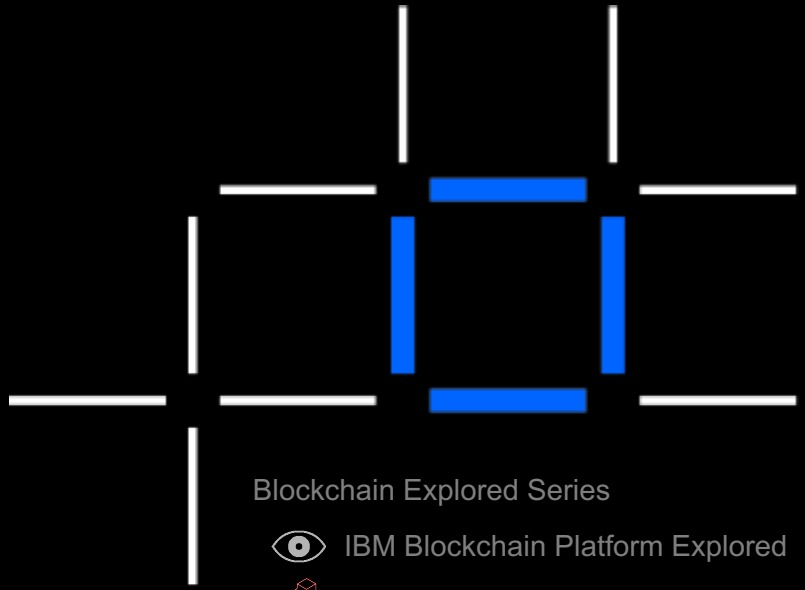
# Modeling Blockchain Applications

Turning business concepts into technical concepts

*Austin Grice*
*austin.grice@ibm.com*



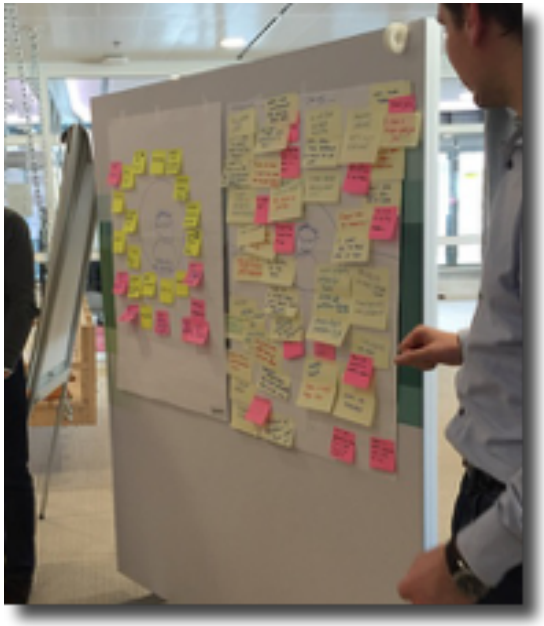
Blockchain Explored Series

- IBM Blockchain Platform Explored
- **Modeling Applications**
- Architectures Explored
- Fabric Explored
- What's New in Tech

# Blockchain topics

- Consider the topics we've discussed for blockchain solutions so far:
  - The **business problem** we're trying to solve
  - The **participants** involved (users and organizations)
  - The **assets**
  - The **transactions**, underpinned by **contracts**

- The goal now is to move these topics into to a machine readable form and eventual deployment to a blockchain system



# Example: Commercial Paper


## Business Problem?

- Commercial paper is a means of providing short term financing to companies
- Trust requirement and well-defined business network make a good fit for blockchain

## Participants?

- MagnetoCorp (Issuing organization)
- Matt Lucas (MagnetoCorp employee)
- “the bearer” (could be many of these)

## Assets?

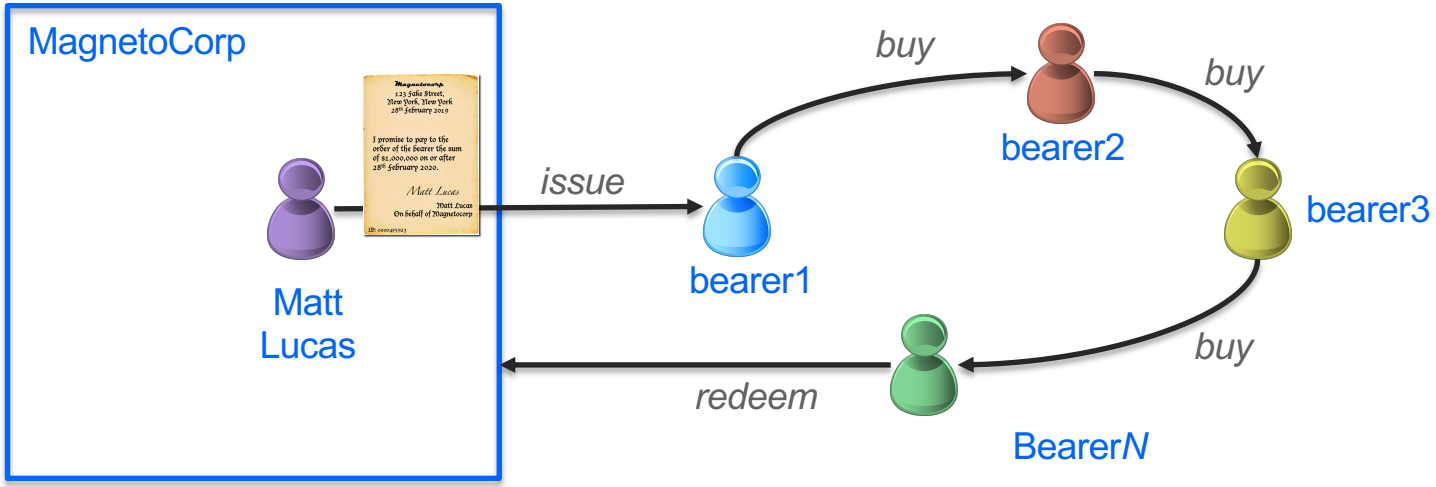
- The Commercial Paper (!)
- $1,000,000

## Transactions?

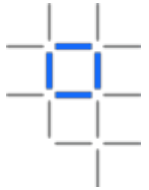
- Issue Buy Redeem

# Commercial Paper: Transaction Lifecycle

(omitting cash flows)



# Example: Commercial Paper



Q: How do we get from these business concepts to something that can run on a blockchain?

A: We will *model* them

**Business Problem?**

- Commercial paper is a means of providing short term financing to companies
- Trust requirement and well-defined business network makes a good fit for blockchain

**Participants?**

- Magnetocorp (Issuing organization)
- Matt Lucas (Magnetocorp employee)
- “the bearer” (could be many of these)

**Assets?**

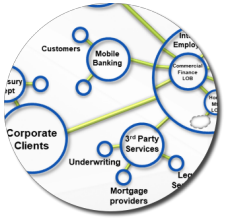
- The Commercial Paper
- $1,000,000

**Transactions?**

- Issue  Buy  Redeem

# Modeling is the mapping of business concepts into technical concepts…





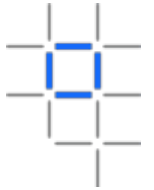

| Assets | Contracts | Transactions | Business Networks | Participants |
|---|---|---|---|---|
| Data structures in a pre-agreed format | Algorithm to modify asset state | Single invocation of a contract's algorithm | Computer network topology (c.f. internet) | Digital certificate for each user/organization |

- Models don't need to be *complete*, but they need to be *sufficient* to solve the problem at hand
  - e.g. You don't need to model each cylinder of an engine if you're tracking the overall owner of a car

# Modeling users and organizations

- Every organization, user and system component has an **identity** in the blockchain network
  - Used for identifying actors in the network, signing and encrypting information
- There are standard ways of expressing identity (e.g. X.509 certificates)
- Things to consider:
  - Who is a user in the blockchain system
  - How certificates are issued (and revoked)
  - The relationship between users and organizations, and between organizations

# (Simplified) transaction process



1. Each administrator **deploys** chaincode to peers in the network

2. The end-user application **connects** to a gateway peer

3. The application **queries** the available chaincodes on the peer

4. The application **invokes** an available chaincode with a set of input parameters

5. The blockchain network **executes** the chaincode, agrees the output and updates the ledger/world state on all peers

6. The peer **notifies** the application is notified that the transaction has been completed

```
const gateway = new Gateway();
const wallet = new FileSystemWallet('./WALLETS/wallet');
try {
  await gateway.connect(ccp, {
    identity: 'admin',
    wallet: wallet
  });

  const network = await gateway.getNetwork('market1234');
  const contract = await network.getContract('commercial-paper');

  // issue commercial paper
  const paper = await contract.submitTransaction('issue', 'ibm', '1000000', '2019-03-31');

} catch (error) {
  console.log(error);
} finally {
  gateway.disconnect();
}
```

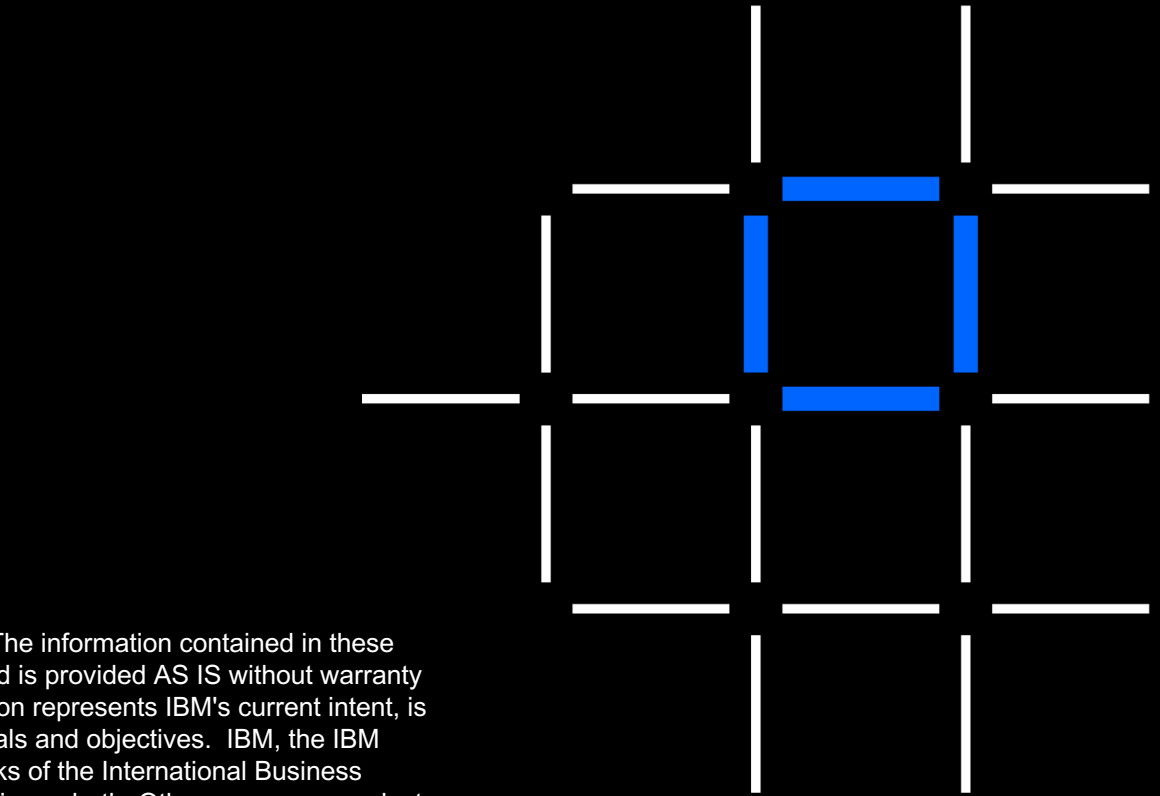
# Thank you

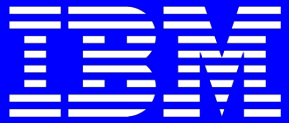
IBM **Blockchain**

www.ibm.com/blockchain

developer.ibm.com/blockchain

www.hyperledger.org




IBM