Collaboration Cards (Sample and Template)

Special cases of Specification Cards are so-called Collaboration Cards. They allow to specify how two or more human and/or non-human Actors are to interact in order to support some specific workflow. Here is an example taken from the system stack of a telecommunications provider:

A Sample EAI Workflow

Where Enterprise Application Integration (EAI) is based on a simple Message Broker, e.g. a JMS Server or a TUXEDO Bus, Adapters are needed to implement Use Case specific work.

In the following Sample Specification this Use Case is about

- A_ **OrderMS** : an Order Management System,
- A_ **OProc** : an Order Processing System,
- A_ **ResMS** : a Resource Management System,
- A_ **GIS** : a Geographic Information Management System
- and Helper Components such as
  - C_ **OrderMS_GUI** : a Dialog Interface to the Order Management System
  - C_ **Poller** : a TUXEDO process
  - C_ **Manager**
  - C_ **Worker**
  - C_ **Adapters_and_Extractors**

EAI Workflow is meant to transport an Order from **OrderMS** to **OProc** thereby contacting – for either validation or completion of the order – additional systems such as, e.g. **ResMS**:

Despite the fact that this workflow involves no less than 9 components, it was easy to describe in form of the following Collaboration Card which is certainly easy enough to understand:

C_OrderMS_GUI allows an **OrderMS Dialog** supporting the user to enter a new order. As soon as the user finishes entering enough order details, OrderMS will store the new **Business Event** in the OrderMS Database and will mark it as being in status **committed**. Via a DB trigger, a record will be added to the OrderMS BUSINESS_EVENT table to indicate that a new event is ready to be	

transported to OProc.

Remark: Committing an event that is an order is implemented by sending an ADDORDER message to the Workflow Manager. Orders in this sense are represented by records in the SW_OPPORTUNITY table. (ADDORDER is to be understood as “add order to OProc”).

A **Poller** process (TxS_VA_BEVEN_POLLER)

is to poll the Transaction Database, is to detect the new record, and is then to feed it to a TUXEDO queue named BEVENT.

A **Manager** process (TxS_VA_MANAGER)

is to read the queue, is to find the order there, and is then to update the **Workflow Table**, i.e.

- A serial number (= SR Number) will be created to have a key for identifying this Business Event instance,
- the workflow necessary to process the order will be broken down into so-called Work Items (each of them being a sequence of TUXEDO Services),
- and the Manager is to create records each of them describing one such service activation and also to which Work Item it belongs.

A Work Item is a complete description of the – fully automated – activity bringing the order from one status to a following one (see the document “Order Statuses” for a spec of all possible status values).

The purpose is to collect data from TP systems – such as GIS and ResMS – to do more validation. Some Work Items even have to feed data to ResMS.

As soon as the Manger updates the Workflow Table the Manager will send a message (the SR Number) to one of many Worker Processes – all of them are based on the same executable.

The **Worker** process receiving the message will know there is a new job to process:

The Worker will read the Workflow Table to see which steps exactly shall be performed.

As long as there is at least one step not yet marked finished, there will be exactly one step that can be started now. The Worker will start it i.e. the Worker will work to finish the step. Work in this sense is calling TUXEDO services.

Whenever a service returns, the worker will update the Workflow Table (by changing the status flags therein indicating how far the

service succeeded: value 5 is for indicating failure) NOTE: Even for calls that succeed, the records will NOT be removed. OrderMS is informed about success or failure because the Worker will call • service VA_WRKFLWSTA in case of success, • service VA_WRKFLWSTF in case of failure. As far as a service to be called is to rely, e.g. on ResMS services, it is implemented by the **ResMS Adapter**. Other **Adapters** support data retrieval from the GIS (to see whether there is Home Zone coverage) and for customer credit checking. There is a particular Business Event named ADDORDER. It contains an item activating – via TxS_VA_WRKFLWMGR – the so called Extractor:	
	An **Extractor** process (TxS_VA_EXTRACTOR) is to • extract the order from the OrderMS Database, • and transform its presentation to have it available in form of a so-called TLV Tree.
A later item in the same ADDORDER event then feeds the Extractor-created TLV Tree to the OProc Adapter: The **OProc Adapter** • is to transform the content of the buffer back to a TLV Tree, • and is then to feed this tree node by node to OProc via this systems C++ API.	
	OProc is to store the order – in status **committed** – in the OProc database, is to initiate provisioning, and is also to generate data for Billing.

This example proves: Collaboration Cards can say more and are better to maintain than UML Interaction Diagrams (i.e. Sequence and Collaboration diagrams).