DB

Foto: kanver, Kirsty Pargeter - fotolia

Towards Enabling Deletion in Append-Only Blockchains to Support Data Growth Management and GDPR Compliance

Hyperledger Fabric Architecture WG - November 18th 2020

DB Systel GmbH | Dr. Michael Kuperberg | Blockchain and DLT Solutions | 2020-11-18

D_BLOCKCHAIN

DB Systel

Digital bewegen. Gemeinsam.

Disclaimer / Legal Notice

All statements in this slide deck are to be considered as non-binding information.
They reflect the opinion of the presenter and are in no way statements made on behalf of the
Deutsche Bahn AG, DB Systel GmbH or any of the companies which are part of the DB holding.

Any mention of a product or of a trademark is not to be understood as a recommendation / endorsement / valuation or as a judgement. Trademarks are owned by the respective holders.

No guarantee is given that the provided information is correct, complete or up-to-date.
Distribution or copying of contents from this presentation (or parts thereof) is only allowed based on
written permission from the author and copyright holder: Michael.Kuperberg@deutschebahn.com

Alle Darstellungen in diesem Foliensatz sind als unverbindliche Information zu verstehen. Sie geben lediglich die Meinung des Autors wieder und sind nicht als Aussage im Namen der Deutschen Bahn AG, der DB Systel GmbH oder eines Unternehmens der DB-Holding zu verstehen.

Die Erwähnung eines Produktes oder einer Handelsmarke ist nicht als Empfehlung oder als Bewertung zu verstehen. Die jeweiligen Trademarks gehören ihren Eigentümern.

Für Korrektheit, Aktualität und Vollständigkeit der Information wird keine Gewähr übernommen. Vervielfältigung oder Reproduktion von (Teil-)Inhalten oder vom Vortrag ausschließlich mit schriftlicher Genehmigung des Autors und Copyright-Inhabers: michael.kuperberg@deutschebahn.com.

Deutsche Bahn AG / DB Systel GmbH ←······→ Blockchains/DLTs

DB Systel GmbH:

- full-service IT provider with >1 billion € revenue
- 4400+ employees (Frankfurt, Berlin, Erfurt, UK)

Since 2016: pursuing the "Blockchains & Distributed Ledger Technologies" topic in a systematic way

Why do we deal with blockchains?

- Today's public transportation is a network of providers with many opportunities in backend integration
- BRCS: Blockchain-based Rail Control System
- SSI: Self-Sovereign Identity

Copyright: Deutsche Bahn AG / Rico Emersleben https://mediathek.deutschebahn.com

Problem Statement and Research Goals

- Blockchains (append-only WORM semantics) do not support deleting or overwriting data in confirmed blocks - however, **many industry-relevant use cases require the ability to delete on-chain data**
- Especially important when PII is stored (GDPR!) or when data growth has to be constrained
- Existing attempts to reconcile these contradictions compromise on core blockchain paradigms
  - some include backdoor-like approaches such as central authorities with elevated rights
  - others use specialized chameleon hash algorithms in chaining of the blocks

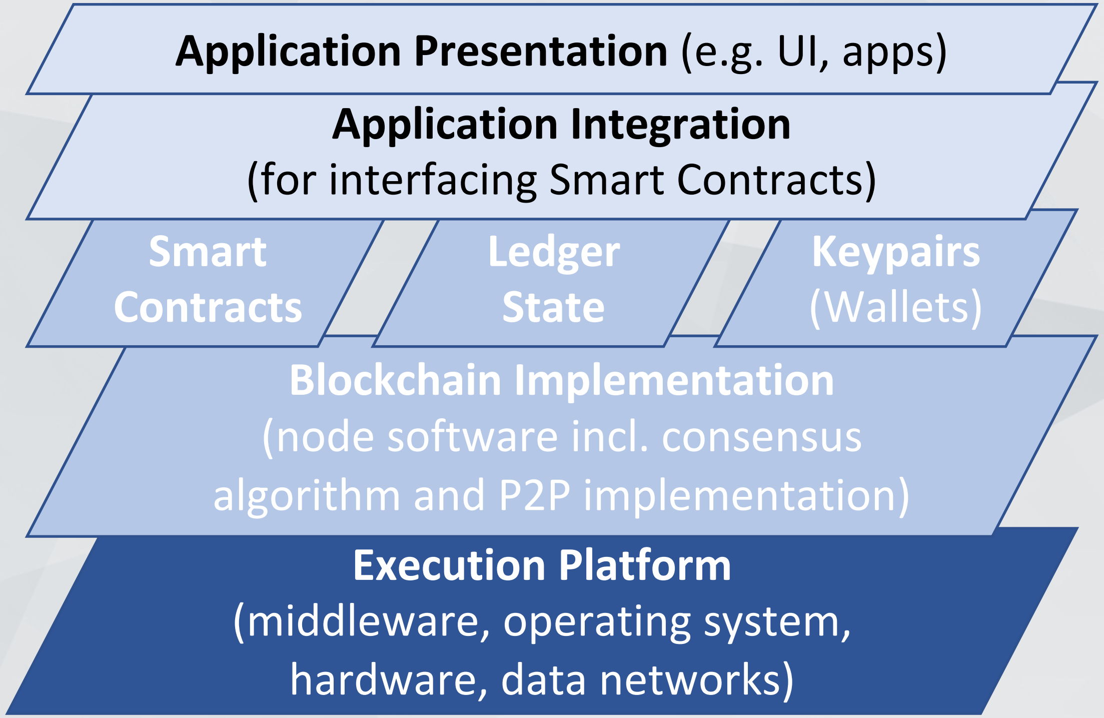
- Our contribution: a novel architecture for the blockchain ledger and consensus, using **a tree of context chains with simultaneous validity**
  - A context chain captures the transactions of a closed, well-defined group of entities and persons
  - → Context isolation enables consensus-steered deletion of an entire context without side effects

- This architecture supports truncation, data rollover and separation of concerns, helps fulfill GDPR regulations but is also different from sidechains and state channels

Why is there no deletion in current blockchain/DLT protocols and products?

- Cryptocurrencies (Bitcoin, Ethereum mainnet et al.) have >100GB of history, but offer light clients
- Enterprise-grade blockchains: scalability is still the „elephant in the room“: known but postponed
  - Going consortial rather than public
  - Workloads often rather limited
  - Projects store PII off-chain (only hashes / fingerprints are stored on-chain, if any)
  - Hoping that storage costs decrease fast
  - Hoping that sharding/private data helps
  - Steering around 10-year-storage scenarios

Deletion-affected blockchain layers



Application Presentation (e.g. UI, apps)

Application Integration (for interfacing Smart Contracts)

Smart Contracts

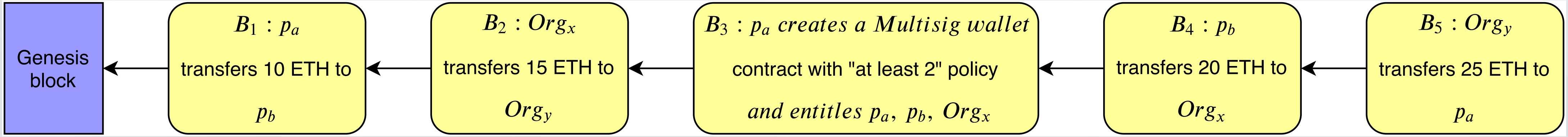
Ledger State

Keypairs (Wallets)

Blockchain Implementation (node software incl. consensus algorithm and P2P implementation)

Execution Platform (middleware, operating system, hardware, data networks)

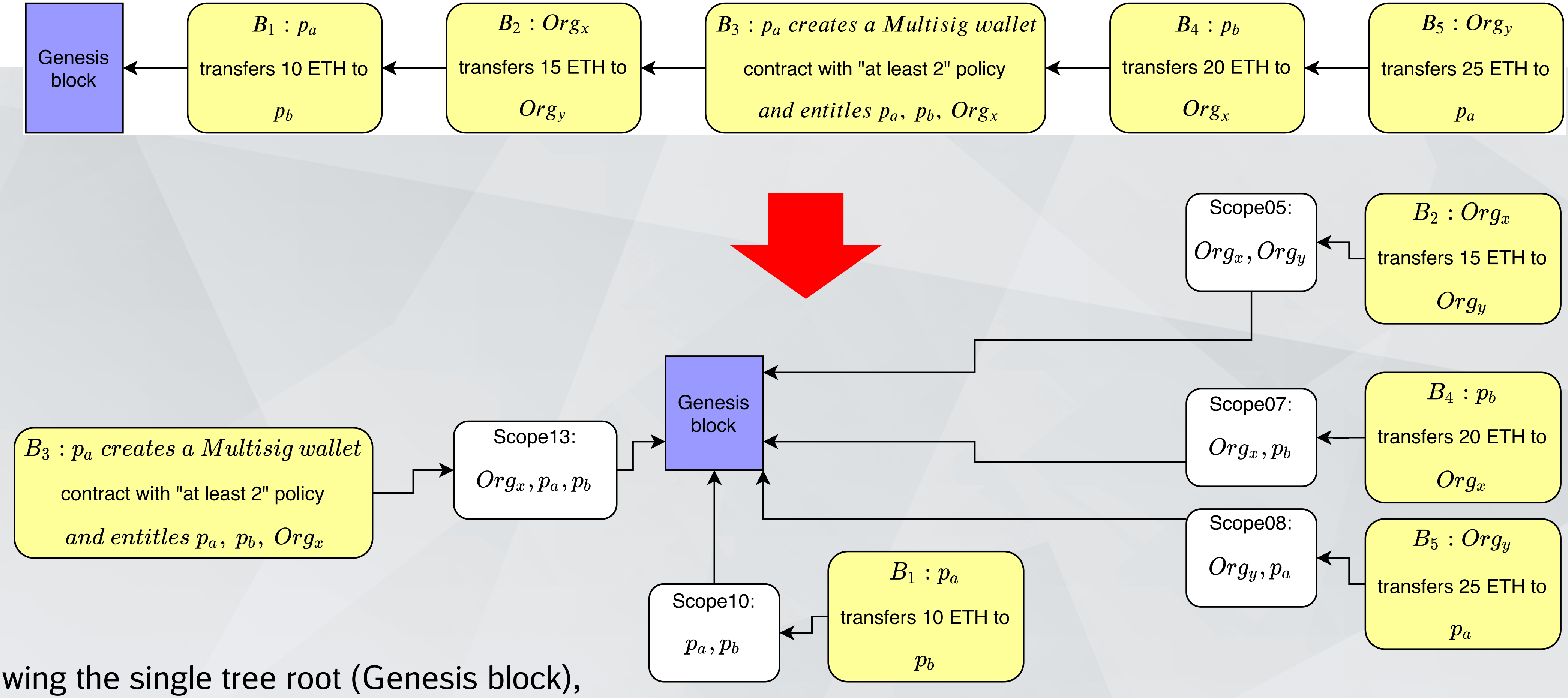
Non-Erasability through Intermingling of Contextes


Genesis block
B_1 : p_a transfers 10 ETH to p_b
B_2 : Org_x transfers 15 ETH to Org_y
B_3 : p_a creates a Multisig wallet contract with "at least 2" policy and entitles p_a, p_b, Org_x
B_4 : p_b transfers 20 ETH to Org_x
B_5 : Org_y transfers 25 ETH to p_a

- Linear block**chain** mixes transactions concerning persons p_a and p_b (and organizations Org_x and Org_y)
- Deleting a transaction/block breaks the hash-pointer-based chaining
- All blocks are needed by all network participants to verify integrity

- Remedy (our contribution): create **context chains** for each (business) relationship, i.e. separate business concerns
  - results in a tree (see next slide), where tree branches do not conflict (i.e. this is not forking!)
  - continue to verify each new (proposed) transaction against overall information (in all tree branches)

Erasability through Separation of Contexts (1)



Genesis block

$B_1 : p_a$ transfers 10 ETH to p_b

$B_2 : Org_x$ transfers 15 ETH to Org_y

$B_3 : p_a$ *creates a Multisig wallet* contract with "at least 2" policy *and entitles* $p_a,\ p_b,\ Org_x$

$B_4 : p_b$ transfers 20 ETH to Org_x

$B_5 : Org_y$ transfers 25 ETH to p_a

Scope05: Org_x, Org_y

Scope07: Org_x, p_b

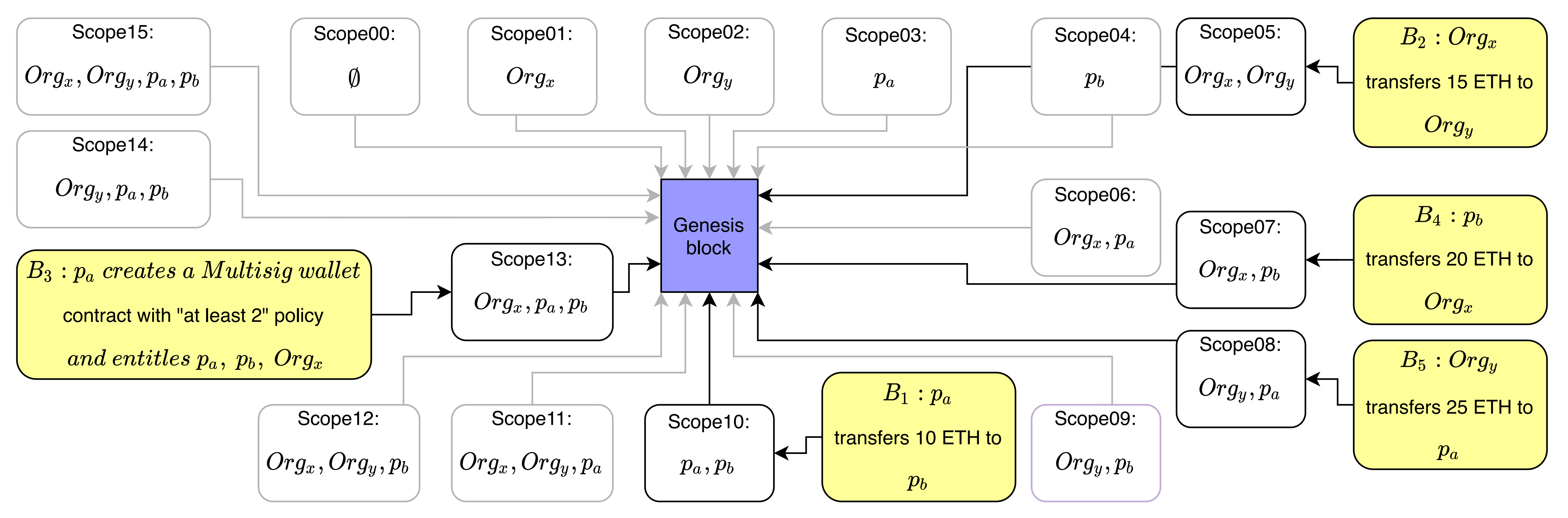
Scope08: Org_y, p_a

Scope10: p_a, p_b

Scope13: Org_x, p_a, p_b

- Following the single tree root (Genesis block), each branch is headed by a scope definition
- Each tree branch is linear (no further branching)
- Each transaction is sorted unambiguosly into one single branch

Erasability through Separation of Contexts (2)


Scope15: Org_x, Org_y, p_a, p_b
Scope00: $\emptyset$
Scope01: Org_x
Scope02: Org_y
Scope03: p_a
Scope04: p_b
Scope05: Org_x, Org_y
$B_2 : Org_x$ transfers 15 ETH to Org_y
Scope14: Org_y, p_a, p_b
Genesis block
Scope06: Org_x, p_a
Scope07: Org_x, p_b
$B_4 : p_b$ transfers 20 ETH to Org_x
$B_3 : p_a$ creates a Multisig wallet contract with "at least 2" policy and entitles p_a, p_b, Org_x
Scope13: Org_x, p_a, p_b
Scope08: Org_y, p_a
$B_5 : Org_y$ transfers 25 ETH to p_a
$B_1 : p_a$ transfers 10 ETH to p_b
Scope12: Org_x, Org_y, p_b
Scope11: Org_x, Org_y, p_a
Scope10: p_a, p_b
Scope09: Org_y, p_b

- Only create scopes on-demand (greyed-out scopes would be created when needed)
  - In the example with 2 Persons and 2 Organizations, we could have 2^4=16 contextes *at most*
  - Yet the additional space for context roots (potentially exponential) becomes negligible when the number of per-context transaction grows

State-of-the-Art and Related Work (1)

References numbered as in conference paper (which discusses further related work).

- **Chameleon hashes** per se are studied in a number of publications, such as [13] or [14]
  - [11]: “chameleon hash” functions as the foundation for rewritable and redactable blockchains; Accenture’s announcement [12] builds on these publications of Ateniese et al., as do the US patents US9959065B2 and US9967088B2
  - However, chameleon hashes as in [11] **involve trapdoor-like elevated rights which may undermine trust and decentralization**, it also requires the replacement of core cryptographic routines in a given blockchain protocol; in contrast to that, our work introduces erasability without introducing rewriteability at individual block level
  - In [15], Huang et al. extend chameleon hashes to the domain of Industrial IoT; they utilize TCH (Threshold Chameleon Hashes) and ASCS (Accountable-and-Sanitizable Ch. Signatures)
  - Other publications (e.g. [16], [17]) apply the chameleon hash concept to other blockchain products and domains
  - In [18], Lee et al. use **truncated hash values** for transaction-level modifiability, using sidechains for the transaction modification process
- [19]: Florian et al. introduce **functionality-preserving local erasure** (FPLE) for UTXO-based cryptocurrencies such as Bitcoin
  - each node operator decides on its own (and for individually selectable transactions) whether the selected UTXO outputs are stored on the owned node or not
  - despite being called “functionality-preserving”, the approach has some consequences on protocol level: unconfirmed incoming transactions that reference erased data are considered invalid

State-of-the-Art and Related Work (2)

References numbered as in conference paper (which discusses further related work).

- In [20], Deuber et al. devise a formalized approach for redaction of a permissionless blockchain such as Bitcoin
  - a modified block structure and a voting round on the transaction that proposes the editing of a mined transaction
  - no provisions for an objecting user to oppose (or to veto) a redaction are discussed or proposed, though
  - the approach has an additional limitation: the redaction policy approach “does not allow monetary transactions to be edited”
- In [21], Puddu et al. explore an approach (µchain) for making blockchains mutable, i.e. editable/rewriteable
  - µchain is described as as applicable to PoA, PoW and PoS blockchains
  - a prototypic implementation based on Hyperledger Fabric is hosted on GitHub
  - the major difference between our approach and µchain is that **µchain cannot delete or mutate individual transactions or blocks which have already been persisted**
  - instead, µchain introduces compensation-like transactions that are appended to the blockchain, but *appear* to retroactively change a specified transaction that happened in the past
- **Some distributed ledgers come without append-only semantics** (i.e. without the blockchain-typical WORM pattern)
  - such as BigchainDB [24], which relies on consensus to prohibit rewrites
  - still, even such non-WORM DLTs/blockchains still do not provide facilities for controlled, consensus-enforced deletion

Conclusions and Future Work

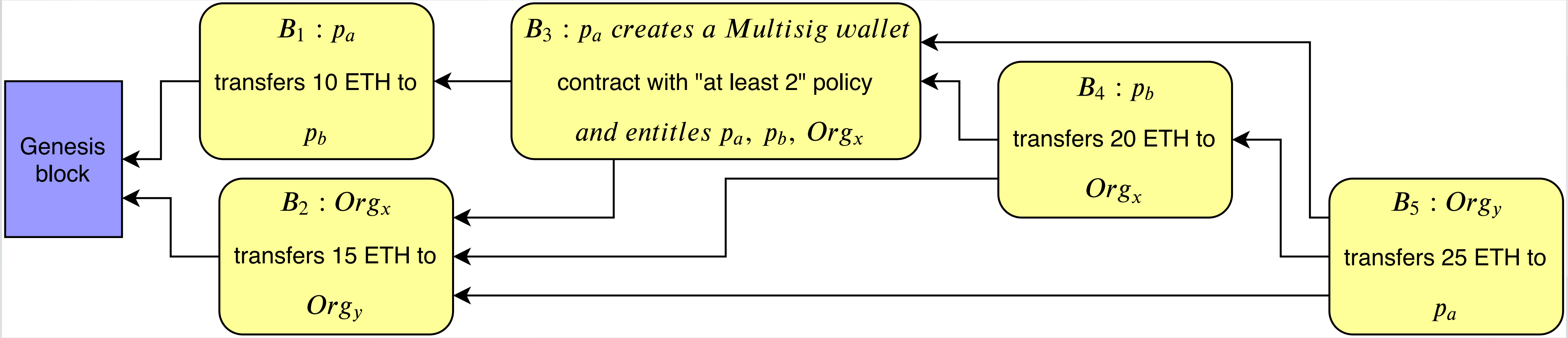
- **A novel solution for adding deletion capabilities to append-only (WORM) blockchains:**
  - using the “context chain” architecture pattern, a separation of concerns leads to a **non-linear ledger structure** with accompanying, clear rules of transaction placing
  - Context chains are complemented by **consensus-driven decision making for deletion**, ensuring that deletion is not endangering auditability and trustworthiness of the decentralized blockchain/ledger
  - Extended design aspects include ledgers that do not use linear-only chaining, non-cooperating (or absent) network participants and the effects of non-absolute majorities on the erasability of data

- **Unlocked opportunities**: space savings, GDPR compliance, business needs: rollover/balancing and truncation

- **For future work**, we want to formally express the persistence and liveness properties of the proposed solution
  - Planned implementation targets a major enterprise-grade DLT (e.g. Hyperledger Fabric, R3 Corda, etc.)
  - Likewise, we plan to study erasability in ledgers which are used as foundations for self-sovereign identity, such as Sovrin’s Plenum ledger

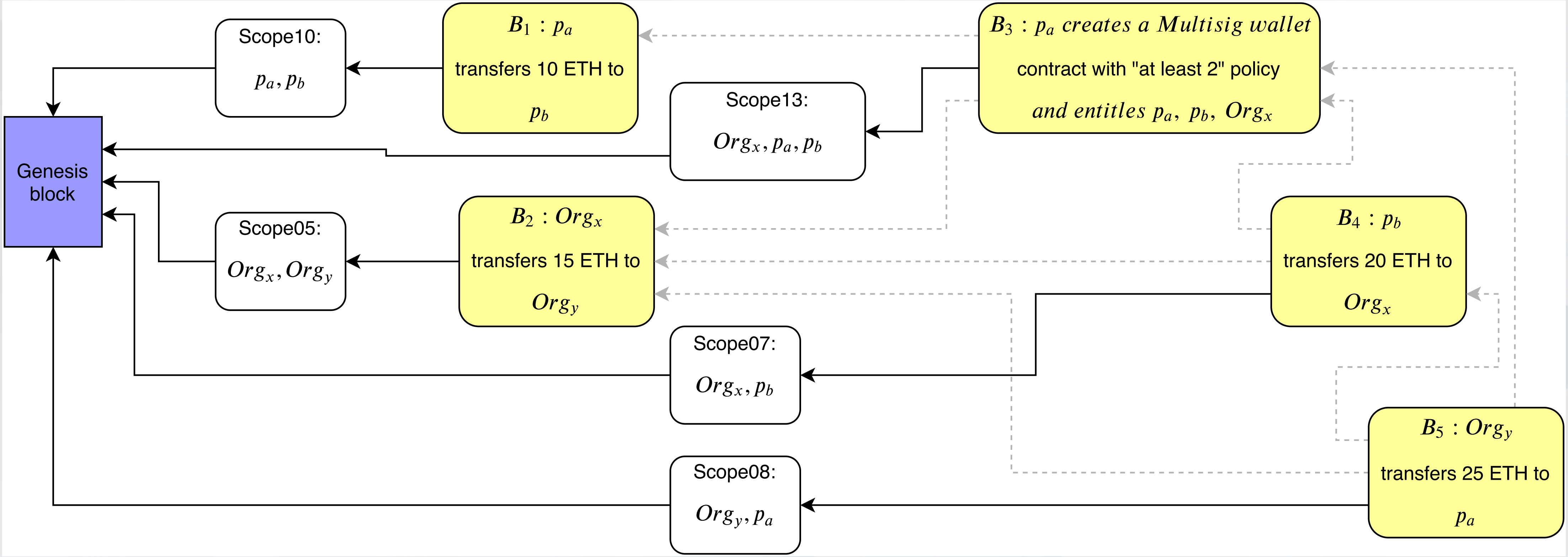
- Additionally, we plan to measure the **performance and scalability** of our approach, including the consensus phase; we also intend to offer a **security analysis** once a working implementation is available

Erasability for Non-Linear Hash-Concatened DAGs (1)


Genesis block
B_1 : p_a transfers 10 ETH to p_b
B_2 : Org_x transfers 15 ETH to Org_y
B_3 : p_a creates a Multisig wallet contract with "at least 2" policy and entitles p_a, p_b, Org_x
B_4 : p_b transfers 20 ETH to Org_x
B_5 : Org_y transfers 25 ETH to p_a

- Same transactions as before, but now linked differently (incl. blocks with 2 predecessors and with 3 predecessors)
- Note that block timestamping provides a monotonous block ordering even in non-linear DAG ledgers

Erasability for Non-Linear Hash-Concatened DAGs (2)



Scope10: p_a, p_b
$B_1 : p_a$ transfers 10 ETH to p_b
$B_3 : p_a$ *creates a Multisig wallet* contract with "at least 2" policy *and entitles* p_a, p_b, Org_x
Scope13: Org_x, p_a, p_b
Genesis block
Scope05: Org_x, Org_y
$B_2 : Org_x$ transfers 15 ETH to Org_y
$B_4 : p_b$ transfers 20 ETH to Org_x
Scope07: Org_x, p_b
$B_5 : Org_y$ transfers 25 ETH to p_a
Scope08: Org_y, p_a

- Same transactions as before, same scope definitions as before
  - but now linked differently (incl. blocks with 2 predecessors and with 3 predecessors)
  - grey-dashed links are re-enacting the original DAG’s linking

Thank you!

Work Email: michael.kuperberg@deutschebahn.com

Twitter of working group: https://twitter.com/DB_Blockchain

Homepage of working group: https://blockchain.deutschebahn.com

LinkedIn:
https://www.linkedin.com/in/michael-kuperberg-8a612625/

Private Homepage (incl. publication list & talk announcements):
https://www.michaelkuperberg.com

Dr. Michael Kuperberg
Chief Blockchain Architect

Blockchain and DLT Solutions

Michael.Kuperberg@deutschebahn.com
DB Systel GmbH
Jürgen-Ponto-Platz 1
60329 Frankfurt am Main, Germany
https://www.dbsystel.de