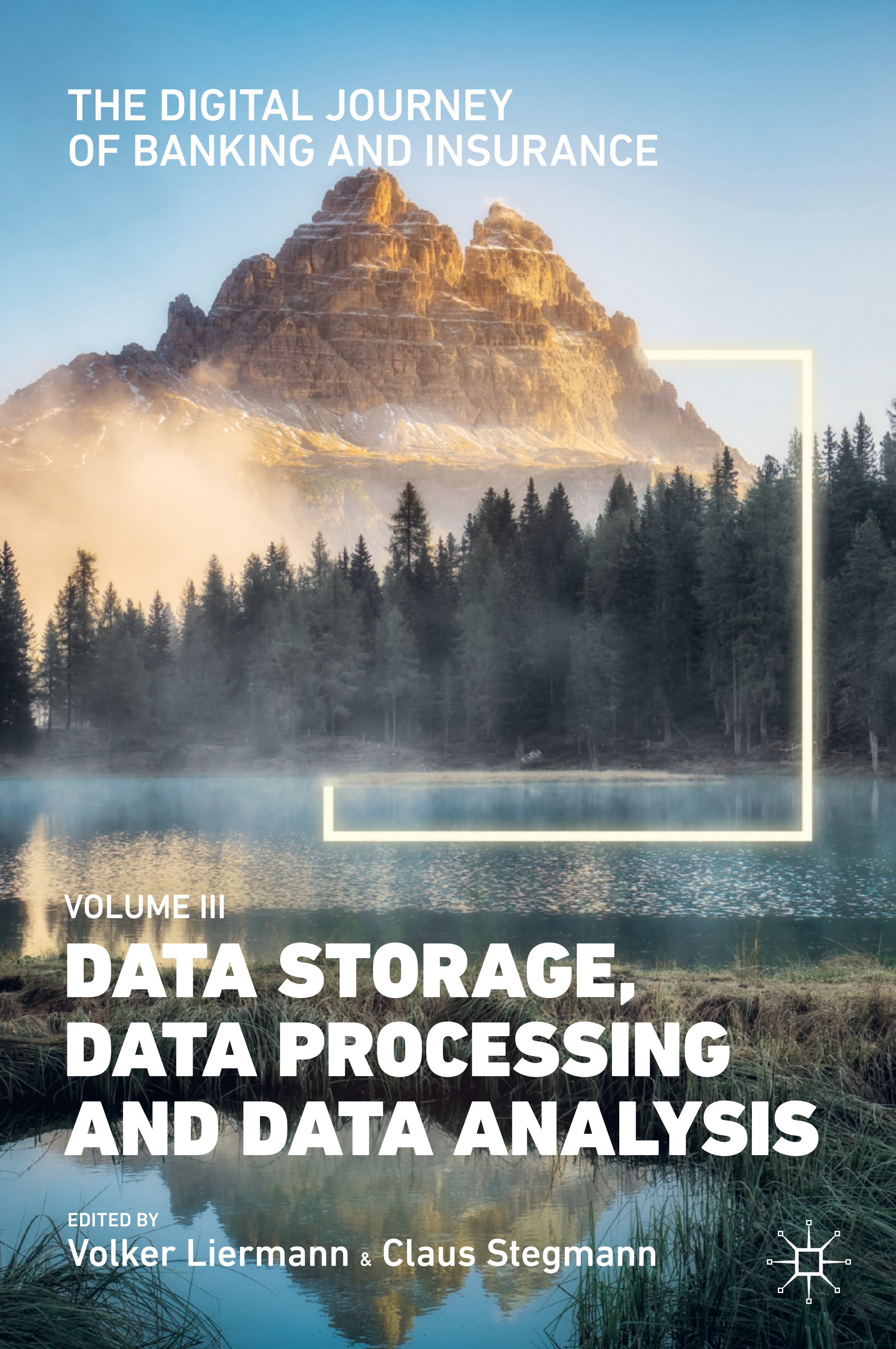


THE DIGITAL JOURNEY OF BANKING AND INSURANCE

VOLUME III

DATA STORAGE, DATA PROCESSING AND DATA ANALYSIS

EDITED BY

Volker Liermann & Claus Stegmann

The Digital Journey of Banking and Insurance, Volume III

"Virtually all financial institutions have embarked on ambitious digital journeys, both to provide better products and customer experience more efficiently and in response to the threat of industry disruption by FinTech competitors. There is no doubt that there will be winners, and there will be losers. I am convinced that *The Digital Journey of Banking and Insurance* series is indispensable reading for the future winners."

—Thomas C. Wilson, *CEO, President and Country Manager at Allianz Ayudhya*

"*Data Storage, Processing, and Analysis*, the last volume of *The Digital Journey of Banking and Insurance*, gives in-depth insights into technological aspects which is essential for successful digital transformation."

—Dr. Carsten Stolz, *CFO Baloise Group*

"Technological aspects are essential for successful digital transformation and so I like to get in-depth insights by *Data Storage, Processing, and Analysis*, the last volume of *The Digital Journey of Banking and Insurance*."

—Gerhard Lahner, *COO of Vienna Insurance Group*

"We do remember when we started our digital journey, but we do not know when it will be over. Therefore, we are definitely in the middle. The book series *The Digital Journey of Banking and Insurance* is a must-read for of all of us."

—Christian Peter Kromann, *CEO, SimCorp*

"Although the subjects described in this book are technical, the authors find a way to explain them in a comprehensible way. An up-to-date book for this subject."

—Bernhard Hodler, *Former CEO Julius Baer Group*

Volker Liermann · Claus Stegmann
Editors

The Digital Journey of Banking and Insurance, Volume III

Data Storage, Data Processing and Data Analysis

palgrave
macmillan

Editors

Volker Liermann
ifb SE
Grünwald, Germany

Claus Stegmann
ifb Americas, Inc.
Charlotte, NC, USA

ISBN 978-3-030-78820-9 ISBN 978-3-030-78821-6 (eBook)
https://doi.org/10.1007/978-3-030-78821-6

Cover credit: Stutterstock/Blue Planet Studio

This Palgrave Macmillan imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Acknowledgments

The three-book series was the natural next step from the book “The Impact of Digital Transformation and FinTech on the Finance Professional” and an exciting project for us. We look back with gratitude at the many discussions with clients, partners, and colleagues at ifb. Without this vital community, such an undertaking would not be possible.

We would first like to thank all contributors (clients, partners, and colleagues), whose expertise was invaluable in exploring and formulating such a comprehensive work with a wide overview and deep insights. Their insightful feedback helped us to sharpen this work to this amazing level.

In addition, we would like to thank Tula Weis and her team from Palgrave Macmillan for their advice and support in this project.

We like to thank Satzanstalt for supporting us in the development and realization of the book covers idea.

We would also like to thank our colleagues Julia Horstmann, Davin Radermacher, and Jenny Klein for their support in all the small, but important things that make such an undertaking a success.

Cologne, Germany Volker Liermann
March 2021 Claus Stegmann

Introduction to Volume III—Data Storage, Data Processing and Data Analysis

The business models of financial sector companies were always and are still information-driven. Digital transformation and the stronger customer focus (driven by the fintech companies and Big Tech) demand information on customer behavior (NBO,[1] NBA[2]). These behavioral customer patterns are the key to continuous revenue generation.

John Naisbitt wrote in his well-known book Megatrends, "We are drowning in information but starved for knowledge." (see Naisbitt, 1982). Handling data has become a key—if not the most decisive—capability and skill an organization in the financial sector needs. Handling data (including analyzing data) is the task to transform information into knowledge.

To be more precise, handling data means collecting, storing, and transforming (and analyzing) data.

Three major trends—especially in external digitalization—are driving the data handling process: Trend A: increase in available data, Trend B: accelerated speed in data processing, Trend C: special structures for optimized storage and querying of complex and unnormalized[3] data structures. These three trends are mirrored in technologies: Trend A is reflected in the new cluster databases like Hadoop (or AWS-S3, Google Bigtable), making the handling

[1] Next best offer (see May, 2019).

[2] Next best action.

[3] The process of organizing the fields and tables of a relational database is called database normalization. Normalization helps to minimize redundancy and dependency.

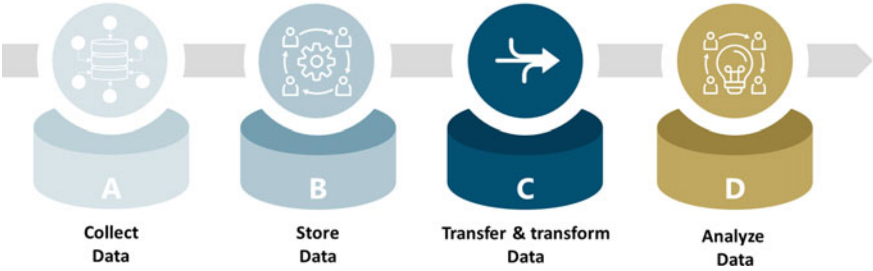


Fig. 1 Data process (© ifb SE)

of high data volumes possible and affordable. Trend B is shown in streaming technology (like Kafka, see Steurer, 2021) making real- or near-time data provision possible. Streaming technology was in place long before the digital transformation sped up, and the first steps with Kafka Standalone faced several challenges. The modern architecture concepts like Lambda, Kappa and Delta architectures (see Krätz and Morawski, Data Infrastructures—Lambda Architecture and Other Architectures, 2021) combine traditional architecture patterns providing stability with the dynamism and speed of streaming technology. Other facets of Trend B are in-memory databases (IMDB), making it feasible to handle huge data volumes in the blink of an eye. Trend C materializes in different specialized databases (like document-based databases and graph databases) as well as in distributed ledgers.

Mass Data and Data Availability

Trend A (increase in available data) has its origin in the earlier days of the internet. The internet was growing, and Google needed to store this growing data as a key capability of a search engine. Google was looking for affordable mass data storage. Google File System (GFS) is a proprietarily distributed file system.[4] Some of its components influenced Hadoop and its derivatives and advancement.

Now these two elements, (A) growing volume of available data and (B) technology to handle this data (at reasonable cost), started to interact and to scale up with ever more continuous acceleration. Google transformed its business model from a search engine (with a focus on advertising) to other

[4] Google File System targets an efficient, reliable access to data stored in large clusters of commodity hardware.

data-driven revenue models (DDRM). Imitating this success, a whole sector of data-driven revenue business models has arisen.

After seeing the success of DDBM, other sectors started to explore their opportunities, transforming data into benefits for their existing business model. The setup is made up of four kinds of sources: (1) existing data (data unused until now), (2) newly collected data (which was lost because it was deleted or not properly archived), (3) external data (e.g., Google, Facebook), (4) the intelligent linkage of the previous three (see Fig. 2).

Fig. 2 Data process (© ifb SE)

The Internet of Things (IoT) is a collective term for technologies of a global infrastructure within information societies. It makes it possible to network physical and virtual objects with each other and to let them work together through information and communication technologies. The Internet of Things produces an immense volume of data every day.

Another aspect coming with growing data availability is the way data is available. The traditional association was structured data (in a relational database, maybe differentiated by dimensions and key figures). The newly collected data was digitally (or electronically) available but in an unstructured way.

Speed and Streaming

Dynamism in modern (streaming/integrating) architecture has two dimensions. Operational dynamism (the data availability has improved) and a change-enabling dynamism (due to the micro-service component, changes (examples can be found in (Steurer, 2021) can be deployed much faster).

When looking at outside digitalization, the need to speed up is more obvious than in inside digitalization. Identifying and solving the right problem for the customer at the right time is the only way to stay near the customer in an increasingly competitive market (if this is the target of the company's business model[5]).

Speed for its own sake does not necessarily provide benefits. For example, instant payment[6] was not well-received in the early days in 2017. In contrast, Zalando (and other e-commerce companies) offer a two-week payment term and bring the payment closer to a debit card payment.

It must be considered that (not only) Millennials[7] and Generation Z[8] use communication and media differently and dispose disparately of availability of goods and services. Young people's changing demands and utilization have always impacted older generations (in this case, Generation X[9] and Baby Boomers[10]).

The important transactions in life, like taking out a mortgage, are more often perceived as special occasions that should be celebrated. They are more likely to be performed in a branch office, involving direct interactions with another human being. In contrast, other services like checking an account (or checking all accounts across banks via an aggregator app) or a consumer credit (which is less of a financial transaction than a surplus to buy goods) do not require direct interaction. In retail banking transactions, the right timing or, more accurately, the right context will win the deal. The enablers here are the relevant data and infrastructure with a speed capability.

Things look different in corporate banking, for now. Corporates are changing, and the finance departments of medium and large companies will

[5] Some analysts see a wider spread coming for institutes providing products (manufacturer) and institutes composing products to a solution (orchestrator).

[6] SEPA Instant Payment.

[7] Or Generation Y: Generation born in the period from the early 1980s to the late 1990s.

[8] Gen Z for short, this is the simplified term used to describe the successor generation to Generation Y.

[9] The term Generation X (also Gen X) usually refers to the generation following the Baby Boomers.

[10] Baby Boomers refers to the generation born during the periods of rising birth rates (the "baby boom") following World War II or wars in other countries.

go through a transformation in the next decade, demanding more standardized and easier to use interfaces (API[11]). These digital improvements will serve the business model targets of corporates and will put speed pressure on the availability of financial transactions.

Trends A & B are relevant for insurance companies in the same way.

These trends will affect and change the infrastructure (driven by outside digitalization). It does not make sense to work with two kinds of infrastructures (in terms of employees' technological skills). It is therefore expected that the modern streaming architecture patterns and infrastructures will become available to the finance department (even without a business case[12]).

To summarize, modern streaming architectures (like Delta architecture) are about to be driven by outside digitalization (customer-related) and by inside digitalization (optimizing processes and data flows).

In-Memory Databases

An in-memory database (IMDB) is an innovative database management system that uses the main memory of a computer as data storage. In-memory databases can be distinguished from conventional database management systems, which use hard disk drives for this purpose. Many standard software vendors[13] and a number of open-source frameworks[14] offer in-memory databases as a tool or toolset. Most of the in-memory databases have also implemented column-oriented storage and query to improve performance.

In-memory databases are fast and extremely valuable in selected applications, but the required hardware can be costly. A function-driven distribution of data among the different types of databases (hot, warm, and cold storage), called data tiering, is extremely useful and can reduce costs.

[11] Application programming interface.

[12] A business case for only speeding up accounting processes is a challenge because the value of timely information will always be viewed differently.

[13] SAP—HANA (for details see Kopic et al., 2019), Oracle—TimesTen, Microsoft—Hekaton.

[14] Apache Ignite, Redis, VoltDB.

Unstructured Data

Addressing the challenge of unstructured data, a new kind of database (NoSQL[15]) was established. Well-known implementations include Riak, Apache Cassandra, CouchDB, MongoDB (see Bialek, 2021), and Redis. Most of the cluster databases like Hadoop can be file-based and therefore have at least components of NoSQL databases.

Unstructured data can be text (like in an email or a PDF file) already containing electronic characters, but it also covers text scans or other kinds of graphics, videos, and spoken recordings. Text scans, graphics, videos and spoken recordings can be classified as raw material. Voice detection, OCR[16] and image processing are well-established tools to transform the raw formats into a character-driven (or object-driven) format. Natural language processing is the standard toolset to transform content from a character-driven format to structured data.

The ability to process unstructured data (especially character-driven data) opens up an endless stream of data that can be transferred into structured information using natural language processing.

The new availability of data generates demand to put the different entities in a context of interconnectedness by showing the connections ("Everything is connected" see also (Enzinger & Grossmann, 2019). To handle this connection, a graph[17] (originated from the mathematical object[18]) is an extremely powerful tool. Another new database class has been established to handle this connection data: graph databases and special-purpose query languages like Cypher (for more details see Bajer et al., 2021).

Distributed Ledger

In terms of storing data, distributed ledger with all its facets is a huge subject. Starting in 2008 with the Nakamotos Bitcoin whitepaper (Nakamoto, 2008), Bitcoin has gained visibility and attention. After the price peak at the end of 2017, it has now (March 2021) moved to new high values. In Bitcoin, the

[15] NoSQL (Not Only SQL) refers to databases that follow a non-relational approach and break with the paradigm of relational databases.

[16] Optical character recognition.

[17] A graph consists of nodes and edges (see Biggs et al., 1986).

[18] Graph Theory is a branch of discrete mathematics and theoretical computer science (see Diestel, 2017).

transfer and storage of values is paramount, or, if we oversimplify it, it is payment without an intermediator.

The next topic in the distributed ledger universe is smart contracts and the enrichment of the storage and payment functions by an almost endless set of functions that can implement the intentions in or behind legal contracts (dividend payments, digital rights, …). The most popular technology in the domain of public blockchains[19] is Ethereum. To implement private blockchains, Corda[20] and Hyperledger[21] are popular frameworks.

The different alignments of distributed ledger are summarized under the umbrella of the abbreviation DeFi (decentralized finance), highlighting the decentralized and distributed approach. Decentralized finance emphasizes the lack of intermediator, making it an experimental, novel form of financial market based on smart contracts and decentralized autonomous organizations.

Smart contracts enable participants to implement tokens with a variety of characteristics and features. The tokenization of real-world assets like Everledger (Foreverhold Ltd., 2020) has become more and more popular, while the mirroring of real-world assets into a distributed ledger can digitalize and optimize processes happening around real-world assets.

Another important aspect in the digital world is self-sovereign identity (SSI). This also allows control over the way personal data is shared and used. Self-sovereign identity allows a person, organization, or machine to create and fully control a digital identity without requiring permission from an intermediary or central entity Distributed ledger technology has certain technological advantages, making the distributed ledger an optimal platform for implementing SSI.

Data Analysis

The last part of the data journey in Fig. 1 is data analysis. In this subject area, data mining and analysis can be applied as well as machine learning and deep learning. There are various baskets of different open-source proprietary tools, frameworks, and platforms available, and they are improving and

[19] The concept of a public blockchain is behind most major cryptocurrencies. Access to this blockchain variant is open to any participant, so anyone who wants to execute transactions, validate blocks, and view the entire history of the blockchain is allowed to do so.

[20] See Corda, 2020.

[21] See The Linux Foundation, 2020.

transforming continuously to serve the purpose of transforming data into information to support the core business model of the company.

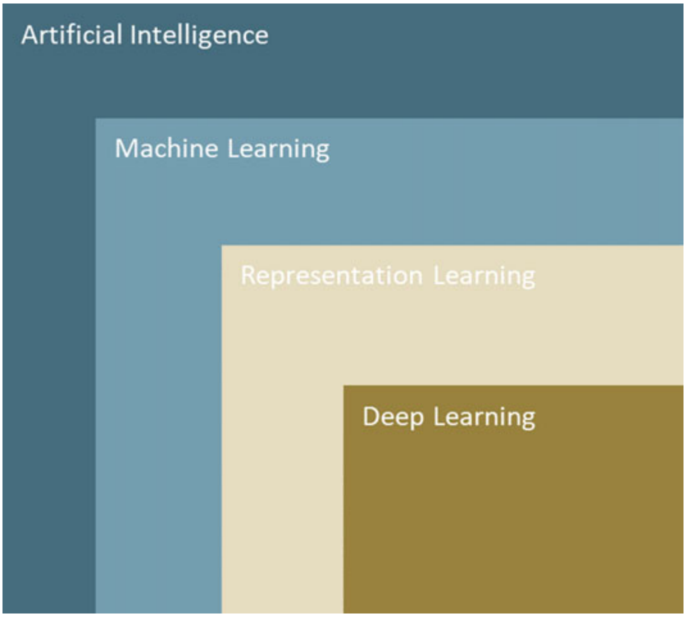


Fig. 3 Overview artificial intelligence (© ifb SE)

Figure 3 shows the traditional subset visualization of artificial intelligence. Machine learning has evolved from different statistical disciplines. A good definition of machine learning is given in (Chakraborty and Joseph, 2017) and an introduction can be found in (Liermann et al., Mathematical Background of Machine Learning, 2019). Representation learning[22] aims to replace manual feature engineering, using techniques allowing feature detection or other raw data classification by representations. Deep learning is a subset of machine learning methods that implements artificial neural networks (NN) by forming an internal structure with numerous hidden layers between the input layer and the output layer. See (Goodfellow, Bengio und Courville, 2014) for a comprehensive introduction to deep learning and (Liermann et al., Deep Learning—an Introduction, 2019) for an introduction to deep learning.

[22] Sometimes also referred to as feature learning.

Overview of Book Series "The Digital Journey of Banking and Insurance"

This book is the third volume of the three-volume book series "The Digital Journey of Banking and Insurance." The first volume "Disruption and DNA" focuses on change and the things staying stable in the banking and insurance market (outside view) as well as the effect on accounting, risk management, and regulatory departments (inside view). The inside view is rounded off with an analysis of cultural alterations.

The second volume "Digitalization and Machine Learning Applications" mainly emphasizes use cases as well as the methods and technologies applied (such as processes, leveraging computational power and machine learning models).

This volume "Data Storage, Processing and Analysis," the last one of the series, considers how to deal with data. The angle shifts over the volumes from a business-driven approach in "Disruption and DNA" to a strong technical focus in "Data Storage, Processing and Analysis," leaving "Digitalization and Machine Learning Applications" in-between with business and technical aspects.

Literature

Bajer, Krystyna, Sascha Steltgens, Anne Seidlitz, and Bastian Wormuth. 2021. "Graph Databases." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Bialek, Boris. 2021. "Digitization and MongoDB." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Norman L. Biggs, E. Keith Lloyd, and Robin J. Wilson. 1986. *Graph Theory 1736–1936*. London: Oxford University Press.

Chakraborty, Chiranjit, and Andreas Joseph. 2017. *Staff Working Paper No. 674—Machine Learning at Central Banks*. London: Bank of England.

Corda. 2020. *Corda*. Accessed December 15, 2020. https://www.corda.net/.

Diestel, R. (2017). *Graph Theory*. Springer.

Enzinger, Philipp, and Stefan Grossmann. 2019. "Managing Internal and External Network Complexity." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Foreverhold Ltd. 2020. *Everledger.* Accessed December 15, 2020. https://www.everledger.io/.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2014. *Deep Learning.* http://www.deeplearningbook.org. MIT Press. http://www.deeplearningbook.org.

Kopic, Eva, Bezu Teschome, Thomas Schneider, Ralph Steurer, and Sascha Florin. 2019. "In-Memory Databases and Their Impact on Our (Future) Organizations." In *The Impact of Digital Transformation and Fintech on the Finance Professiona*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Krätz, Dennis, and Michael, Morawski. 2021. "Architecture Patterns—Batch & Real Time Capabilities." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Liermann, Volker, Sangmeng Li, and Norbert Schaudinnus. 2019. "Deep Learning—an Introduction." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Liermann, Volker, Sangmeng Li, and Norbert Schaudinnus. 2019. "Mathematical Background of Machine Learning." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

May, Uwe. 2019. "The Concept of the Next best Action/Offer in the age of Customer Experience." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Naisbitt, J. (1982). *Megatrends: Ten New Directions Transforming Our Lives*. Warner Books.

Nakamoto, Satoshi. 2008. *Bitcoin—a Peer-to-Peer Electronic Cash System.*

Steurer, Ralph. 2021. "Kafka—Real-Time Streaming for the Finance Industry." In *The Digital Journey of Banking and Insurance, Volume III: Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

The Linux Foundation. 2020. *Hyperledger.* Accessed December 15, 2020. https://www.hyperledger.org/.

Contents

Big Data and Special Databases

Streaming

Data: A View of Meta Aspects

Distributed Ledger

Machine Learning and Deep Learning

Notes on Contributors

Eljar Akhgarnush Implementation Consultant at ifb group since April 2018, gained knowledge in software and financial sector topics during his studies as well as the various positions he has since held. Starting out with a focus on financial supervision and regulation, he soon shifted his attention to the technical side and agile project management. After having received his B.Sc. in business administration from CAU Kiel, he gained his M.Sc. in international economics and policy consulting at OvGU in Magdeburg. He has since remained keen on exchanging knowledge and exploring new areas.

Krystyna Bajer has been a Consultant at ifb group in the Data Information Team since the beginning of 2020. She holds a degree in business mathematics and financial services risk management and has been able to gain experience in the banking and insurance industry. Since October 2020, she has been assisting in a project at an insurance company in the area of data architecture.

Boris Bialek Global Head of Enterprise Modernization, leads the industry practices at MongoDB and specifically focuses on the modernization of banking solutions. His work focus is digital transformation and true innovation implementing exciting solutions, be it a new mobile payment platform for a US banking group or risk and treasury platform for a G-SIB that allows real-time data reconciliation. Before joining MongoDB, he worked for many years with FIS, IBM, Dell and Compaq Computers. He was one of the founding members of the SAP LinuxLab, implementing the first ever Linux client with SAP. He obtained an M.Sc. degree from Karlsruhe Institute of Technology.

Fabian Bruse Director, has worked at ifb group since 2011. He started his career in the regulatory reporting sector as a software tester and later moved on to SAP BW and SAP BA development with a particular focus on IRR and CRA modules for customers in Germany and Luxembourg. His more recent projects include modern ETL and reporting processes where he has the role of a—partially remote—Scrum Master (PSM2 certified). Since 2017, he has also coordinated the technical part of the ifb Blockchain Team and administrated the ifb Hyperledger system on Kubernetes. Fabian has a degree in physics from the University of Bonn.

Marie Kristin Czwalina Senior Consultant, has been working at ifb group in the Core Banking Team since 2020. She deals with further development in the area of innovation management and digitalization topics around the areas of sales and processing in banks, and specializes in process automation, customer-oriented advice and the implementation of GDPR requirements in IT applications. She studied business informatics with a focus on IT management in her master's degree at the FOM University of Applied Sciences.

Milan den Heijer is a Managing Consultant at ifb group. He has been working in IT consulting in the financial sector for six years. His focus is on data modeling, ETL modeling and data warehousing. In these fields, he has gained experience with platforms such as SAP BW, SAP FSDM and SAP HANA. Moreover, he has experience in the fields of master data management, data governance and metadata management. He has a background in physics and astronomy.

Dr. Johannes Floß has been a consultant with ifb group since 2019. He mainly works on IFRS 17 implementation projects with a specialization on SAP FPSL and SAP PaPM. As a second topic, he develops data science tools for the insurance business, e.g. the prediction of churn rates with the help of machine learning algorithms. Before his time at ifb, Johannes was a research assistant at the University of Toronto's Centre for Quantum Information and Quantum Computing, studying fundamental problems like quantum chaos and light-matter interaction. He holds a Ph.D. in chemistry from the Weizmann Institute of Science, Israel.

Jens Freche Managing Consultant and Team Lead Data Management, has worked at ifb group since 2011. He started his career in data integration for Oracle and SAP systems. Later, he moved on to SAP BW development, data governance, SAP FPSL and IFRS 17 standard for customers in various countries including Germany, Luxembourg, Switzerland, the USA, Japan, Chile and Israel. Since 2018, he has been the Head of the Data Lineage and Data Governance Team. Jens has a Master of Science in Mathematics from the University of Applied Science Aachen.

Dennis Kraetz Partner at ifb group, has been working in the consulting industry for more than 15 years, focusing on finance transformations in the financial services sector. Within ifb, he leads the consulting practice for Cross Industries, developing solutions in the context of architecture, information and transformation management. In recent years, he has been concentrating on emerging architecture patterns and their application in the banking and insurance industry. Dennis holds a degree in business administration.

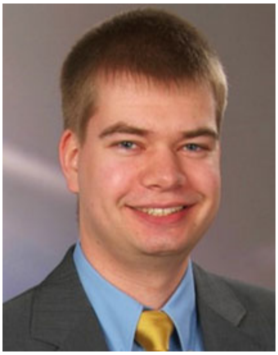
Heinrich Krebs Electrical Engineer (FH). Heinrich joined ifb in 2015 leaving a career in science. Since then, he has worked as a consultant for different clients and participated in several ifb working groups on the analysis of new technologies, especially blockchain implementations and self-sovereign digital identities.

Matthias Kurfels Director at ifb SE, has been a consultant and trainer for banks, capital management companies and financial service providers since 2010. Before that, he was working for German savings banks and a savings bank association for more than 25 years.

His focus is on regulatory requirements for corporate and risk management, IT governance, internal audit and compliance. He is also head of ifb's internal working group for regulatory aspects of blockchain technology.

Dr. Sangmeng Li Senior Consultant at ifb SE, has primarily worked as a data scientist for quantitative risk management in the financial industry with a focus on data analysis, risk modeling and technical implementation. She received her doctorate in mathematics from the University of Münster, having conducted research on stochastic differential equation and Monte Carlo simulation as part of her Ph.D.

Volker Liermann Partner at ifb group, worked in the banking industry for over two decades, primarily focusing on financial risk management. Throughout his career, he has focused on developing integrated and comprehensive frameworks to help organizations correctly project risk at a strategic and tactical line of business and departmental level. He has also focused on developing frameworks to integrate stress testing and regulatory stress tests. In recent years, his focus has shifted to digitalization, machine learning and digital processes including improvements to classical financial and non-financial risk management. He has a background in economics and a degree in mathematics from the University of Bonn.



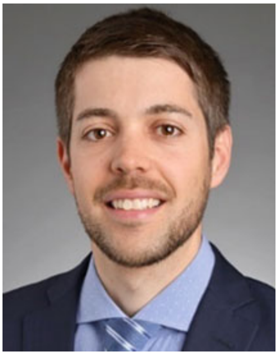
Michael Morawski Director, has worked at ifb group since 2008. He has conducted various projects in Germany and abroad mainly in the context of regulatory reporting and controlling, often involving ETL pipelines in systems like SAP SEM Banking, SAP Bank Analyzer, SAP BW, SAP HANA and lately also Hadoop. He focuses on the challenges posed by Big Data in the financial industry. Since 2018, he has coordinated the ifb internal Hadoop working group. Michael has a degree in biology from the University of Wuerzburg.

Georg Schmidt Managing Consultant, has worked at ifb group since October 2013. He started his career in the accounting sector as a developer for source data integration with SAP BW in Colombia and later in Peru. He also worked with a German automobile bank in the area of data transformation, working with the Hadoop tool stack, such as HDFS, Hive, Spark, Sqoop, Oozie, HBase and Kafka. Similarly, he developed a Spark data pipeline using Atlas as data catalog for an Austrian insurance company. Georg holds a master's degree in administration and informatics from the University of Potsdam.

Dr. Anne Seidlitz has been working as a consultant in the financial sector for more than four years. As part of the Business Intelligence & Data Governance Teams at ifb group, her focus is on the topics data integration, data warehousing, data modeling and data governance. She studied physics and holds a doctoral degree in this field from the Martin Luther University Halle-Wittenberg.

Claus Stegmann has as Co-CEO of ifb group—an international consulting firm—acquired extensive know-how over the last three decades in the financial industry regarding finance transformation, risk management and regulatory compliance. He is intensively engaged with the current challenges of the financial industry, which result from strong changes to customer behavior, a changing competitive environment and new technologies due to digitalization. He has also co-authored books on Stress Tests in Banks, Basel III as well as Digitalization in the Finance Industry, and graduated from Business School at the University of Passau, Germany.

Dr. Sascha Steltgens is a Consultant in ifb group's Information Management competence center. With a specialty in data integration, his work focuses on data delivery for all kinds of purposes in the financial sector. Since January 2019, his work has assisted banks and insurance companies with the implementation of state-of-the-art database technology. He studied molecular biology as well as biochemistry and holds a Ph.D. in biochemistry from Heinrich-Heine University Düsseldorf.



Ralph Steurer Director at ifb, has been working in the financial services industry for over ten years as architect and technical project manager. Over the years, he transitioned from SAP implementation projects (SAP Bank Analyzer and SAP HANA XSA) to software development, focusing on Apache Kafka, Java, NodeJS and ReactJS. His academic background includes a bachelor's degree in computer science from Zurich University of Applied Sciences.

Stefan Strube Senior Consultant, has been working at ifb group since 2018. He has supported well-known clients from the banking and insurance industry, most commonly dealing with both business and technology related topics and primarily focusing on business analysis, regulatory affairs and data analytics.

In addition, he acts as the product owner on the development of Modern Data Management, where he addresses the influence of the Digital Transformation as well as Open Banking on the development of German and European banks, and combines this with his work on modern business warehousing applications like SAP BW/4HANA.

He graduated with a Bachelor of Science from the University of Bonn and a Master of Science from the University of Göttingen.

Dr. Jeyakrishna Velauthapillai has been working in consulting since 2019 with a focus on SAP products such as SAP FPSL in combination with IFRS 17 topics. He is especially interested in the digital transformation of the insurance industry and its challenges. Additionally, he studied economics and mathematics, holds a doctoral degree in economics, and has experience in agent-based modeling and financial mathematics.

Bastian Wormuth has been working in consulting for over 17 years with a focus on information management and data analytics, most of the time in the financial industry. He leads the Business Intelligence & Data Governance Teams at ifb group. As an architect and data governance expert, he helps financial institutions to optimize data management processes and to manage data risk. Bastian obtained a degree in mathematics at the University of Technology Darmstadt and has published several papers on knowledge discovery and data analytics.

List of Figures

Data Lineage

Digitization and MongoDB—The Art of Possible

Graph Databases

Data Tiering Options with SAP HANA and Usage in a Hadoop Scenario

Kafka: Real-Time Streaming for the Finance Industry

Architecture Patterns—Batch and Real-Time Capabilities

Kafka—A Practical Implementation of Intraday Liquidity Risk Management

Data Sustainability—A Thorough Consideration

Special Data for Insurance Companies

Digital Identity Management—For Humans Only?

Overview Machine Learning and Deep Learning Frameworks

Methods of Machine Learning

List of Tables

Graph Databases

Special Data for Insurance Companies

Methods of Machine Learning

Big Data and Special Databases

Data availability and data technology stimulate each other continuously. The internet has made mass data available for almost every important (and unimportant) subject. The volume forced Google to develop a concept to deal with such amounts of data: the BigTable[1] and the MapReduce[2] concept as a part of the Google File System. The availability of technology (especially as a cost-efficient open-source implementation[3]) then opens up for other Big Data processing use cases, such as customer clustering analysis or (when including the time dimension) the prediction of a customer journey.

Driven by the business requirements, topic-specific database variants like graph databases or other NoSQL databases (document store, key-value store, object database, …) have been established in the market. There is no perfect NoSQL database. Every type of database has advantages and disadvantages depending on the subject it is applied to. The evolution of specific types of database shows the demand for application-specific database types (in-memory DB, cluster DB, graph DB, document DB, …). Once the technology is implemented and available (ideally as open source[4]) new use cases are mapped, and sometimes surprising applications can arise from a tool in the right hands. For example: graph databases are used in the context of anti-money laundering (AML) to analyze connected persons and accounts. This

[1] See (Chang et al. 2006).

[2] MapReduce (see Dean and Ghemawat 2004) is strongly associated with Hadoop, incorrectly. While the Hadoop ecosystem has already further developed with spark and Databricks, the MapReduce concept is no longer of importance.

[3] See details on Hadoop in (Akhgarnush, Bröckers and Jakoby 2019).

[4] Which happens quite often if the technology is relevant.

application is certainly not the most obvious application for graphs (nodes and edges), but formulation of the challenge AML as a graph delivers stable and reliable results.

The first chapter in this part (Freche, den Heijer und Wormuth, 2021) tackles the subject of data lineage. The Basel Committee on Banking Supervision regulation announced its “Principles for effective risk data aggregation and risk reporting” in 2013 (BCBS 239, 2013). Principles include requirements demanding a data lineage.[5] The chapter discusses the regulatory requirements but also explains the need for data lineage for internal and external business requirements. The chapter closes with an overview of the most common tools for data lineage.

The second chapter (Bialek 2021) explores and analyzes the need for organizational flexibility and how to achieve it in a cloud-based environment. Most institutions face the situation of a silo-oriented environment. The chapter describes MongoDB as a solution to offer organizational flexibility to an institute. The chapter illustrates different paths leading to the cloud and a scalable environment for data modeling and data management.

The part’s third chapter (Bajer et al. 2021) looks at a special database class: the graph database. The chapter first explains why graphs and their ability to document and analyze connections are an important tool in our connected world. The chapter provides the technical background, such as data model, storage and visualization of graphs as well as providers of tools for this special database. The discussion of use cases in the graph context closes the chapter.

The final chapter of this part (Morawski and Schmidt 2021) provides a summary of data tiering options in SAP HANA given different surroundings. Motivated by the cost-pressure originating from in-memory databases (IMDB), the chapter explains the need for an application-driven provision of storage capacities. The chapter presents blueprints for how to implement data tiering of SAP HANA databases with different Hadoop environments (e.g. spark and SAP Vora). The tools presented include the SAP Data Lifecycle Manager.

Literature

Akhgarnush, Eljar, Lars Bröckers, and Thorsten Jakoby. 2019. “Hadoop—a standard framework for computer clusters.” In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

[5] Even if the term data lineage is not mentioned explicitly.

Bajer, Krystyna, Sascha Steltgens, Anne Seidlitz, and Bastian Wormuth. 2021. "Graph Databases." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

BCBS 239. 2013. *Basel Committee on Banking Supervision (BCBS) 239.* Accessed December 15, 2020. https://www.bis.org/publ/bcbs239.pdf.

Bialek, Boris. 2021. "Digitization and MongoDB." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Chang, Fay, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. 2006. *Bigtable: A Distributed Storage System for Structured Data.* Mountain View, CA: Google Inc.

Dean, Jeffrey, and Sanjay Ghemawat. 2004. *MapReduce: Simplified Data Processing on Large Clusters.* Mountain View, CA: Google Inc.

Freche, Jens, Milan den Heijer, and Bastian Wormuth. 2021. "Data Lineage." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Claus Stegmann and Volker Liermann. New York: Palgrave Macmillan.

Morawski, Michael, and Georg Schmidt. 2021. "Data Tiering Options with SAP HANA and Usage in a Hadoop Scenario." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Data Lineage

Jens Freche, Milan den Heijer, and Bastian Wormuth

1 Introduction and Motivation

In the financial sector, the main driver for a firm to maintain a high standard of data lineage documentation is compliance. On the one hand, external compliance is required by law by the regulator, and on the other, large organizations often have internal best practices to facilitate compliance which are necessary due to the scale of their IT landscapes. Apart from considering data lineage as a burden, investing in detailed and transparent documentation of a firm's data and data flows in its IT landscape on different levels can offer significant benefits.

J. Freche (✉) · M. Heijer · B. Wormuth
ifb SE, Grünwald, Germany
e-mail: Jens.Freche@ifb-group.com

M. Heijer
e-mail: Milan.denHeijer@ifb-group.com

B. Wormuth
e-mail: Bastian.Wormuth@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_1

1.1 Regulatory Requirements

One important regulatory requirement for financial organizations is imposed by the Basel Committee on Banking Supervision regulation 239 (BCBS 239 2013). BCBS 239 does not mention the term data lineage directly. The regulation requires that data as well as the related metadata relevant for the risk aggregation be defined clearly, especially for their structures and impact. This requirement cannot be fulfilled without structured metadata management and a schematic representation of the data lineage. The data lineage is fundamental for tracking the consumption of data and its origin (paragraphs 21, 33, and 52) (Data Cross Roads, n.d.).

Furthermore, data governance is a central part within BCBS 239. Although there is no explicit definition for data governance, business processes and roles associated with data management must be defined clearly. This information can only be used in a meaningful way in combination with the existing data lineage.

The handling of and requirements for personal data are described in the General Data Protection Regulation (GDPR). This regulation forms an essential part of the current requirements. The GDPR demands technical and procedural measures to show and document the processing and utilization of personal data. Without a high-quality data lineage, it is not possible to govern the usage of personal data. The data flow as well as the data provided to external systems or clients cannot be documented and demonstrated.

1.2 Group-Wide End-to-End Documentation

Experience gained from different projects for banking, insurance, and corporate clients shows that environments and systems are growing in all parts of groups and companies. Moreover, each system's and environment's needs for overall documentation of data and metadata increase as well.

20 years ago, an organization's metadata was something constant and changes to it were rare. Nowadays, however, the nature of metadata has become very dynamic. No tracing of changes and dependencies causes high risks for each organization. If this is neglected, an organization's survival is put at stake. Therefore, the realization and awareness for group-wide end-to-end documentation become more and more important, alongside any regulatory requirements. Such end-to-end documentation includes much more than just the dependencies of all data and metadata. Data lineage and data governance information are interdependent as roles and business processes are essential

to create group-wide documentation. Implementing and introducing this approach is the only way to get a group-wide overview and define responsibilities for metadata and related processes. Combining each relevant process step and its access to the data lineage would provide the chance to understand all dependencies between the action of each role on the one hand and the technical realization on the other.

1.3 Benefits of Data Lineage

In the ideal scenario, in which an organization has high-quality documentation of its data lineage, significant strategic advantages emerge.

1.3.1 Documentation, Understanding of Data and Data Flows

Section 1.2 suggests that comprehensive documentation consisting of lineage, governance, and business process information enables standardized documentation and knowledge transfer. This knowledge should not only reside in individual employees, but also in general documentation allowing everyone to read and understand dependencies and responsibilities. Fluctuations in staff or developments in IT systems can be handled in a much more efficient way.

Most IT architectures have gradually developed over time. To help maintain an overview of all metadata and appropriate data flows, having a data lineage is indispensable. In understanding the group-wide data, metadata, and data flows, a data lineage is very important. Based on the data lineage and associated business processes as well as integrated Data Governance enables a clear assignment of requirements and data dependencies.

1.3.2 Introduction of New Systems and Software

One of the biggest challenges while changing or introducing new software or IT systems is the consideration of all dependencies within the IT landscape, the responsibilities, as well as involved interfaces and data flows between all components. Having an end-to-end data lineage keeps the effort for any kind of adaptation to a minimum.

1.3.3 Error Detection and Troubleshooting

Incorrect data at the end of a data flow process while sourcing data from different systems and environments is well known. In this case, it is often difficult to determine potential errors and, moreover, to establish the person or group responsible for the part of the data flow causing the error. Understanding your data, hence having documentation of the data lineage and the responsible roles, will make error detection and troubleshooting much more efficient.

1.3.4 Data Lineage as a Prerequisite for Data Governance and Data Quality

Two important components of regulatory requirements are the existence of data governance and the assurance of a good data quality mechanism. Both requirements cannot be fulfilled without the knowledge of the underlying data lineage. Improvements to the data quality are based on the data lineage, because any kind of quality check is imprecise without the combination with the data lineage. For example, if a data quality issue in a data element is found, it is important to notify the consumer of this data element that their data is incorrect so that this person is aware of a possible error in a report. If the authority reporting the data quality issue has a high-quality data lineage at their disposal, the task of notifying the impacted consumers and/or reports can be performed in a much more efficient way. Furthermore, the role definition and responsibilities, which are the essence of data governance, are less useful without the connection to the data lineage.

1.3.5 Centralized Maintenance and Master Data Management

In some cases, a system-independent data definition is important in combination with knowledge of the origin of the data. To reach this goal, it is crucial to combine the data lineage with master data management to reach a single point of truth for data definition and the dependencies of different data within different steps in several systems.

1.3.6 Information Security

In the event of a security threat, high-quality data lineage documentation allows you to quickly determine if data is compromised and where this data resides.

1.3.7 Elimination of Redundancies

A data lineage analysis or an impact analysis enables you to identify potential redundancies and allows you to eliminate them. It is therefore an asset to have end-to-end data lineage documentation in place.

1.3.8 Impact Analysis

Having a data lineage enables the possibility of impact and lineage analysis. Based on this, one important advantage of data lineage is the possibility to find out where data originates from. For that, two dimensions must be considered. On the one hand, the granularity of the data lineage (see Sect. 2.2), and on the other, the way of presenting the lineage and impacts between different objects.

2 How to Define Data Lineage and Challenges

There is no universal definition of the term data lineage. However, there is a common understanding of what this term means. This common understanding usually revolves around the lifecycle of data. One perspective is the path along which a record of data traverses one or more systems, another is the columns through which a dataset moves. Raw data is generated through a certain process, often refined through ETL processes and consumed by applications or users. Data lineage is the documentation of the origin of the data, the transformation it undergoes, and where the data is consumed. This is, however, a relatively narrow understanding of data lineage. This definition can be extended to include, for instance, different levels (e.g., database column, database table, data flow between systems, etc.) or business processes which produce and consume this data. In this section, we discuss several aspects.

Another term often mentioned in the context of lineage is impact analysis. Impact analysis attaches further semantics to the lineage: lineage only contains the information that, for example, two columns are related, whereas an impact analysis results in the information of how one column is impacted if data in the other column is changed or deleted.

2.1 Horizontal vs. Vertical Data Lineage

A distinction is often made between horizontal and vertical lineage. Horizontal lineage describes a physical lineage through a data warehouse: from a landing area, through a staging area, via a core area into an outbound layer. A vertical lineage, on the other hand, focuses on the design of a database. It includes a business model, also known as a conceptual data model. This model is an abstract business view of the data. This serves as a design layer for a physical implementation in the form of a physical data model, which is a depiction of the physical artifacts on the database. These models are often directly linked, or are linked through a logical model, forming a bridge between business and IT. Horizontal and vertical lineage are therefore different concepts and demand an entirely different approach to the documentation of the data lineage. A vertical lineage revolves around the design process and thus relates abstract objects, such as a business partner, to physical implementation, such as a part of a star schema. A horizontal lineage shows the actual data flow and contains information about where the data enters the system, flows through the system, and is consumed.

These concepts are illustrated in Fig. 1. In this figure, the horizontal data lineage is shown in the form of physical models for the inbound, core, and outbound layers of the data warehouse. Vertical data lineage consists of a glossary, a conceptual (CDM), logical (LDM), and physical (PDM) data model.

2.2 Granularity Level of Data Lineage

Another important concept is the granularity level or detailedness of data lineage. Referring to horizontal lineage, the highest level of detail is achieved if the relationships between columns or fields can be traced throughout an IT landscape. The other extreme case is where only high-level data flow between IT systems are documented. In most cases, it can be stated that the finest granularity implies the highest quality. This is, however, not straightforward.

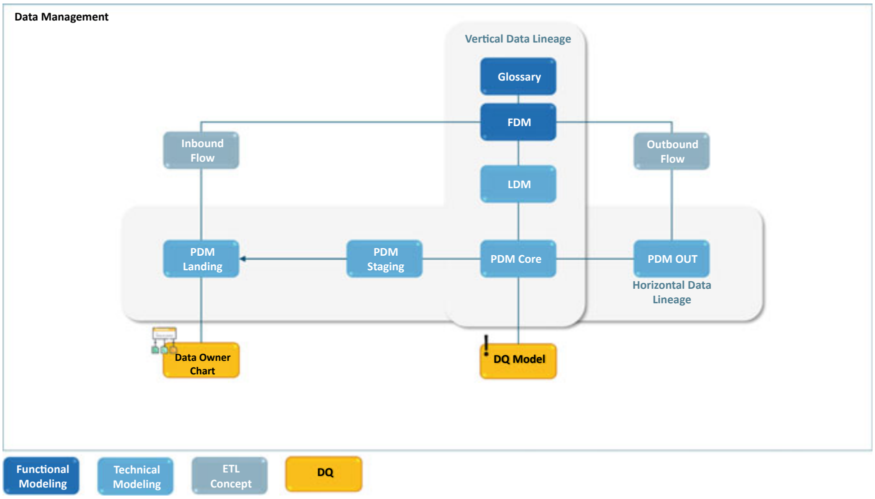


Fig. 1 Data management landscape: horizontal vs. vertical lineage together with data governance and data quality (© ifb SE)

For example, the following circumstances make it difficult or impossible to achieve a very detailed lineage.

2.2.1 Complex Transformations and Algorithms

In the case of complex transformations, for example, when programming code is applied to the data, to achieve a certain degree of data lineage code must be parsed. Even in the case of very common data query languages such as SQL, this can already be a complex task, depending on the presence of in-line views, etc. In the case of object-oriented programming languages, this task can easily become an insurmountable challenge. The approach often applied to this problem is to link the input and output parameters of the program to the program as an object. Obviously, this is an abstraction to a courser level of granularity and a loss of information.

2.2.2 Black-Box (Closed-Source) Third-Party Applications

In this case, input data is sent to an application, then transformed by the application, and output parameters are returned. Whatever transformations the data undergoes in the application, and what the relationship between the input and output fields is exactly, are unknown. This also implies a break in the lineage and a loss of detailedness.

2.2.3 Data Lineage Covering Business and Regulatory Requirements

Although a data lineage is a very good instrument covering the requirements described in Sect. 1, we are facing limits and restrictions. Data lineage information as well as ETL processes are often based on query or programming languages, e.g., SQL or SAP ABAP, processes in proprietary software, or complex algorithms. The usage of these implementations contains code parsing and additional reverse engineering steps, which can be expensive, especially for complex issues. In this case as well as in the black-box situation, consideration of costs, detailedness, and the outcome are important.

2.3 Multidimensional Lineage Including Additional Governance

A data lineage analysis is often constrained to data elements. This scope can, however, be extended to include concepts such as data governance and business processes. Data governance governs the responsibilities and accountabilities regarding the usage of data. Commonly, a data steward and a data owner are defined for a certain data element. A data steward is responsible for administrating this data element and the data owner is accountable for whatever happens to this data element. To extend a data lineage analysis to include data governance information boils down to analyzing which person or role is responsible and accountable for the data.

Data originates from and is consumed by business processes. A (data) lineage analysis can be extended by business processes. Process elements can be associated with data artifacts. This offers detailed insights into which data is consumed and generated by which processes. This is illustrated in Fig. 2, which shows a comprehensive metadata management layer that bundles data lineage information for all areas such as data governance and data models.

3 Approaches to Create a Data Lineage

Approaches for creating a Data Lineage can be divided into Snapshot Approach and the Model- or Process-driven data lineage approach. Data lineage is often acquired by taking inventory of the current state of the systems. This effort must be repeated. This is not a maintainable approach due to the dynamic nature of most IT systems. Moreover, the need for manual intervention often arises, which makes this process time-consuming

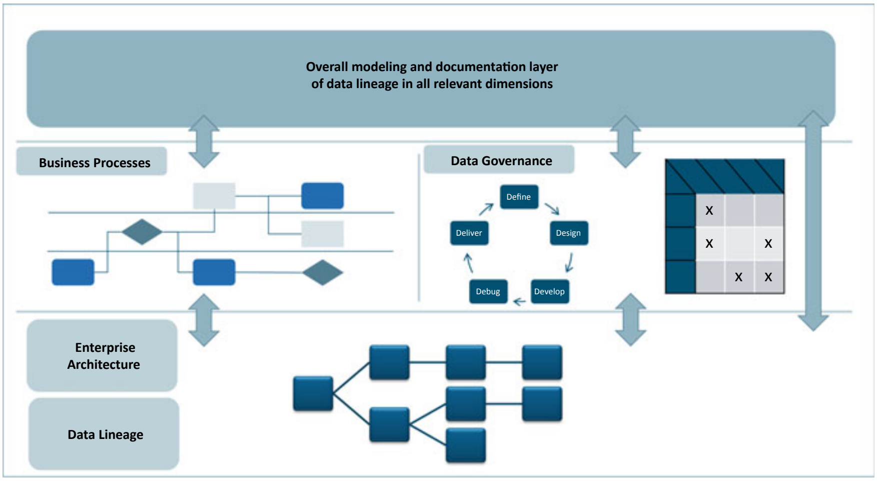


Fig. 2 Data lineage as an overall layer in which the relationships between data elements, business processes, and data governance roles are collected and governed (© ifb SE)

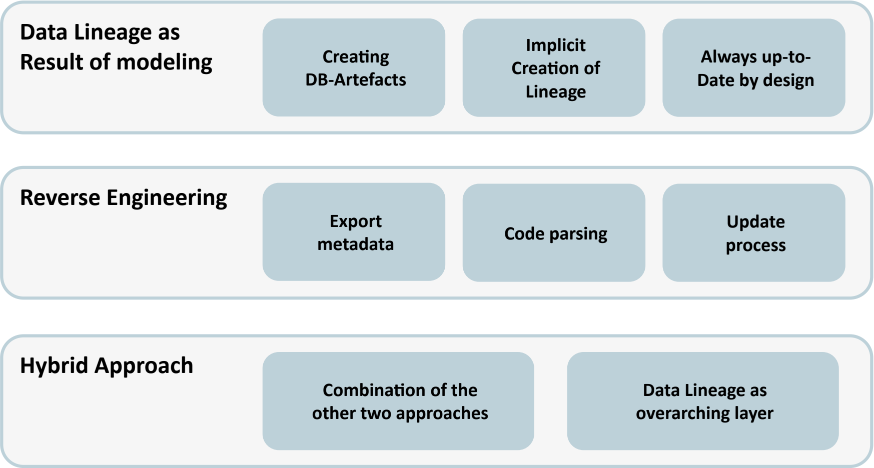

Data Lineage as Result of modeling
Creating DB-Artefacts
Implicit Creation of Lineage
Always up-to-Date by design
Reverse Engineering
Export metadata
Code parsing
Update process
Hybrid Approach
Combination of the other two approaches
Data Lineage as overarching layer

Fig. 3 Three approaches to create a data lineage (© ifb SE)

and prone to error. The way to tackle these problems is by designing processes in such a way that data lineage becomes a (side) product of the design process.

Project experiences in different companies show that the following three approaches are possible (see Fig. 3).

Figure 3 shows three approaches to create a data lineage. The first is an approach where the data lineage results from modeling of data or data flows. The second is an approach in which the lineage information is extracted from existing systems. The hybrid approach is where the lineage information is extracted from existing systems. The hybrid approach is a combination of both.

3.1 Data Lineage as a Result of Modeling: Model-Driven Approach

In the case of relational databases, a conceptual model can be created containing an abstract representation ("whiteboard sketch") of the business objects involved. From this model, a logical and physical data model can be derived. In this process, the model objects can be linked to each other, directly resulting in a vertical data lineage. If, from this physical data model, database artifacts are generated via DDL code, data lineage is a by-product of the melding process. This results in a lineage which is by design up to date and error-free. The same is possible for ETL processes: if a modeling tool is used to map one data element to another, and code is generated from this model, the data lineage is contained in the model.

3.2 Creating Data Lineage by Reverse Engineering

Not all metadata can be created via a model-driven design process. For example, if these components already exist or if it is a closed third-party system, it is not possible to create the artifacts including appropriate metadata. However, the data lineage is required as an overall layer as shown in Fig. 2. Therefore, the metadata of existing environments must be reverse-engineered. Within this approach, further aspects must be considered:

- Is automation possible for the reverse engineering process?
- Which granularity can be reverse-engineered, on an attribute, entity, or a higher level?
- A synchronization process is essential to keep the data lineage up to date.

3.3 Hybrid Approach

The hybrid approach combines the procedure characterized in Sects. 3.1 and 3.2. To get an overall data lineage, some components and environments will be constructed via a model-driven architecture (see Sect. 3.1), the data lineage is therefore given. The data lineage of other systems or parts of systems, which cannot be constructed through a model-driven process, will be depicted via reverse engineering. By combining these two approaches, an environment-independent data lineage can be realized.

4 Tools Used for Data Lineage

4.1 SAP PowerDesigner

SAP PowerDesigner (SAP PowerDesigner, n.d.) is an example of a classic data modeling tool. It offers all the models required for modeling a relational database: a conceptual, a logical, and a physical model. From a physical data model, SQL DDL scripts can be generated and deployed. On the other hand, databases can be reverse-engineered into models and models can be converted into each other, e.g., a logical data model can be generated into a physical data model and vice versa. Moreover, it offers other models such as an enterprise architecture model, a requirements model, and a business process model. These models are very useful for modeling additional lineage aspects related to Data Governance, organizational chart, and data quality, e.g., by using a requirements model.

PowerDesigner contains many possibilities to link objects with different link types. For example, one can connect a database column to a person or role in an organizational chart and a business process chart. This tool is highly customizable and offers the possibility to implement custom object classes such as a data quality rule. Scripting functionality allows the user to automate processes and extract information.

4.2 ETL Tools

Analogous to a data modeling tool such as SAP PowerDesigner, there are several data integration tools on the market which yield data lineage information in the modeling process. Examples of such tools are SAS DI (SAS Data Integration Studio, n.d.) and Ab Initio (AB INITIO, n.d.). These tools have built-in data flow visualizations and functions such as drill down. Therefore, by using such tools, up-to-date technical lineage documentation is ensured.

4.3 Data Lineage Extraction Tools

The number of tools dedicated to the purpose of extracting data lineage information from different types of systems is rapidly increasing. Some examples of such tools are Octopai (Octopai, n.d.), D-QUANTUM (D-QUANTUM synabi, n.d.), and Manta (MANTA, n.d.). These tools connect to a variety of databases, ETL tools as well as tools used for reporting and analytics and extract data flows and data elements. Although these tools undoubtedly add value to the data lineage analysis process, they do have limits.

One of these limits is that they are unlikely to cover all systems with a high level of precision. Large companies often have a large variety of tools in place and one data lineage extraction tool is unlikely to be able to extract all of their lineage information. To solve this problem, different data lineage extractors must be used, resulting in a patchy data lineage contained in different tools.

Moreover, closed systems do not allow third-party tools to access their metadata, so extracting metadata is a difficult task. Another issue could be that these programs parse programming code to derive the lineage information. Even in SQL, defining a data lineage from an INSERT/UPDATE stored procedure is ambiguous: consider a set of input and output parameters. If one input parameter (e.g., a date dimension table) is only used to modify a value of another input parameter, is this date information part of the data lineage path?

Also, the snapshot character of these types of tools must be considered: if the data lineage is extracted for a data warehouse and a reporting tool, these paths must be extracted simultaneously. If the lineage information for the data warehouse on the one hand and the reporting tool on the other would be extracted at different points in time, the lineage could be inconsistent. Hence, the update frequency of data lineage is a very important issue.

4.4 Apache Atlas

Another tool used for metadata management is Apache Atlas (Apache Atlas, n.d.). Atlas is a metadata and data governance solution designed for a Hadoop ecosystem for distributed computing and distributed storage. In Hadoop, several different solutions such as Spark, Hive, Sqoop, etc., often run in parallel and each have their own role. This, however, results in difficulties documenting the data elements and the data flows in the system, since no central mechanism exists to manage these objects, often resulting in high manual effort to achieve consistency.

Atlas solves this problem by connecting to these different systems and gathers their metadata into a central environment. It can extract metadata on the different Hadoop artifacts created in the different Hadoop applications using predefined types. Moreover, the user also has the ability to create new types. Atlas also allows the extraction of ETL processes to derive a physical data lineage. In many cases, this results in a column-level data lineage.

Atlas is an open tool, it has a modern GUI, but also offers the possibility to retrieve JSON-based content via a REST API.

The subject of data governance is contained in Atlas, because it is also integrated with Ranger, which is the designated Hadoop tool to manage data security. Hence, Atlas' metadata is used here to set rights and permissions regarding the data for all the tools in the Hadoop landscape.

4.5 Graph Databases

One topic that should be mentioned in the field of data lineage is graph databases. A graph database stores data in the form of nodes and edges. They are a type of non-relational (NoSQL) database. This concept focuses on the relationships between data elements, and therefore graph databases are naturally suitable for modeling and analyzing data lineage. Each node represents a data element, and each edge represents a connection between two elements. This connection can be interpreted as any type of relationship: a hierarchy (e.g., table–column), a transformation (e.g., column–column, table–table), etc.

One of the most used graph databases is Neo4j (Neo4j, n.d.). Neo4j has a graph query language called Cypher. By writing Cypher queries, it is possible to analyze data lineage paths, perform impact analyzes, and construct reports. Whereas in a relational database, the analysis between relationships involves many joints, this graph query language is naturally suitable for these analyzes.

Analogous to the previous section, graph databases can be used to reverse engineer the data lineage from systems. Whereas data lineage extractors actively connect to systems to collect their metadata, graph databases only represent a framework for storage and analysis, and interfaces to the systems from which the lineage information is to be extracted must be built.

Further, more detailed information on graph databases can be found in chapter (Bajer et al. 2021).

5 Conclusion

In this chapter we compare an ex-post reverse engineering approach with an ex-ante model-driven data lineage approach. The first connects to the systems in the IT landscape, extracts the data lineage information, and creates a model. In the latter, data and ETL processes are modeled and the physical implementation as well as the data lineage is generated from these models. This is the ideal scenario but is not always feasible. This can present a problem, since reverse-engineering approaches possibly do offer a complete picture of the data and its dependencies. If an organization has an incomplete overview of its data and data flows, it presents a high risk.

To improve the quality of the data lineage documentation, consumers of data lineage as well as software vendors play a role. In organizations with a complex IT infrastructure, processes must be optimized to facilitate the documentation of data lineage as the basis for the modeling process. This has several dimensions. For example, large and complex software routines should be avoided. Instead, processes should be subdivided into elementary building blocks and should be standardized as far as possible. Another example is the integration of a data lineage step in the organization's business processes. All business processes involving a modification of metadata should also include a step in which the data lineage information is updated. This is valid for automated extraction of data lineage, but perhaps even more crucial for updates of the data lineage where automation is not possible and manual updates must be made.

Software vendors also play a large role in improving data lineage documentation. For example, modern ETL tools offer the possibility to develop data pipelines without programming effort by using a user-friendly GUI. If this data pipeline is deployed, the program creates programming code. Ideally, data lineage information should also be generated in a graph-like structure and exposed to other programs via, for example, a REST API, or directly pushed into a central data lineage graph database. Tools such as Ab Initio (AB INITIO, n.d.) and SAP Profitability and Performance Management (SAP Profitability and Performance Management, n.d.) already offer the possibility to model ETL without the need for programming. These tools could be extended with a data lineage interface. If such an approach is not feasible, at the very least these tools should offer the possibility for other systems to connect to their metadata repository so that the user can reconstruct the data lineage themselves.

Literature

AB INITIO. n.d. *AB INITIO*. Accessed January 20, 2021. https://www.abinitio.com/en/.

Apache Atlas. n.d. *Apache Atlas*. Accessed January 20, 2021. https://atlas.apache.org/.

Bajer, Krystyna, Sascha Steltgens, Anne Seidlitz, and Bastian Wormuth. 2021. "Graph Databases." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

BCBS 239. 2013. *Basel Committee on Banking Supervision (BCBS) 239.* Accessed December 15, 2020. https://www.bis.org/publ/bcbs239.pdf.

Data Cross Roads. n.d. *Data Cross Roads.* Accessed December 15, 2020. https://datacrossroads.nl/2019/03/17/data-lineage-103/.

D-QUANTUM synabi. n.d. *D-QUANTUM synabi.* Accessed December 20, 2020. https://synabi.com/.

MANTA. n.d. *MANTA.* Accessed December 20, 2020. https://getmanta.com/.

Neo4j. n.d. *Neo4j.* Accessed December 17, 2020. https://neo4j.com.

Octopai. n.d. *Octopai.* Accessed December 17, 2020. https://www.octopai.com/.

SAP PowerDesigner. n.d. *SAP PowerDesigner.* Accessed December 18, 2020. https://www.sap.com/germany/products/powerdesigner-data-modeling-tools.html.

SAP Profitability and Performance Management. n.d. *SAP Profitability and Performance Management.* Accessed December 18, 2020. https://www.sap.com/netherlands/products/profitability-and-performance-management.html.

SAS Data Integration Studio. n.d. *SAS Data Integration Studio.* Accessed December 18, 2020. https://support.sas.com/en/software/data-integration-studio-support.html.

Digitization and MongoDB—The Art of Possible

Boris Bialek

1 Introduction

It is easy to argue that nearly every sector of the modern world connects in some way or another to the payment and financial industry. This is a multi-trillion dollar industry that has evolved with and just as quickly as technology has in the last two decades, virtually changing how we interact interpersonally, technologically, and financially, as consumers, customers, shareholders, etc. The financial industry is evolving and molding so quickly to consumer needs with media that did not even exist too long ago, such as mobile wallets, peer-to-peer payments, and ecommerce payments, that the market is ripe for disruption and change. Consumers only request greater convenience and higher security. As long as these needs are met, you can argue "anything goes."

These industry disruptions can now be called industry leaders, as they have been adopted as the new norm. Digital transactions happen faster and more easily, and leading startups are responding to changing expectations, with $213.5 billion invested into disruptive fintech companies. Since the publication of the initial Risk & Treasury solution sketches in "The Impact of

B. Bialek (✉)
MongoDB Inc., Zürich, Switzerland
e-mail: boris.bialek@mongodb.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_2

Digital Transformation" (Bialek 2019), MongoDB[1] has become the go-to data solution for financial technologies. Many fintech startups build directly on the MongoDB Cloud solution from the get-go. This is to be expected considering they have no legacy to be dependent upon, so they are in a privileged position to break into an established industry where the majority have been using the same database technology for easily the last 50+ years. For example, fintech companies such as Featurespace initiated their software using MongoDB to power their fraud detection platforms. Featurespace monitors individual behaviors and detects anomalies to identify risk and catch new fraud attacks in real time, as they happen. The software is deployed at highly companies such as HSBC, NETS, and over 200 other financial institutions. In parallel, other independent software vendors (ISVs) started to engage and now standard solutions of Finastra Payments, VolantePay, and even Temenos Transact (formerly Temenos T24) Core Banking are available with a MongoDB backend. Not to mention that naturally a large number of financial service providers quietly build their own in-house solutions and see those affiliations as differentiators.

The interesting part of the discussion is being able to distinguish the drivers for these financial companies to start working with the MongoDB document model and not go the "standard" route of the SQL model. After all, SQL has been the norm for financial banking and payment services for 50+ years, as we noted. Why change now? Is it worth it? When we consider software and applications, we have all witnessed personally how the frontend, design, graphics, and user experience of software have evolved swiftly, minimally, and beautifully over the last two decades, and we reap the benefits from this as we adopt an eye for the more aesthetically pleasing and functionally simplistic. With that in mind, it makes sense to drive this same evolution forward with the backend as well with regard to databases, especially when databases are the literal foundation of any function for every piece of code written. The document model MongoDB invented was the beginning of a data revolution currently taking place. The document model took off just as Big Data, data variety, and real-time analytics became crucial to how applications function.

So what are the drivers for MongoDB's cloud platform? They can be grouped into three major categories:

[1] For more detailed information on MongoDB, see MongoDB (n.d.).

(1) Organizational flexibility: The introduction of DevOps[2] and alignment of resources with functions (call it tribes, squads, etc.) are the antithesis of waterfall methodologies and long-term planning. Flexibility is a paramount requirement for success. Moving fast and with agility does not have to mean you "break everything" as you go, but rather it is simply the option to move quickly and safely in response to global trends and market patterns. This is essential for any tech organization.

(2) De-siloing existing applications: Solutions implemented as "packages" like ERP[3] that did not deliver the integrated approach originally promised but rather locked data and processes into a monolith. Breaking up these monoliths and becoming flexible again is considered the prerequisite for modernization and innovation. It allows for a more holistic approach to an organization, with easier insight and an overview of the ins and outs of your entire business.

(3) Journey to the cloud: The move away from one's own infrastructure and toward a flexible on-demand model seamlessly. The implementation of "on-board" cloud services provisioned through cloud provider marketplaces requires changes in the solution design, especially when it comes to utilization of cloud services.

2 Organizational Flexibility and Data Domains

Organizational flexibility has a significant impact on data and vice versa. This is because, without flexibility, your data remains stagnant, or at the very least slowed, and data that is a day, an hour, or sometimes even minutes delayed, if you are dealing with customer personalization or targeting, is enough to make your data useless. Fresh data serves a fundamentally critical function. If not used while it's hot, it becomes useless. If your organizational structure is not flexible enough to respond and adapt to continuously make the necessary changes to use data at its most relevant, then it can become frustrating very quickly.

[2] Describes an approach to improving collaboration between software development and IT operations. It is a portmanteau of Development and IT Operations. DevOps aim to shorten the system's development life cycle.

[3] Enterprise resource planning describes the task of planning, controlling, and managing resources in a timely manner and in line with the company's purpose.

The deconstruction of long-term organizations into actionable units is closely linked to the implementation of micro-service-based IT and fit-for-purpose updates, implementations, and changes. This requires a new concept for data from the get-go. Quarterly schema changes in a monolith application planned for months and then rolled out after more months of testing is not an option anymore for flexible, agile teams. The introduction of data domains that encapsulate a concise dataset not driven by the applications but by business logic and that can be shared across multiple teams synergistically is a necessary prerequisite.

MongoDB's document model allows various teams to share the most "critical resource data" while jointly collaborating on the same datasets and domains, all while being independent in their actions. It sounds obvious enough, however, it is quite a rare occurrence for organizations to collaborate seamlessly without data hiccups. All the teams agree they need a plan going forward, but execution makes it complicated if the database technology does not sync with their needs. As an example, say a client domain team would like to enhance payment information and enrich it with additional categories of spending given by the customer via their mobile application. In earlier days, the payment data store was deeply embedded inside the payment rails and changing that data would have been impossible as each transaction is distributed into dozens of tables which are in control of the payment vendor solution and can change with any release update. Therefore, the normal path of action would be to copy to another database where the data can then be enriched and stored for this function. This would also include introducing a new set of PCI[4] data, which would most likely need specific treatment.

With today's ISO20022 format[5] on payments and payment data stores implemented with MongoDB (as the smartest way to utilize a highly optimized JSON-based database where documents are defined and being generated by the payment process in JSON to begin with), the payment document can be enriched. This happens by a sub-document for "client categorization" not impacting the ongoing payment process. The section can actually change often, specifically at the beginning of the journey when the development team and the beta tester community drive agile changes on a daily basis. The same daily changes that would have otherwise been planned for months in advance, as is often the case for RDBMS[6] schemas, and then injected at

[4] Payment Card Industry Data Security Standard.

[5] ISO 20022 is an electronic data interchange standard used by financial institutions. ISO 20022 is sometimes referred to as UNIFI (UNIversal Financial Industry message scheme). The ISO 20020 message types are intended to replace the MT formats (Message Types, MT) currently used in the SWIFT community.

[6] Relational Database Management System (see Codd 1990).

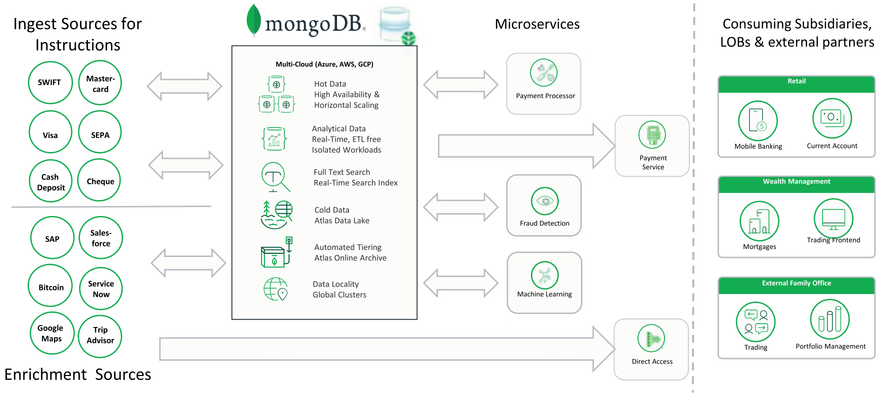


Fig. 1 MongoDB overview (© MongoDB)

specific maintenance dates. The idea of daily changes without any downtime is only possible through the concept of the document model.

What would this architecture design look like? Below you can see a general example (Fig. 1).

The central payment store in MongoDB consists of hot data which is the actual payments themselves, and this payment data store is highly available by design. MongoDB is implemented in a three or more nodes solution that allows distribution over multiple availability zones in the cloud or data centers on the premises. The standard deployment mechanism is a profile via Kubernetes submitted into the MongoDB Operations Manager or Cloud Manager to creation.

Additional resources can be identified for analytics through the MongoDB workload isolation design mechanism. This allows the decoupling of transactional data streaming with fixed response needs to secondary applications. In our example above, the secondary cases are data enrichment through additional sources and provisioning to downstream applications.

The MongoDB-based payment data store allows independent communication with the payment engine, which in the example above is Volante Pay. Additionally, real-time fraud detection, with Iguazio[7] as the data science platform used as an example above, gets connected through the same APIs and can act in parallel or in real-time event streaming depending on the requirements.

For additional search functionalities, MongoDB Atlas Search is available without additional implementation of custom code. MongoDB Atlas Search

[7] See iguazio (n.d.).

implements a fully deployed "Apache Lucene,"[8] which is the basis for several other commercial solutions such as Solr[9] or Elastic.[10] With MongoDB there is no need to copy data at any time between the operational data store and the search index. As a final step in completing the data life cycle, the online archive and the data lake for retention are integrated, too.

The various DevOps teams can access the data for their needs and provision their applications based on their requirements and different programming languages. GO[11] and RUST[12] for the mobile banking team and then Python for risk analytics, for example, have no impact whatsoever on the data store and can coexist. As the teams are flexible and agile, they can start the de-siloing of the monolithic solution stacks.

3 De-siloing Applications

MongoDB is the data solution that allows greater flexibility and agility, not being bound by a single data format or rigid ORM, but still holds "the best of relational features," such as rich secondary indexing, aggregations, and materialized views. It allows the fundamental combination of objects through joins and unions—the merge functions that SQL is built upon. Last but not least MongoDB comes with ACID compliance. ACID are the four defining attributes for any true database: Atomicity, Consistency, Isolation, and Durability. While these four nouns sound very sterile and academic, they simply mean that data remains in the state that a normal person would expect it to remain in. Basically, writes happen in a certain order and nothing ever gets lost. This is the default for any functioning data.

When de-siloing existing applications, the development team has the dilemma of changing code bases while continuing operations. While there is always the option of a "big bang" switch from old to new, this approach sounds exciting but is never reasonable when mission-critical operations are in play (think about the switch to an untested payment framework at a G-SIB[13]). The successful approach needs to be a decomposition of functions out of the existing environment and then the enablement of the new functionality

[8] See The Apache Software Foundation (2020).

[9] See The Apache Software Foundation.

[10] See Elasticsearch B.V. (n.d.).

[11] Go is an open-source programming language originally designed by Google (see Google 2020).

[12] Rust is a multiparadigm system programming language developed by Mozilla Research (see Rust Core Team 2020).

[13] Global Systemically Important Banks (see Financial Stability Board 2020).

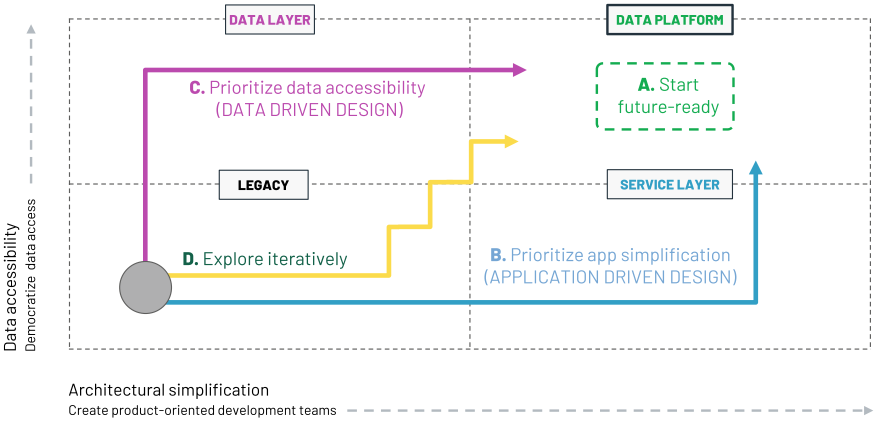

DATA LAYER
DATA PLATFORM
C. Prioritize data accessibility
(DATA DRIVEN DESIGN)
A. Start
future-ready
LEGACY
SERVICE LAYER
D. Explore iteratively
B. Prioritize app simplification
(APPLICATION DRIVEN DESIGN)
Data accessibility
Democratize data access
Architectural simplification
Create product-oriented development teams

Fig. 2 Architectural simplification process (© MongoDB)

without breaking the necessary, existing components. In case of an existing SQL, relational organized database, this would require constant adjustments of the schema and would inject unexpected results as schemata of 10,000 or more tables, not an uncommon scenario.

This is where MongoDB's document model enters as the savior. Documents can be flexibly adjusted, and schema is applied only where and when needed. In the case of ISO20022, data type correctness in payment documents is a requirement but additional new innovative enrichment data used for scoring through real-time machine learning can flexibly emerge and does not need to be in prescribed formats. MongoDB documents deliver the strong type data formats the developers use from the RDBMS world as well. This allows a rapid conversion from RDBMS typing to MongoDB documents. Additionally, MongoDB provides types and APIs for more advanced functions, like GIS[14] information, graph data, and even streaming inside the same document combined. MongoDB allows fraud detection and prevention ("What is the distance of the card to the merchant and is there a relationship between buyer and product group…") through a combination of GIS, graph, and payment information inside one single document versus slower access to multiple data sources, the combination and interpretation of those data, often in conjunction with type transformations, and then finally the analysis of data.

For this situation, there would be four options available—A, B, C, or D, shown in Fig. 2.

[14] Geographic Information System.

Option A, “starting future-ready” is, of course, ideal but unrealistic unless you are a fresh start-up starting from scratch as a completely new development. For anyone else, it would be impossible due to timing, complexity, and “big bang launch” risk that is frowned upon in the financial industry. More likely you carry some sort of legacy technology that is holding you back, difficult to manage, expensive to change, and too risky to change. Where do you go from there?

The architecture team tends to move to Option B, “Prioritizing app simplification,” by refracturing the application. The argument against this option is the existing technical debt and, in almost all cases, the solution architects have some very specific product challenges they would like to improve—only these often have nothing to do with de-siloing the entire system.

Option C, “Prioritizing data accessibility,” is all about first optimizing the data infrastructure, often introducing “the new next generation schema,” and ignoring the bridging between old and new.

MongoDB and the document model allow a fourth path, however, where smaller amounts of functions and their data get extracted out of the monolith, converted into a self-contained micro-service and the related application code gets modernized. This allows immediate innovation on the “released” data that is now available for other application purposes as well. The de-siloing has started, in this case. Often compared to the untangling of a ball of wool, the first service may take some thought processes and manual skills, but after the second and third service is established the team becomes skilled and the process accelerates.

The second part of the journey for de-siloing is the simple fact that MongoDB is polymorphic in its programming architecture and drivers. While this may sound rather mundane from a business user perspective, the developers had to maintain in RDBMS days so-called “Object Relational Mappers” (ORMs), which took care of the translation of business logic objects to the tables the RDBMS expected (admittedly PL/1 and Cobol did not have that problem when the solution of RDBMS was designed in the 1970s). These ORMs introduced another layer that prolonged the monolith life span.

Take, for example, Santander Spain, a globally recognized bank. It was stuck with a disjunct mortgage platform that did not allow a true digital transformation and experience. Untangling the solutions from the document submission for application, application processing, and underwriting to the actual mortgage processing took six months without taking the solution offline. The resulting system then was truly digital and now enables the new data services to be integrated into other solutions and functions, such as the mobile banking interface. The new digital solution reduced complexity by 33% versus the old monolith and was ranked #1 in the Aqmetrix[15] global ranking (Global Banking & Finance Review 2020). The whole transformation was only possible by applying the previously discussed path using the MongoDB document model. Since going live with the mortgage solution, the mobile banking application has been added to the same digital design of MongoDB documents as the foundation for data access, proving that the approach of iterative de-siloing does indeed work and can be a true accelerator for the digital journey in finance.

4 The Three Different Paths Leading to the Cloud(s)

The second driver for introducing MongoDB is the appearance of THE CLOUD. Moving to a cloud platform is seen by many as the ultimate path to digitization, but more often than not it actually increases problems instead of removing them. Basically, what this means is moving to the cloud can be a part of your modernization solution, rather than the modernization solution. If you are running on a legacy, siloed database and you digitize it to the cloud, basically you have the same infrastructure but in the cloud. Where is the modernization? Let us discuss the steps organizations are taking to digitize, with what is and is not working.

4.1 Lift and Shift

In 2015, the stampede to the cloud started out with a simple assumption: lifting workloads to the cloud makes everything easier and cheaper when it comes to IT. Moving existing solutions—even 20 years old—and moving the workloads from in-house datacenters (often virtualized) to the cloud introduced the concept of "lift and shift." The fact that there was zero innovation and the solutions still have the same architecture and monolithic challenges was ignored and even solutions vendors suddenly called their hosted solutions "cloud-ready." However, more often than not, these solutions were manually hosted services comparable to the ASP model of the 1990s. In contrast to expectations, the management of these hosted systems in the cloud was not easier and costs were skyrocketing as certain factors were not included in the initial planning.

[15] See Aqmetrix (n.d.).

In 2020, accelerated by the ongoing logistical challenges of the COVID-19 pandemic, companies stopped their lift and shift projects or concurred that the initial lift and shift was not bringing any benefits besides moving the servers somewhere else. A second wave of cloud improvements therefore had to follow fast, leading back to the de-silo and innovation discussion from earlier of the four paths to de-siloing. Plus, the data challenges, often related to Oracle and other legacy RDBMS, still needed to be addressed. Simply moving legacy RDBMS tables to cheaper open-source options sounded good in principle, but again the underlying silo structure and lock-up of data in monolith blocks needed a different solution.

4.2 The Cloud Provider Option

Beyond the original lift and shift option, the emergence of micro-services and encapsulated business services, for example, address lookups for validation, introduced "off-the-shelf" of functionality that various providers offer. Besides a broad variety of different databases, often cloned to avoid licensing and support requirements, services range from very basic data integration and ETL to advanced functions like AWS personalization that tries to offer a simplified machine learning environment for website personalization. Each of the three large cloud providers (Amazon, Azure, and Google) offers hundreds of services and subservices. While there are many arguments for the deployment of these cloud services, most arguments refer to their simplicity and how easy it is to jumpstart quickly with them; recent years have shown that there are some serious downsides to deep exploitation of these services.

Architects and developers often like to ignore the fact that these services are truly vendor proprietary. Some of them are taken off the market when not successful (remember Azure Cortina services?), while others are simply too complex and introduce a new class of maintenance needs that are unexpected. Plus, the introduction of DevOps principles does not remove operations per se, although that is what some think. A French tier-1 bank experienced this when ending up with 150 services inside a single component for its mobile banking including multiple database products for which it had no skills. Secondly, the proprietary nature of broad exploitation will not allow the fast-moving of the solution to a different cloud provider if this is mandated by the various regulators. Many regulators stipulate the requirement of being able to move a solution from one cloud provider back to premises or another cloud provider within 12 hours.

Looking at this problem from a data viewpoint, the introduction of different flavors of PostgreSQL,[16] as an often-touted database of choice, ignores the fact that cloud providers customized and changed the database to an extent that an easy move between two providers is impossible. Mentioning moving proprietary systems like AWS Aurora, Dynamo, Redshift, Google BigQuery, or the various flavors provided by Azure is generating interesting reactions in discussions. It is difficult to even connect these solutions INSIDE one single provider: going multi-cloud or cross-cloud is raising the complexity to stratospheric complexities. Security, key management, and interchange points are only some of the challenges to be mastered.

4.3 MongoDB Cloud—The Cloud Data Solution

The introduction of MongoDB Cloud[17] available on the three providers in one standard. Identical environment is the only offering compliant with many regulatory requirements taking the regional specifics, GDPR, and other privacy policies into account. MongoDB Cloud can even introduce a data solution across all three providers in a single cluster setup. Be it for availability purposes or to utilize different services from the providers across a joint dataset, MongoDB is flexible in its concepts and is the foundation for data services across a variety of use case areas, be it transactional or real-time analytics with machine learning enrichments. Additionally, the simplicity of the MongoDB Cloud is automating the whole data life cycle and data lineage flow. Data generated in transactional applications can be utilized in real-time analytics without any ETL or transformations thanks to the polymorphic data representation discussed earlier. Each document gets automatic expiration dates set through the so-called TTL indexing. After the given date arrives, the document is automatically moved from the active dataset to the Online Archive. In the Online Archive, the data is still accessible through federated data access. Afterward, the data can be moved to the long-term archive in the data lake where it resides until destruction, normally after seven years for most regulators.

Additional MongoDB is available on several other cloud providers' environments where the usual three providers are not available. Alibaba Apsara for MongoDB or TenCent Cloud are both available inside the China Big

[16] PostgreSQL is a free, object-relational database management system (ORDBMS) (see The PostgreSQL Global Development Group, n.d.). PostgreSQL is included in most Linux distributions. Apple ships PostgreSQL as the default database from the Mac OS X Lion (10.7) release onwards.

[17] See MongoDB (n.d.).

Firewall and allow the same MongoDB data foundation as deployed everywhere else. Yandex for Russia or OVH in many other locations are available as well. This allows the roll-out for branches that are not able to work on the usual providers and avoids redundant development or worst-case competitive disadvantages in specific markets through sup-par solution support.

5 Summary

MongoDB's unique combination of inherent flexibility from its data model, horizontally scalable architecture, and highly coveted developer usability make it the ideal fit for the complexities of payment data and data enrichment. Whether you are building from scratch and want strong tooling for handling transactional data across channels, or you are a mature enterprise looking to modernize and consolidate siloed and inaccessible transactional data stores, an agile, fast-moving document model adoption combined with MongoDB's powerful data platform helps derive the most value from your data.

Literature

Aqmetrix. n.d. *Aqmetrix.* Accessed December 15, 2020. https://www.aqmetrix.com/.

Bialek, Boris. 2019. "MongoDB—The Journey from a Relational to a Document-Based Database for FIS Balance Sheet Management." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Codd, Edgar F. 1990. *The Relational Model for Database Management.* Boston, MA: Addison-Wesley.

Elasticsearch B. V. n.d. *Elastic.* Accessed December 15, 2020. https://www.elastic.co/.

Financial Stability Board. 2020. *2020 List of Global Systemically Important Banks.* Basel: Financial Stability Board.

Global Banking & Finance Review. 2020. *Santander Best Bank for Digital Services.* Accessed December 15, 2020. https://www.globalbankingandfinance.com/santander-best-bank-for-digital-services-spain-2020-renews-its-app-to-make-it-more-personal-and-useful/.

Google. 2020. *Go.* Accessed December 15, 2020. https://golang.org/.

iguazio. n.d. *Bring Your Data Science to Life.* Accessed December 15, 2020. https://www.iguazio.com/.

MongoDB. n.d.-a. *MongoDB Cloud.* Accessed December 15, 2020. https://www.mongodb.com/cloud.

MongoDB. n.d.-b. *MongoDB. The Application Data Platform.* Accessed December 15, 2020. https://www.mongodb.com/.

Rust Core Team. 2020. *Rust.* Accessed December 15, 2020. https://www.rust-lang.org/.

The Apache Software Foundation. *Apache Solr.* Accessed December 15, 2020. https://solr.apache.org/.

The Apache Software Foundation. 2020. *Lucene.* Accessed December 15, 2020. https://lucene.apache.org/.

The PostgreSQL Global Development Group. n.d. *PostgreSQL: The World's Most Advanced Open Source Relational Database.* Accessed December 15, 2020. https://www.postgresql.org/.

Graph Databases

Krystyna Bajer, Anne Seidlitz, Sascha Steltgens,
and Bastian Wormuth

1 Introduction

One major challenge for companies today is to analyze their data and gain knowledge and competitive advantage through understanding the relationships, correlations, and connections between different kinds of data. Graph databases are a powerful tool to find complex and dynamic relationships in highly connected data. Connected data is data whose interpretation and value require an understanding of the ways in which its elements are related (Robinson et al. 2013). But what is a graph database? It is a database that uses the graph structure and stores data as nodes, edges, and properties. Graph databases are based on graph theory, which is a mathematical construct on how to describe objects and the relationships between them (see Sect. 1.1).

In a relational database, entities are stored in tables and the relationship between entities is realized by joining tables with common keys. When representing many relationships, this representation can get complex and rigid. Graphs are simpler, they consist of nodes (*math.: vertices*), which have properties and labels. Nodes are connected by relationships (*math.: edges*), which have a type, usually a direction, and can also have properties. Edges represent an abstraction that is not directly implemented in a relational

K. Bajer (✉) · A. Seidlitz · S. Steltgens · B. Wormuth
ifb SE, Grünwald, Germany
e-mail: Krystyna.Bajer@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_3

database. Consequently, a graph database has no rigid table structures and is highly performant when querying complex relationships (Liermann and Tieben 2021). However, no uniform language like SQL (Structured Query Language) for relational databases exists to query data from a graph database. A popular language is Cypher, which is used by the Neo4j graph database. Typical providers and query languages are summarized in Table 1.

1.1 Mathematical Background

In the eighteenth century, Leonhard Euler introduced the basic idea of graph theory by solving the famous "Königsberg bridges problem", which subsequently led to the concept of an Eulerian graph (Wilson 2013). The city of Königsberg is separated by the River Pregel. The two islands were connected to each other and to the two mainland portions of the city by seven bridges. Euler proved that there is no path through all parts of the city that would cross each of those bridges once. This is a classic example of an optimization problem in graph theory. Another one is the "travelling salesman problem", i.e., to complete a circuit of the shortest length in a graph. The first textbook in this field was written by Dénes König in 1936 (König 1936).[1]

In graph theory, a simple graph $G = (N, E)$ is an ordered pair N and $E.N$ is a set of *nodes (math.: vertices)* and E is a set of *edges*. An edge $(n, n^{'})$ joins the nodes n and $n^{'}$. The nodes n and $n^{'}$ are *end vertices* of this edge and are *adjacent* to one another.

The most popular graph model variant is the property graph, which is a directed[2] labeled multigraph where the edges are directed, nodes and edges are labeled and can have properties, and there can be multiple edges between any two nodes. Properties are key/value pairs that represent metadata for nodes and edges (Fletcher et al. 2018). Figure 1 in Sect. 2.1 shows a data model and an explicit database example.

A single-labeled directed multigraph can be described as a tuple $G = (N, E, L, \delta, \lambda_N, \lambda_E)$, where N is a finite set of nodes, E is a finite set of edges, and L is a finite set of labels. The edge function $\delta : E \rightarrow N^2$ associates edges with pairs of nodes, $\lambda_N : N \rightarrow L$ is the node labeling function and $\lambda_E : E \rightarrow L$ is the edge labeling function. An edge $(x, y) \in E(G)$ is represented as a triple $(v, w, v^{'})$, where $v = \lambda_N(n)$, $w = \lambda_E(e)$, and $v^{'} = \lambda_N(n^{'})$ (Fletcher et al. 2018). A path ρ in a graph G is defined as a sequence of edges $(v_0, w_0, v_1), (v_1, w_1, v_2), \ldots, (v_{m-1}, w_{m-1}, v_m)$, where

[1] For an English translation see König and Tutte, Theory of Finite and Infinite Graphs (2013).

[2] In a directed graph, the edges connecting two different nodes have different meanings, depending on their direction. In an undirected graph, an edge connecting two nodes has a single meaning.

Table 1 List of graph database technology providers (© ifb SE)

Company	Product	License	Query language	URL
Neo4j	Neo4j	Community Edition is GPLv3, commercial, and AGPLv3 options for enterprise and advanced editions	Cypher	www.neo4j.com
ArangoDB	ArangoDB	Free Apache 2, Proprietary	AQL	www.arangodb.com
OrientDB	OrientDB	Community Edition is Apache 2, Enterprise Edition is commercial	SQL-like	www.orientdb.com
Microsoft	Microsoft Azure Cosmos DB	Commercial	SQL-like	http://azure.microsoft.com/-services/-cosmos-db
Amazon	Amazon Neptune	Commercial	Gremlin, SPARQL	http://aws.amazon.com/neptune
IBM	IBM Graph	Commercial	SPARQL	https://www.ibm.com/uk-en/marketplace/graph
SAP	SAP HANA	Commercial	OpenCypher	www.sap.com
Oracle	Oracle Spatial and Graph	Proprietary	PGQL	https://www.oracle.com/

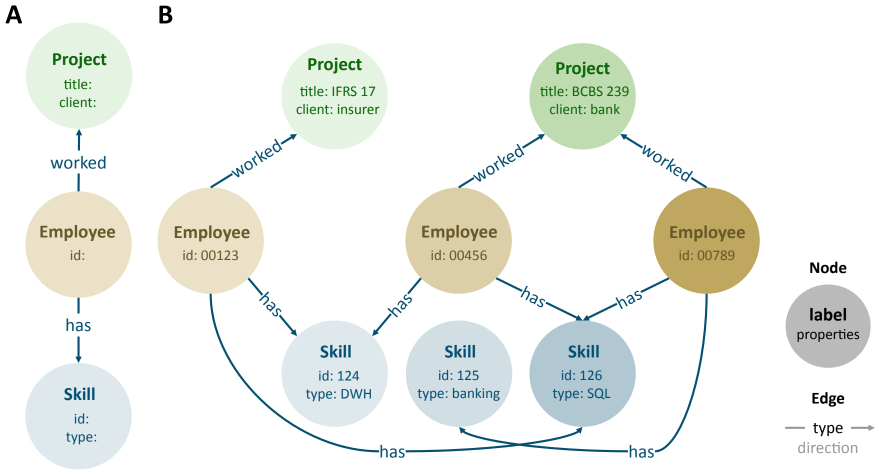


Fig. 1 Graph database model and example database (© ifb SE)

v_0 is the start node and v_m is the end note of the path. The label of ρ is the sequence of labels $w_1, w_2, \ldots, w_{m-1}$.

1.2 Graph Databases in Financial Services

Enterprises inside and outside of the financial sector want to analyze their data in an efficient way to gain the most possible knowledge and benefits. When the interesting information is in the relationship or connection in between data, for example, finding patterns indicating fraud, graph databases, and algorithms are an efficient tool for data analysis.

One prominent example is anti-money laundering. It is possible to model various entities involved, e.g., clients, accounts, transactions, and the relationship between these entities as a graph. By tracking how and where money is moving you can find patterns and detect money laundering behaviors. Section 4.1 discusses the "Panama Papers" as a special use case for detecting financial fraud in more detail.

Financial assets are very complex and substantial risks arising from mutual dependencies might be hidden, as the financial crisis in 2008 showed. To improve risk management, asset graphs can help to gain a better understanding of the relationships between assets and hence a clearer view of risks. They can also be used for real-time pricing of derivatives by considering the interdependencies among different assets, and therefore accurately reflect the risk and reward ratios (neo4j 2017).

A less obvious application for graph databases in the financial services sector is to handle information management with the help of a metadata graph. Data lineage and data flows are modeled as a graph for regulatory compliance and a complete picture of data and systems across the organization. It is possible to apply graphs for network and IT infrastructure monitoring to identify dependencies enabling network planning and impact and root cause analysis.

Other business areas where graph databases are already being used are master data management, identity and access management, and cybersecurity (Mathur 2020).

2 Technical Implementation

2.1 Data Model

Graph databases serve a large variety of purposes. In relational databases, which are the majority of databases not only in the financial sector, data is stored in normalized tables with columns and rows. Storing data in tables is viable for most types of data (especially data that is supposed to be aggregated or used to calculate KPIs) but may not be appropriate for other types of data occurring in the financial sector. If the data's main purpose is to be analyzed by its inherent connections, graph databases are the best choice. Whereas relational databases become inconvenient and slow if they are queried over a lot of database objects. Graph databases are built to support queries that span many database objects.

Nodes and edges have characteristics depending on the data model and information they are supposed to provide. The data model in Fig. 1A offers an impression of how data could be organized in a graph. In this case, the model defines characteristics for each database instance (nodes, edges). Nodes can have properties (key–value pairs) like IDs, a title, a type, other numeric values, etc., whereas edges have a type, a label, or any other numeric attribute[3] and also a direction.

Figure 1B shows a graph database of three employees (based on the data model in Fig. 1A), their skills, and the projects they worked in (nodes) as well as the employees' relationships (edges) to each type of node. It is imperative to the graph data which type of edges the data model allows. This example graph does not give any information about the relationships the nodes with type

[3] This refers to a weighted graph, where an edge can be weighted to qualify its weight or strength, e.g., cost, length, distance.

"Human" have with each other; those nodes are only indirectly connected. In order to display the relationships between the nodes of type "Human", it would be possible to add the edge of type "Colleague" to the data model, which would connect all three nodes of type "Human". A new type of node called "Employer" and a new edge of type "is employed" would, however, add more value than just the edge of type "Colleague". This is because the information "Employer" and "is employed" implies that all three nodes of type "Human" are colleagues, since they are employed at the same firm. But this new node and this new edge would additionally provide information about the employer.

A graph data model also carries information about the number of edges a single node of a particular type can have. In our example (Fig. 1B), the nodes of type "Human" can have multiple edges to nodes of type "Skill" and type "Project", but these cannot have any edge to any other type of nodes. If a "Project" node could have an edge to the "Human" type node, the graph database would increase in informational value, but it may lose its focus.

Obviously, the data model of any database is crucial for its value and purpose but in our example, the graph data model distinguishes a human-to-human relationship database from a skill- and experience-driven database. Hence, the data model of a graph database is imperative to its use case and small tweaks of the data model may add a disproportionate increase in information. This is because, in connected data, the nodes, whatever they may represent, are of less interest than the edges connecting the nodes. It is the edges that provide the interesting information. However, the bottom line is that, graph databases are a flexible storage technique for data that is highly connected and difficult to store in a relational model.

2.2 Storage

How graph databases are stored depends on the service provider. In general, a distinction is made between native and non-native storage of graph data. In a graph database where storage is native, each node and each edge are individual entities, somewhat comparable to object-oriented programming. In non-native storage graph databases, the data is stored in tables, introducing another layer of complexity to the interactions between database management system, database, and storage. Further, NoSQL storage types of graph databases include key–value storage or document-oriented databases. On a side note, most graph database providers utilize the ACID (atomicity, consistency, isolation, durability) set of properties for their database transactions. This ensures data validity despite errors, power failures, and other calamities.

2.3 Providers

The industry leader for graph databases is Neo4j (DB-Engines 2020), but Neo4j is most certainly not the only graph database technology provider. The company's competitors include Microsoft, Amazon, IBM, SAP, and Oracle, but also smaller firms like ArangoDB, OrientDB, Virtuoso, etc. Table 1 gives an overview of some of the technology providers.

2.4 Visualization of Graph Data

Unsurprisingly, visualization of graph data depends on the use case. A popular example is any navigation software, where information about the route to take is visualized on a map rather than presented as a table. To display graph data on a map leaflet.js (www.leafletjs.com) provides an open-source JavaScript library. Data representation on a map to visualize a route is obviously intuitive, as is the graphical display of nodes and edges corresponding to human interactions with anything. With these graphical views of graph data, it is easy to find highly connected nodes, recognize patterns, abnormalities, or other areas of interest. Providers like cytoscape.js (js.cytoscape.org), d3.js (d3js.org), neovis.js (for Neo4j graph databases only), and others offer software for graphical illustration of graph databases. Since a graph database can store any type of data, there are tools for visualizing graph data as charts, e.g., amCharts (amcharts.com), Chart.js (chartjs.org), and Tableau (tableau.com). Graph databases are also accessible with popular tools like R or Python. For R, Neo4j provides a package called neo4r including an API to access the Neo4j GraphDB with R. Several other packages help with visualization (e.g., visNetwork) of the data, making R a powerful and fully customizable tool to work with graph data in not only financial data science. For more information, please refer to (Liermann and Tieben 2021). Implementations for Python include NetworkX, a Python language software package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks (Klein 2020).

All of the above mentioned tools offer a large variety of options to display graph data. However, everything depends on the use case. As an example, with tools like Tableau, it is possible to display financial fraud data by country/state (Barrasa 2016). Visualizing the data as such would allow executives to easily adjust products to locally higher financial risk. Displaying any travel-related number is not a challenge either, but visualization of queries with complex results becomes incomprehensible as a graphical view. Meaning that as soon as there are many nodes and edges to be displayed, a graphical view might be useful for developers or analysts but might not be the representation of choice for executives. Graph data analysisis therefore a double-edged sword: graph databases are fast and enable the user to find connections between nodes that are complex to find but visualizing these query results remains a challenge.

2.5 User-Friendly Approach to Graph Databases

For customers that do want to set up a graph database but do not want to start from scratch or with an empty, ETL tool-like feeling, vendors such as Structr (Structr GmbH 2020) provide a graph database framework to effortlessly get started with graph databases. Such tools deliver a codeless approach understandable for everyone to graph data modeling as well as data analysis with graphs.

3 Analysis of Graph Databases

The analysis of simple graph databases can be carried out with the help of queries or with algorithms for complex graph databases. Graph query languages serve this purpose. In this section, different queries are presented, which can be grouped into adjacency, pattern matching, reachability, and analytical queries. In the analytical queries, algorithms are discussed in more detail, whereby three selected algorithms are presented. In the last section, a selection of graph query languages are introduced with a description and a note of which graph model they are applied to.

Adjacency Queries

The adjacent query refers to both edges and nodes. For example, in case of an edge between two nodes, these two nodes are adjacent. Equally, if two edges have a common node, these edges are adjacent. Frequently asked and investigated subjects would be whether node v is the neighbor of node $v^{'}$or who is the adjacent node to node v. Although these queries seem simple at first glance, they can be a challenge for sparse graphs.

To be able to define more complex neighborhood queries, the adjacency term can be extended (for more information Fletcher et al. [2018]).

For the adjacent query, it is possible to use an adjacency matrix. This matrix reflects relationships between the individual nodes. Each node has one row and one column, with n nodes a $n \times n$ matrix is created. In a simple graph, without edge weights and multiple edges, this matrix is a (0,1)-matrix with zeros on its diagonal. Due to the fact that this problem can be represented as a matrix, it is possible to apply methods of linear algebra.

Adjacency queries are particularly important in issues of influence. They are used to find individuals with similar interests (recommendation systems) or to obtain information. The adjacency matrix may further be used to calculate the path length in graphs. For example, assume a directed graph $G = (V, E)$ without edge weights or multiple edges with an adjacency matrix A. Then, the path length between node i and node j, which contains k edges, can be calculated by the power of the adjacency matrix A^k. Using A^k, it can be determined in row i and column j how many paths meet the requirement (Wikipedia 2020).

Pattern Matching Queries

This query searches a graph database for the set of all subgraphs that match a specific graph pattern. A simple example of such a searched graph pattern would be a small graph in which the edges and nodes are labeled by variables. The variables indicate unknown data and define the output of the query. In other words, values are assigned to these variables at the end of the query. An example would be the search for the colleagues $?x$, $?y$ of M1, who are also colleagues (Fletcher et al. 2018).

$$(M1, colleague, ?x), (M1, colleague, ?y), (?x, colleague, ?y)$$

where $?x$ and $?y$ are variables.

Pattern matching is used in areas such as within the pattern recognition field, to identify communities and social positions in social networks and in protein interaction networks.

Reachability Queries

The task of this query is to evaluate whether given nodes are connected by paths. They are also modeled as path or traversal problems in connection with graph databases, which allow restrictions by nodes and edges.

The areas in which the reachability queries are used range from social networks, discovering people with common interests, to biochemistry, to find specific biochemical paths between distant nodes. They are also used as a basis for the shortest path analysis.

Analytical Queries

Analytical queries measure quantitatively and usually in aggregated form topological features of the graph's database. They are supported either by ad hoc functions that hide complex algorithms or by special operators. Simple analytical queries include, for example, aggregate operators of query languages, such as i and max. These can be used to determine the number of nodes, the number of neighbors of a node (degree of a node), the length of a path, or the shortest path between two nodes. Algorithms are used for more complex analytical queries. In the following, a selection of such algorithms is presented (for more examples see (Hunger and Augsten 2020).

- **Shortest Path** calculates the shortest path of a node and all other nodes in a graph. Another name for the Shortest Path algorithm is the Dijkstra algorithm. It received this name from its developer, the Dutch computer scientist Edsger Dijkstra. This algorithm is used among others in Google Maps. In general, cities represent nodes and edges represent the road network. The edges of such a graph can also distinguish between motorway roads, country roads, and roads with toll costs. Thus, not only the shortest route but also the fastest or most cost-effective route can be determined.
- **PageRank** measures the importance of nodes in a graph. The importance of a node is measured by how many indirect and direct relationships point to a node. The more relations a node has, the higher its weight and the greater its effect. Google is famous for this algorithm. There, the search results are sorted according to their accumulated importance. Another example of the use of the PageRank algorithm can be found on Twitter and in biology to predict chain reactions within an ecological system.
- **Betweenness Centrality** is similar to the Shortest Path algorithm. It measures the number of shortest paths in a graph that run from a node. If a node is most often on the shortest paths, it forms a bridge between clusters and has a higher centrality value. Such nodes have a high influence on the information flow in a graph. This algorithm is mainly used for network analysis and fraud detection.

Many providers such as Neo4j, Orient, and Palantir offer tools for various applications, which take over the task of graph analysis. Neo4j has, for example, developed a tool (Privacy shield) which complies with the requirements of the European Union General Data Protection Regulation or a tool (Fraud Detection) for banks, insurance companies, and e-commerce to detect fraud (Sadowski and Rathle 2015).

3.1 Graph Query Languages

In comparison to relational databases, graph databases do not have a uniform language. This is because graph databases have no predefined standard. Over time, different query languages for different databases have developed (Fletcher et al. 2018).

Data query languages can be separated into the following groups: languages for edge-labeled graphs, languages for hypergraphs, languages for nested graphs, languages for property graphs, and RDF query language.

The most commonly used is Cypher, which is the query language of Neo4j and belongs to the category of property graphs. The easiest query in Cypher involves an expression with a clause START, MATCH, and RETURN, which can be used as follows:

```
START x=node:Employee(name="Name")
MATCH (x)-[:colleague]->(y)
RETURN y.name
```

In the example above, the name of a colleague of `x` is sought. The `START` command defines the node of the graph that is the start point. `Match` is used to find the pattern and `RETURN` specifies what the query should deliver. Further, Cypher can calculate paths from node `(a)` to node`(b)`. In the following example, only outgoing edges are recognized and the solution is stored in the path variable `p`.

```
p=(a)-[:knows*]->(b)
```

In addition, Cypher can compute specific operations on nodes, edges, attributes, and paths using built-in functions. In the example above, the shortest way from node `(a)` to `(b)` could be determined by the function `shortest Path(p)`. For more information about Cypher, please refer to (Liermann and Tieben 2021).

Table 2 contains additional graph query languageswith a description and grouping.

4 Business Use Cases

While Sect. 1.2 gave a rough overview of use cases for graph databases in the financial sector, this section focuses on the description of three explicit business use cases.

Table 2 Additional graph query languages with a description and grouping (© ifb SE)

Language	Description	Group
Gremlin	Open source-based query language for various graph databases (Neo4j, OrientDB, or DEX)	Languages for property graphs
SPARQL	W3C specified query language for RDF data models	RDF query languages
Blueprints	One of the first libraries created for the property graphs usable for different graph databases	Languages for property graphs
GraphQL	SQL-like query language	Languages for property graphs
Rexster	Multi-supported HTTP/REST interface for querying graph databases over the Internet	

4.1 Fraud Detection—Panama Papers

In 2016, the International Consortium of Investigative Journalists (ICIJ) exposed a highly connected network of illicit offshore bank accounts by analyzing a data leak of 2.6 terabytes consisting of 11.5 million leaked files from the Panamanian law firm Mossack Fonseca.

They used a graph database consisting of 840,000 nodes and 1.3 million relationships. The graph contains key entities such as "Company", "Client", "Intermediary", "Address", and "Officer" to reveal connections between these nodes. All these entities have a lot of properties, such as document numbers, share amounts, source ID, addresses, and citizenship. These entities have specific relationships, for example, an officer is the "director of" or "shareholder of" a company or has "similar name and address as" the intermediary. In this way, you can find suspicious relationships, such as companies that control other companies in the same country through a company in an offshore zone. An example is "Regula Limited", an offshore legal entity registered in the Bahamas and the British Virgin Islands which was a subsidiary of Deutsche Bank. By querying the node(s) "Regula Limited" and all the companies it is connected to, you can see that this entity serves as an officer of several other offshore legal entities, and also served as the "intermediary" for several offshore entities.

By analyzing the connections between the nodes, ICIJ revealed that roughly 500 banks had registered nearly 16,000 shell companies for Mossack Fonseca clients. The British banking giant HSBC and its subsidiaries alone account for more than 2300 of the companies (Woodman 2016).

4.2 Lufthansa—In-Flight Entertainment System Management

An unexpected use of graph databases is applied by Lufthansa's in-flight entertainment management system (see Wilmes 2013 for more information). In order to provide customers with in-flight entertainment, a plane carries storage and broadcast units. To update the entertainment content, a switch of storage devices (HDDs or SDDs) is necessary. How can storage devices, the planes carrying them as well as the planes' current location be quickly identified in order to prevent a movie from being shown to customers? Or how can movies be controlled if all necessary licenses for a freshly installed movie are present? For this task, Lufthansa chose a graph database to manage its in-flight digital assets and its licenses and build a dashboard for the ground personnel to quickly identify and replace a storage device while any plane is at an airport.

4.3 Navigation Systems

Navigation systems, such as Google Maps and many more, use different algorithms to guide from a starting point to an end point in the best possible way. Among others, the shortest path algorithm is used, like the Dijkstra algorithm (see Sect. 3) and variants of it. This segment shows how a navigation system works using the Dijkstra algorithm. Assuming that a city is referred to in this example, nodes are intersections and edges are roads. At the nodes are the points where you can choose a path (edge). It is important to note that not all edges are the same. When deciding which road to use, you choose the one that will lead you to your destination first. Several factors are considered in this decision, such as traffic lights, the size of the road, traffic volume, speed limits, and so on. To illustrate this in the graph, weights are assigned to the edges, which reflects the estimated travel time of this section. If the edge between node X and node Y is described with the number 4, the estimated distance between X and Y on this path takes four minutes. With the help of the Dijkstra algorithm the "minimum distance" is determined. Meaning that if a start and end point is given with weighted edges in the graph, the sum of the weights, in other words the travel time, is minimized, which is the time between start and end point (Moussa 2020; Sanders et al. 2007; Crovari 2019).

The algorithm works as follows. Each node has assigned "costs". This cost is the value of the minimum path to reach this node. Before the algorithm starts, each node has an infinite cost, as no path has been found to reach this node yet. The start node has the cost of zero; as you are already at the start there are no costs to get there. Furthermore, the algorithm has a list of all possible nodes through which the end point can be reached. The algorithm now repeats a certain process, first the one with the lowest cost is extracted from the list of nodes to be visited. The nodes that have not yet been visited but are also accessible are assigned costs. If the cost of this node is lower, the cost of the "cheaper" node is selected, otherwise the cost of the first node is kept. The visited node is marked as visited and the process starts from scratch until the destination is reached. Mathematically, it has been shown that the Dijkstra algorithm always finds the shortest route, including navigating over roads that would otherwise never have been considered. Information such as the traffic situation and traffic jams can be included in the calculation by adjusting the weights.[4]

5 Summary

We observe not only the establishment of specific software and consulting companies for graph technologies as relevant contributors to the IT landscape, but also major software players like IBM, SAP, and Oracle adding graph technologies to their standard portfolio. The constantly growing amount of information, often highly connected data from social networks or similar, increases the demand for tools to support the efficient storage and analysis of such data sets. Therefore, graph technologies are here to stay and will remain an important piece of equipment for the modern data scientist.

Literature

Barrasa, Jesús. 2016. *Graph DB +Data Virtualization =Live Dashboard for Fraud Analysis*, November 30. Accessed November 5, 2020. https://jbarrasa.com/2016/11/30/graph-db-data-virtualization-live-dashboard-for-fraud-analysis/.

Crovari, Pietro. 2019. *GOOGLE MAPS AND GRAPH THEORY.* Impactscool Magazine, May 20. Accessed November 9, 2020. https://magazine.impactscool.com/en/speciali/google-maps-e-la-teoria-dei-grafi/.

DB-Engines. 2020. *Ranking.* solid IT gmbh. Accessed November 6, 2020. https://db-engines.com/en/ranking.

[4] Example of Dijkstra algorithm: https://www.youtube.com/watch?v=UG7VmPWkJmA&feature=youtu.be&t=33.

Fletcher, George, Jan Hidders, and Josep Lluis Larriba-Pey. 2018. *Graph Data Management.* Springer.

Hunger, Michael, and Stephan Augsten. 2020. *Graph-Analytik.* Dev Insider, September 21. Accessed November 6, 2020. https://www.dev-insider.de/5-wichtige-graph-algorithmen-im-ueberblick-a-963343/.

Klein, Bernd. 2020. *Python Course.* Accessed December 4, 2020. https://www.python-course.eu/networkx.php.

König, Dénes. 1936. *Theorie Der Endlichen und Unendlichen Graphen: Kombinatorische Topologie Der Streckenkomplexe.* Chelsea.

König, Dénes, and W. T. Tutte. 2013. *Theory of Finite and Infinite Graphs.* Translated by Richard McCoart. Springer Science & Business Media.

Liermann, Volker, and Marian Tieben. 2021. "Use Case—NFR—Using GraphDB for Impact Graphs." In *The Digital Journey of Banking and Insurance, Volume II—Digitalization and Machine Learning*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Mathur, Nav. 2020. "Graph Technology for Financial Services." *neo4j.* Accessed November 12, 2020. https://neo4j.com/whitepapers/financial-services-neo4j/.

Moussa, Ramy. 2020. *How Google Maps Work: Fast Route Planning*, February 15. Accessed November 11, 2020. https://algorithmyou.com/2020/02/15/artificial-intelligence/how-google-maps-work-fast-route-planning/.

neo4j. 2017. *neo4j*, May 24. Accessed November 11, 2020. https://neo4j.com/blog/financial-services-neo4j-financial-asset-graphs/.

Robinson, Ian, Jim Webber, and Emil Eifrem. 2013. *Graph Databases.* O'Reilly Media, Inc.

Sadowski, Gorka, and Philip Rathle. 2015. *Fraud Detection Discovering Connections with Graph Database Technology.* Neo4J, January. Accessed December 16, 2020. https://neo4j.com/whitepapers/white-paper-fraud-detection/.

Sanders, Peter, Schultes, and Dominik. 2007. "Engineering Fast Route Planning Algorithms." *algo2.iti.kit.edu.* Springer-Verlag Berlin Heidelberg. Accessed December 17, 2020. http://algo2.iti.kit.edu/documents/routeplanning/weaOverview.pdf.

Structr GmbH. 2020. *Structr.* Accessed December 15, 2020. www.structr.com.

Wikipedia. 2020. "Adjazenzmatrix." *Wikipedia*, June 10. Accessed November 4, 2020. https://de.wikipedia.org/wiki/Adjazenzmatrix.

Wilmes, Michael. 2013. "Slideshare." *slideshare.net.* Accessed November 7, 2020. https://de.slideshare.net/neo4j/inflight-asset-management-with-neo4j-michael-wilmes-graphconnect-london-2013.

Wilson, Robin J. 2013. "History of Graph Theory." In *Handbook of Graph Theory*, edited by Jonathan L. Gross, Jay Yellen and Ping Zhang. CRC Press.

Woodman, Spencer. 2016. *Global Banks Team with Law Firms to Help the Wealthy Hide Assets.* Accessed November 9, 2020. https://www.icij.org/investigations/panama-papers/20160404-banks-lawyers-hide-assets/.

Data Tiering Options with SAP HANA and Usage in a Hadoop Scenario

Michael Morawski and Georg Schmidt

1 Motivation

The rise of in-memory databases slike SAP HANA is widely discussed and how this can be part of a Big Data IT landscape working together with Hadoop to use the best of both worlds. The developments of recent years, the basic terms and technologies are described briefly. The focus of the main section will be on using Hadoop as a cheap data sink for SAP HANA using the Spark Controller. Other Hadoop-HANA connection alternatives are explored as well.

In the last few decades, the prices for RAM dropped significantly, which enabled more and more companies to switch from a traditional RDBMS to in-memory databases like SAP HANA. As the TCO has become affordable and an increasing number of SAP solutions are not only adjusted and optimized for HANA but are now designed for HANA (e.g., S4/HANA), this step toward in-memory database technology is the next logical step for most of the companies to renew their IT landscape (Kopic et al. 2019). The number of S4/HANA subscribers between Q1 2015 and Q1 2020 has

M. Morawski (✉) · G. Schmidt
ifb SE, Grünwald, Germany
e-mail: Michael.Morawski@ifb-group.com

G. Schmidt
e-mail: Georg.Schmidt@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_4

increased by a factor of 38, which proves that the market has recognized the benefits of SAP HANA (Holst 2020). The switch to SAP HANA also provides new possibilities regarding data mining, such as ad hoc queries, as reading from RAM is significantly faster than reading from disk. Nevertheless, RAM is still more expensive than traditional disk storage, which is a pain point when one of the use cases of the company is the storage of massive amounts of data and conducting analyzes of this data. Organizations that use the RAM memory efficiently and keep only the information required in-memory make the performance improvement last. In such a scenario, a data aging and tiering strategy comes into play.

1.1 Trends in Data Technology

The trend toward in-memory databases like SAP HANA meets another ongoing topic of the last couple of years: Big Data (Fig. 1).

Big Data is one of the current topics that appeared several years ago and still has a continuously high level of interest, which does not seem to be declining. The reason is that Big Data is the foundation for running machine learning or enabling data science. These have been on the rise for the last five years and have got high potential to change the way business decisions are prepared and taken. In 2001, Doug Laney first presented what distinguishes Big Data from other data processing: the three Vs—volume, velocity,

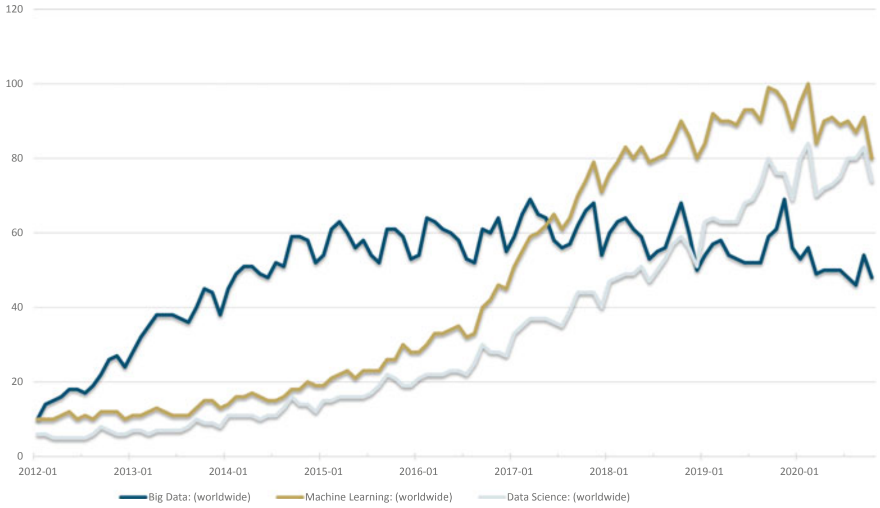


Fig. 1 Interest in the topics Big Data, machine learning, and data science (*Source of data* Google 2020)

and variety (Akhtar 2018). The estimated **volume** of data in 2020 is about 40 ZB (Zettabytes, 40 billion Terabytes) (Gantz 2012). Of this, about a third is believed to have a Big Data value, meaning that this information can contribute to generating new insight and is worth being stored and processed. The term **velocity** addresses fast data generation and the ability to store this. The amount of generated content per day is especially big in social media: 4 Petabytes of new data are uploaded to Facebook, 500 million tweets are posted on Twitter, 0.7 million hours of videos are uploaded to YouTube (Vuleta 2020). This also shows the different kinds of formats that are stored—pictures, videos, texts: these are all examples of unstructured data and are addressed by the term **variety** in the Big Data definition. This definition of Big Data simply describes the technical ability to store and handle huge amounts of data coming from various sources. Therefore, the responsibility for the topic of "Big Data" in a company normally lies with the CIO and the IT department.

Handling the first three Vs is the technical basis for **data science**, which aims at the processing of already existing and new data sources to support business decisions and develop new fields of business. Data science is normally driven by the business department and adds five more Vs to the Big Data definition as it is more about handling and processing data. For example, the **veracity** must be proven, such as data quality, trustworthiness, biases, noise, and abnormality of data. The **variability** can be an issue as the same data could have different meanings in a different context. In such a scenario, a business glossary can help, accompanied by a data governance strategy. In a Big Data scenario especially, the **volatility** is quite high, leading to the consequence of having a guideline on how long the data is valid and stored. The retention time for data should directly be linked to the **value** of the data. For this purpose, the best approach is to categorize the data into three classes: strategic, tactical, and operational data (Singer 2017) (Fig. 2).

For instance, running an ATM as a bank at a certain location produces different types of data. Strategic data from an ATM would be the daily cash amount withdrawn from the ATM leading to the long-term decision to shut down the ATM or not. Operational data would be sensor data to indicate that the ATM is running out of cash or has a defect. This type of data has got a low half-life and mostly indicates that immediate action is needed. As the value of data decreases over time at different speeds, the retention times should reflect this fact. Another aspect of data science in Big Data is that you cannot see the wood for the trees, meaning that it is important to have an eye on the **visualization** to present the amounts of data in a comprehensible and multidimensional view that is easy for management to understand.

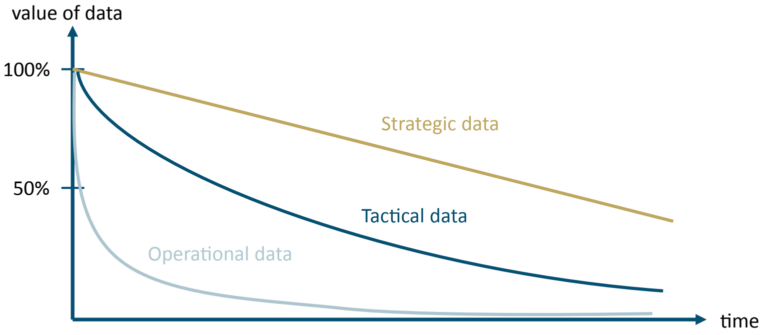


Fig. 2 Value of data over time (© ifb SE)

For **machine learning**, Big Data provides massive amounts of input data to train and test a machine learning model, which then may carry out an automated screening of result data as an advanced internal control system (ICS), serve as fraud detection/prevention, or support automated business decisions. Many more use cases are currently being discussed and developed, which of course is a whole field of its own and is currently hyped with the term “Industry 4.0.”

1.2 SAP HANA and Hadoop: Best of Both Worlds

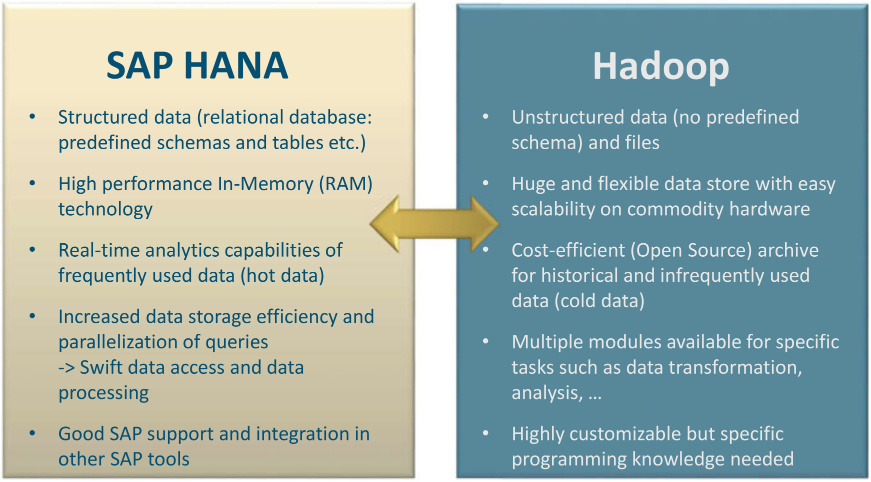
Regarding Big Data, Hadoop has become one of the most widespread solutions. The word “Hadoop” in this article does not refer to the narrow sense only including Hadoop Distributed File System (HDFS) and the MapReduce algorithm, but subsumes the ecosystem of open source tools provided by the Apache Foundation and third parties (Akhgarnush et al. 2019). These modules cover a wide range of use cases, such as authorizations (Ranger), data governance (Atlas), NoSQL databases (Cassandra, HBase), in-memory data processing (Spark), or mimicking relational databases (Hive) (Du 2018).

To cope with the huge amounts of data in a Big Data scenario, Hadoop applies **schema on read**, which enables it to store huge amounts of data in the cluster, also providing the additional possibility to store unstructured or semi-structured data. The data processing and the schema application can then take place afterwards. Hadoop provides the flexibility to cover a data lake and data warehouse scenario. SAP HANA applies the opposite approach. As a relational database, it works with predefined schemas and tables applying a **schema on write** to ensure the consistency of the data model. This makes the reading from tables faster than the schema on read in addition to the usage of

RAM. Whereas Hadoop makes use of cheaper disk storage, which is cheaper and more often faster to expand (Fig. 3).

The upgrade approaches of SAP HANA and Hadoop differ. SAP HANA has **vertical scaling** (scale-up), meaning that the hardware must be upgraded (e.g., faster CPU or more/faster RAM) and a scale-out option, whereas Hadoop makes use of **horizontal scaling** (scale out) by simply adding additional nodes to the cluster to store more data and process these. Hadoop scales linear with the number of nodes in the cluster. Both ways have pros and cons. For example, scaling up has the downside of a possible lock-in effect (manufacturer dependency), which limits expandability and software support in the future and is no long-term solution as the servers can only be upgraded to their performance limits. Horizontal scaling has the disadvantage of larger space requirements and higher electricity and cooling costs, but it offers the flexibility of adding resources when needed as well as the possibility of vertical scaling at a later point in time.

As huge amounts of data can be written to Hadoop and processed, it can serve multiple roles in a Big Data usage scenario together with SAP HANA. Besides using Hadoop for **storage** with its huge disk capacity of the cluster nodes, it can also serve as a **processing engine** to utilize the CPU and RAM of each worker node in the cluster. With this approach, CPU-intensive calculations (e.g., Monte Carlo simulations [Wu and Pan 2018]) can be outsourced to the Hadoop cluster. It is even possible to have a breathing cluster size, so-called elastic scale, especially if the Hadoop cluster is hosted



SAP HANA

- Structured data (relational database: predefined schemas and tables etc.)
- High performance In-Memory (RAM) technology
- Real-time analytics capabilities of frequently used data (hot data)
- Increased data storage efficiency and parallelization of queries -> Swift data access and data processing
- Good SAP support and integration in other SAP tools

Hadoop

- Unstructured data (no predefined schema) and files
- Huge and flexible data store with easy scalability on commodity hardware
- Cost-efficient (Open Source) archive for historical and infrequently used data (cold data)
- Multiple modules available for specific tasks such as data transformation, analysis, …
- Highly customizable but specific programming knowledge needed

Fast data processing and analysis combined with **massive, cost-efficient data storage**

Fig. 3 SAP HANA and Hadoop: the best of both worlds (© ifb SE)

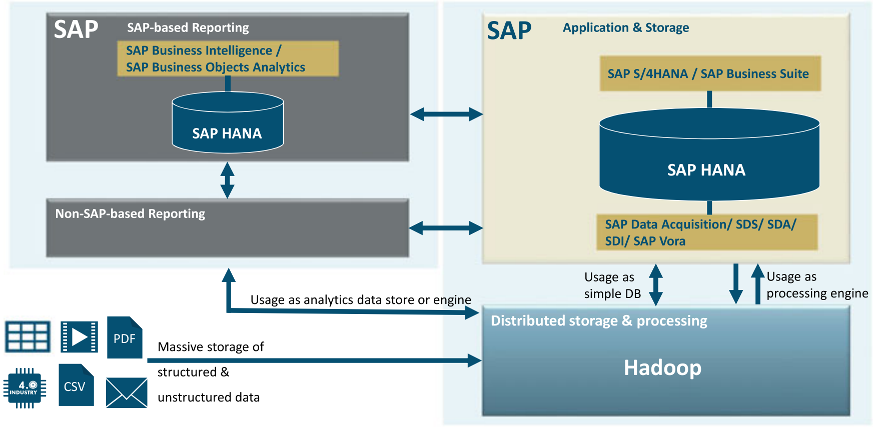

SAP
SAP-based Reporting
SAP Business Intelligence / SAP Business Objects Analytics
SAP HANA
Non-SAP-based Reporting
SAP
Application & Storage
SAP S/4HANA / SAP Business Suite
SAP HANA
SAP Data Acquisition/ SDS/ SDA/ SDI/ SAP Vora
Usage as simple DB
Usage as processing engine
Usage as analytics data store or engine
PDF
CSV
4.0 INDUSTRY
Massive storage of structured & unstructured data
Distributed storage & processing
Hadoop

Fig. 4 SAP HANA and Hadoop side by side (© ifb SE)

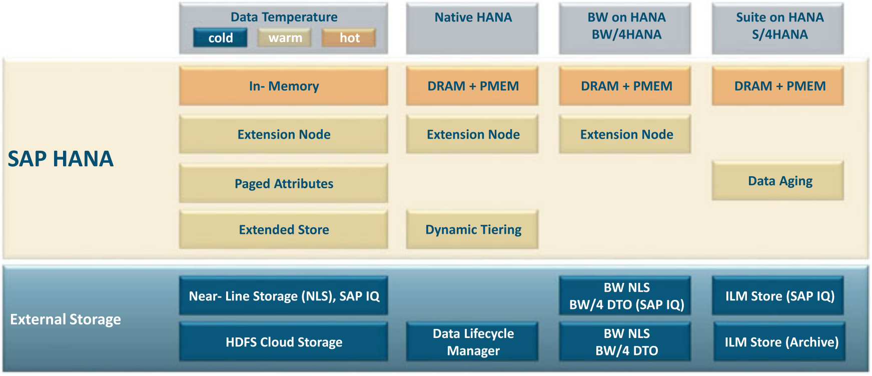
in the cloud. In on-peak hours, the number of cluster nodes is increased to serve the higher demand, e.g., in a batch-driven ETL load, and then will be shut down afterwards making use of the on-demand pricing policy of the cloud service providers (Fig. 4).

As Hadoop HDFS can serve as a **simple and cheap data sink** to outsource data, it is an attractive candidate for storing the cold data of SAP HANA which is not used often. This is because it can store various data formats of unstructured data (pictures, movies, audio, files), semi-structured data (e-mails), and structured data (tables from other databases). In the following sections, an overview will be given of different data tiering options in SAP HANA and a deeper look will be taken at data tiering with SAP HANA and Hadoop using the Data Lifecycle Manager with the Spark Controller.

2 Data Tiering

2.1 Overview Data Tiering Options in SAP HANA

As already mentioned, a company running a Big Data scenario faces a trade-off between having easy and performant access to the actual data with a high business value and the costs of storing these huge amounts of data. Therefore, a data lifecycle management strategy must be in place to break the circle of accumulating growth of data and the expansive hardware needed for storage without losing the insights from them. This strategy should be set up by the business department as it can assign the business value to the data stored in



	Data Temperature cold warm hot	Native HANA	BW on HANA BW/4HANA	Suite on HANA S/4HANA
SAP HANA	In- Memory	DRAM + PMEM	DRAM + PMEM	DRAM + PMEM
	Extension Node	Extension Node	Extension Node	
	Paged Attributes			Data Aging
	Extended Store	Dynamic Tiering		
External Storage	Near- Line Storage (NLS), SAP IQ		BW NLS BW/4 DTO (SAP IQ)	ILM Store (SAP IQ)
	HDFS Cloud Storage	Data Lifecycle Manager	BW NLS BW/4 DTO	ILM Store (Archive)

Fig. 5 Data tiering options for different HANA-based applications. The combination of native HANA and Data Lifecycle Manager will be discussed further in the following sections (© ifb SE)

the system and must live with the consequences of lower performance. As the value of data decreases over time (see Fig. 2), old operational data is not normally accessed at a high frequency (cold data), which makes it a perfect candidate for outsourcing to a system with a lower TCO, but still having the possibility to access the data for reporting insights, if needed. Different data tiering options exist for each data temperature depending on the SAP HANA application run by the company (Ruediger 2019).

The difference between tiering and archiving is that tiered data can still be accessed by the application via SQL, whereas archived data is removed from the database and stored in a separate file and can only be accessed in read-only mode. Later, the archived data can be deleted or reimported to the database (if needed) according to the information lifecycle strategy of the company (Fig. 5).

2.1.1 Hot Data Tiering

SAP HANA stores its hot in-memory in **DRAM** (dynamic random-access memory) which can be extended by **PMEM** (persistent memory) having the advantage of lower TCO (Claussen 2018). The data in PMEM stays in-memory after a system shutdown in contrast to DRAM. It is a convenient way to increase the amount of hot data, but is not a replacement for DRAM, which is still required in the SAP HANA system. New data records and data that are accessed often are held in-memory and therefore considered hot data.

2.1.2 Warm Data Tiering

Warm data is not often used (mainly for lookups) and updated sporadically. HANA native and BW on HANA and BW/4HANA offer the possibility of **extension nodes** which make use of the HANA scale-out feature (SAP 2020c). The SAP HANA scale-out landscape consists of one master node and several slave nodes. One or many (starting with only one is recommended) slave nodes can be used as extension nodes. This node of the HANA landscape is exclusively reserved for warm data. Extension nodes use the memory in an efficient way as they can use more than 50% of the HANA database allocation limit (up to 200%) (SAP 2019). A hot HANA database node must normally reserve half of its memory for the processing of data besides storage. Extension nodes provide more storage capacity by loading and unloading data not fitting into memory to disk. They can also use weaker and fewer CPUs for the same amount of memory in comparison to a hot data node. This makes the extension nodes cheaper for storage and slower than a hot data node. In a HANA native scenario, the customer application must take care of the categorization of warm data, the partitions specification, and access patterns. For BW, the data distribution, partitioning, and access paths are fully controlled by the application. So, for BW, this is the preferred solution for warm data storage. Extension nodes are covered by the HANA standard license.

A HANA native scenario provides another option for warm data storage: **Dynamic Tiering**. So, what is the difference between this and extension nodes? Extension nodes provide the complete functional scope of SAP HANA whereas Dynamic Tiering lacks some features, such as geospatial and time series functions. Extension Nodes should provide higher performance than Dynamic Tiering if access patterns and partitions specifications are taken care of. Dynamic Tiering is a component of SAP HANA that offloads warm data to disk-based extended storage. In this scenario, there are three kinds of tables: column tables located in-memory (hot data) and extended tables (warm data) stored on disk. The warm data in this extended storage (extended tables) is available for queries, updates, and deletions and can be joined with hot in-memory data. These tables are managed by the HANA data catalog. And there are multistore tables which are located partition-wise in-memory and the extended store. Multistore tables support partition pruning to improve performance and optimize the loading of data into memory. This enables an easy partition-wise movement of data from one storage type to the other with SQL statements or the Data Lifecycle Manager (DLM), which you will see in Sect. 2.2.3.

The SAP Business Suite on HANA and the S/4HANA applications offer an exclusive solution to deal with the different data temperatures: **Data Aging** needs to be activated and is supported with SAP NetWeaver 7.4 SP05 onwards (SAP 2020e). It does not need a separate license and is part of the framework from this version onwards. The prerequisite that data of a table participates in Data Aging is that the table is part of at least one Data Aging Object (maintained via transaction: DAGOBJ) and part of one Partition Object (maintained via transaction DAGPTC). As soon as the table is assigned for Data Aging an additional column (_DATAAGING of type DATA_TEMPERATURE (YYYYMMDD)) is added automatically which is the basis for the time selection partitioning (aging). From this column, the data temperature can be derived as only the hot data has the value "00000000" and horizontally partitions the data. In a normal scenario the hot data is accessed because this data is in a separate partition loaded into memory. If needed, it is possible to set the data temperature context allowing access to all data, hot data, or data with a specific temperature (greater or equal to a certain date). An Aging Run must be scheduled (transaction DAGRUN) to move the data to another partition according to the date in the data aging column which is actively set by the application, taking into account the business point of view, status, existence checks, and cross-object dependencies of the object (Jarkowski 2016). All non-hot partitions are marked as page-loadable, meaning that the columns are defined as **Paged Attributes**. This provides a warm tier solution allowing the users to load and unload table columns page by page from the HANA persistence layer into memory.

2.1.3 Cold Data Tiering

Like for warm data, there are different options per SAP HANA application. All cold data tiering options rely on external storage providing lower TCO. For native HANA applications, the Data Lifecycle Manager (DLM) is used to store data in the cloud or Hadoop. The benefit is that the tool takes care of a rule-based data movement between the stores.

BW/4HANA and BW on HANA have a data tiering concept called **Data tiering Optimization** (DTO) based on advanced Data Store Objects (aDSO) which provides the user the possibility to set up partitions depending on the data temperature (Witzel, SAP BW/4HANA Data Tiering Optimization 2018). For the outsourcing of cold data, Near Line Storage (NLS) is used which can be SAP IQ to stay in the SAP universe. An open-source alternative is Hadoop connected via the Spark SQL Adapter (SAP 2020b). The NLS is customized as an external connection in the transaction SM59. When setting up an aDSO there is the possibility to specify which data temperatures are stored in this object. The data temperature of the data in this aDSO is derived by a date field, such as the reporting year. With the help of ranges on this date field, the data is split into different data temperatures. For instance, data of the current year is considered hot data and kept in-memory, the previous year is stored on the Extension Node (see Sect. 2.1.2) as warm data and the rest is outsourced to the NLS. For the outsourcing to the external system, a Data tiering Optimization job must be scheduled on a regular basis. The frequency of this job should be aligned with the partition ranges (= size of partition). Fine granular partitions should be avoided as well (Witzel, Data Tiering Optimization with SAP BW/4HANA 2020). Typically, the number of partitions is the product of available HANA scale-out nodes and the number of ranges. If an aDSO holds high amounts of data, it can be helpful to split the aDSO into multiple aDSOs using Semantic Groups as the splitting condition before applying the DTO split.

SAP S4/HANA and the Suite on HANA can make use of the **ILM** Store to outsource cold data to an external database (SAP IQ) or filesystem like HDFS on Hadoop. ILM is part of SAP Netweaver and the external storage needs to be set up during customization. ILM Store uses the WebDAV interface standard for the connection and fulfils the BC-ILM 3.0 requirements for ILM solutions. This also includes the transmission of minimum and maximum expiry dates and legal holds (enforcing non-deletability for audit reasons) to the database for full control of the data lifecycle (SAP 2020d). It also supports other use cases besides data archiving (active data and system):

- Retention management (end-of-life data)
- System decommissioning (end-of-life system).

For audit or reporting reasons, the data must be accessed via ILM (Retention Warehouse) to be able to report on the data via BW or HANA views (Piramanayagam 2017).

2.2 Data Tiering with Hadoop and Data Lifecycle Manager (Native HANA)

2.2.1 Spark Controller

To understand the evolution of the HANA-Hadoop connection, a short history of the Spark Controller will be given. With HANA SPS06, SAP provided the possibility to expose Hive tables as virtual tables via SDA (Smart Data Access) (Abyson 2017). This data could be joined with data in SAP HANA creating calculation views on top of Hive and HANA tables. The downside was that the data on the HANA side is in-memory whereas the Hive data is stored on disk. As soon as a query is started joining HANA and Hive data, an on-the-fly MapReduce job is executed on the Hadoop side which takes some time to finish, especially if several Hive tables are to be joined. Therefore, with SPS07, SAP provided the feature to materialize the Hadoop data as a Hive table. This materialized table can then be used for the join with the HANA data in the calculation view. It eased the pain, but for critical queries an ETL load from Hadoop to HANA could be a solution. Another speed-up was provided with SPS08 and 09 when it was possible to use the MapReduce functions in vUDFs (virtual User Defined Function) to bypass the Hive layer. The imbalance memory vs. disk between HANA and Hadoop was removed when Apache Spark entered the stage and gained huge attention, for example, by winning the Gray Sort Benchmark in 2014 against a Hadoop cluster ten times the size in a third of the time (Sort Benchmark 2020). This gain in performance became possible by storing data in-memory enabling real-time data processing. With SPS10, SAP introduced the Spark Controller to connect HANA and Hadoop. The Spark Controller code is written in Scala—one of the three languages natively supported by Apache Spark besides Java and Python. The Spark Controller is installed on the Hadoop nodes and can be deployed via the cluster manager (e.g., Ambari or Cloudera Manager) and supports HANA-specific query execution and data transfer in a columnar format. Secure communication enables the user to read and write data from/to Hadoop.

The data stored in the Hive metastore or the Spark Catalog is visible as virtual tables through the remote source interface of SDA. This means that you can use the modeler perspective in HANA Studio or the Web IDE to consume and visualize the data in Hadoop. Therefore, the user can consume all the data available in Hive tables, an RDD (Resilient Distributed Dataset) or a Data Frame. The Spark Controller provides a unidirectional connection from Hadoop to SAP HANA (described in Fig. 6 as Hive Metastore Scenario) (SAP 2020a).

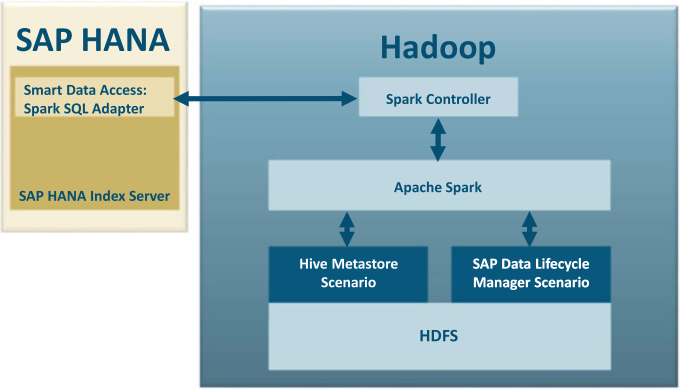


Fig. 6 HANA-Hadoop connection with the Spark Controller (© ifb SE)

When using the Data Lifecycle Manager (DLM), the cold data can be accessed by SAP HANA through the Spark Controller. The outsourced cold data is stored in HDFS files and the data is not readable for other users in the Hadoop universe to be compliant with data protection. The rules to determine the data temperature are completely managed in the Data Lifecycle Manager (see Sect. 2.2.3).

2.2.2 SAP Vora

To continue the evolution of the HANA-Hadoop connection, SAP developed SAP HANA Vora, which was renamed SAP Vora in March 2017 as a stand-alone product (Arcangeli 2019). SAP Vora is an in-memory query engine which leverages and extends the Apache Spark execution framework to provide interactive analytics on Hadoop in combination with SAP HANA. Therefore, Vora makes use of YARN and Spark and enhances Spark SQL to provide a bidirectional connectivity between HANA and Hadoop. This means that it is not only possible to consume data from Hadoop via SDA but also HANA data in Spark on the Hadoop side with the help of Vora. These features mainly address data scientists, to enable them to swiftly combine data from both systems without setting up an ETL process with data doubling for explorative data insights and business models. As Hadoop can also store unstructured data, Vora supports browsing through HDFS via a data browser. Like in HANA Studio, Vora also provides an SQL editor for ad hoc queries and an OLAP modeler to create new tables and views. Additionally, it also has a Graph, Times Series, and Document Store engine which enhance business analysts' possibilities to handle and store different data formats. SAP

Vora supports the main Hadoop distributions and is faster than the Spark Controller as it runs natively on each node in the cluster and has a deeper integration to run compiled queries for accelerated data processing. Vora can also be used to connect SAP HANA to the Data Lifecycle Manager to outsource cold data to Hadoop.

SAP Vora was integrated into SAP Data Hub as the Big Data data flow solution. In March 2020, SAP released SAP Data Intelligence 3.0 as the product succeeding SAP Data Hub 2.7. SAP Data Intelligence 3.0 can be hosted in the cloud and on-premises. It combines the capabilities of SAP Data Hub and SAP Leonardo Machine Learning Foundation. Therefore, SAP Data Intelligence addresses the three on-trend topics mentioned in Sect. 1.1.

2.2.3 Data Lifecycle Manager and Data Export from SAP HANA to Hadoop

The Data Lifecycle Manager (DLM) is part of the SAP HANA Data Warehousing Foundation and is used for native HANA applications to move data from one temperature store to another and finally outsource cold data to external storage like Hadoop or SAP IQ in an automated way without losing the possibility to access data. The external storage destinations must be set up once and on the HANA side, a separate schema with pruning views is created automatically to address the cold data in queries. Afterwards, a DML profile must be set up to specify the source and target of the data and create a DLM Data Movement Rule (compiles in the background into a HANA stored procedure) on which the data temperature is derived (typically a date). Two DLM profiles can be combined to achieve multitiering. The first DLM profile moves hot data partition-wise to a Dynamic Tiering Node or Extension Node as hot/warm data management. The second DLM profile can then be used to manage the movement of the table partitions from the warm storage to the cold storage destination. It is also possible to configure Persistence Objects in DLM to move groups of tables to a different data temperature in a consistent way.

DLM has a web-based user frontend and is available for both HANA XS-Classic and XS-Advanced Application Server stacks (Meier 2018). Figure 7 shows a high-level overview of the storage destinations for an XS-Classic setup. DLM with an XS-Advanced application server does not support Dynamic Tiering as warm storage and connects to Hadoop as cold storage via Spark Controller or SAP Data Hub (Vora).

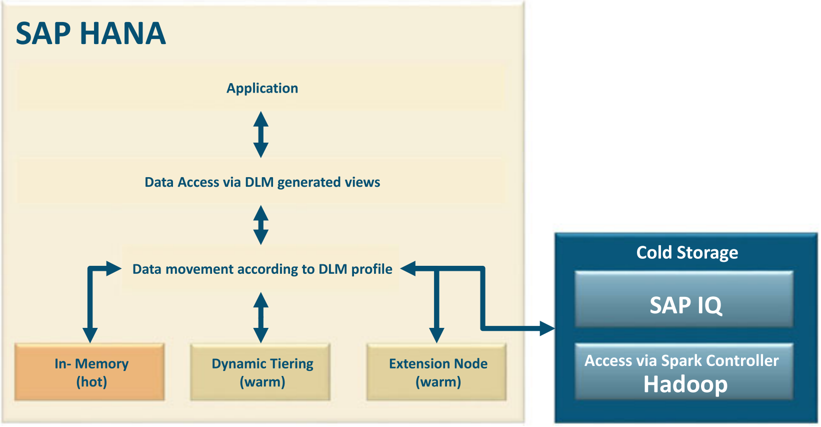
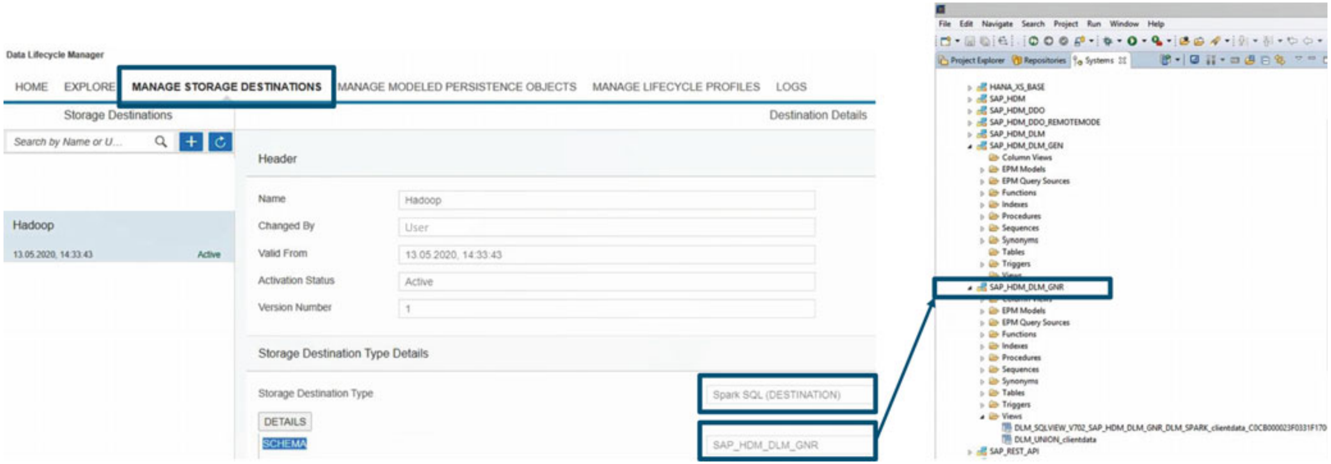

SAP HANA
Application
Data Access via DLM generated views
Data movement according to DLM profile
In- Memory
(hot)
Dynamic Tiering
(warm)
Extension Node
(warm)
Cold Storage
SAP IQ
Access via Spark Controller
Hadoop

Fig. 7 DLM storage options in XS-Classic scenario (© ifb SE)

For a look and feel insight into data tiering with HANA and Hadoop, we used a HANA 2.0 with SP04 and XSA in combination with a Cloudera distribution CDH 6.1.1. The connection between both systems was established with the Spark Controller which has been installed separately.

After setting up Hadoop as cold storage, DLM automatically creates a schema in which the union and pruning views are located to combine hot and cold data for later usage (see Fig. 8). The DLM profile specifies the table for which data should be tiered, the generated union and pruning view and the rule which should be applied to determine the data temperature (see Fig. 9). Also, the distribution of records in hot and cold storage is displayed. After a change in the DLM rule, a forecast is made to show the future distribution of records in both stores. The actual movement of affected table rows takes place during a DLM run. The Trigger Type (scheduled or manual) of the DLM run is part of the DLM profile customizing. Run logging takes place to indicate if all rows have been moved successfully. With the parameter “Relocation Direction” in the DLM profile, the data movement (hot → cold, cold → hot, or bidirectional) can be specified which happens during a DLM run. With the option “bidirectional” it is possible to relocate data from Hadoop to HANA if needed. In this case, a table of artificial client data with 100,000 records was used.



Hadoop
13.05.2020, 14:33:43 Active

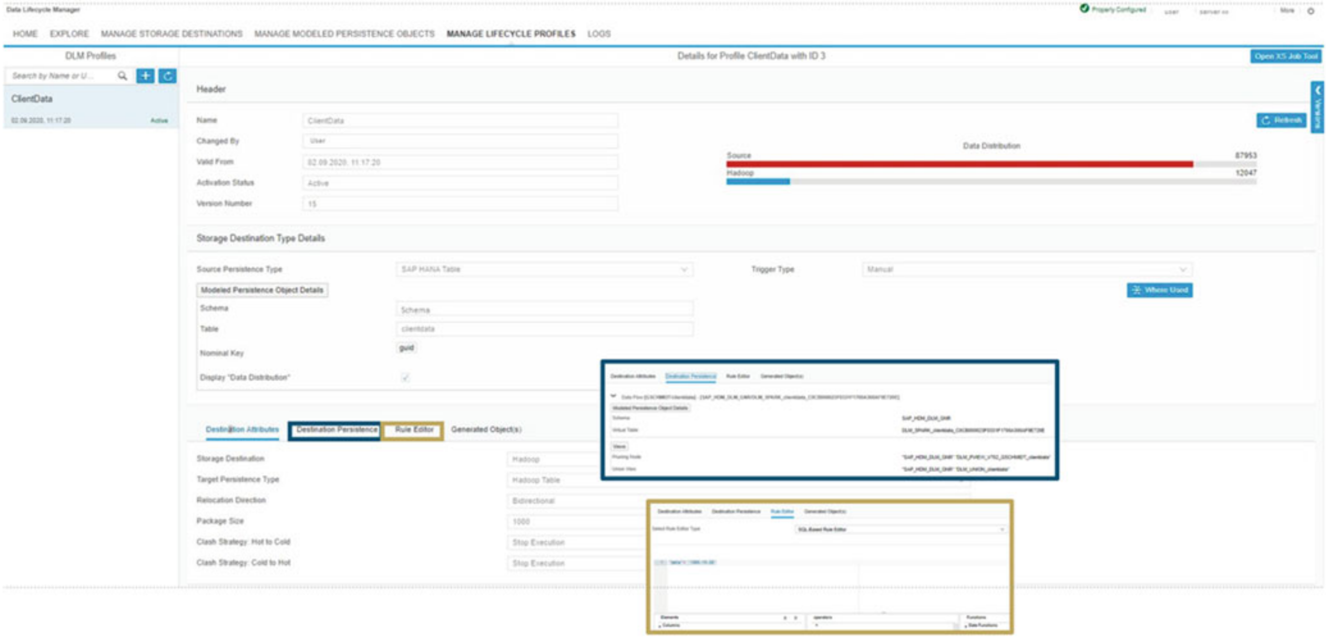
Header

Name	Hadoop
Changed By	User
Valid From	13.05.2020, 14:33:43
Activation Status	Active
Version Number	1

Storage Destination Type Details

Storage Destination Type	Spark SQL (DESTINATION)
SCHEMA	SAP_HDM_DLM_GNR

Fig. 8 DLM storage destination setup with Spark Controller and generated schema on HANA side (© ifb SE)



Details for Profile ClientData with ID 3

ClientData
02.09.2020, 11:17:20 Active

Header

Name: ClientData
Changed By: User
Valid From: 02.09.2020, 11:17:20
Activation Status: Active
Version Number: 15

Source 87953
Hadoop 12047

Storage Destination Type Details

Source Persistence Type: SAP HANA Table
Trigger Type: Manual

Modeled Persistence Object Details

Schema: Schema
Table: clientdata
Nominal Key: guid
Display "Data Distribution": ✓

Destination Attributes | Destination Persistence | Rule Editor | Generated Object(s)

Storage Destination: Hadoop
Target Persistence Type: Hadoop Table
Relocation Direction: Bidirectional
Package Size: 1000
Clash Strategy: Hot to Cold: Stop Execution
Clash Strategy: Cold to Hot: Stop Execution

Fig. 9 DLM profile setup based on a HANA table and a rule-based data export (© ifb SE)

3 Conclusion

There are several methods of data tiering for HANA-based solutions depending on the application on top of it. For companies running SAP HANA and Hadoop, this combination is a good tandem to cope with the challenges of Big Data. SAP is also aiming for more integration of HANA and Hadoop in the future, in order to have connections in both directions with Vora in its SAP Data Intelligence tool, which also integrates other trends in data technology, such as data science and machine learning. As storage with low TCO, Hadoop is a good add-on to SAP HANA and in a data tiering scenario it can be used to outsource cold data, but still enables access to the data for reporting or historic analysis purposes. With the Data Lifecycle Manager, it is possible to automate the data movement with transparent rules on a regular basis to maximize the value of hot storage in SAP HANA and still have access to all data on demand. Having the benefit of both different technologies to hand will improve performance on the database and still generate the value and insights from the huge volume of data organizations must deal with rather than just scaling up the system.

Literature

Abyson, Joseph. 2017. *Understanding the Basics of Big Data, Hadoop and SAP 'HANA' VORA*, July 18. Accessed September 10, 2020. https://blogs.sap.com/2017/07/18/understanding-the-basics-of-big-data-hadoop-and-sap-vora/.

Akhgarnush, Eljar, Lars Broeckers, and Thorsten Jakoby. 2019. "Hadoop—A Standard Framework for Computer Clusters." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Akhtar, Syed Muhammad Fahad. 2018. *Big Data Architect's Handbook.* Birmingham: Packt Publishing Ltd.

Arcangeli, Silvio. 2019. *SAP Data Hub and SAP Data Intelligence: Streamlining Data-Driven Intelligence Across the Enterprise*, July 17. Accessed September 10, 2020. https://blogs.sap.com/2019/07/17/sap-data-hub-and-sap-data-intelligence-streamlining-data-driven-intelligence-across-the-enterprise/.

Claussen, Courtney. 2018. *SAP's Data Tiering Options for Data Accessibility, Performance and Cost Management*, August 7. Accessed September 10, 2020. https://blogs.sap.com/2018/08/07/saps-data-tiering-options-for-data-accessibility-performance-and-cost-management/.

Du, Dayong. 2018. *Apache Hive Essentials.* Birmingham: Packt Publishing Ltd.

Gantz, John F. 2012. *The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East*, December. Accessed August 17, 2020. https://www.emc.com/leadership/digital-universe/2012iview/index.htm.

Google. 2020. *Google Trends*, October 7. Accessed October 7, 2020. https://trends.google.de/trends/explore?date=all&q=big%20data,machine%20learning,data%20science.

Holst, Arne. 2020. *Quarterly Number of SAP S/4HANA Subscribers Worldwide, from 2014 to 2020*, June 23. Accessed September 5, 2020. https://www.statista.com/statistics/590976/sap-hana-s4hana-customer-numbers/.

Jarkowski, Bartosz. 2016. *How to Perform Data Aging in S/4HANA*, October 3. Accessed September 5, 2020. https://blogs.sap.com/2016/10/03/how-to-perform-data-aging-in-s4hana/.

Kopic, Eva, Bezu Teschome, Thomas Schneider, Ralph Steurer, and Sascha Florin. 2019. "In-Memory Databases and Their Impact on Our (Future) Organizations." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Meier, Axel. 2018. *Understanding the SAP Data Lifecycle Manager (DLM) Tool*, December 24. Accessed September 10, 2020. https://blogs.sap.com/2018/12/24/understanding-the-sap-data-lifecycle-manager-dlm-tool-2/.

Piramanayagam, Muthusamy. 2017. "Unified Legacy Decommissioning Using SAP Information Lifecycle Management." *SAP*, October 16. Accessed September 10, 2020. https://blogs.sap.com/2017/10/16/unified-legacy-decommissioning-using-sap-information-lifecycle-management/.

Ruediger, Karl. 2019. "Recommended Data Tiering Approaches for SAP and Native Applications." *SAP*, March 7. Accessed September 10, 2020. https://blogs.sap.com/2019/03/07/recommended-data-tiering-approaches-for-sap-and-native-applications/.

SAP. 2019. "SAP HANA Extension Nodes—Frequently Asked Questions." *SAP*, August 30. Accessed September 10, 2020. https://www.sap.com/documents/2018/05/9878c71f-037d-0010-87a3-c30de2ffd8ff.html.

———. 2020a. "Getting Started with SAP HANA Spark Controller." *SAP Help Portal*, September 11. Accessed September 11, 2020. https://help.sap.com/viewer/6437091bdb1145d9be06aeec79f06363/2.0.3.1/en-US/9213000d628547f9b6c09693cf412be7.html.

———. 2020b. "Hadoop as a Near-Line Storage Solution." *SAP Help Portal*, September 10. Accessed September 10, 2020. https://help.sap.com/viewer/107a6e8a38b74ede94c833ca3b7b6f51/2.0.3/en-US/d935c9e9866b413693c72f7841b3b459.html.

———. 2020c. "SAP HANA–Native Storage Extension." *SAP*, June. Accessed September 10, 2020. https://www.sap.com/documents/2019/09/4475a0dd-637d-0010-87a3-c30de2ffd8ff.html.

———. 2020d. "SAP Information Lifecycle Management." *SAP Help Portal*, September 10. Accessed September 10, 2020. https://help.sap.com/doc/c3b6ed a797634474b7a3aac5a48e84d5/1610%20002/en-US/frameset.htm?7fe188e04fdd462e8ec330bb80efc389.html.

———. 2020e. "SAP Note 2416490—FAQ: SAP HANA Data Aging in SAP S/4HANA." *SAP*, June 23. Accessed September 10, 2020. https://launchpad.support.sap.com/#/notes/2416490.

Singer, Michael. 2017. "The Half-Life of Data and the Role of Analytics." *Oracle*, February 22. Accessed August 28, 2020. https://blogs.oracle.com/analytics/the-half-life-of-data-and-the-role-of-analytics.

Sort Benchmark. 2020. "Sort Benchmark Home Page." *Sort Benchmark*, September 10. Accessed September 10, 2020. http://sortbenchmark.org/.

Vuleta, Branka. 2020. "How Much Data Is Created Every Day?" *Seedscientific*, January 30. Accessed September 21, 2020. https://seedscientific.com/how-much-data-is-created-every-day/.

Witzel, Gordon. 2018. "SAP BW/4HANA Data Tiering Optimization." *SAP*, January 3. Accessed September 10, 2020. https://blogs.sap.com/2018/01/03/sap-bw4hana-data-tiering-optimization/?preview_id=586084.

———. 2020. "Data Tiering Optimization with SAP BW/4HANA." *SAP*, March 2. Accessed September 10, 2020. https://www.sap.com/documents/2017/12/b8ff485c-e47c-0010-82c7-eda71af511fa.html.

Wu, Minglei, and Jingchang Pan. 2018. *Research on Monte Carlo Application Based on Hadoop*, February 2. Accessed August 27, 2020. https://doi.org/10.1051/itmconf/20181703021.

Streaming

Data streaming was already popular in the first decade of our century, but its popularity has skyrocketed since the open-source availability of Kafka. Originally developed by LinkedIn, the streaming framework was donated to the Apache Software Foundation in 2012. Now branded Apache Kafka, it is used for processing and storing data streams. Kafka provides an interface for loading and exporting data streams to third-party systems.

While pure streaming systems have certain drawbacks, some interesting architecture patterns have evolved through the combination of the need for streaming technology and stability. The new architecture patterns, such as Lambda, Kappa and Delta Lake, are expressions of the business needs.

Although the applications of streaming technology, are located in the area of outside digitalization, it is obvious that this technology will impact the internal data processing architecture (inside digitalization). The low latency and real-time ability will first strike and build a business case in the customer context but will soon be followed by internal leveraging of speed improvements in the streaming platform, which is necessary for the external needs.

The part starts with a chapter introducing the streaming framework Apache Kafka and giving an overview of the fundamentals provided by the framework (Steurer 2021). The chapter discusses the most common challenges posed by standard software and how Apache Kafka can overcome these obstacles. The chapter ends with a line-up of use cases illustrating the abilities of this framework.

The second chapter of this part (Krätz and Morawski, Data Infrastructures—Lambda Architecture and Other Architectures 2021) introduces the major architecture patterns combining streaming and batch technology. The chapter explains in which contexts streaming components deliver added value to an institute. This explanation is followed by a comprehensive discussion of the components in a Lambda and Kappa architecture. In addition to a comparison of the architecture patterns Lambda and Kappa, the latest enhancements in streaming architecture patterns are discussed.

The last chapter of the part (Liermann , Li and Steurer, Kafka—A Practical Implementation of Intraday Liquidity Risk Management 2021) offers a practical example of a streaming architecture in the context of intraday liquidity risk management. A machine learning model, developed to predict the daily cumulative flow of specific clients, is embedded into a streaming architecture based on Apache Kafka. Real-time projections are visualized in an R-Shiny dynamic dashboard.[1]

Literature

Krätz, Dennis, and Michael Morawski. 2021. "Architecture Patterns—Batch & Real Time Capabilities." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Liermann, Volker, and Sangmeng Li. 2021. "Dynamic Dashboards." In *The Digital Journey of Banking and Insurance, Volume II—Digitalization and Machine Learning*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Liermann, Volker, Sangmeng Li, and Ralph Steurer. 2021. "Kafka—A Practical Implementation of Intraday Liquidity Risk Management." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Steurer, Ralph. 2021. "Kafka—Real-Time Streaming for the Finance Industry." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

[1] Details of dynamic dashboards are available in (Liermann and Li, Dynamic Dashboards 2021).

Kafka: Real-Time Streaming for the Finance Industry

Ralph Steurer

1 Introduction

Based on our experience, many companies often choose standard software because they are looking for best practices and out-of-the-box functionality. They also do not want to invest in custom development. However, we often found that standard software could not fulfill all requirements and a significant amount of effort was spent on extending the software. We also encountered architecture landscapes that consisted of many data silos, manual workarounds, and a wide variety of technologies. This was often the cause of slow performance and the inability to implement new features quickly. These challenges are of course not only limited to standard software.

Apache Kafka (we will refer to Apache Kafka from here on as just Kafka) is an open-source real-time event streaming platform originally developed by LinkedIn (see Apache Software Foundation 2020). It is now part of the Apache Software Foundation, with Confluent (see Confluent 2020) as the main contributor. It was designed from the beginning to handle massive data volumes in real time. The wide adoption of Kafka by many well-known tech companies (Uber, Netflix, LinkedIn, Lyft, etc.) certainly proves this. Over the years, Kafka also became very popular among financial services companies as

R. Steurer (✉)
ifb International AG, Zürich, Switzerland
e-mail: Ralph.Steurer@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_5

they also had to cope with increasing data volumes and required real-time processing.

Kafka offers messaging, processing, and storage of events all in one platform. Common technical challenges that many use cases face can therefore be solved in one central place. Kafka can also connect to many different data sources and integrate with business applications. This makes Kafka a very powerful platform for developing many different types of use cases in a cost-efficient way. In addition, Kafka can also be combined with standard software in order to solve the challenges mentioned at the beginning. For example, high-volume data processing or data integration can be built in Kafka while standard software is used for running best practice business logic.

1.1 Structure of the Article

In order to understand the benefits of Kafka for the finance industry, we first need to take a look at some of the challenges that exist with many IT architectures we encountered. After a quick tour of the Kafka fundamentals, we will deep dive into a sample set of use cases that are ideally suited to Kafka.

2 Finance Industry's Common Challenges with Standard Software

2.1 Limited Functionality and Negative User Experience

Based on our experience, many companies often choose standard software because they are looking for best practices and out-of-the-box functionality. They also do not want to invest in custom development. The reality that we encountered was sometimes a bit different, however. In many cases, standard software could not provide all required functionality because the use cases were very company-specific. As a result, a lot of effort was sometimes spent on enhancing the solution and closing the functional gap, including custom development.

Being able to get real-time results is an important requirement for many business users as it can provide a competitive advantage. However, results can often only be produced in long-running batch jobs as the whole architecture was not designed to be event-driven. This also makes it impossible to perform ad hoc simulations, for example.

2.2 Cost

Once a company has invested heavily in a certain type of standard software, it can become very challenging to use other software as well. Companies cannot choose the best tool for the job anymore but are "locked-in" with one vendor. This dependency makes it difficult to escape increasing license fees.

An additional major cost factor is the implementation project. Applications with business content often come with very complex configuration options as the software needs to be highly generic. The configuration effort can therefore be quite substantial, requiring highly specialized consultants. In addition, the cost of closing the functional gap through custom development is often underestimated, as previously explained.

Once an application is live, additional costs occur for maintenance and upgrade. Depending on the amount of custom development, upgrading to the latest release can be quite time-consuming. With increasing data volume, costs for hardware and data management increase. As horizontal scaling (adding more low-cost servers) is often not possible in an easy way, more expensive vertical scaling (adding more RAM, CPU, etc., to the same server) is applied instead.

2.3 Inability to Adapt to Changes Quickly

Not every company needs results in real time, one might argue. This argument is often used to justify deficits in application architectures that consist of many data silos, manual workarounds, and a wide variety of technologies. A complex architecture makes it not just difficult to produce real-time results but also to implement new features quickly. The more complex an IT architecture is, the more time it takes to implement changes or new requirements. Even small changes often require extensive testing for the following reasons:

- Custom enhancements only work for use cases that were known at the time of development. New use cases might require redevelopment of those enhancements
- Lack of unit tests
- Unclear dependencies due to complex architecture
- Complex data extraction procedures to obtain required test data as there is no central storage.

2.4 Data Silos

Data silos often prevent companies from using the full potential of their data to generate new business offerings for their customers. From our experience, small changes like requesting a new data attribute in reporting can end up in multiple change requests and in some instances even in a new project. This is often caused by the complexity of the interface changes.

Sometimes, a business warehouse solution is used to collect data from source systems and to provide a central data platform for business applications. These solutions mostly get their data from batch jobs running overnight and not in real time. They are also often used to implement application-specific logic, data aggregation, error handling, and other custom functionality. As a result, these business warehouses can become very complex and difficult to maintain with increasing data volume.

3 Kafka Fundamentals—A Quick Tour

Technically, Kafka is a distributed system, consisting of several nodes called brokers. Brokers store events in topics, which are immutable commit logs. Topics can be partitioned to enable parallel processing. A topic partition stores events in the order they were written. This is very important, for instance, in bank transaction processing where account balances must be checked. Kafka requires comparatively little memory due to its use of the operating system's disk cache. This makes it a very cost-efficient technology.

Events (often called messages) consist of a key and a value. There are no direct foreign key constraints to other topics like in a relational database. Like many other NoSQL database technologies, this simplified (or denormalized) data model makes Kafka much easier to scale. Kafka is designed from the core to be horizontally scalable, which means that you can simply add more commodity hardware to linearly increase performance. In contrast, many traditional database systems offer only (or at least recommend) vertical scaling. The reason is that, with strongly normalized data models, it is easier to just add more CPUs or RAM to a server than to connect multiple ones together and still be able to manage all the table dependencies.

Another important feature of Kafka is that it scales independently from the microservices that implement business logic. The Kafka cluster itself only stores data in topic partitions. You can use any deployment infrastructure to run your microservices and connect them to Kafka via Kafka client library (available for Java and other popular languages).

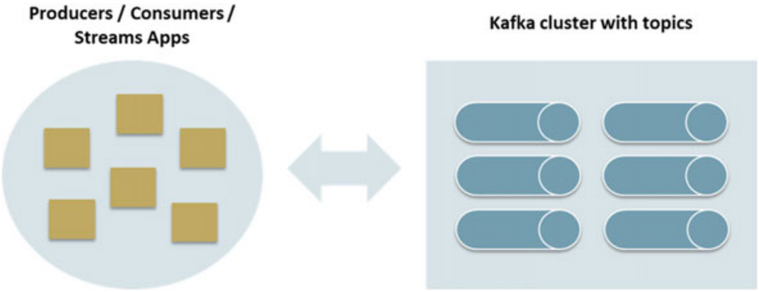

Producers / Consumers / Streams Apps
Kafka cluster with topics

Fig. 1 Kafka overview (© ifb SE)

Kafka Connect is another important feature. It allows you to connect many different databases and applications with Kafka. It provides standard connectors for SAP, Oracle, and other well-known software vendors. It also provides CDC (change data capture) to replicate databases into Kafka in real time.

Microservices that interact with a Kafka cluster are divided into three categories. Producers, Consumers, and Streams applications (a combination of the former two). Java is the most common language to implement business logic for Kafka, but other popular languages are possible as well. Kafka Streams DSL offers a rich set of data transformation operations. With just a few lines of coding, you can do joins, aggregations, windowing, and many other operations. Kafka Streams also provides so-called state stores (embedded Rocks DB) for stateful operations. They can also be used to cache results and expose them via REST API to other applications. State stores are fault-tolerant and provide an in-memory buffer for fast key-value access (Fig. 1).

Fault tolerance in Kafka is achieved by replicating topic partitions over available brokers in a Kafka cluster as depicted in the diagram below. Only the leader replica of each partition (shown in dark blue) is actively used while the other replicas provide redundancy. Kafka automatically redistributes data in case one instance of a microservice or broker fails (Fig. 2).

Kafka offers three types of messaging semantic: At least once, at most once, and exactly once. For finance applications, exactly once semantic is especially important as it ensures that no event gets lost or processed twice. For example, if you read an event from a topic, perform some transformations, and store the result in another topic, the whole operation will be rolled back if one of the three steps fails.

A detailed introduction to Kafka can be found on the official Apache Kafka website: (Apache Software Foundation 2020).

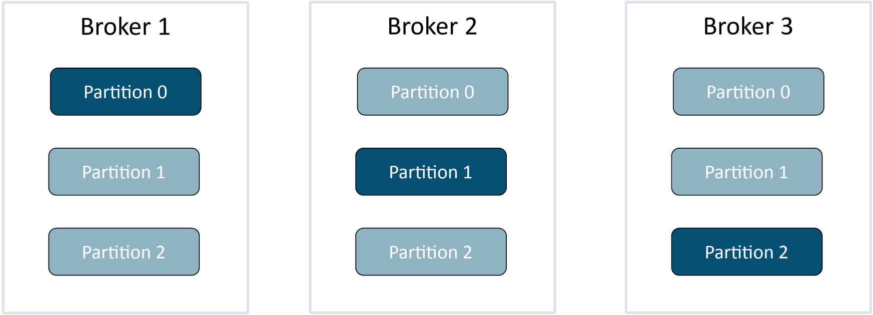


Fig. 2 Topic partitions and replicas (© ifb SE)

4 Kafka Use Cases

Kafka is an ideal solution for solving the challenges described in Sect. 2. It also offers a very powerful platform to implement use cases for which standard software might not be the right fit. These use cases tend to have the following requirements:

- Millions or billions of records need to be processed in real time,
- Calculations need to be performed for each event individually,
- Decoupling of data producers from consumers,
- Reproduction of past results,
- Event processing needs to include real-time checks (e.g., fraud detection),
- Source data has to be collected from several different applications and databases,
- A high degree of custom business logic.

Because Kafka offers storage, processing, and messaging in one platform, it is a great solution to handle all these common technical challenges in one place so that use cases can focus on their business logic. Kafka also helps to reduce costs for the following reasons:

- Kafka's standard version requires no software licenses,
- Kafka requires less expensive hardware,
- There are various cloud offerings for Kafka that reduce maintenance and administration effort,
- The number of different technologies can be reduced,
- Kafka and especially Java are well-known technologies in the tech world. This makes finding skilled resources much easier.

As mentioned before, business logic is implemented in lightweight microservices that run outside of Kafka. This enables the development of basically any type of use case because one is not limited to the feature set of a particular development workbench or software vendor. Many best-of-breed solutions from the open-source world can also easily be leveraged. The following table provides a selection of use cases from the finance industry and explains why they would benefit from using Kafka.

Use case/requirements	Why Kafka?
Cash flow simulation	
• Generate cash flows for 1M + contract conditions, using various input parameters such as patterns, reference rates, etc • Ad hoc simulation of different market data scenarios • Ability to apply custom business logic • Ability to process and store large data volumes	• Kafka can store large data volumes in a cost-efficient way as it mainly stores it on disk • Kafka was designed from the core to process billions of events in parallel. Contract conditions are events just like clickstreams from a web page • Kafka Connect provides standard connectors for many applications and databases • Business logic for cash flow simulation can be implemented in Streams apps and unit tested (e.g., with Junit)
Risk	
• Calculation of various risk measures, stress testing, etc., based on multiple sources • Real-time risk calculation, such as intraday liquidity forecasting[a] • Implementation of custom risk calculations • Integration of machine learning frameworks and platforms	• Kafka offers various possibilities to interface with machine learning frameworks such as R-Server (REST APIs, Connectors, etc.). It can centrally store and distribute data for model training • Risk calculations are often very customer-specific. Kafka, together with other open-source solutions, can be used to build a bespoke risk calculation platform • Kafka's real-time streaming capabilities allow instant risk results
Accounting pre-processing	

(continued)

(continued)

Use case/requirements	Why Kafka?
• Custom calculation of input factors for accounting • Enrichment of transactions with additional attributes (e.g., derivation of profit centers, debit/credit accounts, etc.) • Integration with accounting solutions, such as SAP S/4 HANA • Integration with various source systems • Able to process millions of transactions in near real time • Aggregation of detailed transactions into general ledger documents	• Calculations and transformations can be implemented in Kafka Streams apps and fully unit tested (e.g., with Junit) • Kafka Connect offers standard connectors for many applications and databases (including SAP HANA) • Using Kafka consumer applications, results can be sent to a variety of different interfaces • Source data can be stored for a defined period of time in Kafka topics and reprocessed at any time • Kafka can easily be scaled to process billions of transaction events
Application and system integration	
• Connection of many heterogenous source systems with business applications • Central storage of source data and single source of truth • Offer bulk load as well as event streaming • Ability to detect and deal with incorrect records • Perform data transformations, data quality checks, and other operations • Decoupling of data consumers from producers	• Kafka Connect offers standard connectors for many applications and databases • Source data can be stored for a defined period of time in Kafka. This also decouples data consumers from producers • New consumer applications can subscribe to existing Kafka topics as needed without reloading data from source systems • Consumer applications can read data from topics based on timestamps (allowing time travel) • Transformations, data quality checks, and other typical data integration tasks can be implemented in Kafka Streams apps (microservices) and fully unit tested
Transaction processing	

(continued)

(continued)

Use case/requirements	Why Kafka?
• Transactions, such as payments, need to be checked in real time, including fraud detection • Multiple business applications require transaction information for accounting and analytics • Users need to be notified in case a fraud incident occurs (or other incidents)	• Kafka's topics act as message queues. Transactions can be processed through several topics and checks can be implemented in between them. This also allows the easy integration of third-party applications • Incorrect transactions can be "parked" in dedicated topics while correct ones continue to get processed • Through Kafka's scalability features, fraud detection and other checks can be performed in real time to mitigate risk • Using Kafka Streams apps and state stores, account balances can be checked in real time. Kafka Streams apps also allow implementing notification services • Business applications can subscribe to the relevant topics to instantly get their required transaction information
Profit	
• Calculation of profit key figures based on various source data and parameters • Profit calculations often occur on a very detailed level (product or transaction), which requires a solution to handle very high data volumes • Custom calculations need to be applied • Results need to be produced in near real time	• Using Kafka Connect and Streams apps, source data from many different applications and databases can be loaded into Kafka topics and joined together • Profit calculations can be implemented in Java microservices, for example with full unit test coverage. In addition, Kafka Streams API provides many data transformation options (aggregation, map, etc.) • As it is easily scalable, Kafka can handle billions of events in real time. It can store data in a cost-efficient way and also integrates with cloud storage solutions • Results can be sent to downstream reporting applications or directly provided to UIs via Kafka state stores and REST APIs

For details see (Liermann et al. 2021)

The next sections take a closer look at some of these use cases.

4.1 Accounting Pre-processing

In our demo application, we implemented simplified pre-processing for insurance accounting. Incoming payment events are enriched with accounting information and aggregated into general ledger documents. Our pre-processing also includes the calculation of profit and loss release factors based on actuarial patterns. All calculations and transformations are event-driven and require no process scheduling. The dashboard (developed in ReactJS) is updated in real time as soon as new results are produced. It uses a REST API to read results from Kafka state stores (Fig. 3).

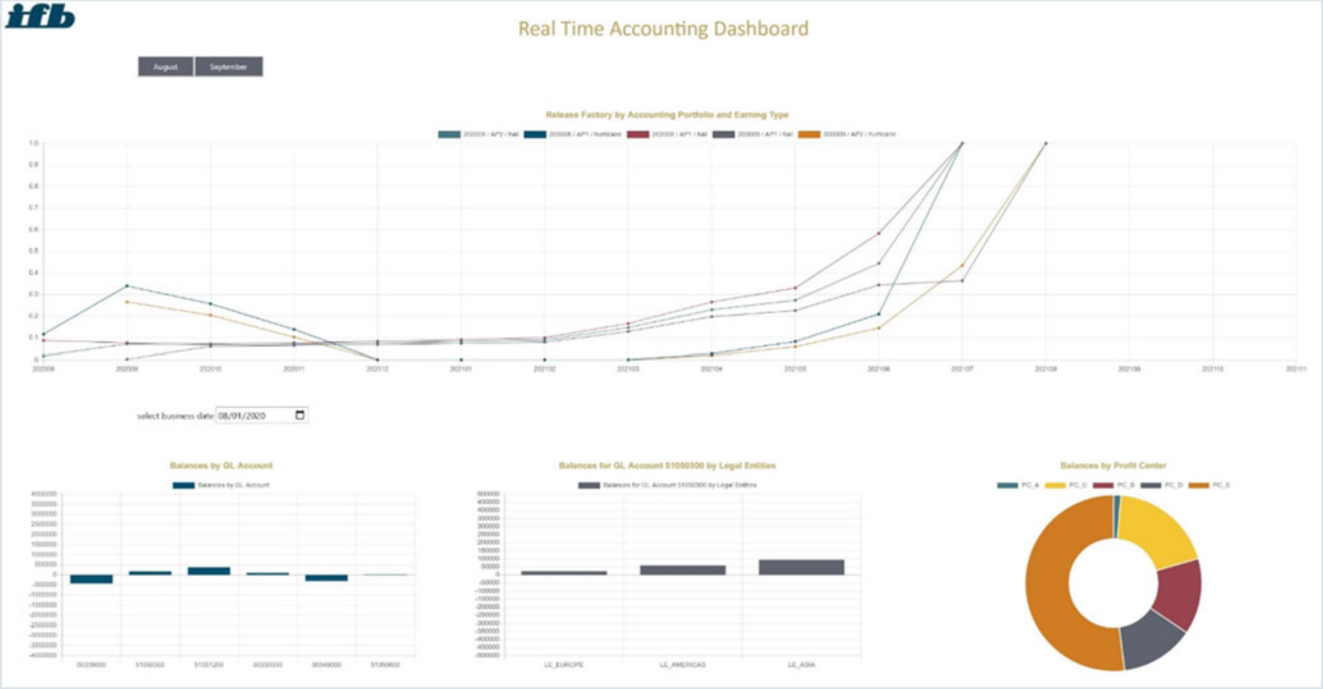
Accounting pre-processing sometimes requires complex calculations and not just simple transformations and derivations. As these calculations are often very custom in nature, standard software cannot always provide the required functionality. The release factors in our demo application are one example of such calculations. They are based on actuarial patterns that aim to predict the future risk of different natural events, such as hail or hurricane, in order to defer the realization of profit from insurance premiums accordingly (i.e., when the corresponding risk does not exist anymore). The following list provides an overview of the required calculation steps:

1. Actuarial earning patterns are matched with individual coverages based on their characteristics.
2. Future cash flows are calculated based on coverages' premium, start date, duration, configured day count method, and matched earning pattern.
3. Detailed cash flows are aggregated based on the configured closing period (monthly, quarterly, etc.) and rolled up to accounting portfolio level.
4. Results must be kept available over several past periods in order to facilitate comparisons.

The diagram below depicts our data flow for the pre-processing results shown in the lower part of the dashboard. This data flow is responsible for enriching incoming payment events with accounting information (e.g., debit/credit accounts) and aggregate them into posting documents (Fig. 4).

The following briefly explains the data flow depicted in the diagram above:

1. Payment events and contract information are streamed into Kafka topics via Kafka Connect from source systems.
2. A Kafka Streams application (microservice) called "Source Joiner" combines the two sources into one so that payment events contain all accounting relevant product attributes.



select business date 08/01/2020

Fig. 3 Example dashboard (developed with ReactJS) (© ifb SE)

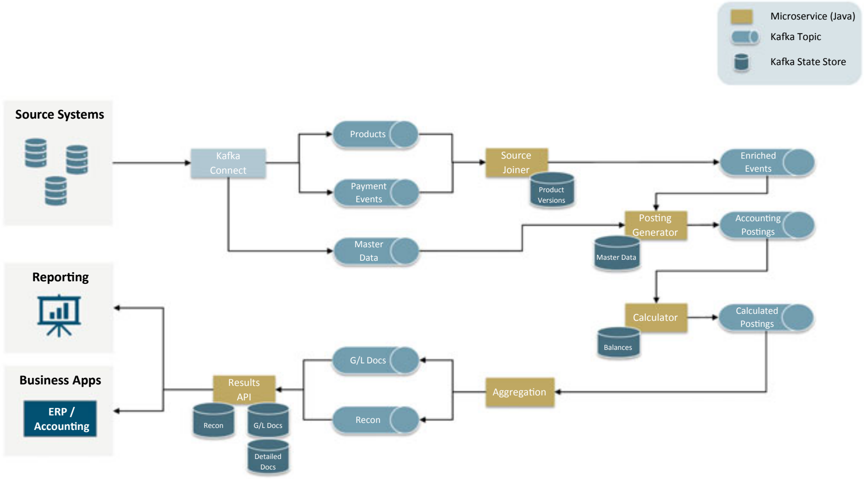


Fig. 4 Data flow pre-processing for accounting (© ifb SE)

3. Master data, such as general ledger accounts, is also loaded via Kafka Connect.
4. Immediately after the join, another microservice uses the master data from its state store to create debit and credit postings for each incoming event.
5. Some events might require additional calculations based on balances, which are performed by the calculator microservice.
6. Lastly, an aggregation service groups the detailed postings into general ledger (G/L) documents as required by the downstream accounting systems. It also generates reconciliation keys to link G/L docs to their underlying detailed postings. In order to suppress intermediate results, the aggregation service uses a Kafka state store.
7. All accounting results are cached in Kafka state stores and exposed via REST API in the results service.

With Kafka, events do not need to be collected into a database and then processed in batches using a scheduling tool. Each event can be processed immediately when it occurs. However, this is not always possible as certain calculations require data from multiple events (e.g., step 6 above). In order to solve this problem, Kafka state stores and the Stream Processor API can be leveraged to cache events. This also allows the suppression of aggregation results for a defined period of time. In our test, a time window of a few seconds was sufficient to produce complete aggregation results. In case some events arrive late, delta postings are generated based on the cached previous results.

In this example, Kafka is the only data persistence. However, for more complex or ad hoc analysis, results can also be streamed to a dedicated reporting database. For long-term storage, data can also be streamed to Hadoop or to cloud storage solutions (e.g., AWS S3).

The advantage of such an architecture is that standard accounting applications can store less data, which reduces costs for licenses and hardware. It also shows that you can easily outsource high-volume transaction processing to Kafka while only sending aggregated results to business applications.

4.2 Application and System Integration

Two common types of integration middleware are ETL (Extract, Transform, Load) and ESB (Enterprise Service Bus). ETL tools are often used to transfer large amounts of data from source systems into target applications via batch jobs. They can contain complex transformation rules, often configurable via GUI. ESB tools, on the other hand, are used to transfer individual events between application processes in real time. For example, an invoice creation in one system can trigger an action in another system.

Apache Kafka can combine both types of middleware into one platform. Using Kafka Connect and CDC (Change Data Capture), it can extract large chunks of data from databases and ingest them into topics. For streaming single events, Kafka producers can, for example, provide REST APIs. As events in Kafka are persisted (retention period can be configured), the same event can be consumed multiple times. If the target application would not be available when the event is sent, it can simply consume it once it is back online. Being able to offer both types of integration middleware in one platform reduces complexity as it requires fewer tools with possibly expensive licenses.

There is one disadvantage with Kafka compared to traditional integration middleware: Kafka does not offer a GUI for configuring transformations or designing data flows. If you already have, for example, an ETL tool that performs complex transformations, it is best to load the result into Kafka instead of recreating the same logic in Kafka. This disadvantage is somewhat compensated for by Kafka's unit test capabilities. The TestTopologyDriver simulates an actual Kafka cluster so that source code can be tested directly on a developer's computer. This reduces testing effort and leads to more robust software. Kafka also integrates well with common monitoring and logging tools.

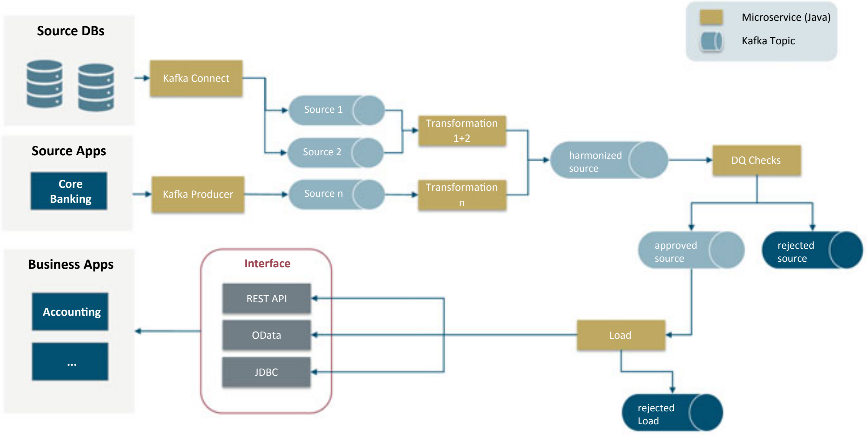


Fig. 5 ETL example based on Kafka (© ifb SE)

The following diagram depicts an example of ETL data flow in Kafka. We built a similar one to load data from an SQL DB into an SAP accounting application via OData service (Fig. 5).

In this example, source data is streamed into Kafka using Kafka Connect as well as via the Kafka producer application. There are many ways such a producer can be implemented. For example, it can directly query the source application's database or offer a REST API to receive data.

From our experience, it is usually very helpful to first load source data into Kafka before performing any transformations. Depending on the source data, one or several transformation microservices process the raw events and generate harmonized source data records. In the next step, data quality checks are applied, and incorrect records are sent to a dedicated error handling topic "rejected source" (also called "dead letter queue"). Finally, the successfully validated records are loaded into the target applications. In this example, business applications do not need to implement special logic to poll Kafka topics as data gets pushed to their standard interfaces from the Kafka consumer application.

Data flows consisting of Kafka topics and well-defined microservices simplify application integration and make it more transparent. Separate topics for erroneous records serve as the basis for error handling and subsequent reprocessing. All business applications can get their data from one central platform and are decoupled from source systems.

5 Deployment

Our deployment infrastructure for the example applications consists of:

- Serverless Kafka on Confluent Cloud
- Elastic Kubernetes Services (EKS) on AWS for Java microservices.[1]

Confluent Cloud abstracts away the complexity of a Kafka cluster. It scales automatically without having to install and administer zookeepers or brokers manually. This greatly simplifies a Kafka deployment and allows companies to quickly adopt Kafka with only a small investment.

Kafka can also be run on other clouds (e.g., AWS, Google Cloud, or Microsoft Azure) as well as on-premise. Confluent Platform offers a very easy installation and comes with additional community components (e.g., KSQL and Schema Registry).

6 Summary

Kafka and many other open-source technologies offer a cost-efficient way to modernize IT landscapes and enable innovation. They often come with strong community support and a vast repository of tutorials. It is also much easier to find skilled developers for open source than for proprietary technology. However, there is a certain learning curve with Kafka that one should not underestimate. It requires in-depth technical knowledge in order to be successful. But this is a long-term investment that will certainly pay off in the future.

We have often encountered software implementation projects that took years to finish because the software was not the right fit for the given requirements. The percentage of out-of-the-box functionality that could actually be used was sometimes quite low. As a result, software implementation projects became more custom development projects.

Due to its capabilities, Kafka is a great fit for many use cases and can also be used in combination with standard software. Increasing data volumes and the ability to process events in real time require a platform such as Kafka. It is not necessary to completely replace current architecture with Kafka all at once. One can start with just a few use cases in order to gain experience and

[1] See Amazon Web Services (2020).

add more over time. Cloud infrastructure also makes it easier to start with Kafka as only a limited investment is required.

Literature

Amazon Web Services. 2020. *Amazon Elastic Kubernetes Service.* Accessed December 16, 2020. https://aws.amazon.com/eks/.

Apache Software Foundation. 2020. *Apache Kafka.* Accessed December 16, 2020. https://kafka.apache.org/.

Confluent. 2020. *Confluent.* Accessed December 16, 2020. https://www.confluent.io/.

Liermann, Volker, Sangmeng Li, and Ralph Steurer. 2021. "Kafka—A Practical Implementation of Intraday Liquidity Risk Management." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Architecture Patterns—Batch and Real-Time Capabilities

Dennis Kraetz and Michael Morawski

1 Introduction—From Greek Letters to Architecture Patterns

When talking about modern architecture patterns in the context of Big Data, you will inevitably come across the Greek letters Lambda, Kappa, and Delta. But what do they mean and what ideas stand behind these different architecture patterns? What has changed in recent years or, in other words, what were the needs and triggers for the evolution and rise of these architecture patterns?

In the last couple of years, we have seen a change not only in technology and the technical abilities to process and store huge amounts of data, but also a shift in the mindset of financial service institutions. Decision-makers have understood that IT applications and the corresponding architecture need to move away from mere number crunching data providers toward creating business values and supporting the decision-making process by providing advanced analytical capabilities.

D. Kraetz (✉) · M. Morawski
ifb SE, Grünwald, Germany
e-mail: Dennis.Kraetz@ifb-group.com

M. Morawski
e-mail: Michael.Morawski@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_6

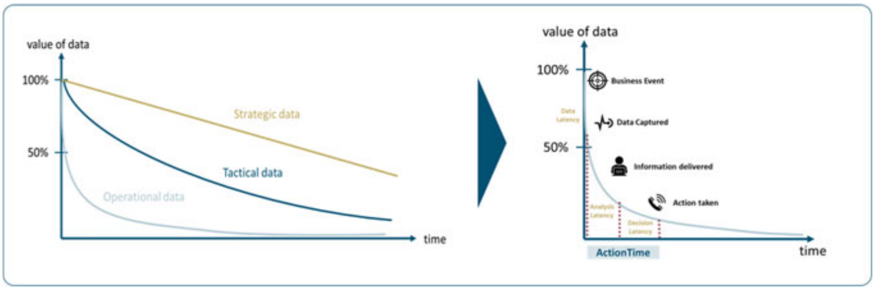


Fig. 1 The value of data over time (© ifb SE)

In addition, new, disruptive business models are emerging that put their focus on DATA, SPEED, and SCALING. Operational data, its processing and its analysis play a major role in these scenarios and a lot of companies see great potential and a competitive edge in the use of these datasets. These datasets, however, all have one thing in common: their business value decreases rapidly from the moment they were captured. Therefore, companies need to react very quickly or otherwise the data is of little use or value (e.g., sensor data, clicks on a website, etc.) (Fig. 1).

To support these business scenarios, also from a technology perspective, IT leaders need to design and provide powerful and future-proof application architectures. They need to find solutions for new emerging challenges like extremely high reliability, quasi unlimited scalability, and processing guarantees, just to name the prevalent ones.

The Big Data ecosystemsprovide a variety of solutions and applications to tackle these challenges. However, getting started in this mostly open-source-dominated world is not easy. There are countless tools and frameworks around, e.g., Hadoop, that cover applications such as batch and stream processing, data ingestion and transformation, security, orchestration, and much more. The amount of time and effort to bring these different tools together into a modular, but cohesive architecture that performs the required functions efficiently and effectively should not be underestimated. In addition, technology is evolving at a fast pace and the required talent is scarce and not easily available in the market (Fig. 2).

Before taking a closer look into the different architecture patterns, there is one last thing that needs to be evaluated. With all the newest technologies, processing power, distributed storage options, and latest design patterns… can you beat the CAP theorem?

But first of all, what is the CAP theorem? The CAP theorem states that it is impossible for a distributed system to provide all three of the following guarantees at the same time:

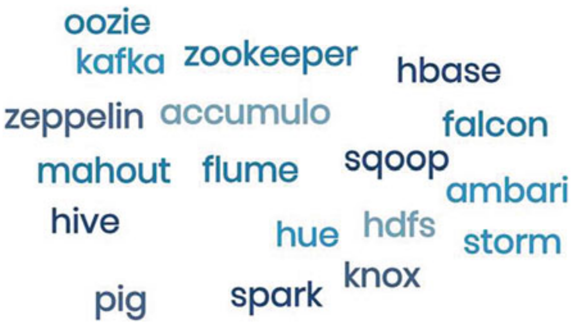


Fig. 2 Big Data ecosystem (example) (© ifb SE)

- Consistency: All nodes see the same data at the same time.
- Availability: Each client can always read or write.
- Partition tolerance: The system can operate despite arbitrary message loss or partial system failure (Fig. 3).

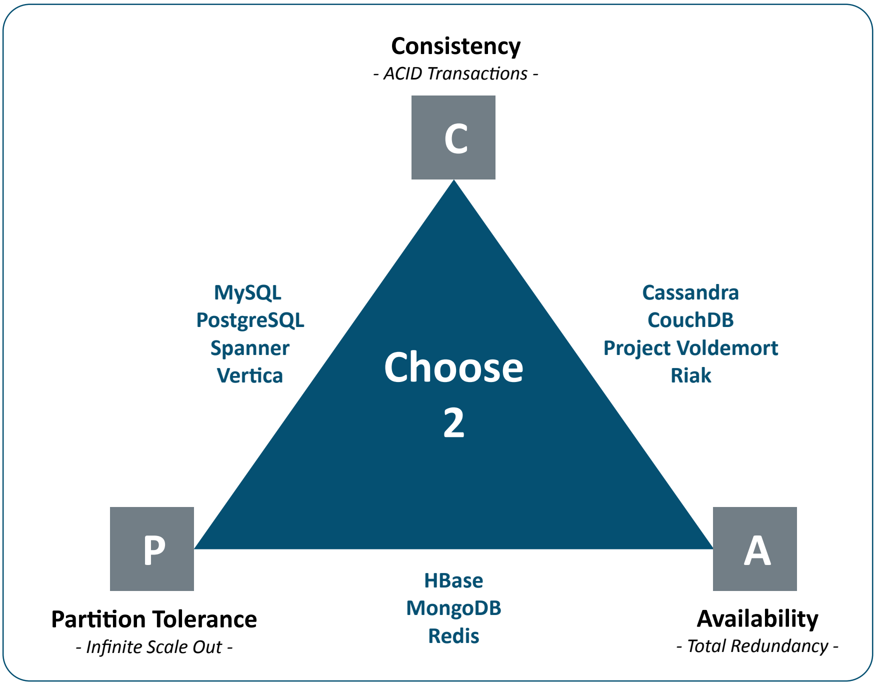


Fig. 3 The CAP theorem © ifb SE

Partition tolerance is mandatory and non-negotiable (requirement to store large amounts of data is obvious) for most applications or systems. This leaves only the remaining two to work with: availability or consistency.

The trade-off between the two is difficult and can lead to critical challenges during the setup and operation of the respective systems. When consistency is the focus, favored by relational databases, a negative impact on performance can be expected. If availability is preferred, as with, for example, NoSQL concepts, the risk of inconsistent data increases significantly.

The CAP theorem as postulated by Gilbert and Lynch (Gilbert and Lynch 2002) represents the academic, ideal state of the three dimensions: consistency, availability, and partition tolerance. However, looking at it from a more practical perspective, there is definitely room to work with. In a real-life scenario, you need to find a balance and find compromises that also work from an economic perspective. Atomic consistency in every aspect across a distributed system is desirable in theory, but not really required in reality. The same is true for perfect availability. Eventually you must address and fulfill the defined business requirements, and, in most cases, this can be achieved without perfectly mastering the three dimensions of the theorem.

Considering the above, let's have a closer look at the different architecture patterns and how they are designed and set up to resolve the challenges at hand.

The following paragraphs will give a short introduction to the different concepts and highlight their advantages and disadvantages as well as use cases and potential tooling.

2 Lambda Architecture

As businesses try to become more data-driven and event-driven, they are facing as a consequence massive amounts of data that need to be stored, processed, and analyzed. The Lambda architecture is an architecture pattern tailored toward data processing that combines a traditional batch pipeline with a real-time stream pipeline.

This approach tries to balance throughput, latency, and fault tolerance by using the batch pipeline on the one hand, providing accurate and comprehensive results, and on the other hand a real-time stream pipeline simultaneously to enable data processing and analytics on online data. The results of both pipelines can then be joined prior to presentation.

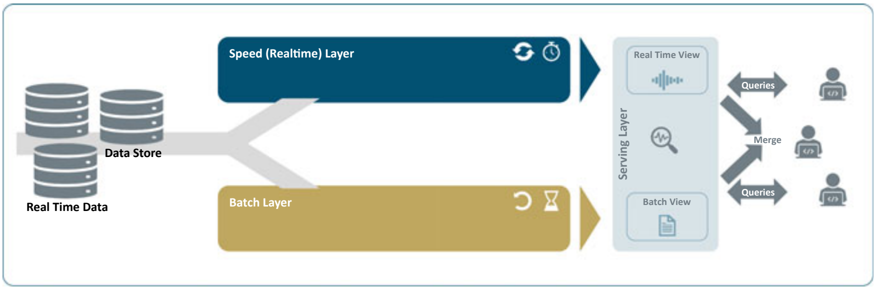


Fig. 4 Overview Lambda architecture (© ifb SE)

The goal of the Lambda architecture is to provide a resilient and fault-tolerant system that can handle Big Data volumes and supports a variety of use cases, all in near real time.

The Lambda architecture consists of three layers as depicted in Fig. 4:

- Speed Layer
- Batch Layer
- Serving Layer.

The following paragraphs give an overview of the three different layers and their function within the Lambda architecture.

2.1 Batch Layer

The Batch Layer is designed to handle and process large volumes of data. It serves as a source of truth for all incoming data since the data is considered immutable and stored as append-only. Results are generated using a distributed processing system and the focus is on accuracy and completeness. It is a scalable data store that swaps in new batch views as they become available. It processes all available data when generating results and is therefore able to fix any inconsistencies or errors before updating already existing views. The Batch Layer acts as a historical archive to hold all data that was loaded into it and is thus very valuable for analytics. The output is usually stored in read-only databases that can be consumed via views.

The most prominent technology used in this area is Apache Hadoop, however, new massively parallel, elastic, relational databases like Snowflake or Big Query are on the rise.

2.2 Speed Layer

In contrast to the Batch Layer, the Speed Layer focuses on real-time processing of the incoming data. The goal is to minimize latency and to provide real-time views of the most recent datasets as fast as possible. It, however, sacrifices accuracy and completeness and does not provide error correction capabilities. The Speed Layer makes up for the disadvantages in latency and processing time of the Batch Layer. The results may not be as accurate or complete as the ones of the Batch Layer, but they are available almost instantly after the data is received. Once processing of the Batch Layer is finished and views of the same data become available, the results of the Speed Layer can be replaced with the more accurate ones from the Batch Layer.

Apache Storm and Spark or SQLStream are technologies that are often used in this area. The output of the stream processing is usually stored in fast NoSQL databases.

2.3 Serving Layer

Results (views) of both Batch and Speed Layer are stored and made available in the Serving Layer. For the views to be useful, the content must be accessible with low latency and the processing must be carried out as parallel as possible to be able to quickly respond to ad hoc queries by the users. This layer can also reindex all data to fix a coding bug or to create different indexes for different use cases.

Technologies like Druid provide a single cluster that can deal with the outputs of both Batch and Speed Layer. Other tools like Apache Casandra or HBase, MongoDB or Elastic Search provide dedicated data stores for the Speed Views, whereas ElephantDB, Apache Hive, or SAP HANA provide dedicated storage for the Batch Layer results.

2.4 Benefits of the Lambda Architecture

There are quite a number of benefits that come along with the Lambda architecture. The append-only data model providing an immutable data source as well as the natural time-based ordering of the data provides a solid architectural basis for further individual extensions.

However, as usual, businesses must make sure that the individual business requirements and use cases are in line with this architecture concept.

Nevertheless, the following features highlight the major benefits of the Lambda architecture and should be carefully considered when thinking about renewing your current landscape (Hazelcast, n.d.)

- **Reduction of latency**: Lambda architecture designs narrow the time window where data cannot be queried or analyzed by taking advantage of the Speed and Batch Layer concept (batch indexing takes a while and stream processing in the meantime provides indexing/access to recent data)
- **Elimination of inconsistencies**: Lambda architecture eliminates the risk of data inconsistency that is often a challenge in distributed systems. Data is processed sequentially (and not in parallel with overlap), and the indexing process ensures that the data reflects the latest state in both the Batch and Speed Layers.
- **Scalability with ease**: Since the Lambda architecture is based on a distributed, scale-out design, extension of the system can be achieved by simply adding more nodes to the respective layers (Batch, Speed, or Serving).
- **High fault tolerance**: This is given by design since Lambda architecture is based on distributed systems. Hardware failure is mitigated as other nodes are available at any time to continue the workload. Failures during indexing of data can be mitigated by re-running the indexing job in the Batch/Serving Layers, while the Speed Layer continues indexing the most recent data. In addition, raw data is saved during the ingestion process and can serve as a system of record. Should any lapses or issues with the indexing code occur, it can be corrected and then rerun to re-index all the data.
- **High agility**: Quick responses to changes in the market/business environment are possible.

2.5 Limitations of the Lambda Architecture

This sounds very promising so far and you might ask yourself why Lambda architectures are not more widely adopted in the industry. As said before, first you need to have appropriate use cases that justify the adoption of the Lambda concept. And of course, there are some challenges as well. A major one is the different code bases that need to be maintained and synchronized in the different layers. Coding in frameworks like Hadoop or Storm is complex anyway, but since the code is usually written toward the framework that it is run in (Batch vs. Run), this is quite complex in the Lambda context and a lot of effort. And not only does the code base need to be aligned, but also the two copies of the data need to be maintained as well.

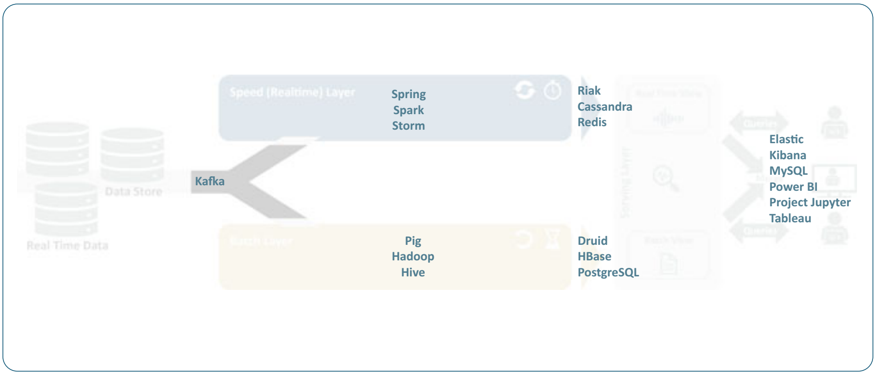


Fig. 5 Tooling options within a Lambda architecture (© ifb SE)

Finally, projects in the Big Data context are often realized with open source tools. As shown in Figs. 2 and 5, there is a whole ecosystem of tools that can be used to implement a data platform. Bringing these tools with their specialized functions together to a cohesive and powerful architecture is a challenge in itself. Finding the talent and skills in the market and/or training existing resources in the IT departments is another one.

And there is still the CAP theorem….

2.6 Summary

As stated at the beginning of this chapter, eventually it is always an individual evaluation and decision whether the Lamda architecture is appropriate and suitable for the use cases at hand. Besides the many benefits stated in Sect. 2.4, Lambda architecture is inherently complex. Keeping two distributed systems in sync is definitely a challenge, not only during the implementation, but also when it comes to maintenance. In addition, various tools (often open source) are used to implement the architecture. As a consequence, a skilled workforce with expert knowledge in niche areas is required, which increases complexity, also from a management perspective.

Thus, the market is looking for simpler alternatives that provide similar benefits without the current challenges of Lambda.

There are three potential solution approaches (Blog T 2015):

- Pure streaming approach:

  Adopt a pure streaming approach and, in addition, use a flexible framework like Apache Samza to provide at least some type of batch processing capability (can be configured to process several messages from the same stream partition in sequence).

- Flexible batch framework:

  Take advantage of a flexible batch framework that is able to process incoming data in micro-batches that are close to real time (e.g., Apache Spark/Spark Streaming or Storm's Trident).

- Unified technology stack:

  Choose a technology stack that combines batch and real time by design. Spring "XD," Summingbird or Lambdoop provide a hybrid computation model for both batch/real-time workflows that can be transparently run at the same time and the results merged automatically.

3 Kappa Architecture

The Kappa architecture was introduced by Jay Kreps in 2014 (Kreps 2014) as an alternative to Lambda architecture. In general, both architectures share the same goals, but the Kappa architecture aims for only one stream of data in contrast to Lambda architecture, and is used in settings where performance of the Batch Layer is not needed to meet the quality standards of service for the user (Fig. 6).

3.1 Layers of the Kappa Architecture

In Lambda architecture, the **Speed Layer** is needed to keep the **Serving Layer** up to date so that users are, for example, able to perform real-time (or at least near-time) analytics. Therefore, the Speed Layer needs to be there to

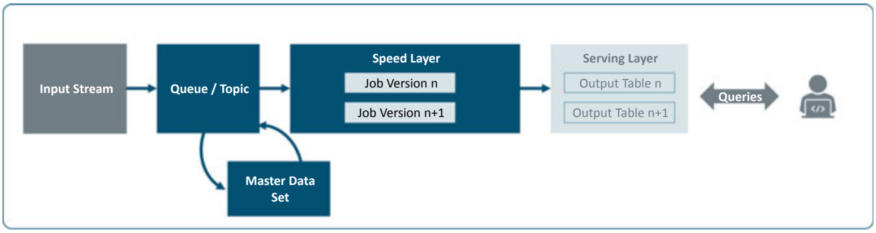


Fig. 6 Overview Kappa architecture (© ifb SE)

cover this requirement. But what about the Batch Layer? Is it necessary? The answer in Kappa architecture is: “No.” Getting rid of the Batch Layer brings the huge benefit of only maintaining the code base of the Speed Layer and eliminating a potential source of errors. But this also means losing a chance to do cross-checks between both pipelines, which can be compensated for by ICS (internal control system) to secure data quality and plausibility.

In Kappa architecture, data is processed in a streaming manner in order of the occurrence of the events. This means that the events are ordered, and the actual state of data is only changed by appending a new event. This event log needs long-term storage in which **all** the events are stored, and these must be **immutable** (“**Master Data Set**” in Link to Fig. 6 is broken **Fehler! Verweisquelle konnte nicht gefunden werden.**). The reason for this is to be able to reprocess the data needed from the event log in case of an error or a coding update in the Speed Layer which impacts the result data in the Serving Layer. As the Master Data Set may become huge, especially in a Big Data scenario, the Kappa architecture is often used in distributed systems which scale horizontally like Hadoop tools, which means that the system is easy to extend for future requirements or data sources.

For reconstruction of historic query results for audit reasons, it can be necessary to store the output of each job version in a separate table. This ensures that this information is not lost if a new job version is deployed in the Speed Layer with a different logic and historic data is reprocessed.

3.2 Benefits and Limitations of the Kappa Architecture

Lambda architecture has the need for two technology stacks to cover the batch and the stream processing, whereas the Kappa architecture has the benefit that only one technology stack for streaming is needed, which lowers the TCO of the software significantly. The tools used in Kappa architecture can be the same as in Lambda architecture (see Fig. 5). A tool often used in this context is Apache Kafka as it scales horizontally, is fault tolerant because of data replication and can easily reprocess data because of the offset of each topic.

In contrast to batch-driven data pipelines, the streaming jobs in the cluster can be deployed with a fixed memory size as there is a constant flow of data over time. This means on the flip side that Kappa’s advantage of providing data almost in real time swiftly shrinks if source data is provided in huge dumps at a single point in time, e.g., after an overnight run of the source system. Therefore, it is an important prerequisite to have this constant input stream of data. If the source of data is a database, this can be achieved via CDC (change data captures) to broadcast the changes on the single database rows to the consuming system with tools like Oracle Goldengate instead of creating a classic database dump once per day. If machine learning methods are applied to the Kappa architecture, fewer resources are needed as the data processing is stretched throughout the day. The absence of a Batch Layer in Kappa architecture not only has benefits regarding lower complexity and maintenance effort, but also has disadvantages regarding error handling. An exception manager is needed to reprocess erroneous data and carry out a reconciliation (Samizadeh 2018).

4 Lambda vs. Kappa Architecture

Like the Lambda architecture, the Kappa architecture serves the same requirements: to have a full up-to-date picture for short-time decisions/reporting and more key-date-driven reporting with enriched and calculated figures for deeper analysis regarding long-term decisions/reporting. Kappa architecture has its strength where the second part of the previous sentence (the calculation of the enriched data) is not too complex and the data does not have too many time dependencies. If this is not the case, it is better to separate the datasets and you automatically arrive at a Lambda architecture to satisfy the needs of real-time analysis (Speed Layer) and reporting on enriched data (Batch Layer). Therefore, both architectures depend mainly on the requirements the individual company has or what it wants to achieve. Depending on the question of whether the reporting on enriched data can be achieved with a streaming approach, Kappa architecture is the better choice to avoid double efforts for maintaining two code blocks (stream/batch) like in Lambda architecture. To complete the picture, there is not only Lambda (stream and batch) and Kappa (stream), but also the classic batch-driven architecture used in the data warehouses where there is no need for a real-time component at all. These three architectures form the range of the spectrum between streaming and batch data processing with the pros and cons already discussed in Sects. 2.4 and 3.2.

5 Other Upcoming Architecture Patterns

In a Big Data scenario, data is often stored in a data lake and not in a data warehouse. The reason for this is that, for example, Hadoop has the “schema on read” to deal with huge amounts of data to be stored, which is much quicker. This is in contrast to the "schema on write" approach of a data warehouse, which has benefits like schema consistency. Despite the fast writing of files in different formats, the data lake has some limitations:

1. Reading and writing are not reliable: As the ACID principle (Wikipedia 2020) is not capable in Hadoop, the reader may see faulty data during write processes into the data lake, making workarounds necessary for these situations.
2. Low data quality: Dumping different sources of data into the data lake often causes quality issues as validation & data governance are missing. Therefore, the data consumer is mostly responsible for these topics.
3. Updating records is hard as entire tables (batch) or partitions (streaming) must be queried, relevant data must be modified, and stored again.

The data lake approach can be combined with structured streaming to achieve the **Delta architecture**, which is an evolution of Kappa architecture and addresses the issues of Lambda architecture. These problems include the **duplication of data** and **inconsistent data** due to the fact that two different teams work on the data pipeline (batch & stream), which means that a validation process is needed. If the validation process brings up an **error or failure** and half of the data has already been processed in the data lake, a partition characteristic, preferably a time dimension (data, timestamp), is needed to go back and reprocess the data again, resulting in huge efforts to achieve a consistent state (Fig. 7).

The Delta architecture in a data lake scenario is referred to by the term **Delta Lake** (Armbrust 2020), which has already become an open software project (delta.io 2020) to tackle the data lake issues mentioned. It enhances the widely used Big Data tool for data processing Apache Spark by the feature of using ACID transactions in a data lake (ACID = Atomicity/Consistency/Isolation/Durability) (Fig. 8).

By adding serializability to the data lake, concurrent reading and writing operations are isolated (the "I" of the **ACID** principle) which addresses weakness 1 of data lakes. This is done by introducing a transaction log to the data

Delta
Lake
Delta
Architecture

Fig. 7 Continuous streaming and the Delta Lake are the core components to achieve a Delta architecture in a Big Data scenario (© ifb SE)

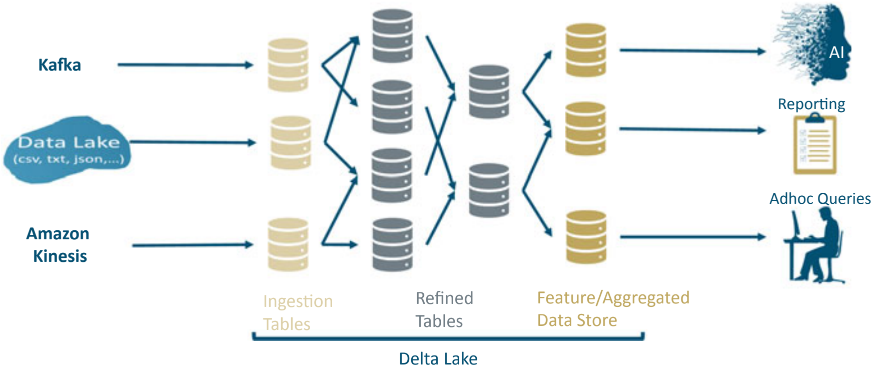


Fig. 8 Delta architecture unifies batch & streaming to achieve a continuous data flow model (© ifb SE)

lake, also known as Delta Log, which stores every transaction (insert, update, delete) of the Delta Lake table. This log serves as a single source of truth to get the latest state of a table. The isolation is achieved by a rule of mutual exclusion if two or more commits are made to the storage at the same time. The conflicts are solved optimistically, and an exception is only thrown in case of an unresolvable error. The implementation looks like this (Yuvaz et al. 2019):

1. Record the starting table version
2. Record reads/writes
3. Attempt a commit
4. If someone else wins, check whether anything you read has changed
5. Repeat.

This results in the concept known as serializability in databases, as a transaction log with a distinct order is created.

The transaction log can be used to provide a data snapshot of the Delta Lake at every point in time to achieve **Time Travel** (= data versioning), which makes it much easier for developers and data scientists to carry out error corrections (weakness 32 of the data lake) or reporting on historic data.

The data in the Delta Lake is stored in Apache Parquet format to make use of compression and the native schema encoding. As the schema naturally changes over time because of new requirements from the business side, it is important to pay attention to this. Schema in this context means the data types, columns, and metadata of a table which are stored in the transaction log as a JSON file, which can also serve as an **audit log** to document all historic table changes in an audit trail. The Delta Lake supports **schema**

evolution (= schema validation) to prevent messy data from being stored to a table addressing weakness 2 of the Data Lake. This means that data can only be written to a table if the schema of the new data matches the table schema. If the new data contains an additional column, a different data type or a column name differing from the target table, the data is rejected, which enhances data quality.

Delta Lake unifies batch and streaming as a table can be both: a batch table as well as a streaming source and sink. This means that, like in the Speed Layer, in Lambda architecture it is possible to provide operational data of a table for analytics in a swift way via streaming. Because of the transaction log in a Delta Lake it is possible to populate the same table with additional information calculated in a batch process at a later point in time. This is simply done by updating the missing columns of the affected rows with the new calculated key figures of the batch process, which corresponds to the Batch Layer of the Lambda architecture. This enables the Delta architecture to serve different users with their specific needs: e.g., the need to access operational data from streaming analytics for insights, for data scientists to evaluate historical time series to develop new models, or for BI users to get their standardized reports (Fig. 9).

This architectural approach of course depends on the use case, the existing prerequisites, and the requirements, but makes use of Big Data technologies and cheap scalable storage, which can be hosted on-premise or in a public/private/hybrid cloud. The Delta Lake library is currently under development and new features will be added in the future. It can be used with the common programming languages used in Big Data settings: Java, Scala, and Python.

6 Conclusion and Outlook

The above described architecture patterns are already widely adopted and proven across several industries, also in the financial services sector. A lot of banks and insurance companies are looking into these architecture patterns to optimize their landscapes in terms of agility, scalability, velocity, and of course cost consideration (TCO).

However, a closer look into the respective individual context is required. A differentiation between the respective functional and technical areas of the application of these architecture patterns is advisable.

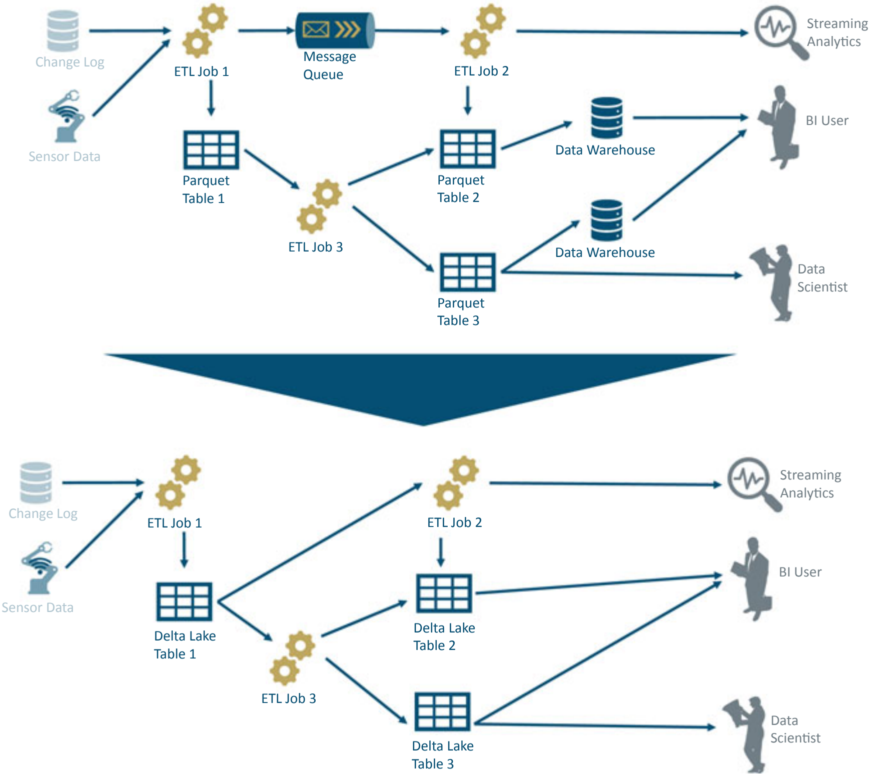


Fig. 9 Combining streaming, data lake, and data warehouse features into a "lakehouse" (© ifb SE)

Without any doubt, there can be major benefits of near-real-time or real-time data processing and analytics capabilities in the context of front-office processes, operational reporting, or internal performance management. However, if we look at the areas where regulatory considerations play a more important role, the picture might change. Requirements concerning governance, audit trail, security, reproduction/retention of data, etc., to name a few, come into play, and the advantages of the data warehouse concept and rigid ETL pipelines including MDM and DQM capabilities might supersede the benefits of the more agile and modern architecture patterns.

Looking at the complexity of today's world and the massive amounts of data being transferred, processed, and stored, enterprises are facing enormous challenges to keep up or even master digitalization. So it is not a question of whether a company should introduce Lambda, Kappa, Delta, or any other architecture pattern. It is about specific business challenges and goals and about defining an efficient and appropriate solution framework. Architecture should always be capability and use-case-driven, all within a scalable, flexible, and easy-to-integrate framework that supports operational and strategic targets and eventually creates business value.

Data fabric and logical warehouse concepts are interesting approaches that focus on connecting rather than collecting data. Here, it is not important whether data is provided via streaming or batch, if it is in a data lake or DWH. The goal is rather to ensure that, from an end-user perspective, the data is available when needed, that it is understandable and that it can be trusted. Governance and quality are key ingredients and the foundation of a successful architectural design and implementation.

If these factors are not considered core components within the architecture right from the beginning, enterprises will end up in a garbage in, garbage out situation, and then it really doesn't matter whether the architecture relied on streaming, noSQL, DWH, or graph DB technology.

Literature

Armbrust, Michael. 2020. "Databricks," August. Accessed October 13, 2020. https://databricks.com/wp-content/uploads/2020/08/p975-armbrust.pdf.

Blog T. 2015. "Lambda Architecture for Big Data," January 5. Accessed October 10, 2020. https://tsicilian.wordpress.com/2015/01/05/lambda-architecture-for-big-data/.

delta.io. 2020. *delta.io*, October 13. Accessed October 13, 2020. https://delta.io/.

Gilbert, Seth, and Nancy Lynch. 2002. "Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services." *ACM SIGACT News*, June: Vol. 33, No. 2. https://doi.org/10.1145/564585.564601.

Hazelcast. n.d. "Hazelcast." Accessed October 20, 2020. https://hazelcast.com/glossary/lambda-architecture/.

Kreps, Jay. 2014. "O´REILLY," July 2. Accessed October 20, 2020. https://www.oreilly.com/radar/questioning-the-lambda-architecture/.

Samizadeh, Iman. 2018. "towardsdatascience.com," March 15. Accessed October 20, 2020. https://towardsdatascience.com/a-brief-introduction-to-two-data-processing-architectures-lambda-and-kappa-for-big-data-4f35c28005bb.

Wikipedia. 2020. "ACID." *Wikipedia.org*, October 13. Accessed October 13, 2020. https://en.wikipedia.org/wiki/ACID.

Yuvaz, Burak, Michael Armbrust, and Brenner Heintz. 2019. "Databricks," August 21. Accessed October 13, 2020. https://databricks.com/blog/2019/08/21/diving-into-delta-lake-unpacking-the-transaction-log.html.

Kafka—A Practical Implementation of Intraday Liquidity Risk Management

Volker Liermann, Sangmeng Li, and Ralph Steurer

1 Introduction

1.1 Initial Situation

The process of intraday managementis highly dynamic and based on the extensive experience of the people involved. The risk management process for intraday liquidity follows the dynamics of the underlying process, but put an independent view in place, based on historical data. Due to the dynamic nature and the criticality of the process, traditional data processes, such as batch, are no longer an up-to-date technology. Streaming technology or a combination of streaming and batch (see Krätz and Morawski 2021) is a contemporary technology choice.[1]

[1] The chapter focuses on streaming technology, but the use case can be easily applied to other design patterns (see Krätz and Morawski 2021).

V. Liermann (✉) · S. Li
ifb SE, Grünwald, Germany
e-mail: volker.liermann@ifb-group.com

R. Steurer
ifb International AG, Zürich, Switzerland

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_7

1.2 General Idea

This chapter combines the time series cluster approach from the chapter (Liermann et al. 2019) with state-of-the-art streaming technology. This covers the dynamic nature of the underlying process as well as the high-frequency data that allows the application of advanced machine learning algorithms to project reasonable future cash flows.

In order to explain the benefits of using Kafka[2] for machine learning, a small liquidity forecasting application was developed. This practical example showcases how a modern and cost-efficient architecture for machine learning applications could look. Instead of describing Kafka in an abstract and general way, we wanted to explain it using a real-world application.

1.3 Structure of the Article

The article first provides a general overview of the architecture for machine learning applications using Kafka. Subsequently, the architecture and data flow for our liquidity forecasting application are presented. Next, we review the mathematical background for liquidity risk and show how R-Server is able to connect to Kafka. The last part contains screenshots from the demo of our liquidity risk application.

2 Practical Implementation

2.1 General Architecture

Machine learning applications, such as intraday liquidity forecasting, require a lot of data for model training. Besides historical data, they also need current datasets in real time in order to make instant predictions about the future. Handling such data volumes in an efficient way is not an easy task, especially since input data often comes from various sources. Data also needs to be harmonized and stored for a certain period of time. Setting up a system architecture for machine learning can thus become a costly and complex endeavor.

[2] An introduction to Apache Kafka can be found in Steurer (2021) and detailed information is provided in Apache Software Foundation (2020).

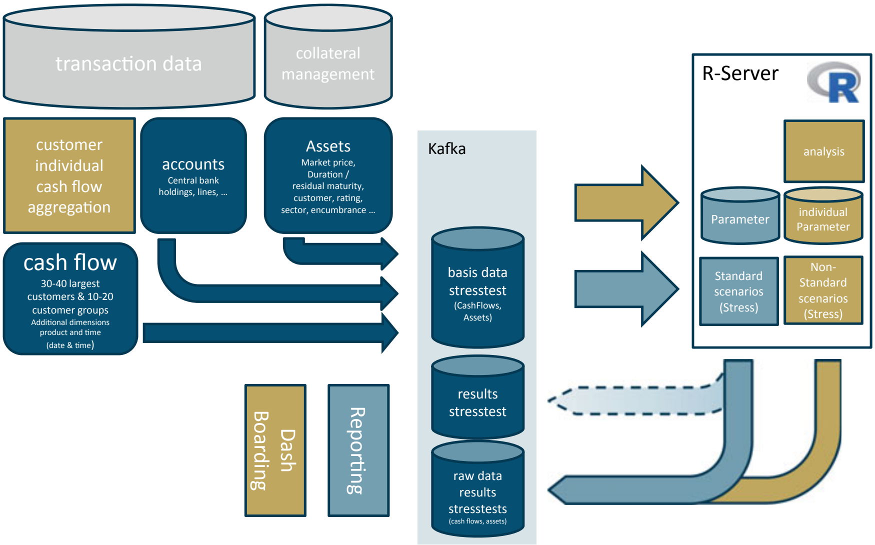

transaction data
collateral management
customer individual cash flow aggregation
accounts
Central bank holdings, lines, …
Assets
Market price, Duration / residual maturity, customer, rating, sector, encumbrance …
cash flow
30-40 largest customers & 10-20 customer groups
Additional dimensions product and time (date & time)
Kafka
basis data stresstest
(CashFlows, Assets)
results stresstest
raw data results stresstests
(cash flows, assets)
Dash Boarding
Reporting
R-Server
analysis
Parameter
individual Parameter
Standard scenarios (Stress)
Non-Standard scenarios (Stress)

Fig. 1 Kafka—machine learning backbone (© ifb SE)

Due to its ability to process huge data volumes in real time, Apache Kafka is a great choice for implementing the backbone of machine learning applications. Kafka can connect to many different databases and applications using standard Kafka connectors. In addition, Kafka topicsprovide data storage and decouple source systems from machine learning applications. Kafka can also stream data to Hadoop for long-term data storage or work in combination with cloud storage solutions (i.e. AWS S3[3]). While model training could also be implemented through Kafka Streams applications, it is more common to leverage Kafka as a "broker" and stream data into dedicated machine learning platforms, such as TensorFlow[4] or R-Server.

The diagram above depicts a typical system architecture for machine learning using Kafka as its backbone (Fig. 1).

2.2 Data Streaming and Preprocessing with Kafka

The following diagram depicts the high-level data flow for our liquidity forecastingapplication (Fig. 2).

For our demo applications, historical data is provided via csv files. In order to simulate the occurrence of events over a full day timeline, a simulator app

[3] See Amazon (2020).

[4] See Google (2020).

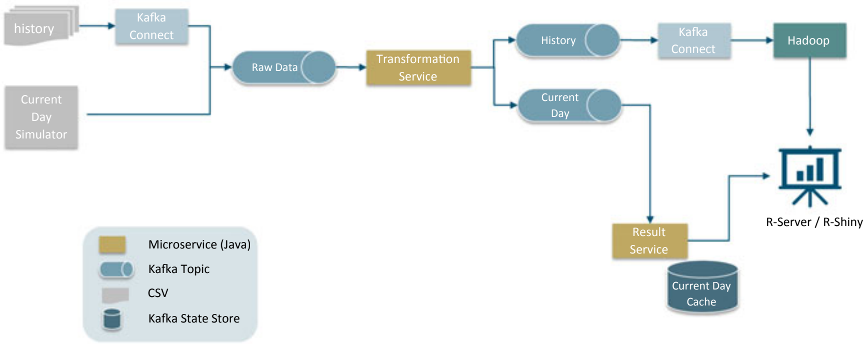


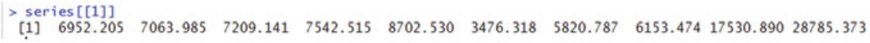
Fig. 2 Intraday liquidity app—data flow (© ifb SE)

was developed that can send events in time-lapse mode. New data could also be sent as csv files because Kafka Connect instantly streams data as soon as a new file arrives.

A transformation service, subscribed to the raw data topic, constantly checks if new records have arrived and transforms them into a harmonized format. The advantage of having a raw source data topic is that we can mirror the source systems and reprocess data from its raw form if needed. Once data is harmonized, all downstream services do not need to worry about data transformation and can simply subscribe to the output topic of the transformation.

All messages are stored in a history topic. For long-term storage, we send them to Hadoop, which is also connected to R-Server. In order to provide real-time liquidity forecasting for the current day, current messages are additionally sent to a dedicated topic (called Current Day). A result service aggregates the messages to the format required by R-Server and caches the results in its state store. It also provides a REST API to expose these results to R-Server.

A few notes regarding data management. Data retention time can be configured for each topic individually. As we store all historical data in Hadoop as well, the history topic does not need to store data for more than a couple of days, but it certainly could. The current day state store (Rocks DB) is backed up by the disk of the server running the microservice and a dedicated Kafka changelog topic. In case the result service fails, data is automatically recovered from disk or from the Kafka topic (in case disk is lost). State stores usually persist only the latest version for each a key in order to save space. State stores can be deleted programmatically or by manually deleting the corresponding local state directory. Individual records can be deleted by sending tombstone records (same key but with null value).



```
> series[[1]]
 [1]  6952.205  7063.985  7209.141  7542.515  8702.530  3476.318  5820.787  6153.474 17530.890 28785.373
```

Fig. 3 Cumulative Cashflow 8:00 (© ifb SE)

2.3 Machine Learning Method Review

In this section, we review the machine learning method for intraday liquidity, which is introduced in Liermann et al. (2019). We are going to use clustering to group historical cash flows with similar structures into the same clusters. Clustering is a statistical tool widely used to analyze sample structure by means of subdivision of data into groups—clusters—based on member similarities. For a given cash flow at a certain time of a new trading day, we aim to predict cash flow for the rest of the day. We search for a cluster which "looks" the most similar or has the highest similarity (called nearest cluster) to the target time series. The members of the nearest cluster should have a structure similar to the target time series and can thus be used to "forecast" the remaining part of the target time series.

The data used for the numerical experiment is computed based on the traded cash flows (inflows or outflows) of 20 clients from 10 different countries, which is generated daily between 8 a.m. and 4 p.m. from 1990.10.01 to 1990.10.15. Cumulation by day and by client cash flows are time series, which are computed as Fig. 2. An example of cumulative cash flow is shown in the following figure, in which the first element is the cumulative cash flow at 8 a.m. (Fig. 3).

In order to compute the similarities between time series, we use two distance metrics: shape-based distance and global alignment kernel distance. Both are specific metrics for time series and introduced in Liermann et al. (2019). Thirteen clusters are extracted, and the clustering result is provided in the Fig. 4.

Suppose that, on a new trading day, the cumulative cash flow is given until a certain time, e.g. until 10 a.m. (black curve). Cluster 8 is suggested to be the nearest cluster. We therefore forecast the target time series (colorful curve) using the members of cluster 8 (Fig. 5).

2.4 R with Kafka

We use package *httr* in R to connect to Kafka by sending a GET request to URL/API introduced in the last section. In this section, we provide a small example with URL https://swapi.dev/api/people/, where STARWARS data including people, films, species, starships, vehicles and planets are saved.

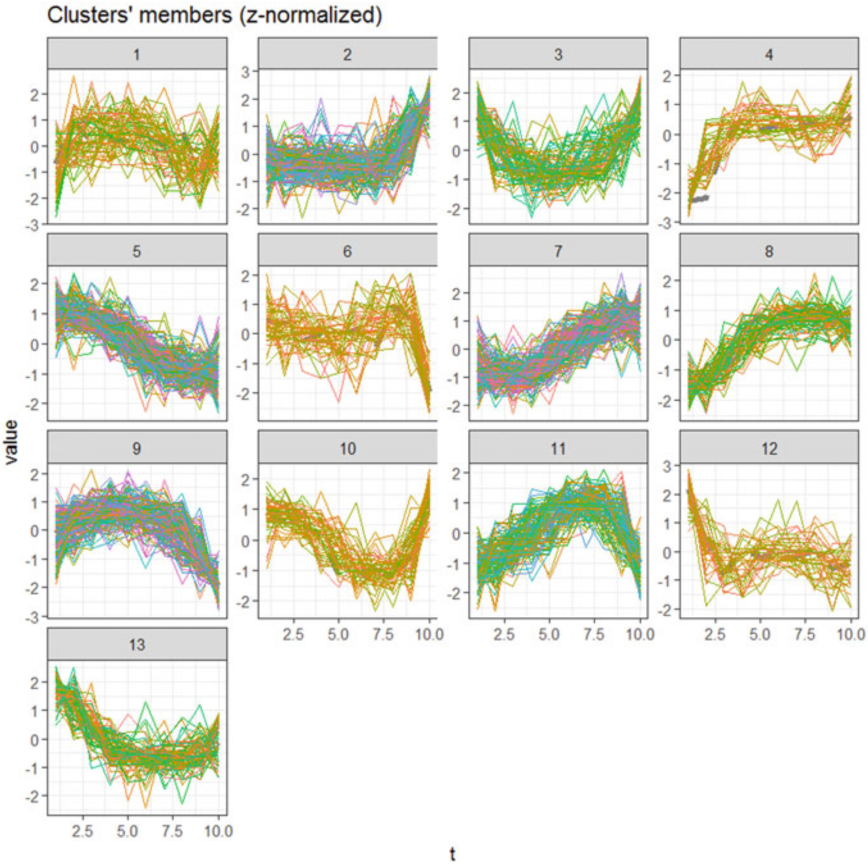


Fig. 4 Example cluster (© ifb SE)

By using the *GET()* function, we retrieve the information identified by the given URL. The object result has the type “response”, and its content can be extracted by using the *content()* function (Fig. 6).

The content information has the JSON format and needs to be transferred into R-data.frame. Here the package *jsonlite* is used (Figs. 7 and 8).

The JavaScript Object Notation (JSON) is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value).

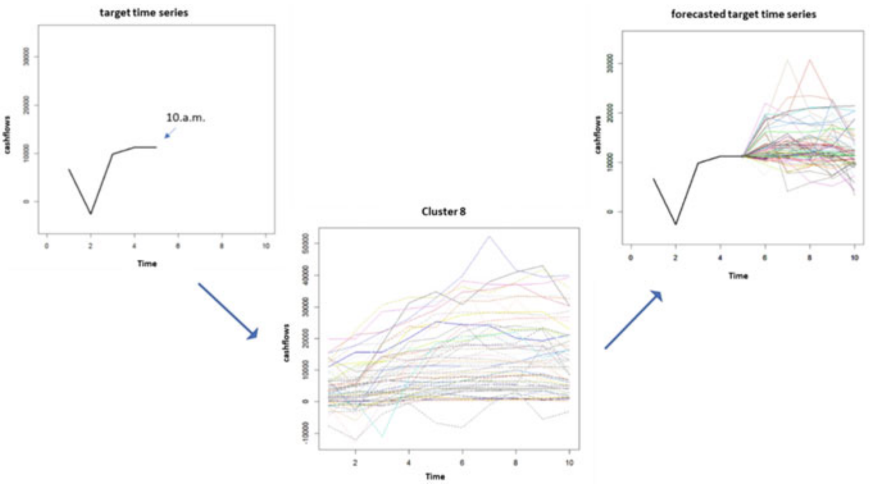
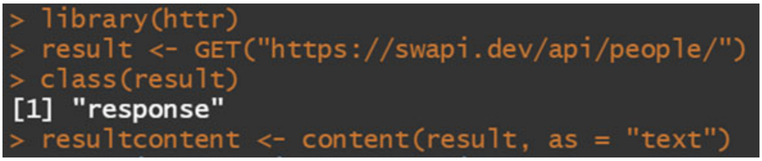
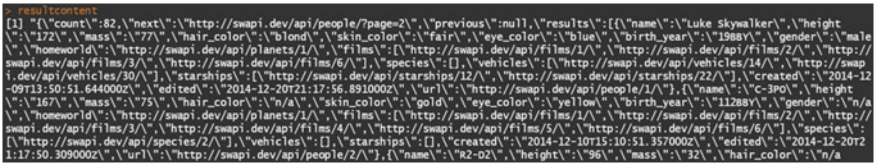


Fig. 5 Prediction—main steps (© ifb SE)



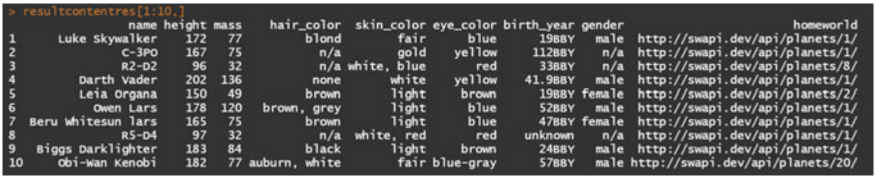
```
> library(httr)
> result <- GET("https://swapi.dev/api/people/")
> class(result)
[1] "response"
> resultcontent <- content(result, as = "text")
```

Fig. 6 Code example (© ifb SE)



```
> resultcontent
[1] "{\"count\":82,\"next\":\"http://swapi.dev/api/people/?page=2\",\"previous\":null,\"results\":[{\"name\":\"Luke Skywalker\",\"height
\":\"172\",\"mass\":\"77\",\"hair_color\":\"blond\",\"skin_color\":\"fair\",\"eye_color\":\"blue\",\"birth_year\":\"19BBY\",\"gender\":\"male
\",\"homeworld\":\"http://swapi.dev/api/planets/1/\",\"films\":[\"http://swapi.dev/api/films/1/\",\"http://swapi.dev/api/films/2/\",\"http://
swapi.dev/api/films/3/\",\"http://swapi.dev/api/films/6/\"],\"species\":[],\"vehicles\":[\"http://swapi.dev/api/vehicles/14/\",\"http://swap
i.dev/api/vehicles/30/\"],\"starships\":[\"http://swapi.dev/api/starships/12/\",\"http://swapi.dev/api/starships/22/\"],\"created\":\"2014-12
-09T13:50:51.644000Z\",\"edited\":\"2014-12-20T21:17:56.891000Z\",\"url\":\"http://swapi.dev/api/people/1/\"},{\"name\":\"C-3PO\",\"height
\":\"167\",\"mass\":\"75\",\"hair_color\":\"n/a\",\"skin_color\":\"gold\",\"eye_color\":\"yellow\",\"birth_year\":\"112BBY\",\"gender\":\"n/a
\",\"homeworld\":\"http://swapi.dev/api/planets/1/\",\"films\":[\"http://swapi.dev/api/films/1/\",\"http://swapi.dev/api/films/2/\",\"http://
swapi.dev/api/films/3/\",\"http://swapi.dev/api/films/4/\",\"http://swapi.dev/api/films/5/\",\"http://swapi.dev/api/films/6/\"],\"species\":
[\"http://swapi.dev/api/species/2/\"],\"vehicles\":[],\"starships\":[],\"created\":\"2014-12-10T15:10:51.357000Z\",\"edited\":\"2014-12-20T2
1:17:50.309000Z\",\"url\":\"http://swapi.dev/api/people/2/\"},{\"name\":\"R2-D2\",\"height\":\"96\",\"mass\":\"32\",\"hair_color\":\"n/a
```

Fig. 7 Content with JSON format (© ifb SE)



```
> resultcontentres[1:10,]
                name height mass    hair_color  skin_color eye_color birth_year gender                               homeworld
1     Luke Skywalker    172   77         blond        fair      blue      19BBY   male  http://swapi.dev/api/planets/1/
2              C-3PO    167   75           n/a        gold    yellow     112BBY    n/a  http://swapi.dev/api/planets/1/
3              R2-D2     96   32           n/a white, blue       red      33BBY    n/a  http://swapi.dev/api/planets/8/
4        Darth Vader    202  136          none       white    yellow    41.9BBY   male  http://swapi.dev/api/planets/1/
5        Leia Organa    150   49         brown       light     brown      19BBY female  http://swapi.dev/api/planets/2/
6          Owen Lars    178  120   brown, grey       light      blue      52BBY   male  http://swapi.dev/api/planets/1/
7 Beru Whitesun lars    165   75         brown       light      blue      47BBY female  http://swapi.dev/api/planets/1/
8              R5-D4     97   32           n/a  white, red       red    unknown    n/a  http://swapi.dev/api/planets/1/
9   Biggs Darklighter    183   84         black       light     brown      24BBY   male  http://swapi.dev/api/planets/1/
10    Obi-Wan Kenobi    182   77 auburn, white        fair blue-gray      57BBY   male http://swapi.dev/api/planets/20/
```

Fig. 8 Content with R-data.frame format (© ifb SE)

2.5 Demo and Screenshots

In this section, we want to introduce a demo. Firstly, a time machine is built and streams the cash flow for a given business day, where the test time stamp is set at 5 seconds and it is considered one business hour in real life.

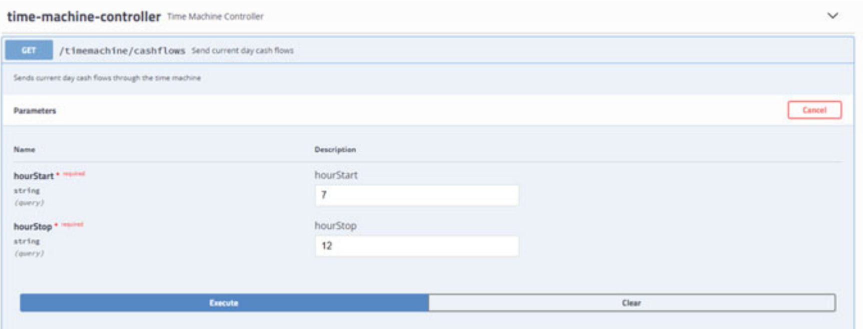
We set the starting hour and the stopping hour of the time machine at 7:00 a.m. and 2:00 p.m. This means that the time machine starts with the cash flow at 7:00 a.m., adds one more cash flow point for each business hour every 5 seconds and ends up with cash flow until 12:00 midday after (12 − 7) * 5 = 25 seconds (Fig. 9).

We extend the Shiny dashboard, which is introduced in article by adding an additional real-time forecasting page. The dashboard is connected to the time machine API given above and is set to refresh the connection every 5 seconds. The nearest cluster and forecast are also handled in real time. We start the dashboard as shown in the following screenshot, where no cash flow arrives and therefore no forecast is handled. The sidebar shows how many cash flow points are collected and which cluster is chosen for forecasting (Fig. 10).

The first cash flow point arrives at 7 a.m. The remaining part of cash flow is forecast by using the historical cash flows from cluster 4. Since we only have one point, the cluster with the closest starting point is then chosen (Fig. 11).

In the next two screenshots, the cash flows for 8 a.m. and 9 a.m. are collected and cluster 4 remains for projecting the forecasted flows (Fig. 12).

We see that the cash flow does not decrease anymore at 10 a.m. and starts to increase slightly. This means that cluster 4 is no longer appropriate. Instead, cluster 7 is chosen as the cluster with the most similar structure (Figs. 13 and 14).



time-machine-controller Time Machine Controller

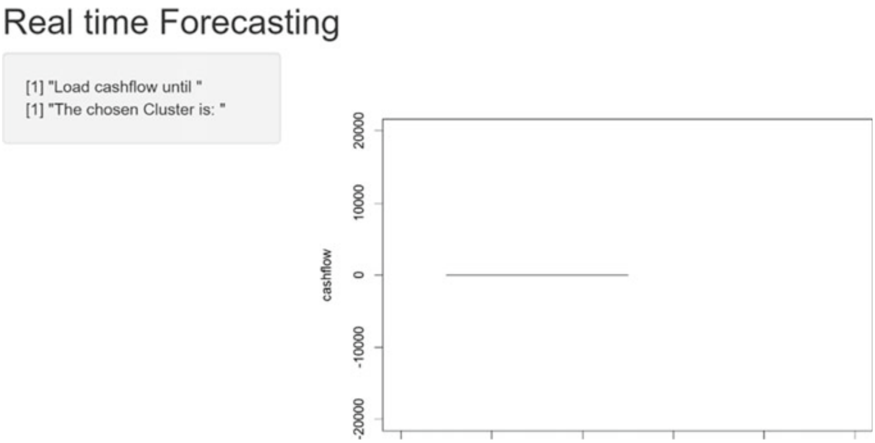
GET /timemachine/cashflows Send current day cash flows

Sends current day cash flows through the time machine

Parameters

Name	Description
hourStart * required string (query)	hourStart 7
hourStop * required string (query)	hourStop 12

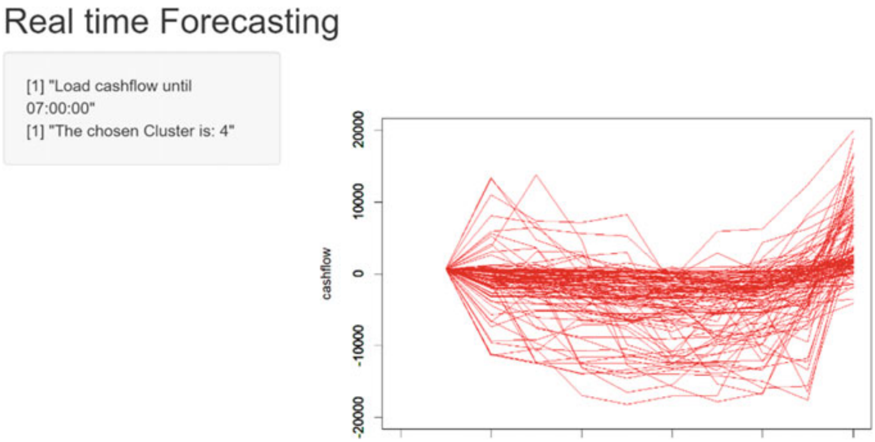
Fig. 9 Screenshot—time machine controller (© ifb SE)



Real time Forecasting

```
[1] "Load cashflow until "
[1] "The chosen Cluster is: "
```

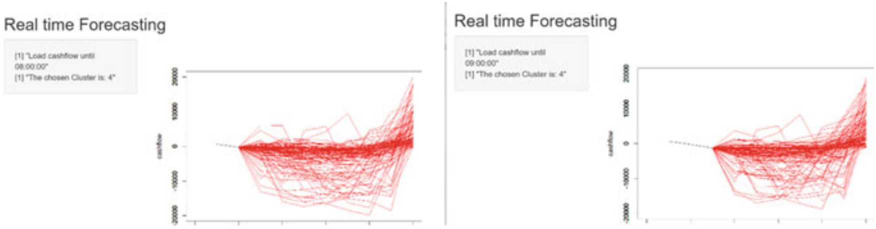
Fig. 10 Screenshot—real-time forecasting—situation at 7:00 a.m. (© ifb SE)



Real time Forecasting

```
[1] "Load cashflow until
07:00:00"
[1] "The chosen Cluster is: 4"
```

Fig. 11 Screenshot—real-time forecasting—situation at 7:00 a.m.—prediction (© ifb SE)



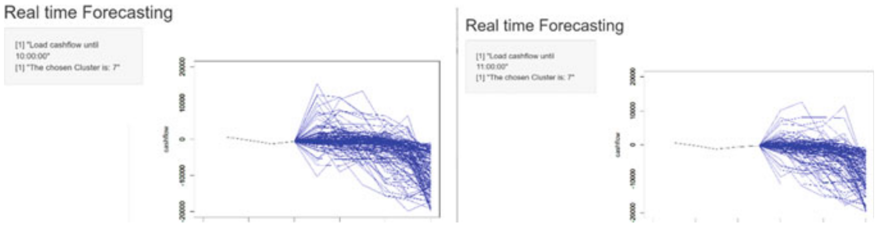
Real time Forecasting

```
[1] "Load cashflow until
08:00:00"
[1] "The chosen Cluster is: 4"
```

Real time Forecasting

```
[1] "Load cashflow until
09:00:00"
[1] "The chosen Cluster is: 4"
```

Fig. 12 Screenshot—real-time forecasting—situation at 8:00 a.m. and 9:00 a.m.—prediction cluster 4 (© ifb SE)



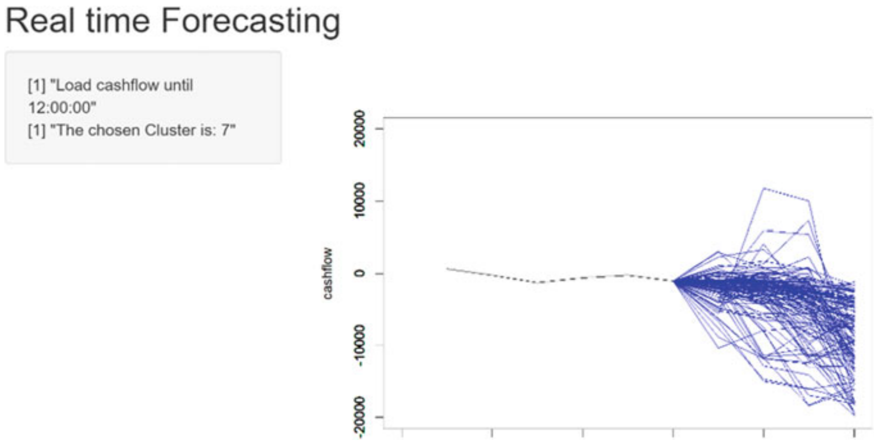
Real time Forecasting

[1] "Load cashflow until 10:00:00"
[1] "The chosen Cluster is: 7"

Real time Forecasting

[1] "Load cashflow until 11:00:00"
[1] "The chosen Cluster is: 7"

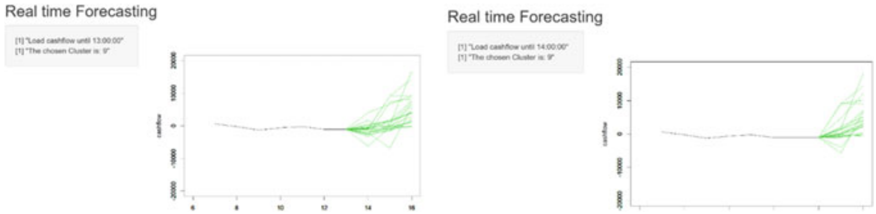
Fig. 13 Screenshot—real-time forecasting—situation at 10:00 a.m. and 11:00 a.m.—prediction cluster 7 (© ifb SE)



Real time Forecasting

[1] "Load cashflow until 12:00:00"
[1] "The chosen Cluster is: 7"

Fig. 14 Screenshot—real-time forecasting—situation at 12:00 midday—prediction cluster 7 (© ifb SE)



Real time Forecasting

[1] "Load cashflow until 13:00:00"
[1] "The chosen Cluster is: 9"

Real time Forecasting

[1] "Load cashflow until 14:00:00"
[1] "The chosen Cluster is: 9"

Fig. 15 Screenshot—real-time forecasting—situation at 1:00 p.m. and 2:00 p.m.—prediction cluster 9 (© ifb SE)

Cluster 7 is consistently chosen for forecasting until 12:00 midday. After that it changes to cluster 9, since the cash flow starts to decrease again (Fig. 15).

With the help of the Kafka streaming technique, we can establish the real-time forecast with higher flexibility. The prediction/forecast is self-retrained during the arrival of more cash flow points.

3 Summary

The chapter offers an extension of the machine learning model for predicting cash flows on a customer level (for the machine learning model see Liermann et al. 2019). The use case is embedded into a real-time data streaming framework using Apache Kafka.

The practicability is shown by an example illustration of the liquidity risk development (see Figs. 10, 11, 12, 13, 14 and 15). The practical application shows how historical cash flow (cumulated flow[5]) data can be used to detect patterns (behavior clustering) in the customer behavior. This clustering is then aligned with customer behavior observed (in real time) on the actual day. This customer behavior alignment can be used to predict the cash flow for the remaining day. While the intraday liquidity (risk) process is a highly intuitive process (relying on the long-term experience of the persons involved), the approach presented can improve the quality of decisions, especially while the process is based on intuition.[6] The ability of the system presented to create analytics and historical predictions can bring the decision-making process from system 1 of Kahneman's cognitive biases model more toward system 2.

As an extension of the environment, historical data in combination with clustering (generated by machine learning models) can be used to identify anomalies in cash flows (or cumulative flows). The ability to identify previously unknown patterns that can suggest a fundamental change is the most important skill for risk management in a continuously and sometimes disruptively changing world.

This chapter shows an interesting combination of data processing and data analysis technology in the context of a business challenge. The general layout of the solution architecture can be transferred to other business applications requiring quick and profound decision-making (e.g. market risk, limit processes).

[5] For definition see Liermann et al. (2019).

[6] System 1 in the concept of cognitive biases by Nobel Memorial Prize in Economic Sciences Laureate Daniel Kahneman, see Kahneman (2011).

Literature

Amazon. 2020. *Amazon Simple Storage Service S3*. Accessed November 28, 2020. https://aws.amazon.com/de/s3/.

Apache Software Foundation. 2020. *Apache Kafka*. Accessed November 28, 2020. https://kafka.apache.org/.

Google. 2020. *Tensorflow*. Accessed November 29, 2020. https://www.tensorflow.org/.

Kahneman, Daniel. 2011. *Thinking, Fast and Slow*. New York: Farrar, Straus and Giroux.

Krätz, Dennis, and Michael Morawski. 2021. "Architecture Patterns—Batch & Real-Time Capabilities." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Liermann, Volker, Sangmeng Li, and Victoria Dobryashkina. 2019. "Intraday Liquidity—Forecast Using Pattern Recognition." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Steurer, Ralph. 2021. "Kafka—Real Time Streaming for the Finance Industry." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Data: A View of Meta Aspects

Data can be stored, transferred and analyzed, and several chapters in the volume are dedicated to the subjects of storing, transferring and analyzing data. The third part picks up a few topics of an overarching nature.

The starting chapter of this part (Akhgarnush 2021) explores the different dimensions of data sustainability. The chapter subsequently touches on the subject of data security compliance. Furthermore, the chapter is dedicated to analyzing social and environmental aspects.

The second chapter (Floß and Velauthapillai 2021) takes a closer look at the data requirements and opportunities of an insurance company. Insurers have been data-driven for centuries, but the Internet of Things and its sensor data open up opportunities for new business models or more accurate data-based predictions.

The third chapter of this part (Czwalina et al. 2021) examines the regulation around data protection. It investigates the existing and significantly varying regulations in certain regions (Europe, USA,[1] Canada and Japan). The chapter explores the impact of data protection laws on business models and technological progress.

Literature

Akhgarnush, Eljar. 2021. “Data Sustainability—A Thorough Consideration.” In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing,*

[1] California.

and Analysis, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Czwalina, Marie Kristin, Matthias Kurfels, and Stefan Strube. 2021. "Data Protection Regulation." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Velauthapillai, Jeyakrishna and Johannes Floß. 2021. "Special Data for Insurance Companies." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Data Sustainability—A Thorough Consideration

Eljar Akhgarnush

1 Introduction

Data is ubiquitous and the starting point of every move. The more we progress—digitalizing our processes—the more tangible data becomes and currently, at the same time, less tangible. How is that you might ask: we produce ever-increasing volumes of data but lack the capability or rather the capacity to process, let alone control all these unimaginable masses. Due to this vacuum and the overwhelming importance of data itself, chaos prevails in dealing with data, and governments around the world recognize the need for regulations but are struggling with right approaches so far. The debate about implementations of digital security measures for organizations has therefore become tense in the face of the increasing threat of cybercrime and data misuse. The digitalization of organizations exposes them to these two issues, while most of them still lack serious commitment to the protection of data (Matthews 2019) (Data Security), but also to ethically and morally correct collecting of data and use of collected data (Data Compliance) (United Nations 2019). Apart from these two components, the question of appropriate data usage arises, meaning, for example, the creation of redundant or obsolete data (Data Trash) resulting in the consumption of more resources.

E. Akhgarnush (✉)
ifb SE, Grünwald, Germany
e-mail: Eljar.Akhgarnush@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_8

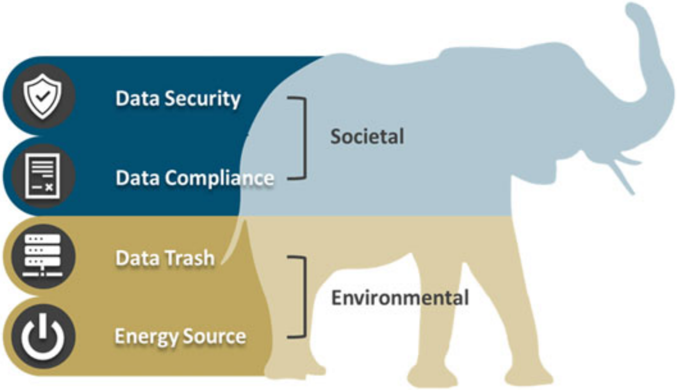


Fig. 1 Data sustainability components (© ifb SE)

Resource consumption leads to the next question: the one regarding energy usage or rather source, i.e. how the energy infrastructure within the data branch looks (energy source). These four introduced components together form the topic of Data Sustainability (Fig. 1).

While the first two, Data Security and Data Compliance, deal with societal aspects like the well-being of individuals and groups, the latter two, Data Trashand Energy Source, provide the link to environmental aspects like resources.

In the following sections, each component will be explained individually, describing in detail the matters they cover. Subsequently, this chapter will conclude with an outlook for conceivable future developments that may shape the discussions yet to come.

2 Data Security

One does not need to have a crystal ball to recognize the inevitable necessity of ever greater security measuresfor global data. Yet it may be surprising to know that Data Security dates back to the late 70s, when computers evolved from exclusive devices to proliferating practical data processing terminals (Denning 1982). The difference now to back then lies in the previously mentioned ubiquity of not only the internet per se but also the sheer quantity of data. Digitalization in its current state and proliferation—as seen all over the business world—enables new opportunities for businesses but also endangers their security and that of their customers. It is the nature of progress that when new technologies arise, so do new problems (Fig. 2).

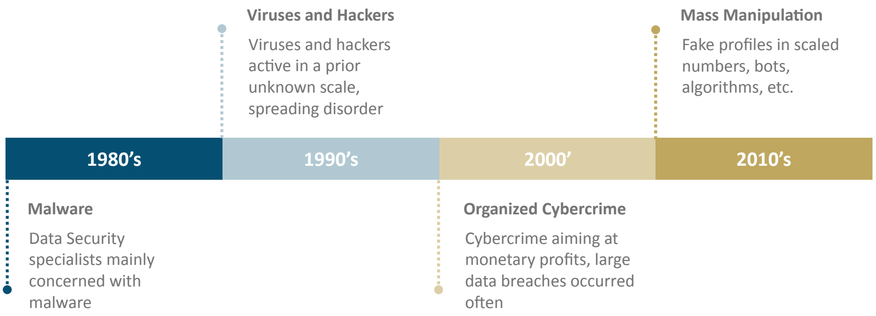

Viruses and Hackers
Viruses and hackers active in a prior unknown scale, spreading disorder
Mass Manipulation
Fake profiles in scaled numbers, bots, algorithms, etc.
1980's
1990's
2000'
2010's
Malware
Data Security specialists mainly concerned with malware
Organized Cybercrime
Cybercrime aiming at monetary profits, large data breaches occurred often

Fig. 2 Change of threats within data security (© ifb SE)

While in the 1980s Data Security specialists were mainly concerned with malware, in the 1990s viruses and hackers were active on a previously unknown scale, spreading disorder throughout the internet. One example was the hacking of the U.S. Department of Defense's computer systems by three teenagers, where they gained control of computer systems operated by the government, the military and private sectors (Foote 2018). These extensive cyber threats gave rise to more sophisticated data security measures like data encryption, data resilience (backups), virus detection steps, etc.

In the 2000s, cybercrime activities took on a new scale yet again, becoming organized in the sense of aiming for monetary profits. Large data breaches occurred often, for example, the Adobe data breach in October 2013 with 153 million user records and 3 million encrypted customer credit card records impacted, Canva in May 2019 with 137 million user accounts impacted, LinkedIn twice in 2012 and 2016, Equifax in July 2017, eBay in May 2014 and many more. Due to their ever-increasing numbers, but also improving adaptability regarding new technologies (e.g. AI) and communication in order to collaborate among themselves, organized groups of hackers still pose an immediate threat. In a recent example from 2019, fraudsters used artificial intelligence to mimic the voice of a company's CEO, resulting in a fraudulent transfer of €220,000 (Stupp 2019). It takes technology only seconds or minutes to learn how to mimic a voice. AI enabling new cybercrime activities does not stop there: even manipulated videos can be created, where a person can be seen and heard saying things they never said (Vincent 2017).

However, in the 2010s, a new kind of cyber threat for companies, customers and governments alike arose in parallel with the known dangers, specifically the one concerning mass manipulation within the everyday life of social media users (also including LinkedIn). The manipulation takes place with so-called fake profiles in scaled numbers, bots, algorithms and such. Influencing opinions, spreading fake news and misleading societies by defamation of company brands and political figures are just some methods of the recent, additional face of cyber threat. The term cyber war gets to the heart of the matter here.

As we have seen, Data Security is constantly adapting to the evolving internet landscape, since digital criminals—like in many branches, but here especially so—will always find new ways to circumvent or break social norms and laws. This makes it all the more important to ensure physical security beforehand, i.e. minimizing the risk of unauthorized hardware access or damage and breaches due to human failure or misbehavior. Further education can be the key to the latter.

The link to sustainability is in the development of data security measures, when digitalizing a company. That does not mean digitalizing for the sake of digitalization, but for the sake of higher sustainability goals. Approaching digitalization in this way requires the company governance to consider various data security methods—quantum, multi-cloud security, encryption, tokenization, backup, etc.—resulting in a sustainable approach to data from customers, suppliers and so on (Looker Data Sciences 2018). In view of current discussions on this topic and its relevance per se, customers, business partners and governments alike pay special attention to this aspect. This makes it a crucial aspect for future endeavors and, in the end, for the success or failure of the company.

3 Data Compliance

The next component within the Data Sustainability framework is about complying with data protection or rather regulations on dealing with data. For instance, the "General Data Protection Regulation" governs the way data is protected in the European Union.[1] But the term Data Compliance goes beyond this: As the German Data Ethics Commission (in German "Datenethikkommission") states, numerous ethical, sociological and legal questions arise, which circle around the desired role and the shaping of new technologies in our society (Datenethikkommission 2019). When we look at the component in this way, we also need to consider topics like artificial intelligence systems, including the widely used and equally questionable algorithms in businesses and social networks. Additionally, data quality assurance plays a significant role, i.e. the content of the data: truthfulness, accessibility, depiction, etc.

[1] More details on the GDPR can be found in **Es ist eine ungültige Quelle angegeben.**.

A sustainable strategy within the scope of Data Compliance lies in striving for the utmost transparency toward customers, suppliers, partners and other stakeholders regarding data handling—of course, without harming the business by, for example, leaking confidential information to competitors. The transparency strategy should be followed in parallel to efforts going beyond governmental data compliance regulations. Besides being transparent about their data handling and efforts going beyond what is required, the affected stakeholders need easy and simple access to, for example, data settings: many websites throughout the world wide web still lack easy and simple user interfaces for visitors to accept, decline or adjust their data protection settings according to the European GDPR. Similarly, data must be provided with certain quality standards—depending on the recipients (government, customers, etc.)—which results in the underlining of the previous factor of accessibility.

There may have been a time, where violating the factors, as described in the previous paragraph, had sufficient incentives to go on with immorally and unethical data handling behavior. Today, however, the pressure from society, governments, the environment and competitors to abide by moral and ethical rules is too big to ignore. Violating data compliance rules results in a variety of downsides, where the company risks its long-term business prospects: reputational risks are more salient today, governmental sanctions are more frequent, calls for boycotts are more common, trustworthiness declines easier and so on. It is therefore in the interest of companies themselves to comply with data handling rules and go "overboard" with their own aspirations in this field.

4 Societal Conclusion on Data Sustainability

The Data Ethics Commission identified six different social benchmarks (or ethical and legal principles) linked to data sustainability. Their purpose lies in the sustainable handling of data, and they should be understood as guidelines. In this article, these benchmarks are connected to the societal components Data Security and Data Compliance. Moreover, these principles represent goals that sustainable data handling should meet or at least always focus on (Fig. 3).

Privacy: The right to privacy is intended to preserve an individual's freedom and the integrity of his or her personal identity. Potential threats to privacy include the wholesale collection and evaluation of data about even the most intimate of topics.

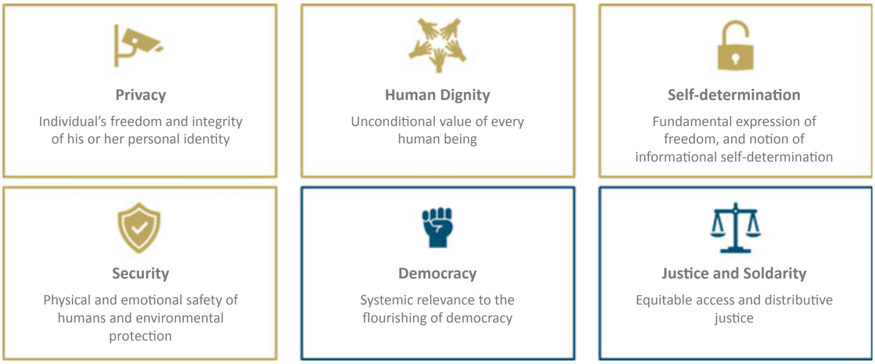

Privacy
Individual's freedom and integrity of his or her personal identity
Human Dignity
Unconditional value of every human being
Self-determination
Fundamental expression of freedom, and notion of informational self-determination
Security
Physical and emotional safety of humans and environmental protection
Democracy
Systemic relevance to the flourishing of democracy
Justice and Soldarity
Equitable access and distributive justice

Fig. 3 General ethical and legal principles according to the Data Ethics Commission (© ifb SE)

Human Dignity: Human dignity is a principle that presupposes the unconditional value of every human being, prohibiting such practices as the total digital monitoring of the individual or his or her humiliation through deception, manipulation or exclusion.

Self-determination: Self-determination is a fundamental expression of freedom and encompasses the notion of informational self-determination. The term "digital self-determination" can be used to express the idea of a human being a self-determined player in a data society.

Security: The principle of security relates not only to the physical and emotional safety of humans but also to environmental protection, and as such involves the preservation of vital assets. Guaranteeing security entails compliance with stringent requirements, e.g. in relation to human/machine interaction or system resilience to attacks and misuse.

Democracy: Digital technologies are of systemic relevance to the flourishing of democracy. They make it possible to shape new forms of political participation, but they also foster the emergence of threats such as manipulation and radicalization.

Justice and Solidarity: In view of the vast amounts of power being accumulated using data and technologies, and the new threats of exclusion and discrimination, the safeguarding of equitable access and distributive justice is an urgent task. Digitalization should foster participation in society and thereby promote social cohesion.

5 Data Trash

As mentioned in the introduction to this article, we produce ever-increasing volumes of data. This is data with partial redundancy, i.e. a significant portion of it could be avoided. There are options like "prefers-reduced-motion" within operating systems, but most browsers (Steiner 2019) also reduce data intake by minimizing animated motions. Some websites offer a reduced view, which limits the display of the website feature to functionality. Moreover, some messenger services advertise their data economy, intentionally and actively contributing to a data-saving society (Singh 2021). However, these are simply drops in the bucket. Among others, streaming services, social media, company data (including Internet of Things and transactional data), etc. account for the biggest share of data creation (Marr 2018).

A redundancy in social media lies in, for example, the display of the same content to users again and again. Snapchat with its temporary uploading/sharing system—whether intentionally or unintentionally—provided one way of handling the redundancy in social media and beyond. This is where real-time analytics come into play: switching from retrospective analysis to real-time analytics enables companies to store data for a shorter time (if storing is required at all). That means, the progress of technology not only leads to ever-increasing data volumes but also ever-increasing data analytics velocities. For companies, therefore, digitalization should not only mean going digital for the sake of modernizing, but also for the sake of using as well as storing data more efficiently. For this purpose, the right partnerships and systems need to be put in place, leveraging the quality of data and its interpretation for predictions. This also implies breaking up data silos.

The component of Data Trash is relatively unexplored, thus presenting possibilities for companies and other stakeholders alike to come up with more ways aside from real-time analytics, temporary storage and reduced views. Furthermore, it forms the link between data and the environment across the topics of scope and resources, shifting the focus to nature itself.

6 Energy Source

Energy Source is the component that links Data Sustainability directly to environmental issues. While many are already paying close attention to climate-friendly behavior, only a few are aware of the extensive environmental protection potential that lies dormant every day right under their fingertips: on computers, smartphones and tablets (E.ON SE 2018). Because every megabyte of cloud storage used, every email or Google search ends up in one of the numerous energy-hungry datacenters around the world and causes CO_2 emissions (not to mention the use of the devices themselves). And unfortunately, not all mainframes for our digital life are powered by green electricity. In this context, the following questions arise, among others: How much resources do data consume? What does the energy mix of the datacenters look like? Which inefficiencies exist within the energy infrastructures? Are there any smart circular system solutions? Which known or potential risks of the datacenter setup exist?

The energy consumption of datacenters is not only restricted to computing power, but also to the complex cooling systems. The more data we produce, the more computing power we need, and thus the more cooling is required. Though there are already attempts to tackle the issue of the latter, it is far from resolved. Volkswagen Financial Services, as an example, has set up a datacenter model from IBM in Braunschweig, Germany, where the waste heat is fed to private households as well as offices and used there for heating (Müller and Ostler 2019). In return, the cooled water is fed back into the datacenter. Another attempt is provided by Microsoft, which installed datacenters on the seafloor to improve the overall reliability of datacenters (Roach 2020). Nevertheless, there is a varying limit to how many heating datacenters underwater biotopes can take without altering the environment in a harmful way (e.g. the temperature). In northern parts of countries like Finland or Norway, the ambient temperature is sufficient for cooling.

Going back to the matter of energy consumption, without any progress in energy efficiency the consumption via internet applications would double every 1–2 years (Kölbel and Paal 2018). In the era of streaming services and 5G it is up to companies and customers alike to question their own behavior. Is it necessary to be connected and have access to data 24/7? Are there any lean data selections possible? Is it necessary to store that amount of data? Do I have to stream this now or could I use a DVD instead or even do without it entirely? Providing intelligent solutions to these questions without losing any customers will be a difficult task for companies, in order to act sustainably in their data infrastructure while remaining customer friendly. Establishing a green energy mix for the datacenters is one of the easier obstacles, even though the digital world threatens to cancel out the CO_2 savings made by renewable energy sources.

7 Environmental Conclusion on Data Sustainability

Considering the benchmarks or goals introduced in the Societal Conclusion on Data Sustainability, there are some corresponding deductions regarding the environmental aspect, too. These should be understood as guidelines, when implemented in a sustainable framework (Fig. 4).

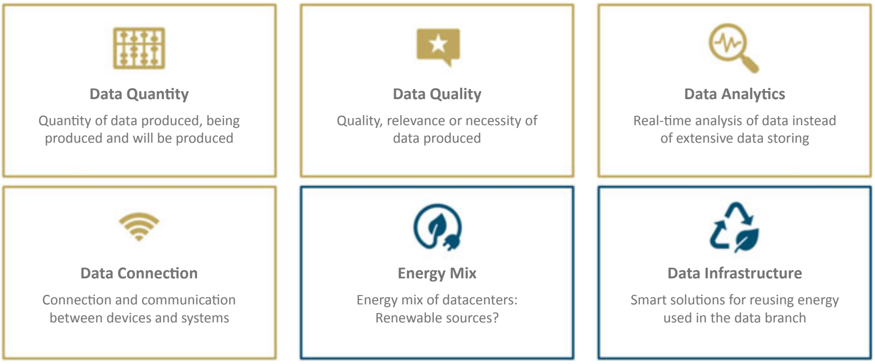
Data Quantity: Limiting the amount of data created will have direct positive repercussions on resource consumption. Avoiding generating Data Trash should be a priority for individuals as well as groups. It requires questioning customer behavior and company structures or business models.

Data Quality: Producing the right amount of data is effective but ensuring the production of the right data is efficient. Thus, it must be ensured by the source of the data creation that it is the appropriate quality in the sense of corresponding information content.

Data Analytics: Further progress and early implementation of these ensure a reduction in data storage through faster deductions from data. Therefore, real-time analysis and streaming of data enable market agents of the data branch to increase both efficacy (quantity) and efficiency (quality).

Data Connection: 5G scales up, among others, the Internet of Things to previously unimaginable levels. The question of how connected our environment of devices and systems needs to be remains. Providing data solely for the sake of communication, thus, temporary storage, can be a solution.

Energy Mix: Supplying datacenters and similar with renewable energy power is in line with current sustainability endeavors regarding clean energy. There is no limit to such sources (solar, wind, geothermal, tidal and wave, etc.).



Data Quantity
Quantity of data produced, being produced and will be produced

Data Quality
Quality, relevance or necessity of data produced

Data Analytics
Real-time analysis of data instead of extensive data storing

Data Connection
Connection and communication between devices and systems

Energy Mix
Energy mix of datacenters: Renewable sources?

Data Infrastructure
Smart solutions for reusing energy used in the data branch

Fig. 4 Environmental data sustainability guidelines (© ifb SE)

Data Infrastructure: Creativity and progress go hand in hand when it comes to finding smart solutions in the infrastructure of, for example, datacenters, and how to reuse energy or rather minimize the impact of the stress put on resource consumption and the environment.

8 Summary

The topic of Data Sustainability may seem a lot to digest, but at the same time its components seem logical considering what awaits us soon. To avoid costly re-imagining of a company's own data landscape, Data Sustainability strategies should be developed and followed from the start. In addition, companies that engage in this topic early on can build a strong reputation as a modern, thinking organization. Further progress in technological and substantive areas will enable new ways of handling the data branch. Once again, far-sighted and flexible companies will come out stronger in this ever-changing and fast-moving world we live in, now.

Literature

Datenethikkommission. 2019. "Gutachten der Datenethikkommission." www.bmi.bund.de. October 23. Accessed June 4, 2020. https://www.bmi.bund.de/SharedDocs/downloads/DE/publikationen/themen/it-digitalpolitik/gutachten-datenethikkommission.pdf?__blob=publicationFile&v=6.

Denning, Dorothy Elizabeth Robling. 1982. *Cryptography and Data Security.* Monterey, California, May.

E.ON SE. 2018. *Internet: So hoch ist der Stromverbrauch des World Wide Web.* Accessed June 20, 2020. https://www.eon.de/de/pk/strom/strom-sparen/stromverbrauch-internet.html.

Foote, Keith D. 2018. *A Brief History of Data Security*, July 25. Accessed June 23, 2020. https://www.dataversity.net/brief-history-data-security/#.

Kölbel, Ralf, and Gabor Paal. 2018. *Faktencheck: Ökobilanz von Suchmaschinen*, September 4. Accessed June 25, 2020. https://www.swr.de/wissen/20-jahre-google-umweltfacts-zu-suchmaschinen/-/id=253126/did=22378814/nid=253126/d2azhl/index.html.

Looker Data Sciences. 2018. *Data Security.* Accessed June 18, 2020. https://looker.com/definitions/data-security.

Marr, Bernard. 2018. *How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read*, March 21. Accessed June 27, 2020. https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/?sh=60068bb060ba.

Matthews, Kayla. 2019. *How Digital Transformation Changes Security Needs*, January 19. Accessed June 28, 2020. https://www.information-age.com/digital-transformation-changes-security-needs-123478114/.

Müller, Dietmar, and Ulrike Ostler. 2019. *Würdig für den Deutschen Rechenzentrumspreis?* April 5. Accessed July 2, 2020. https://www.datacenter-insider.de/rz-abwaerme-versorgt-400-wohnungen-a-816813/.

Roach, John. 2020. *Microsoft Finds Underwater Datacenters are Reliable, Practical and Use Energy Sustainably*, September 14. Accessed July 4, 2020. https://news.microsoft.com/innovation-stories/project-natick-underwater-datacenter/.

Singh, Jagmeet. 2021. *WhatsApp Vs Signal, Telegram, Facebook Messenger: What Data Does Each App Collect?* January 8. Accessed January 25, 2021. https://gadgets.ndtv.com/apps/news/whatsapp-facebook-messenger-telegram-signal-imessage-user-data-collect-privacy-labels-apple-app-store-2349216#:~:text=In%20contrast%2C%20Signal%20seems%20to,highlighted%20on%20the%20App%20Store.

Steiner, Thomas. 2019. *Prefers-Reduced-Motion: Sometimes Less Movement is More*, March 11. Accessed July 10, 2020. https://web.dev/prefers-reduced-motion/.

Stupp, Catherine. 2019. *Fraudsters Used AI to Mimic CEO's Voice in Unusual Cybercrime Case.* August 30. Accessed July 18, 2020. https://www.wsj.com/articles/fraudsters-use-ai-to-mimic-ceos-voice-in-unusual-cybercrime-case-11567157402.

United Nations. 2019. *Big Data for Sustainable Development.* Accessed June 14, 2020. https://www.un.org/en/sections/issues-depth/big-data-sustainable-development/index.html.

Vincent, James. 2017. *New AI Research Makes it Easier to Create Fake Footage of Someone Speaking*, July 12. Accessed July 24, 2020. https://www.theverge.com/2017/7/12/15957844/ai-fake-video-audio-speech-obama.

Special Data for Insurance Companies

Jeyakrishna Velauthapillai and Johannes Floß

1 Introduction

In the early years of the digital revolution, most of the data came from websites: web searches, social media posts or online shopping. There is nothing special about this kind of data for an insurance company compared to, say, a retailer or a bank. Information about a newborn baby might be used by a retailer to sell baby clothes, by a bank to offer a mortgage for a bigger flat and by an insurance company to convince the customer of the need for life insurance. Different products for sure, but in all three cases the data was used for the same thing: targeted marketing. Even location data is mostly used for this purpose, e.g. to tell a potential customer about that fancy new coffee place right around the corner, or to identify the frequent traveler who might need a new suitcase.

In recent years, a new data source has emerged that could be to insurance companies more than just a new source of marketing data: wearables and the Internet of Things. They allow real-time data of an insured "object," be it a car, a house or the health of an individual, to be obtained. This individualized

J. Velauthapillai · J. Floß (✉)
ifb SE, Grünwald, Germany
e-mail: Johannes.Floss@ifb-group.com

J. Velauthapillai
e-mail: Jeyakrishna.Velauthapillai@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_9

data can have an impact of seismic proportions on the insurance business; it even endangers the whole business model itself. The core idea of an insurance is the pooling of different risk profiles into a common portfolio, and thereby lowering the overall risk. In the extreme scenario, personalized real-time data of the insured object (or subject) might allow for perfect predictions, leaving no uncertainty and thus completely prevent the pooling. Even though the extreme scenario will most likely never be reached, an increase in the accuracy of the risk calculations by orders of magnitude is possible—enough to radically change the insurance business: loss of high-risk individuals from the potential customer base and intense competition for the remaining low-risk customers.

An insurance company that does not want to go the way of the dodo has to adapt to the new situation. It has to learn how to include the modern data sources in its business model and how to make use of the potential of Big Data and artificial intelligence for risk assessment, marketing and—importantly—risk mitigation. Only in this way will it be possible to both compete for low-risk customers and still underwrite high-risk individuals.

This chapter shall help the interested financial professional as a starting point in their endeavor in the new insurance world. Using two prominent examples—wearables in Life and Health and telematics in car insurance—we will discuss what business opportunities will arise from individualized real-time data, what data sources one can use, what the challenges are and what steps can be taken already today. We will also have a look at data aggregators, which can help an insurer to ease the move to the modern data sources. And finally, we will discuss how the new data sources might change the insurance business and how an insurer should get prepared for these changes.

2 Wearables in Life and Health

Wearables, i.e. computer devices that can be worn on the body as a wristband, piece of jewelry or part of clothing, are a relatively old invention: as early as 1961, Thorp and Shannon developed a shoe with a built-in computer to cheat at the roulette table (Thorp 1998) (they claim not to have used it for more than a proof-of-concept trial). However, wearables that can track health-related data are a more recent development: in 2007, Fitbit released one of the first wrist-worn fitness trackers. Around the same time, smartwatches with similar capabilities appeared on the market. Current models can measure health characteristics ranging from step count to heart rate to sleep cycles and may even include external measures like pollution and UV exposure. And while not all products have turned out to be a success (Google Glass comes to mind), in 2019 alone 250 million devices were sold, with growth rates of 10% (Barbosa 2019). Including smartphones with their limited capability to collect fitness data as well, a large amount of personalized, real-time health data is potentially available.

2.1 Opportunities

How could an insurer use this data? Three main business opportunities have been identified: customer engagement, risk reduction and risk assessment.

Traditionally, health insurers struggle to engage with their customers. Contacts between policyholder and issuer are mostly sickness-related—not a good starting point for a conversation. Having an app on the customer's smartphone or smartwatch allows the insurer to stay in touch on a regular basis in a more pleasant setting. Indeed, since John Hancock introduced policies that included a free fitness tracker, they have reported an increased customer engagement (Barbosa 2019). Besides, such products might help to attract more fitness-prone, low-risk customers—an important aspect in countries where individual, risk-adjusted pricing in health insurance is limited or even prohibited by law. One should notice that this opportunity does not arise from the health data, but rather the wearable itself (non-health data from the wearable like location, contact lists, etc. are of course helpful in marketing, but not the topic of this chapter).

In many legislations, health insurers cannot adjust the premiums to the actual risk and may not reject insurance based on pre-existing conditions. Thus, it is important for an insurer—and in case of a high-risk customer base even vital—to lower the risk profile of its customers. Wearables offer an opportunity to do just that, i.e. to encourage customers to lead a more healthy lifestyle: simple push notifications to nudge the customer to leave their sofa and go for a walk, or setting and tracking workout goals to motivate the customer to pursue a regular exercise regime. It was even reported that wearables have been successfully applied to monitor and manage blood sugar levels of diabetic patients (Spender et al. 2019). Some studies show that these measurers may indeed lead to a healthier lifestyle and reduce claims in the short term, although it is yet to be seen whether these changes also persist in the long run (Spender et al. 2019) (Fig. 1).

The third opportunity is the use of personalized health data for risk assessment. Real-time data from a wearable could allow for a faster and more automated underwriting process. Having individual data could allow for a

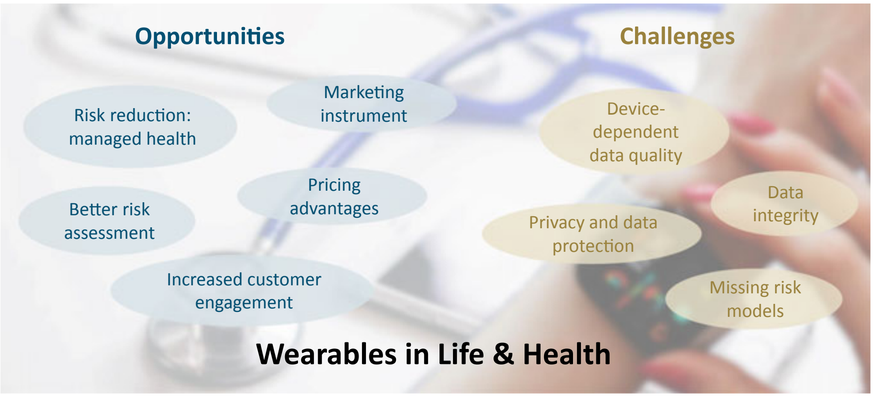

Opportunities
Challenges
Risk reduction: managed health
Marketing instrument
Better risk assessment
Pricing advantages
Increased customer engagement
Device-dependent data quality
Data integrity
Privacy and data protection
Missing risk models
Wearables in Life & Health

Fig. 1 Wearables in Life and Health—opportunities and challenges (© ifb SE)

more precise risk assessment, enabling more aggressive pricing in case of low-risk customers and generally lowering the risk costs. It is this third opportunity that makes most use of the wearables' health data, and we will now have a closer look at the challenges that lie ahead when employing this new data source.

2.2 Challenges

There are several obstacles to overcome before health data from wearables can be used in risk assessment, risk reduction and pricing: obtaining the data, ensuring its reliability and inserting the data into a risk model.

The first problem—obtaining the data—comes in two shapes: on the one hand, the customer has to wear a tracking device. While some individuals, in particular sport enthusiasts, may wear a fitness tracker permanently, other customers might need an incentive to even wear the device. Also, multipurpose devices like smartwatches instead of single-purpose fitness trackers may increase the usage. However, this advantage is bought by a reduced data quality (see below). Secondly, the customers have to be convinced to share their data with the insurer. Data scandals in recent years are not helping: studies have found that consumers are afraid to share their data with a health insurance supplier, fearing higher premiums in case of unhealthy behavior or the inability to leave such a program once entered (Tuzovic 2019). An insurance company will have to build trust to overcome these fears. Furthermore, consumers have become aware of the value of their data. Policyholders need to see a real benefit in wearing a fitness tracker and sharing their data, like reduced premiums, better service or discounts in partner programs—additional costs that narrow the business case.

A second problem is data reliability and consistency. As shown in the use case examples in part 2 of this series, consistent data is an indispensable condition for data science, and thus the challenge of data reliability is probably the biggest for the use of wearables in insurance risk models.

So far, most consumer wearables can only reliably measure the heart rate (Barbosa 2019). The reason is twofold. On the one hand, the sensors in wearables are often not optimized for a specific task, especially for multipurpose devices like smartwatches. Or they are simply of too low quality, in order to keep the device affordable. On the other hand, the algorithms that convert the sensor data into the desired measurable may not be sufficiently developed to fulfil the task accurately. Additionally, there is data discrepancy between different models, caused by a different choice of hardware by the manufacturers or differing software. Even with the same manufacturer, there will usually be discrepancies between older and newer models.

The policyholder themselves also introduces problems of data reliability due to different usage patterns. Some customers may wear the tracker every time they leave the house, others only every other day and the third group stops using it after an initial phase. Even worse, some customers may opt directly for fraud to obtain incentives: "A common example of data falsification consists in strapping a wearable device to a dog to generate the necessary daily step count" (Barbosa 2019).

The best solution to device-related reliability issues is for the insurer to develop their own fitness tracker with components and software of sufficient quality—an investment which most likely destroys the business case. User-related differences, on the other hand, would best be solved by the exact opposite approach: allow customers to wear a device of their own choosing, possibly a multi-purpose device like a smartwatch they willfully wear anytime. Finally, the "classic" solution to solve both problems would be to accept the high deviation within the data and use the law of large numbers to obtain usable average values. But then the real advantage of the new data source, individual health data, is lost. Currently, we do not see a viable way to achieve device- and user-related reliability simultaneously.

The final obstacle is the current lack of rating tables for the data that can be obtained by wearables (Abraham 2016). Although it is known that, for example, low daily activity correlates with increased morbidity, the precise functional relationship between, say, step count and claims is unknown (Barbosa 2019). Even worse, these functional relationships differ significantly between individuals (Hecksteden et al. 2018). An insurer would thus not only have to develop these functional relationships, but also to classify the policyholders correctly. A futile attempt for the time being, considering that even leading scientists are just at the beginning of this research.

Finally, once all these obstacles are overcome, the benefits might still be small. Customers in Life and Health insurance are generally unwilling to switch, and the pricing advantage obtained from wearable health data is unlikely to allow for sufficient discounts to overcome this barrier (Barbosa 2019).

2.3 Conclusion

As we have seen, the challenges regarding data reliability are too large at the moment to use health data from wearables for risk models or pricing. A significant increase in data reliability is necessary and even then the results may not offer a competitive advantage that outweighs the investment. However, does wearable technology still have a place in Life and Health insurance? We think the answer is maybe.

Firstly, the business opportunity for better marketing and customer engagement does not depend on data quality. However, it does not depend on wearables per se either: the main task here is to create a well-designed engagement program. A wearable might be part of this program, but it is not a necessary one.

Secondly, risk reduction does not rely on high-quality data either. But then again, we would argue that it does not depend on a wearable either. If the aim is to encourage more active behavior, the rudimentary step counter of a smartphone does the job as well. For other targets like reducing alcohol consumption and smoking, we fail to see how fitness data is of use at all. However, the wearable can be an incentive to engage in sport and can also be used as a reward for joining a program to increase one's health.

3 Telematics in Car Insurance

There are two kinds of risk factor in car insurance: static and dynamic ones. The former includes, for example, driving experience and car type, and, as the names suggests, are static during the usual insurance period of one year. Dynamic factors include distance traveled, road type, driving style, speeding or intoxication. They may change from trip to trip for the same driver and are—unlike most static factors—at least partially under the control of the policyholder. The most important dynamic factor is probably mileage: its inclusion into actuarial models significantly increases homogeneity within risk groups and leads to the elimination of some discriminatory characteristics like gender as risk factors (Tselentis et al. 2016).

With the exception of mileage, traditional car insurance completely neglects dynamic risk factors; instead, static factors are used as proxies. This leads to rather inhomogeneous risk groups (Ayuso et al. 2019). Additionally, the dynamic risk factor mileage is usually estimated at the beginning of the insurance period, thus becoming static, and the policyholder is left with the risk of over- or underinsurance.

3.1 The Different Kinds of Telematics Insurance

Telematics, using actual data from on-board units or even smartphones, offer a way to account for dynamic risk factors by directly including them. Telematics insurance comes in four different forms: usage-based insurance (UBI), pay-as-you-drive (PAYD), pay-how-you-drive (PHYD) and manage-how-you-drive (MHYD). Our definition of these four terms is given in Table 1; note that different definitions are used throughout the literature.

UBI is the simplest form and not that different from traditional insurance; however, instead of estimating the mileage beforehand and leaving the risk of over- or underinsurance with the policyholder, the premium is calculated only at the end of the coverage period taking the exact milage into account. Technically, UBI does not rely on any new data source at all, self-reporting suffices, though data from an on-board unit might be used to prevent fraud.

PAYD takes not only the exact mileage, but also other non-behavioral risk factors into account, e.g. kind of road or time of day. Thus, a mile driven in clear daylight on the motorway results in a lower premium than a mile at night over a narrow mountain pass. Unlike for UBI, telematics is actually needed here. However, current technology is sufficient: time and location in combination with map data will yield the necessary information. Thus, the data may even come from a smartphone instead of an on-board data unit. Also, the risk models more or less already exist.

PHYD finally adds behavioral risk factors to the menu: speeding, driving style or intoxication. Since behavioral risk factors are among the most important ones for car accidents (Stavrakaki et al. 2020), the benefits of PHYD to the insurer are clear. For the customer, the advantage of PHYD is that it offers them the opportunity to influence their risk and thus their premiums. In particular, low-risk individuals within a high-risk group could benefit the most.

Table 1 Definitions of the different kinds of telematics insurance (© ifb SE)

Method	Definition	Telematics?
Traditional	Apart from mileage, only static risk factors. Mileage is estimated beforehand and no rebate for lower mileage	No
Usage-based insurance (UBI)	Like traditional, but the premium is calculated at the end of the insurance period and based on exact mileage	Optional, to prevent fraud
Pay-as-you-drive (PAYD)	Like UBI, but non-behavioral risk factors like kind of road or time of day are included in premium calculation	Yes
Pay-how-you-drive (PHYD)	Like PAYD, but additionally behavioral risk factors (speeding, excessive acceleration, etc.) are included	Yes
Manage-how-you-drive (MHYD)	Like PHYD, but policyholder is taught how to drive more safely and lower their premium	Yes

PHYD comes with a lot of challenges similar to the ones we have seen for wearables in Life and Health: data quality for characteristics measured for PHYD varies between devices of different suppliers (Sun et al. 2020). An insurer should therefore install its own unit—with hardware of sufficient quality—in the policyholder's car. However, next to adding costs by not using sensors already existing in the car, this approach also contradicts the idea of PHYD: instead of data relating to an individual driver and their behavior, one only obtains data related to a car. The alternative approach is to use data recorded by a smartphone—but then being exposed to problems with data reliability and, worse, data integrity. Finally, once one has accurate data, one also needs to develop accurate risk models that include the new risk factors (then again, this is a general problem with any new risk factor).

MHYD is in principle the same as PHYD. However, in addition to calculating the premium based on the driving style, one also offers the driver suggestions on how to drive more safely and thus lower the premium. There are a couple of business opportunities for MHYD. Firstly, it allows insurers to offer high-risk drivers competitive premiums by turning them into low-risk drivers. Also, a car insurer that ended up for one reason or another with a high-risk customer base can lower its risk exposure by MHYD policies. Finally, it would be a good offer to corporate customers like trucking companies which could lower their costs by MHYD. Importantly, MHYD can already be implemented with low-quality data and crude risk models, if one leaves out the adjustment of the premium as in PHYD and only concentrates on managing the customers' driving style.

3.2 Which Way to Go?

Traditional car insurance covers the most important dynamic risk factor—which is mileage—but usually leaves the risk of under- or overinsurance with the customer. This counters the basic idea of insurance—taking the risk from the policyholder in exchange for a fee—and is thus a missed business opportunity. Furthermore, traditional insurance covers the car, not the driver—this does not square well with the decrease in car-ownership and increase in car-sharing, at least among the younger population. UBI solves the first problem by definition and can also easily be adapted to become a driver-centered insurance. Importantly, UBI has no technical challenges left.

PAYD has clear advantages for the insurer, by allowing for a better risk assessment and possibly an adjustment of the premiums to the actual risk. There are also no technical challenges left. However, it hardly offers anything to the customer: compared to UBI, the additional risk factors are out of the control of the driver. An insurer would thus have to offer additional incentives to convince customers of PAYD insurance, or alternatively not offer a UBI policy (yet face the risk that the competitors do). PHYD, on the other hand, would offer clear advantages to a customer who considers themselves a safe driver and also offers better risk assessment to the insurer. However, for the time being, it faces technical challenges and also lacks clear risk models. MHYD might present the answer. Even a crude understanding of the functional relationship between, say, speeding and claims costs is sufficient to offer the customer a rebate for safer driving, and thus an incentive to join MHYD. The insurer, on the other hand, could already make use of the well-understood non-behavioral dynamic risk factors, and use the data of early adopters to develop risk models for behavioral risk factors.

One possible legal issue with PAYD and PHYD is intransparency. Since an insurer does not want to reveal its risk models, it can only tell the customer superficially how their driving influences the premium. For legal reasons, the insurer thus might not be allowed to change the premium retroactively, but only for future periods: the model would become static again. Again MHYD offers a solution: instead of an unpredictable premium rebate at the end of the insurance period, the insurer could make a concrete offer in the middle of the period: "Lower your speeding violations by 50% and get a 10% rebate."

In conclusion, UBI is an easy way to offer insurance to drivers who prefer flexibility, and also get prepared for an eventual shift to a driver-fixed instead of car-fixed insurance; MHYD makes it possible to offer competitive insurance to high-risk drivers and can be used as a vehicle to get prepared for behavioral risk factors. Note though that the best risk models include both dynamic and static risk factors and (Ayuso et al. 2019).

4 Data Aggregators and Platform-Based Ecosystems

In order to benefit from AI and the new data sources, an insurer has to first gain access to this real-time, personalized data. One option is for the insurer to provide the data-collecting hardware and services to the customers itself. The insurer would have direct access to and full control of the data. However, in most cases this option would not come cheap, as already discussed in the preceding sections: hardware has to be chosen and possibly adapted for its purpose, and software has to be developed and tested on a range of different devices or at least models. The very short product cycles for smart consumer electronics as well as the rapidly changing consumption behavior of the younger generation mean that the development costs would not be one-off, but repetitive. And it does not help here that neither hardware nor software development are the core business of the average insurance company.

A simple solution to this problem would lie in a collaboration with data aggregators (Spender et al. 2019). These companies collect data on a grand scale from various sources, independently from specific hardware or software and specific services. Furthermore, they prepare the data, link it and use state-of-the-art machine learning and AI methods to analyze the data and to extract new information for the commercialization of the data.[1]

The type and depth of the collaboration between data aggregators and insurers varies considerably in practice. In the simplest case, the insurer would

[1] The term data aggregators is not clearly defined in the scientific literature and includes a variety of companies that collect, aggregate, prepare and analyze the data to sell the derived information to a third party. (Thomas and Leiponen 2016) subdivides these companies into six categories (data suppliers, data managers, data custodians, application developers, service providers, data aggregators) by their depth of involvement in the whole data processing.

just buy the relevant data collected from its customers. However, recently a more integrated approach can be observed: the insurance company becomes part of a large-scale multi-party collaboration ("ecosystem") (Thomas and Leiponen 2016). These ecosystems comprise partners providing the data (suppliers), partners specialized on specific aspects of data management (preparation, analysis, etc.) and the users of the data; some partners may take several roles. At the heart of such ecosystems is a technological platform that establishes the technical and process-related standards and provides the interface for the communication between the involved companies. In most cases, the data aggregators themselves are the owners of those platforms. These ecosystems allow an insurance company to not only benefit from a single kind of new data (e.g. health data from a wearable), but also additional information on their customers (e.g. income statements) with wide-ranging applications in, for example, marketing and cross-selling.

One interesting example for such a platform in the health sector is the COVALENCE Health 360 Platform (Big Cloud Analytics 2020), which is operated by the start-up Big Cloud Analytics. The COVALENCE Platform is based on the Intel IOT Platform and uses the IOT Gateway Technology to transport the data from the wearables to the cloud. Biometric data (sleep quality, activity level, skin temperature, heart frequency and many more) is generated by wearables from Fitbit, Pivotal or other devices and service providers. Big Cloud Analytics links the biometric data with demographic, behavioral and other available data, and uses advanced mathematical forecasting methods to calculate an individual health score (COVALENCE Wellbeing Score) for every customer. The health information is provided via a dashboard back to the customers, and the dashboard is also used to manage the participants' health, e.g. by sending push notifications to encourage more physical activity (insight.tech 2016). Via the dashboard, the insurer can also set incentives for a healthier lifestyle. Thus, the insurance company benefits from the health management capabilities of a wearable without the need to invest in hardware or software at all.

There are two caveats in order here. Firstly, the challenges described in the previous sections regarding data reliability and integrity are not resolved by such a platform, they are simply shifted to other partners within the ecosystem. And secondly, as the insurance company does not collect and own the data itself, it may become dependent on the platform (Fig. 2).

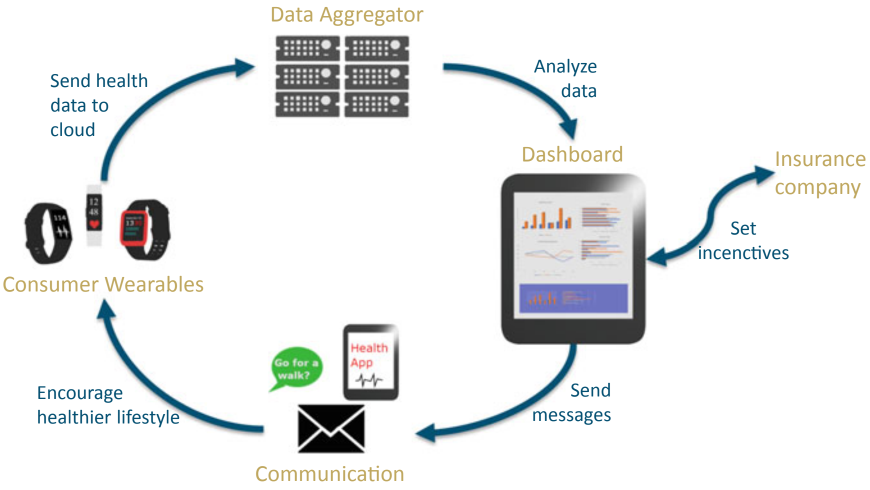


Fig. 2 COVALENCE health analytics platform (© ifb SE)

5 A Peek into the Future

Modern consumer electronics and smart devices allow the collection and transmission of personalized data on a real-time basis. This data could in principle be used by an insurance company to dramatically increase the accuracy of its risk calculations. However, as we have seen, even state-of-the-art devices fail to record data of sufficient quality for risk assessment, and it remains unclear whether this obstacle will be removed within the next decade.

Risk mitigation, on the other hand, becomes possible already with a qualitative understanding of the data. In Life and Health especially, risk mitigation via the help of wearables is gaining traction. Not surprisingly, since this sector is usually highly regulated and insurers are often unable to choose their customers or even set the premiums according to the actual risk. However, insurance companies should not overlook other fields like car insurance, where insurers can easily adapt the premiums to the risk: managed risk policies offer a real benefit to high-risk customers, and thus a business opportunity for the insurance company.

But let us now assume that advances in sensor technology, wearable software and risk modeling will make it possible to assess risk accurately on a personalized level, using real-time data: how would this affect the insurance business?

The simplest answer is: it would make insurance obsolete. The logic behind an insurance business model is to bundle individuals with high and low risk into one portfolio. Individual risks are pooled and shared, and this diversification leads to a reduction of the total risk. Of course, even with the best hardware and state-of-the-art AI, one will never reach a level where uncertainty is completely removed, but one could surely imagine scenarios where insurance could become unaffordable for a large fraction of potential customers. Low-risk customers, on the other hand, who with the help of technology could prove their low risk, might use this information to enjoy lower premiums. The insurance customer base would separate into non-insurable high-risk groups and low-margin low-risk groups. Adding insult to injury, data aggregators could allow any new entrants to the insurance business to gain huge amounts of risk data on the first day—established players lose their information advantage.

The first question is, how would society react to such a development? A look at the health insurance business can help: even without new data sources, risk assessment can lead to prohibitively high premiums for customers with pre-existing conditions. Almost all countries reacted to this with strong regulations, ranging from outlawing rejecting new customers to setting the premiums by law to purely tax-funded systems. In many countries, health insurers can only compete via their service level, not via premiums, and one might expect this development also for other insurance sectors. However, the situation might be completely different for sectors where individuals are considered responsible for their high premiums themselves. An obvious example is car insurance. If pay-how-you-drive schemes become more widespread and car insurance premiums mostly depend on the driving style, it would be rather surprising if society would protect dangerous driving by regulating the premiums.

The second question is: how could insurance companies react to this scenario? The answer is already here: by becoming service providers in a wider sense. Insurers have to change from companies that purely take care of the financial compensation in case of an adverse event to service providers who help their customers to live a safer and better life. By helping their customers to lower their risk, the insurer can not only underwrite high-risk customers left for good by the competition, but also mitigate effects from having to take in high-risk customers by law. And where the premium is set by the government, or the margin is very low due to fierce competition for low-risk individuals, additional services can help to set oneself apart from the competition and protect the margin.

One can already see this transition happening: car insurance policies with included roadside assistance, or the health insurer that helps its customers to stop smoking. Modern data sources and communication channels can be of great help in this endeavor.

Literature

Abraham, M. 2016. *Wearable Technology: A Health-and-Care Actuary's Perspective.* London: Institute and Faculty of Actuaries.

Ayuso, M., M. Guillen, and J. P. Nielsen. 2019. "Improving Automobile Insurance Ratemaking Using Telematics: Incorporating Mileage and Driver Behaviour Data." *Transportation* 46 (3): 735–752.

Barbosa, J. 2019. "The Business Opportunities of Implementing Wearable Based Products in the Health and Life Insurance Industries." Lisbon.

Big Cloud Analytics. 2020. *COVALENCE Health 360 Platform.* Accessed September 10, 2020. https://big-cloud-analytics.myshopify.com/collections/basis-peak-health-analytics-evaluation-kits.

Hecksteden, Anne, Werner Pitsch, Friederike Rosenberger, and Tim Meyer. 2018. "Repeated Testing for the Assessment of Individual Response to Exercise Training." *Journal of Applied Physiology* 124 (6): 1567–1579.

insight.tech. 2016. *Turning Healthcare Into Value.* January 1. Accessed September 10, 2020. https://www.insight.tech/health/turning-healthcare-data-into-value-2.

Spender, A., C. Bullen, L. Altmann-Richer, J. Cripps, R. Duffy, C. Falkous, M. Farrell, T. Horn, J. Wigzell, and W. Yeap. 2019. "Wearables and the Internet of Things: Considerations for the Life and Health Insurance Industry." *British Actuarial Journal* 24: 1–31.

Stavrakaki, Anna-Maria, Dimitrios I. Tselentis, Emmanouil Barmpounakis, Eleni I. Vlahogianni, and George Yannis. 2020. "Estimating the Necessary Amount of Driving Data for Assessing Driving Behavior." *Sensors* 20 (9): 2600.

Sun, S., J. Bi, M. Guillen, and A. M. Pérez-Marín. 2020. "Assessing Driving Risk Using Internet of Vehicles Data: An Analysis Based on Generalized Linear Models." *mdpi.com.* May 9. Accessed August 15, 2020. https://www.mdpi.com/1424-8220/20/9/2712.

Thomas, L., and A. Leiponen. 2016. "Big Data Commercialization." *IEEE Engineering Management Review* 44 (2): 74–90.

Thorp, E. O. 1998. "The Invention of the First Wearable Computer." *Digest of Papers: Second International Symposium on Wearable Computers* (Cat. No. 98EX215), 4.

Tselentis, Dimitrios I., George Yannis, and Eleni I. Vlahogianni. 2016. "Innovative Insurance Schemes: Pay as/How You Drive." *Transportation Research Procedia* 14: 362–371.

Tuzovic, S. 2019. "Treat or Trick: We Asked People How They Feel About Sharing Fitness Data with Insurance Companies." *The Conversation.*

Data Protection—Putting the Brakes on Digitalization Processes?

Marie Kristin Czwalina, Matthias Kurfels, and Stefan Strube

1 Introduction

The data generated in connection with emerging technologies is valuable from a business management perspective for several reasons: the amount of data produced and the ease of sharing information, the increasing processing power of the available machines, the combinability of information, and the durability of information, resulting in the fact that sovereignty over data is accompanied by power or influence.

The mere storage of information enables companies to analyze the behavior of a user and utilize the behavioral data, for example, for cookies-based advertising on the Internet. By harnessing information about a user's behavior, the advertising displayed on the site is tailored to the interests of the user. This increases the chance that the user feels addressed by the advertisement and this might lead to a business transaction. On the one hand, the evaluation of information by means of self-learning systems generates insights

M. K. Czwalina (✉) · M. Kurfels · S. Strube
ifb SE, Grünwald, Germany
e-mail: Marie-Kristin.Czwalina@ifb-group.com

M. Kurfels
e-mail: Matthias.Kurfels@ifb-group.com

S. Strube
e-mail: Stefan.Strube@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_10

that were previously beyond human capabilities, but on the other, it also leads to justified fears about the power that emanates from these decisions. An example of this is AI-controlled personnel selection procedures. Amazon tested a personnel selection tool up to 2018 which disadvantaged women and possibly encouraged other prejudices. The software was therefore never officially introduced (Sackmann 2018). The easy transferability of information between interested parties makes data misuse possible, because it has either been passed on illegally or transferred so frequently that its origin can no longer be traced.

In recent years, numerous events have come to light that attracted media attention because data was misused, or its use did not correspond to the original purpose of its collection. This public attention has not always resulted from outrage over violations of the law, but also moral sympathy for the data transfer (e.g. Edward Snowden and Wiki-Leaks founder Julian Assange) or the use of data, as in the case of Cambridge Analytica, which analyzed information about Facebook users and made it available for political advertising (Dachwitz 2020; Information Commissioner's Office 2020).

The protection of personal datais becoming more important as digitalization and the ability to evaluate information have increased. It is precisely for this reason that data protection regulations have been tightened at international level in recent years. The question therefore arises as to whether these tougher rules slow down digitalization and which challenges data protection laws pose for emerging technologies.

2 Data Protection

Data protection is the definition of principles, conditions and limitations by governments and authorities in order to mandate a higher degree of privacy for personal data management and includes penalties for data flow and data processing violations (Jia et al. 2018, p. 3). The principles, conditions and limitations apply to personal data. The definition of personal data in this context of data protection efforts varies. For this section, personal data is data or information relating to an individual, which enables the identification of this individual (Politou et al. 2018, p. 3).

An integral principle of data protection is educating the concerned parties about their rights in order to allow individuals to make informed decisions about the sharing of personal data and empower them to decide on the usage of personal data by data processing companies. For these companies, data protection is considered a guideline for the planning and design of data processing activities (European Commission 2020). This results in a direct link between data protection and development activities for modern technology as these technologies serve as instruments for the realization of data processing activities. In terms of processing activities, the concept Privacy by Design should be utilized as this modern principle for technology development demands the realization of data minimization, purpose limitations, transparency and control within development practices (Politou et al. 2018, p. 4).

In summary, data protection is a legal construct created by legislators to protect the data of an individual, and Privacy by Design is a design principle for software to foster the conscious handling of personal data. This leads to the question of why data protection and Privacy by Design are necessary?

In a data-driven society, data is perceived as a valuable asset due to the potential to create added value by collecting, analyzing and trading data for its commercial value (Politou et al. 2018, p. 3). In terms of personal data, traits, behaviors, footprints, work and leisure habits are monetized by producing targeted products, services, financial offers, advertising and healthcare solutions (Jia et al. 2018, p. 2). This development is reinforced by technological progress, for example, Big Data or machine learning, which offer new dimensions of collecting and processing personal data in order to personalize services based on the profiles deducted from the collected data sets (Wachter 2018, p. 3).

The commercial value of data monetized in companies exploits the privacy of the individuals through increased accessibility, re-identification, secondary use, exclusion and decisional interference. This results in the individual being a consumer of goods, information and services manufactured by and a public producer of valuable data for these companies (Jia et al. 2018, p. 2).

As personal data is a core element of technology-driven innovations from the perspective of companies, the individual as a producer of data tends to lose control and part of their independence and maturity, for example, by distributing personal data decentralized over the Internet without being aware of the data processing procedures utilized by companies to derive the commercial value of their data. The inferential analysis and linkage of disparate records conducted by these procedures can result in discriminatory treatment for the individual or an influence on the formation of an opinion (Politou et al. 2018, p. 4; Wachter 2018, p. 3).

In a survey researching the usage and perception of social media, 91% of the respondents stated the feeling of losing control over personal data collected and monetized by the social media networks. In the same survey, 61% of the respondents specified the need for more privacy protection mechanisms (Rainie 2018). Privacy focuses on the protection of the personal space of an individual and, in broader terms, the principles of data protection can contribute to privacy. But the principles of data protection focus on the processing of personal data and not on the privacy of the individual itself (Politou et al. 2018, p. 2). This focus is interpreted differently by national legislations. The scope of the concepts of personal data, the conditions for legitimate data processing and the extent of penalties differ according to national data protection legislation. Section 3, therefore, offers a review of a selection of data protection regulations.

3 Instances for Data Protection Regulation

Due to globalization and digitalization, approaches to data protection cannot be limited regionally. Nevertheless, there are different legislations which interpret personal data and its protection differently. In order to clarify how data protection affects digitalization, Sect. 3 provides a brief comparison and highlights special features.

3.1 EU—General Data Protection Regulation (GDPR).

The General Data Protection Regulation, also called "EU GDPR" or "EU Data Protection Basic Regulation," defines the circumstances under which personal data of EU citizens can be collected and processed. This regulation came into force on May 25, 2018 and is binding for all companies and institutions working with personal data of EU citizens. It is important to point out that the regulation applies to companies and institutions located within the EU, and to companies collecting or processing data of EU citizens (GDPR, General Data Protection Regulation 2016).

The goal of this directive is to guarantee the protection of personal data for all EU citizens and to ensure a uniform and free movement of data within the EU. Therefore, the EU GDPR specifies seven general principles for the processing of personal data (GDPR Section 5, General Data Protection Regulation 2016):

- **Faithfulness, lawfulness, transparency**: Personal data will be processed only in the way and to the extent discussed in the initial collection. The responsible person collecting the data must act in full transparency and disclose their identity. The owner of the data has the right to disclose their data at any time.

- **Purpose restriction**: The purpose of the data processing must be determined prior to the processing and must be clear and lawful. Use for a different purpose is only permitted in accordance with the legal basis. Consequently, the data may only be disclosed if the data owner has given their consent or on the basis of a justification by the data owner.
- **Data minimization**: Only data that is appropriate, significant and relevant should be collected.
- **Accuracy**: The data collected must be correct and complete before it is used. Incorrect data must be either corrected or deleted.
- **Storage limitation**: Data must be deleted immediately as soon as the goal of the data collection is fulfilled. Thus, data may only be retained until the purpose for which it was collected has been fulfilled. Furthermore, everyone can claim the "right to be forgotten." This means the data owner can order the data processing company to delete related personal data.
- **Integrity and confidentiality**: All stored data must be treated confidentially and protected against unauthorized processing, for example, by unqualified persons or parties. To this end, the company must implement both technical and organizational measures.
- **Accountability**: The data controller must demonstrate to the public and the relevant authorities that it fully complies with all data collection regulations.

In addition to the seven general principles of personal data processing, there are also other principles that apply to the consent to data processing. The approval of data processing is a central element of the new EU GDPR. The principle of forbiddance applies here, which means that without the prior consent or permission of the data owner, storage and processing is not permitted. In the event of non-compliance, sanctions of up to 4% of annual turnover can be imposed, which will be reviewed by established authorities within the EU countries (GDPR, General Data Protection Regulation 2016).

3.2 USA—California Consumer Privacy Act (CCPA).

In contrast to Germany ("BDSG-neu") and the EU (GDPR), there is no uniform data protection law in the USA. There are regulations for individual sectors such as trade or healthcare. In addition, these regulations sometimes differ from one state to another. These include the California Consumer Privacy Act (CCPA), which came into force in California on January 1, 2020 (CCPA, State of California 2020).

Companies and service providers bound by the CCPA must respect the rights of consumers. The most important point is that the law binds companies to inform consumers about the collection of personal data and their rights and opportunities. The CCPA must be complied with by all companies, service providers and revenue-generating entities that collect personal information and operate in California. The location of the company's registered office is irrelevant. In addition, one of the following three points must be fulfilled for the law to take effect (CCPA, State of California 2020):

- The annual revenue (gross) is greater than 25 million US dollars
- Per year, personal information is gathered from more than 50,000 customers, households or networked devices in California and used for commercial purposes
- Half or more of the annual revenue comes from the sale of personal information from Californian customers, households or network-enabled devices.

Overall, the CCPA offers these three crucial data subject rights in particular:

- Right of access: You may obtain information about the storage and use of your data by exercising your right of access and information. In doing so, you can find out which data has been collected and what it has been used for. In addition, the information is provided about which third parties have received which data (CCPA Section 1798.100, State of California 2020; CCPA Section 1798.110, State of California 2020; CCPA Section 1798.115, State of California 2020).
- Right of deletion: The CCPA provides consumers with the "right to be forgotten," whereby individuals can have all data that has been collected by a company or service provider deleted. There may be certain exceptions to this rule, which are set out in paragraph d (1–9) (CCPA Section 1798.105, State of California 2020).
- Right of opt-out: Consumers are offered the chance to protest against the sale of their personal data to third contractors. In addition, the disclosure of the data to third parties can be refused (CCPA Section 1798.120, State of California 2020).

The affected parties include all California residents and customers from around the world, as it also affects California-based companies collecting customer data outside of California.

3.3 Canada—Privacy Act

The Privacy Act in Canada came into effect on July 1, 1983 and provides guidance to Canadian government institutions on how the Canadian government should handle personal information about Canadian citizens. The main points of the Privacy Act are as follows (Privacy Act, Canada 2020):

- A government institution may only collect personal information if it is necessary for the performance of the institution's functions, and if not, no personal information may be retained (Privacy Act—Section 4, Canada 2020).
- An institution must inform the data owner about the purpose of the data collection; there are sometimes exceptions (Privacy Act—Section 5(2), Canada 2020).
- Personal data may only be used for the purpose for which it was collected. If a different use is intended, the permission of the data owner must be obtained (Privacy Act—Section 7, Canada 2020).
- Personal data must be treated confidentially and may not be disclosed without the permission of the data owner (Privacy Act—Section 8, Canada 2020).
- Every Canadian citizen and permanent resident of Canada has the right to access and, if necessary, correct information under the control of governmental institutions (Privacy Act—Section 12, Canada 2020).
- In addition, there is a Privacy Commissioner in Canada who deals with complaints from Canadian citizens regarding their data (Privacy Act—Section 29, Canada 2020).

3.4 Japan—Act on the Protection of Personal Information (APPI)

The Act on the Protection of Personal Information (APPI) came into force on April 1, 2005. On May 30, 2017, the APPI was again fundamentally revised and reformed. The reason for the reform was the desire to adapt the data protection guidelines to international standards. There are also a number of supplementary guidelines published by the Personal Information Protection Commission (PPC), which affect the general public or specific sectors (finance, medical, employment and telecommunications) (APPI, Japan 2016).

The APPI applies to anyone who controls and processes personal data, both private individuals and legal entities. In this context, the individuals or organizations must use the personal data for activities that serve the purpose of the company. It is not mandatory that the company or organization acts for profit. The areas press, politics, religion and publishing are partially excluded from the APPI (APPI, Japan 2016).

The PPC is the primary regulatory authority in connection with the APPI. It must manage and execute the following tasks (APPI, Japan 2016):

- The privacy policy and the individual's interest must be guaranteed when handling personal information and data.
- It provides information to foreign data protection authorities and, to a limited extent, may provide information for police investigations by foreign government agencies.
- It has the primary investigative, advisory and instructional authority under the APPI. This includes investigating the activities of PICs with an anonymized procedure for reviewing personal data (Sect. 4) and, if necessary, giving advice or instructions if the personal data is not handled in the manner that would be expected under the APPI.

The authority to initiate investigations in the context of data protection may, under certain circumstances, be delegated to competent ministers, with the exception of advisory and executive powers.

3.5 Summary and Brief Comparison

In summary, we have reviewed the privacy policies of different nations in this chapter. The guidelines of the European Union (GDPR), United States of America (CCPA), Canada (Privacy Act) and Japan (APPI) were selected, examined in detail and compared with each other. Special features were highlighted and presented.

In addition, a brief comparison of the different guidelines shall be carried out. The first point that stands out is that the Privacy Act from Canada already came into force in 1983. Other directives, such as the GDPR or the CCPA, are significantly younger or have been largely adapted recently, such as the APPI. Furthermore, there are differences in the jurisdiction of the guidelines, as they have different levels of rigor. For example, the Privacy Act protects the data of Canadian citizens, whereas the guidelines of the GDPR and CPPA are much further reaching.

In addition to these differences, there are also points that are very similar in the various directives. One example is the "right to be forgotten." Furthermore, companies and institutions must disclose the personal data collected about individuals. This data may have to be changed and adjusted if it is not correct. In some cases, there are still special regulations and case distinctions in individual guidelines. Due to the scope required for this, it is not possible to go into these special cases in full at this point.

The following chapter focuses on the impact of data protection on the business sector and technological progress. Particular attention is paid to the topics of data processing, for example, in relation to software development or the testing of software, as well as the transfer of data. Furthermore, limitations in connection with emerging technologies and retention of data are considered in relation to the topic "right to be forgotten."

4 Impact of Data Protection Regulations on Businesses and Technological Progress

The data protection regulations of governments and public institutions aim to protect people from having their data made available to unauthorized persons or having their data misused, rather than the protection of data itself (as the term would suggest). The regulations focus on protection against unlawful processing of personal data, promoting transparency in data processing ("Where does the data come from? Who uses it and for what purpose?") and, due to the sometimes very high penalties (as is the case for the European GDPR), they ensure conduct in accordance with the rule of law. The data protection laws help to enforce the principle of "informational self-determination" by authorizing the data owner to determine the data collector and the scope of the data processing.

Against this background, data protection regulations are of great importance to consumers and the protection of their personal data. Imagine what a world without data protection laws would look like, where everyone is exposed to arbitrariness (even by government authorities), cannot object to the misuse of their data and has no right to have incorrect data deleted. On the other hand, this necessary protection of consumers' data implies that companies are not allowed to utilize it in the most profitable way. This leads us to the challenges companies are facing with regard to processing data in compliance with the law. The following explanations always refer to the European GDPR, which was briefly presented in Sect. 3.1.

4.1 Challenges for Data Processing due to Purpose Restrictions

4.1.1 Software Development and Testing

As already mentioned, the stored personal data must be used for a specific purpose. The use of personal data for purposes the data owner is not aware of or has not expressly agreed to is not permitted (with the exception of legally based purposes). This has concrete consequences for the use of data for test purposes. For instance, if the proper functioning of a new database for customer relationship management (CRM) or a recruiting application is to be checked during development or test operation, it must not contain any data that can be assigned to specific persons. If data from real persons (i.e. from the real database) is used, this is considered improper use of data (the customers have not explicitly provided their data for testing purposes nor does a legally based purpose apply).

Test scenarios must be as realistic as possible to fulfill their purpose, for example, to determine whether data from one system arrives correctly in another system via a new interface. Sometimes, test databases with pseudo-entries are used for this purpose, but they are created with a lot of effort and time and usually do not meet the flexible requirements of the testers. Corresponding data fields such as e-mail addresses must still be recognizable as e-mail addresses in order for the business application to work without entering an error state. Certain information must also be linked to the data, for example, account numbers or customer identification numbers. It is possible to use real data for test purposes under very strict conditions. However, this requires, among other things, that the test data is deleted immediately afterward. The tests would then no longer be traceable, which would contradict recognized auditing principles. A different procedure should therefore be chosen, whereby the data can be modified in such a way that it prevents the original persons from being traced, but still remains suitable for the test procedure.

One possibility is the pseudonymization of data, i.e. personal data can only be assigned to the data subject by means of additional information, which is stored separately (see the definition of "pseudonymization" in Article 4(5) GDPR). However, the pseudonymization of data does not prevent this data from being considered personal data according to Article 4(1) GDPR. In this paragraph, personal data is defined as "any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier." Thus, even indirect identification (such as via a reference table in which a name is assigned to the ID of the first table) does not prevent it from being personal data subject to the GDPR. But pseudonymization reduces the risk of disclosure and misuse of personal data and is a common way to deal with the restrictions implied by the GDPR (THE EUROPEAN PARLIAMENT AND THE COUNCIL OF THE EUROPEAN UNION 2017).

4.1.2 Restrictions for Data Trading

The purpose-limited use of personal data in accordance with Article 6 of the GDPR poses a challenge for emerging technologies in the area of data trading. The data collected about a person and their behavior is, on the one hand, useful to the company collecting the data and, on the other, a valuable resource for other companies. This is because data in itself has a value which can be enhanced by combining it with information collected by third parties. It is an intangible asset that can be sold for a profit.

For example, it can be useful for a company to combine the data collected about its customers with information from third parties, such as social networks. In this way, the interests of the users can be analyzed in detail and utilized for target-group-specific offers. Or vice versa, by selling data about customers to third parties.

These business interests are clearly in opposition to the regulations of the GDPR. According to Article 6 of the GDPR, it must be proven that the data is processed lawfully. For this purpose, at least one of the conditions below must be complied with and proven in writing.

- "The data subject has given consent to the processing of his or her personal data for one or more specific purposes;
- processing is necessary for the performance of a contract to which the data subject is party or in order to take steps at the request of the data subject prior to entering into a contract;
- processing is necessary for compliance with a legal obligation to which the controller is subject;
- processing is necessary in order to protect the vital interests of the data subject or of another natural person;
- processing is necessary for the performance of a task carried out in the public interest or in the exercise of official authority vested in the controller;

- processing is necessary for the purposes of the legitimate interests pursued by the controller or by a third party, except where such interests are overridden by the interests or fundamental rights and freedoms of the data subject which require protection of personal data, in particular where the data subject is a child."

Article 7(1) adds: "Where processing is based on consent, the controller shall be able to demonstrate that the data subject has consented to processing of his or her personal data." Due to these harsh restrictions, data can only be traded if the person from whom the data originates has expressly consented. A company can hope for the naivety of customers giving their consent without being aware of the scope and consequences. Or perhaps it can lure customers with a promise that the data trading is in some way useful or offers the customer additional value (and the customer is convinced that this added value is worth their consent!). However, no matter by what means or under what conditions this consent is obtained, it is always required. This makes data trading considerably more difficult, which limits or in some cases completely excludes certain business models based around Big Data.

4.2 Restrictions for Using Artificial Intelligence (AI) and Algorithm-Based Decision-Making

The progress systems using algorithm-based decision-making have made in recent years is enormous. This is mainly due to the sharp drop in the cost of computing power. However, it is still not predictable whether and when a complete digitalization of human thinking (so-called "strong artificial intelligence") will be possible. Nevertheless, so-called "weak artificial intelligence," i.e. the mapping of individual aspects of human intelligence, has made great progress. This includes, for example, knowledge databases which collect and store information but independently link and interpret information. It also includes the recognition, analysis and prediction of action patterns expressed, for example, in automated recommendations ("Other customers have also expressed interest in the following articles: …").

However, data protection regulations clearly limit the possibilities for using artificial intelligence for decision-making. Article 22(1) requires decisions not to be made based solely on automated processing, including profiling. There are some exceptions to this general requirement, such as the explicit consent of the data subject or the fact that automated decision-making is necessary for the conclusion or performance of a contract.

This requirement is accompanied by the provisions of Article 15, which give a data subject the right to request information about the processing of their personal data. The right to information expressly includes information about the automated decision-making processes, i.e. an explanation of how the algorithm comes to the result. In doing so, the company must provide information about "the logic involved, as well as the significance and the envisaged consequences of such processing for the data subject." Of particular importance here is the fact that the right of access refers to the processing logic of the data. But this is exactly what the black box is, because in most cases it is not at all comprehensible how the algorithm came to a certain decision.

It is obvious that the legislator wants to protect data subjects from becoming victims of unexplainable decisions made by a machine. As long as the decision-making process or the decisive factors influencing it cannot be understood, the right of information according to Article 15 GDPR remains an uncertainty factor that cannot be remedied.

4.3 Impacts of the Storage Limitations and the "Right to be Forgotten"

At the beginning of the computer age, data storage was expensive, which resulted in the storage of only necessary data. Today, data can be stored almost indefinitely and retrieved at any time. In order to forecast further or track development, it is in the interest of companies to store data for as long as possible. But the GDPR puts strong limits on the almost unlimited storage possibilities.

According to Article 5(1) GDPR, data storage is permitted "for no longer than is necessary for the purposes for which the personal data is processed." There are a few legal exceptions, such as the public interest or data storage for scientific and historical research purposes. In all other cases, the principle of the GDPR strictly regulates the duration of data storage. This requirement is added by the provision in Article 13(2) GDPR: "The controller shall, at the time when personal data is obtained, provide the data subject with the following further information necessary to ensure fair and transparent processing: (a) the period for which the personal data will be stored, or if that is not possible, the criteria used to determine that period…".

Limiting the time during which data may be stored (and then must be deleted!) is a great challenge. Every piece of information has a certain lifespan. This means, for example, that there is no longer any need to store it once the customer business has been fulfilled. However, there are still legal storage periods for invoices, letters or other business documents. In addition, data on business transactions that have already been successfully concluded with the customer can be useful for future business negotiations. So where do you set the limit? The requirement of limiting data storage raises numerous questions during practical implementation, such as: Which criteria are suitable to determine the point in time when the purpose of the data storage is fulfilled, and it must therefore be deleted? How to proceed with data backups stored geographically widespread in other countries or continents? How can the data to be deleted be identified if it has become part of risk models (e.g. data on the payment behavior of customer groups used for flat-rate credit assessments)? In order to answer these questions, an acceptable balance must be struck between being compliant with the law and the advantage of utilizing personal data during their lifecycle within the company. This can be perceived as an opportunity for new systems and models providing features by default.

A related requirement concerns the right of a data subject to be forgotten, which is enshrined in Article 17 of the GDPR: "The data subject shall have the right to obtain from the controller the erasure of personal data concerning him or her without undue delay and the controller shall have the obligation to erase personal data without undue delay where one of the following grounds applies: …". Probably the most common reasons for immediate deletion of data include points b) and c) of Article 17(1) GDPR, which specify the following:

"(b) the data subject withdraws consent on which the processing is based according to point (a) of Article 6(1), or point (a) of Article 9(2), and where there is no other legal ground for the processing;

(c) the data subject objects to the processing pursuant to Article 21(1) and there are no overriding legitimate grounds for the processing, or the data subject objects to the processing pursuant to Article 21(2)."

A simple written declaration is therefore sufficient to object to the data's further processing or to demand the deletion of data.

The challenges in practical implementation also apply to the "right to be forgotten," e.g.: How can personal data be removed from backups? Can legally sound reasons be given that allow further, temporary storage of data in backups, e.g. until the backup cycle has expired? How to deal with data that has become part of risk models?

Digitalization must provide legally compliant answers to these questions and the processes must take the requirements of the GDPR into account.

One way to take these requirements into account is to use the concept of "privacy by design," which data processing companies are also required to do under Article 25(1) GDPR. This concept comprises seven principles which aim to ensure that data protection regulations are observed in the best possible way as early as the hardware and software development phase.

The first three principles should be mentioned by way of example, as they provide a good impression of the objective of Privacy by Design (Cavoukian 2011):

- **"Principle 1—Proactive not Reactive; Preventative not Remedial**: The Privacy by Design approach is characterized by proactive rather than reactive measures. It anticipates and prevents privacy invasive events before they happen. Privacy by Design does not wait for privacy risks to materialize, nor does it offer remedies for resolving privacy infractions once they have occurred – it aims to prevent them from occurring. In short, Privacy by Design comes before-the-fact, not after.
- **Principle 2—Privacy as the Default Setting**: We can all be certain of one thing — the default rules! Privacy by Design seeks to deliver the maximum degree of privacy by ensuring that personal data are automatically protected in any given IT system or business practice. If an individual does nothing, their privacy still remains intact. No action is required on the part of the individual to protect their privacy—it is built into the system, by default.
- **Principle 3—Privacy Embedded into Design**: Privacy by Design is embedded into the design and architecture of IT systems and business practices. It is not bolted on as an add-on, after the fact. The result is that privacy becomes an essential component of the core functionality being delivered. Privacy is integral to the system, without diminishing functionality."

The consideration of the seven principles of Privacy of Design causes additional effort in software development, which of course affects the costs. Whether the consideration of these principles in software development impedes technological progress can be discussed controversially. In any case, these principles restrict companies in the collection of user information and thus also prevent evaluations of this information. But such clear guidelines are in the interest of citizens, who have a right to and need protection of their informational self-determination.

5 Summary and Outlook

In general, data protection regulationsaim, on the one hand, to protect the informal self-determination of individuals by applying regulations to the collection, processing or storing of personal data and, on the other, to inform individuals about their rights in order to enable them to make informed decisions on sharing personal data. The European Union, for example, stated in a review of the effects of the GDPR in 2020 that the awareness of individuals in the European Union regarding data protection rights has increased since the release of the GDPR (European Commission 2020, p. 8).

For companies, data protection regulations offer stringent guidelines on dealing with personal data. These guidelines entail changes to the business models of companies and require assessments on the content of data processing, including utilizing emerging technologies to comply with the applicable data regulation. For emerging technologies like Big Data or artificial intelligence, companies need to invest in manpower and resources in order to adjust procedures, processes and algorithms collecting, processing and storing personal data. These adjustments result in an increase in labor costs for monitoring algorithms or adjusting existing solutions (Li et al. 2019, p. 2ff).

The impact of these implications must be monitored in order to determine whether companies with a business model relying on emerging technologies and companies situated in nations with strict data protection regulations face major disadvantages over competitors where data protection implications do not apply.

In Singapore, for example, a government subsidy program enables banks to gain full insight into their customers' financial situation, and the central availability of data enables banks to tap into a previously under-served market. The data is collected via the banks' mobile applications and requires the annual approval of the customers. Due to the stricter data protection laws in the European Union, a complete analysis of the customer data by banks so that they can offer additional business is only possible with the customer's consent to the individual activity. Although principal banks manage and process a customer's account movements, they have no insight into the reasons for a transaction or the sender and recipient. For this reason, European banks see companies entering the market from customer-oriented branches of the value chain as serious competition. Through the data generated from non-banking-related business transactions with customers, these companies have a more detailed view of their customers (Hein et al. 2020, p. 23). The GDPR and new market entries are forcing European banks to rethink their business models to gain insight into the available customer data and beyond through new offers. This development shows that not only companies from the technology sector are affected by the regulations, but also companies that want to integrate these technologies into their business models.

For companies servicing markets where different data protection standards apply, the question arises as to which standards a service will be tailored to or whether it is wise to no longer provide a service in specific markets.

In addition to these implications, for companies with data-protection-compliant solutions, the current market situation poses a competitive advantage as individuals and companies relying on the provision of services are willing to spend more on services or solutions complying with data protection standards (Capgemini Research Institute 2018, p. 22).

Literature

APPI, Japan. 2016. *Act on the Protection of Personal Information*, December 21. Accessed August 24, 2021. http://www.japaneselawtranslation.go.jp/law/detail/?id=2781&vm=2&re=02.

Capgemini Research Institute. 2018. "Seizing the GDPR Advantage: From Mandate to High-Value Opportunity." *capgemini.com*, May 17. Accessed August 23, 2021. https://www.capgemini.com/wp-content/uploads/2018/05/GDPR-Report_Digital.pdf.

Cavoukian, Ann. 2011. "Privacy by Design: The 7 Foundational Principles—Implementation and Mapping of Fair Information Practices." *iapp.org*, January 1. Accessed August 23, 2021. https://iapp.org/media/pdf/resource_center/pbd_implement_7found_principles.pdf.

CCPA Section 1798.100, State of California. 2020. "California Consumer Privacy Act—Section 1798.100." In *California Consumer Privacy Act*, edited by State of California CCPA, Section 1798.100. California.

CCPA Section 1798.105, State of California. 2020. "California Consumer Privacy Act—Section 1798.105." In *California Consumer Privacy Act*, edited by State of California CCPA, Section 1798.105. California.

CCPA Section 1798.110, State of California. 2020. "California Consumer Privacy Act—Section 1798.110." In *California Consumer Privacy Act*, edited by State of California CCPA, Section 1798.110. California.

CCPA Section 1798.115, State of California. 2020. "California Consumer Privacy Act—Section 1798.115." In *California Consumer Privacy Act*, edited by State of California CCPA, Section 1798.115. California.

CCPA Section 1798.120, State of California. 2020. "California Consumer Privacy Act—Section 1798.120." In *California Consumer Privacy Act*, edited by State of California CCPA, Section 1798.120. California.

CCPA, State of California. 2020. *California Consumer Privacy Act.* California, January 1.

Dachwitz, Ingo. 2020. "Netzpolitik.org." *Abschlussbericht der Datenschutzbehörde: Nein, der Cambridge-Analytica-Skandal fällt nicht in sich zusammen*, October 23. Accessed August 23, 2021. https://netzpolitik.org/2020/abschlussbericht-der-datenschutzbehoerde-nein-der-cambridge-analytica-skandal-faellt-nicht-in-sich-zusammen/.

European Commission. 2020. *Data Protection as a Pillar of Citizens' Empowerment and the EU's Approach to the Digital Transformation.* Brussels, June 24.

GDPR Section 5, General Data Protection Regulation. 2016. "General Data Protection Regulation—Section 5." European Union, May 4. Section 5.

GDPR, General Data Protection Regulation. 2016. *General Data Protection Regulation.* European Union, April 27.

Hein, Christoph, Hanno Mußler, and Franz Nestler. 2020. "Der gläserne Bankkunde." *Frankfurter Allgemeine Zeitung*, December 9, 2020: 23.

Information Commissioner's Office. 2020. "ICO Investigation Into Use of Personal Information and political Influence." *ico.org.uk.* October 2. Accessed August 24, 2021. https://ico.org.uk/media/action-weve-taken/2618383/20201002_ico-o-ed-l-rtl-0181_to-julian-knight-mp.pdf.

Jia, Jian, Ginger Zhe Jin, and Liad Wagman. 2018. "The Short-Run Effects of GDPR on Technology Venture Investment." *NBER Working Paper Series No. 25248*, November: 1–43.

Li, He, Lu Yu, and Wu He. 2019. "The Impact of GDPR on Global Technology Development." *Journal of Global Information Technology Management* 22 (1), January 24: 1–6.

Politou, Eugenia, Efthimios Alepis, and Constantinos Patsakis. 2018. "Forgetting Personal Data and Revoking Consent Under the GPDR: Challenges and Proposed Solutions." *Journal of Cybersecurity*, March 26: 1–20.

Privacy Act—Section 12, Canada. 2020. "Canada Consolidation Privacy Act." In *Canada Consolidation Privacy Act*, edited by Privacy Act Canada, Section 12. Canada.

Privacy Act—Section 29, Canada. 2020. "Canada Consolidation Privacy Act." In *Canada Consolidation Privacy Act*, edited by Privacy Act Canada, Section 29. Canada.

Privacy Act—Section 4, Canada. 2020. "Canada Consolidation Privacy Act." In *Canada Consolidation Privacy Act*, edited by Canada Privacy Act, Section 4. Canada.

Privacy Act—Section 5(2), Canada. 2020. "Canada Consolidation Privacy Act." In *Canada Consolidation Privacy Act*, edited by Canada Privacy Act, Section 5(2). Canada.

Privacy Act—Section 7, Canada. 2020. "Canada Consolidation Privacy Act." In *Canada Consolidation Privacy Act*, edited by Privacy Act Canada, Section 7. Canada.

Privacy Act—Section 8, Canada. 2020. "Canada Consolidation Privacy Act." In *Canada Consolidation Privacy Act*, edited by Canada Privacy Act, Section 8. Canada.

Privacy Act, Canada. 2020. *Canada Consolidation Privacy Act*, November 17.

Rainie, Lee. 2018. *Americans' Complicated Feelings About Social Media in an Era of Privacy Concerns*, March 27. Accessed August 23, 2021. https://www.pewresearch.org/fact-tank/2018/03/27/americans-complicated-feelings-about-social-media-in-an-era-of-privacy-concerns/.

Sackmann, Christoph. 2018. *Die Tücken der Intelligenz: Künstliche Intelligenz erachtet Bewerbungen von Frauen als minderwertig - Amazon muss reagieren*, December 12. Accessed August 23, 2021. https://www.focus.de/finanzen/boerse/die-tuecken-der-intelligenz-amazon-schaltet-ki-ab-die-bewerbungen-von-frauen-als-minderwertig-erachtete_id_9741890.html.

THE EUROPEAN PARLIAMENT AND THE COUNCIL OF THE EUROPEAN UNION. 2017. "2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 27 April 2016 on the Protection of Natural Persons with Regard to the Processing of Personal Data and on the Free Movement of Such Data, and Repealing Directive 95/46/EC." *Official Journal of the European Union*, April 27.

Wachter, Sandra. 2018. "Normative Challenges of Identification in the Internet of Things: Privacy, Profiling, Discrimination, and the GDPR." *Computer Law & Security Review* 34 (3), June 16: 436–449.

Distributed Ledger

Distributed ledger is an impactful and multifaceted subject in the financial industry and in the ongoing digital transformation. It includes payment (like Bitcoin), stable coins[1] (like Diem[2]) and tokenization in general, to name just a few of the popular aspects behind distributed ledger.

Another (often) underestimated and undervalued aspect in the general data context and digital transformation is self-sovereign identity. Data is seen as a valuable and unprotected good that Big Tech can use at their discretion. Data protection regulations (e.g. in Europe, see subsection 3.1 in (Czwalina et al. 2021) have started to protect personal data and have already impacted certain business models where the regulations are applied.[3]

In a first step, the chapter (Kurfels et al. 2021) identifies the importance of verifiable credentials in digital identification, in (digital) business models and in digital transformation in general. The chapter then explores the idea of self-sovereign digital identities (SSI) and introduces a technical blueprint to implement such an SSI using distributed ledger technology (Hyperledger Indy[4]).

[1] Stable coins are offers “backed” by real assets, or the binding promise to exchange the provided token back into a real currency of asset. They are cryptocurrencies whose price can be controlled by active or automatic monetary policy with the aim of low volatility in relation to a national currency, a basket of currencies or other assets.

[2] Formerly known as Libra, the stable coin of Facebook.

[3] China does not have any initiative to protect clients’ data and establishes an unlevel playing field against regions with a data protection regulation.

[4] See (The Linux Foundation, Hyperledger Indy 2020).

Literature

Czwalina, Marie Kristin, Matthias Kurfels, and Stefan Strube. 2021. "Data Protection Regulation." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Kurfels, Matthias , Heinrich Krebs, and Fabian Bruse. 2021. "Digital Identity Management—For Humans Only?" In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

The Linux Foundation. 2020. *Hyperledger Indy.* Accessed December 15, 2020. https://www.hyperledger.org/use/hyperledger-indy.

Digital Identity Management—For Humans Only?

Matthias Kurfels, Heinrich Krebs, and Fabian Bruse

1 Theoretical Basis of Digital Identity Management

Before the practical application of blockchain-based identity management can be described, it is necessary to understand the theoretical basics. These point out why identification is crucially important and in which ways identification beyond doubt can be ensured in daily life but especially on today's Internet.

1.1 Why Is Identification so Important?

Since the second human being appeared on Earth, it has always been important in social interactions to know who the one you want to get in contact with really is. Why is this in the nature of man? Unknown persons can appear

M. Kurfels (✉) · H. Krebs · F. Bruse
ifb SE, Grünwald, Germany
e-mail: Matthias.Kurfels@ifb-group.com

H. Krebs
e-mail: Heinrich.Krebs@ifb-group.com

F. Bruse
e-mail: Fabian.Bruse@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_11

threatening and create fear, because you cannot evaluate whether any dangers emanate from them. In contrast, acquaintances usually convey a feeling of security, because you can fall back on previous experiences (whether positive or negative) to the current situation and adjust your behavior accordingly. The longer and deeper a positive relationship with a person becomes, the more likely it is that a close friendship can develop.

Identity is often linked to a name or a designation. However, this is not limited to people. Children often give names to their dolls, teddy bears and pets. Cars have a chassis number, which is of considerable importance not only for customer service but also for possible identification after a theft. Every file stored on a computer has a name, every server has a unique name on the network and almost every technical device connected to the Internet has a globally unique MAC address. All this helps to identify or uniquely determine a person, animal, object or other thing.

1.2 How to Identify a Person Beyond Doubt

In order to establish the identity of a person or an object beyond doubt, as many precisely defined characteristics as possible are required by which a person or an object can be recognized. These characteristics are also the "source of trust". For instance:

A long-time business partner you would like to meet makes you an offer: "I will send a young, reliable assistant to the airport to pick you up for our appointment". But you don't know this young man personally. How can you be sure you are being picked up by the right person? Your business partner tells you certain external characteristics, such as that he is blond, that he will be wearing a light brown jacket, a dark blue shirt and a yellow tie. He will also have a black umbrella and a thin light brown briefcase. Besides, you already know that it must be a young man. There will probably be several blond men with light brown jackets at the airport. However, the more specific characteristics you know and the more unusual these characteristics are, the more likely you will be able to identify your ride in the crowd without any doubt.

In addition to the procedure described in the example, there are other methods by which a person can be identified beyond doubt, whereby an identification is always based on a specific "source of trust" (see Fig. 1):

- You usually recognize your friends, neighbors or colleagues by their face (sometimes by their voice before you can see them). The source of your trust is constant social interaction with them. You meet your colleagues

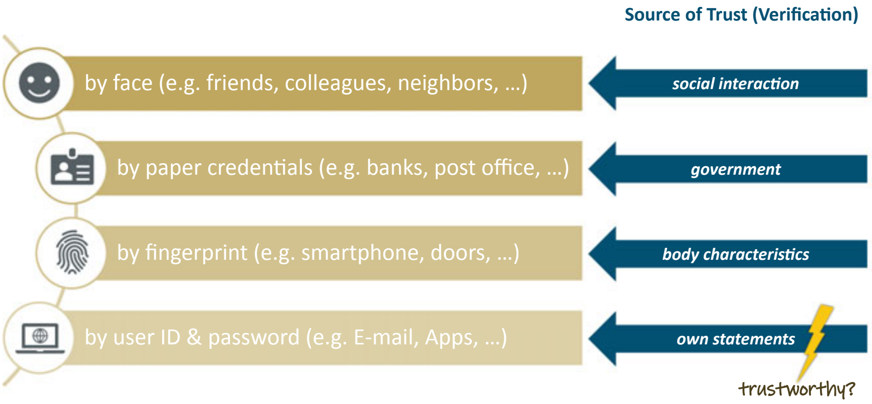

Source of Trust (Verification)
by face (e.g. friends, colleagues, neighbors, ...)
social interaction
by paper credentials (e.g. banks, post office, ...)
government
by fingerprint (e.g. smartphone, doors, ...)
body characteristics
by user ID & password (e.g. E-mail, Apps, ...)
own statements
trustworthy?

Fig. 1 Common methods of identification (© ifb SE)

every day at the office, do something with your friends almost every weekend or see your neighbors on the street or in the elevator several times a month.

- Another form of unequivocal identification of a person is to identify them by means of an official document. You have probably never seen an identity card from friends, neighbors or colleagues but you trust that it is the person who introduced themselves to you by name or whose name is on the office or apartment door. In business, however, it is often not enough for a previously unknown person to give their name. Opening a bank account or renting a car requires official documents such as an identity card or driver's license, which are legally binding. In these cases, the source of trust is the government or the authority issuing the document. Due to the high degree of protection against forgery of the documents and their security features, the recipient can be confident the person presenting the document is also its legitimate holder.
- Fingerprints or an iris scan of the eye are also well-known means of identifying a person. These are unchangeable biometric features, and therefore provide a high level of trust and security.

These three procedures described have something in common: identification always takes place in connection with the personal presence of the person to be identified. However, this is not possible with identification on the Internet. What then? A common procedure is identification by a user ID and a valid password. However, the "source of trust" is extremely doubtful, because identification is only based on the information a person has given about themselves. So how can you be sure that this information is correct?

The solution is based on the confirmation by a reliable third party, which creates an additional level of trust.

1.3 Verifiable Credentials as an Additional Level of Trust

The model of verifiable credentials (see Fig. 2) is based on the assumption that two parties who do not know each other (and therefore do not trust each other) trust a third party who is proven to be reliable. But why does the proof of trustworthiness only succeed via a third party? Why is this proof not possible for the two parties originally involved?

The key to understanding lies in the combination of three essential elements:

- **The construction of the credential**. It is created by a third party with complex cryptographic procedures that cannot simply be imitated.
- **The availability of the credential**. It is stored in a personal digital wallet and can only be accessed by the credential holder, no one else.
- **The verification procedure**. The credential can only be confirmed if the verifying party has read access to a database of the credential issuer and knows the algorithms used to prove the credential.

The interaction of all three components creates a basis of trust that would not be possible without the cooperation of the third party (the authorization issuer). However, as with all cryptographic procedures, the following key attributes must also be given:

- The issuer of the credential is trustworthy.

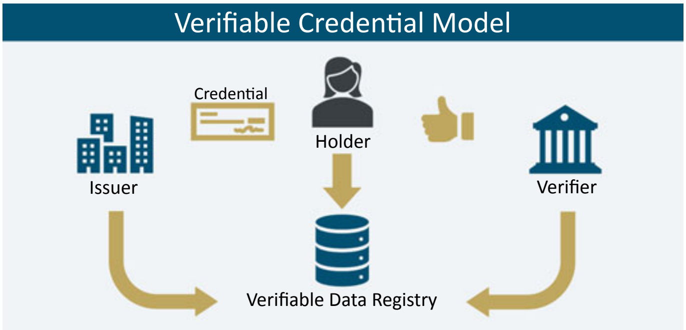


Fig. 2 Verifiable credential model (© ifb SE)

- The credential was issued to the entity presenting it.
- The claims were not tempered with.
- The credential has not been revoked.

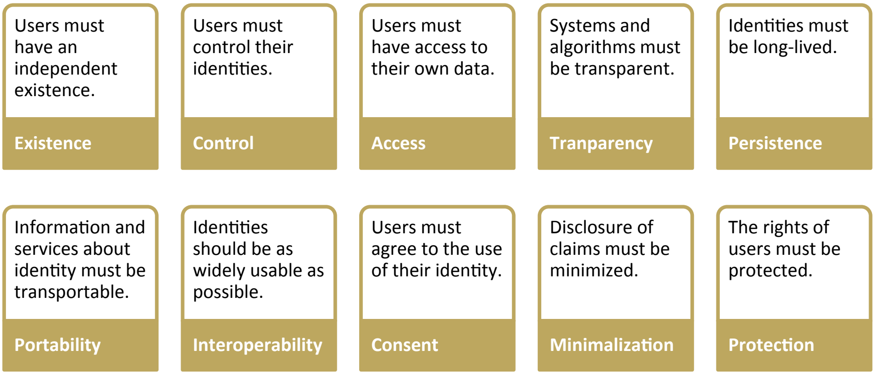
The system described above allows the holder to use an authorization without informing the issuer, but is still dependent on the issuer: first when the credential is issued and later when it is maintained. What happens if an issuer excludes certain holders or groups of holders from receiving an authorization for political or other reasons? What if an issuer disappears from the market for any reason? What if dictatorial regimes exert pressure on the issuer to revoke existing authorizations without reason? In all these cases, the holder of the credential is powerless if they lose it. A solution to this problem is using a self-sovereign digital identity.

1.4 Self-Sovereign Digital Identity (SSI)

The central idea behind using SSI is that the owner of the digital identity manages their own data and can determine when and how it is made available to others. This involves a change of perspective, as the German Blockchain Association wrote: "Unlike existing identity solutions that are structured from the perspective of the organization that provided an identifier, self-sovereign identities are structurally set out to work from the perspective of the individual or entity that is the subject of a given identifier" (German Blockchain Association 2018).

However, this change of perspective does not mean that—unlike with the procedure described above for a user ID with associated password—anyone can simply claim something about themselves without others being able to verify it. SSI rather means that other people or organizations can verify the claims someone makes about their identity with cryptographic security.

In the context of SSI, an additional confirming authority is only required at the time the credential is created. The confirmation takes place once and does not need to be re-issued every time the digital identity is used. This advantage results from the fact that the identity confirmation is stored as an unchangeable part of the digital identity. This allows the holder of the digital identity to use it for identification purposes even if the confirming entity no longer exists or is offline.



Users must have an independent existence. Existence

Users must control their identities. Control

Users must have access to their own data. Access

Systems and algorithms must be transparent. Tranparency

Identities must be long-lived. Persistence

Information and services about identity must be transportable. Portability

Identities should be as widely usable as possible. Interoperability

Users must agree to the use of their identity. Consent

Disclosure of claims must be minimized. Minimalization

The rights of users must be protected. Protection

Fig. 3 Ten principles of self-sovereign identity (© ifb SE)

Christopher Allen, the vice chairman of the W3C Credentials CG,[1] which is working on standards for decentralized identity, formulated in 2016 ten principles (Allen 2016) by which self-sovereign identity will be shaped (see Fig. 3):

2 Technical Implementation of SSI

To dive into the technical aspects of the current state of SSI technology and philosophy, we will discuss the DID (decentralized identifier) and the DID document in general before discussing different implementation approaches.

Over the course of this chapter we will also briefly discuss some key concepts of cryptography as needed to understand its usage in the SSI context. We will begin with "public key cryptography", a concept that is based on mathematical algorithms that create a pair of "keys" from a large random number. These keys are numbers themselves, but we think of them as chunks of data. One of the keys from the pair is the public key, the other one the private key. Using special mathematical algorithms, you can use the keys to encrypt and decrypt any other data. This concept is also called "asymmetrical cryptography" because data encrypted with the public key can only be decrypted by the private key from the same pair, and vice versa. The basic case (see Fig. 4) would be that Person B (Bob) uses Person A's (Alice's) public key to encrypt something only Alice can decrypt with her private key. Not

[1] W3C Credentials CG is a part of the World Wide Web Consortium (W3C), the main international standards organization for the World Wide Web.

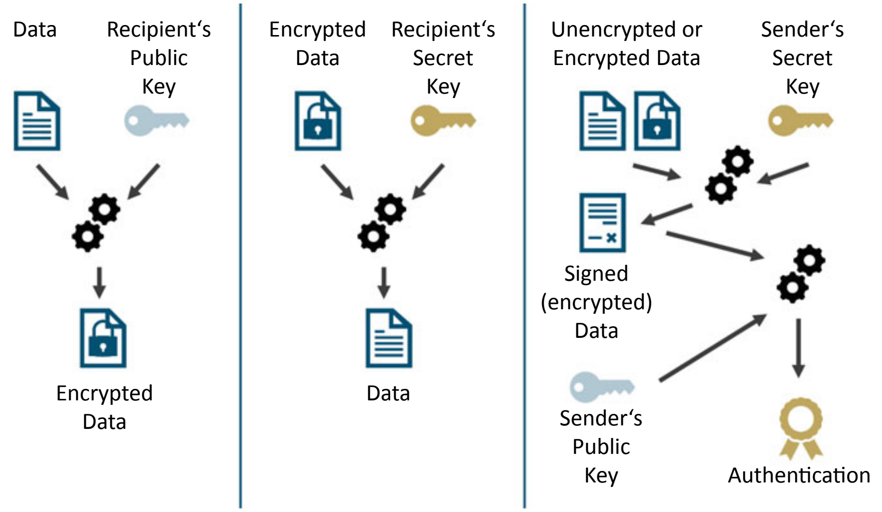


Fig. 4 Crypto standard functionality (© ifb SE)

even Bob could reverse the encryption. However, Alice can encrypt something with her private key, and anyone could decrypt it with Alice's public key. This process is called "signing" and is a way for Alice to endorse or authenticate the data. Signing is the basis for a lot of applications we will discuss.

With these basics in mind, we dive into SSI concepts. Current design approaches for digital identity systems have defined four layers in their model, which need standardization, each one building on the functionality the layer before provides. These layers are, from most fundamental to most advanced:

1. Decentralized Identifiers (DID)
2. Decentralized Key Management Systems (DKMS)
3. DID Auth
4. Verifiable Credentials.

We will discuss these and go into detail of what the proposed standards shall accomplish.

2.1 Decentralized Identifiers

To address and describe digital identities, a specialized naming system needs to be implemented. "Decentralized identifiers" (DIDs) explicitly point to one digital identity, specifically the "DID document" describing said identity.

On July 31, 2020, the World Wide Web Consortium (W3C) published their draft on core architecture, data model and representations of DIDs.[2]

Based on that draft, a decentralized identifier would be a string that looks like this:

```
did:example:123456789abcdefghi
```

It consists of three parts, separated by colons. First the "URL[3] identifier", in this case "did". Second is the "DID method", in this case "example". Third and finally is a "DID-method-specific identifier", here, as an example, an increasing series of numbers and letters. In this format, the DID adheres to the URL naming conventions and is hierarchical. This means, that the "URL identifier" dictates that a "DID method" has to follow, and the DID method then dictates how the "DID-method-specific identifier" has to look.

The first piece of information, "DID", tells any system that it is dealing with a decentralized identifier and said system should process the string, that is the identifier, accordingly. Just like an Internet browser processes the "http:", "https:" or "mailto:" identifiers it encounters in URL-style hyperlinks.

The second part of the DID names the method. The DID method is a specific set of rules on how to process the DID. The keyword "example" is not a working DID method, though. As of June 18, 2020, the World Wide Web Consortium (W3C) published a draft for a register of DID methods.[4] There, you can find methods like "did:web", "did:peer" or "did:sov". Each of these methods is linked to a document openly defining how to handle the DID in question. These methods must adhere to certain minimum requirement rules defined in the W3C document mentioned above. For example, all DID methods must, as one requirement, specify how to handle CRUD operations. That is **c**reate, **r**ead, **u**pdate and **d**elete/deactivate a DID within their method. In theory, this should allow programmers to make their applications work with DIDs that were originally created in one of these methods, providing compatibility between different systems using DIDs and thereby forming a global identity infrastructure.

The third part of the DID is a unique ID derived by some algorithm described in the document that details the DID method. Each method has

[2] Permanent Link: https://www.w3.org/TR/2020/WD-did-core-20200731/.

[3] Uniform Resource Locater (as defined by RFC3986).

[4] Permanent Link: https://www.w3.org/TR/2020/NOTE-did-spec-registries-20200618/.

its own approach on how to handle this part of the DID. It is, however, a requirement that this makes the DID unique.

As mentioned above, the DID alone is not an identity, yet. It is a URL, so it provides a location of something: the DID document. The DID document describes the digital identity, most specifically the cryptographic keys to identify the identity. But it can go further than that and define special rules on what cryptographic keys are allowed to update the identity, what services the DID is allowed to be used with and what the service endpoints are that are working on behalf of the DID.

As such, the DID document is the centerpiece of the digital identity, while the DID itself is its address. The digital identity, however, is not necessarily limited to the digital representation of a human subject. The DID subject could be a person, organization, group, physical thing, logical thing, digital thing, such as an automated process, etc.

One of the methods currently listed in the aforementioned register is "did:sov". This method makes use of the Sovrin distributed ledger and the Sovrin Foundation infrastructure, which is specifically designed and used for self-sovereign identity purposes. The definition is publicly available at its GitHub webpage.[5] Within the method definition it is described how a decentralized identity must be defined and registered within the "did:sov" method. The DID document and information tied to it are stored on a distributed ledger. Said distributed ledger is implemented as a blockchain making the DID documents widely available and tamper-proof.

Another "DID method", for example, is "did:peer". This method's approach is to keep all the data related to a digital identity on the user's own devices. No information needs to be made public on a distributed ledger, unless one would want a digital identity to build up a public reputation. This might be essential for organizations that need to authenticate identities of others.

2.2 Decentralized Key Management System (DKMS)

On the Internet, when a secured page is opened in a web browser the server provides a key with which data is to be encrypted. But an attacker could impersonate the server and send its own key and establish an encrypted and seemingly secure connection. To prevent this there is a public key

[5] Link: https://sovrin-foundation.github.io/sovrin/spec/did-method-spec-template.html.

infrastructure (PKI) in place, meaning there are organizations (companies, non-profits, governments, etc.) that issue a certificate that a public key belongs to whomever (or whatever domain name) it says it does. The keys with which these authorities sign the certificates are usually rolled out with a device's operating systems or browser software. They are called certificate authorities (CA). The issue of trust is therefore circumvented, if you trust that your device's library of CA certificates is not compromised. Usually your device does imply this.

A different method is a "web of trust". Here you only explicitly trust keys you personally vouch for. Keys that are vouched for by someone you explicitly trust, you trust implicitly. Keys that are explicitly vouched for by keys you implicitly trust, you also implicitly trust. And so on. To document to others that you vouch for someone's public key means you sign the key with your secret key and publish the signed key. Such a system is, for example, used in the OpenPGP[6] environment.

With the possibility that everything can have a digital identity representation, relationships can, and should, be defined. For example, you can create an identity for yourself, defining a cryptographic key pair as a means for other systems to verify your identity. But what if these keys are compromised? You better generate another key pair or even multiple key pairs and store them somewhere safe. The DID document then gets a flag that only the majority of these key pairs are allowed to update the DID document, so you can be fairly certain that a single compromised key pair wouldn't allow someone to take over the digital identity by issuing and signing an update to the DID document with the compromised key.

You can even get digital identities for the devices you own, so your cell phone, laptop and desktop computer could all have their own identities as well as cloud-based services. You can grant each of these digital identities a different level of authority to manage their own personal digital identity, which means that the cloud-based service might automatically identify as the personal digital identity to, for example, receive large files and store them in an encrypted data vault, but it may not identify with the personal digital identity to request the opening of a bank account in your name. Meanwhile, your cell phone is allowed to handle all claim verification requests, which means that the pressing of a button in a mobile app is enough to provide proof that the subject of the digital identity is over 21 years old to a dance club's digital identity requesting this information.

[6] Open Pretty Good Privacy—an encryption standard often used for e-mail end-to-end encryption.

To manage all these keys and DIDs associated with them, software tools are used. These are called "wallets" because they share the functions of real-life wallets. The decentralized key management system (DKMS)[7] is an emerging standard for storing keys, DIDs and other related information and for procedures on how to recover keys or access to DID documents if they are lost or compromised. This standard is currently being refined by the non-profit OASIS community.

For recovery of lost keys, two methods are featured. One, where the key to recover the identity is in cold storage, for example, on a USB drive in a bank vault. The other is to split the key into "shards" and send these, encrypted, to "trustees". These are known and trusted people who then, in concert, can allow me to rebuild my secret key. This is called "social recovery".

Without the need to have just one public identity, it is possible to have one for each relationship toward another identity. Today, social login is widespread on the Internet, where identity providers (IdP) are used to identify you to a service. This, however, means all our user accounts on different web services are correlatable. Using separate digital identities specifically for each of your required web services minimizes this correlation up to making it untraceable. If consistently conducted, one will quickly have more than a thousand DIDs in use. This is another reason to standardize wallet apps and allow digital identities to move from wallet to wallet and device to device, as necessary. Standards would allow the automation of all required procedures, like updating the DID documents with new endpoints and new DIDs to the devices that now host the wallet, as well as invalidate old ones.

Within the proposed DKMS the idea is that, above the DID layer of the system, there are also cloud agents and edge devices available. Edge devices represent the devices that are operated "on the edge" of the network, meaning with direct contact to the entity that the DID represents, for example, a smartphone and its wallet app being the edge device for somebody and their DID. But since the smartphone is not online all the time it makes sense to have a cloud agent where automated services act on a person's behalf 24/7. This is often described as being similar to mail servers accepting emails and storing them until an email program (the edge device) collects them. Communication between cloud agents and edge devices is also part of the proposed standard, meaning that all services can operate with each other, allowing users to use the implementations they trust most.

[7] https://github.com/hyperledger/aries-rfcs/blob/master/concepts/0051-dkms/dkms-v4.md.

2.3 DID Auth

While your Internet browser uses PKI to authenticate a server you connect with, it is rarely the case that you use a similar approach to authenticate yourself to the server. And while such implementations exist, authentication with a cryptographic key is more often used for Secure Shell or VPN services. For a digital identity infrastructure with lots of possible connections between devices that are not part of the PKI infrastructure (which is mainly used for public web and email servers) there needs to be a standard in place that allows for DIDs and their respective keys to authenticate any device they open a connection with.

The Internet Engineering Task Force (IETF) and the Decentralized Identity Foundation (DIF) are working on ways for standard implementations of an authentication protocol based on DIDs. The DIF's reference implementation uses Javascript Object Signing and Encryption (JOSE) to show what an authentication process can look like. It is available on GitHub[8] under the Apache License 2.0 and is based on OpenID Connect (OIDC), which itself is based on the OAuth 2.0 framework.

2.4 Verifiable Credentials

This really is the core element of digital identity. The three layers we discussed above are the foundation on which one digital identity can safely be managed and connect to another digital identity. But for now, the identities are just the information collected in the DID document, and they only govern how the identity is administrated and communicated with. The DID document does not contain any information about who the digital identity represents, for example, the name of a person. That's where credentials come into play.

The W3C is also working on a standard way to issue claims, verify them and thereby create verifiable credentials. Drafts are available for the verifiable credentials data model,[9] implementation guide[10] and possible use cases.[11] The basic concept describes four roles in handling verifiable credentials:

The **Issuer** is the entity who creates a "claim" and ties it to a "subject". The **Verifier** is the entity who verifies the claim about the subject. The **Subject**

[8] https://github.com/decentralized-identity/did-auth-jose/.

[9] https://www.w3.org/TR/vc-data-model/.

[10] https://www.w3.org/TR/vc-imp-guide/.

[11] https://www.w3.org/TR/vc-use-cases/.

is the entity about whom the claim is created. The **Holder** in the end gets the verified credential to store in a wallet. **Entity** thereby refers to a digital identity represented by a DID.

Often Subject, Holder and Issuer are the same entity. For example, someone could issue a claim that they are a German citizen. If the Verifier, some office of the German administration, verifies this claim, the credential should be held by the citizen, in their wallet. This verified credential can then be presented to whoever asks.

In another case, the company (Issuer) that hosts my web server (Subject) issues a claim that it has an A+ rating for its security configurations. The rating organization (Verifier) agrees and sends the verified claim to my wallet (Holder).

The W3C discusses a wide variety of claims for which the data model shall be standardized so that they can be understood by different implementations and within a variety of DID methods. Examples are claims about academic achievements or medical prescriptions as well as financial creditability and, of course, basic information like citizenship, age, address and name.

3 Applications and Use Cases for Digital Identities

There are two additional use case examples we would like to discuss in more detail to illustrate how an SSI implementation could work and change the way we go about things.

3.1 Verifiable Work History Credentials

For consulting companies like us, references to previous projects and skills acquired there are an essential part of the bidding process for new projects. Some prospective clients want to know if the consulting company and/or the individual consultants it is sending on the project have experience in a single field of work or a specific project role or if they worked on projects for a specific number of days or worth a specific amount of money.

With verifiable credentials and digital identity management, a consulting firm could issue claims to its current (and former) clients and have its projects verified. The individual work aspects of the consultants who worked on these projects could also be verified, which would then be issued to the consultant’s respective digital identity as well as the company’s (see Fig. 5).

Consultants would have their work history with them, not only when their company tries to get them into a new project, but also when they apply for another job. Likewise, prospective clients can automatically verify all the credentials the consulting firm claims to have, down to each single work experience of any consultant offered.

Employing certain cryptographic methods, such as zero-knowledge-proof implementations, the consulting company could even keep secret who exactly these previous clients were, if not permitted to give them as a reference, but the prospective new client could still be sure that the credential is valid. It is also possible for the consulting firm to only deliver cryptographic proof that the total value of their projects in the past surpasses a certain benchmark level, without revealing the actual number and thereby keeping the daily rates of the consultants in question a secret.

3.2 From Personal to Corporate Identities

The principles of digital identifiers not only have their benefits in proving eligibility by age or wage for individuals as seen in the previous chapters. They can also be applied to the public sector to gain information about companies.

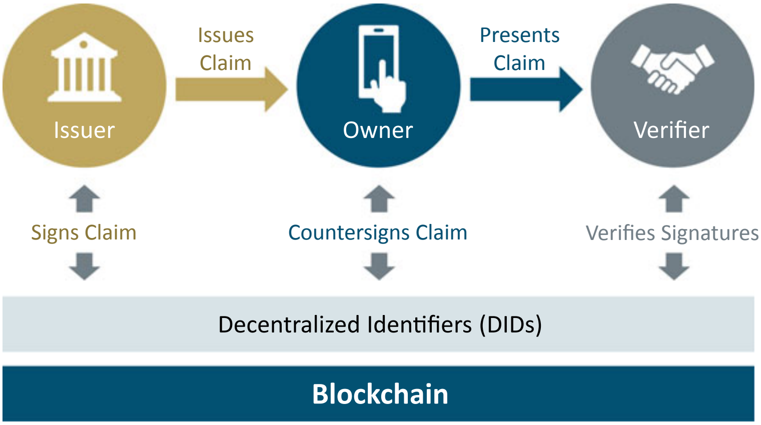


Fig. 5 Blockchain-based DID method[12]

[12] Following the source on https://blockchainaustralia.com.au/self-sovereign-identity-zero-knowledge-proofs-future-of-data/.

Companies could make the web safer and less prone to phishing attacks by using their DID to simply identify themselves back to customers. But there are more attributes that are of common interest. Let's take the *rating* as an example since it plays a critical role in risk management and is therefore of great interest for banks.

Only a few rating agencies create independent but quite untransparent ratings for the biggest corporations. The fact that the agencies are paid by the issuers creates a "conflict of interest: Issuers could, and do, 'buy' high ratings from willing sellers, the rating agencies" (Hill 2010). On the other hand, smaller companies and self-employed persons are rated by their bank itself, also without a second opinion.

By using a blockchain this whole process could be made more transparent and more independent: Imagine a company, a risk set provider (could be the bank) and a verifier (another independent service) (see Fig. 6). If a company requires risk data for any reason, a request for creation can be initiated toward the provider who will generate the first set of risk data and save it to the blockchain. In the second step of the process, the verifier will check the data and then either support it by issuing a risk certificate or reject it with the request for updating toward the risk set provider. This procedure can be continued as often as desired. The more verifiers support the provider's opinion (e.g. the company is a client of different banks with different rating systems), the more precise and significant the risk set will become.

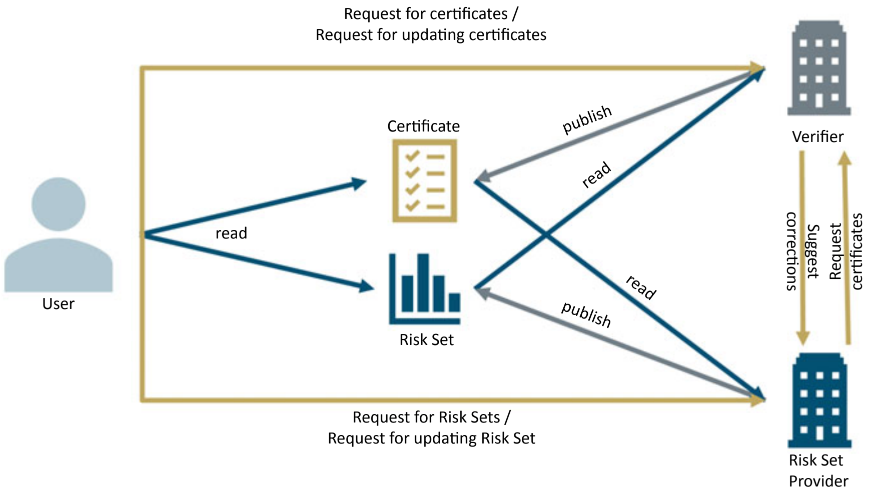


Fig. 6 The rating assessment process (© ifb SE)

This approach has advantages for all parties involved: The rating assessment is not dependent on the opinion of a single rating agency or a single assessor. The company itself can follow the whole process via the immutable blockchain and eventually use the issued certificate to sign documents or use it for "zero-knowledge-proof"-like processes to prove the company's rating. And the user of the rating (i.e. creditor) can be sure that the rating assessment has not been improperly influenced by the company because it has been validated by many verifiers.

This basic principle can also be applied to all other types of risk information, for instance, all types of securities and investment funds. All you need are a risk set provider and at least one verifier, ideally multiple verifiers. For securities, the actual performance could even be a verifier for the expected risk (using smart contract logic).

4 Conclusion

As this article outlined, digital identities have both the theoretical foundations and the procedures for practical application. The only question is: When and how will this technology be usable in everyday life and actually improve the identification beyond doubt of business and contractual partners? The essential prerequisite for this is the framework conditions that governments must create and support in practice. The Province of British Columbia (Canada) is actively working on standards and hosting proof-of-concept implementations of self-sovereign identity approaches on its websites. In Germany, ifb group's main area of business activity, the Federal Government also feels responsible for promoting the use of digital identities in accordance with its blockchain strategy (German Federal Ministry for Economic Affairs 2019). In order to test the practical implementation of self-sovereign digital identities, the Federal Government plans to hold an innovation competition in accordance with this strategy. It intends to test and evaluate the interoperability of different technologies and providers.

The electronic identity card (eID) has been in existence in Germany since November 2010 and could now be used by 62 million citizens because their ID card has the necessary integrated chip (German Federal Ministry of the Interior 2020). The challenge now is to create sufficient incentives for companies and private individuals to use this functionality in everyday life. This requires companies to design their processes and communication channels (i.e. online portals and apps) in a way that allows customers and business partners to use electronic identification without any problems. Ultimately, the confidence of citizens that electronic identity procedures are secure and yet easy to use must also be strengthened in order to create a true basis of trust.

Once the use of digital identities and identification procedures, especially that of self-sovereign identity (SSI), has gained acceptance, it will open up previously unimagined possibilities for fast and secure processing of all types of business transactions. It will also significantly improve the security of transactions and contracts concluded electronically.

Literature

Allen, Christopher. 2016. "The Path to Self-Sovereign Identity," April 25. Accessed August 23, 2021. http://www.lifewithalacrity.com/2016/04/the-path-to-self-sovereign-identity.html.

German Blockchain Association, Identity Working Group of the German Blockchain Association. 2018. " www.bundesblock.de," October 23. Accessed August 23, 2021. https://www.bundesblock.de/wp-content/uploads/2019/01/ssi-paper.pdf.

German Federal Ministry for Economic Affairs. 2019. *Blockchain Strategy of the Federal Government*, September 18. Accessed August 23, 2021. https://www.bmwi.de/Redaktion/EN/Publikationen/Digitale-Welt/blockchain-strategy.html.

German Federal Ministry of the Interior. 2020. *Zehn Jahre Personalausweis mit Online-Ausweisfunktion*, October 29. Accessed August 23, 2021. https://www.bmi.bund.de/SharedDocs/pressemitteilungen/DE/2020/10/10-jahre-personalausweis.html.

Hill, Claire. 2010. *University of Minnesota Law School—Why Did Rating Agencies Do Such a Bad Job Rating Subprime Securities?* Accessed August 23, 2021. https://scholarship.law.umn.edu/cgi/viewcontent.cgi?article=1084&context=faculty_articles.

Machine Learning and Deep Learning

The final part of the volume takes a closer look at how to analyze data using machine learning and deep learning. Open-source availability was one of the accelerators of the success of the models implemented in the different programming languages and embedded into different frameworks. Open-source availability of the major frameworks set a conveniently low barrier to enter this variety of models, and therefore provided the opportunity for experimentation with low initial costs.

There are also several good proprietary machine learning and deep learning frameworks provided by software and standard software vendors. Even the standard software vendors offer interfaces for the open-source implementation of machine and deep learning frameworks, to extend and continuously improve their variety of models.

The first chapter (Liermann, Overview Machine Learning and Deep Learning Frameworks 2021) gives a comprehensive but not exhaustive overview of machine learning and deep learning frameworks. The chapter seeks the origins and the circumstances of the frameworks' emergence. In addition, the programming languages that are used to build and to interface with the frameworks are examined. The second chapter (Liermann and Li, Methods of Machine Learning, 2021) covers a few additional aspects of machine learning like model validation and imbalanced data. Furthermore, model interoperability is discussed. A more detailed introduction to machine learning in general can be found in (Liermann, Li and Schaudinnus, Mathematical Background of Machine Learning 2019).

Literature

Liermann, Volker. 2021. "Overview Machine Learning and Deep Learning Frameworks." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Liermann, Volker, and Sangmeng Li. 2021. "Methods of Machine Learning." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Liermann, Volker, Sangmeng Li, and Norbert Schaudinnus. 2019. "Mathematical Background of Machine Learning." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Overview Machine Learning and Deep Learning Frameworks

Volker Liermann

1 Introduction

There are four operations to perform with data: (A) to store data, (B) to transfer and structurally transform data and (C) to analyze data to find patterns, which could either be used to group and understand data or (D) could be used to translate the pattern to other contexts including the projection of patterns onto the future (see Fig. 1).

The first two points are classical data management tasks, but with a high disruption potential in terms of the tools used. In these areas, new tools like in-memory databases (e.g. SAP Hana[1]), cluster-databases (e.g. Hadoopxx[2] or Amazon S3, Google Cloud Storage, Azure Data Lake) or streaming technology itself (Kafka[3]) or in combination with data stores (Lambda and Kappa architectures) are already established and up and running.

The next points in Fig. 1 aim to discover information in the available data. These areas (and the tools available on the market to perform this task) will

[1] See Kopic et al. (2019).

[2] See Akhgarnush et al. (2019).

[3] See Steurer (2021).

V. Liermann (✉)
ifb SE, Grünwald, Germany
e-mail: Volker.Liermann@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_12

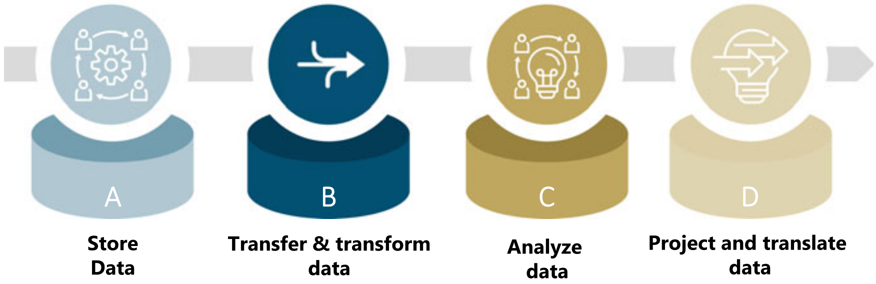


Fig. 1 Four actions for data (© ifb SE)

be the focus of the analysis carried out in this chapter. It is obvious that frameworks or toolsets to some extent try to target all four areas to provide an end-to-end experience. Nonetheless, we will focus on what the discussed frameworks contribute to the analytical areas.

The chapter is structured as follows: Sect. 2 explores the origins of the frameworks (standard software, universities, Big Tech and others). The main block Sect. 3 includes a comprehensive but not exhaustive list of machine learning and deep learning frameworks. In Sect. 4 the focus lies on the programming language used for implementing the frameworks and the offered programming language available to connect to the frameworks. Section 5 summarizes the chapter and analyzes the benefit of standard software and open-source frameworks.

2 Origins

In this first section we will try to discover the different roots of the frameworks and the intentions behind the projects.

To understand one of the origins, it is helpful to consider how software was developed. When a company needs to solve a problem, it can buy tools from the market or develop these tools in-house. In the early days of software development (middle of last century), most of the software was developed in-house. Then a standardization started, and software vendors understood that many problems were similar and similar tools were needed. The best example for this effect is Microsoft Office or SAP R/3. The standard software vendors still deliver tools for the problems they identify to generate sufficient profit to justify development costs (or acquisition costs[4]).

Over time, many software packages have been developed for various problems and markets, even for niches. Now, some companies have discovered that their self-developed software can be a differentiator (eg.LinkedIn,[5] Google[6] and not the in-house legacy systems of banks and insurance companies). Making the in-house development public—using the open-source idea—was a game-changer (see Liermann, Open Source and GAFAM 2021). This allowed the community to reuse and, more importantly, further develop the tools.

This dynamism in further development has led to the establishment of an entire sector by the open-source initiatives themselves (e.g. the Apache Software Foundation[7] or the Linux Foundation, the associated sub-divisions such as Hyperledger[8] or LF AI & Data[9]).

Another origin is academia, where new ideas and tools are born and discussed. Many universities provide software packages as applications of the mathematical and statistical algorithms. These universities conduct research on the algorithm themselves as part of computer science research. Common starting points for such university-driven frameworks are Ph.D. theses or research groups developing the frameworks for teaching and research. A typical next step often consists of university spin-offs or start-ups providing professional support (following the pattern of Redhat (Linux) and Cloudera (Hadoop) (Fig. 2).

2.1 Standard Software Vendors

All major standard software vendors provide frameworks for analysis using machine learning and deep learning. An initial differentiation in this area should be carried out by separating the standard software vendors into those who are rooted in statistical analysis (like SAS, Mathematica, …) and those that have a broader functional base (like Oracle, SAP, IBM and Microsoft).

[4] While it has become more and more common to buy successful companies (start-up or established companies) to lower the development risk (or, to be more precise, to transform the development risk into an integration risk).

[5] The initiator of Kafka, see Steurer (2021).

[6] Google did not initiate Hadoop but contributed to the concept (BigFiles and later Google File System), which had an impact on Hadoop.

[7] See The Apache Software Foundation (2020).

[8] See Bettio et al. (2019) and The Linux Foundation, Hyperledger (2020).

[9] See The Linux Foundation, LF AI & Data (2020).

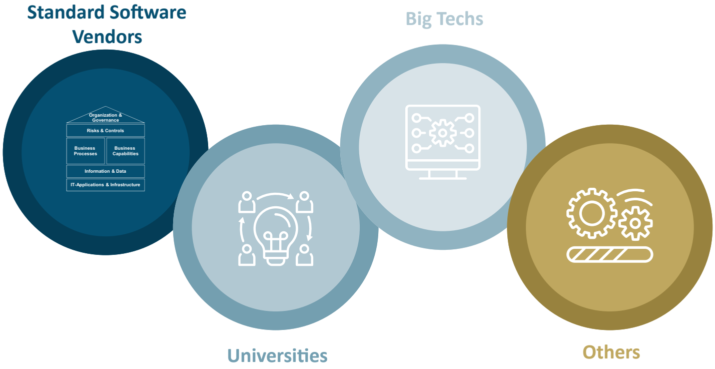


Fig. 2 Machine learning/deep learning frameworks—origins (© ifb SE)

Standard software vendors with roots in statistical analysis merely integrated the new algorithms into their existing and well-established software components. Data composition, data cleansing and version control were the requirements existing for data analysis, which were already met by these vendors. More and more, they integrated programming language interfaces (R, Python), enabling them to integrate open-source frameworks and enrich the zoo of models available to the data scientists.

The other standard software vendors have a strong record in data management and provide a good integration of their frameworks into the systems where the relevant data for analysis (e.g. customer data for CRM[10] incl. NBO, NBA,[11] transaction data or flow data like payments) is first generated and stored. The integration of open-source software is often offered, even though the extent and the technology sophistication differ in the group. The only vendor with its own differentiating model is IBM with Watson. Microsoft (and to some extent IBM) have contributed significant frameworks to the open-source community.[12] Microsoft provides open-source (Infer.NET, CNTK,[13] ML.NET, DeepSpeed, LightGBM) as well as proprietary frameworks (Azure ML) for machine learning in general and deep learning in particular.

[10] Customer relationship management.

[11] Next best offer, Next best action (see May 2019).

[12] Microsoft is located somewhere between the standard software vendors and Big Tech (GAFA or GAFAM) and has a special role in the ML/DL framework area.

[13] The Microsoft Cognitive Toolkit.

2.2 Universities

A significant number of American, Australian[14] and European universities or research institutes have produced ML/DL frameworks (and even some data mining and data management frameworks[15]). The following list of examples is non-exhaustive: University of California, Berkeley (Caffe[16]), the Polytechnic University of Catalonia or BarcelonaTech (OpenNN), CERN—European Organization for Nuclear Research (ROOT), Technical University of Dortmund (RapidMiner), Georgia Institute of Technology (mlpack), University of Konstanz (KNIME), University of Ljubljana and Jožef Stefan Institute (Orange), University of Massachusetts Amherst and the University of Pennsylvania (MALLET), Ludwig Maximilian University of Munich (ELKI), Université de Montréal (Theano), French Institute for Research in Computer Science and Automation in Rocquencourt (scikit-learn), University of Waikato, New Zealand (Weka/MAO), Wisconsin-Madison and University of Texas (Octave), University of Waterloo (Maple).

In many cases, the starting point was driven by research or educational intentions. After an initial phase at the university, these frameworks were further developed by a working group either in the university or across universities. One strategy often observed is the establishment of a start-up from a university spin-off, including models with community and enterprise software distributions/versions (see Sect. 3.4 in Liermann, Open Source and GAFAM 2021).

2.3 Big Techs

One of the most powerful origins for ML/DL frameworks are the American Big Techs[17] and other technology-oriented companies (such as LinkedIn or Netflix). GAFA in particular are able[18] to invest in these technologies and provide their own developments via an open-source license.[19]

[14] Referring to the continent, or New Zealand, to be precise.

[15] E.g. KNIME (Konstanz Information Miner).

[16] Caffe was replaced by Caffe2 in 2017 and then integrated into PyTorch by Facebook.

[17] Especially the American Big Techs (GAFA), and we have already discussed the localization of Microsoft (standard software vendor or Big Tech) in this chapter.

[18] Due to their "deep pockets" filled with revenues from advertising (particularly Google and Facebook) or other differentiating activities like Amazon's cloud services.

[19] Like GNU General Public License (GPL), GNU Lesser General Public License (LGPL), Mozilla Public License, BSD license (Berkeley Software Distribution) or Apache Software License (for details see Sect. 2.3 in Liermann, Open-Source Software 2021).

Google's Brain Team developed the open-source software library for machine learning called TensorFlow. First developed under the name DistBelief, it was widely used by various Alphabet companies. The rebranding was applied as TensorFlow became the second generation of deep learning frameworks at Google. Additionally, Google developed a special integrated circuit to support the heavy lifting in the model training: the Tensor Processing Unit (TPU). TPUs are AI accelerators using ASIC.[20] They are an alternative to GPUs,[21] which are widely used to accelerate the model training for deep learning models. The leverage provided by a TPU or a GPU is model-specific. While CNNs[22] are well-suited for TPUs, some fully connected neural networks can perform well on GPUs. The training of RNNs[23] can achieve good results with classical CPUs.

Facebook's AI Research lab (FAIR) was released in 2016. This was based on Torch, the open-source machine learning library PyTorch. In 2017, Facebook announced Caffe2 (see Facebook Open Source 2017). One year later, Facebook announced it would merge Caffe2 and PyTorch. PyTorch is used by other technology companies like Tesla and Uber.

With PaddlePaddle, Baidu[24] offers the largest open-source deep learning platform in China. PaddlePaddle is used in industries throughout China.

2.4 Others

Besides the well-known frameworks from the previous section, some other groups are worth mentioning.

2.4.1 Open-Source Foundations

The American non-profit corporation Apache Software Foundation contributes open-source software to many sectors like Hadoop and other projects. Driven by the user requirements, the Apache Software Foundation extended the project scope to machine learning (Apache SINGA) and deep learning (MXNet).

[20] Application-specific integrated circuit.

[21] Graphics processing unit.

[22] Convolutional neural networks or Deep convolutional networks (DCN), see Sect. 3.6 in Liermann et al., Deep learning—an introduction (2019).

[23] Recurrent neural network, see Sect. 3.2 in Liermann et al., Deep learning—an introduction (2019).

[24] Baidu is a multinational technology company focused on services and products in an internet-related context and artificial intelligence (AI). Baidu is one of the Chinese Big Tech companies (BATX).

2.4.2 Hardware Vendors

The hardware vendors, and especially some of the integrated circuit manufacturers, started to develop their own toolkits designed to best leverage their integrated circuits. Intel with Intel Data Analytics Acceleration Library and the Intel Math Kernel Libraryis an example of this approach. Intel also developed PlaidML, a portable tensor compiler which can generate OpenCL, OpenGL, LLVM or CUDA[25] code.

Nvidia took a different approach with its deep learning software stack NVIDIA CUDA-X AI, by integrating the existing and well-established frameworks[26] into its software stack. As the name suggests, the integration into its GPUs is established via CUDA.

With Core ML, Apple provides a machine learning framework that can be used to integrate machine learning models into iOS and OSX apps. Core ML supports models designed and trained with Create ML and also models from other frameworks, such as TensorFlow.

2.4.3 New Software Vendors

This short section is to be differentiated from the "standard" software vendors because the vendors mentioned here have been in business for less than thirty years, which does not mean they cannot become standard in their areas or that these vendors do not already have this status. The companies and products mentioned below focus on Big Data analysis.

H2O is an open-source software from the company H2O.ai (previously 0xdata) for Big Data analytics. Three Stanford professors form a panel to advise H2O on scientific issues.

Palantir Technologies is a software company around Big Data analytics. Its main products are Palantir Metropolis (quantitative analytics, data integration and information management) and Palantir Apollo (continuous delivery system using a micro-service architecture).

Faculty AI (formerly ASI Data Science), based in London, is a British technology company. Faculty AI provides a platform for machine learning/deep learning and uses—among others—Jupiter Notebooks and R-Studio.

[25] Compute Unified Device Architecture by Nvidia is an application programming interface (API) model and a parallel computing platform.

[26] Caffe/Caffe2, Chainer, Microsoft Cognitive Toolkit, MATLAB, Mxnet, PaddlePaddle, PyTorch, Tensorflow, Torch and Mathematica's Wolfram Language.

Artelnics is a Spanish start-up further developing the deep learning open-source framework OpenNN, which was initially developed by Polytechnic University of Catalonia in Barcelona.

Preferred Networks, Inc. is a Japanese venture company. Preferred Networks was one main contributor (among others[27]) to the deep learning framework Chainer. Preferred Networks decided to redirect its effort to Facebook's open-source deep learning framework PyTorch and will only provide maintenance for Chainer.

ThetaRay is a Big Data analytics company with headquarters in Hod HaSharon (Israel) and offices in New York and Singapore.

Qunatumblack is part of the McKinsey group and provides the open-source framework Kredo for developing data and machine learning pipelines. Other so-called strategy consultants have expanded—often with separate companies—into the space of data science.

3 Frameworks

The frameworks listed in this section are structured by machine learning (which is a superset of deep learning), deep learning and integrated data handling, data mining and modeling tools.

Figure 3 shows the relationship between the generic term "artificial intelligence" and the subsets "machine learning" (representation learning) and "deep learning". Artificial intelligence (AI) can be further divided into strong and weak AI. Strong AI[28] refers to algorithms that can implement consciousness and self-awareness. We are still many decades away from strong AI, nonetheless it is a research subject addressed by a number of organizations.[29] Weak AI[30] is limited to specific and narrowed tasks. Natural language processing and the Big Tech voice assistants are examples of weak AI.

The line-up presented in this section can—at the most—be comprehensive at the time it was written[31] and will never be exhaustive.

[27] Intel, Nvidia, Microsoft and IBM.

[28] Also referred to as artificial general intelligence (AGI) or full AI.

[29] See Baum (2017).

[30] Also referred to as narrowed AI.

[31] It is to be expected that things will have already moved forward by the time the book is published.

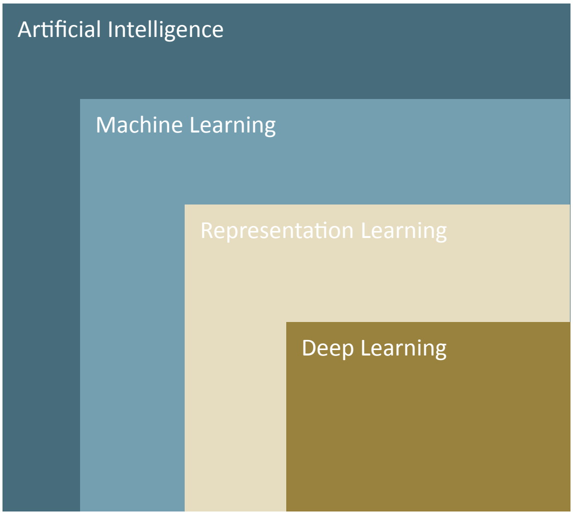


Fig. 3 Overview artificial intelligence (© ifb SE)

3.1 Machine Learning

The frameworks are of different importance to the corporations and the community in terms of functional scope and integration into cloud platforms. The chosen order is alphabetical and does not reflect the importance, coverage or complexity of the frameworks. The following compilation will only touch on the frameworks/platforms extremely lightly as most of them are worth a dedicated chapter. The list is extensive but not exhaustive.

3.1.1 Amazon Machine Learning

With Amazon Machine Learning, Amazon provides a platform and tools to create machine learning applications. In the platform, data can be provided by S3 (simple storage services), Redshift and RDS methods. Amazon Machine Learning supports the common frameworks like PyTorch, TensorFlow and MXNet.

3.1.2 Ayasdi

Ayasdi was founded in 2008 and provides a platform for data analysis. One differentiator is the Topological Data Analysis (TDA) capacity. The company has its roots at Stanford university.

3.1.3 Apache SystemDS

Apache SystemDS (previously, Apache SystemML) is an open-source machine learning system designed to scale at Spark and Hadoop clusters. SystemML was initially developed in 2010 by IBM.

The main features in the framework are: (A) Multiple execution modes, including Standalone, Spark Batch, Spark MLContext, Hadoop Batchand JMLC. (B) Automatic optimization based on data and cluster characteristics to ensure both efficiency and scalability. (C) Programming interface via R-like and Python-like languages.

3.1.4 DAAL

Data Analytics Acceleration Library by Intel (Intel DAAL) offers algorithmic building blocks for data analysis stages for Big Data problems. DAAL supports data platforms like Hadoop, Spark, R and MATLAB.

3.1.5 Dlib

Dlib[32] is a C++ written free software library for machine learning, image processing and machine vision. An interface for Python is provided.

3.1.6 ELKI

Environment for DeveLoping KDD-Applications Supported by Index-Structures (ELKI) is a research project that was originally developed at Ludwig Maximilian University of Munich and is now continued at the Technical University of Dortmund. It is a modular software package ("framework") written in Java for knowledge discovery in databases. It focuses on procedures for cluster analysis, outlier detection and the use of index structures in such procedures.

[32] See Dlib (2020).

3.1.7 Google Prediction API

The site ml.googleapis.com provides an API for machine learning functionalities and is part of the Googles APIs like Google Maps (an overview of all Google APIs is provided in Google Inc. 2020). The Google Prediction APIs are part of the AI Platform Training and Prediction API (Training).

3.1.8 H2O.ai

H2O.ai[33] implements algorithms from the field of statistics, data mining and machine learning (generalized linear models, K-Means, random forest, gradient boosting and deep learning). Interfaces can be used with R, Python, Apache Hadoop and Spark. H2O.ai is open source.

3.1.9 IBM SPSS Modeler

IBM acquired the statistical software package SPSS in 2009 and rebranded it IBM SPSS Modeler. The IBM SPSS Modeler offers data mining and text analytics.

3.1.10 IBM Watson Studio

In February 2011, Watson, a computer program developed by IBM, competed in Jeopardy! against record champions Ken Jennings and Brad Rutter. Watson won the three-day tournament. Most people therefore associate Watson with a computer winning Jeopardy!, but it is also the brand[34] for IBM's data science platform.

IBM Watson Studio offers a workspace for data scientists including multiple open-source and collaboration tools. The platform can be deployed on-premises, in the cloud and on desktops. It supports various languages (R/Python/Scala) and integrates open-source tools including RStudio and Spark.

[33] See H2O.ai, h2o.ai Overview (2020).

[34] Formerly known as Data Science Experience or DSX.

3.1.11 Infer.NET

Infer.NET is an open-source software library implemented in.NET providing machine learning models. Infer.NET offers models for different kinds of standard machine learning problems such as clustering, recommendation or classification. A team at Microsoft's research center in Cambridge initiated the development in 2004. Before releasing it under an open-source license (MIT License) in 2018, it was released in 2008 for academic use.

3.1.12 LightGBM

The LightGBM framework[35] offers a comprehensive set of tree-based learning algorithms focusing on gradient boosting algorithms (like GBT,[36] GBDT ,[37] GBRT,[38] GBM,[39] MART[40]). It is a free and open-source framework developed in the first place by Microsoft targeting performance and scalability.

LightGBM supports Linux, Windows and macOS and has interfaces to C++, C#, R and Python.

3.1.13 LIONsolver

LIONsolver is a proprietary software package for business intelligence, analytics and data mining by Reactive Search srl. It includes the modeling of intelligent optimization[41] (branded by "Learning and Intelligent OptimizatioN"). Learning and Intelligent OptimizatioN builds an integration of online machine learning models into the optimization algorithms; the combination is capable of learning from recent runs and from user feedback.

Although LIONsolver has a proprietary license, a free academic license for non-commercial use is available (LIONoso).

[35] See also Microsoft Corporation (2020).

[36] GBT—Gradient boosted trees.

[37] GBDT—Gradient boosted decision trees.

[38] GBRT—Gradient boosted regression trees.

[39] GBM—Gradient boosting machine.

[40] MART—Multiple additive regression trees.

[41] Intelligent optimization is automated creation and selection of improving solutions.

3.1.14 Microsoft Azure Machine Learning

Microsoft Azure Machine Learning (Azure ML) composes ML frameworks and a set of tools to develop and set up machine learning and AI services. The platform includes the deployment and management of machine learning models, the use of AutoML and the setup of pipelines.

3.1.15 Microsoft Azure Cognitive Services

Azure Cognitive Services contain both cognitive and perceptual intelligence applying natural language, OCR/form understanding, processing, machine translation, face recognition, computer vision, neural speech synthesis, speech recognition, speaker recognition and business decision services. Microsoft uses many of the provided APIs, SDKs and customizable services in its own products like Bing, Teams, Office, Xbox or even Windows.

3.1.16 Mahout

Apache Mahout was initially released in 2009 by the Apache Software Foundation, aiming to offer free implementations of distributed and scalable machine learning algorithms. In recent years, the primary implementation platform shifted from Hadoop Spark.

Java/Scala libraries can be used to provide algorithms for the Mahout platform. Java/Scala is used to provide linear algebra and statistics. Mahout is open source and is further developed by a community.

3.1.17 MALLET

MALLET is a machine learning toolkit for language processing written in Java. MALLET is a toolkit aiming for statistical natural language processing (NLP), information extraction, topic modeling document classification, cluster analysis and other machine learning tasks in the context of text analysis.

3.1.18 ML.NET

ML.NET was originally developed by Microsoft and is a free software offering a machine learning library. It supports Python models when used together with NimbusML[42] but was initially designed for the Microsoft programming languages C# and F#. ML.NET can handle binary classification, multi-class classification and classical regression tasks.

Other models like anomaly detection and recommendation systems are now included, and Microsoft plans to include deep learning models in future versions. The framework is designed to provide machine learning and prediction capabilities to.NET developers. ML.NET sits on top of .NET Core and .NET Standard. Therefore, ML.NET is cross-platform (Windows Linux and macOS).

3.1.19 mlpack

mlpack is a machine learning library built on top of the Armadillo[43] library implemented in C++. mlpack is supported by contributions from a community and the Georgia Institute of Technology.

3.1.20 MOA

Massive Online Analysis (MOA) is developed by the University of Waikato, New Zealand, as a free open-source project. MOA is written in Java and supports the analysis of evolving data streams, by using data mining and machine learning experiments. MOA is designed for Big Data streams in real time and large-scale machine learning. MOA is connected to Weka (see Sect. 3.4.15 in this chapter) using a bi-directional interface.

3.1.21 NeuroSolutions

NeuroDimension developed the proprietary neural network development environment NeuroSolutions aiming to provide a user-friendly interface to design, train and deploy neural networks. NeuroSolutions has an icon-based network design interface. NeuroSolutions can be used for time-series

[42] Python module that provides Python bindings for ML.NET.

[43] Armadillo is a library for linear algebra (see Conrad Sanderson and Ryan Curtin 2020).

prediction and classical data mining as well as classification and multivariate regression.

3.1.22 Oracle Data Mining

Oracle Data Mining (ODM) is an optional part of Oracle Database Enterprise Edition. ODM includes data mining and data analysis algorithms for, for example, regression, prediction, classification, anomaly detection and feature extraction. It provides tools to create, manage and deploy data mining models into the oracle database environment.

3.1.23 Oracle AI Platform

Oracle AI Platform provides a platform for data science and machine learning libraries implemented in python. The platform is open source and part of Oracle's Platform as a Service (PaaS) initiative. Oracle—like other vendors—benefits from the libraries of the scientific and data science communities that have implemented many of the complex data science algorithms (including some of the most popular deep learning frameworks, like TensorFlow).

3.1.24 PolyAnalyst

PolyAnalyst is a proprietary data science software platform for text mining, data mining, machine learning and predictive analytics. The platform was developed by Megaputer Intelligence. PolyAnalyst has a user-friendly graphical interface to organize data collection and analysis steps in nodes of a flowchart. These nodes can represent data import, data preparation, data visualization, data analysis and data export.

3.1.25 RCASE

Root Cause Analysis Solver Engine (informally RCASE) by Warwick Analytics, a spin-out company, is a proprietary algorithm developed for an automated solution of root cause analysis.[44] RCASE was initially developed on top of research at Warwick University (Warwick Manufacturing Group).

[44] RCASE offers methods aiming to identify the root causes of faults or problems. It has applications—among others—in incomplete data and dirty data cleansing.

3.1.26 SAP Leonardo Machine Learning

SAP Leonardo Machine Learning is an umbrella to the Predictive Analytics Library(PAL) situated in the SAP Hana database, SAP Predictive Analytics and the SAP Leonardo Machine Learning Foundation.

SAP Leonardo Machine Learning Foundation

The SAP Leonardo Machine Learning Foundation offers an infrastructure with a range of machine learning applications and services. It includes the ability for a contribution and deployment of one's own machine models. The platform addresses data management, model management (by providing a model repository store) and accessibility to trained models. The functionalities are rounded off by secure communication channels and an authorization concept.

SAP Predictive Analytics Library

SAP Predictive Analytics Library is an application function library (AFL) from SAP-defining functions which can be embedded into SAP HANA SQLScript procedures performing analytic algorithms.

3.1.27 SAP InfiniteInsight

KXEN Inc. was acquired by SAP AG in 2013 and was initially founded by Roger Haddad and Michel Bera. KXEN targets predictive analytics tasks. The KXEN Modeler was rebranded as SAP InfiniteInsight. SAP InfiniteInsight automates many predictive analytics efforts.

3.1.28 SAS Enterprise Miner

SAS Enterprise Miner is one of the components of the statistical software suite for data management developed by SAS Institute. SAS Enterprise Miner targets data mining and predictive analytics. The SAS component has a graphical environment including designing and executing steps in deploying a predictive analytic model.

3.1.29 scikit-learn

scikit-learn (formerly scikits.learn) is a free machine learning software library in Python. It provides various algorithms for classification, regression and clustering, including support vector machines, random forest, gradient boosting, k-means and DBSCAN.[45] SciKit (short for SciPy Toolkit) is based on the NumPy and SciPy numerical and scientific Python libraries, as is scikit-image, for example. scikit-learn is largely written in Python. Some core algorithms were implemented in Cython for performance reasons.

3.1.30 Shogun

Shogun is an open-source toolbox from the field of machine learning. It can be used to solve regression and classification problems and to learn Hidden Markov Models, among other things. Shogun is implemented in C++ and offers interfaces to MATLAB, R, Octave and Python. The focus of this toolbox is bioinformatics applications.

3.1.31 Spark MLlib/SparkML

The function of MLlib (and its successor SparkML bundle) is to make typical machine learning algorithms available for distributed Spark systems. The library on top of Spark Core is a distributed machine learning framework integrated into the distributed memory-based Spark architecture.

3.1.32 XGBoost

XGBoost[46] is an open-source library providing gradient boosting framework with interfaces to C++, Java, Python, R, Julia, Perl and Scala. XGBoost provides different distributed gradient boosting models (such as GBM,[47] GBRT[48] and GBDT[49]).

[45] Density-Based Spatial Clustering of Applications with Noise is a popular data mining algorithm for cluster analysis (see Ester et al. 1996).

[46] See XGBoost Contributors (2020).

[47] GBM—Gradient boosting machine.

[48] GBRT—Gradient boosted regression trees.

[49] GBDT—Gradient boosted decision trees.

XGBoost is integrated into scikit-learn for Python users. R users can access XGBoost via the R package caret. The supported operating systems are Windows, Linux and macOS.

3.2 Deep Learning

The frameworks/platforms listed here have their key area in deep learning models and modeling. As in the list of machine learning frameworks/platforms, the list presented is extensive but not exhaustive.

3.2.1 Apache MXNet

Apache MXNet is an open-source deep learning framework, aimed at the data scientists' steps of training and deploying deep neural networks. Apache MXNet supports a comprehensive set of programming languages (e.g. C++, Python, R, Java, JavaScript, Julia, Scala, MatlabGo, Perl and Wolfram Language).

3.2.2 BigDL

BigDL is an open-source deep learning framework initiated by Intel. BigDL is designed for a distributed use in Apache Spark or Hadoop.

3.2.3 Caffe/Caffe2

CAFFE (Convolutional Architecture for Fast Feature Embedding) was a deep learning framework, initially developed at the University of California, Berkeley. It included different types of neuronal network designs, such as CNN, RCNN and LSTM as well as fully connected neural networks. GPU- and CPU-based acceleration computational were supported by Caffe (e.g. NVIDIA cuDNN or Intel MKL).

In 2018, Caffe2 was merged into PyTorch (see Sect. 3.2.21 in this chapter) and will be developed further in this context, but NIVIDA has its own fork called NVCaffe. NVIDIA Caffe is an NVIDIA-maintained fork of BVLC Caffe tuned for NVIDIA GPUs, particularly in multi-GPU configurations.

3.2.4 Chainer

Chainer was written purely in Python on top of NumPy and CuPy. It is an open-source deep learning framework initially developed by the Japanese venture company Preferred Networks and supported by IBM, Intel, Microsoft and Nvidia. Chainer was the first to implement the define-by-run approach for deep learning.

In 2019, Preferred Networks (one of the prime supporters) decided to redirect its development effort from Chainer to PyTorch.

3.2.5 CNTK

The Microsoft Cognitive Toolkit[50] (CNTK) is an open-source deep learning framework formerly developed by Microsoft Research. CNTK describes neural networks as a series of computational steps via a directed graph.

CNTK is no longer actively supported and further developed by Microsoft.

3.2.6 Darknet

Darknet is an open-source neural network framework specialized in object recognition written in C and CUDA. The You Only Look Once (YOLO)[51] feature is supported by Darknet.

3.2.7 Deeplearning4j

Deeplearning4j is a free, cross-platform program library for machine learning. Deeplearning4j is written in Java and is compatible with any other language running on a JVM (Scala or Clojure). The underlying computations are implemented in C and C++.

To bridge the gap between Deeplearning4j and widely used but Python-based program libraries, Deeplearning4j implemented an application programming interface (API) via Keras. Keras allows the import of artificial neural network models from the most popular frameworks (including TensorFlow, Caffe, Torch and Theano systems). The implemented models include

[50] Sometimes referred as The Microsoft Cognitive Toolkit.

[51] YOLO is a one-step, real-time object detection system. A one-step detection strategy means that the images to be analyzed need to be read only once.

distributed parallel algorithm versions that integrate with Apache Spark and Hadoop.

Commercial support is provided by the company Skymind.

3.2.8 Deep Learning Toolbox (MATLAB)

Deep Learning Toolbox is a proprietary framework by MATLAB for the design and implementation of deep neural networks with pre-trained models. Models can be exchanged with other frameworks like TensorFlow and PyTorch through the ONNX format. They can also be imported from TensorFlow-Keras and Caffe/Caffe2.

3.2.9 DeepSpeed

DeepSpeed is an open-source deep learning acceleration library for PyTorch. DeepSpeed targets the reduction of memory use and processing capacity in order to train large, distributed models with more intelligent parallelisms on the given computer hardware. DeepSpeed is implemented in Python, CUDA and C++ .

3.2.10 DyNet

DyNet is written in C++ (with bindings in Python). Primarily developed by Carnegie Mellon University, DyNet is a neural network that has GPU support for computing-power-intensive training. DyNet performs well with networks that have dynamic structures and change for every training instance.

3.2.11 fast.ai

fast.ai is a non-profit research group targeting deep learning and artificial intelligence. In 2018, fast.ai released the free open-source deep learning library fastai[52] located on top of PyTorch.

[52] See fast.ai (2020).

3.2.12 Flux

Flux[53] is one of the few comprehensive machine learning libraries written in Julia[54] (Flux.jl). The open-source machine learning ecosystem and library supports GPU (via CuArrays.jl) and TPU.

3.2.13 Gluon

Gluon, initiated in 2017 by Amazon and Microsoft, is a Keras-like API layer for MXNet and CNTK. By using Gluon, developers can create machine learning models using a simple Python API and start off with pre-built neural network components. Gluon provides an API for deep learning.

3.2.14 MXNet

Apache MXNet is an open-source deep learning software framework developed by the Apache Software Foundation. MXNet is aimed at training and deploying deep neural networks. It has a comprehensive coverage of supported programming languages (including C++, Python, R, Java, JavaScript, Scala, Julia, MATLAB, Go, Perl and Wolfram Language).

MXNet is designed to be distributed on dynamic cloud infrastructure, allowing almost linear scale with multiple GPUs or CPUs by providing a distributed parameter server.

3.2.15 Neural Lab

Neural Lab is an AI company based in Hong Kong. Neural Lab offers a free neural network simulator with the same name integrated in Microsoft Visual Studio. It provides a visual environment to design and test neural networks.

[53] See flux (2020).

[54] Julia is a high-level programming language targeting numerical and scientific computing. Julia can also be used as a general-purpose language with high execution speed. Julia was created with the goal of being as easy on statistics as R is.

3.2.16 Open AI

OpenAI LP is a company that, controlled by the non-profit organization OpenAI Inc, is engaged in artificial intelligence (AI) research. OpenAI is a competitor of the Alphabet company DeepMind.[55]

GPT-3 (Generative Pretrained Transformer 3), which is an artificial intelligence in natural language processing, is the first product of OpenAI and is available to a selected audience via the OpenAI API.

3.2.17 OpenCV

OpenCV is a free program library focusing on image processing and computer vision. It is written for the programming languages C, C++, Python and Java and is available as free software under the terms of the BSD license.

The DNN module of OpenCV can read in the networks pre-trained by popular deep learning frameworks.[56]

3.2.18 PapplePaddle

PaddlePaddle (PArallel Distributed Deep LEarning) is a deep learning platform, initially developed by Baidu for products at Baidu such as NLP (natural language processing), translation and image processing.

PaddlePaddle, as the only independent R&D deep learning platform in China, has been officially open sourced to professional communities since 2016.

3.2.19 PlaidML

PlaidML is an advanced and portable tensor compiler initially developed by Vertex.AI, which was acquired by Intel in 2018. Intel is still developing PlaidML and an open-source project.[57] PlaidML connects to Keras, ONNX and nGraph.

PlaidML supports all major operating systems: Linux, macOS and Windows.

[55] DeepMind (formerly Google DeepMind) is a company specializing in artificial intelligence (AI) programming. DeepMind was founded in September 2010 and acquired by Google Inc. in 2014.

[56] The following libraries are supported: Caffe, TensorFlow, Torch, Darknet, models in ONNX format.

[57] See plaidML (2020).

3.2.20 TensorFlow

TensorFlow is the second most popular deep learning framework. It is a framework for data-stream-oriented programming and machine learning. TensorFlow (initially DistBelief, started in 2011) was originally developed by the Google Brain team for Google's internal needs and released in 2017 under the Apache 2.0 open-source license.

TensorFlow version 2.0 (also known as TensorFlow 2), released in September 2019, cleaned up and extended the API. In particular, the Keras interface became the focused API for modeling deep learning models. TensorFlow 2 now stores models including weights and computations, making them much easier to share.

The main differentiator is the integration of the Tensor Processing Units (TPU)[58] developed by Google. With this special hardware, the algorithms of the TensorFlow program library are processed particularly quickly and efficiently. TensorFlow supports Keras among other interfaces. Naturally, TensorFlow supports GPU acceleration via CUDA as well.

Besides the mobile iOS and Android, the supported operating systems include Windows, Linux and macOS.

3.2.21 Theano

Theano is an open-source machine learning library written mainly in Python (and CUDA). It was published in 2007 by the Montreal Institute for Learning Algorithms. Theano converts the computational instructions into C++ or CUDA code that runs efficiently on normal computer processors (CPU) or on graphics processing units (GPU). Theano can be used as a backend for the Keras framework.

3.2.22 Torch/PyTorch

PyTorch is one of the most popular deep learning frameworks. It is an open-source machine learning library primarily developed by Facebook's AI Research lab (FAIR). FAIR used PyTorch and Caffe2 in its development. FAIR tries to bridge the two models by the Open Neural Network Exchange (ONNX). Caffe2 was integrated into PyTorch in 2018.

[58] Application-specific chips, to support and accelerate machine learning optimized for tensor operations.

Although the Python interface is well-known due to the name and the development focus, PyTorch has an additional C++ interface. PyTorch is based on Torch, which was mainly written in the Lua programming language but has parts written in C, C++ and CUDA.

3.2.23 SINGA

Apache SINGA is an open-source machine learning library initially developed by the DB System Group at the National University of Singapore in 2014. SINGA is now a top-level Apache project providing a flexible architecture for scalable distributed training focusing on health care applications. The main differentiator is training parallelization and distributed deep learning by partitioning the model and data onto cluster nodes.

3.3 Special Frameworks for Natural Language Processing (NLP)

The multitude of machine learning and deep learning frameworks mentioned in the two previous subsections have components that address the standard tasks[59] for natural language processing (such as Chainer, Deeplearning4J, Keras, scikit-learn, TensorFlow and Theano, to name just a few).

NLP[60] is an important skill in digital transformation in outside digitalization (client-oriented) and inside digitalization (targeting the internal organization and processes).[61] Nonetheless, this section cannot cover all frameworks and toolkits relevant for NLP. Apache OpenNLP

The Apache OpenNLP library written in Java is a machine-learning-based toolkit by the Apache Software Foundation initiated in 2004. OpenNLP targets the processing of natural language.

[59] NER (Named Entity Recognition), Word embeddings and Seq2Seq models (see Sect. 4 in Liermann et al., Deep learning—an introduction 2019).

[60] Including to some extent Natural Language Understanding and Natural Language Generation.

[61] For inside and outside digitalization (see Liermann and Stegmann, Introduction 2021).

3.3.1 Apache cTAKES

Apache cTAKES[62] is a special-purpose NLP library. The open-source system extracts clinical information from electronic health records consisting of unstructured text. It is built using the UIMA.[63]

3.3.2 BERT/mBERT

BERT[64] was released together with the BERT paper: Pre-training of Deep Bidirectional Transformers for Language Understanding (Devlin et al. 2019).BERT offers a transformer-based machine learning technique for natural language processing. The implementation of BERT by PyTorch is provided with Google's pre-trained models.

mBERT stands for Multilingual BERT (mBERT) and gives sentence representations for 104 languages, to offer a toolkit for many multilingual tasks.

3.3.3 DeepNL

DeepNL is a natural language processing pipeline performing standard tasks like NER and SRL[65] based on a deep learning neural network architecture. DeepNL is written in Python.

3.3.4 ERNIE

ERNIE is Baidu's language framework. ERNIE (Enhanced Representation through kNowledge IntEgration) was introduced in 2019. It uses a multi-layer transformer like BERT. The version 2.0 offers continual pre-training for language understanding. ERNIE is part of Baidu's PaddlePaddle.

[62] clinical Text Analysis and Knowledge Extraction System.

[63] Unstructured Information Management Architecture framework. UIMA was initially developed by IBM.

[64] Bidirectional Encoder Representations from Transformers.

[65] Systematic Role Labeling.

3.3.5 Gensim

Gensim is targeted to perform sentiment analysis and unsupervised topic modeling. It supports large corpora topic modeling, corpora document indexing and corpora similarity retrieval.

3.3.6 Microsoft Icecaps

Microsoft Icecapsis a conversational modeling open-source toolkit. It is designed to build customized neural conversational systems. Microsoft Icecaps provides a set of tools covering the general NLP tasks and the most recent conversation modeling.

3.3.7 nlpnet

nlpnet offers natural language processing tasks based on neural networks. The Python and Cython written toolkit provides the user with part-of-speech tagging, semantic role labeling and dependency parsing. nlpnet uses the Natural Language Toolkit (NLTK), a Python library for human language data processing.

3.3.8 OpenNMT

OpenNMT targets neural machine translation and neural sequence learning. Starting in December 2016, this open-source ecosystem was initiated by the Harvard NLP group and SYSTRAN. Since then, the project has been used in several research and industry applications. It is currently maintained by SYSTRAN[66] and Ubiqus.[67] OpenNMT offers a TensorFlow implementation (OpenNMT-tf) and a PyTorch implementation (OpenNMT-py).

[66] SYSTRAN (SYStem TRANslation) is a software system for machine translation founded in the 1960s.

[67] Ubisqus is a language service provider.

3.3.9 SpaCy

SpaCy, written in Python and Cython, is designed to provide access to the more recent developments in NLP. It connects with other well-established libraries like Gensim and scikit-learn. SpaCy is a lightweight deep learning library. Sitting on top of SpaCy is Thinc.

3.3.10 Stanford CoreNLP

CoreNLP is a natural language processing library written in Java. An NLP pipeline aiming to take in raw text and produce a set of annotations is the main component of CoreNLP.

3.3.11 Texar-PyTorch

Texar-PyTorch is an open-source framework using PyTorch. It aims to support a broad set of text tasks like text generation tasks, content manipulation, machine translation and dialog.

3.3.12 Transformers

Transformers is a natural language processing toolkit for PyTorch and TensorFlow 20.0. Initially it was developed for PyTorch (named pytorch-transformers and pytorch-pretrained-bert). Transformers implements a general-purpose architecture[68] for NLU[69] and NLG[70] covering more than 30 pre-trained models and a deep interoperability between the major deep learning frameworks TensorFlow 20.0 and PyTorch.

3.4 Integrated Data Handling, Data Mining and Modeling Tools

In this subsection, we only mention the frameworks with a close integration of data pre- and post-model processing into the machine learning and deep learning frameworks the chapter focuses on. All standard software vendors have comprehensive tools for extracting, transforming and loading

[68] For the NPL frameworks BERT, DistilBert, GPT-2, RoBERTa, XLM and XLNet.

[69] Natural Language Understanding.

[70] Natural Language Generation.

data as well as tools for reporting and visualizing results. The line-up in this subsection is far from exhaustive and would be worth a separate chapter.

3.4.1 Artelnics

Artelnics is a Spanish company offering software to develop and apply artificial intelligence. Neural Designer and OpenNN are two of the products offered partially under open-source license.

3.4.2 Alteryx

Alteryx is a California-based computer software company. Its products aim to make advanced analytics accessible to any data worker (data scientist and data analyst).

3.4.3 Dataiku

Dataiku is an AI and machine learning company founded in 2013. Dataiku developed the Dataiku Data Science Studio software. The software aims to allow companies to exploit their data through predictive methods.

3.4.4 Neural Designer

Neural Designer is a data mining proprietary software for general machine learning. It implements algorithms for regression analysis and pattern recognition. The Neural Designer is based on an artificial neural network developed in the open-source program library OpenNN.

OpenNN

OpenNN (Open Neural Networks Library) is a program library written in C++ that implements neural networks. The library can be used for machine learning, data mining and predictive analytics tasks. OpenNN started as a research project funded by the European Union and is now developed further by Artelnics, a start-up company.

3.4.5 Dplyr / tidyverse

Dplyr is one of the core packages of the tidyverse in the R programming language authored primarily by Hadley Wickham. Dplyr was launched in 2014. Primarily, it assembles functions designed to enable data frame manipulation in an intuitive, user-friendly way.

Dplyr is, like Pandas, a popular R package for juggling (joining, transforming and grouping data).

3.4.6 Faculty AI

Faculty Science Ltd. (formerly ASI Data Science and Advanced Skills Initiative Ltd.), a British technology company based in London, is an applied AI company founded in 2014. Faculty Science clients include the UK government.

3.4.7 KNIME

KNIME, the "Konstanz Information Miner", is a free software for interactive data analysis with a user-friendly interface. Machine learning and data mining methods can be lined up in a modular pipelining concept. KNIME offers an easy and fast assembly of modules for data pre-modeling processing[71] and post-modeling processing.[72]

Since June 2008, a Zurich-based company (KNIME AG) has also established professional technical support and consulting services for the KNIME platform.

3.4.8 Kedro

Kedro[73] is an open-source Python framework for production-ready machine learning models. Kedro was initially developed by QuantumBlack and was transformed into an open-source license after the acquisition of QuantumBlack by McKinsey.

[71] Extraction, transformation, loading (ETL).

[72] Analysis, visualization and reporting.

[73] See QuantumBlack (2020).

3.4.9 Orange

Orange is an open-source software by the University of Ljubljana (Slovenia) for data mining and machine learning including data visualization. The core element is visual programming of the user interface. This is implemented in user-designed widgets which are linked together to form workflows. Orange is supported by the standard operating systems (Windows, Linux and macOS) and can also be installed from the Python Package Index repository via pip.[74]

3.4.10 Palantir Technologies

Palantir Technologies, Inc. is a US-based provider of software and services specializing in Big Data analytics. Founded in 2004, Palantir's first customers included federal agencies in the United States Intelligence Community (USIC). Palantir extended its customer base to include the financial and pharmaceutical industries.

The company is particularly known for two software projects: Palantir Gothamand Palantir Foundry. Palantir Gotham integrates structured and unstructured data and is used by counterterrorism and cyber risk analysts. Palantir Foundry is a data integration, information management and quantitative analytics software. The software can be applied to different datasets and reveals trends, relationships and anomalies, using predictive analytics.

3.4.11 Pandas

Pandas is a program library for the Python programming language that provides tools for data management and analysis. In particular, it contains data structures and operators for accessing numerical tables and time series.

3.4.12 RapidMiner

RapidMiner, previously called YALE (Yet Another Learning Environment), is a machine learning and data mining environment. YALE has been developed since 2001 by the Chair of Artificial Intelligence at the Technical University of Dortmund. In late 2013, RapidMiner received funding from venture

[74] Pip—a Python package-management system—targets installing and managing Python software packages.

capital firms. The capital was raised and used for the US market strategy. RapidMiner's corporate headquarters are in Boston.

RapidMiner is sold under a proprietary license and offers certain components with reduced functionality as open source.

3.4.13 ROOT (TMVA with ROOT)

ROOT[75] is an object-oriented free software developed at CERN which is used for the analysis of data. ROOT is executable under the operating systems Linux, macOS and Windows, among others. Originally developed for analyzing data in the field of particle physics, ROOT is now used in additional application areas such as data mining.

The Toolkit for Multivariate Data Analysis with ROOT (TMVA) sits on top of ROOT. TMVA aims to provide integrated projects for machine learning, multivariate classification in particular, and other regression techniques targeting applications in high-energy physics.

3.4.14 ThetaRay

ThetaRay was founded in 2013 by two University professors from Yale University and Tel Aviv University. ThetaRay's platform utilizes patented mathematical algorithms developed by the company founders. The platform is also used to uncover patterns in Big Data datasets.

3.4.15 Weka

Weka (Waikato Environment for Knowledge Analysis) is a software that provides various techniques from the fields of machine learning and data mining. WEKA's workbench is divided into the following components: preprocessing, modeling and visualization.

The software is an integral part of the book "Data Mining: Practical Machine Learning Tools and Techniques" (see Witten et al. 2011).

[75] See ROOT/TMVA team (2020).

3.5 Mathematical Software

The platforms listed here have their origins in mathematical and scientific programming and have been in the marketplace for quite some time. Most of them have expanded their functionality toward machine learning and deep learning.

3.5.1 Mathematica

Mathematica by Wolfram Research is a proprietary software package representing well-known and widely used mathematical and scientific program packages. Mathematica 1.0 was initially launched in 1988.

It supports supervised and unsupervised machine learning tools for data, images and sounds including artificial neural networks.

3.5.2 MATLAB

MATLAB is a commercial software of the US company MathWorks for the solution of mathematical problems and the visualization of the results. MATLAB is primarily designed for numerical calculations using matrices. The name (MATrix LABoratory) originates from this target.

MATLAB was developed in the late 1970s at the University of New Mexico to help students to use the programming libraries through a user-friendly interface without programming skills. MATLAB is utilized by scripting a proprietary programming language.

3.5.3 Sage

Sage or SageMath is a free software for solving mathematical problems. Sage is available for Linux and macOS and Windows (see SageMath 2020).

4 Programming Language

In practical work with machine learning and deep learning frameworks, two tasks are important in terms of the programming language used: (A) the application of the models in the framework and (B) the extensibility of the framework and its models.

In the following subsection, we give an overview of the programming languages used to implement the framework and how to use the framework. The latter is the main task for classical data scientists to conduct their data analysis (addressed in Subsection 4.2). The implementation of new models or optimization algorithms is a less common task executed by experts and—depending on the complexity of the problem—requires in-depth modeling and programming skills (addressed in Subsection 4.1).

The last Subsection 4.3 deals with the challenge of transferring model structures or trained models from one framework to another.

4.1 Implementation

Implementation (and thus extension) is often carried out in the classical programming languages: C++ (and similar C#, C) and Java. Often only certain (performance-demanding) parts are implemented in C++ and Java. Python is also quite common, especially in the field of deep learning.

Examples of C implementation are Caffe/Caffe2, CNTK, Infer.NET, LightGBM, OpenNN and Wolfram Mathematica. Apache Mahout, Deeplearning4j, Weka and H2O are examples of Java. In some cases, Java and Scala are both used (MXNet, Mahout). PyTorch, Chainer and Theano are implemented in Python only. Many machine learning and almost all deep learning[76] frameworks offer GPU support via Nvidia's CUDA.

Less common but also used are Julia and Go (used by Apache's Mahout and MXNet) and Cython.[77] Exotic but nonetheless worth mentioning are Lua (Torch), Fortran and F# (Orange, Octave, ML.Net).

A special role is played by R,[78] which is the open-source variant of S.[79] Due to its statistical roots, the scripting language R has a great variety of models and libraries (R packages) providing ML models and even some DL models written in R. There are bridges/interfaces to Java, C and Python,[80] so even if the frameworks do not support R, in most cases their own implementation of a bridge to ML/DL frameworks is possible. Most frameworks offer an interface to R (see the following subsection).

[76] Driven by the algorithmic demand (for a simple example see subsection 2.1.4 in Liermann et al., Deep learning—an introduction 2019) for training the deep learning networks, which is to some extent similar to the requirements for 3D graphics.

[77] The programming language is a superset of the Python programming language and aims to give C-like performance to most code parts in Python with additional C-oriented syntax.

[78] R is a free software environment and programming language for statistical computing and graphic representation. The development started in 1993.

[79] S is a statistical programming language developed in 1970 at Bell Laboratories.

[80] The R package reticulate offers an integration of python code to R see Ushey (2020).

All frameworks mentioned in this subsection are examples; the list is not exhaustive.

4.2 Interfaces

Data scientists mainly use Python and R to bind data from the data lakes or other data sources, to parameterize the models and to analyze the statistical quality of the models. To implement a trained model into a production environment, often Java (and sometimes Python) is used because of the lean integration into web applications and other surroundings.

Data scientists used to be split into R lovers and Python lovers, but nowadays a growing number of data scientists use both languages for model development and training. The deep learning models traditionally have more of a Python flavor and classical machine learning (without deep learning) is closer to R.

Other common languages used for interfacing (model training and model optimization) are C, C++ and Java and JavaScript. Software vendors provide their proprietary languages to use models like MATLAB, Wolfram Language and SAS language.[81] More exotic languages are Julia, Swift[82] and Kotlin.[83]

4.3 Model Description Standards and Universal Interfaces to Frameworks

The frameworks are driven by different stakeholders and initiatives with varying targets and missions. Therefore, the frameworks naturally started out in different directions. Some of the research groups or developer groups used different frameworks in their daily routines (like FAIR with PyTorch and Caffe2).

The selection of the best framework for a given problem becomes a challenge. In the competition between the frameworks, the problem definition outside the ordinary application can be a challenge. The appropriate model might be only available in certain frameworks or the framework is not integrated into the cloud environment the institute is using. Model description formats have therefore evolved and offer interfaces to different frameworks.

[81] The SAS language is a statistical analysis programming language designed by Anthony James Barr at North Carolina State University and is the standard language for the SAS software modules.

[82] App programming language of iOS and other Apple operating systems.

[83] Kotlin designed for interoperability with Java. Kotlin's JVM version is based on the Java Class Library.

Given the fact that some frameworks still have advantages compared to others, the demand for interoperability and an easy exchange between the frameworks is rising. Two approaches offer a solution to this challenge: (A) a universal interface on top of a framework (e.g. Keras) or (B) a standard for describing the model topology (ONNX and NNEF).

This subsection will highlight just three of the popular multi-framework interfaces.

4.3.1 Keras

One of the most popular interfaces to artificial neural networks is Keras. This is an open-source deep learning library written in Python. It was initiated by François Chollet in 2015. Keras provides a unified interface for several backends, including TensorFlow, Microsoft Cognitive Toolkit and Theano. The Keras target is to make the use of these libraries as beginner- and user-friendly as possible.

Keras is part of the Tensorflow Core API; however, Keras continues as a standalone library. According to François Chollet, it was not intended as an interface for Tensorflow only, but as an interface for many libraries. With the release of Keras 2.4, multi-backend support was discontinued; with this release, Keras refers directly to the implementation of Tensorflow 2.

4.3.2 nGraph

nGraph Compiler aims to improve and develop AI workloads by bridging existing models to any deep learning framework including the and deployment toward a variety of hardware targets.

4.3.3 NNEF

Neural Network Exchange Format (NNEF) was designed by the Khronos Group. It aims to enable data scientists to easily exchange trained networks from the training framework into other deployment environments.

4.3.4 ONNX

ONNX (Open Neural Network Exchange)[84] is an open format for representing deep learning models. ONNX allows data scientists to exchange models between different frameworks offering the best combination of these frameworks for them. ONNX addresses two aspects: (A) Framework interoperability[85] and (B) Shared optimization.[86]

Microsoft, Amazon, Facebook and other partners jointly developed ONNX as an open-source project. In November 2019, ONNX moved under the umbrella of the LF AI Foundation (which is a sub-organization of The Linux Foundation).

ONNX.js is a JavaScript library for running ONNX models in web browsers and on Node.js. ONNX.js can run on both the CPU and the GPU.

5 Summary

The line-up provided in this chapter is only a snapshot and needs to be updated continuously. Even as a snapshot, this chapter can only claim to be comprehensive but never exhaustive.

Most of the frameworks support Python and R (the classical data scientist open-source languages).

A specific analysis would be worthwhile, to explore how deeply and conveniently the frameworks are integrated into the cloud provider or an on-premises environment (ML and DL platforms). In general, the integration of the standard software vendors among themselves (and sometimes even among competitors) is good and already quite sophisticated. Long-term support of these integrations might be a drawback for pure open-source frameworks, although there are promising approaches.

[84] See The Linux Foundation, Open Neural Network Exchange (2020).

[85] Allow developers to more easily move between frameworks, some of which may be more desirable for specific phases of the development process, such as fast training, network architecture flexibility or inferencing on mobile devices.

[86] Allow hardware vendors and others to improve the performance of artificial neural networks of multiple frameworks at once by targeting the ONNX representation.

Literature

Akhgarnush, Eljar, Lars Broeckers, and Thorsten Jakoby. 2019. "Hadoop—A Standard Framework for Computer Clusters." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Baum, Seth. 2017. *A Survey of Artificial General Intelligence Projects for Ethics, Risk, and Policy.* Working Paper 17-1, Global Catastrophic Risk Institute.

Bettio, Martina, Fabian Bruse, Achim Franke, Thorsten Jakoby, and Daniel Schärf. 2019. "Hyperledger Fabric as a Blockchain Framework in the Financial Industry." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Conrad Sanderson, and Ryan Curtin. 2020. *Armadillo.* Accessed December 15, 2020. http://arma.sourceforge.net/.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *Pre-training of Deep Bidirectional Transformers for Language Understanding.* Google AI Language.

Dlib. 2020. *Dlib.* Accessed December 15, 2020. http://dlib.net/.

Ester, Martin, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise." *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96).*

Facebook Open Source. 2017. "Caffe2 Open Source Brings Cross Platform Machine Learning Tools to Developers." *Caffe2.* April 18. Accessed December 15, 2020. https://caffe2.ai/blog/2017/04/18/caffe2-open-source-announcement.html.

fast.ai. 2020. *fastai.* Accessed December 15, 2020. https://docs.fast.ai/.

flux. 2020. *flux.* Accessed December 15, 2020. https://fluxml.ai/.

Google Inc. 2020. *Google APIs Explorer.* Accessed December 15, 2020. https://developers.google.com/apis-explorer.

H2O.ai. 2020. *h2o.ai Overview.* December 15. Accessed December 15, 2020. http://docs.h2o.ai/h2o/latest-stable/h2o-docs/index.html.

Kopic, Eva, Bezu Teschome, Thomas Schneider, Ralph Steurer, and Sascha Florin. 2019. "In-Memory Databases and Their Impact on Our (Future) Organizations." In *The Impact of Digital Transformation and Fintech on the Finance Professiona*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Liermann, Volker. 2021. "Open-Source Software." In *The Digital Journey of Banking and Insurance, Volume II—Digitalization and Machine Learning*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Liermann, Volker, and Claus Stegmann. 2021. "Introduction." By Volker Liermann and Claus Stegmann, The Digital Journey of Banking and Insurance, Volume I–Disruption and DNA. New York: Palgrave Macmillan.

Liermann, Volker, Sangmeng Li, and Norbert Schaudinnus. 2019. "Deep Learning—An Introduction." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

May, Uwe. 2019. "The Concept of the Next best Action/Offer in the age of Customer Experience." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Microsoft Corporation. 2020. *LightGBM.* Accessed December 15, 2020. https://lightgbm.readthedocs.io/.

plaidML. 2020. *plaidML.* Accessed December 15, 2020. https://github.com/plaidml/plaidml.

QuantumBlack. 2020. *Kedro's Documentation.* Accessed December 15, 2020. https://kedro.readthedocs.io/en/stable/.

ROOT/TMVA team. 2020. *Toolkit for Multivariate Data Analysis with ROOT.* Accessed December 15, 2020. http://tmva.sourceforge.net/.

SageMath. 2020. *Sage.* Accessed December 15, 2020. https://www.sagemath.org/.

Steurer, Ralph. 2021. "Kafka—Real Time Streaming for the Finance Industry." In *The Digital Journey of Banking and Insurance, Volume III—Data Storage, Processing, and Analysis*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

The Apache Software Foundation. 2020. *The Apache Software Foundation.* Accessed December 15, 2020. https://www.apache.org/.

The Linux Foundation. 2020. *LF AI & Data.* Accessed December 15, 2020. https://lfaidata.foundation/.

———. 2020. *Hyperledger.* Accessed December 15, 2020. https://www.hyperledger.org/.

———. 2020. *Open Neural Network Exchange.* Accessed December 15, 2020. https://onnx.ai/.

Ushey, Kevin. 2020. *The Comprehensive R Archive Network.* Accessed December 15, 2020. https://cran.r-project.org/web/packages/reticulate/index.html.

Witten, Ian H., Eibe Frank, and Mark A. Hall. 2011. *Data Mining: Practical Machine Learning Tools and Techniques.* San Francisco, CA: Morgan Kaufmann.

XGBoost Contributors. 2020. *XGBoost.* Accessed December 15, 2020. https://xgboost.ai/.

Methods of Machine Learning

Volker Liermann and Sangmeng Li

1 Introduction

This chapter is an amendment to the Machine Learning chapter in the book "The Impact of Digital Transformation and Fintech on the Finance Professional" (Liermann etal., Mathematical background of machine learning, 2019).

Model validationis a crucial point in the whole model development process and one of the major tasks of a data scientist. In some models (e.g. ensemble methods[1]), meta-parameters can be used to optimize models by the number of false positives and false negatives, depending on the application of the model.

[1] Ensemble methods, sometimes also referred to as ensemble learning, are supersets of machine learning models. The idea is to use an intelligent combination of different models, so the model is more flexible and less data-sensitive. Prominent examples are Random Forest, Adaptive Boosting (AdaBoost) and Gradient Boosting.

V. Liermann (✉) · S. Li
ifb SE, Grünwald, Germany
e-mail: Volker.Liermann@ifb-group.com

S. Li
e-mail: Sangmeng.Li@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_13

The data we found in the practical work can pose challenges for the data scientist. Often the dataset is not big enough to get stable model results. Sometimes the dataset does not include the relevant information for the prediction required by the business problem. One special, but not infrequent challenge is that the dataset is sufficient in size but only has few true positives, implying a problem for some models. This data constellation is called "imbalanced data".

Models can find structures and use them for predictions of different types. Model explainability is an important model feature, not only because of the ethical requirement of non-discrimination. Model interpretability can build trust in a model on the one hand and on the other can give insights into the structures of the dataset.

The chapter is structured as follows: The second section looks at the different aspects of model validation and is followed by a section covering the challenges in imbalanced datasets. Before the summary in section five, section four discusses the different aspects of interpretability (local, global and intrinsic).

2 Model validation

After training a model, it is important to evaluate whether it can make the correct prediction or not. In this section, we will introduce some widely used performance metrics, which measure various aspects of model performance. Note that we consider the binary classification problem where the label takes two values: 1 and 0.

2.1 Confusion Matrix and Classification Thresholds

Confusion matrix is the most important classification metric. The metrics introduced in the coming sections are built based on this.

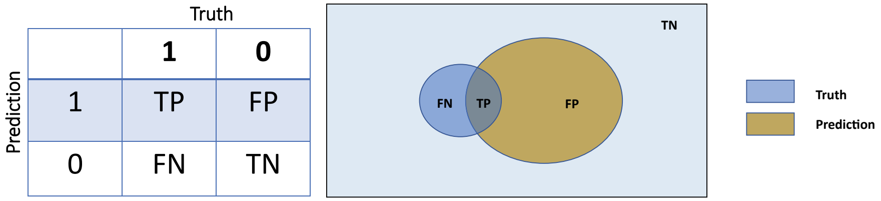
As illustrated above, the confusion matrix consists of four elements, which show how well the prediction matches the truth. Here, the class "1" is considered the positive class. We interpret TP, FP, FN and TN in the following by using the illustration in Fig. 1.

- TP (true positive): The predicted positives are actually positive.
- FP (false positive): The predicted positives are false and actually negative.
- FN (false negative): The predicted negatives are false and actually positive.
- TN (true negative): The predicted negatives are actually negative.

Note that FP and FN are also called statistic type 1 error and type 2 error, respectively.

The machine learning algorithms aim to predict a probability (score) of a class membership. Therefore, it is necessary to convert such probabilities into class labels. The choice of the class to be converted into is determined by a parameter called "threshold", which takes a value between 0 and 1. Simply speaking, the predictions whose scores are smaller than the threshold are assigned to the negative class (0); the predictions larger than the threshold are assigned to the positive class (1) (Table 1).

Additionally, we provide the following two extreme cases where the threshold is set up to be extremely large or small. The large threshold leads to many false positives and the small threshold causes more negative samples to be incorrectly predicted as positive (Figs. 2, 3 and 4, Tables 2 and 3).



Prediction \ Truth	1	0
1	TP	FP
0	FN	TN

Fig. 1 Confusion matrix (© ifb SE)

Table 1 Class assignment with threshold = 0.5 (© ifb SE)

Prediction	1	2	3	4	5	6	7	8	9
Score	0.56	0.48	0.58	0.93	0.26	0.56	0.61	0.12	0.87
Label	1	0	1	1	0	1	1	0	1

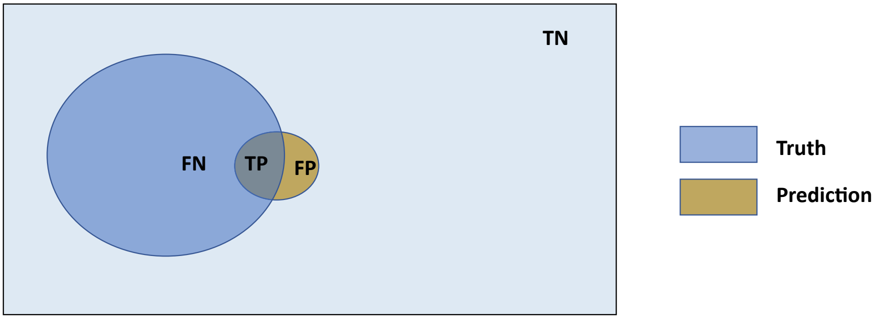


Fig. 2 Class assignment with threshold = 0.9 (© ifb SE)

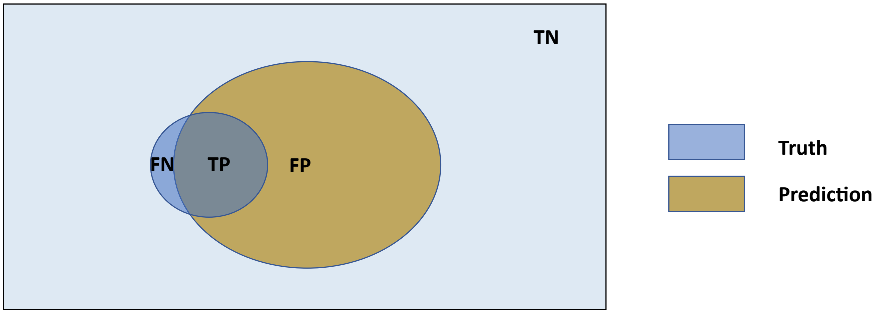


Fig. 3 Class assignment with threshold = 0.3 (© ifb SE)

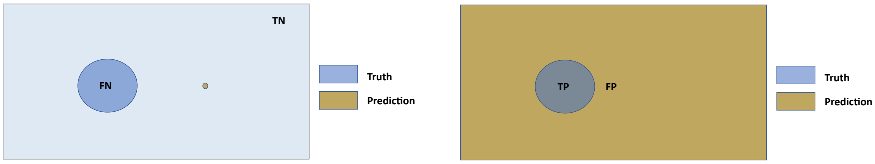


Fig. 4 Two extreme predictions (© ifb SE)

Table 2 Class assignment with threshold = 0.9 (© ifb SE)

Prediction	1	2	3	4	5	6	7	8	9
Score	0.56	0.48	0.58	0.93	0.26	0.56	0.61	0.12	0.87
Label	0	0	0	1	0	0	0	0	0

Table 3 Class assignment with threshold = 0.3 © ifb SE

Prediction	1	2	3	4	5	6	7	8	9
Score	0.56	0.48	0.58	0.93	0.26	0.56	0.61	0.12	0.87
Label	1	1	1	1	0	1	1	0	1

2.2 ROC Curve and ROC-AUC

We can generate different confusion matrices when the thresholds are changed. In this section, we are going to introduce the **Receiver Operator Characteristic curve (ROC curve)**. This curve is plotted at various threshold values of the True Positive Rate (TPR) against the False Positive Rate(FPR), where the TPR and FPR are defined as follows:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

Equation 1: True Positive Rate (TPR)

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

Equation 2: False Positive Rate

The ROC curve is **monotone increasing but not strictly**, since both TPR and FPR do not decrease if the threshold increases. This shows how well the model can separate the positive and negative classes. Note that the ROC curve **starts from the point (0,0) and ends at point (1,1).** The following two examples illustrate two extreme cases. In the example on the left-hand side, there is no sample predicted as positive. Therefore, we have TP = FP = 0, which leads to both TPR and FPR being equal to zero. In contrast, all samples are predicted as positive in the example on the right-hand side, so that FN = TN = 0, which leads to both TPR and FPR being equal to one.

In Fig. 5, we plot the probability distribution of positive and negative classes and assume that both distributions nearly overlap each other. This leads to the FPR and TPR increasing nearly at the same speed when the threshold becomes larger. In this case, the ROC curve is a line between point (0,0) and (1,1). This is the worst situation, and the model does not learn any useful structure from the data.

In contrast to that, Fig. 6 illustrates the ideal situation in which both classes are separated sharply. In this case, the predicted positives are always true until the whole positive class is predicted correctly. This causes the ROC curve to jump sharply from point (0,0).

In order to compare different ROC curves, we introduce another performance metric called **AUC (area under the curve)**, which measures the entire area underneath the ROC curve. Mathematically speaking, AUC can be computed along the integral of the ROC curve between (0, 0) and (1, 1). It is not hard to figure out that the AUC of the worst situation (Fig. 5) is equal

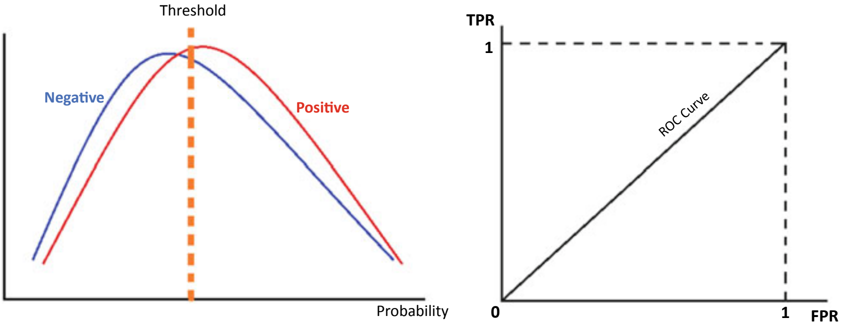


Fig. 5 Example "the worst model" (© ifb SE)

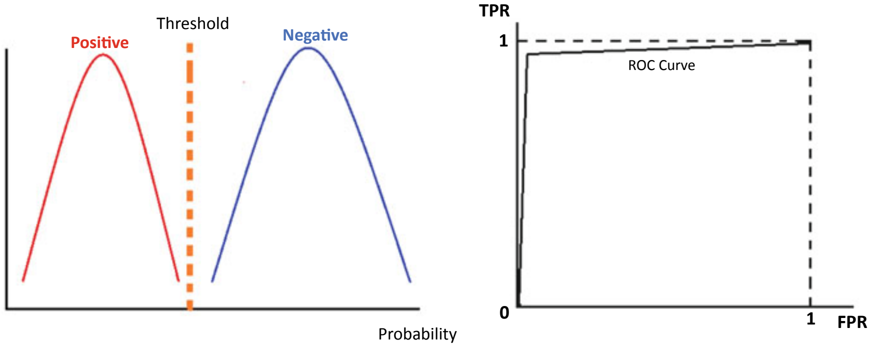


Fig. 6 Example "the ideal model" (© ifb SE)

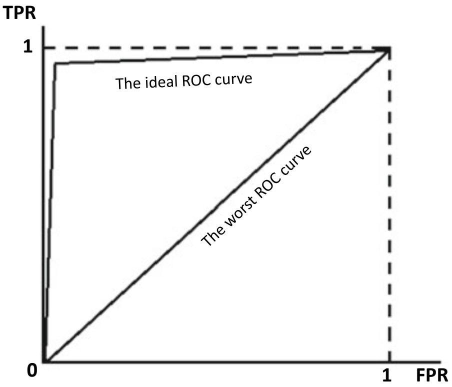


Fig. 7 ROC-AUC (© ifb SE)

to 0.5 and the ideal AUC (Fig. 6) should almost take the value of 1. The greater the AUC is, the more sharply the model can separate both positive and negative classes (Fig. 7).

2.3 Accuracy

Accuracy is a performance metric which measures how many samples are predicted correctly, defined as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{Equation 3: Accuracy}$$

2.4 Precision and Recall

Precision and Recall are performance metrics which are defined as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Equation 4: Precision

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Equation 5: Recall

Precision quantifies the number of positive class predictions that belong to the positive class, while Recall shows how many positive class members are predicted correctly. In contrast to Accuracy and AUC, the true negative will not be taken into consideration. In the next section, we will introduce a specific problem, whereas Precision and Recall are more appropriate for model validation.

3 Imbalanced Data

Imbalanced data is a common problem in classification where the classes are distributed unequally in the dataset (Johnson and Khoshgoftaar 2019). Medical diagnosis is inherently affected by imbalanced data where the occurrence of some sub-pathologies is scarcer than others (Shilaskar and Ghatol 2018). In this book series, we provide two application examples from the area of financial risk management which suffer from imbalanced data as well. The first example (Liermann et al., Batch processing—Pattern recognition 2019) aims to detect anomalies in batch processes that monitor banking key risk indicators (KRI) and key performance indicators (KPI). The second example, insurance fraud prevention, is provided in (Enzinger and Li 2021). This has relied on many intuitions in the past and is automated by using machine learning and deep learning.

We use a dataset from Kaggle as an example (Kaggle 2020), which contains 284,807 credit cardtransactions classified into two classes: “Fraud” and “Not Fraud”. Figure 8 illustrates that the classes are distributed imbalanced, where only 0.17% of samples are classified as being fraudulent.

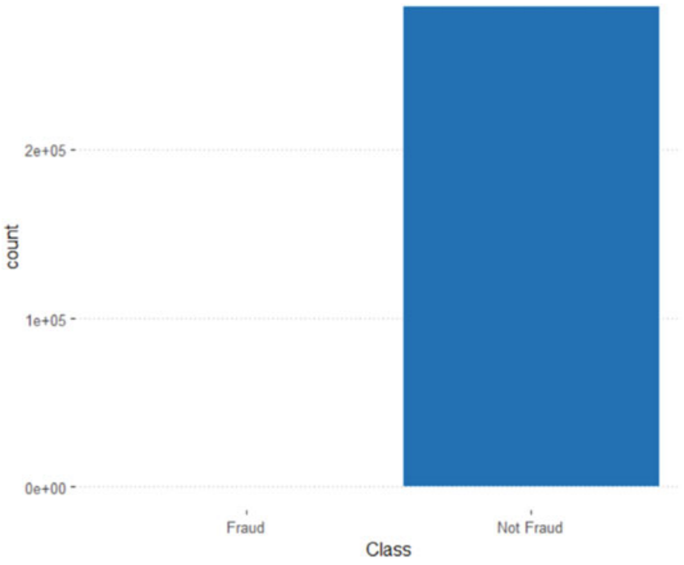


Fig. 8 Unequal distribution of classes "Fraud" and "Not Fraud" (© ifb SE)

The impact of imbalanced data on machine learning cannot be ignored. The unequal distribution of classes affects model training and causes underfitting, where the model can recognize patterns from the majority class but ignores the minority class. However, the minority class is always the one we target to make predictions, for instance, the class "Fraud" in fraud prevention and the class "Anomaly" in anomaly detection. Additionally, in the case of imbalanced classes, the validation metrics AUC and Accuracy might mislead the model performance. Given the credit card dataset introduced above, we assume that we predict all credit card transactions to be not fraudulent. Since only 0.17% of transactions are fraudulent, we are still able to achieve an extremely satisfying Accuracy of 99.83%, even if we do not train the model.

Recalling the definition of Accuracy given in Eq. 3, the true negative (TN) is also taken into consideration. A large number of true negative misclassifications leads to a high level of Accuracy. Similarly, it also decreases the false positive rate (FPR). As a result, the ROC curve increases abruptly after starting from the point (0,0) and ROC-AUC becomes significantly larger.

To end this section, we provide three of the most popular methods for dealing with imbalanced data.

3.1 Oversampling and Undersampling

This method is data based and aims to adjust class distribution by reducing the size of the majority class (undersampling) or duplicating the minority class (oversampling). In practice, oversampling is used more often than undersampling since the data structure might become incomplete after undersampling, which might cause underfitting.

3.2 Cost-Sensitive Learning

Cost-sensitive learning suggests adjusting the cost function and considering different misclassification errors under different cost levels. This method can not only be used for dealing with the imbalanced problem but also for fulfilling various business requirements.

Assuming that—for an insurance company—it is more expensive to misclassify a fraudulent claim than to misclassify a normal claim as fraudulent: In this case, we adjust the cost function by assigning larger weight to FN (false negative). The model mainly seeks to minimize the number of false negative misclassifications.

3.3 Performance Metrics

As we mentioned above, AUC and Accuracy are not appropriate for measuring model performance in case of imbalanced classes. In fact, we are only interested in the Accuracy of the minority class, but both AUC and Accuracy take the Accuracy of the majority class into consideration. In this case, we suggest using Precision and Recall for measuring model performance, where the true negative (TN) is not involved. In addition, we provide another metric F1-Score, which is defined as follows:

$$F1 = 2 * \frac{\text{Precision} + \text{Recall}}{\text{Precision} * \text{Recall}} = \frac{\text{TP}}{\text{TP} + 0.5 * (\text{TP} + \text{FN})}$$

Equation 6: F1 Score

This takes the weighted average of Precision and Recall.

4 Model Interpretability

The recent development of increasingly complex predictive models is causing growing interest in the transparency and explainability of models (Guidotti et al. 2018). In fact, the better the predictive power of the model is, the worse is the interpretability. As we discussed in other sections, with the growth in use of machine learning models within the banking and financial sector, it is more necessary to develop explainable and transparent models which can be easily modified and understood. In addition, model risk measurement becomes one of the challenges for banks and insurance companies, in which the regulators require financial institutes to supply transparent models for which model risk can be computed quantitatively. In this section, we provide an overview of popular techniques for extracting model interpretability (Mengnan Du 2019).

4.1 Intrinsic Interpretability

Intrinsically interpretable models have self-explainable structures. They include linear models, decision trees, attention models, etc. In linear regression, the coefficient parameter can be interpreted as feature contribution, which indicates the direction and intensity of the relationship between the features and labels. Decision trees are rule based and the feature that tends to split nodes closer to the root of a tree will result in a larger contribution (Fig. 9).

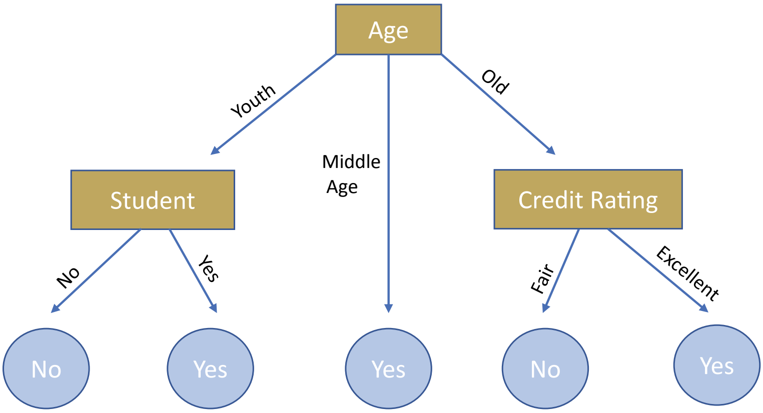


Fig. 9 Example of decision tree (© ifb SE)

$$y = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \beta_3 * x_3$$

Equation 7: Example of linear regression

4.2 Global Interpretability and Shallow Model Tree

In the professional literature (Guidotti et al. 2018), global and local categorizations are discussed. There, global interpretability means that the users can understand how the model works globally by understanding the structure and parameters of the model. For the linear model and the decision tree illustrated above, we can achieve a good global interpretability since the model itself has simple and explainable structures. In contrast to these, a neural network or an ensemble model may result in better prediction power but less interpretability due to the complexity of model structures.

High predictive power of models comes at the cost of transparency and interpretability. Data scientists search for techniques that can improve model interpretability without losing prediction accuracy. A shallow model tree is a tree-based model and enables high transparency and high predictive accuracy at the same time. It consists of a set of rules that forms a tree structure. Each leaf contains a simple predictive model. In Fig. 10, we provide an example of a model tree, in which each leaf contains a simple linear model. The model tree has a tree-based structure. Unlike classical decision trees, each leaf contains a simple model instead of a fixed value. This may increase predictive power but still delivers a high level of transparency. For more details about the model tree, we refer the reader to (Klaus and Gjergji 2019).

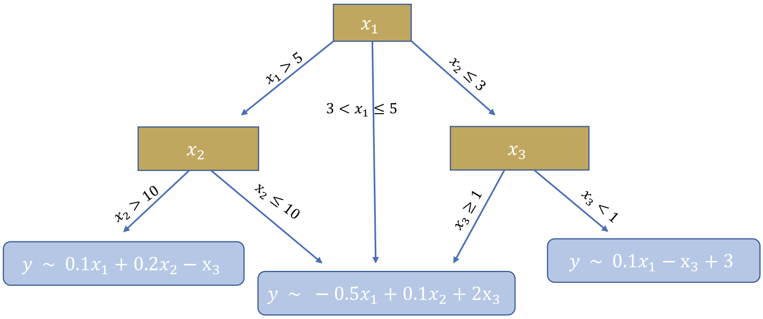


Fig. 10 Shallow model tree (© ifb SE)

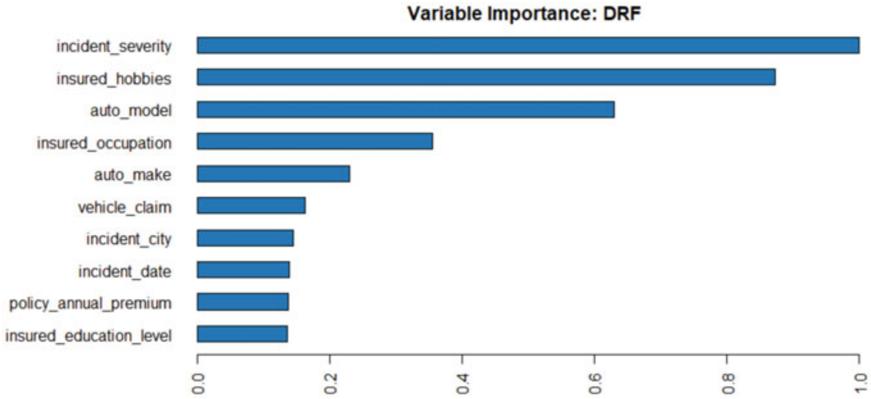


Fig. 11 Variable importance (© ifb SE)

4.3 Local Interpretability and Feature Importance

Unlike global interpretability, local interpretability aims to identify the contribution of each feature toward a specific model prediction. Feature importance is a famous metric that measures the feature contribution in several forms (Fig. 11).

4.3.1 Model-Specific Feature Importance

We take random forests as an example. The tree-specific feature importance computes the average decrease in Gini impurity across all trees by removing the feature. This is also used by default in several machine learning platforms.

4.3.2 Permutation Feature Importance

The decrease of model performance after permutating the value of features indicates the features' importance. This method is not preferable since it requires a long computation time when the number of features or the size of the dataset are large.

5 Summary

The chapter covers three important aspects of the implementation of practical machine learning models: model validation, model interpretability and imbalanced data.

The second section introduces the common key figures to judge whether a model is appropriate for the business purpose it was designed for. In addition, these key model figures can help to adjust certain models by leveraging the amount of true and false positives. Some machine learning models already have meta-parameters to control the quota of true and false positives.

Model interpretability is a key for model acceptance. Section four discusses the main components of model interpretability, addressing the rising complexity[2] of the models. The classical feature importance says a lot about the dataset in the context of the question raised by the business.

Certain problems like data quality and error handling are not the first subjects that spring to mind when thinking of imbalanced data (because in certain phases, the applications produce many errors or data transfer and transformation and spoil the data quality). In the lifetime of an implementation (in most cases), the number of faulty transactions and scale of data quality issues decreases (not always before the application is put into production). Hoping for improved system integrity[3] will lead to fewer true positives[4] and so to the challenge of imbalanced data. Thus, imbalanced data will be an issue to be faced at some point.

Literature

Enzinger, P., and S. Li. 2021. "Fraud Detection Using Machine Learning Techniques. In V. Liermann, & C. Stegmann." In *The Digital Journey of Banking and Insurance, Volume II—Digitalization and Machine Learning*. New York: Palgrave Macmillan.

Guidotti, R., A. Monreale, S. Ruggieri, F. Turini, F., and Giannotti, and D. Pedreschi. 2018. "A Survey of Methods for Explaining Black Box Models." *ACM Computing Surveys*, August.

Johnson, J. M., & T. M.Khoshgoftaar. 2019. "Survey on Deep Learning with Class Imbalance." *Journal of Big Data* 6: 27.

[2] One specific aspect that is not covered in this chapter is the interpretability of deep learning models. Local surrogate models (LIME—Local Interpretable Model-Agnostic Explanations) and Shapley values offer solutions to this subject. See also Molnar (2020).

[3] Data quality increase and system error reduction.

[4] Faulty transactions.

Kaggle. 2020. *Kaggle: Open Datasets*. December 15. Retrieved from Kaggle: https://www.kaggle.com/mlg-ulb/creditcardfraud.

Klaus, B., and K. Gjergji. 2019. "A Gradient-Based Split Criterion for Highly Accurate and Transparent Model Trees." *Preprint*.

Liermann, V., S. Li, and N. Schaudinnus. 2019. "Batch Processing—Pattern Recognition." In *The Impact of Digital Transformation and Fintech on the Finance Professional,* edited by V. Liermann and C. Stegmann. New York: Palgrave Macmillan.

———. 2019. "Mathematical Background of Machine Learning." In *The Impact of Digital Transformation and Fintech on the Finance Professional*, edited by V. Liermann and C. Stegmann. New York: Palgrave Macmillan.

Mengnan Du, N. L. 2019. "Techniques for Interpretable Machine Learning." *Preprint*.

Molnar, C. 2020. *Interpretable Machine Learning—A Guide for Making Black Box Models Explainable.* Leanpub.

Shilaskar, S., and A. Ghatol. 2018. "Diagnosis System for Imbalanced Multi-Minority Medical Dataset." *Soft Computing* 23: 4789–4799.

Summary

Volker Liermann and Claus Stegmann

This final part summarizes the third volume, with all its data-related aspects including architecture patterns for streaming, special databases (like graph or document-based databases) and some specific aspects of distributed ledger and self-sovereign digital identities (SSI). Furthermore, the part summarizes the whole book series.

1 Summary: Volume III—Data Storage, Processing and Analysis

This final part summarizes the third volume, with all its data-related aspects including architecture patterns for streaming, special databases (like graph or document-based databases) and some specific aspects of distributed ledger and self-sovereign digital identities (SSI). Furthermore, the part summarizes the whole book series.

V. Liermann (✉)
ifb SE, Grünwald, Germany
e-mail: Volker.Liermann@ifb-group.com

C. Stegmann
ifb Americas, Inc, Charlotte, NC, USA
e-mail: Claus.Stegmann@ifb-group.com

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6_14

Data is important! And so is an ability to process and analyze data in the best and most efficient way at minimum cost. So far, the vision of a head of digital transformation and the vision of managers target the implementation of a data-driven business model. But is a data-driven business model the holy grail of our time? Are the Big Tech and fintech companies so successful because they are data-driven?

The success of the Big Tech and fintech companies originates from the customer-oriented approach and—this is only true for Big Tech—deep pockets allowing them to follow the customer-oriented approach with a long-term perspective. Customer-oriented approaches can leverage data to understand the customer needs and deliver at the right time and via the best channel.

The data volume has significantly increased, and there is an improved ability to access and process unstructured data as well as new and differently structured data (like connected graph databases). Text processing and the whole universe of natural language processing enable the institutes to better understand the client and get a more comprehensive picture of the client's needs and desires. Additionally, the integration of external data can provide further insights and helps by identifying the customer needs and desires.

Data protection in the form of regulations or self-sovereign digital identities (SSI)[1] is not contradictory to the customer-oriented needs and desires exploration. The connection between data protection and the identification of customer-oriented needs will stay a connection charged with tension.

In comparison to the other volumes[2] of the book series, this volume is by far the most technical. The technical view is important to understand the potential and the ways to leverage the technology, which is one of the key skills for successful digitalization.

The major benefits of the new technological innovations are primarily found in outside digitalization. The use cases in inside digitalization are already clear. They can improve the process, reduce the workload for the internal departments and reduce costs.

Data processing will improve with the help of streaming technology and the new architecture patterns (Lambda, Kappa and Delta Lake). The ability to store and handle mass data is at hand and special database types—like in-memory databases (IMDB)—can speed up mass data handling. The variety of frameworks providing models for analyzing data is overwhelming and completes the toolset (see [Liermann, Overview Machine Learning and Deep Learning Frameworks 2021] for details).

[1] Which is definitely more customer-oriented.

[2] Volume I—"Disruption and DNA" and Volume II—"Digitalization and Machine Learning".

The tools are there, but they have to be put in the right business context to leverage them. The book series "The Digital Journey of Banking and Insurance" with the two additional volumes can provide support for a successful institute-specific digital journey.

2 Summary: The Digital Journey of Banking and Insurance

Digital transformation is an irreversible process, already started by most of the institutes. Even those who claim to be not so affected by digital transformation (like corporate banking) already entered this one-way street a long time ago. "Our business is so special, there is no alternative to us" sounds like whistling in the dark. Open banking will enable existing and new market participants to compose the most suitable solution for niche market segments and at scale.

Institutes will differ when it comes to the speed they choose on the digitalization path. It is not clear that an early start will guarantee and define the winner (or the group of survivors). An institute that sets up a highly advanced technology infrastructure but fails to bring technology and business together (missing the opportunity to leverage technology) will crash out in the middle of the race (and will not deliver value to the customer). Culture and the ability to change are very important for a successful digital journey that generates results.

Some institutions still puzzle over the important steps to take. A wise selection in the periodization is required, so that the one-way street does not turn into a dead end eventually. The journey has to be an individual one. The individual way will always be driven by the institutes' strengths and the customer alignment most institutes have. Customer needs and desires can and will change. To stay customer-oriented and customer-aligned, the institute must stay close to the customer (otherwise the institute enters the mass of homogeneous products and economies of scale).

Data and personal touch points (closeness to the customer) will allow an understanding of the needs and how these needs can be met. Deep and comprehensive data analysis can help to understand the customer not only at a point in time but also in a period of time. This means identifying the customer journey, either by data or by expert judgement, and determining the client's situation. This is mapped to the different customer journeys that, at a certain point in the journey, will demand a solution or a set of solutions served by the institution to the customer. The time dimension in this predictive case and model generates the need for advanced models, raises the data volume and causes challenges in the data processing. The time dimension brings complexity, but it can help to identify customer opportunities early on, and at best earlier than the competition.

When speaking about leveraging technology, there is always a target (improving processes, reducing costs in inside digitalization or getting closer to the customer and understanding the needs better to sell more). The impressive ideas originate from thinking outside the box, leaving the pattern with which we perceive the world. An organization that can establish a culture that lets people think outside the box, go new ways in understanding the customer and be able to use technology in different contexts than what it was designed for: such an organization is well-prepared for the changes to come.

Literature

Liermann, Volker. 2021. "Overview Machine Learning and Deep Learning Frameworks." In *Volume III: Data Storage, Processing, and Analysis—The Digital Journey of Banking and Insurance*, edited by Volker Liermann and Claus Stegmann. New York: Palgrave Macmillan.

Index

V. Liermann and C. Stegmann (eds.), *The Digital Journey of Banking and Insurance, Volume III*, https://doi.org/10.1007/978-3-030-78821-6