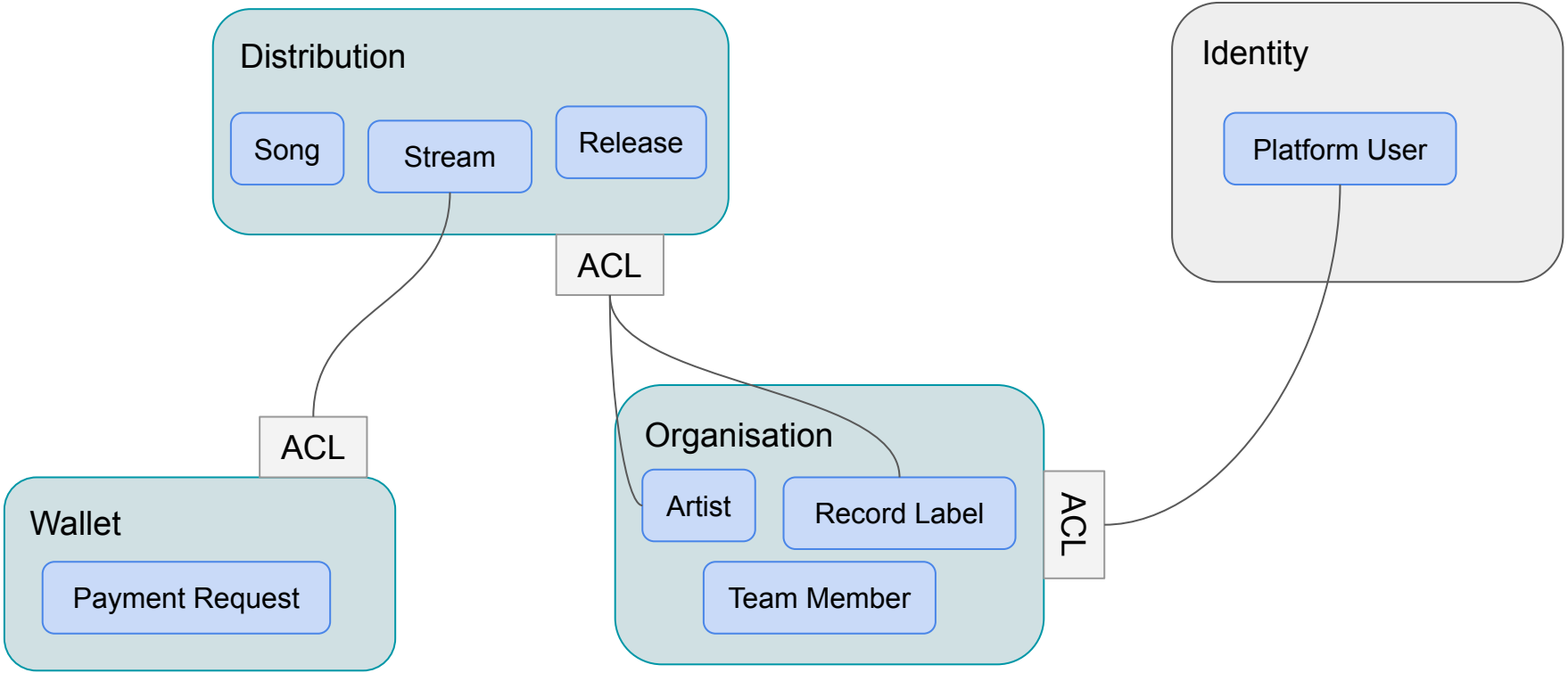
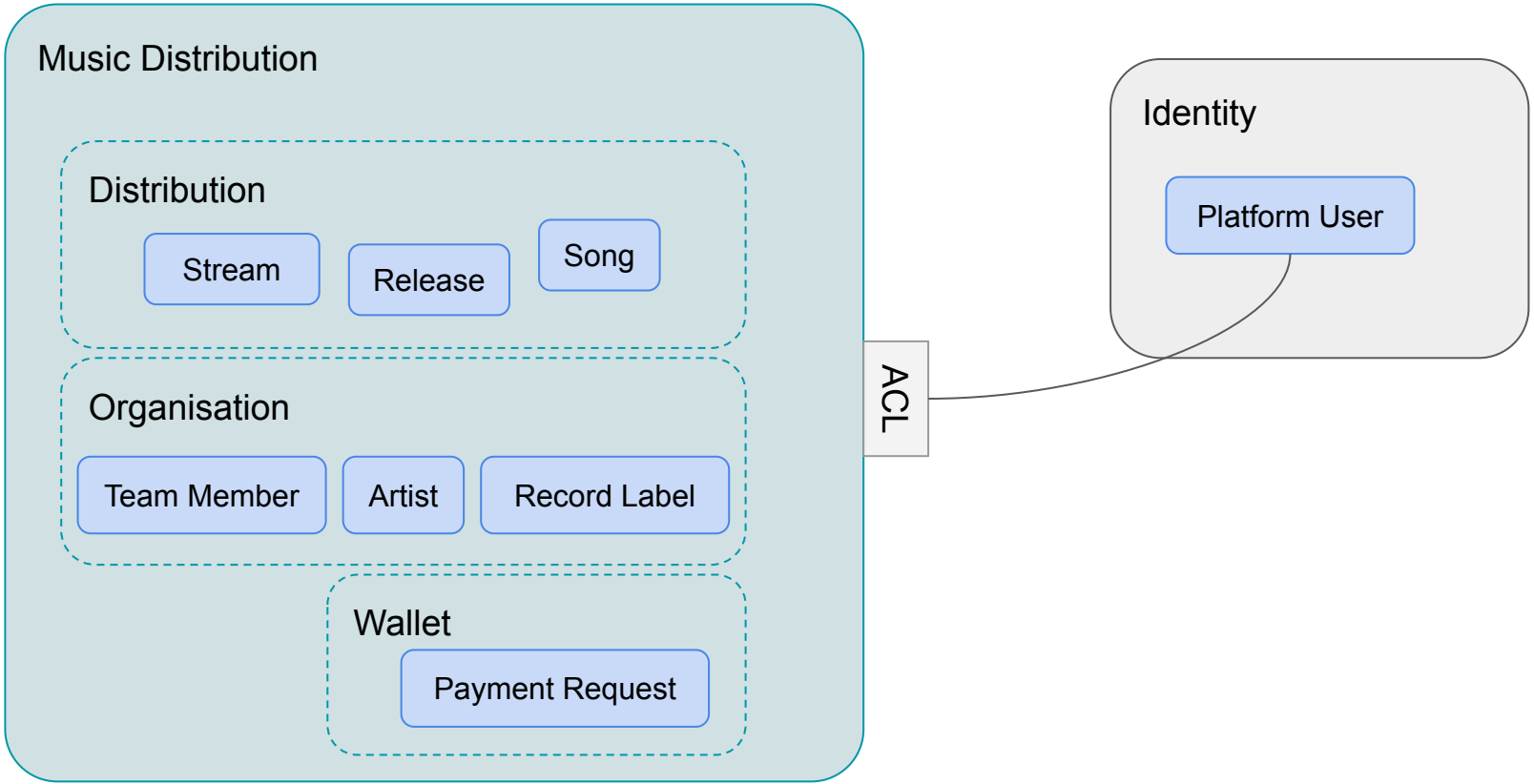
Music Distribution

for a streaming platform

Context Mapping - Potential



Context Mapping - proposal for MVP



Initial Design Flaws - state inconsistency

```
enum ReleaseState:
  case Created
  case Proposed
  case Approved
  case Withdrawn

case class Release(
  …
  date: Option[LocalDate],
  state: ReleaseState
)

// date: None, state: Approved
```

```
enum ReleaseState:
  case Created
  case Proposed(date: LocalDate)
  case Approved(date: LocalDate)
  case Withdrawn

case class Release(
  …
  state: ReleaseState
)
```

Initial Design Flaws - streams for monetization

Using timestamp is not the best idea, because:

- Late delivery of data from streaming platform
- Race conditions while saving to persistence
- Other option: mark all as counted - inefficient

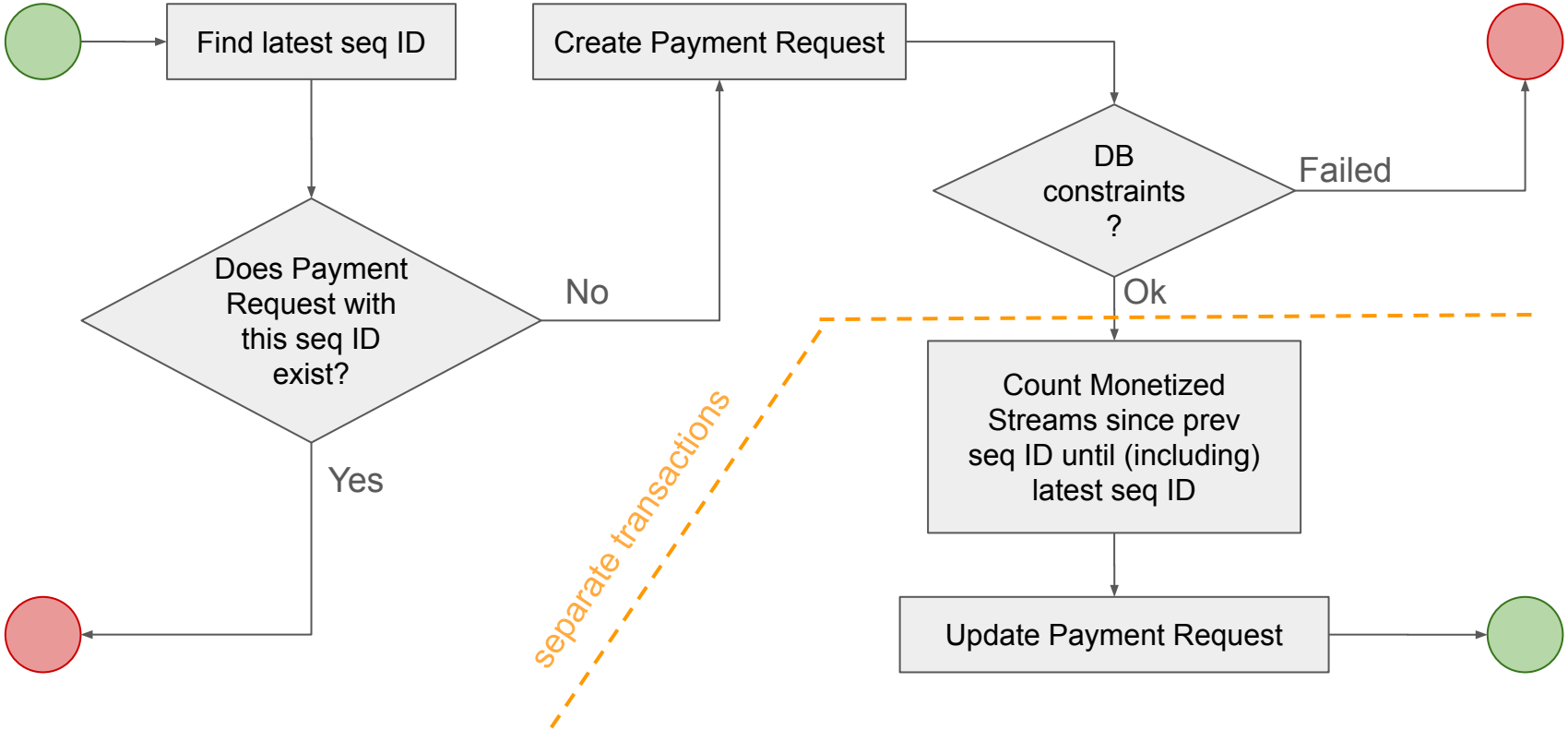
A solution:

Sequence Identifier

Implementation design - Distribution for Streaming

- Use events
- Some scheduler checks everyday for approved releases with current date
- Scheduler triggers update **Approved** -> **Distributed**
- This update is propagated as an event by
  - CDC - triggered by update on db;
  - or, using transactional outbox
  - or, pushed from code (although simple and good for MVP, it’s error prone)
- One issue I see here is that we may need more release states:
  - **Approved**
  - **SetForDistribution**
  - **Distributed**

Implementation design - Payment Request



Next steps

- API - possibly simple REST-ish one (unblocks demoing)
- Persistence - allows checking scalability of the model
- Mock Streaming Provider integration, allows to develop further:
  - Distributing Release
  - Withdrawing Release
  - Filing Payment Requests
  - Reports
- Identity service - for all fings Auth*
- …and many more