

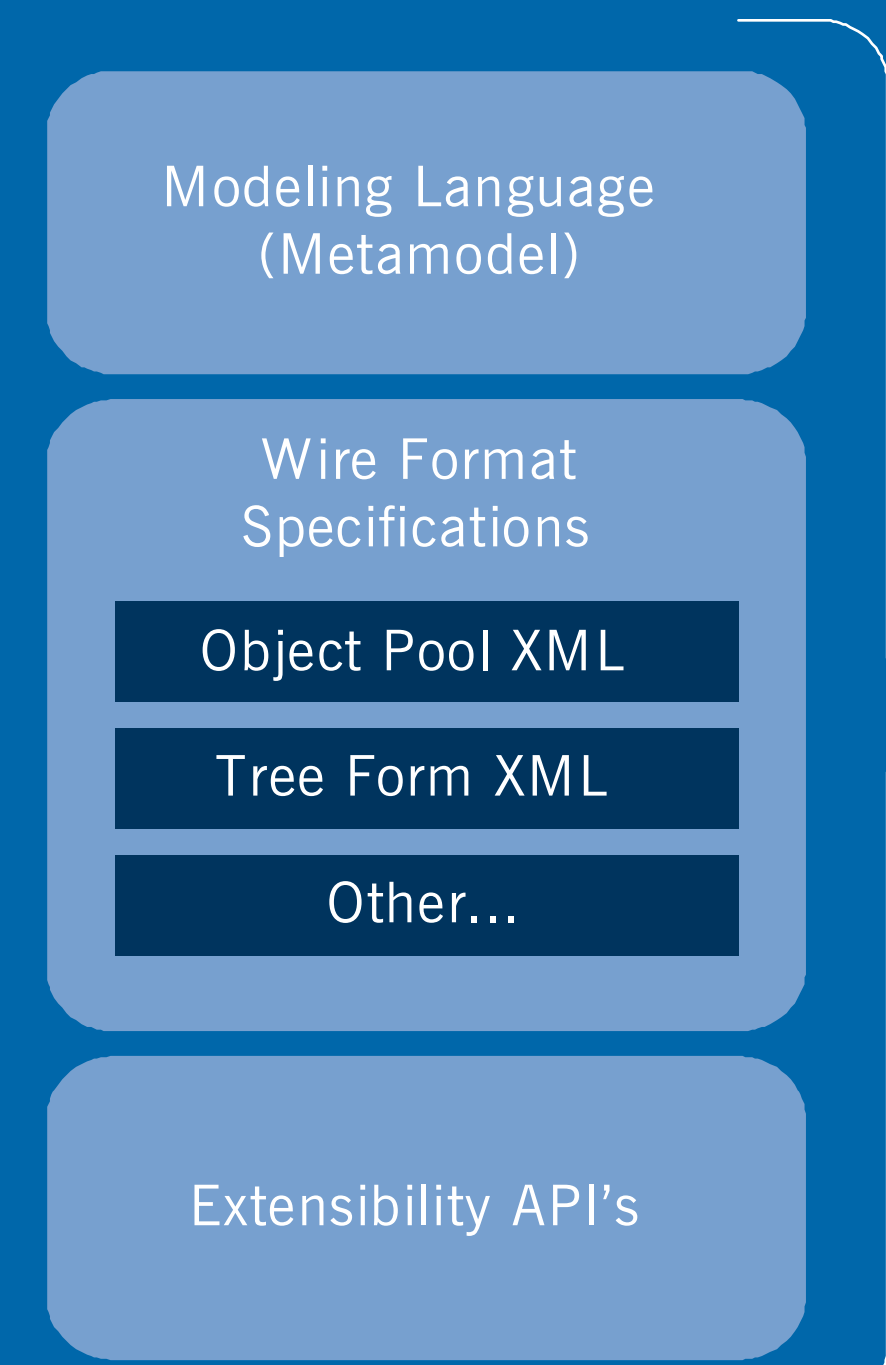
MODeX

Model-Oriented Data eXchange

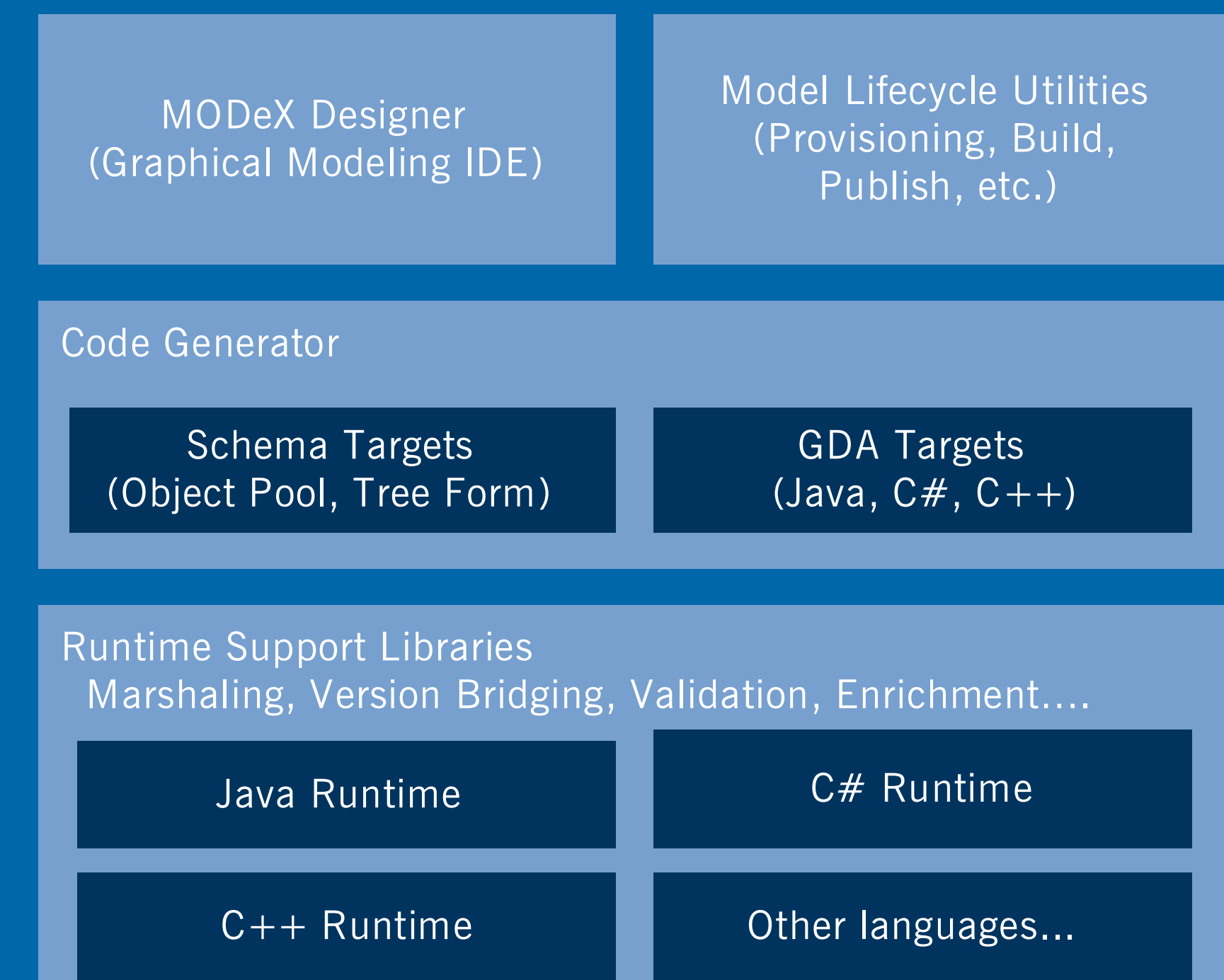
Challenge: Create a domain-specific modeling language for enterprise data integration

MODeX is a model-driven solution for Enterprise SOA. Where general-purpose SOA and MDD toolsets focus on developer productivity, MODeX is additionally concerned with enterprise-scale issues like versioning, standardization, and reducing long-term barriers to system integration.

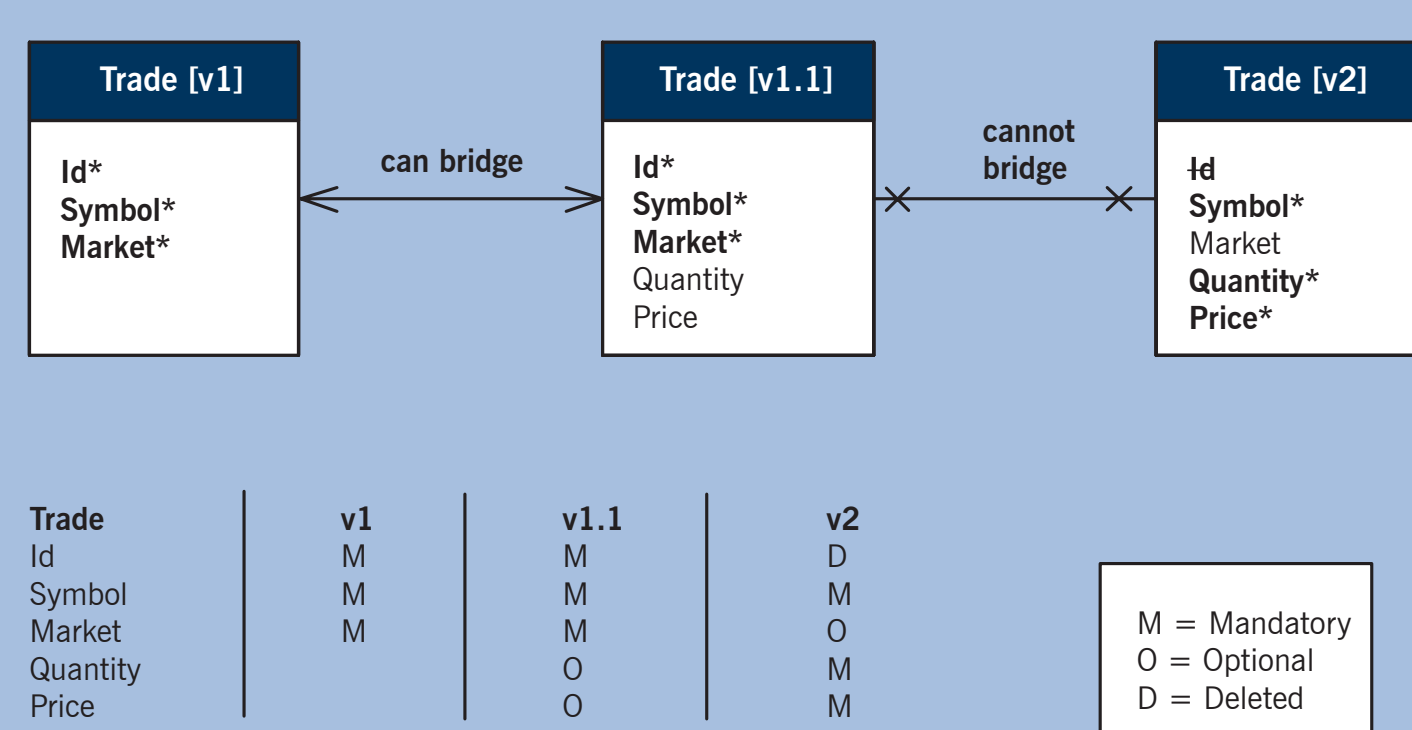
Specifications



Core Technologies



Versioning



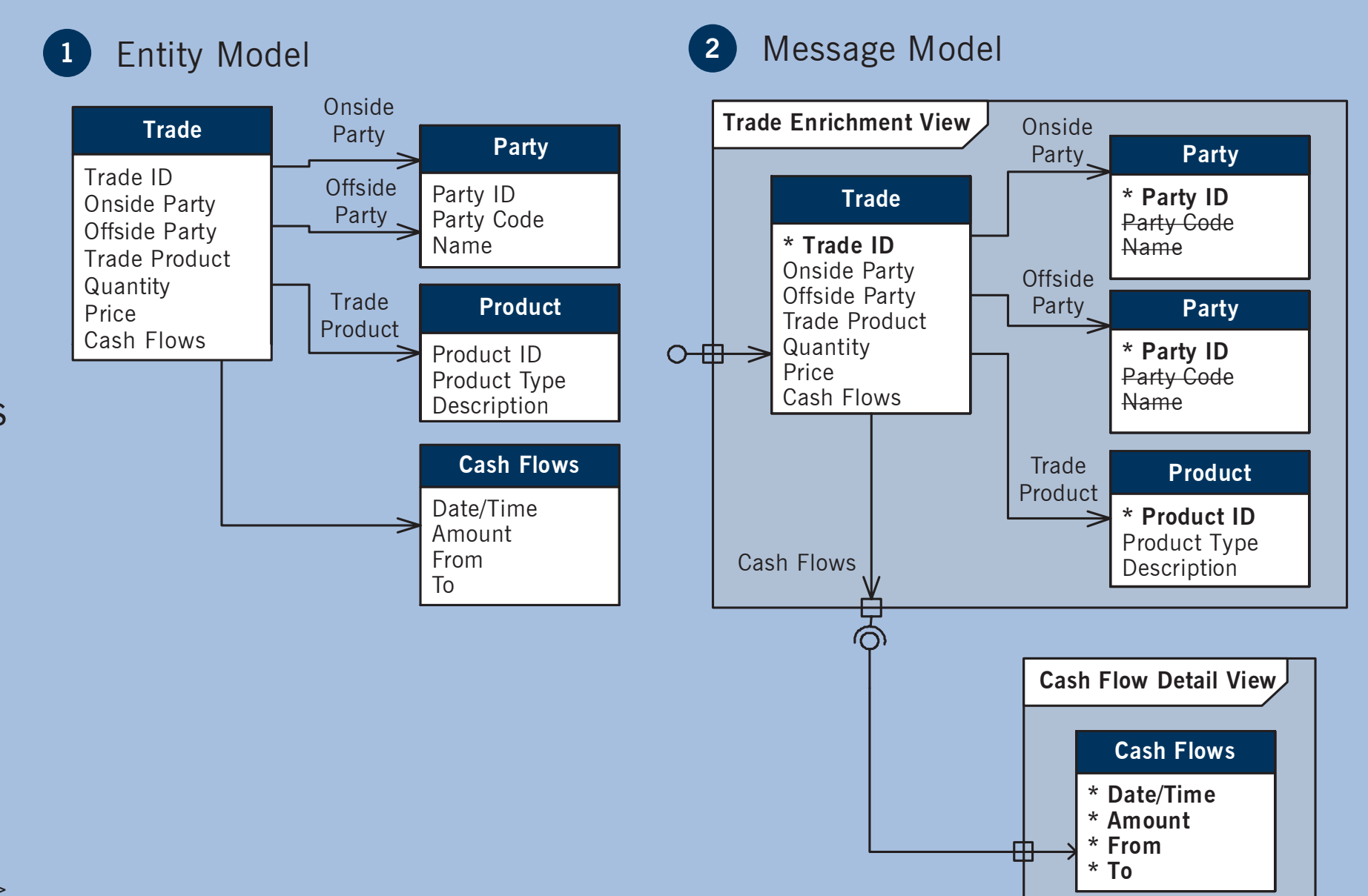
Minor versions can add optional fields. Runtimes support minor version bridging, and unknown field data pass-through.

Major versions can completely alter fields: add, remove, delete, change optional/mandatory status. Does not work with existing contracts, but tooling can allow smoother major version migration.

Modeling tool maintains multiple copies of Entity for every version, minor or major.

Message Contracts

- Building blocks for message contracts
- Specified in terms of the underlying entity model
- Allows presence constraints: optional, required, excluded
- Extensible to allow additional constraints

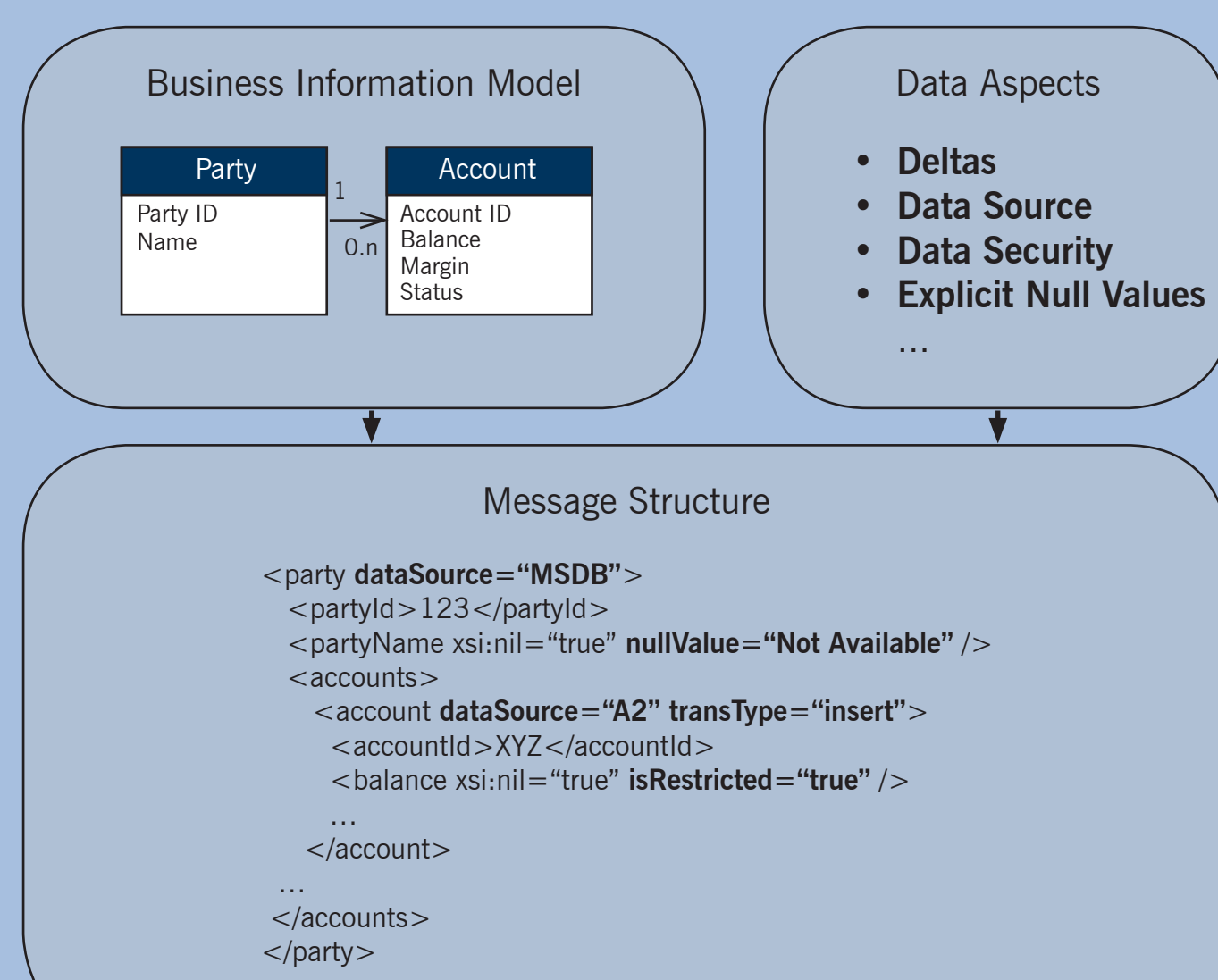


3 XML Message

```
<Trade
  tradeId="123" >
  <InsideParty partyId="P1" />
  <OffsideParty partyId="P2" />
  <Product productId="xyz"
    productType="bond" />
  <CashFlows
    dateTime="2007-08-24 09:25:35"
    amount="982734987" from="P1" to="P2" />
  <CashFlows
    dateTime="2007-08-26 11:02:15"
    amount="789342" from="P1" to="P2" />
</Trade>
```

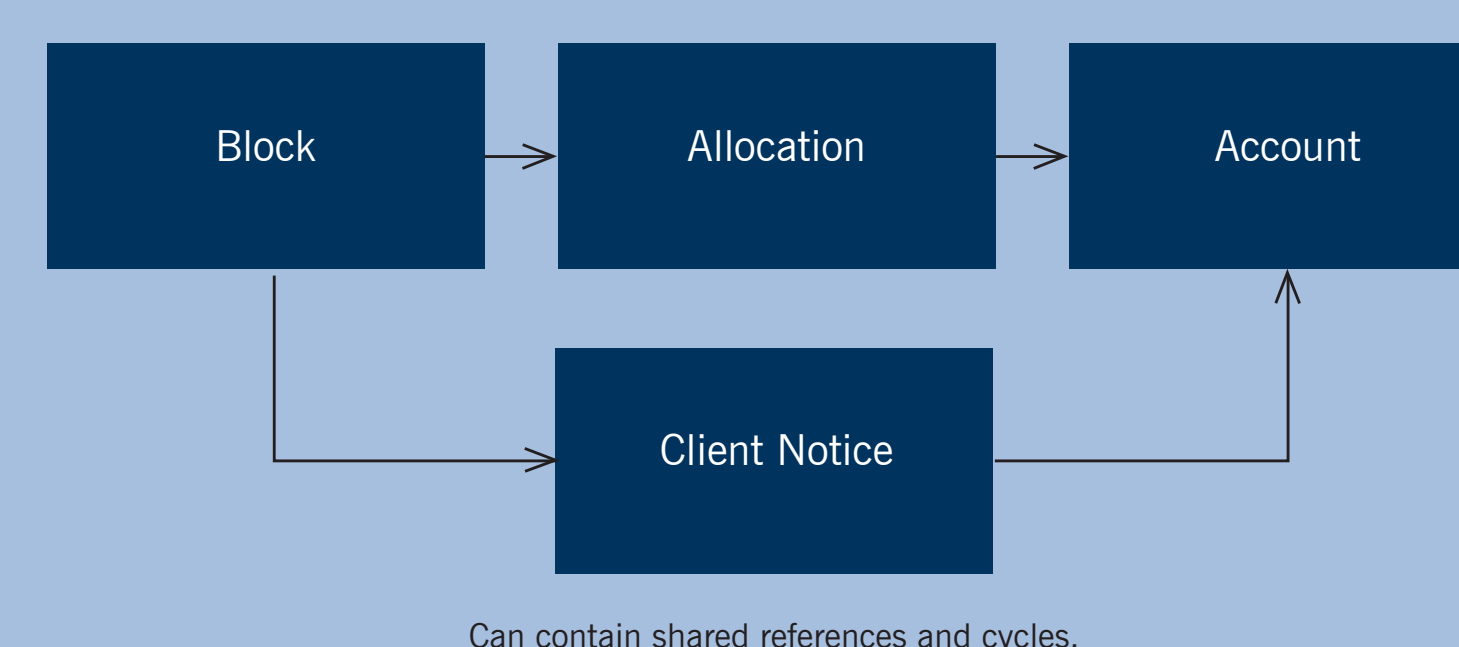
Data Aspects

- A way of specifying cross-cutting concerns in messaging
- Allows introduction of metadata without changing the underlying entity model
- Extensible to allow new aspects to be woven in as necessary



Full-Fidelity Messaging

1 Object Graph



2 XML

```
<Block blockId="8927340">
  <Allocation allocationId="982734">
    <Account modex:objectId="P1"
      accountId="12349037"
      accountCode="ABC"
      type="DEF" />
  </Allocation>
  <ClientNotice date="2007-09-17"
    status="sent">
    <Account modex:objectId="P1" />
  </ClientNotice>
</Block>
```

Shared objects detected by marshaller, and assigned unique ID. Unmarshaller will reconstruct object graph with full fidelity.

3 API

```
Block b = BlockMsg.unmarshalFromDocument(doc);
ClientNotice cn = b.getClientNotice();
String accountCode =
  cn.getAccount().getAccountCode();
```

Rich Validation Rules

- Ability to extend validation beyond what schema allows.
- Templates for common validation patterns:
 - If Field 1 = X, then field 2 must = A, B or C
 - If Field 1 is populated, Fields 2 and 3 must also be populated.
 - ...
- Integration with JBoss Rules for open-ended expression of complex rules

Federated Model Repository

- Allows loosely coupled versioning and lifecycle management across model domains.
- Each domain chooses independently which release of each model it works with.
- Domain owners coordinate on major model changes

Data Store Integration

- Semi-Static Enumerations:** a primitive field whose values are validated against a database or data service.
- Enrichment On Demand:** dynamic lookup and population of values as object graph is traversed, and as values are requested