Middleware and Web Services

Lecture 6: Integration Patterns

doc. Ing. Tomáš Vitvar, Ph.D.

tomas@vitvar.com • @TomasVitvar • http://vitvar.com

Czech Technical University in Prague

Faculty of Information Technologies • Software and Web Engineering • http://vitvar.com/courses/mdw

Modified: Sun Dec 09 2018, 22:25:29
Humla v0.3

Enterprise Service Bus

- ESB is a central intermediary in SOA
  - *Types of services: shared and infrastructure*
  - *Types of processes: Technical and Business*
- ESB Application
  - *Application running on an application server*
  - *Exposes functionality via Web service interface*
  - *Allows to communicate with various messaging protocols*
- Integration Patterns
  - *Technical-level interoperability – message broker*
  - *Location transparency*
  - *Dynamic routing*
  - *Data transformations – mediator*
  - *Resequencing of messages*
  - *Session pooling*
  - *Service orchestrations – BPMN, BPEL*
  - *Message enrichment*

Major Vendors

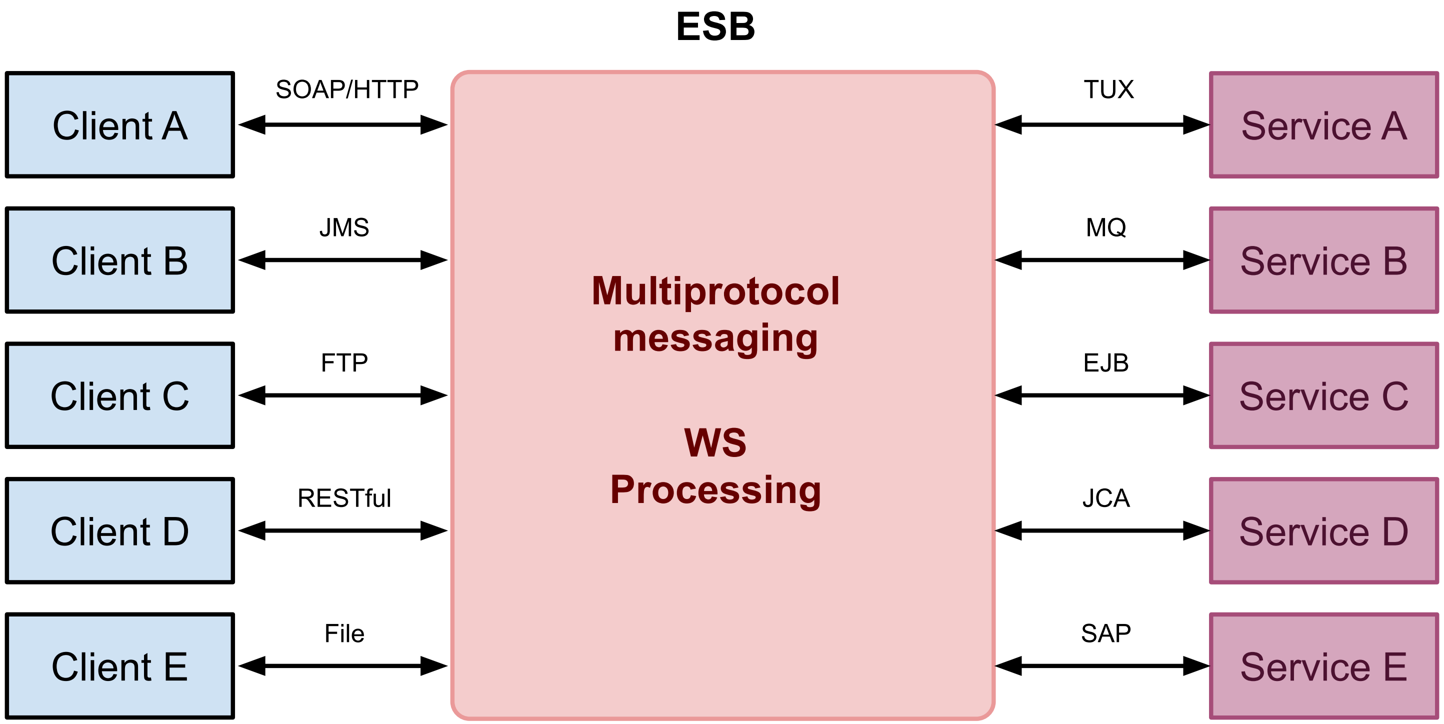
- Oracle
  - *Oracle Service Bus (OSB)*
  - *Oracle SOA Suite*
  - *Oracle Enterprise Gateway (OEG)*
- IBM
  - *IBM WebSphere*
- SAP
  - *SAP NetWeaver*
- Microsoft
  - *.NET Framework*
  - *BizTalk server*
- Opensource
  - *JBoss*
  - *Apache ServiceMix*
  - *WSMX – Semantic Web Service Execution Environment*

Integration Patterns

- Applied in implementation of business services and processes
  - *Usually a combination of more patterns*
- Technical patterns
  - *Deals with technical aspects of service communication*
  - *Message broker – technical-level interoperability*
  - *Location transparency*
  - *Session pooling*
- Business patterns
  - *Deals with business aspects (message content) of service communication*
  - *Dynamic routing*
  - *Data transformations – mediator*
  - *Service orchestrations – BPMN, BPEL*
  - *Message enrichment*
  - *Resequencing of messages*

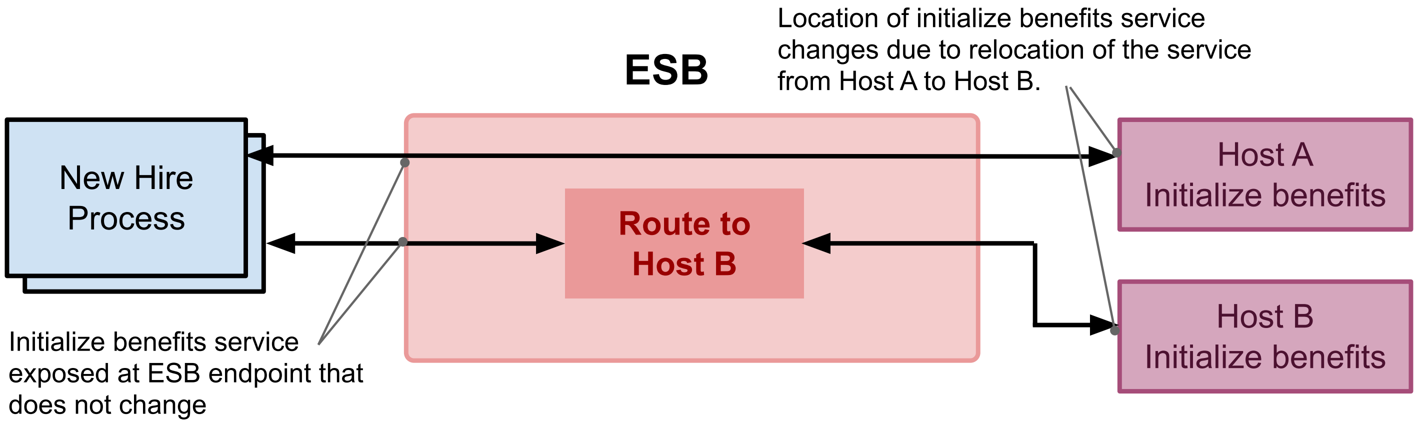
Message Broker

- Message broker
  - *ESB can mix and match transports both standard and proprietary*



Location Transparency

- Location transparency
  - *ESB can hide changes in location of services*
  - *Such changes will not affect clients*
  - *Can also be used for load balancing for multiple service instances*

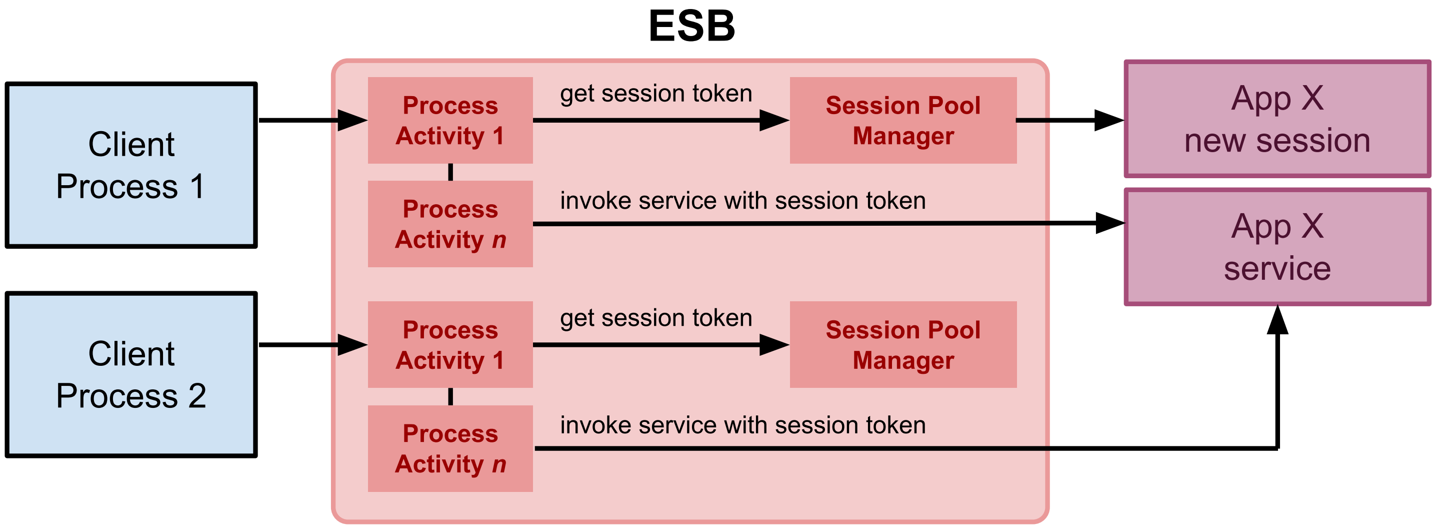


Location of initialize benefits service changes due to relocation of the service from Host A to Host B.

Initialize benefits service exposed at ESB endpoint that does not change

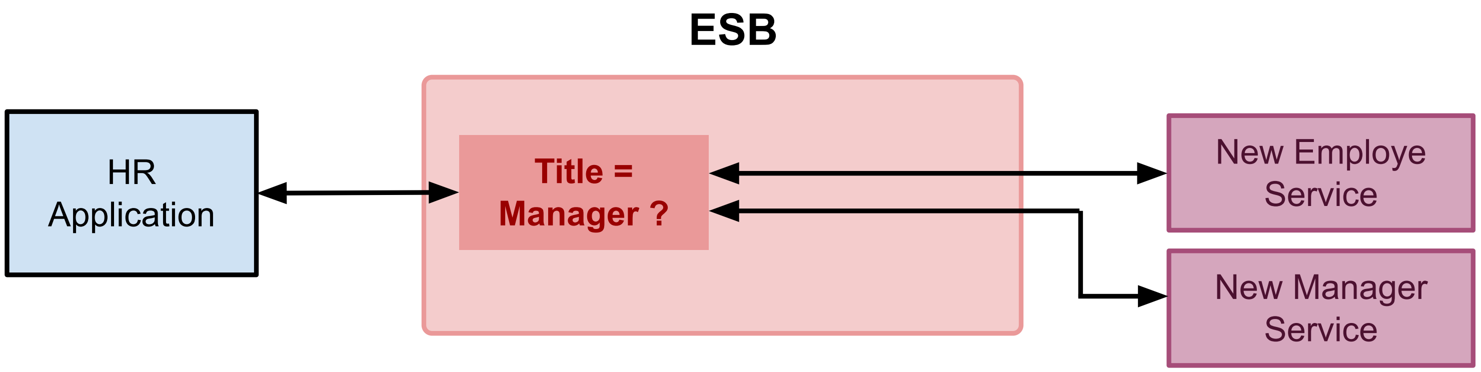
Session Pooling

- Session Pooling
  - *ESB can maintain a pool of connections (session tokens) to a back-end app when creating a new connection is expensive*
  - *A single session token can be reused by multiple instances of business processes*



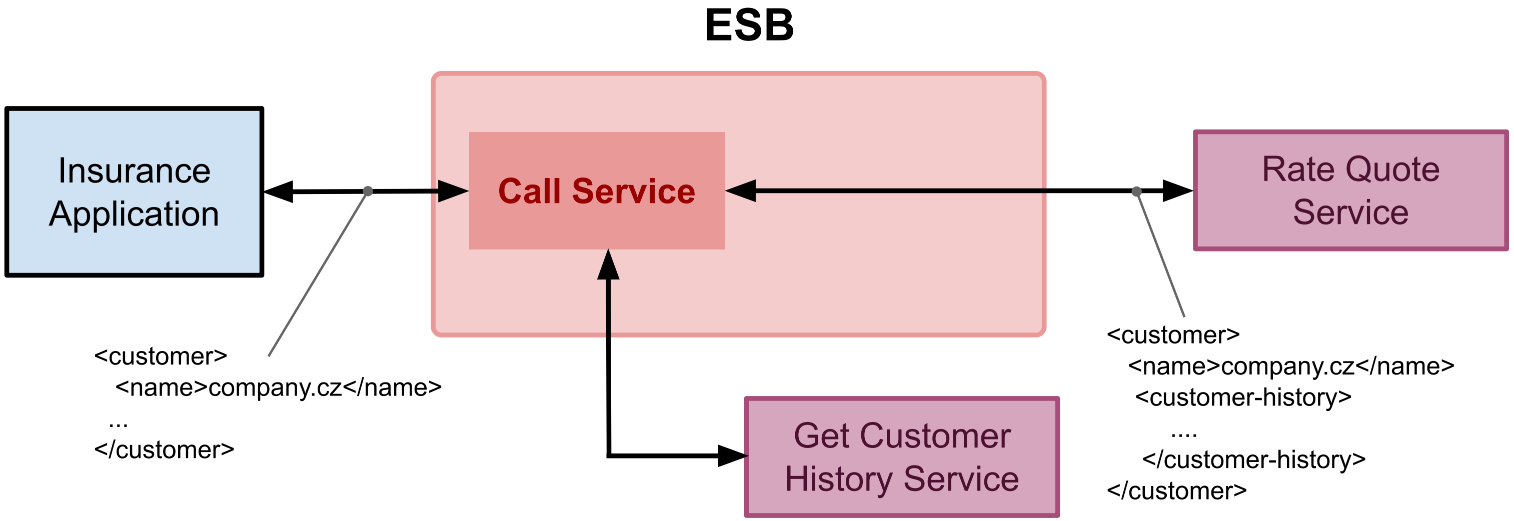
Dynamic Routing

- Dynamic routing
  - *ESB exposes a service that routes to various back-end services based on message contents.*



Message Enrichment

- Message enrichmenet
  – *Enriches a message before invoking back-end application service.*



```
<customer>
  <name>company.cz</name>
 ...
</customer>
```

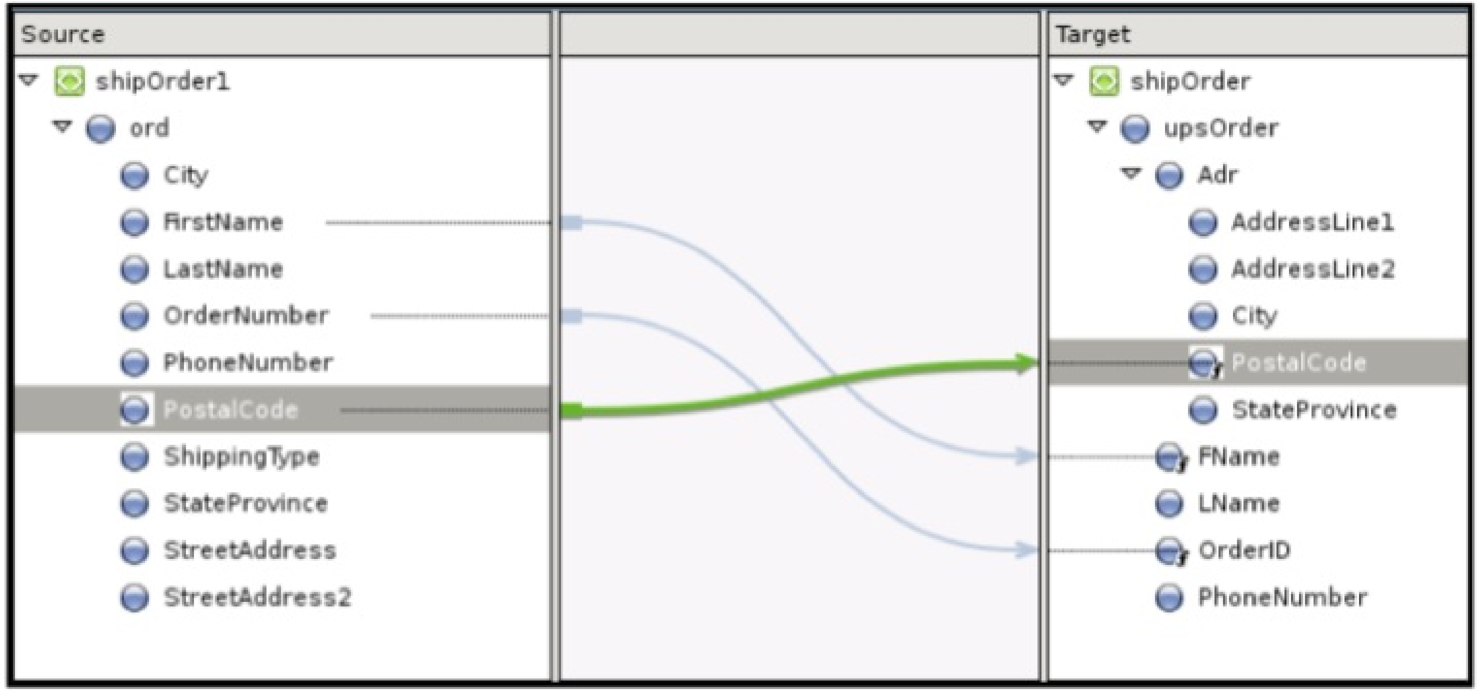
```
<customer>
  <name>company.cz</name>
   <customer-history>
      ....
    </customer-history>
</customer>
```

Data Transformation

- Data transformation phases:
  - *Definition of mapping and execution of mappings*
- Definition of mappings (design-time)
  - *A mapping associates one data structure to another data structure and defines a conversion between them.*
  - *Mapping languages*
    - → *graphical for design that translates to XSLT, XQuery*
    - → *Sometimes implemented in 3rd gen. languages (e.g., Java)*
- Execution of mappings (runtime)
  - *application of mappings to instance data*
- CDM terminology
  - *Application Business Message – back-end app format*
  - *Enterprise Business Message – CDM format*

Definitions of Data Mapping Example

- Source and target schemas
  - *Source: Order – flat data structure*
  - *Target: UPS order with address as a sub-entity*
  - *Differences in names of entities*
  - *Conversion function applied to postal code*



Service Orchestration

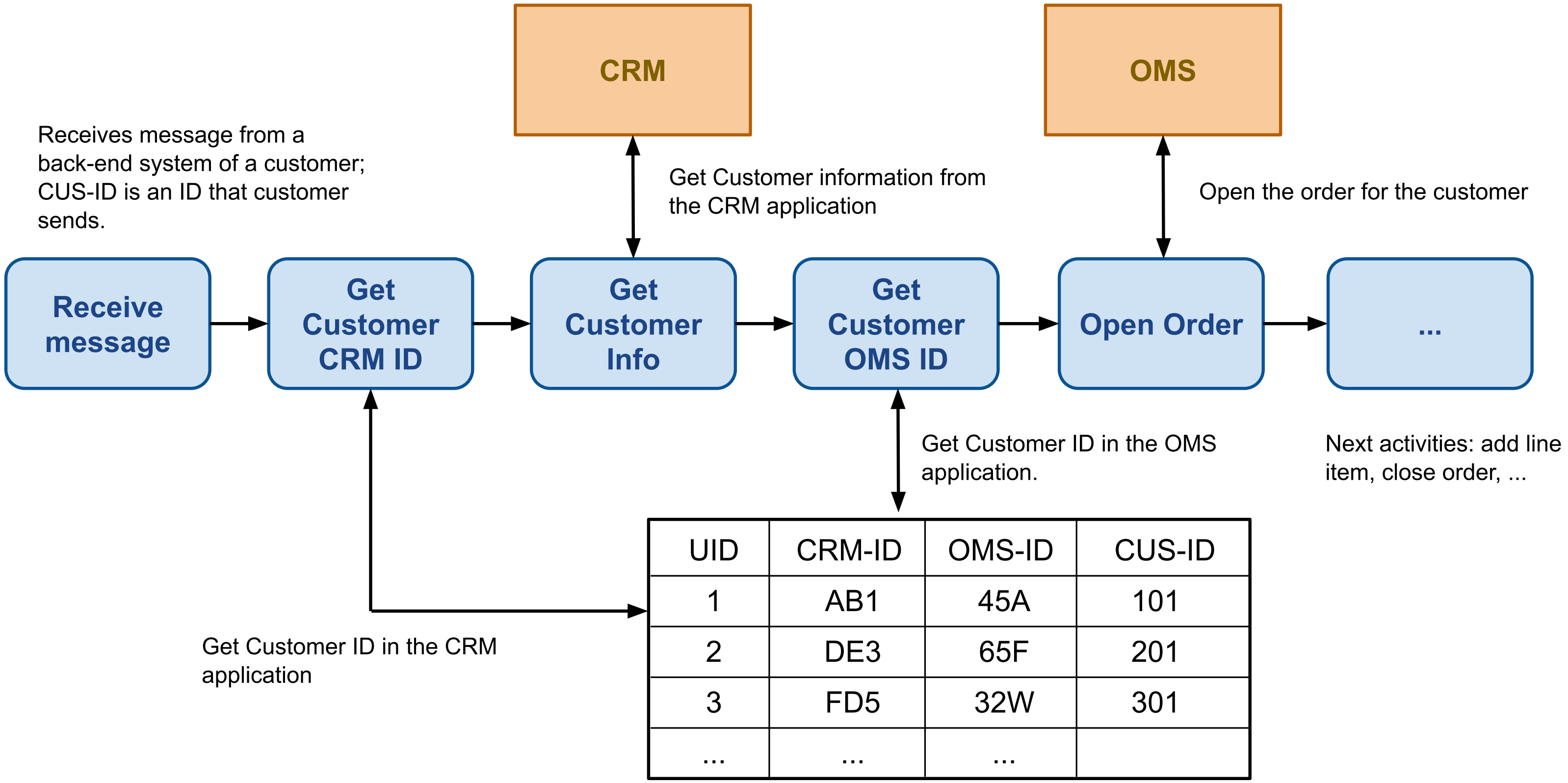
- Orchestration of multiple business services
  - *Includes transformation, message enrichmenet, service callouts, etc.*
  - *A step in orchestration is an activity*
- Patterns
  - *Sequential processing of activities*
  - *Parallel processing of activities with synchronization points*
  - *Decision branches, iterations*
- Technologies
  - *Graphical languages*
  - *Standard representations: BPEL, BPMN*
  - *Proprietary, for example OSB uses graphical language that translates to XQuery*
- Good design
  - *Orchestration faciliates communication in CDM*
  - *Orchestration handles key-mapping*

Key Mapping

- What is key mapping
  - *Key = identifier of en entity in a back-end application*
  - *Key Mapping = a mapping of an ID of an entity in one system to an ID of the same entity in another system.*
  - *Key mapping is realized using universal IDs (UID)*
- Example
  - *A customer MOON exists in CRM and OMS systems*
  - *In CRM system, MOON has an* `CRM-ID=AB1`
  - *In OMS system, MOON has an* `CRM-ID=45A`
  - *Key mappig allows to map the* `CRM-ID AB1` *to the* `OMS-ID 45A`
  - *Key mapping is a table*

```
CRM-ID → UID → OMS-ID
```

Key Mapping Example



Receives message from a back-end system of a customer; CUS-ID is an ID that customer sends.

Get Customer information from the CRM application

Open the order for the customer

Get Customer ID in the OMS application.

Next activities: add line item, close order, ...

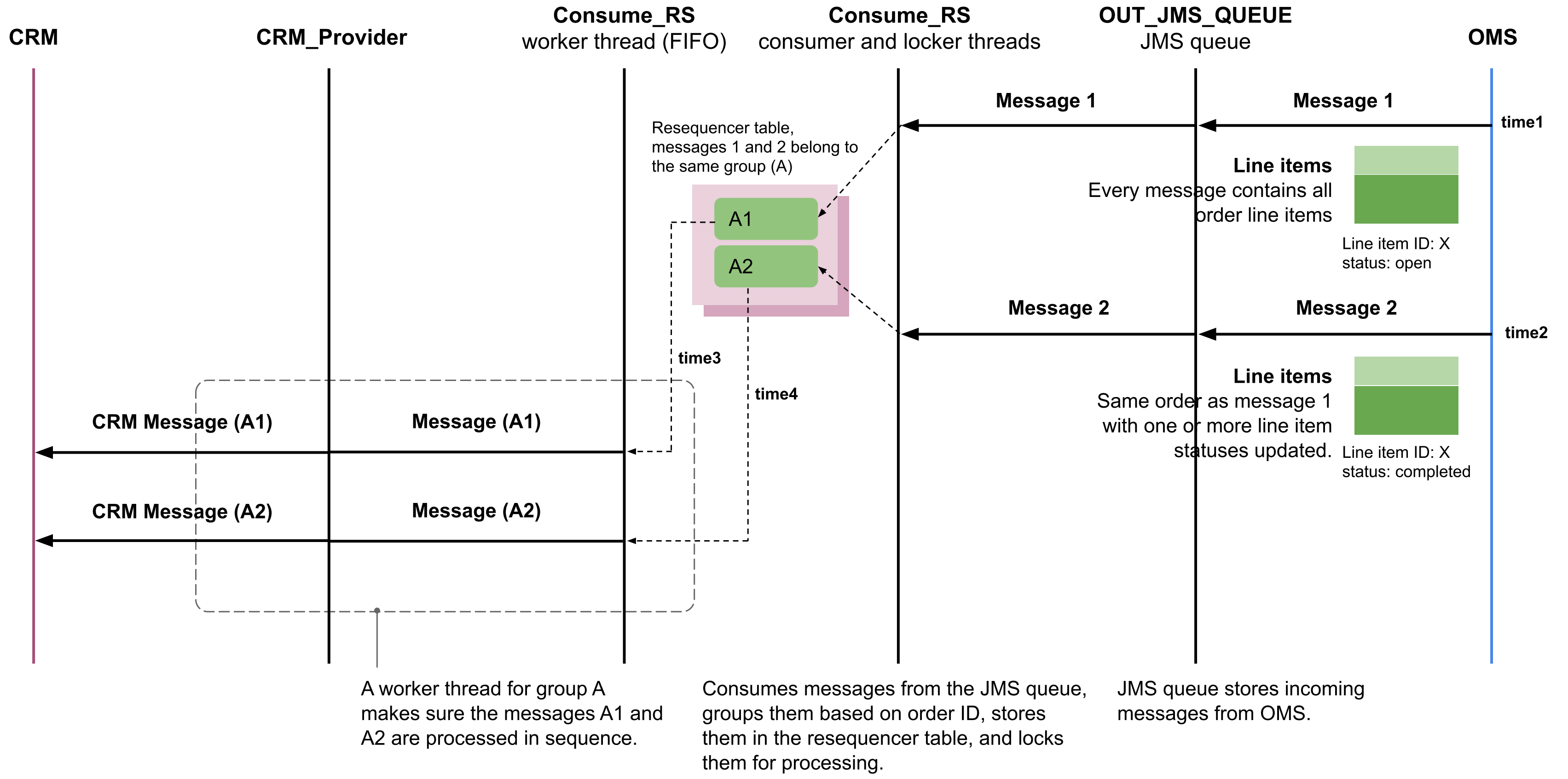
Get Customer ID in the CRM application

UID	CRM-ID	OMS-ID	CUS-ID
1	AB1	45A	101
2	DE3	65F	201
3	FD5	32W	301
...	...	...	

Key mapping:
Business Entity CUSTOMER

Message Sequencing

- Resequencer in update sales order
  - *Every order line item needs to update its status several times (e.g. open, completed)*
  - *Resequencer makes sure that the update status messages arrive to CRM in the same order as they were created in OMS system (FIFO resequencer)*


CRM
CRM_Provider
Consume_RS
worker thread (FIFO)
Consume_RS
consumer and locker threads
OUT_JMS_QUEUE
JMS queue
OMS
Message 1
Message 1
time1
Resequencer table, messages 1 and 2 belong to the same group (A)
A1
A2
Line items
Every message contains all order line items
Line item ID: X
status: open
Message 2
Message 2
time2
time3
time4
Line items
Same order as message 1 with one or more line item statuses updated.
Line item ID: X
status: completed
CRM Message (A1)
Message (A1)
CRM Message (A2)
Message (A2)
A worker thread for group A makes sure the messages A1 and A2 are processed in sequence.
Consumes messages from the JMS queue, groups them based on order ID, stores them in the resequencer table, and locks them for processing.
JMS queue stores incoming messages from OMS.