Integrations

What is integration?

- Interaction with 3rd party application
- Credentials
- RPC, SOAP, REST
- Import and export of entities

During implementation - find a balance between cost and performance

Ways to integrate

- Simple controller or CLI command with hardcoded configuration
- Simple controller or CLI command with configuration at the system configuration
- Proper implementation with configuration at the system configuration
- Proper implementation with configuration at the integration channel

Simple controller or CLI command with hardcoded configuration

+ Easy to build
+ Quick implementation
- Not configurable
- Not flexible
- Hard to support
- Only one instance

Simple controller or CLI command with configuration at the system configuration

+ Easy to build
+ Quick implementation
+ Configurable
- Not flexible
- Hard to support
- Only one instance

Proper implementation with configuration at the system configuration

+ Configurable
+ Flexible
+ Easy to support
- Hard to build
- Slow implementation
- Only one instance

Proper implementation with configuration at the integration channel

+ Configurable
+ Flexible
+ Easy to support
+ Multiple instances
- Hard to build
- Slow implementation

Configuration sources

- System configuration
  - One on the global level
  - One per each scoped entity
    - (ex. Organization -> MS Exchange)
- Integration channels
  - Any amount of integrations
  - Ability to trigger data synchronization
  - Log

Demonstration UI >>>

Integration (sync)

- Import and export via connectors
- Asynchronous (via message queue)
  - But only on a single integration level
  - All connectors and batches run in a single process
- Request history (statuses)
- oro:cron:integration:sync
- Runs automatically every 5 minutes

Demonstration UI + console >>>

Integration flow (direct sync)

- Sync from (ex. manually click sync button)
- MQ
- SyncProcessor::process()
- Iterate over all connectors
- Execute Import job
  - Reader - connector (uses transport to call external source)
  - Processor
  - Writer

Demonstration code >>>

Integration flow (reverse sync)

- Sync to (ex. listening to changes)
- MQ
- ReverseSyncProcessor::process()
- Execute Export job
  - Reader
  - Processor
  - Writer - uses transport to send data

Integration consists of...

- Channel type
- Transport
  - Settings entity
  - Settings form type
- Connectors

Channel Type

- Represents integration type
- Contains label and icon
- DIC tag oro_integration.channel
- [Zendesk example]

Transport

- Represents communication gateway
- Contains reference to integration settings (entity and form type)
- May contain methods to interact with 3rd party application
- DIC tag oro_integration.transport
- [Zendesk example]

Transport Settings: Entity

- Extends OroIntegrationBundle:Transport
- Uses single table inheritance - Doctrine docs
  - Remember about a migration
- Contains settings
- Returns list of settings using ParameterBag
- [Zendesk example]

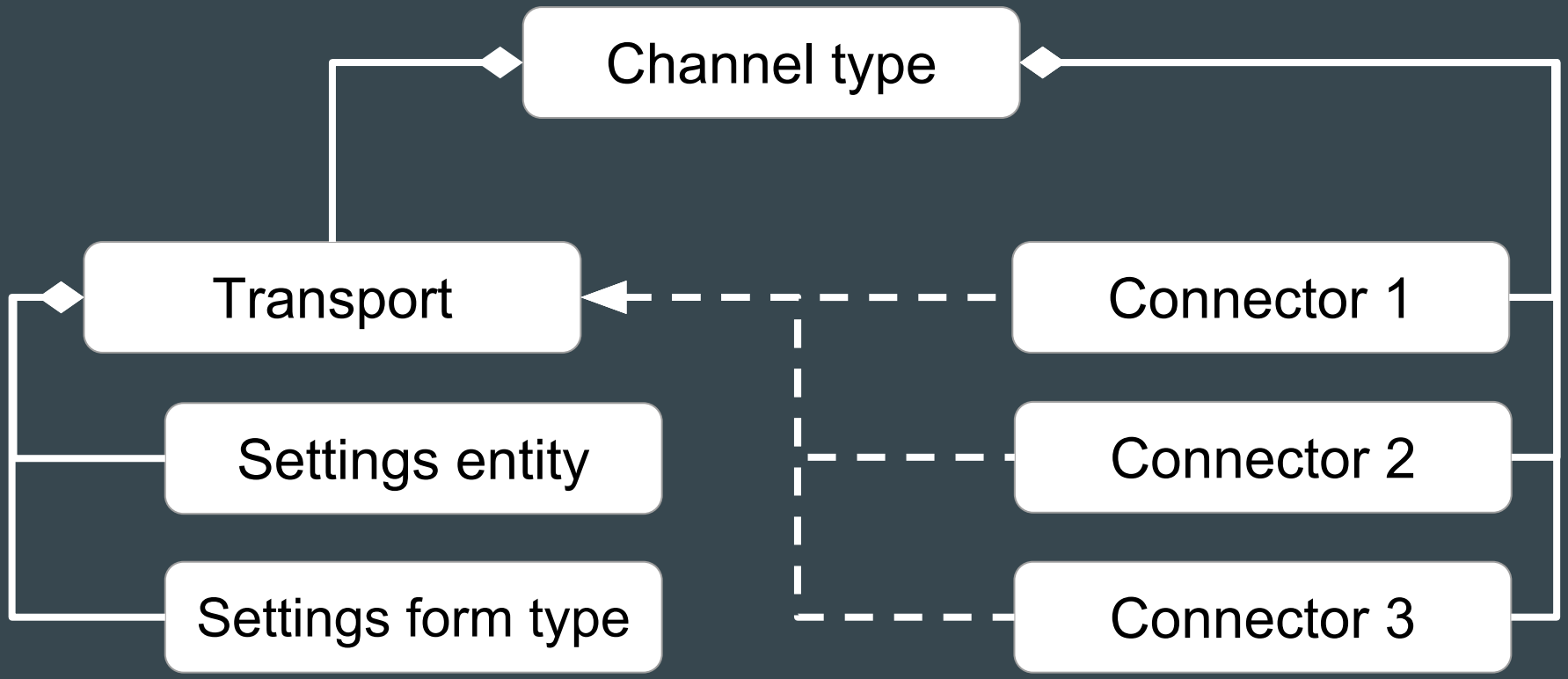
Transport Settings: Form type

- Works with settings entity
- Describes how settings are rendered
- Rendered at the integration create/edit page
- [Zendesk example]

Connector

- Represents import and export of entities
- One-way - only import
- Two-way - both import and export
- DIC tag oro_integration.connector
- [Zendesk example - TicketConnector]

Overall schema



Integration types

- Data based
  - Import and export of entities
  - Connectors
  - Synchronization (sync)
- Operation based
  - Operation calls (local or RPC)
  - No connectors
  - No synchronizations
- Combined

Possible issues

- Performance (data based integrations)
  - Same as for import/export
- No failure tolerance to 3rd party system on a global level
  - E. g. when the other side becomes not available
  - Needs to be implemented within integration level

Docs

- https://oroinc.com/orocrm/doc/current/admin-guide/integrations
- https://oroinc.com/orocrm/doc/current/dev-guide/entities/integration
- https://github.com/oroinc/platform/tree/3.1/src/Oro/Bundle/IntegrationBundle