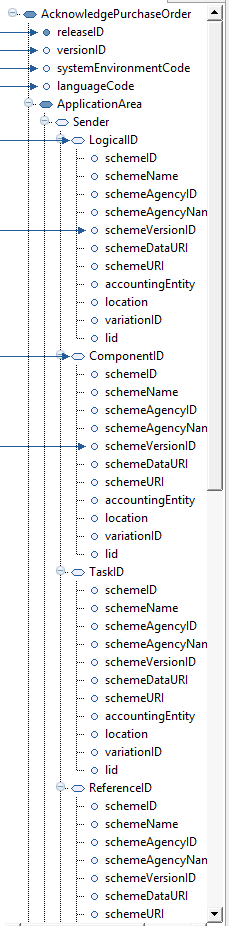
MEC tips and tricks – ION BOD – Acknowledgement and ConfirmBOD

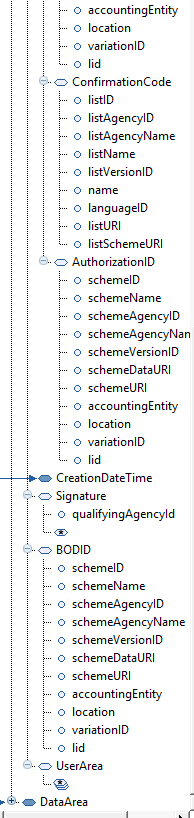
There are 2 different Acknowledgement flows. The first is when a message succeeds and the Acknowledgement is an accepted message type the second is flow is when a message fails and the Acknowledgement is a rejected message. Below is a description of knowledge gained so far with each of these Acknowledgement flows

# General outbound message maps

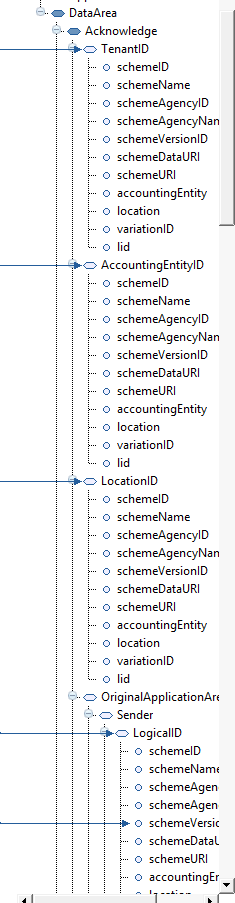
In general, all outbound headers going to ion contain nearly the exact same schema structure. The structure seems to always follow the following format. Below is a typical example. The Data area seems to be the only section that differs from one type of message to another. For example, the data area for acknowledgement messages differ from the data area structure of a process message but all acknowledgement structures appear to follow the same structure. Below are examples. I’ve created a schema for the acknowledgement message (AcknowledgeSBD\_APVoucher20150612) that contains the header and common acknowledgement data area elements for APVouchers. This schema could be used to copy paste into other newly created messages that require acknowledgements.

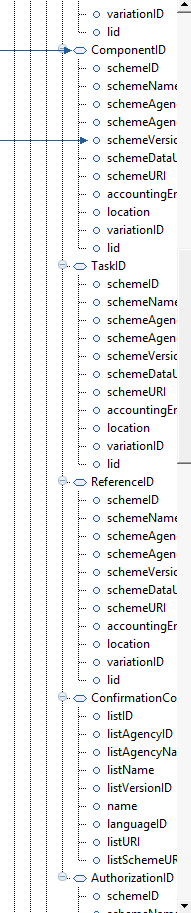
## Typical ION header

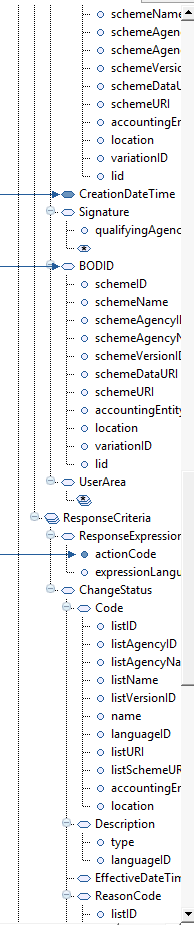


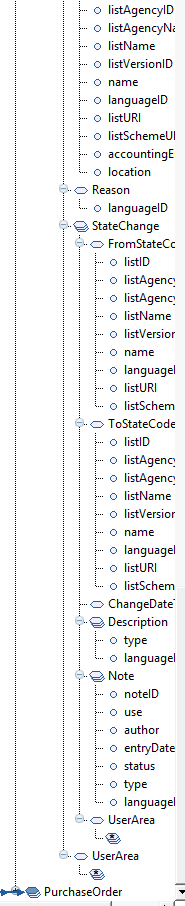


## Typical ION Acknowledgement data area







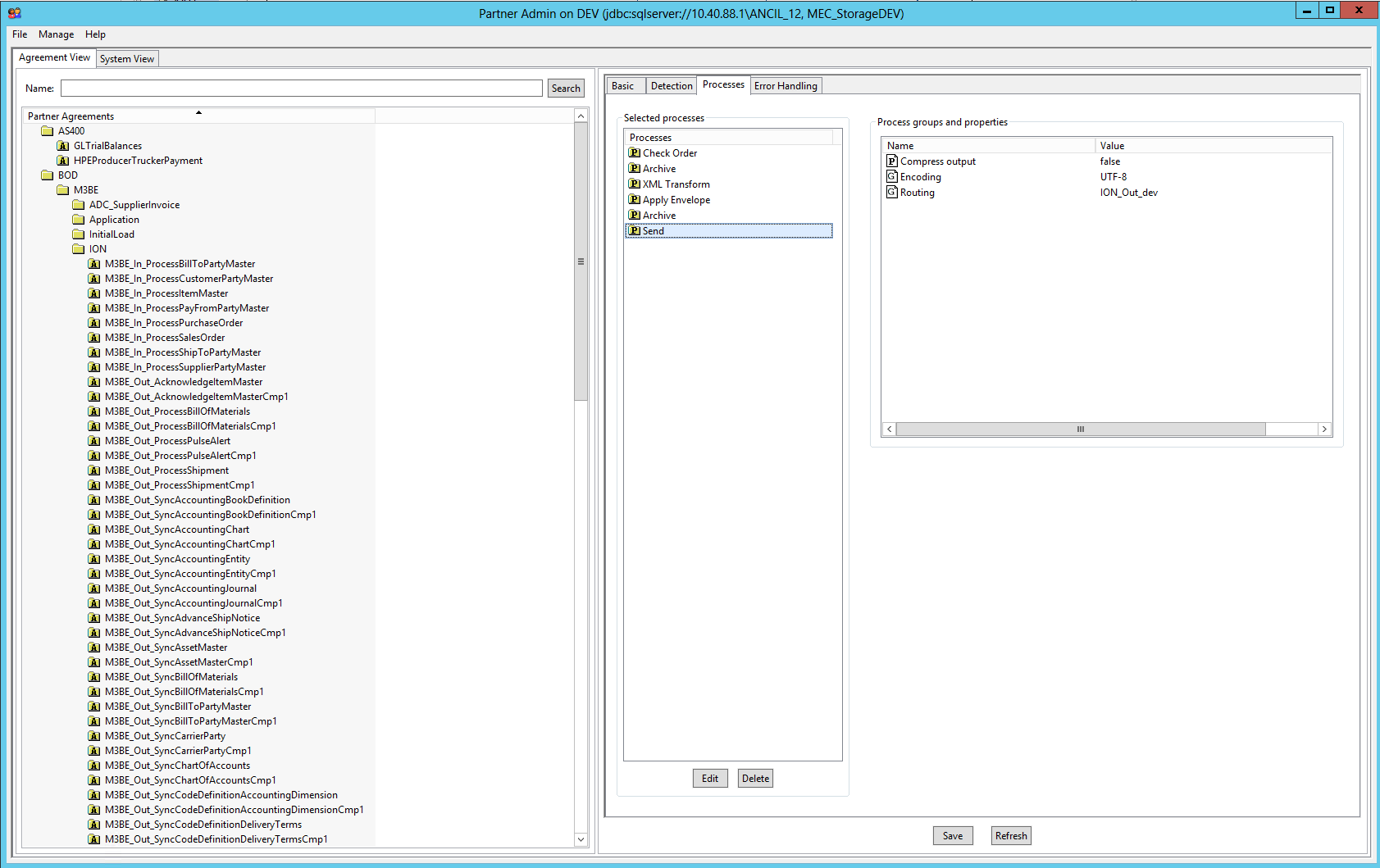


# Acknowledgement success

There are 2 types of outbound Acknowledgement success message “flows”. The first type appears to be coming out of M3 via an event hub trigger. There seems to be only one of these types that are installed on the system ‘M3BE\_Out\_AcknowledgeItemMaster’. The second type of acknowledgement success message comes out of MEC from an inbound message from ION. This type of acknowledgement is basically a ping back to the sender (ION) if the inbound mapping succeeds through the MEC mapping.

## Type 1 – success from M3 through Event hub trigger

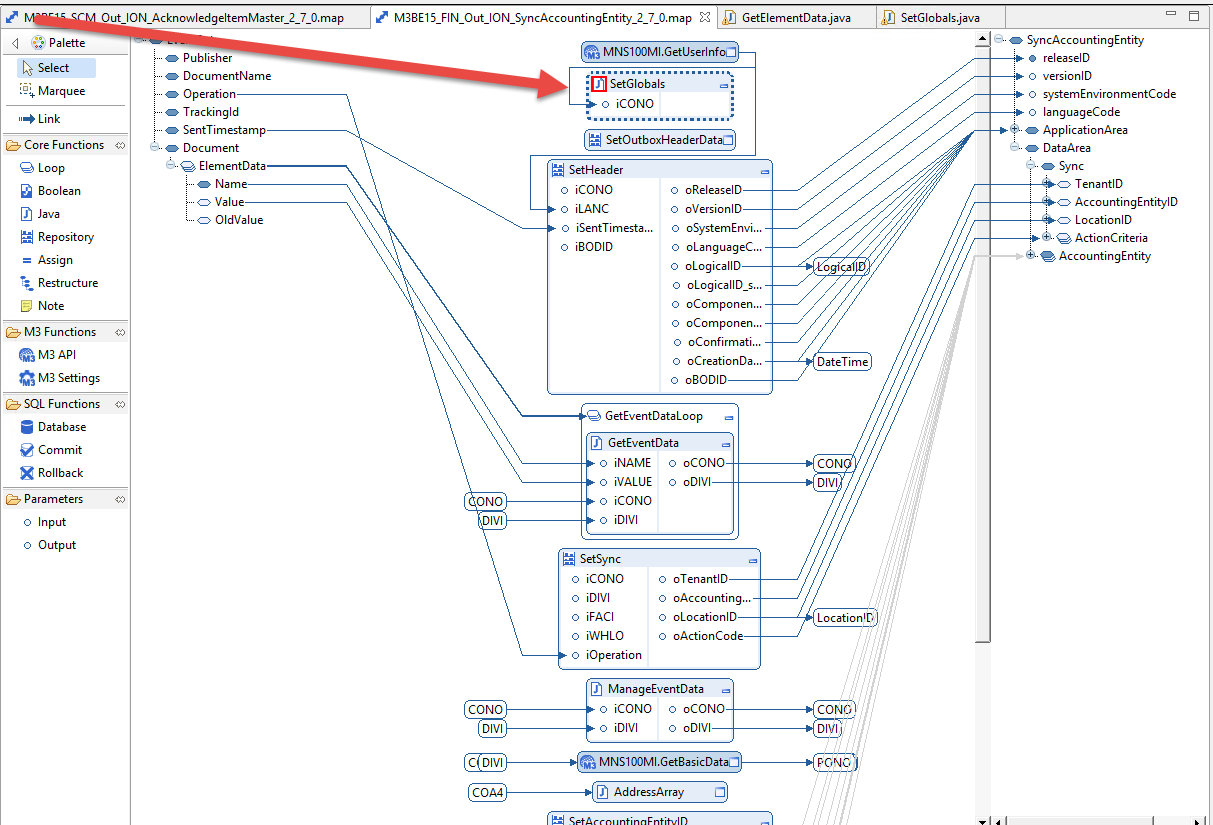
The only example that can be found of this type is ‘M3BE\_Out\_AcknowledgeItemMaster’. This agreement performs a check order process, transforms the message, applies an envelope to the message and sends the message out through ION\_Out\_dev. Below are the examples

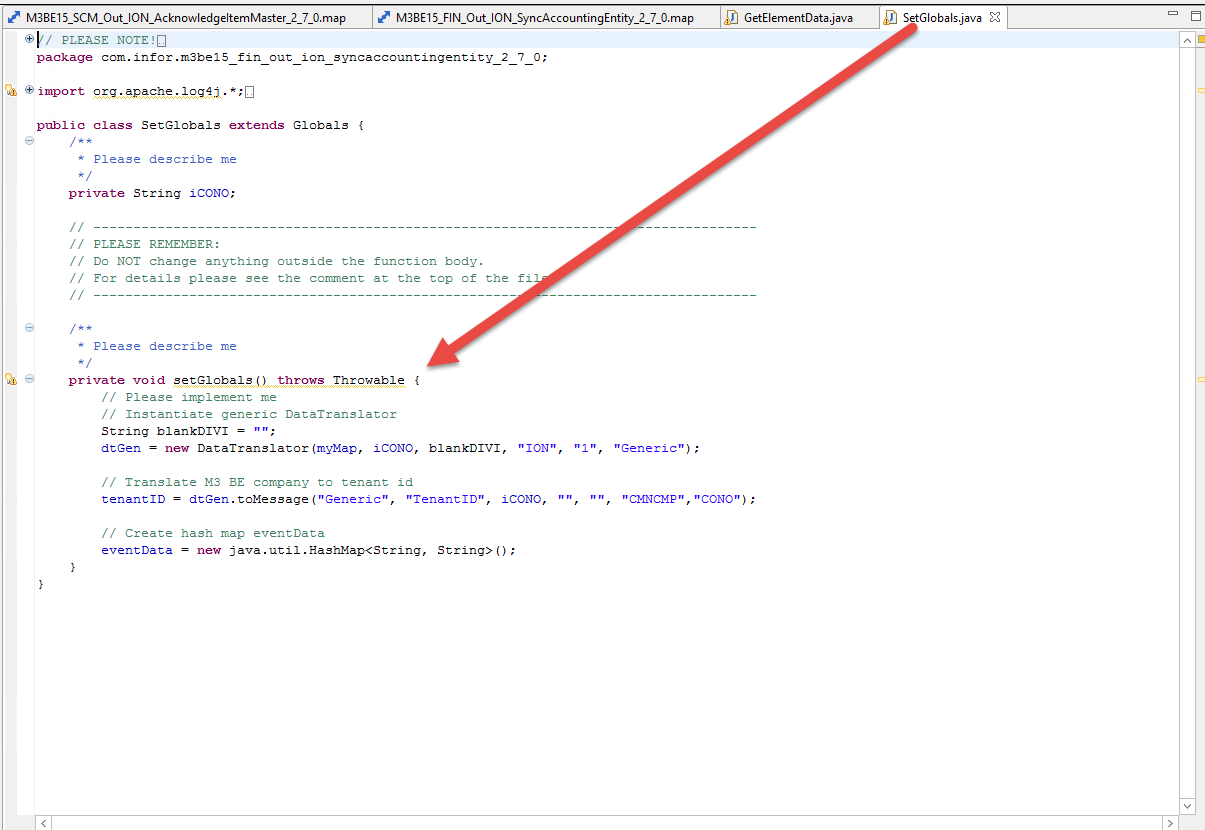


In the map some of the java functions that make up the Type 2 acknowledgement success are in this type but not all. The set globals function is similar but not exactly the same as the initialize function in the type 2. One type function to note is the Get Data Loop which loops through the name and value from the inbound message and uses an IF/ElseIF statement to populate the appropriate outbound variables.

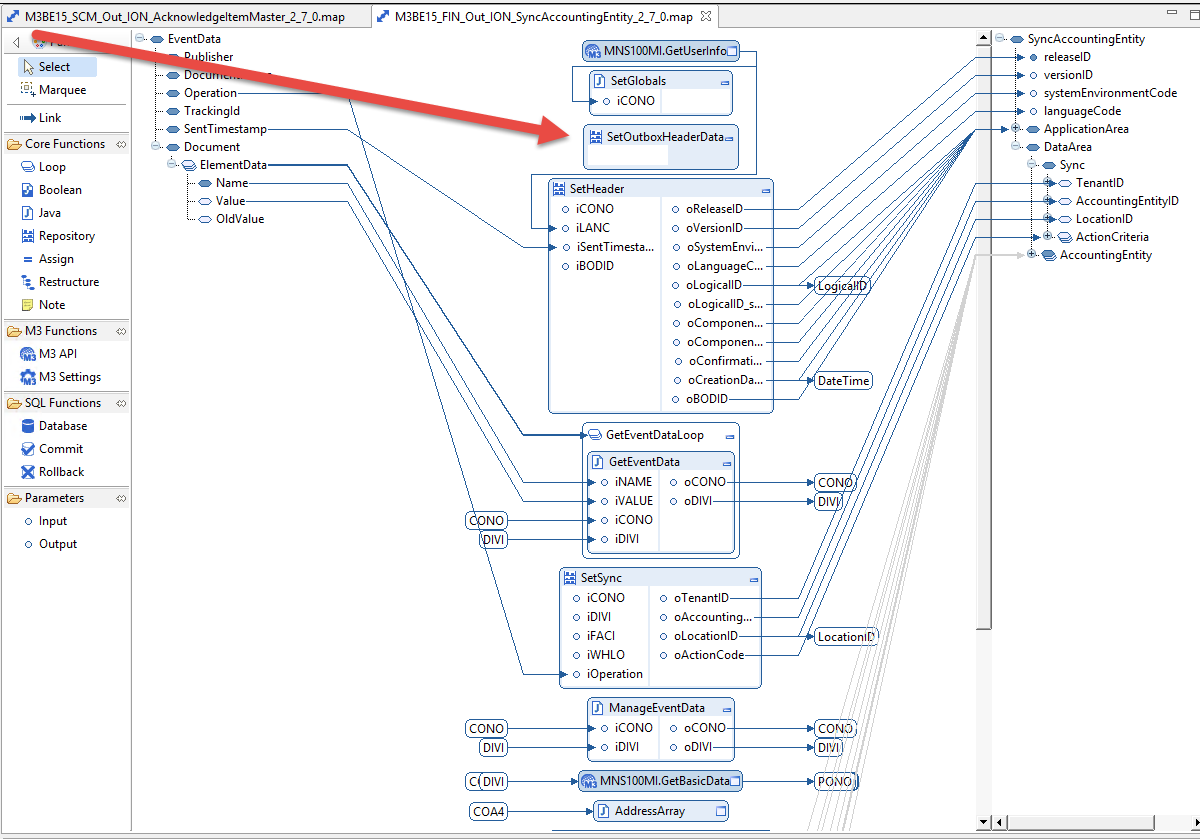
### SetGlobals

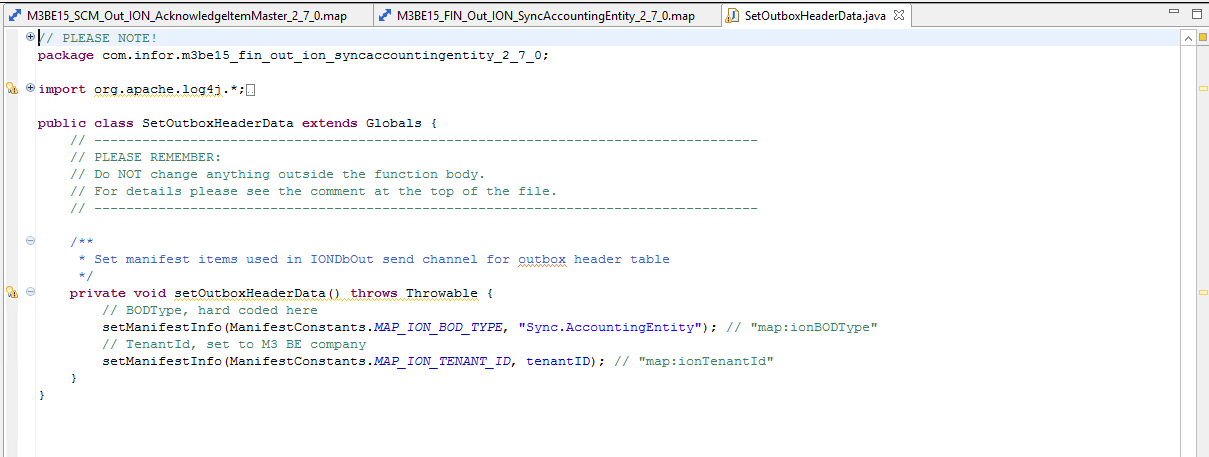
This is similar to the initialize function used in the type 2 messages. It does not set the correlationID since there is no need to correlate with another inbound message because this outbound message is coming from event hub and not an inbound message from ION



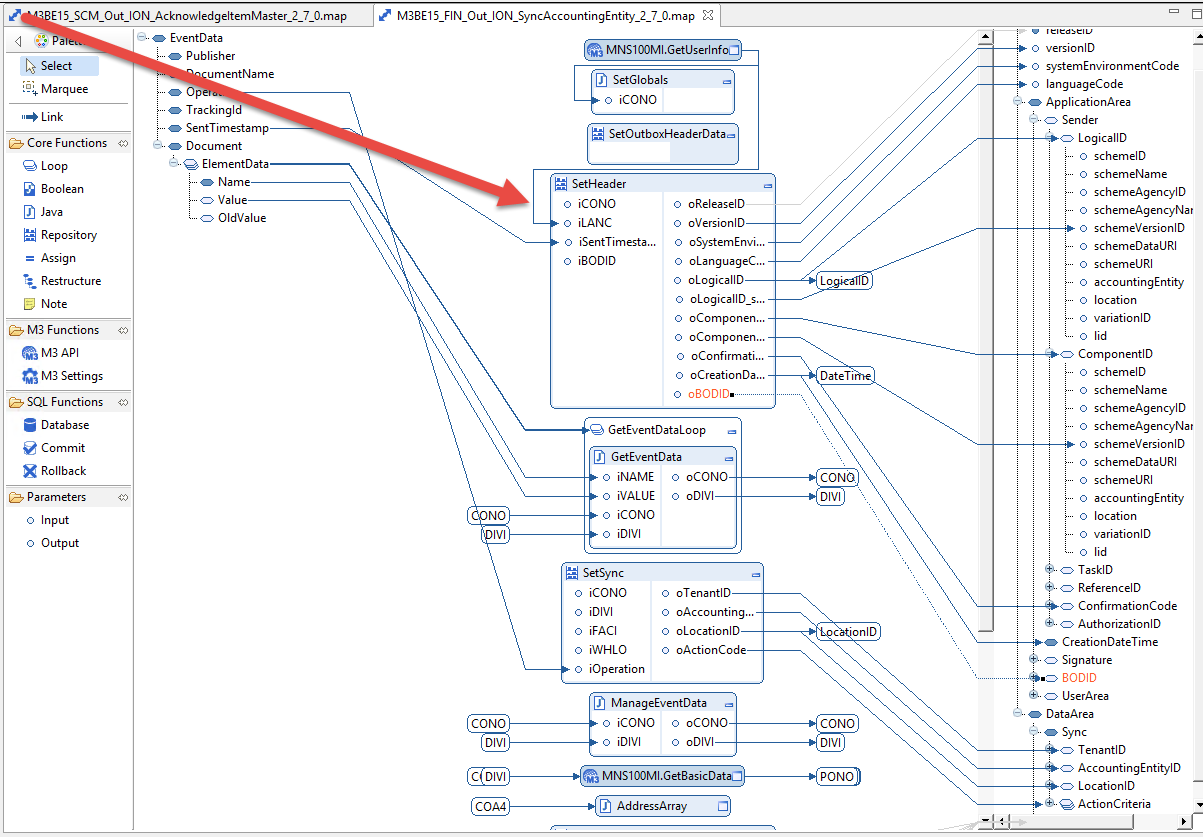


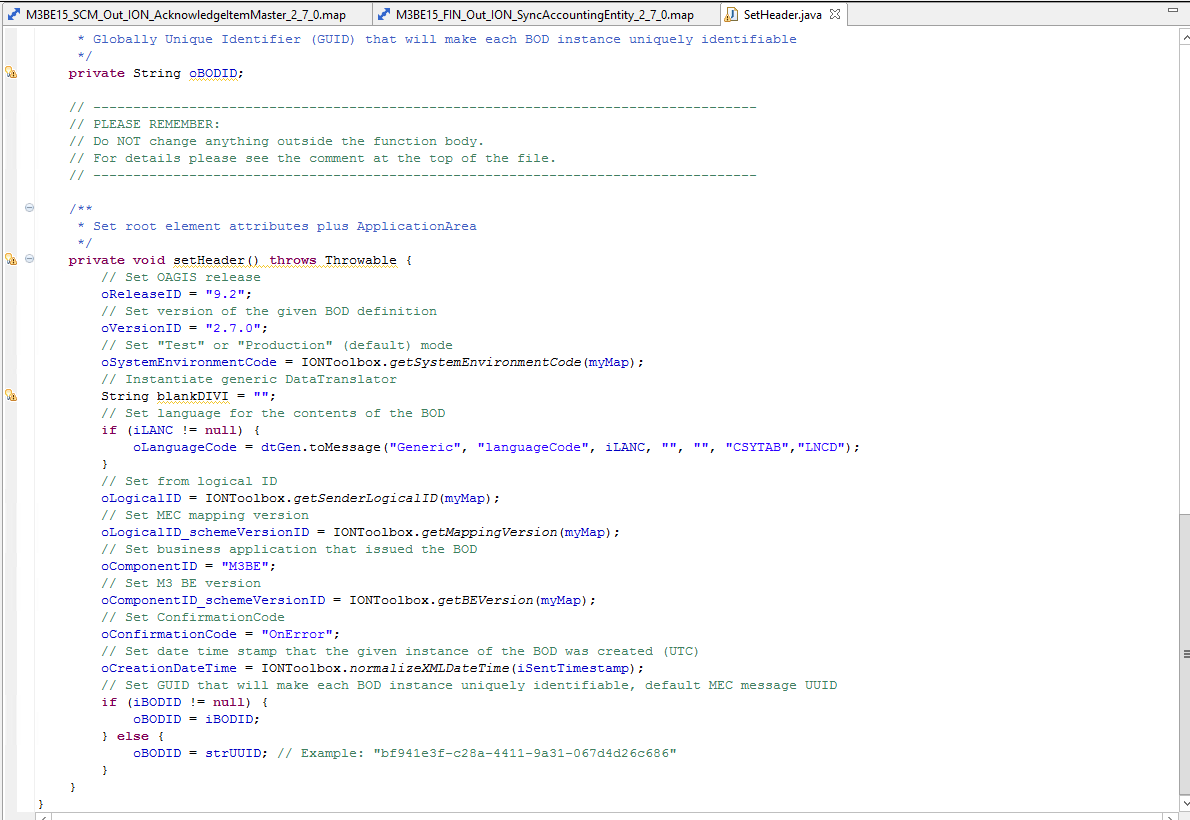
### SetOutboundHeaderData



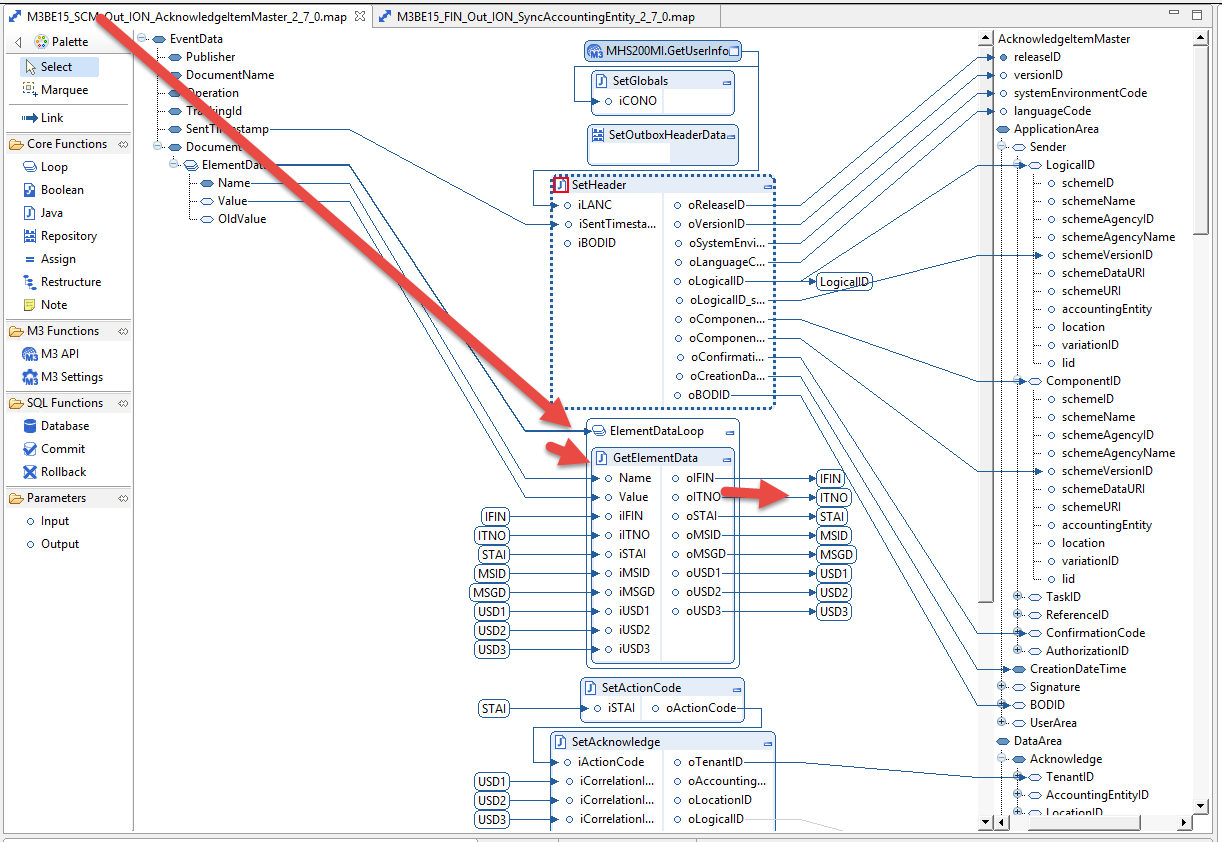


### SetHeader



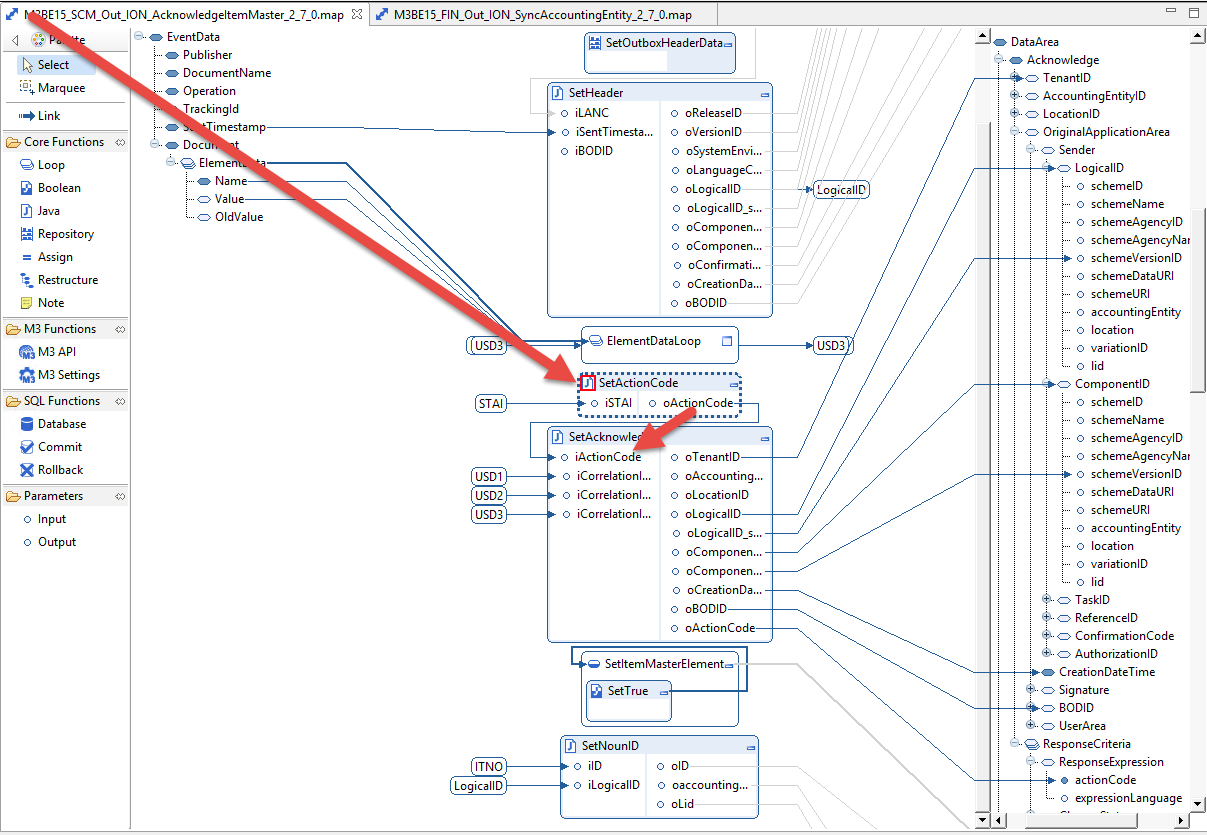


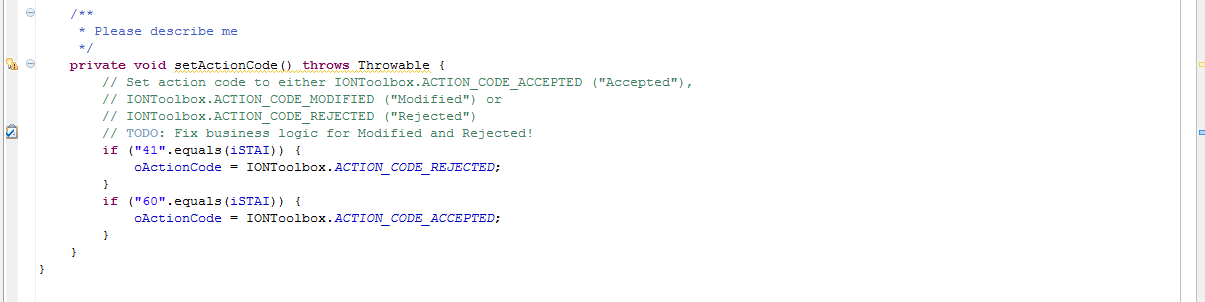
### GetEventDataLoop/GetEventData



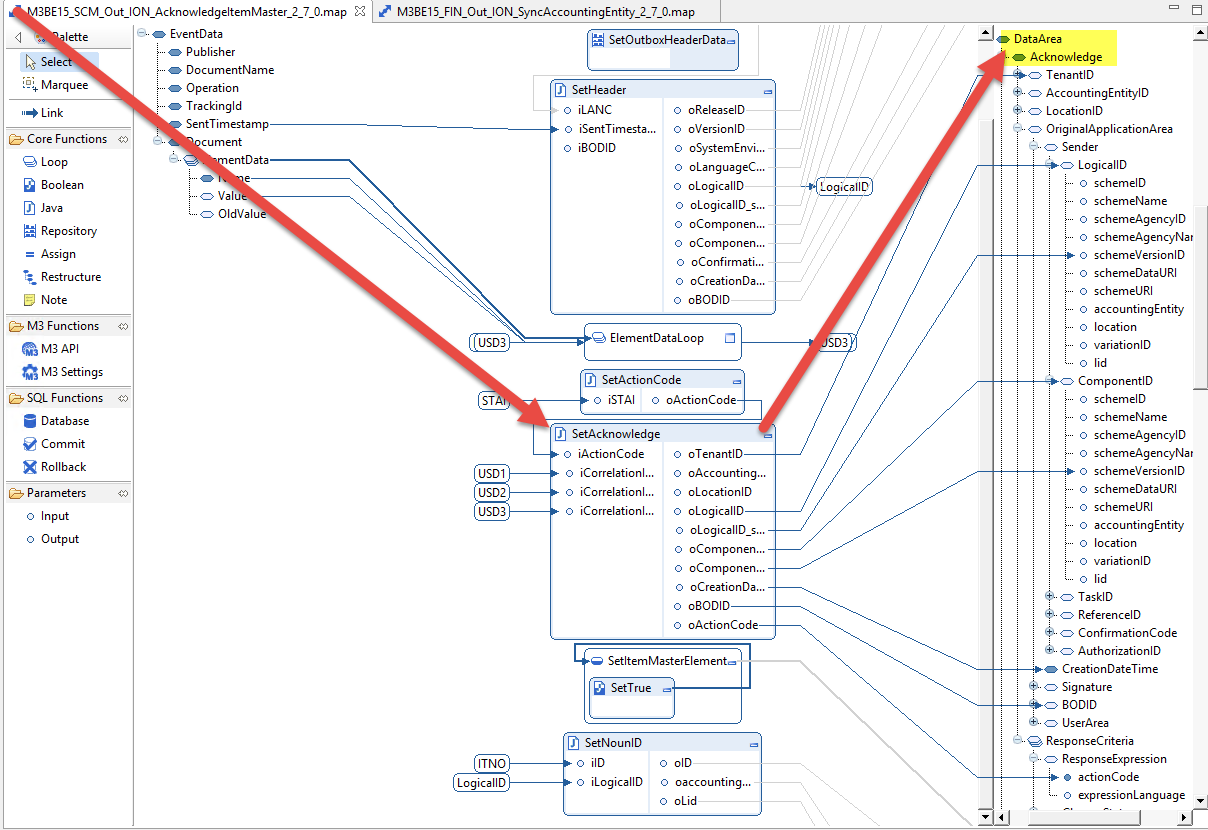


### SetActionCode





### SetAcknowledge

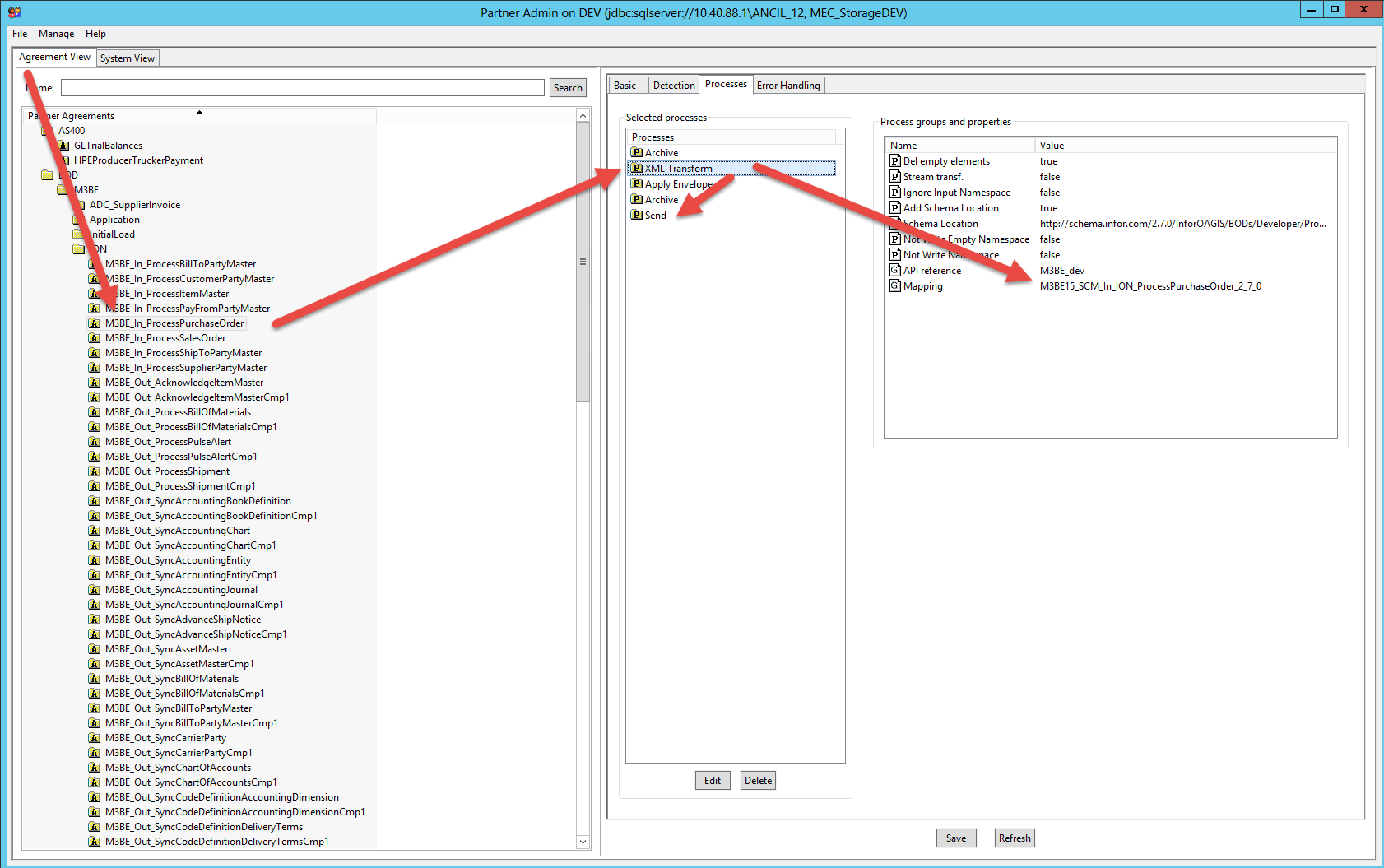


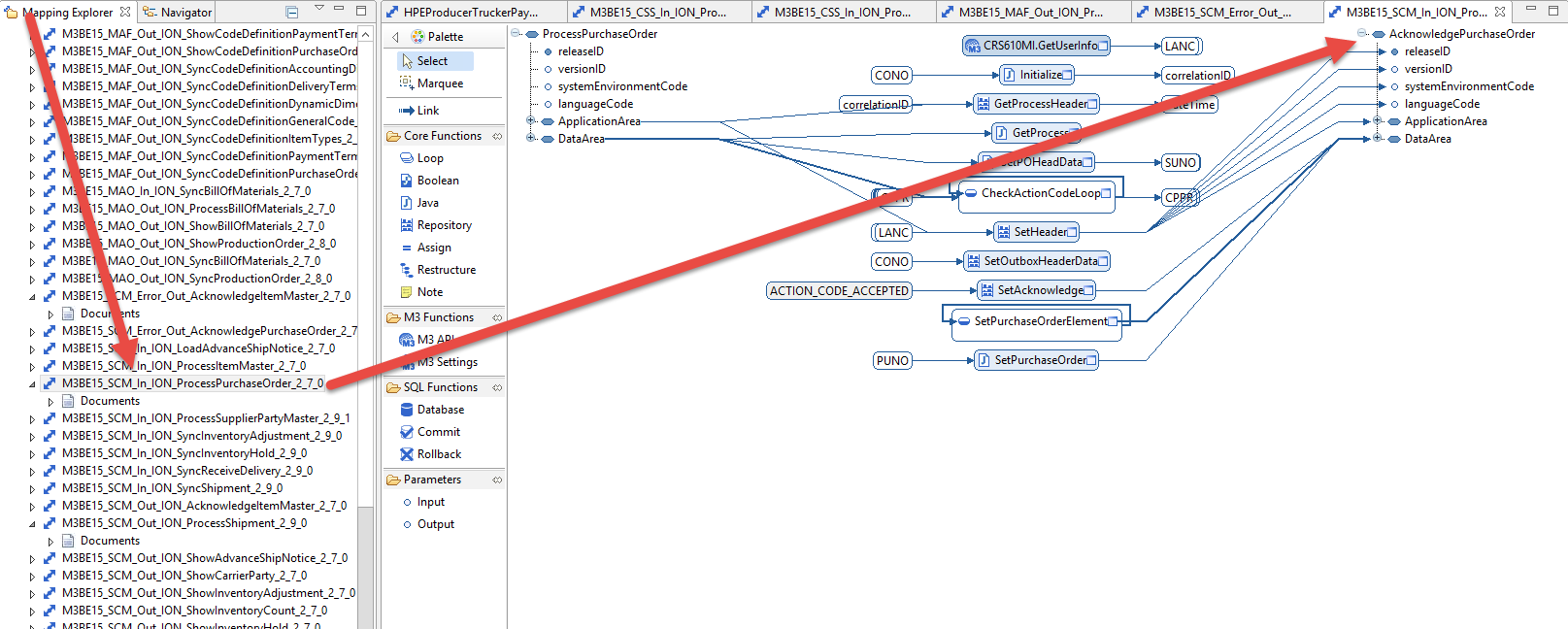


## Type 2 – success from an inbound ION message

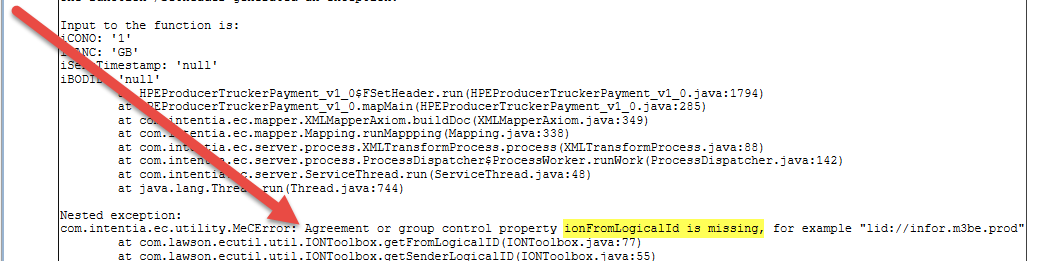
When an incoming message to M3 is successful the examples show that an outgoing acknowledgement message is generated in the map and sent out. From the partner admin, you can see this by noticing that after the xml transform process there is a send process. This indicates that MEC is generating an outbound message in the mapping. The send process normally uses the ION\_Out\_dev channel. This channel has certain requirements in the outbound message that are not well documented.

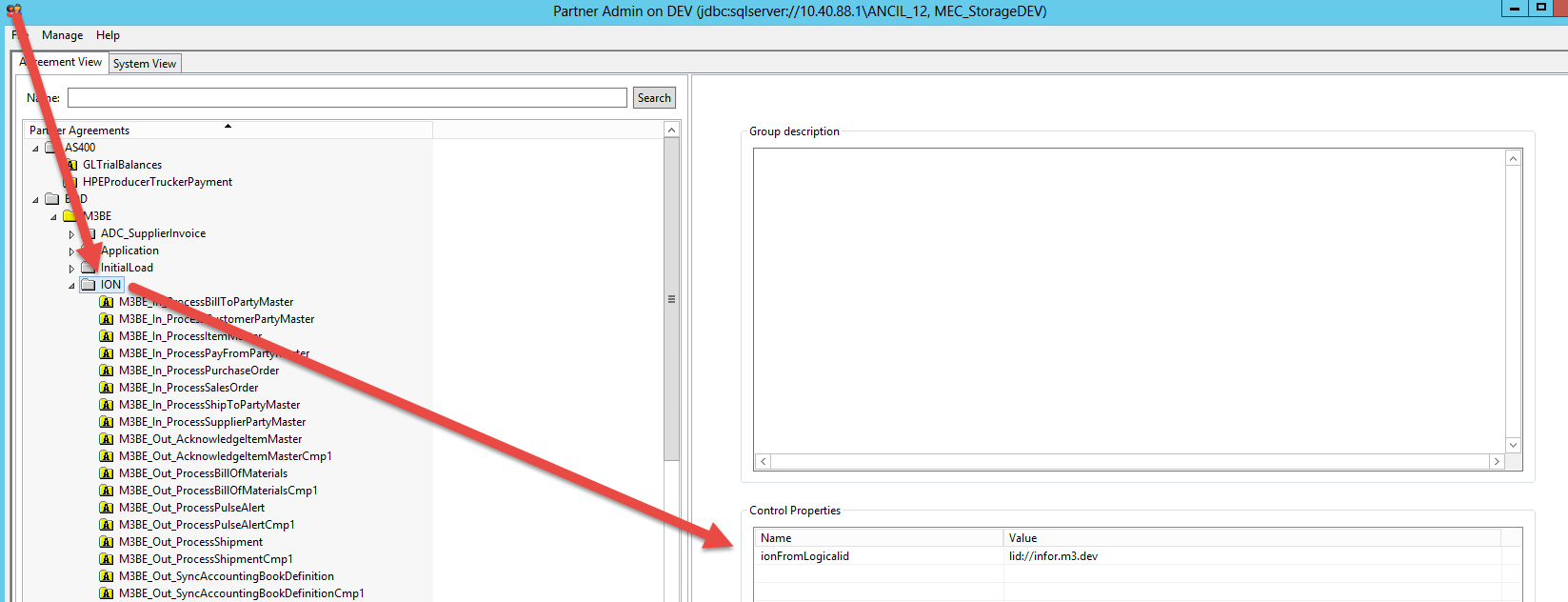
Here is an example of an agreement that has an outbound Acknowledgement message on the success of the inbound message and it’s map demonstrating the outbound acknowledgement map.



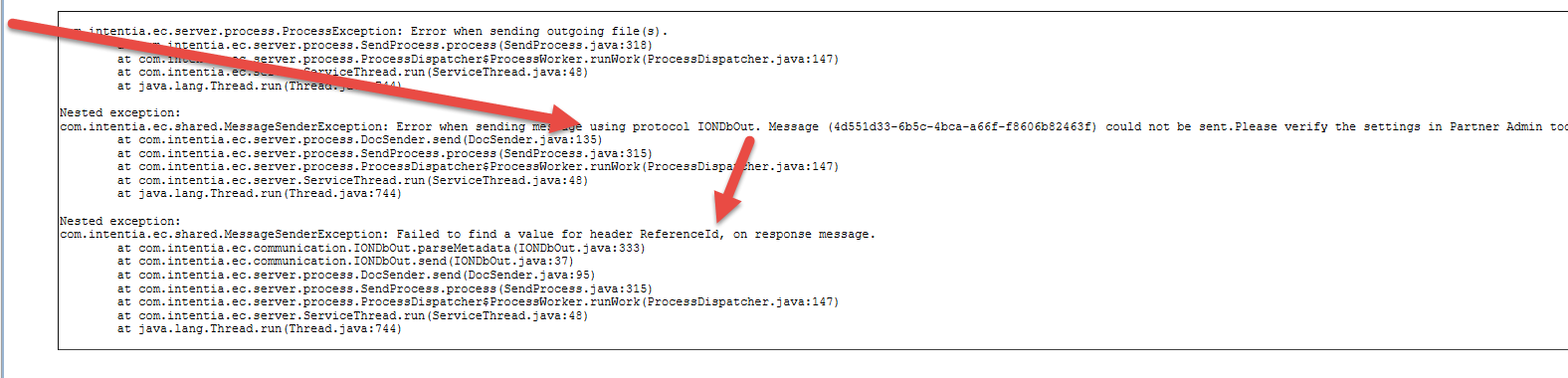


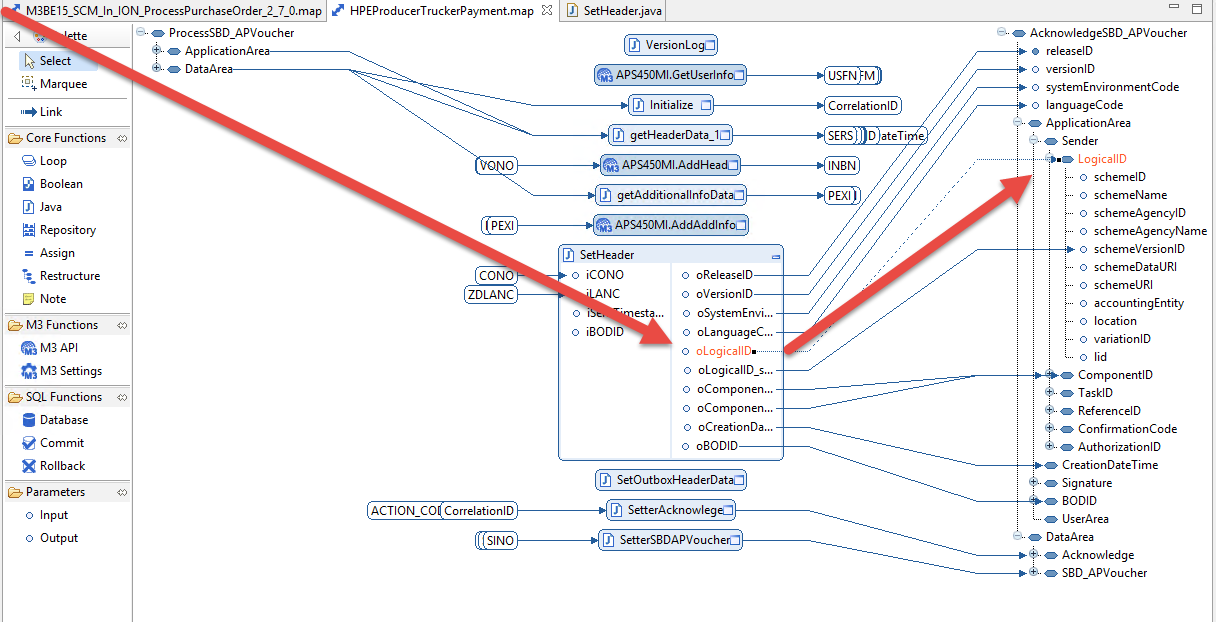
One item that is required in the acknowledgement message is ionFromLogicalId this is typically defaulted in the mapping from the control property in the agreement called ionFromLogicalid. This must be set at the agreement or the agreement group level. Also, notice that there is a LogicalId that is sent to the outbound message under /Acknowledge{messagename}/ApplcationArea/Sender/LogicalId. This is set from IONToolbox.getSenderLogicalID(myMap).

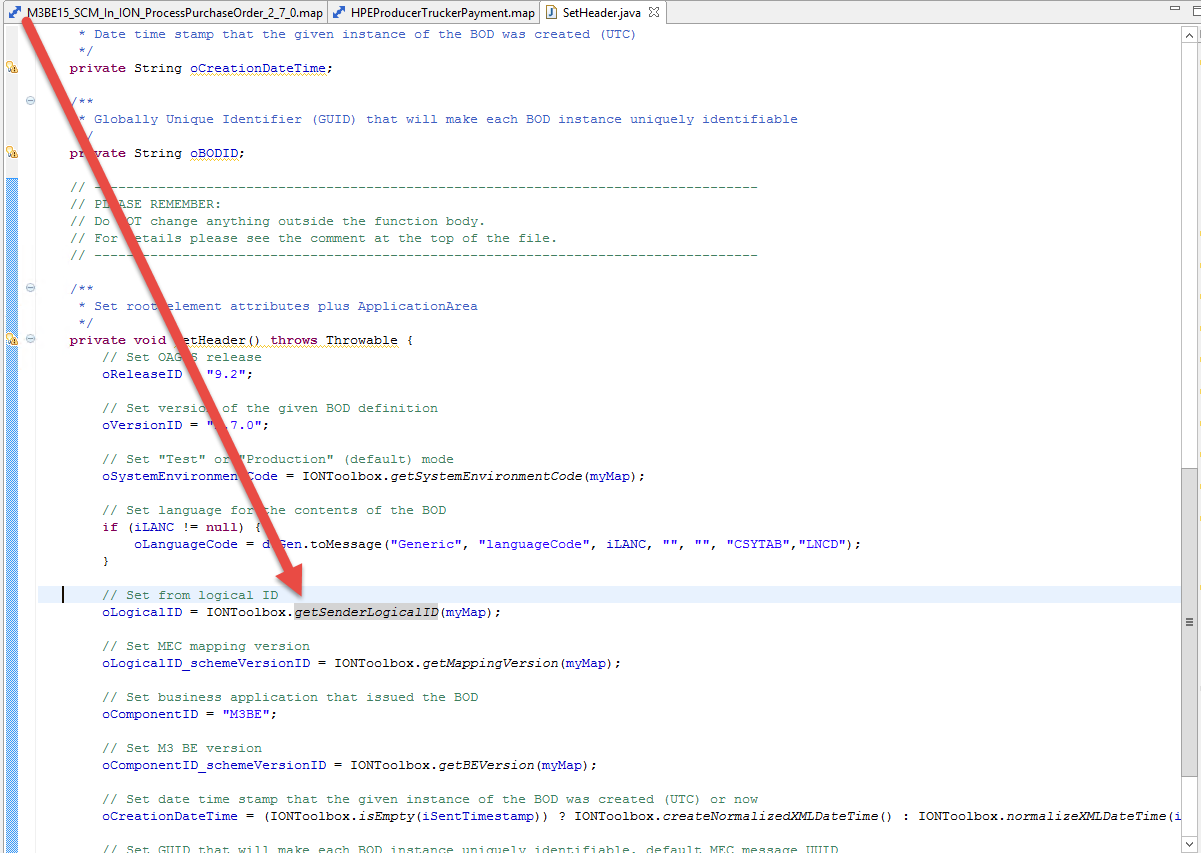




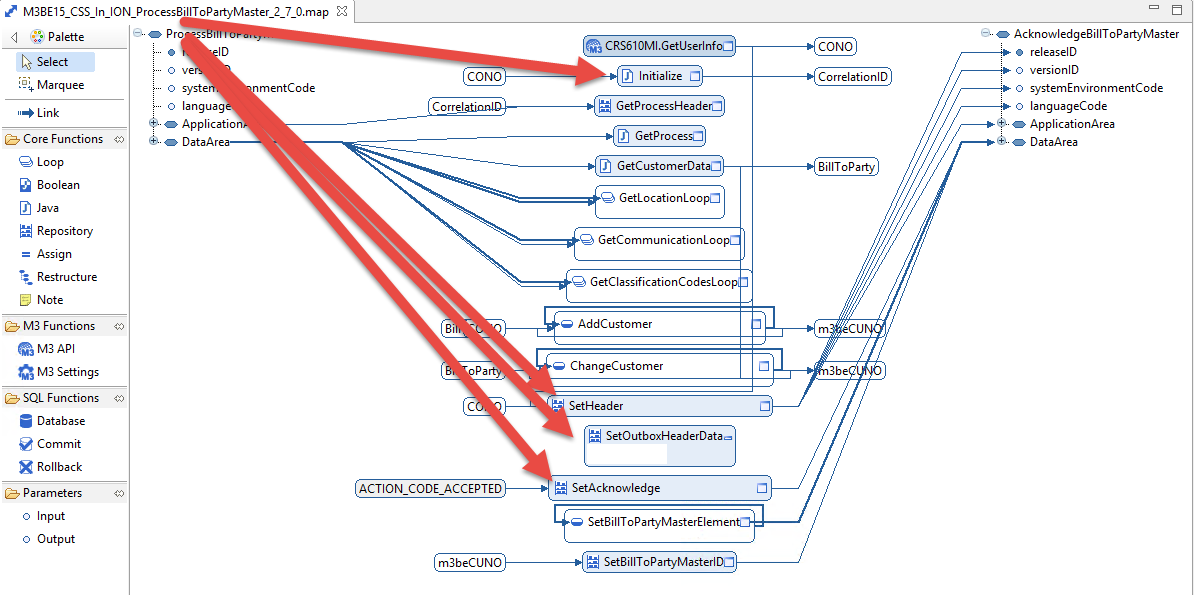
Also required is ReferenceId







In the map for process messages that have an acknowledgment, there are several java functions that always appear that are related to the outbound acknowledgement



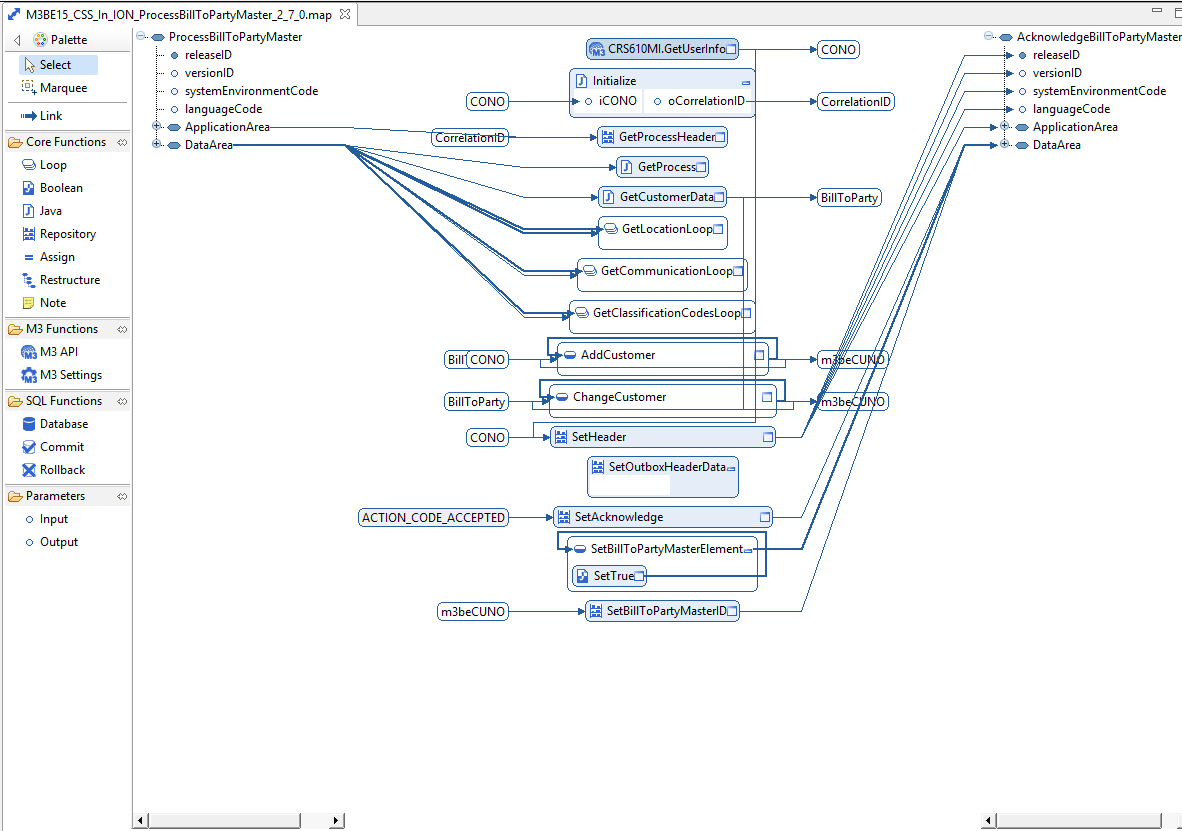
### Initialize

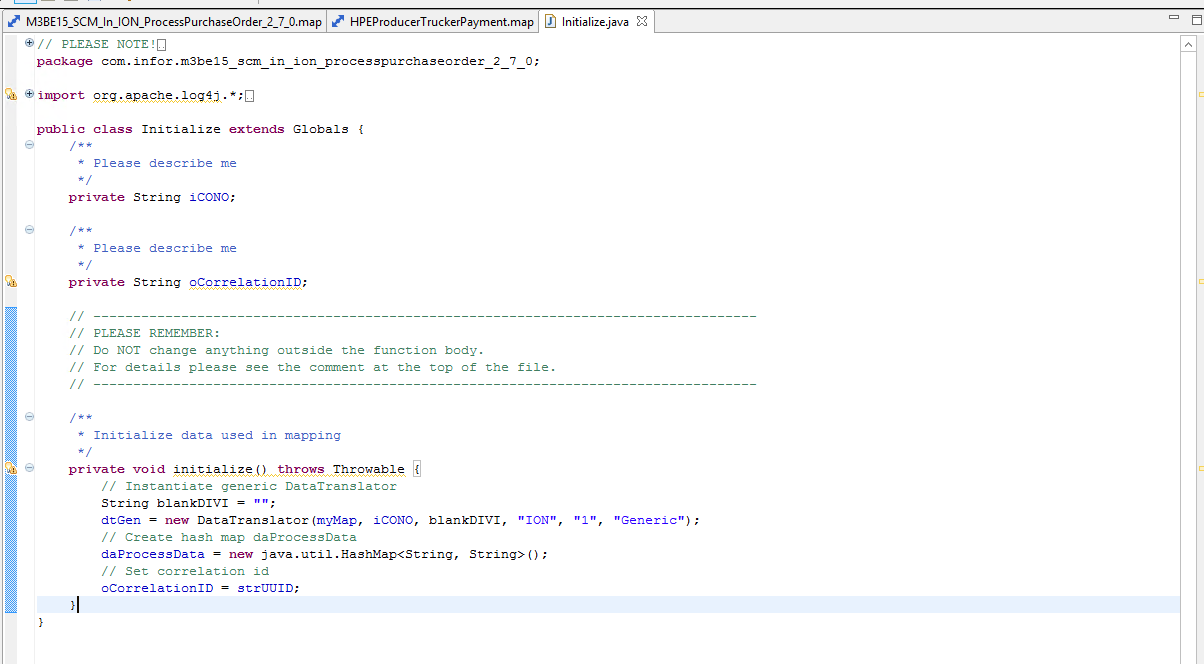
This function almost always sets up some datatranslator objects Hashmaps objects and oCorrelationID

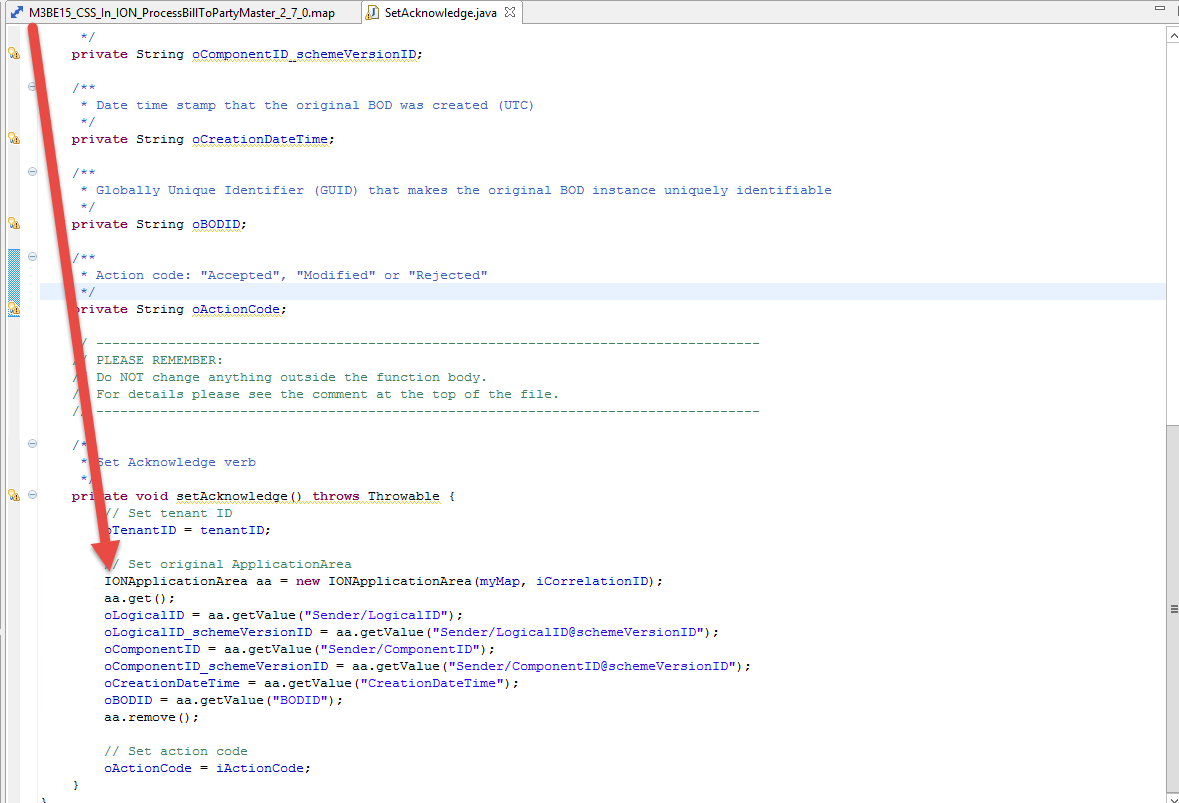
Data translator objects are used to cross walk values that are defined in LSO

Hashmaps are objects used to sort data in a particular order (date order, key value order, etc)

CorrelationID is set to Universal Unique Identifier used at the end of the map during the ‘SetAcknowledgment’ function for creating an instance of IONApplicationArea. IONApplicationArea is used to get the received message from logical ID, message ID and path-value mappings into this ApplicationArea from the MEC database with the given correlation ID as key.



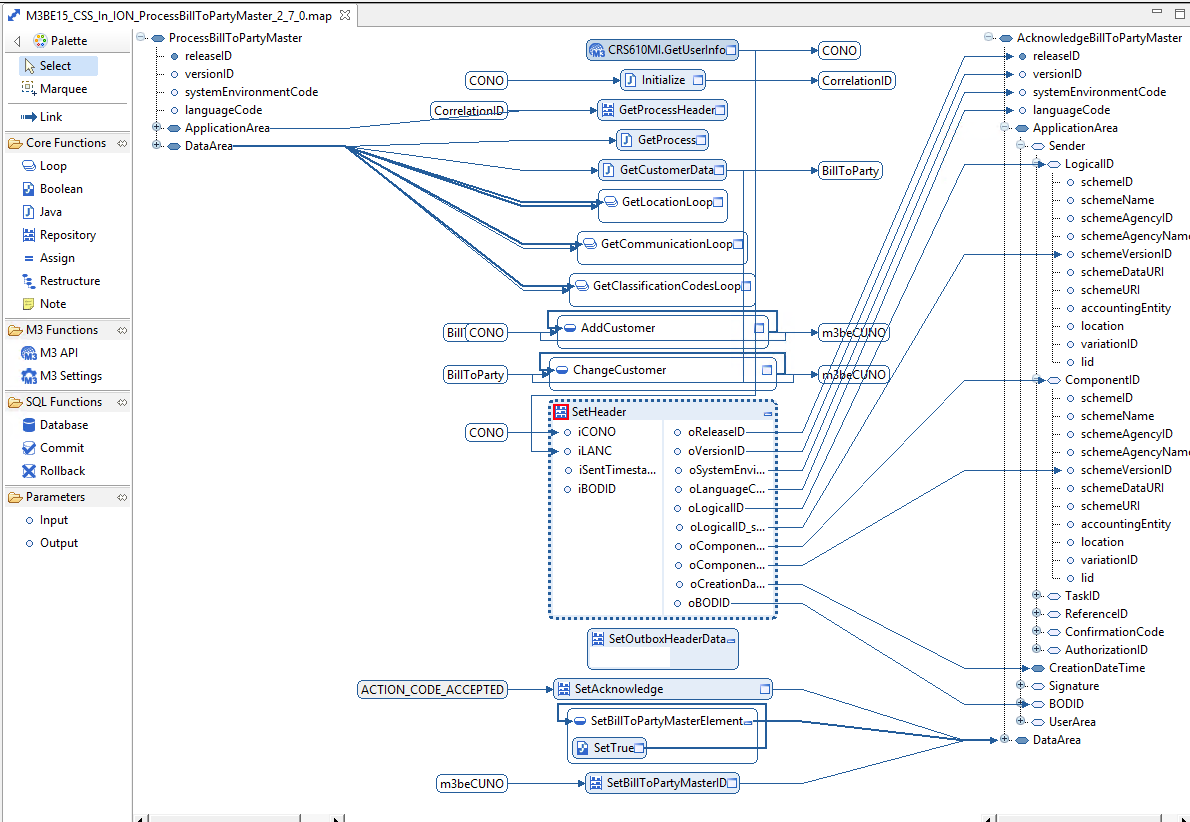


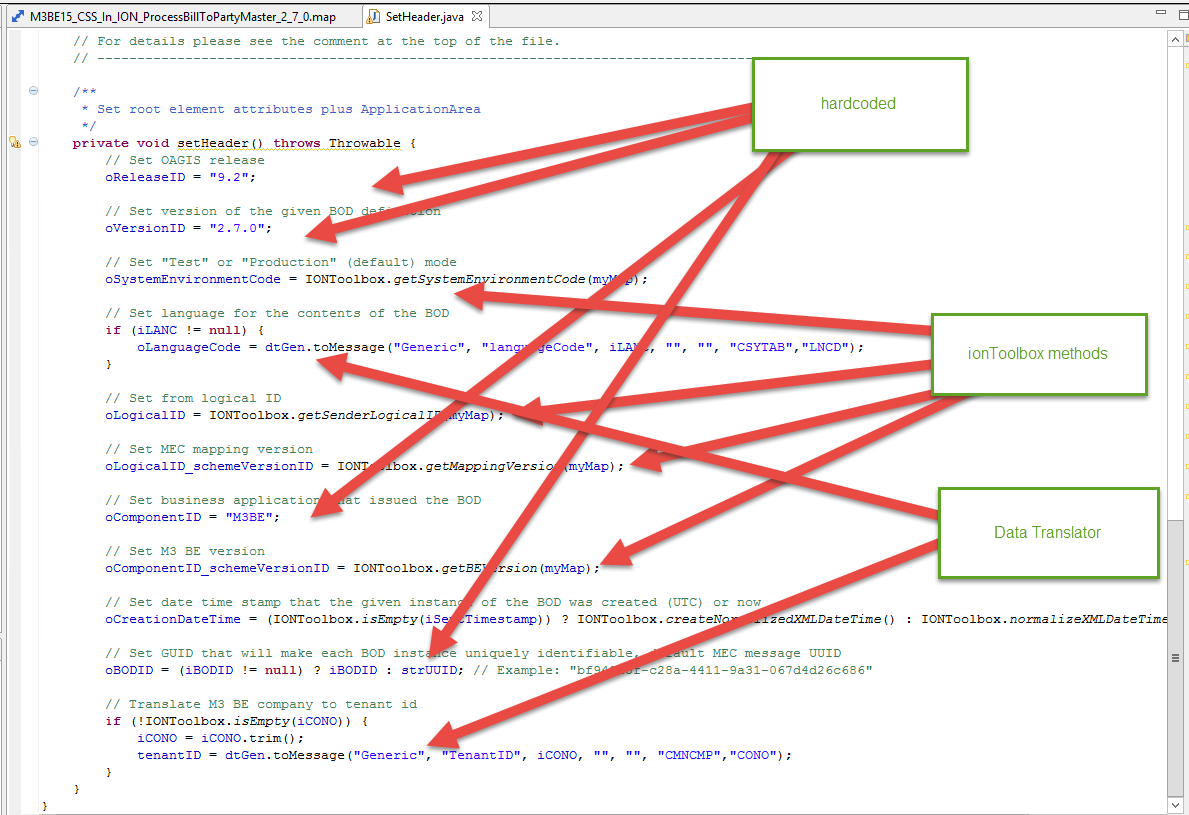


### Set Header

This section sets all the data for the top portion (before the DataArea) of the Acknowledgement message.

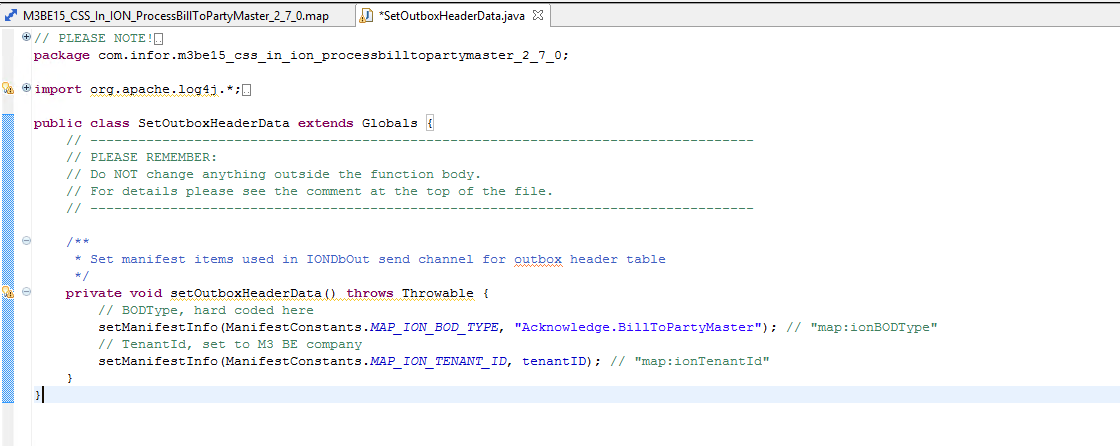
Many of the values are typically hardcoded, set by a IONToolbox method or use a data translator.





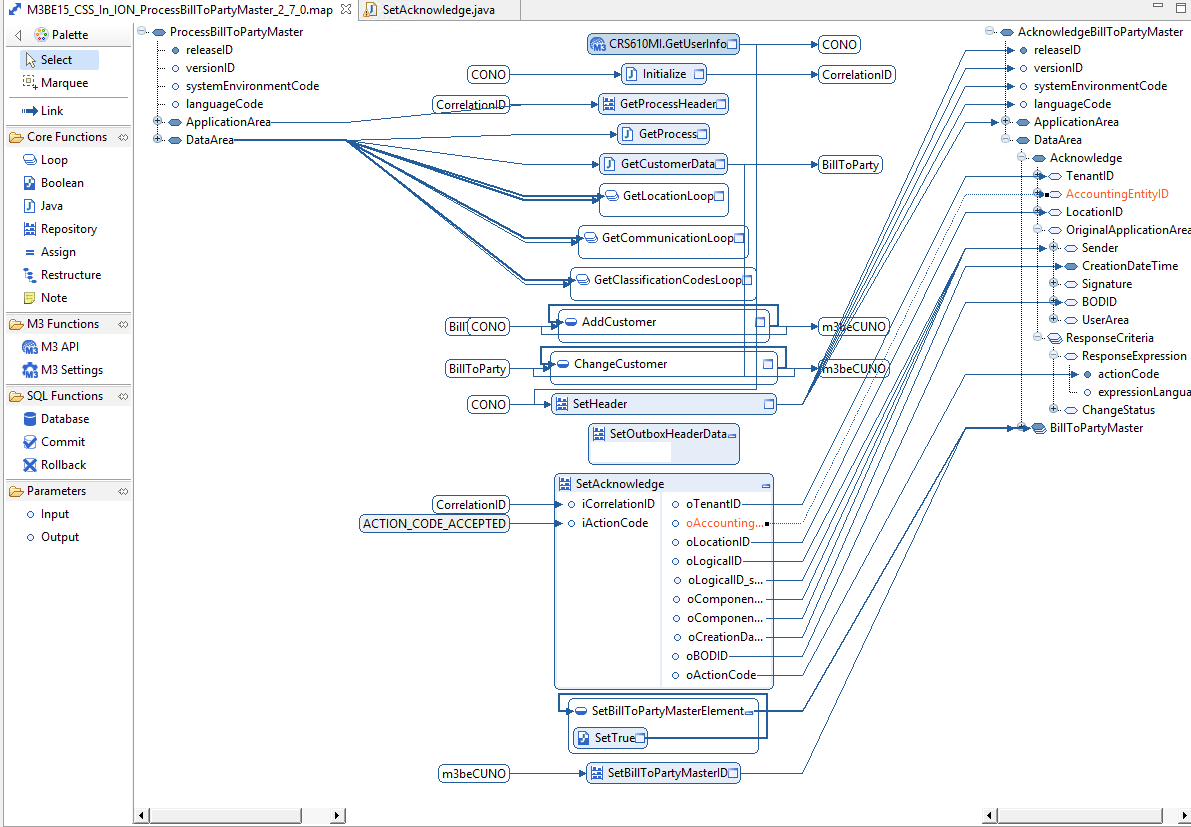
### Set Outbox header data

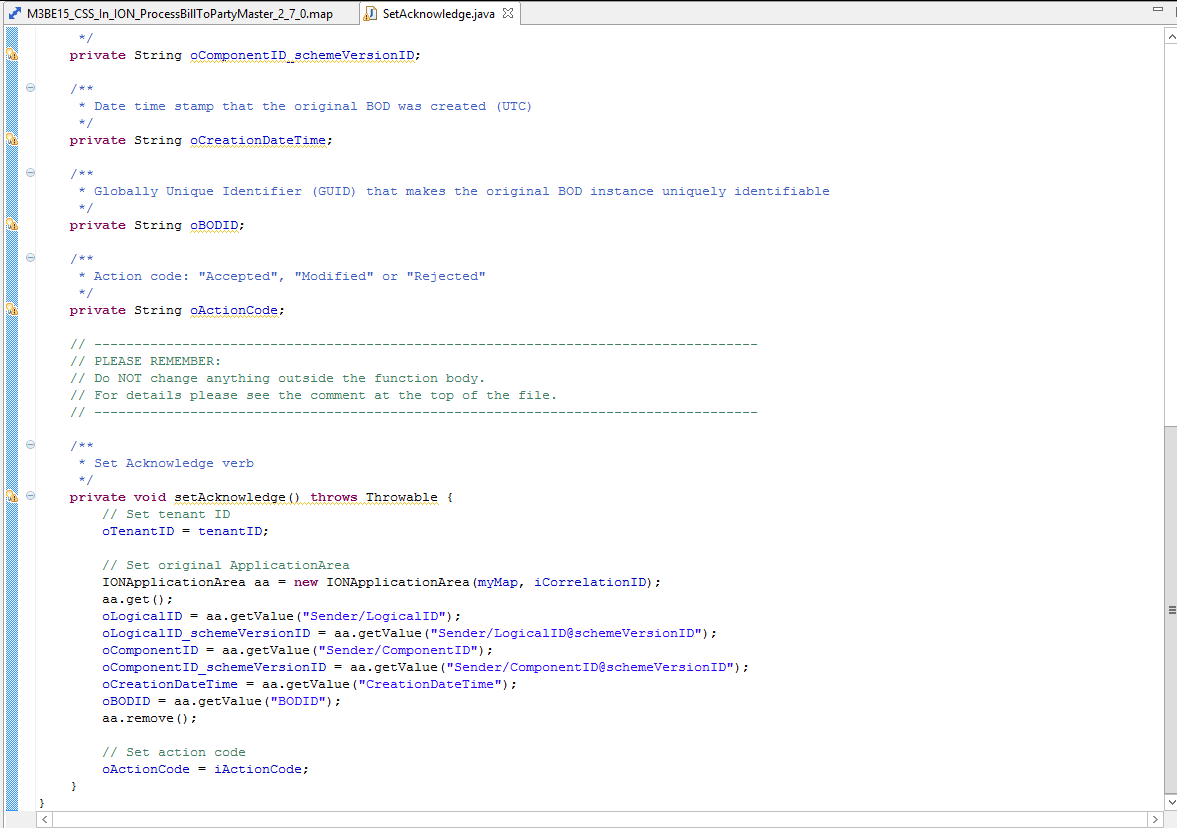
This section sets the manifest data for the ion outbox?



### Set Acknowledge

This java function sets the values that go into the DataArea/Acknowledge. These values come from IONApplicationArea class using the get, getCalue and remove methods. The java code creates an instance of IONApplicationArea and then uses that object to call the methods and set the outbound variables.





# Acknowledgement failure

When an incoming message to M3 is unsuccessful due to errors the examples show that an outgoing acknowledgement message is generated via a confirm process. in the map and sent out. From the partner admin, you can see this by noticing that after the xml transform process there is a send process. This indicates that MEC is generating an outbound message in the mapping. The send process normally uses the ION\_Out\_dev channel. This channel has certain requirements in the outbound message that are not well documented.

Here is an example of an agreement that has an outbound Acknowledgement message on the success of the inbound message and it’s map demonstrating the outbound acknowledgement map.