Getting started with ISO20022

Date: 2016-11-26 10:20:00

What is ISO20022?

ISO20022 is a universal financial messaging standard which covers messages received from or sent to financial institutions which relate to payments, securities, cards, foreign exchange (FX) and trade securities

Are there other ISO financial messaging standards?

ISO15022

The earlier ISO15022 standard covers similar domains to ISO20022. The main problem with ISO15022 is that it was designed prior to standardised messaging formats e.g. XML became popular. Given the ubiquity of XML, many processing languages have in-built ability to parse XML messages, whereas ISO15022 is confined to the financial services industry so is not supported.

Consumers who've had to make an international payment from their bank account may recall that this payment is called a SWIFT payment. SWIFT MT is based on the ISO15022 standard

ISO8583

ISO8583 covers payment card authorisation messages. This standard is commonly used to send information to and from your bank when you use a card terminal within a store to make a payment, or when you receive cash from an ATM (cash machine).

ISO20022 building blocks

ISO2022 can be split in to 3 main building blocks:-

1. Data model – Which is referred to as the ISO20022 business model
2. Data dictionary - Known as the ISO20022 repository
3. Message Definition Reports

ISO20022 Business Model

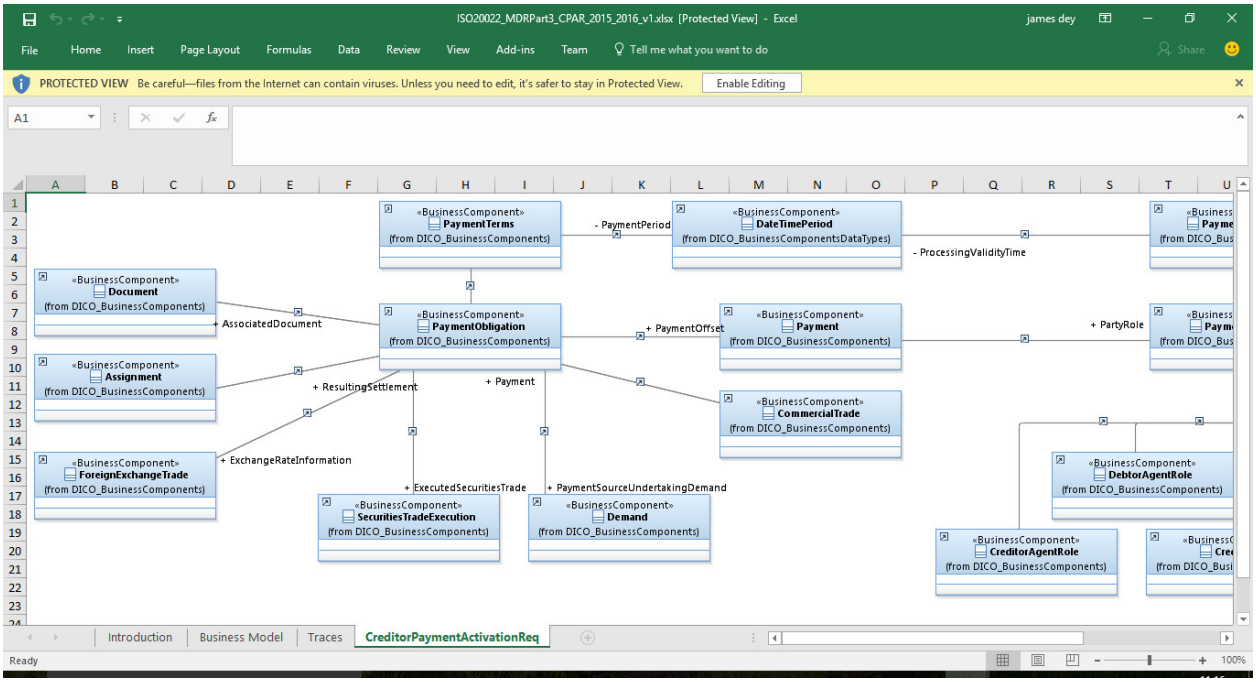
ISO20022 splits the data model by business domain (payments, securities etc.) and also has a separate section for common business components (aka uml classes).

Within the zip file for a particular domain, the business model can be split down in to lower level business capabilities. For example, in the payments business concepts zip file, there are 2 spreadsheets:-

1. Payments
2. Exceptions and Investigations

Within each Excel spreadsheet, there are then tabs which look at the relationships surrounding a particular business component.

The data model is provided as a set of UML class diagrams. Each uml class is referred to as a business component.



Unfortunately, the UML class diagrams are simply images embedded in an Excel spreadsheet, whereas a data architect's job would be a lot easier if they'd used the XML Interchange (XMI) format which is designed for sharing UML diagrams.

The business components don't have any attributes (aka business elements) depicted in the diagrams so you will need to search the repository in order to find this information or look at part 3 of the relevant message definition report (see section below)

Finally, each tab only gives you a partial view on the business model. To create a complete business model, you'll need to add the information stored in multiple tabs on multiple spreadsheets together.

ISO20022 Repository

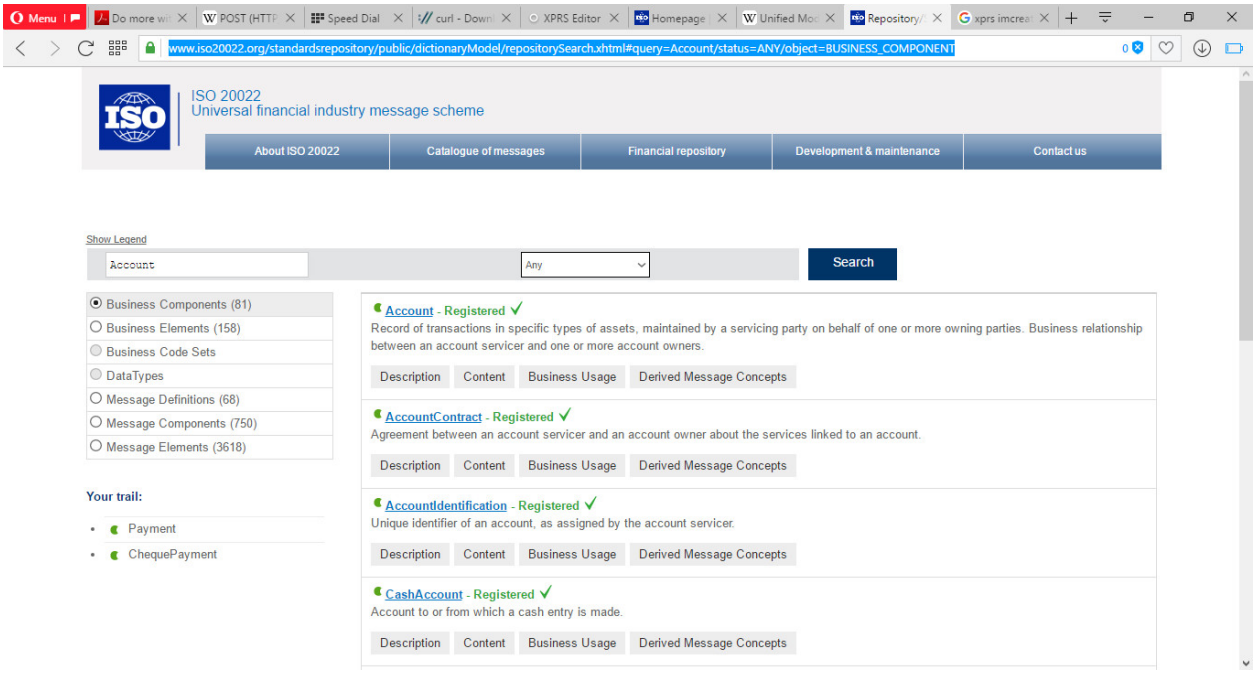
The online repository search is the best way of navigating between the ISO20022 business model and the Message Definition Reports. It provides information about:-

ISO20022 business model (business components, business elements, business codesets, datatypes)

Message Definition Reports (message components, message elements)

If you search, for example, for information about the "Account" business component, it will be returned as the top entry in your search list.

Note: You can check that it is a business component rather than a message component by hovering your cursor over the moon shaped icon to the left of the word "Account". You can also see on the left hand side that there are business elements, message definitions, message components and message elements that refer to "Account" in some shape or form.■



For a business component, you can also see that there are 4 tabs which will provide you with information.

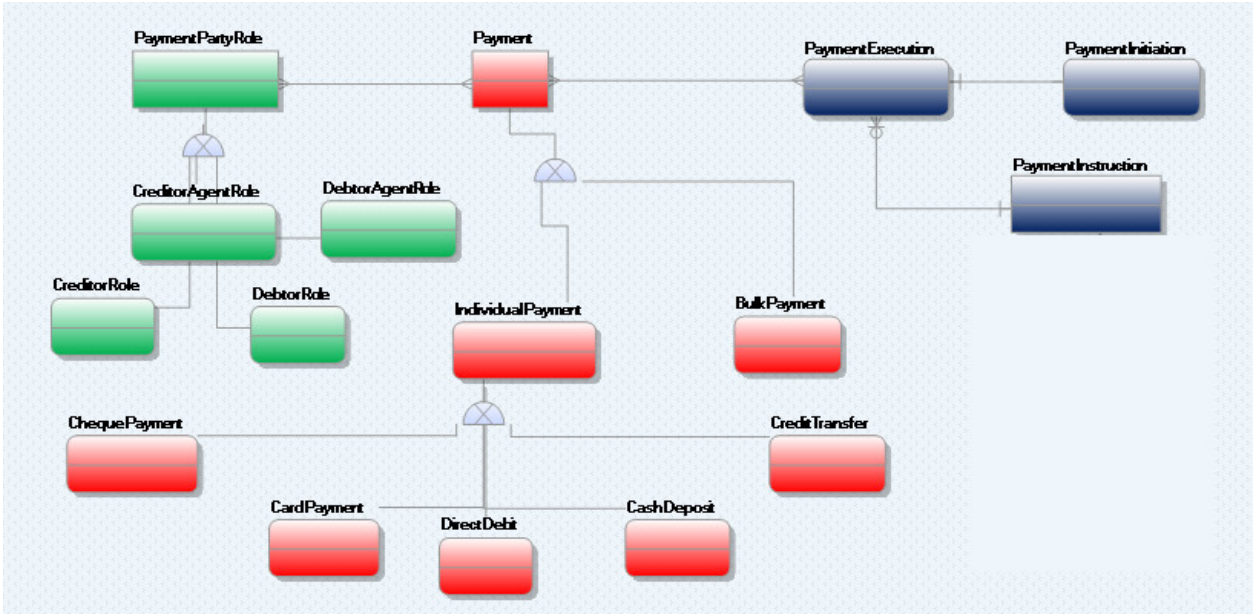
The description tab is a short description of the business component and may contain synonyms.

The content tab contains the business elements (attributes) of the business component.

The business usage tab shows the other business components that this business component has relationships with.

The derived message concepts tab shows which message components reference this business component.

Using the information obtained so far, you should be able to start to cobble together an entity-relationship diagram covering part of the ISO20022 business model. Here's an example of a part of payment covering the 4 main party roles, the different payment methods and the payment initiation and instructions (which make up the payment execution). Missing from the diagram are additional party roles, currency exchange, tax, the underlying payment obligation, account information and the manner in which the payment is settled (money transferred from the debtor to the creditor's account).



In order to utilise the eRepository, ISO20022 suggest that you:-

Download the ISO 20022:2013 ecore implementation metamodel. (which is a zip containing a file with a .ecore suffix). Then generate an Eclipse plugin

Download the e-Repository (which is a zip with a .iso20022 suffix)

3. Load the eRepository using the Eclipse plugin that you generated.

When I started examining ISO20022, It took me a day or two to install Eclipse and follow these steps to find that it provides you a plugin that appears only to allow you to browse the information and I was unable to work out how to extract the information from it, as there was no export capability.

What I did eventually discover was the eRepository in step 3, is actually an xml file. Once I found that, I wrote an xquery script in order to extract the xpath definitions for each message component/message element in each message schema definition. With this information, any xml parser can extract the data from an incoming xml file, and you can then process it and/or save it to a back end data store.

ISO20022 Message Definition Reports (MDRs)

This is where you're most likely to start when you look at ISO20022. The MDRs are contained within the full catalogue of messages and also within domain-specific catalogues e.g. the payments domain catalogue

There is an MDR for each business process (e.g. payments initiation, payments clearing & settlement, notifications and exceptions etc.)

The message definition reports are written for human understanding. They're split into 3 parts:-

Part 1 – is a word document which explains the business scenarios in which the business process is typically used and includes information about the actors (systems, people) involved in the business process and which messages would typically be passed between them.

Part 2 – is a PDF document which, for each message used by the business process, describes:-

a. The overall structure of the message

b. The definitions of any constraints identified within the message specification

c. The message building blocks – detailed information about the message components and message elements

1. Part 3 – provides information about:-

a. The business components & business elements used by the messages

b. Links the message components & message elements in the message to the business components & business elements

c. Provides a uml class diagram depicting the relationships between the business components used in the message (unfortunately missing the cardinality)

It's recommended that you read part 1 first to get a clear understanding of the business process, and then read part 2 to start to understand the information that will be contained in the message, referring back to part 3 to link this back to the business components & business elements that will need to be in your back end data store.

Potential drawbacks of ISO20022

It is impossible to create a standard that satisfies everybody. As the standard has been produced by financial institutions for the purpose of financial institutions, it can be seen as being a little heavy for other developers who don't have a need for all of the information required for processing financial messages, and have more lightweight methods of passing data.

For example, ISO20022 originally exclusively mandated messages to be defined using XML schema definitions (XSDs) which are relatively robust in enforcing the syntax of XML messages by examining their structure, cardinality, data types, whether elements are mandatory or optional and whether they have restricted sets of values (codesets) associated with them.

Nowadays, most web developers have got used to being able to pass data between back end web services and front end web pages using a simpler message format known as JSON. This is much less verbose (i.e. requires less words) than the equivalent XML message (but JSON also not as robust at validating the message as there is currently no standard JSON schema definition agreed).

A key advantage of JSON for web developers is that javascript (most common language for dynamically altering the contents of a web page) can easily parse the data provided by a JSON file and assign a message component to a javascript object and a message element to a javascript object property. The reduction in message size and not requiring an additional library to be installed to parse the message makes JSON more attractive to web developers than XML.