Evolveum

Asynchronous Resources - ActiveMQ

Pavol Mederly
May 2021

Agenda

- What
- When
- How
- A practical example
- Discussion

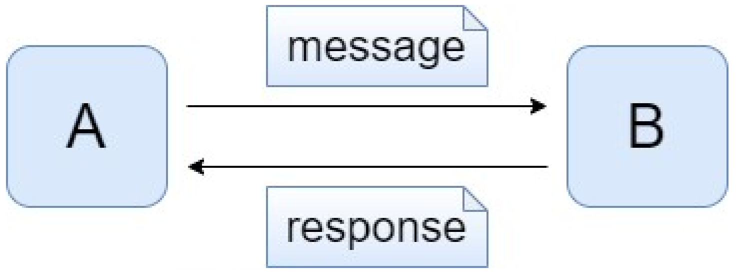
Evolveum

What are Asynchronous Resources?

Synchronous vs. Asynchronous Communication

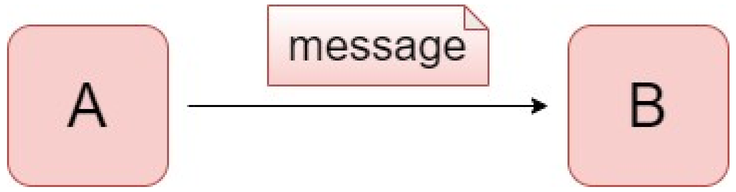
Synchronous

- A party sends a message and **waits for reply**.
- **Only then** it continues with its business.



Asynchronous

- A party sends a message and **does not wait for reply**. (Or does not expect reply at all.)
- It continues with its business **immediately**.



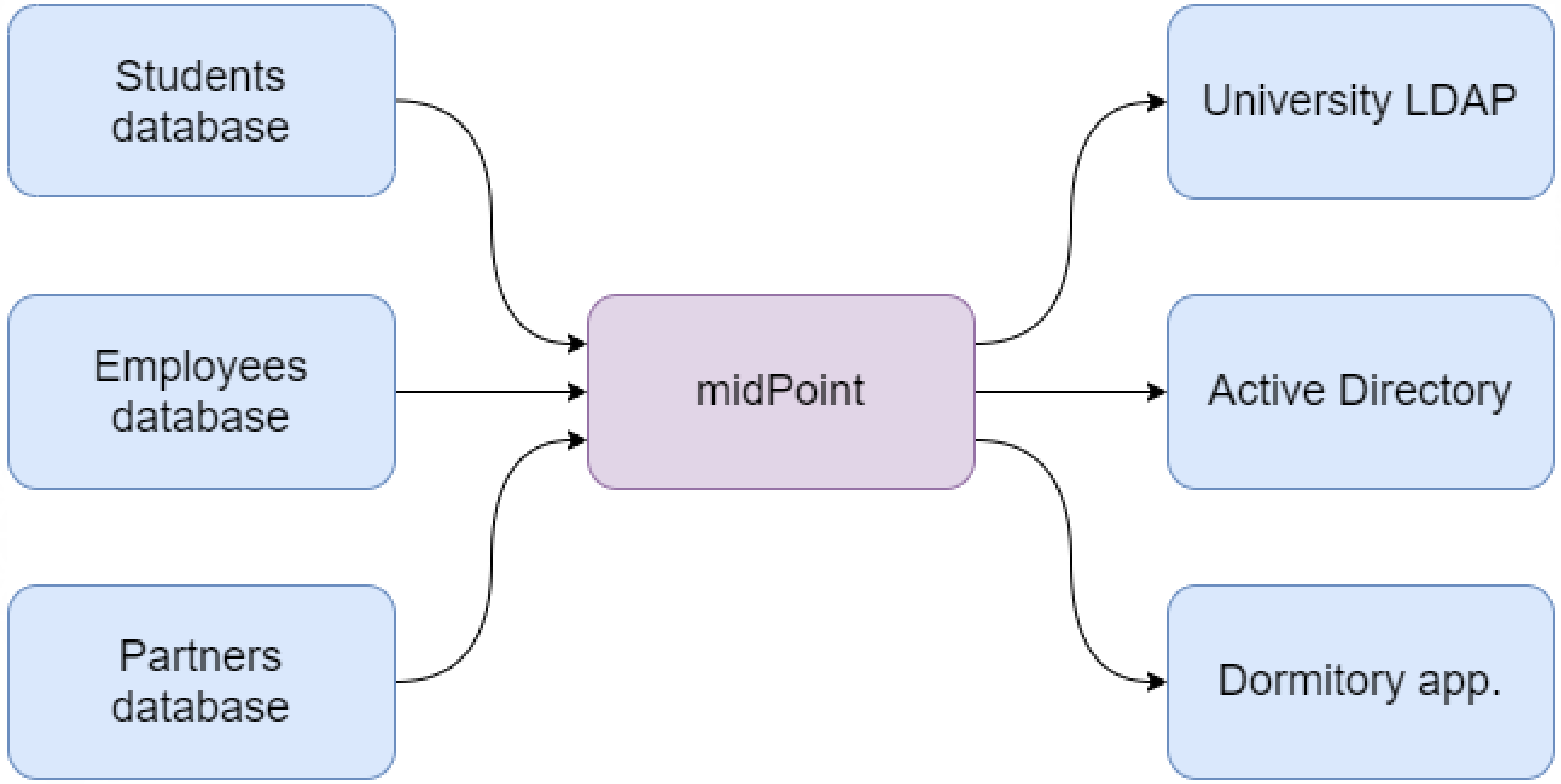
Evolveum

Synchronous vs. Asynchronous Communication in MidPoint

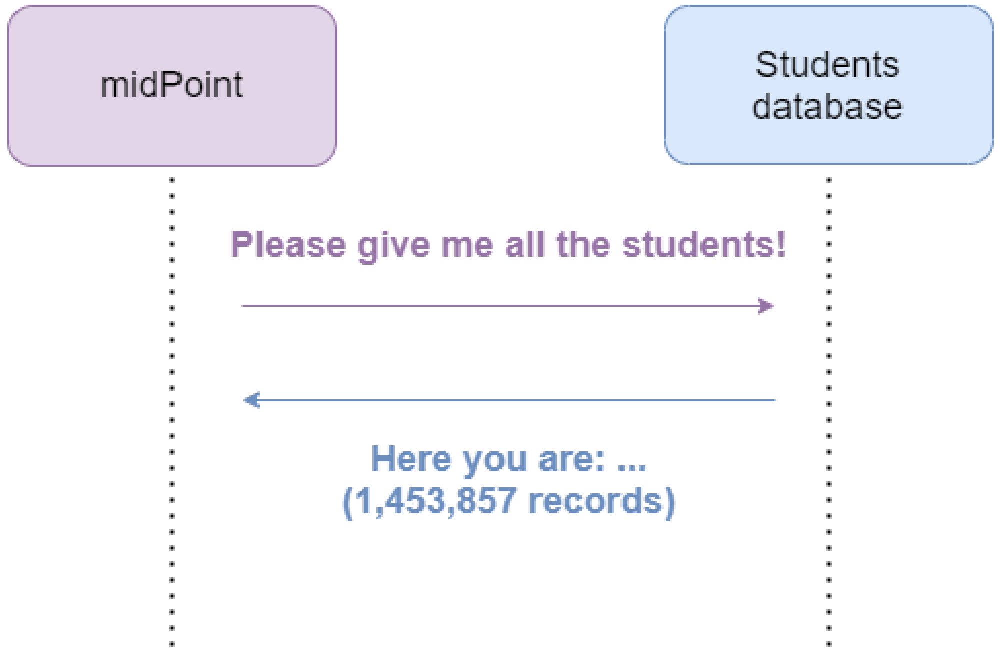
	Synchronous Communication	**Asynchronous Communication**
From Sources	Reconciliation Live Synchronization	Asynchronous Update
To Targets	Standard Provisioning (Reconciliation)	Asynchronous Provisioning

Evolveum

Sample Scenario

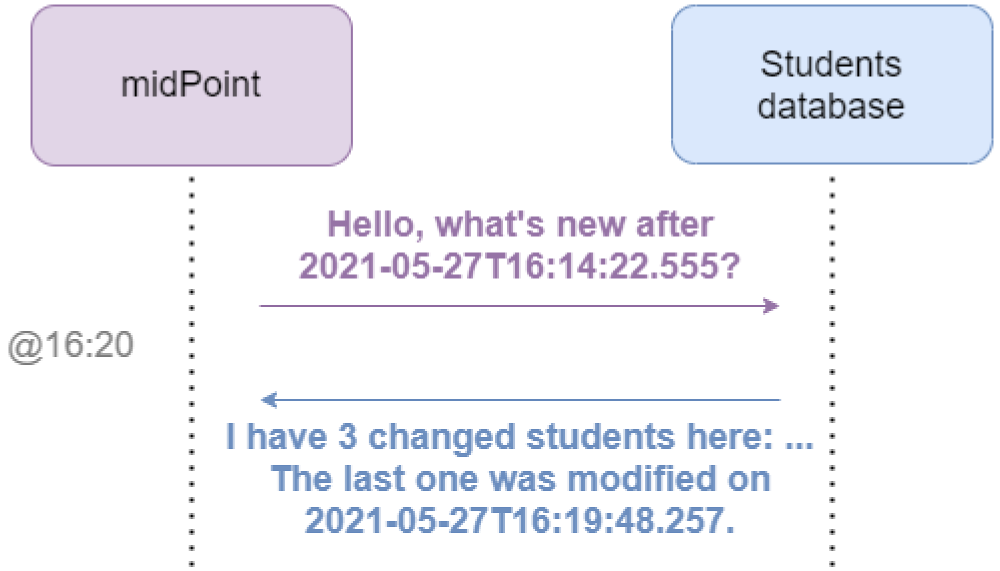


Source Side: Reconciliation

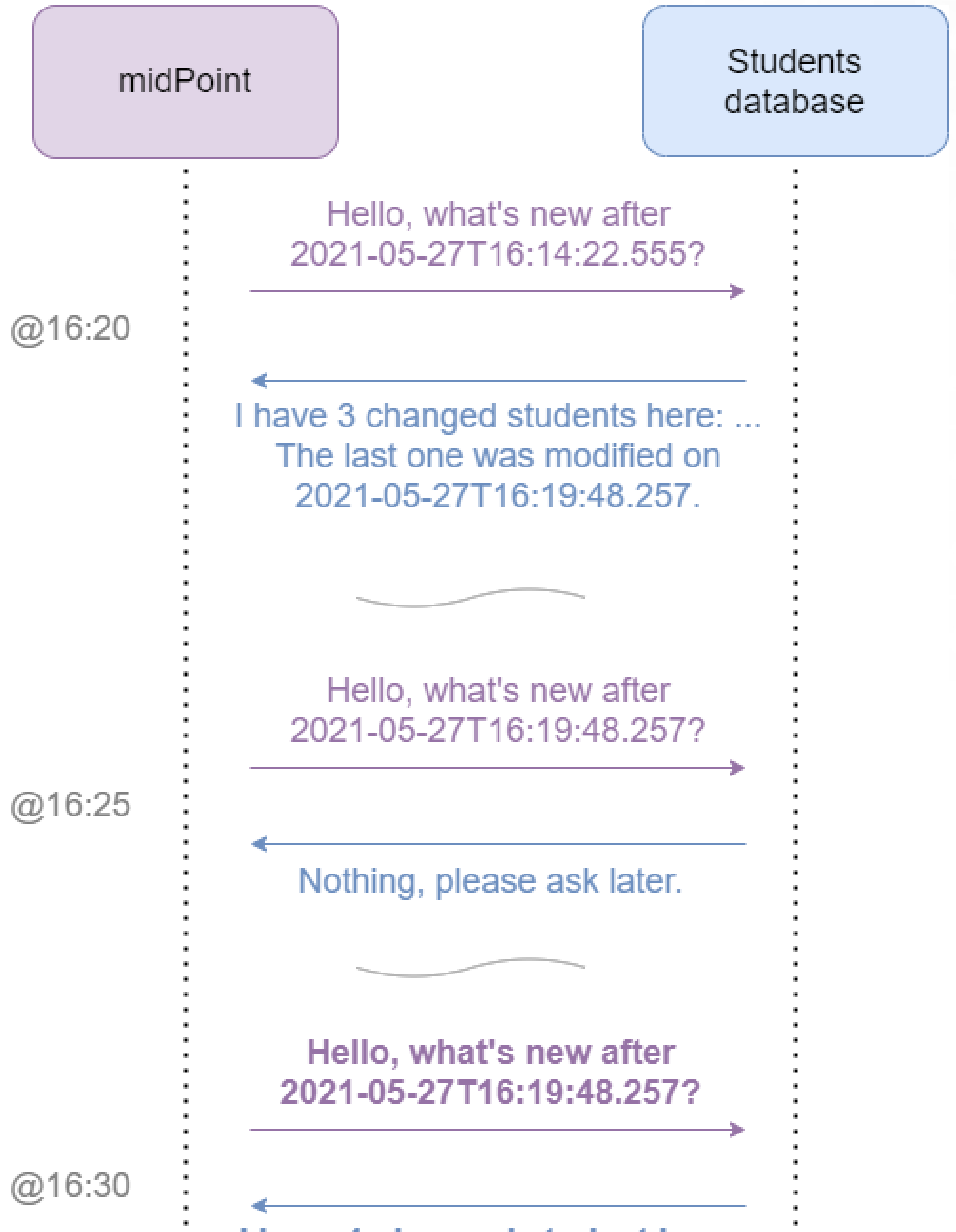


Note: slightly simplified

Source Side: Live Sync


midPoint
Students database
Hello, what's new after 2021-05-27T16:14:22.555?
@16:20
I have 3 changed students here: ... The last one was modified on 2021-05-27T16:19:48.257.

Live Sync



Hello, what's new after 2021-05-27T16:14:22.555?

I have 3 changed students here: ... The last one was modified on 2021-05-27T16:19:48.257.

Hello, what's new after 2021-05-27T16:19:48.257?

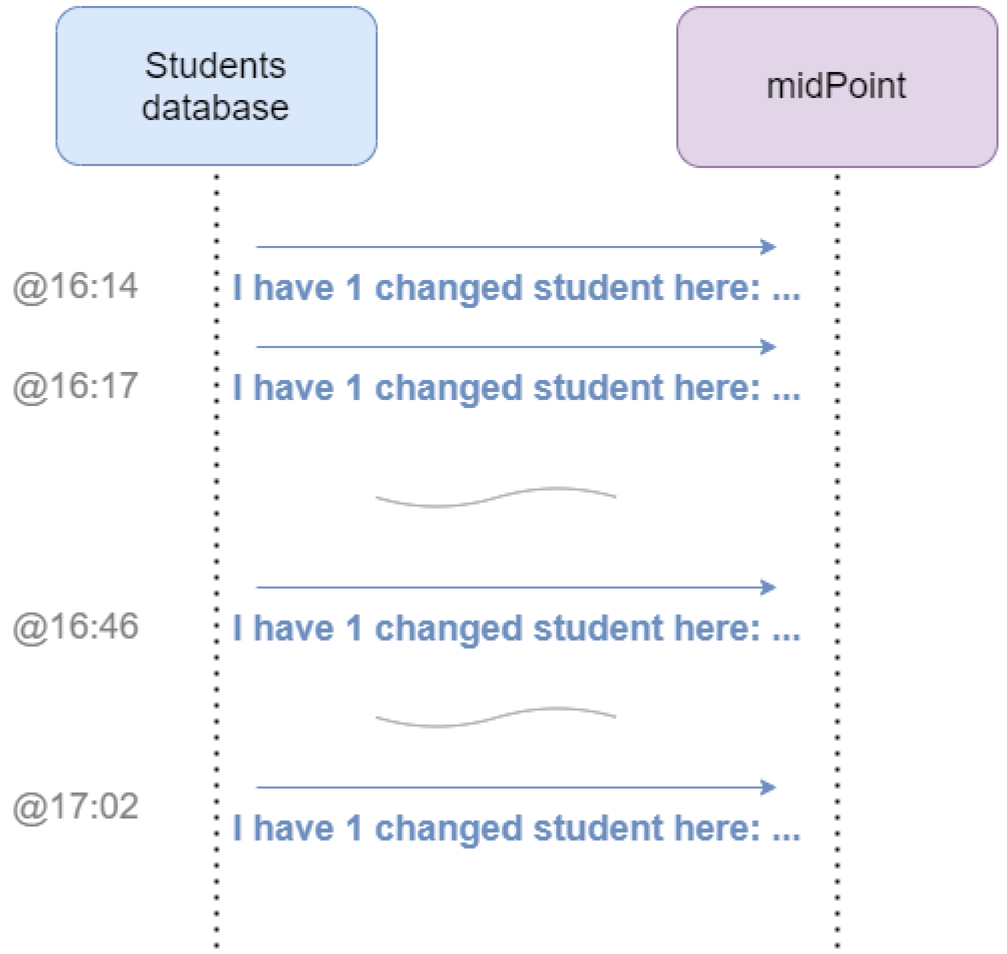
Nothing, please ask later.

Hello, what's new after 2021-05-27T16:19:48.257?

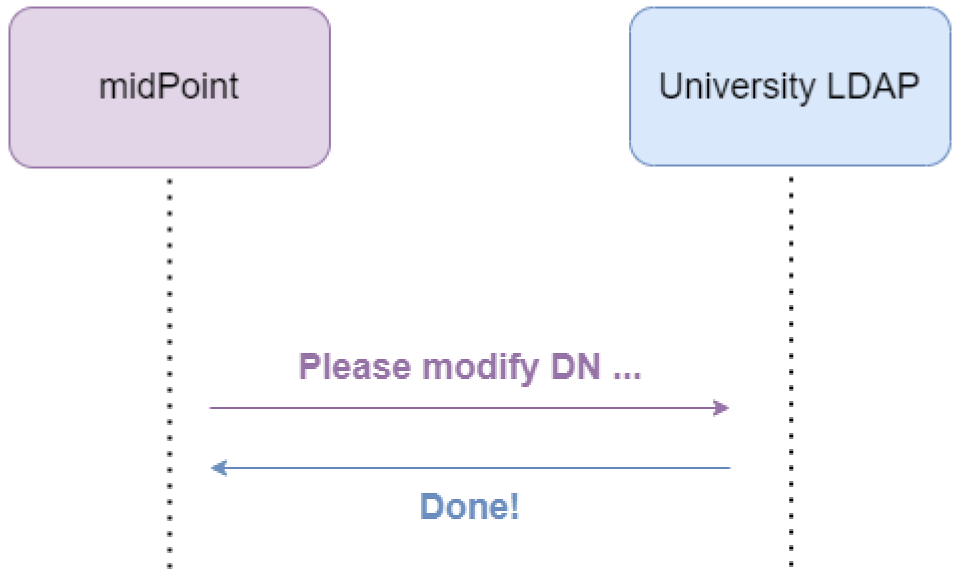
I have 1 changed student here: ... He was modified on 2021-05-27T16:27:31.981.

Evolveum

Source Side: Asynchronous Updates


Students database
midPoint
@16:14
I have 1 changed student here: ...
@16:17
I have 1 changed student here: ...
@16:46
I have 1 changed student here: ...
@17:02
I have 1 changed student here: ...

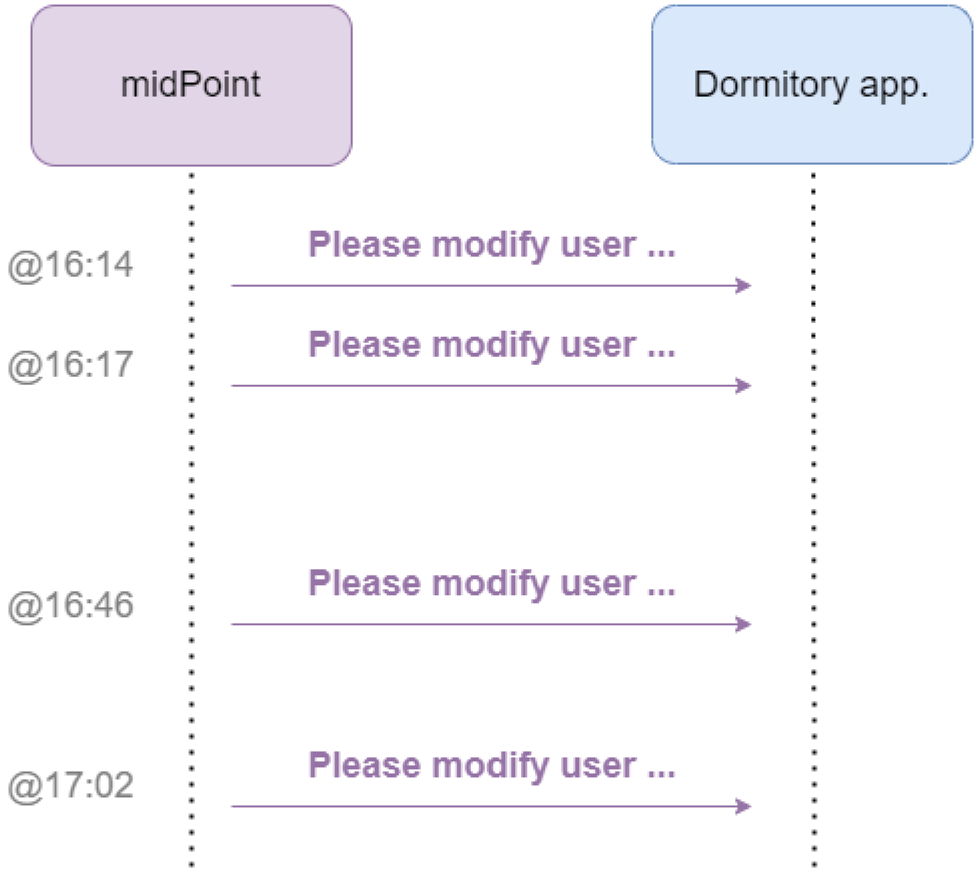
Target Side: Standard Provisioning



The synchronous nature of communication enables immediate reaction to unexpected situations.

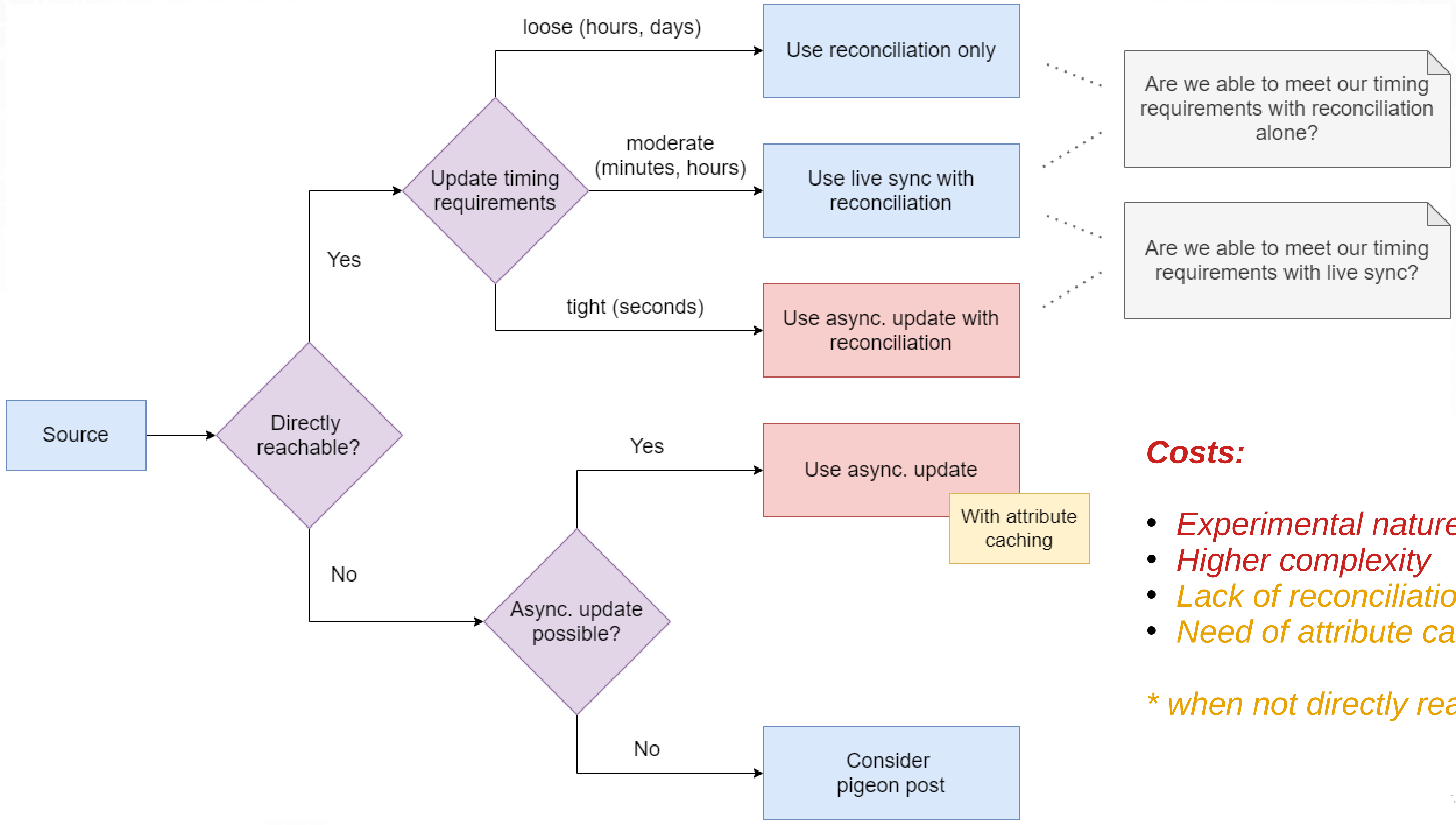
Evolveum

Target Side: Asynchronous Provisioning



When To Use Asynchronous Resources?

Deciding at the Source Side



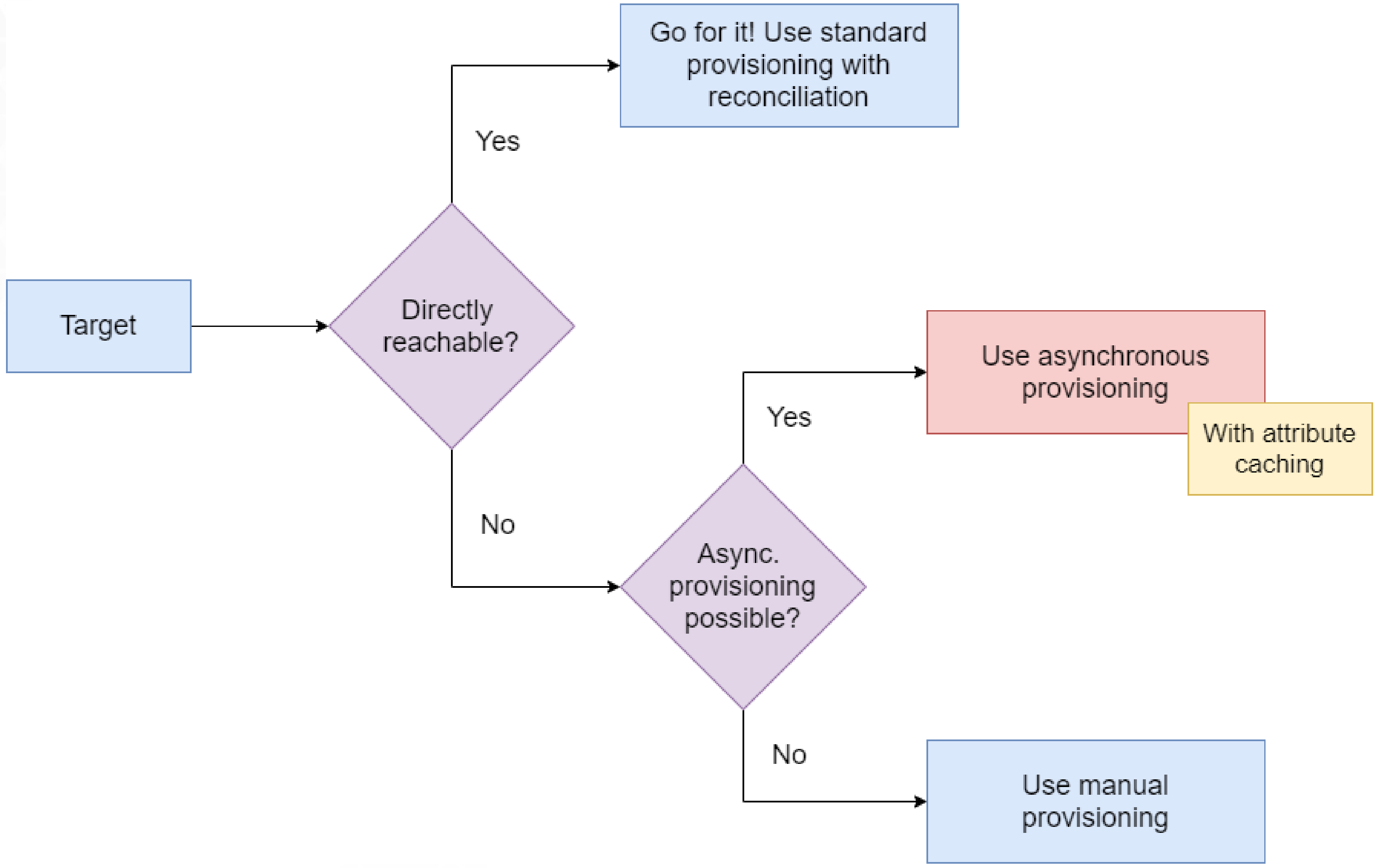
Source
Directly reachable?
Yes
Update timing requirements
loose (hours, days)
Use reconciliation only
moderate (minutes, hours)
Use live sync with reconciliation
tight (seconds)
Use async. update with reconciliation
Are we able to meet our timing requirements with reconciliation alone?
Are we able to meet our timing requirements with live sync?
No
Async. update possible?
Yes
Use async. update
With attribute caching
No
Consider pigeon post

Costs:

- *Experimental nature*
- *Higher complexity*
- *Lack of reconciliation**
- *Need of attribute caching**

** when not directly reachable*

Deciding at the Target Side


Target
Directly reachable?
Yes
Go for it! Use standard provisioning with reconciliation
No
Async. provisioning possible?
Yes
Use asynchronous provisioning
With attribute caching
No
Use manual provisioning

Costs:

- *Cannot immediately react to exceptional situations*
- *Experimental nature*
- *Higher complexity*
- *Need of attribute caching**
- *Lack of reconciliation**

** currently*

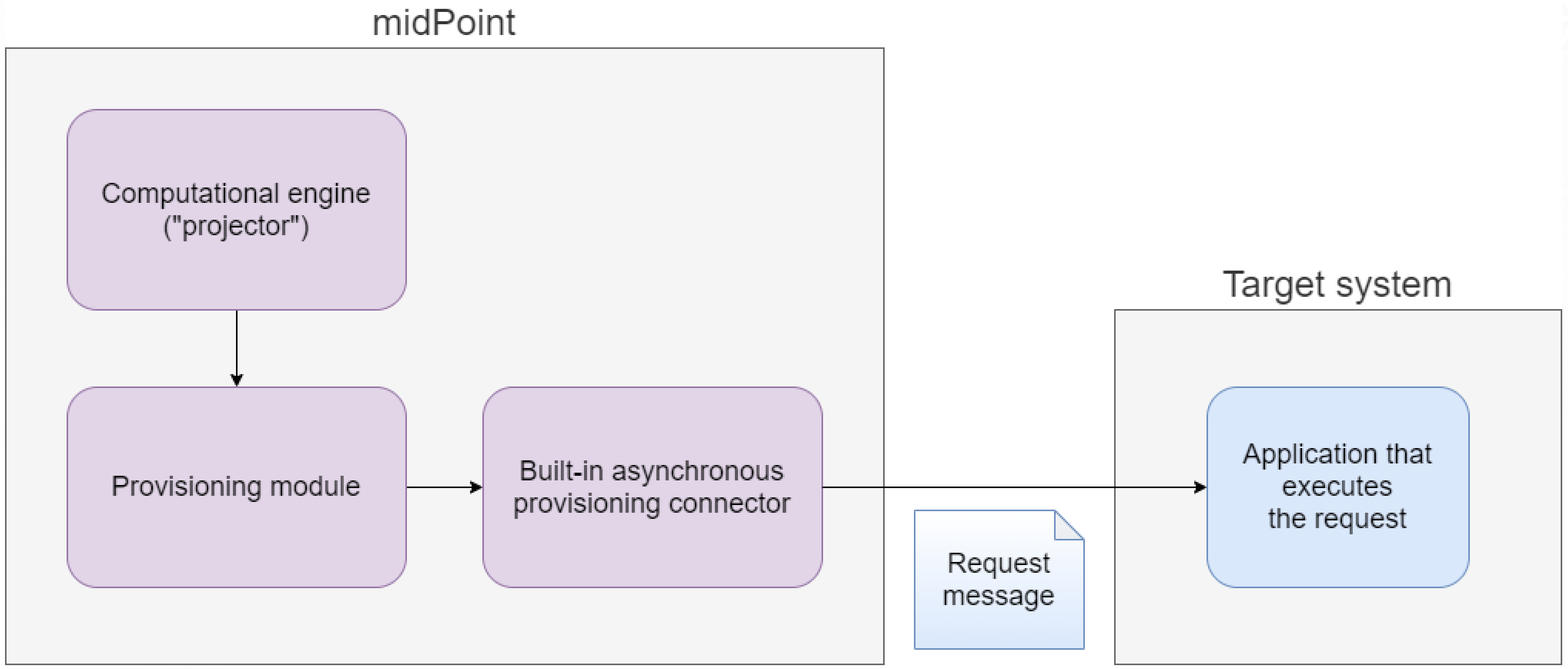
Note: A special case could be a resource with very slow responses. Here the asynchronous provisioning could make sense as well (in the future).

Evolveum

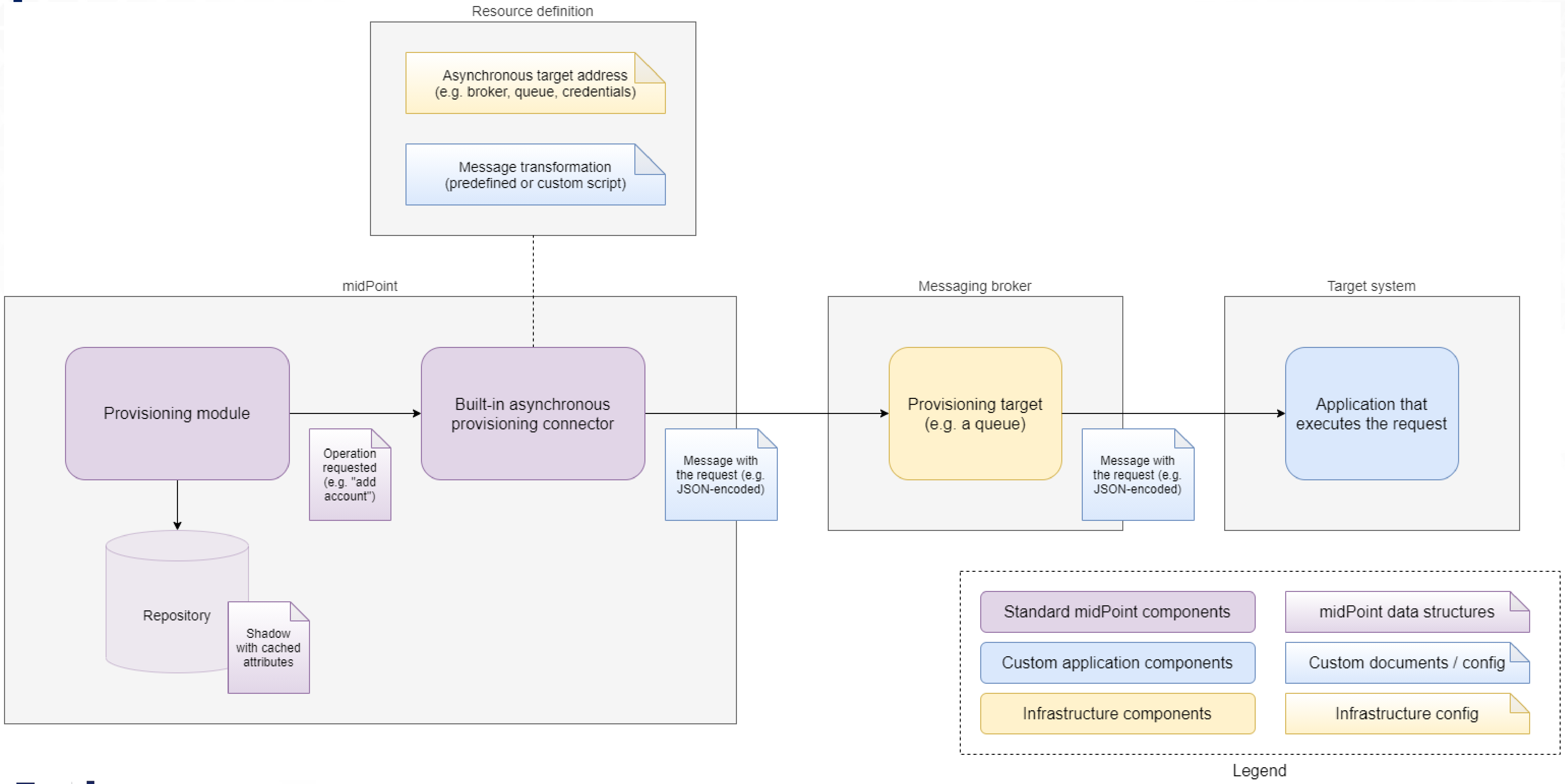
How To Use Asynchronous Provisioning?

Note: Details about asynchronous update (source side) are not in the scope of this webinar.

Overall Schema Simplified


midPoint
Computational engine ("projector")
Provisioning module
Built-in asynchronous provisioning connector
Request message
Target system
Application that executes the request

Overall Schema Details

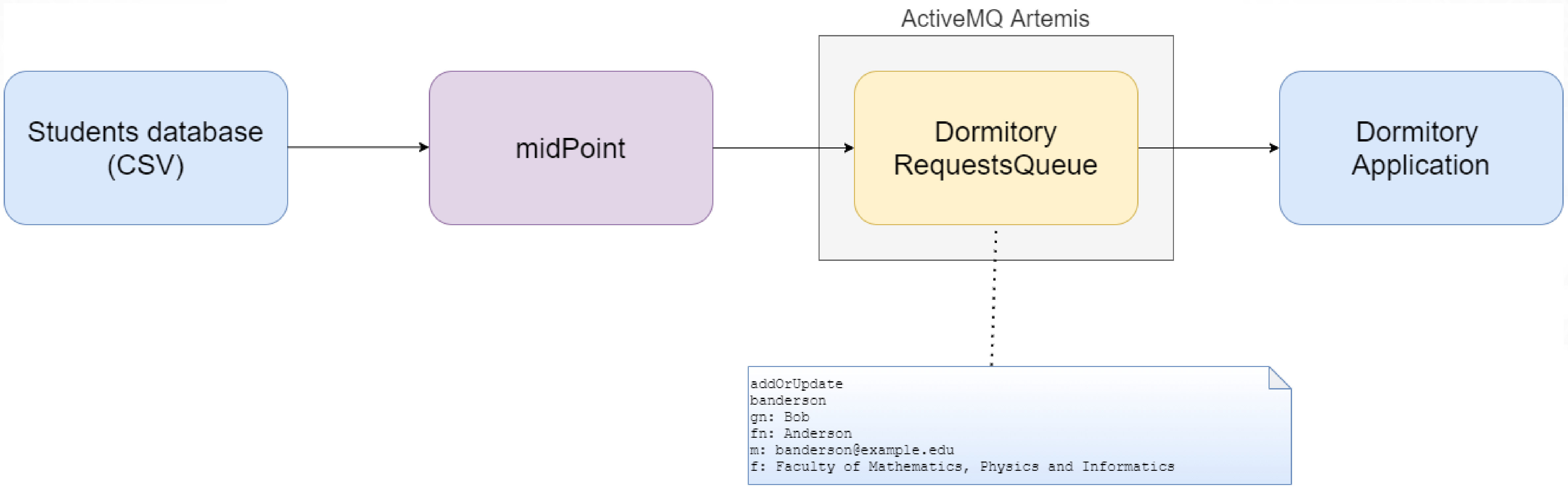

Resource definition
Asynchronous target address (e.g. broker, queue, credentials)
Message transformation (predefined or custom script)
midPoint
Provisioning module
Operation requested (e.g. "add account")
Built-in asynchronous provisioning connector
Repository
Shadow with cached attributes
Message with the request (e.g. JSON-encoded)
Messaging broker
Provisioning target (e.g. a queue)
Message with the request (e.g. JSON-encoded)
Target system
Application that executes the request
Standard midPoint components
midPoint data structures
Custom application components
Custom documents / config
Infrastructure components
Infrastructure config
Legend

Evolveum

Practical Example

Asynchronous Provisioning to Fictitious Dormitory Application via ActiveMQ

Schema of the Example



```
addOrUpdate
banderson
gn: Bob
fn: Anderson
m: banderson@example.edu
f: Faculty of Mathematics, Physics and Informatics
```

Implementation: Resource Definition

```xml
<name>Dormitory Test (async)</name>
<connectorRef type="ConnectorType">
    <filter>
        <q:equal>
            <q:path>connectorType</q:path>
            <q:value>AsyncProvisioningConnector</q:value>
        </q:equal>
    </filter>
</connectorRef>
<connectorConfiguration>
    <conf:targets>
        <jms>
            <connectionFactory>localhostConnectionFactory</connectionFactory>
            <username>admin</username>
            <password>admin123</password>
            <destination>TestQueue</destination>
        </jms>
    </conf:targets>
    <conf:predefinedTransformation>simplifiedJson</conf:predefinedTransformation>
</connectorConfiguration>
```

(1) connector type

(2) where to send messages to

(3) physical message format

```xml
<capabilities>
    <configured xmlns:cap="http://midpoint.evolveum.com/xml/ns/public/resource/capabil
        <cap:read>
            <cap:cachingOnly>true</cap:cachingOnly>
        </cap:read>
    </configured>
</capabilities>
```

(1) attribute caching

```xml
<xsd:sequence>
    <xsd:element name="login" type="xsd:string" minOccurs="0"/>
    <xsd:element name="givenName" type="xsd:string" minOccurs="0"/>
    <xsd:element name="familyName" type="xsd:string" minOccurs="0"/>
    <xsd:element name="email" type="xsd:string" minOccurs="0"/>
    <xsd:element name="faculty" type="xsd:string" minOccurs="0"/>
</xsd:sequence>
```

(3) logical message format (schema)

```json
{
  "operation" : "add",
  "objectClass" : "AccountObjectClass",
  "attributes" : {
    "faculty" : [ "Faculty of Mathematics, Physics and Informatics" ]
  }
}
```

```
java.naming.factory.initial=org.apache.activemq.artemis.jndi.ActiveMQInitialContextFactory
connectionFactory.localhostConnectionFactory=tcp://localhost:61616
queue.TestQueue=TestQueue
queue.DormitoryRequestsQueue=DormitoryRequestsQueue
```

jndi.properties wrapped in a JAR on classpath

```json
{
  "operation" : "modify",
  "objectClass" : "AccountObjectClass",
  "primaryIdentifiers" : {
    "login" : [ "banderson" ]
  },
  "secondaryIdentifiers" : { },
  "changes" : {
    "givenName" : {
      "replace" : [ "BOB" ]
    }
  }
}
```

```json
{
  "operation" : "delete",
  "objectClass" : "AccountObjectClass",
  "primaryIdentifiers" : {
    "login" : [ "banderson" ]
  },
  "secondaryIdentifiers" : { }
}
```

https://docs.evolveum.com/midpoint/reference/resources/asynchronous/outbound/configuration/#resource-definition

Custom Request Message Transformation

```
        <conf:transformExpression>
            <script>
                <language>http://midpoint.evolveum.com/xml/ns/public/expression/language#velocity</language>
                <code>#set ( $request = $requestFormatter.changeMapAsAttributes().identifiersAsAttributes().createRequest() )
#set ( $attrs = $request.attributesSimplified )
#if ($request.isDelete())
delete
$!attrs["login"]
#else
addOrUpdate
$!attrs["login"]
gn: $!attrs["givenName"]
fn: $!attrs["familyName"]
m: $!attrs["email"]
f: $!attrs["faculty"]
#end</code>
            </script>
        </conf:transformExpression>
```

```
addOrUpdate
banderson
gn: Bob
fn: Anderson
m: banderson@example.edu
f: Faculty of Mathematics, Physics and Informatics
```

(1) account addition

```
addOrUpdate
banderson
gn: BOB
fn: Anderson
m: banderson@example.edu
f: Faculty of Mathematics, Physics and Informatics
```

(2) account modification

```
delete
banderson
```

(3) account deletion

https://docs.evolveum.com/midpoint/reference/resources/asynchronous/outbound/configuration/#using-custom-transformation

Evolveum

Live Demo

- d:\mp-home\apache-artemis-2.17.0\bin\artemis.cmd create d:\tmp\broker --user admin --password admin123 --require-login
- paste queue definitions to d:\tmp\broker\etc\broker.xml (addresses)
- d:\tmp\broker\bin\artemis run
- git clone https://github.com/mederly/webinar-async-provisioning-2021-05
- mvn clean install
- mvn exec:java
- mkdir d:\tmp\midpoint-4.3.1\var\lib
- copy d:\tmp\webinar-async-provisioning-2021-05\target\*jndi.jar d:\tmp\midpoint-4.3.1\var\lib
- d:\tmp\midpoint-4.3.1\bin\start

https://github.com/mederly/webinar-async-provisioning-2021-05

Evolveum

Summary

- Asynchronous resources are experimentally supported (both ways)
- Main reasons to use:
  - When there is no direct access (source & target)
  - When we need immediate updates from a source
  - When a provisioning target is too slow (in the future)
- Main things to consider:
  - No direct access → no guarantees of long-term consistency
  - Async provisioning → no direct feedback (limited/slower adaptation to the real state)
  - Higher complexity
  - The current support is experimental

Feel free to experiment, and let us know about your experiences.

Evolveum

Discussion

Questions?

Comments?

For more information please visit https://docs.evolveum.com/midpoint/reference/resources/asynchronous/outbound/

Evolveum

Thank you for your attention

If any questions occur, feel free to ask at **sales@evolveum.com**

Also **follow us** on our social media for further information!

/Evolveum

/Evolveum

/Evolveum

@Evolveum

/Evolveum

Evolveum