TECHNICAL SPECIFICATION

ISO/TS 20022-3

First edition
2004-12-15

Financial services — UNIversal Financial Industry message scheme —

Part 3: ISO 20022 modelling guidelines

Services financiers — Schéma universel de messages pour l'industrie financière —

Partie 3: Lignes directrices pour la modélisation ISO 20022

Reference number
ISO/TS 20022-3:2004(E)

(Blank page)

Contents

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

In other circumstances, particularly when there is an urgent market requirement for such documents, a technical committee may decide to publish other types of normative document:

— an ISO Publicly Available Specification (ISO/PAS) represents an agreement between technical experts in an ISO working group and is accepted for publication if it is approved by more than 50 % of the members of the parent committee casting a vote;

— an ISO Technical Specification (ISO/TS) represents an agreement between the members of a technical committee and is accepted for publication if it is approved by 2/3 of the members of the committee casting a vote.

An ISO/PAS or ISO/TS is reviewed after three years in order to decide whether it will be confirmed for a further three years, revised to become an International Standard, or withdrawn. If the ISO/PAS or ISO/TS is confirmed, it is reviewed again after a further three years, at which time it must either be transformed into an International Standard or be withdrawn.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO/TS 20022-3 was prepared by Technical Committee ISO/TC 68 to complement ISO 20022-1, Overall methodology and format specifications for inputs to and outputs from the ISO 20022 Repository, with detailed modelling guidelines to be used to construct ISO 20022 compliant business transactions and message sets. This Technical Specification should be reviewed and considered for publication as an International Standard once further experience has been gained in using these guidelines and the use of the underlying technology has further stabilized.

ISO 20022 consists of the following parts, under the general title *Financial services — UNIversal Financial Industry message scheme*:

— *Part 1: Overall methodology and format specifications for inputs to and outputs from the ISO 20022 Repository*

— *Part 2: Roles and responsibilities of the registration bodies*

— *Part 3: ISO 20022 modelling guidelines* [Technical Specification]

— *Part 4: ISO 20022 XML design rules* [Technical Specification]

— *Part 5: ISO 20022 reverse engineering* [Technical Specification]

Introduction

The methodology described in this document focus on the development of standardized Business Transactions and Message Sets. The objective of a standardized Business Transaction is to define a commonly agreed solution for communication problems existing among different organizations within the context of a given Business Process.

For a given communication problem in a given business context, several solutions can be developed. The purpose of this document is to explain the different steps a modeller should follow to ensure that ISO 20022 Business Components / Elements, Message Components / Elements, Business Transactions and Messages are defined in a consistent way.

The ISO 20022 methodology is composed of a set of activities. These activities are grouped into the following phases:

- business analysis
- requirements analysis
- logical analysis
- logical design (message design)
- technical design

For each of these activities, this document describes:

- the artefacts needed to start this activity (required input)
- the artefacts that should be the result of this activity (expected output)
- an example – where useful
- any modelling and other guidelines that should be followed or taken into account

It is not the intent of this document to describe what will be the allowed artefacts and/or documents to be submitted to the Registration Authority. This information is described in the ISO 20022 submission templates and their related guidelines.

Examples are provided only to illustrate the modelling methodology and should not be regarded as definitive for the Business Areas described.

For the purpose of ISO/TS 20022-3, the terms and definitions given in ISO 20022-1 apply.

1 Business analysis

A small example of Business Analysis can be found in Annex A (Business analysis: funds industry Business Processes).

1.1 Introduction

1.1.1 Purpose

The purpose of business analysis is to achieve a better understanding of the Business Area and Business Processes for which ISO 20022 compliant Business Transactions and Message Sets will be developed:

- Describing the Business Processes, including the Business Roles and their need for Business Information, helps in the identification of the communication problems that exist among the organizations that take part in these processes. Those communications problems are the main drivers for the next phase (requirements analysis).
- Identifying Business Information manipulated in a Business Area is also important because the Messages - that will be designed in later phases - will contain data elements that are related to this Business Information. An explicit link between Business Elements/Components and Message Elements/Components will be helpful for interoperability, for later maintenance and for change management: if something changes in a Business Area, it will be possible to identify the impact on previously defined Messages.

1.1.2 Key topics

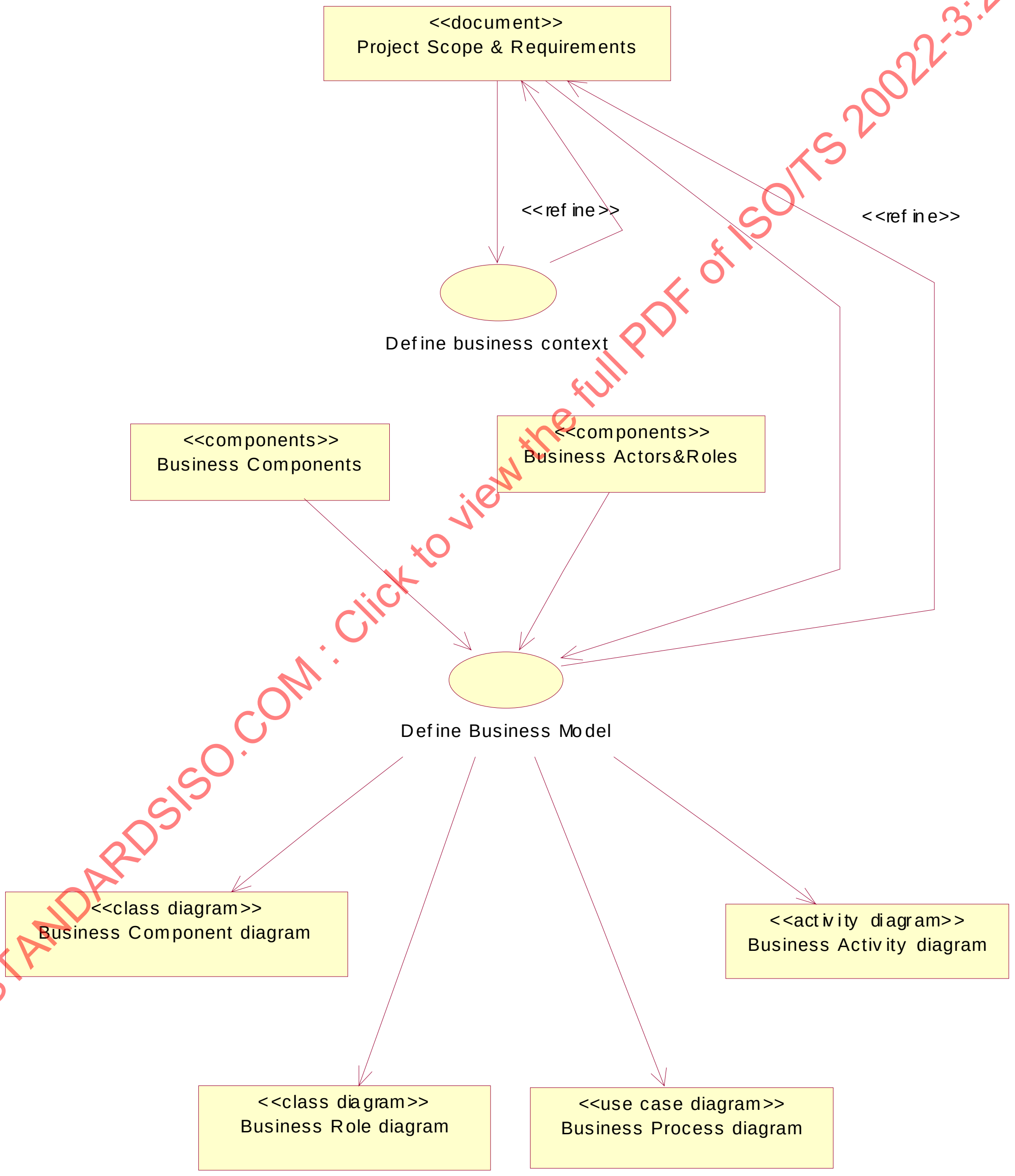
- To identify the business context of the communication problem to be solved.
- To understand the daily business in the Business Area and Business Processes with no special focus on the Business Transactions and Message Sets to be developed.
- To capture the Business Information manipulated within the Business Processes.
- To ensure that all users, such as business experts and standards developers have a common understanding of the Business Area and the Business Processes.

1.1.3 Deliverables

- Textual description of the Business Area (objectives, scope and boundaries)
- Models describing the Business Processes and the Business Information and Business Roles involved in these Business Processes. All model elements are enriched with textual descriptions, including a glossary of business terms.

1.2 Process overview

The below picture shows the different activities (shown as ovals) this process has to follow and what the required inputs and outputs (shown as squares) are. These activities are further detailed in the following paragraph.



1.3 Activities

1.3.1 Define business context

This activity consists of defining and refining:

- Business goal: global objectives and purposes for the considered Business Area.
- Scope: which Business Processes are in or out of scope, which financial instruments are in or out of scope, which Business Actors and Business Roles need to be considered, etc.
- Boundaries: key Business Components and Elements that are handled and/or used within the business context. They can be classified into input information (i.e. information that influences the Business Processes but that is not controlled within the scope of the business context) and output information (i.e. information that is controlled within the scope of the business context and impacts other Business Processes).
- Assumptions related to the Business Area.
- Business requirements (expected functionality) and constraints (e.g. market infrastructures to be part of the solution, performance requirements (T+1), interoperability requirements, Market Practices to be considered)
- Business terms will either be accompanied by a short and clear description to remove possible ambiguities or - preferably - must refer to an existing glossary of business terms.

1.3.2 Define Business Model

To define the Business Model, following steps are necessary:

1. **Business Process diagram**: produce a diagram of all Business Processes, starting from the list of Business Processes that have been identified in the project scope. The diagram should show the organization of the Business Processes, starting from a top-level node (representing the overall business goal). Refinements can be made using AND-decompositions, OR-decompositions, "include" associations and "extend" associations.
2. **Business Activity diagram**: complement the Business Process diagram with one or more activity diagrams for each Business Process. These activity diagrams will provide a better and more detailed understanding of the various activities and Business Roles in the Business Processes and on the interaction between the activities and Business Roles. The activity diagrams should ideally describe the normal flows and the exception flows within each Business Process.

   The type of activity diagrams that is recommended is called "Swim Lanes". Indeed, for each Business Role, there is a "swim lane" indicating the activities performed by that Business Role and, optionally, major states that can be reached by that Business Role.

3. Describe for each Business Process and each activity within a Business Process:
   - the definition (i.e. a description of the activity)
   - the trigger (i.e. an event that causes the start of the activity)
   - the pre-conditions (i.e. conditions that must be fulfilled in order to start the activity)
   - the post-conditions (i.e. conditions that must be fulfilled when the activity is completed)
   - the arguments (i.e. information that is required, created or changed for the execution of the activity)
   - the roles (i.e. functions of Business Actors when they are involved in the activity).

4. **Business Role diagram**: produce a diagram of all Business Roles that are relevant for the defined project scope (these roles have been defined during the previous steps). Go through all existing roles in the ISO 20022 Data Dictionary (DD) and copy the required ones in the Business Role diagram. Complete the diagram with those Business Roles that have been identified and that don't exist yet in the DD. If Business Roles are identified that exist "more or less" in the DD (i.e. there is a significant match but also some important differences) they should be created as new roles and an association should be added in the diagram to the related existing Business Role in the DD. All these additional Business Roles will have to be submitted to the Registration Authority.

5. **Business Component diagram**: Produce a diagram of all Business Components[1] derived from the "arguments" in step three. It can contain inheritance, association and aggregation relations. Go through all existing Business Components in the DD and copy the required ones in the Business Component diagram. Complete the diagram with those Business Components that have been identified and that don't exist yet in the DD. If Business Components are identified that exist "more or less" in the DD (i.e. there is a significant match but also some important differences) or new Business Elements need to be added, create them as new Business Components and add an association in the diagram to the related existing Business Component in the DD. All these additional Business Components will have to be submitted to the Registration Authority.

6. Check the consistency of the Business Model by verifying following points:

   - Are all Business Components and Business Roles that are used in the Business Process descriptions and in the activity diagrams included in the Business Role diagram and in the Business Component diagram?

   - Are all Business Components and Business Roles that have been defined in the diagrams referred to in at least one Business Process or activity (except possibly the ones that have been copied from the DD as basis for a project-specific "specialization")?

   - Is the Business Process diagram consistent with the information in the project scope?

[1] Business Components are defined as classes in the Business Model. A Business Component is the exhaustive definition of a business notion. Note that Business Components will never be used directly in message models because they are generic and don't take into account specific needs of the message context.

- Is the Business Component diagram consistent with the information in the project boundaries?
- Is the lifecycle (create - update - delete) of the used Business Components correctly covered[2] by the Business Process and activity descriptions?

1.4 Guidelines

- Apply a "bird's eye view". At the business analysis level, we want to concentrate on the business and avoid discussing the solution or even the communication problem. This means that we assume there is no communication problem and each of the Business Roles has a perfect knowledge of and access to all information manipulated in any of the Business Processes. Remember always that a "good" Business Process must add tangible business value.
- Focus mainly on Business Processes that provide a lot of added value, by eliciting Business Components or Business Roles. Don't get into details, like "cancel", "modify" or "create", etc. at this stage and don't bother about error handling either. For example, the description of an "Interbank Transfer" Business Process will elicit concepts like "Account", "Credit", etc. The description of the "Cancel Interbank Transfer" will have no specific added value and shouldn't be considered at this stage. These details will be elaborated during the logical analysis, when Business Transactions and Message Sets are defined.
- Concentrate on functional roles, rather than on real-life actors. It's for instance important at this stage to identify the role "Buyer", but it's not yet important to identify whether the buyer is an individual, a corporate or a financial institution.
- Depending on the useful level of detail, one may decide to decompose a Business Process into more detailed Business Processes.
- Normally, roles should only be associated to the most detailed Business Processes and to the activities (i.e. the lowest level).
- Make sure to provide all required information (e.g. a proposed name and a definition) for new Business Components and Business Roles.
- Be careful in the Business Component diagram to use aggregation ONLY to represent real business containment (e.g. a party is NOT contained in an account and there should therefore be no aggregation relation between account and party). Real business containment means that in real life the contained element can never exist without the container (e.g. an account balance cannot exist without an account).
- The Business Component diagram can contain information regarding multiplicity and must indicate the type (represented by a Data Type or a Business Component) of each Business Element.

[2] This doesn't mean that all Business Components must necessarily be created, updated and deleted in the described Business Processes, but that one is sure that it has been covered or that one knows why it doesn't have to be covered

2 Requirements analysis

A small example of requirements analysis can be found in Annex A (Requirements analysis: fund communication requirements).

2.1 Introduction

2.1.1 Purpose

The purpose of the requirements analysis is to define the communication requirements caused by the physical separation of the Business Actors that will execute the various Business Roles in the Business Processes.

The requirements analysis will identify and specify all communication requirements that exist within the agreed scope of Business Processes and activities. It will identify who needs what kind of information, from whom, at what moment. As such, the requirements analysis will provide the specifications for the solution (i.e. the Business Transactions and Message Sets) that will be developed, without going into the actual definition of messages and message flows.

2.1.2 Key topics

- Analysis of the results of the business analysis in order to discover the communication requirements that arise.
- Precise definition of the expected properties of the Business Transactions and Message Sets to be developed (functionality and interaction with the Business Roles).

2.1.3 Main activities

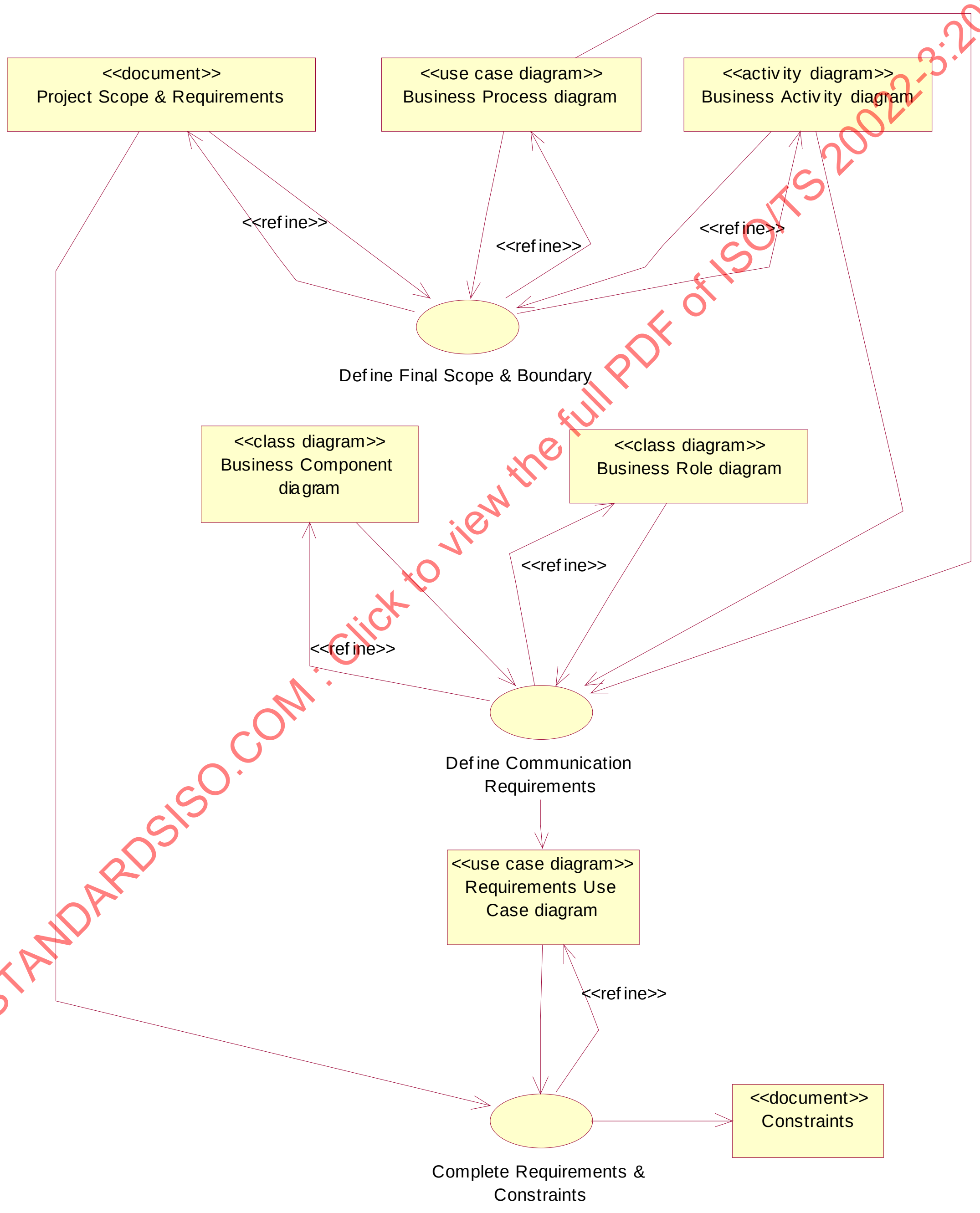
- Identification of the goals of the Business Transactions and Message Sets to be developed (exchange of information and possibly enhanced performance of specific Business Processes or activities).
- Specification of functional (= behavioural) requirements of the Business Transactions and Message Sets to be developed.
- Specification of constraints (= imposed restrictions) of the Business Transactions and Message Sets to be developed.

2.1.4 Deliverables

- Textual descriptions refining the scope and boundaries of the final solution
- Textual descriptions refining and completing the constraints.
- Requirements use cases describing the expected functionality of the Business Transactions and Message Sets to be developed. The description of the use case must include the definition, arguments, triggers, pre- and post-conditions.
- Business Component diagram describing the information used by each of the Business Roles (possibly complemented with textual descriptions of some business related Rules).

2.2 Process overview

The below picture shows the different activities (shown as ovals) this process has to follow and what the required inputs and outputs (shown as squares) are. These activities are further detailed in the following paragraph.



<<document>> Project Scope & Requirements
<<use case diagram>> Business Process diagram
<<activity diagram>> Business Activity diagram
<<refine>>
<<refine>>
<<refine>>
Define Final Scope & Boundary
<<class diagram>> Business Component diagram
<<class diagram>> Business Role diagram
<<refine>>
<<refine>>
Define Communication Requirements
<<use case diagram>> Requirements Use Case diagram
<<refine>>
Complete Requirements & Constraints
<<document>> Constraints

2.3 Activities

2.3.1 Define final scope & boundary

Define the Business Processes and activities that need to be supported by the Business Transactions and Message Sets to be developed. This will be done based on the Business Process and Business Activity diagrams that have been defined during business analysis and based on the requirements and scope that have been defined for this project.

2.3.2 Define communication requirements

At this stage, the “bird's eye view” that was taken during business analysis is abandoned. It is recognised that Business Roles involved in activities in Business Processes don’t have access to all information (i.e. they have only a limited knowledge about Business Components and/or a limited knowledge about the status of other activities in Business Processes). The Business Processes cannot be completed due to this lack of knowledge. In fact, this is the basic problem: we need to make sure that all activities in all Business Processes (that are in the scope) get access to the knowledge they need to get completed. The solution will therefore be to create one or more Business Transactions that take care of all the necessary communication between the Business Roles that are involved in the Business Processes. The Business Transactions will thus ensure that each activity in the Business Processes gets access to the information it needs. The goal of the requirements analysis is to identify and specify these communication requirements (i.e. what information needs to be provided to whom under which conditions, etc.).

Following steps need to be followed:

- Identify, in the Business Activity diagram, the activities that are in the scope and the Business Roles that take part in these activities.
- Define, for each activity to be executed, what information is required. This information consists of the input arguments, possibly information that triggers the activity (e.g. the fact that another activity is terminated) and possibly information necessary to check the pre-conditions.
- Define which information (from the above-defined set of required information) is not available to the Business Role executing this activity (i.e. which information is not owned by the Business Role).
- Create a "requirements use case" for each set of information that needs to be communicated as part of the Business Transactions to be developed and identify which Business Role is able to provide this information. In some cases, multiple Business Roles may provide (parts of) this information (possibly at different moments in the execution of the overall Business Process). Remark that the requirements use case may already exist (if it is needed by another activity).

Describe each requirements use case with following information[3]:

- **Definition**: the goal and functionality of the use case, i.e. which information is it providing to which activity and between which Business Roles.
- **Trigger**: the event that will start the execution of the use case.
- **Pre-conditions**: the conditions that must be fulfilled in order to be able to execute the use case.
- **Post-conditions**: the conditions that must be met when the use case has been executed.
- **Arguments**: the information that is used and produced by the use case.

2.3.3 Complete requirements and constraints

- Based on the analysis done in the previous steps and based on the project scope, write down systematically all constraints (e.g. constraints pertaining to a specific implementation).
- Verify whether there is no impact of these constraints on the functional requirements that have been described in the requirements use cases and adapt these uses cases if necessary.

2.4 Guidelines

- When specifying the requirements, it may be necessary/useful to combine a number of potential requirements use cases (e.g. because they all deal with the same Business Information or because they are always executed together by the same Business Roles) or to zoom in/out of potential requirements use cases (e.g. in order to have a manageable level of detail).

[3] Note that this information in the Requirements Use Case may be based on the corresponding information in the Business Processes or activities, but is not exactly the same.

3 Logical analysis

A small example of logical analysis can be found in Annex A (Logical analysis: Business Transactions).

3.1 Introduction

3.1.1 Purpose

The purpose of the logical analysis is to specify the details of the Business Transactions and Message Sets to be developed.

The focus is therefore on defining the Message Flows and Message Definitions that are needed to get the required information at the right time to the right Business Role. The Business Transactions and Message Sets are still looked at from a pure business perspective, meaning that the focus still remains on the semantics (i.e. the underlying business meaning) and not yet on the syntax (i.e. how to physically represent a Message and a set of Rules). All decisions are driven by the requirements (functional requirements and constraints) that have been identified and specified during the requirements analysis.

The logical analysis is said to be "logical" because it deals with the description of the Business Transactions and Message Sets from an abstract point of view versus a concrete, technical point of view. This abstract approach is needed to allow interoperability between different technical representations.

3.1.2 Key topics

- What is the overall architecture of the solution? What are the subsystems taking part in the exchange of Messages? Do we go for a user-to-user or for a centrally co-ordinated solution?
- What are the Business Transactions? The different Messages can be exchanged according to a number of admissible message flows that need to be identified.
- What are the Messages in terms of their business content? The exchanged information should have a business value. Message Components/Elements are defined from and traced to Business Components/Elements that were identified during the business analysis and requirements analysis.
- Which rules apply to the various Business Transactions and Messages and what is the scope of each rule? If the scope is the Message only, the rule refers only to the content of the Message. If the scope is the Business Transaction, the rule checks the Message

content against the overall Business Transaction information (e.g. the information contained in the previously exchanged Messages).

3.1.3 Main activities

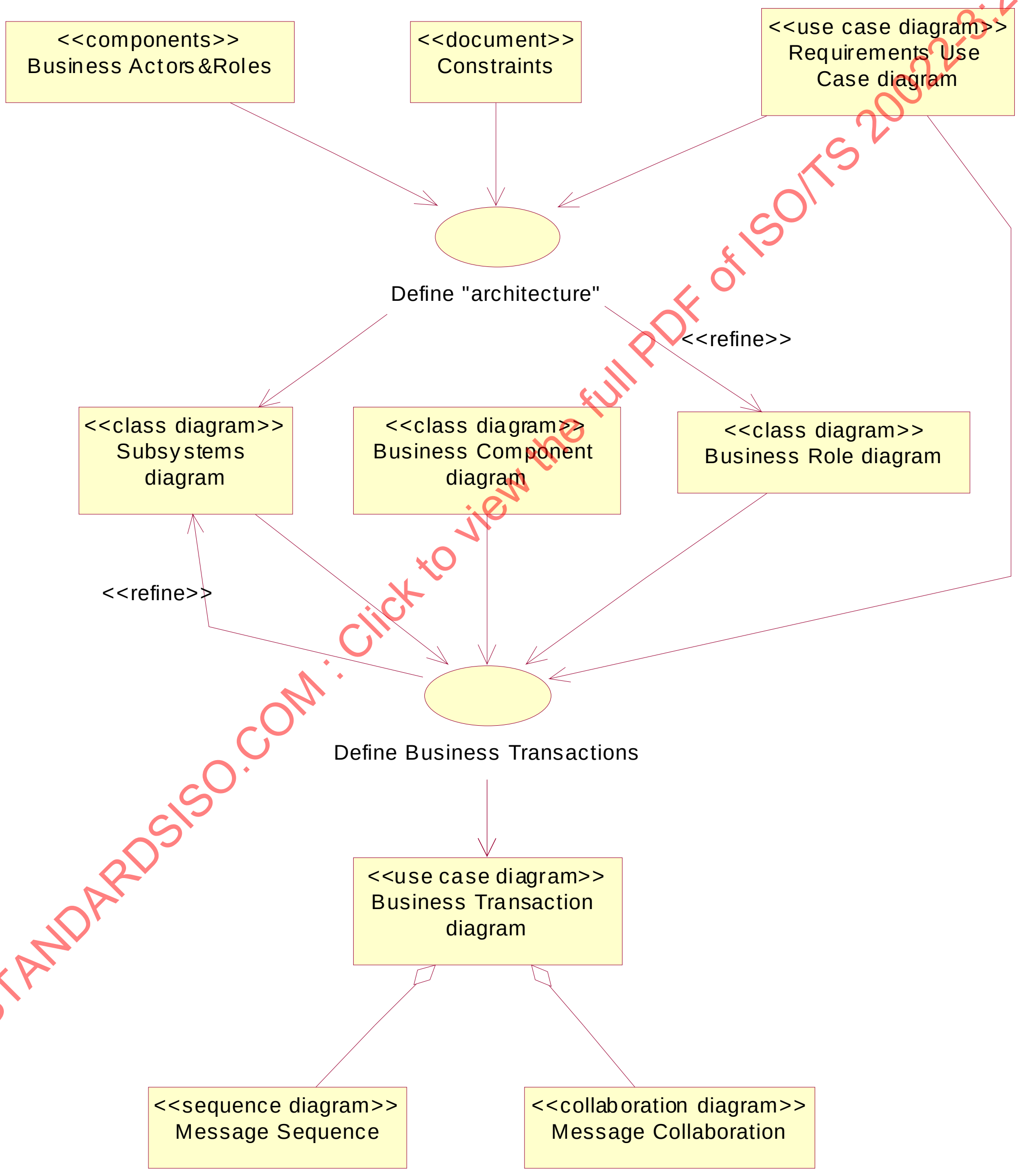
- Identify the overall architecture of the solution.
- Refine the requirements use cases into concrete Business Transactions. Based on the Business Transactions, identify Messages and major message flows and rules related to these message flows.
- Design the Messages i.e. identify the business content, overall structure and organization of the Messages and the rules that apply to the Messages.

3.1.4 Deliverables

- A Business Transaction diagram (containing a structured description of the Business Transactions.
- A textual description of the architecture of the system (subsystems) in case of a centrally co-ordinated system
- Sequence Diagrams (i.e. the typical exchanges of Messages in the context of a Business Transaction.

3.2 Process overview

The below picture shows the different activities (shown as ovals) this process has to follow and what the required inputs and outputs (shown as squares) are. These activities are further detailed in the following paragraph.


<<components>> Business Actors&Roles
<<document>> Constraints
<<use case diagram>> Requirements Use Case diagram
Define "architecture"
<<refine>>
<<class diagram>> Subsystems diagram
<<class diagram>> Business Component diagram
<<class diagram>> Business Role diagram
<<refine>>
Define Business Transactions
<<use case diagram>> Business Transaction diagram
<<sequence diagram>> Message Sequence
<<collaboration diagram>> Message Collaboration

3.3 Activities

3.3.1 Define “architecture”

- Defining the architecture of the solution means identifying the various "subsystems" that will be involved in the Business Transactions. Subsystems are the boundaries of the Business Transactions, meaning that the standards developer will not attempt to standardize the details of the applications that take place at these subsystems. The standards developer is only standardizing the communication (interface) with these applications. Defining the subsystems and the activities they will perform will help to define the required message flow.
- How to identify these subsystems? Although we're not looking for individual Business Actors, it's a fact that there is a relation with the **type** of Business Actors that will play the Business Roles in the Business Transaction. It's therefore good practice to combine at this point the definition of subsystems and the definition of types of Business Actors:
    - Decide whether central systems (e.g. TFM, market infrastructure) will be involved. If this is the case, there will be a subsystem associated to each central system.
    - Associate a subsystem to each type of Business Actors that will play one or multiple related Business Roles.

      In some cases, it will be possible to combine multiple Business Roles into one subsystem. This will have to be decided on a case-by-case basis, based on fundamental similarities or differences in the behaviour of the subsystems. It will also be influenced by the fact that the same Business Actor always plays these various Business Roles or not.

      In other cases, it may be necessary to foresee multiple subsystems for a single Business Role. This will be the case if multiple types of Business Actors can play the same Business Role and if there is a significant influence of the type of Business Actor on the behaviour of the subsystem and on its communication requirements.
- The subsystems can be documented in a class diagram.

3.3.2 Define Business Transactions

- Taking into account the subsystems that have just been defined, describe the way(s) in which the requirements use cases will be realized by the subsystems (i.e. how the subsystems will interact) and what information flows are required for this.

- The Business Transactions will at least be described by text and by a sequence diagram, showing the information flow between the subsystems. Optionally, one can add a collaboration diagram (note that the latter can be derived automatically from the sequence diagram). The description(s) of the Business Transactions will provide information about the information flows (= messages) that need to happen between the subsystems, the sequence in which these flows will take place, exception handling, trigger, pre- and post-conditions related to each information flow.
- For each identified information flow, define the name of the Message, the required Message content and the high-level Message structure (i.e. what information should logically be put together). This information must be documented in the sequence diagram (linked to each information flow). The "Message content" can be identified by taking into account the information that is required and provided by the different Business Activities. At this stage, one can also specify information regarding multiplicity, choices and textual descriptions of rules. Where necessary, one should also identify whether information flows are synchronous or asynchronous, push or pull, etc.
- Refine the subsystems class diagram, if necessary.

3.4 Guidelines

3.4.1 General

- The decision of the basic architecture (i.e. user-to-user or centrally co-ordinated) will in most cases be dictated by the initial requirements.
- Typical examples of types of Business Actors in the financial industry include banks, agents (like brokers, investment managers, etc), corporates, etc.
- Like the Business Roles, the Business Actors will be part of the DD. Within the project they will be added in the Business Role diagram (but with a clear indication that they are Business Actors). The diagram will also show the links between Business Actors and the Business Roles that are played.
- In general each requirements use case can (and will) be realized according to multiple scenarios: for each scenario a separate Business Transaction must be defined and described.
- The Business Transactions must also take into account (and thus describe) the exception handling, the error handling, the acknowledgements, etc. These may result into additional Sequence Diagrams.
- Note that a Business Transaction describes only one possible solution for a single set of requirements. There can be multiple solutions for the same problem. In that case there would be multiple Business Transactions for the same requirements use case.
- When identifying Business Transactions, one might identify that some Business Transactions are specializations, extensions or combinations of others. For that purpose, the Include, Extend, and Generalization relationships can be used.

3.4.2 Message granularity

Initial decisions concerning Message granularity must be taken (following rules will consequently have an impact on 4.4.1.1. Granularity message modelling guidelines):

- Messages need to be precise and fit-to-specific purposes. Having very specific Messages will lead to a higher number of Messages and will therefore require some help for selecting the right Message to use in any particular case. This can be achieved by following the same top-down approach that has been used in the methodology, leading sequentially to the identification of the relevant Business Area, Business Process, Business Transaction and Message.

  The proposed approach leads to Messages that support limited but accurate business functionality. These Messages will have less optional fields. It also results in shorter messages, since a number of elements that are in a message to further define the message functionality, can be removed. All of this simplifies implementation, maintenance and will finally result in higher STP rate.

  Example:

  We don't need the same information when buying a security or selling a security. The returned confirmations will also be different. This should therefore result in different Messages.

- The minimal guideline is to have separate Messages for different functionality (e.g. separate Messages for create, modify, cancel; separate Messages for instructions and reporting).

- With regard to different types of financial instrument, the guideline is to cover them by the same Message unless the distinction is not only in the description of the financial instrument (e.g. other parties must be identified in the Message if another financial instrument is used).

- With regard to "Market Practices", the guideline is to concentrate first on a Market Practice independent Message (i.e. a Message containing all the commonalties). Next we can focus on a specific Market Practice and identify the specific information and the need for more specific multiplicity, data typing and rules. Depending on the required level of differentiation, Market Practices may therefore result in different "variants" of the same Message Definition and/or in Message Definitions covering the needs of multiple Market Practices.

- With regard to multiple subsystems that play the same Business Role, the guideline is similar to the guideline concerning the Market Practices. Concentrate first on the commonalties across the subsystems and identify afterwards the differences in information needs, multiplicity, data typing and rules. Depending on the required level of differentiation, this may result in different "variants" of the same Message Definition and/or in Message Definitions covering the needs of multiple subsystems.

- A useful check to verify whether it makes sense to create multiple (variants of) Message Definitions versus a single one is to consider the impact of removing a particular component or element from a Message Definition in order to create multiple explicit Message Definitions. If the removal implies the creation of a huge number of Messages (e.g. removing the currency from a payment Message to create separate Messages for payment in EUR, in GBP, in USD, etc.), it's not a good idea. On the other hand, if the removal of the component or element leads to a removal or simplification of rules within the Message, it is probably a good idea (e.g. removal of a buy/sell indicator means that other rules can be dropped).
  In those cases when the element that is considered to be removed contains a large number of possible values, it is not advisable to remove it since each of its values might result in a new (variant of a) Message Definition.

4 Message design

A small example of message design can be found in Annex A (Message design: compose Messages).

4.1 Introduction

4.1.1 Purpose

The purpose of message design is to refine the logical model of the solution in order to make it formal (i.e. precise and unambiguous) and identify reusable Message Components.

4.1.2 Key topics

- Which existing Message Components will be used?
- Which new Message Components must be created?

4.1.3 Main activities

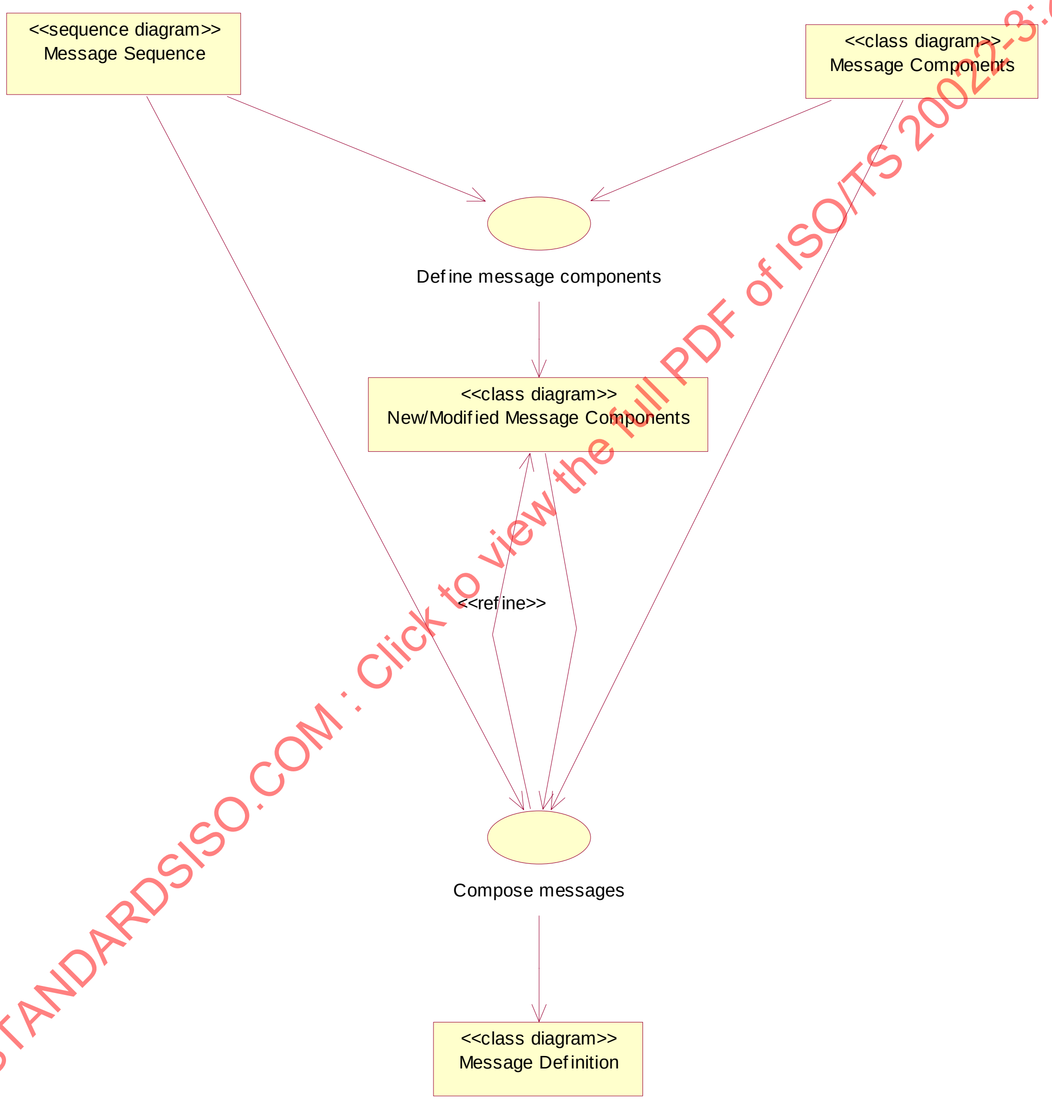
- Formalise the Message contents.
- Identify suitable Message Components
- Consolidate the above into a Message Definition Diagram.

4.1.4 Deliverables

- Message Definition Diagrams
- Textual Rules that are written in a formal language and that complete the formalisation of the logical model.

4.2 Process overview

The below picture shows the different activities (shown as ovals) this process has to follow and what the required inputs and outputs (shown as squares) are. These activities are further detailed in the following paragraph.



4.3 Activity: define Message Components

It is important to understand what the various components in a Message can be. The below **Message metamodel** shows what the allowed components, relationships between components, Data Types, etc… in a Message can be. Any Message must be constructed according to this metamodel.

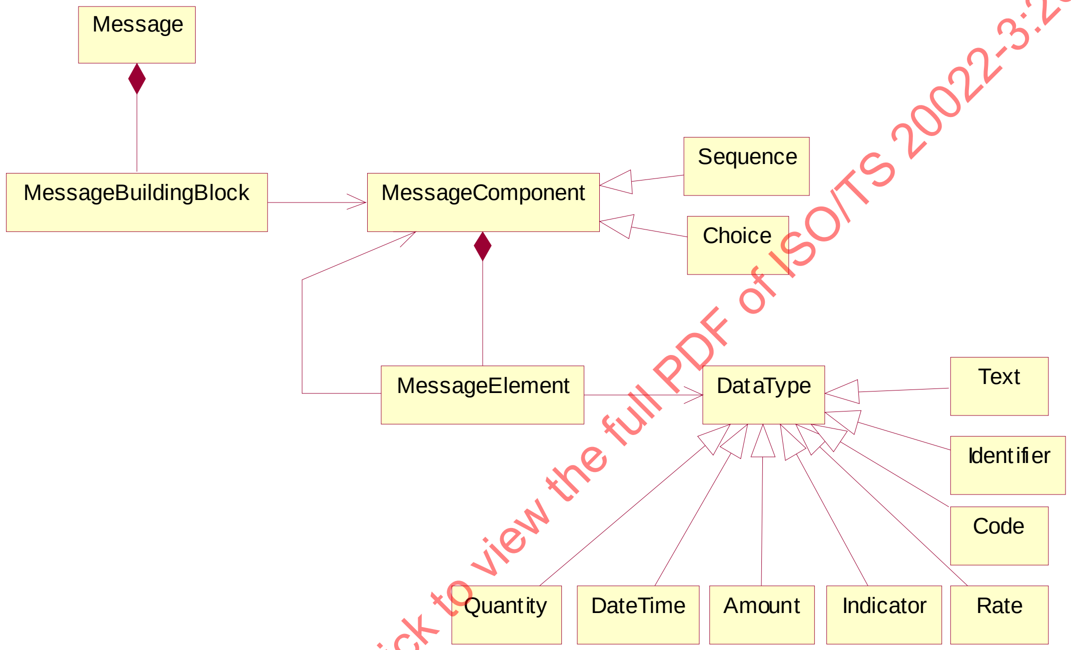


Figure 1: simplified Message metamodel

How to read this metamodel:

A **Message** is structured in **Message Building Blocks**, each representing information that logically belongs together. The structure and content of each Message Building Block is defined by a **Message Component**.

A **Message Component** is composed of **Message Elements.** A Message Element either references another Message Component (modelled either using an aggregation or as the type of an attribute), or has a **Data Type** (always modelled as the type of an attribute). A Data Type is a **Quantity**, an **Identifier**, a **Code**, an **Amount**, etc…

Classes representing Message Components have the <<MessageComponent>> stereotype (indicating a **Sequence** of Message Elements) or the <<ChoiceComponent>> stereotype (indicating a **Choice** between Message Elements).

4.3.1 Common guidelines

4.3.1.1 Message Components are derived from Business Components

Most Message Components are derived from Business Components and most Message Elements are derived from Business Elements[4]. A traceability link will be defined in the Data Dictionary between a Message Component /Element and its related Business Component / Element. There can be several Message Components / Elements defined and traced to one Business Component / Element.

Some Message Components or Message Elements may be present in the Message Definition for “message specific” reasons (e.g. page number, certain references, etc.). These components or elements are said to be “technical” and they are not derived from any Business Component or Business Element.

4.3.1.2 How to select / create the right Message Components for a Message

1. The high level structure that was defined during the logical analysis will give a first indication of what type of information should be kept together.
2. The first (and simplest) case is when a number of Business Elements are needed from one single Business Component. In that case, the DD can be searched for all Message Components that are based on this Business Component. If there is a Message Component that contains the exact required elements with the correct multiplicity and rules, this is the obvious choice.
3. What if there is no Message Component that exactly fits the needs?
    - A Message Component can be reused even if it contains too many elements (if these extra elements are acceptable)
    - A Message Component can be reused if it is less restrictive on multiplicity and/or rules (if a lesser validation is acceptable)
    - Two or more Message Components can be used to get all the required elements. One can either
        - ➢ put the Message Components next to each other in the Message
        - ➢ propose to the RA to create a new Message Component out of the combination.

[4] Note that Business Components will never be used directly in message models because they are generic and don’t take into account specific needs of the message context.

4. If a number of Business Elements are needed from multiple Business Components and there is no need to express relations between the different Business Components, use multiple Message Components by following the approach described above.

5. If a number of Business Elements are needed from multiple Business Components and there is a need to express relations between the different Business Components (e.g. a link is needed between an Account with Account Owner and Account Servicer), the DD should be searched for all Message Components that are based on one of the identified Business Components.
   Concentrate on those Message Components that express already a relation between two (or more) of the Business Components. Try to find a Message Component that contains all the elements that are needed and that expresses the required relationship(s). If this Message Component doesn't exist, try again to find either a Message Component that contains more elements than needed or to combine multiple Message Components. If this isn't an acceptable solution, the creation of one or more new Message Components should be proposed.

4.3.1.3 How to define a new Message Component

- If a Message Component only needs elements from one single Business Component, propose a Message Component that is based on this Business Component and that contains exactly the Message Elements needed, with the required multiplicity and rules (see also 4.3.1.4).

- On top of the above, a Message Component may need additional elements that do not exist in any Business Component and that only make sense in a specific Message, i.e. “technical” Message Elements (see 4.3.1.4). In this case, propose a Message Component as described in the previous point and add the required technical Message Elements – for which no equivalent Business Elements exist – with the required multiplicity and rules.

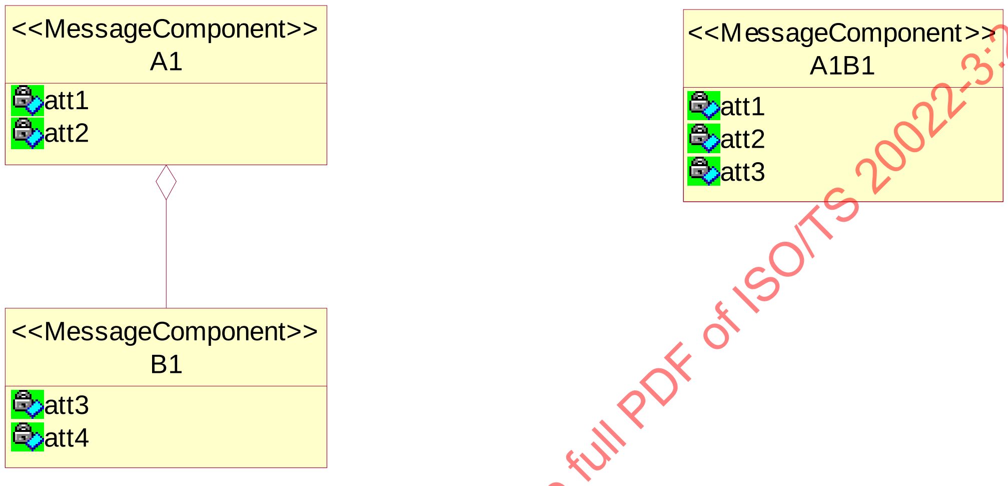
- If a Message Component needs elements from multiple Business Components (because a relationship between the Business Components must be expressed), following options are available:

  - Use several "simple" Message Components (i.e. Message Component based on a single Business Component) and – if necessary – add rules at the Message level to express the relationship.

  - Aggregate "simple" Message Components into a “parent” Message Component, whereby all simple Message Components are “siblings” (e.g. the new component has three Message Components A, B and C). In this case, the modeller will express the fact that the different Message Components belong together and he/she can express the necessary multiplicity information and/or rules. Submit the new Message Component to the RA.

  - Create a new Message Component that expresses the dependency (e.g. the new Message Component A1 is based on Business Component A and contains an aggregation with a Message Component B1 based on Business Component B). In this case, it is possible either to keep an explicit aggregation or to "import" the

Message Elements from the dependent Message Component into the parent Message Component (i.e. using attributes instead of aggregation). This last option should be used carefully as it can express less dependency (e.g. it cannot express that either all the Message Elements coming from Message Component B1 or none should be present). A guideline is to use this option mainly when there's only one Message Element coming from Message Component B1 involved. The picture below shows the two described options (“aggregation” at the left and “import” at the right).



4.3.1.4 What are Message Elements

A Message Element can be:

4.3.1.4.1 A Message Element based on a Business Element

Such a Message Element is a sort of “copy” of a Business Element from a particular Business Component and has the following characteristics:

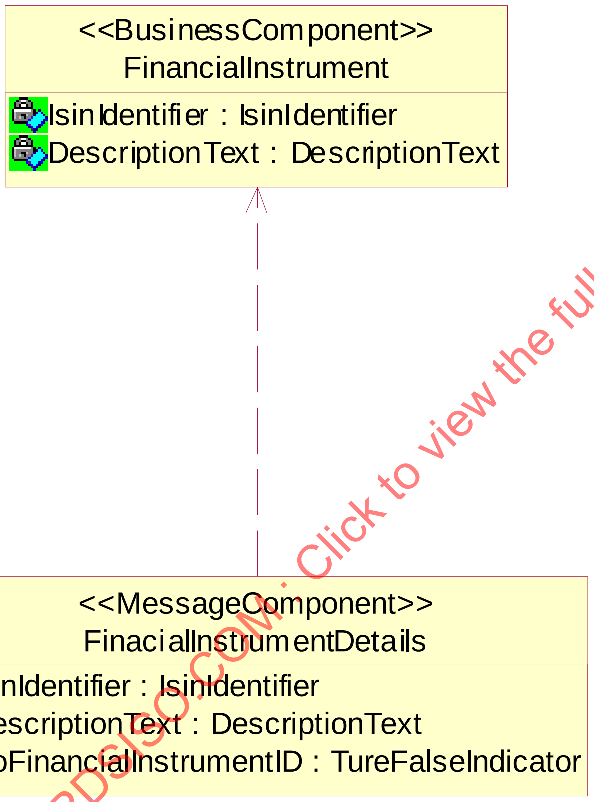
- If the Business Element is typed by a Data Type, the Message Element must be typed by a Data Type as well. This Data Type must either be the same or a more restrictive Data Type of the same Data Type Representation. This last option must be used if the set of allowed values in the Message Element needs to be restricted (e.g. a subset of the list of Codes that is defined for the Business Element).
- If the Business Element is typed by a Business Component, the Message Element must be typed by a Message Component. This Message Component must be based on the Business Component that types the Business Element.
- If the semantic of the Message Element is exactly the same as the semantic of the Business Element, the definition and name of the Business Element must be reused as such.
- If the semantic of the Message Element is richer or more specific than the semantic of the Business Element, the definition and optionally the name of the Business Element must be adapted for the Message Element.

Example: it might be required to reference a specific instance of a Business Element (the date of the "last" entry instead of the entry date) or to reference a calculated data. In this case, the Message Element definition and name must express the specific semantic (e.g. LastEntryDate)

4.3.1.4.2 A technical Message Element

A technical Message Element is a Message Element that only makes sense in a message context. There is no equivalence with, and thus no traceability link to a Business Element.

The following example describes these principles.



<<BusinessComponent>>
FinancialInstrument
IsinIdentifier : IsinIdentifier
DescriptionText : DescriptionText
<<MessageComponent>>
FinacialInstrumentDetails
IsinIdentifier : IsinIdentifier
DescriptionText : DescriptionText
NoFinancialInstrumentID : TureFalseIndicator

Figure 4-1: Deriving a Message Component from a Business Component

1. A Message Component (FinancialInstrumentDetails) is created and a traceability link is added with the corresponding BusinessComponent (FinancialInstrument).
2. Attributes "IsinIdentifier" and "DescriptionText" are copied. Neither the name nor the type is modified.
3. "NoFinancialInstrumentID" was created only for FinancialInstrumentDetails. It has no traceability link. Its purpose is to indicate whether a FinancialInstrumentIdentification Message Component is present in the message, or not. This information can also be derived from the presence/absence of that Message Component in the message.

4.3.2 Advanced guidelines

4.3.2.1 How to aggregate two Message Components

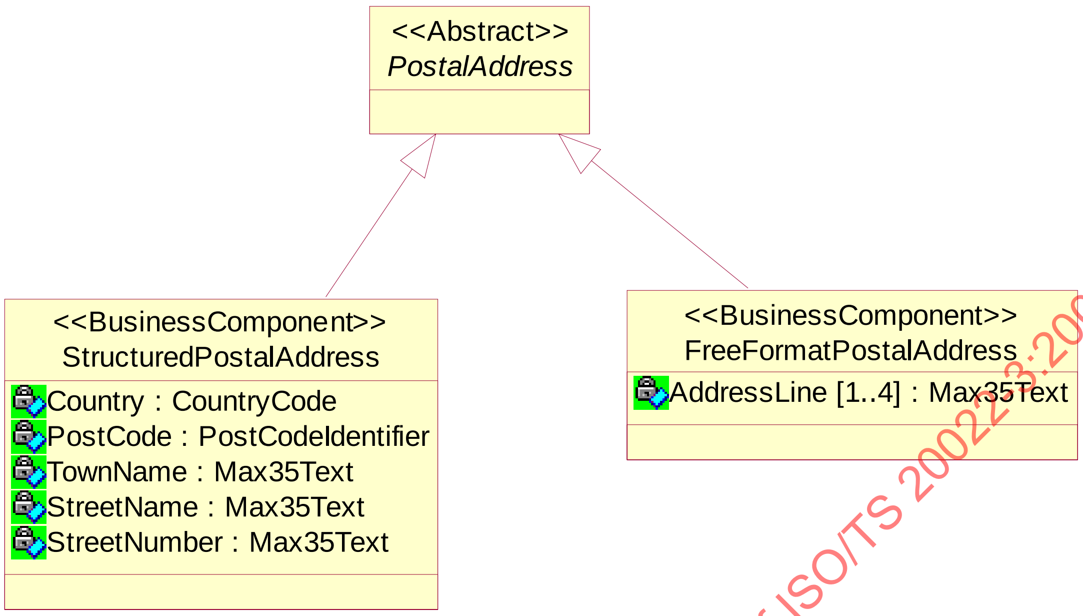
When two Message Components are related, the relation can be expressed either explicitly with an aggregation between the two components, or indirectly by using an attribute.

- When the relationship is modelled as an aggregation link, the link is explicitly shown between the two components. The aggregation “role name and definition” allow the modeller to give additional contextual information. This representation is more visual, but may quickly ‘flood’ the diagram with links.
- When using an attribute, the type of the attribute in one Component points to the other Message Component. In that case, the name and the definition of the attribute allow the modeller to give additional contextual information. The name of the attribute is equivalent to the name given to the aggregation (role name) in the previous case. This approach is less visual (the link is implied), but the diagram seems less ‘flooded’ with aggregations in case of a complex Message. This modelling way is also known in UML as using attributes as ‘foreign keys’.

4.3.2.2 How to handle abstract classes

Message Components should never be developed for an “abstract” Business Component. A Business Component is defined as abstract when it cannot be instantiated. A Message Component is, to some extent, an implementation of a Business Component. Hence it doesn’t make sense to have an implementation for abstract Business Components.

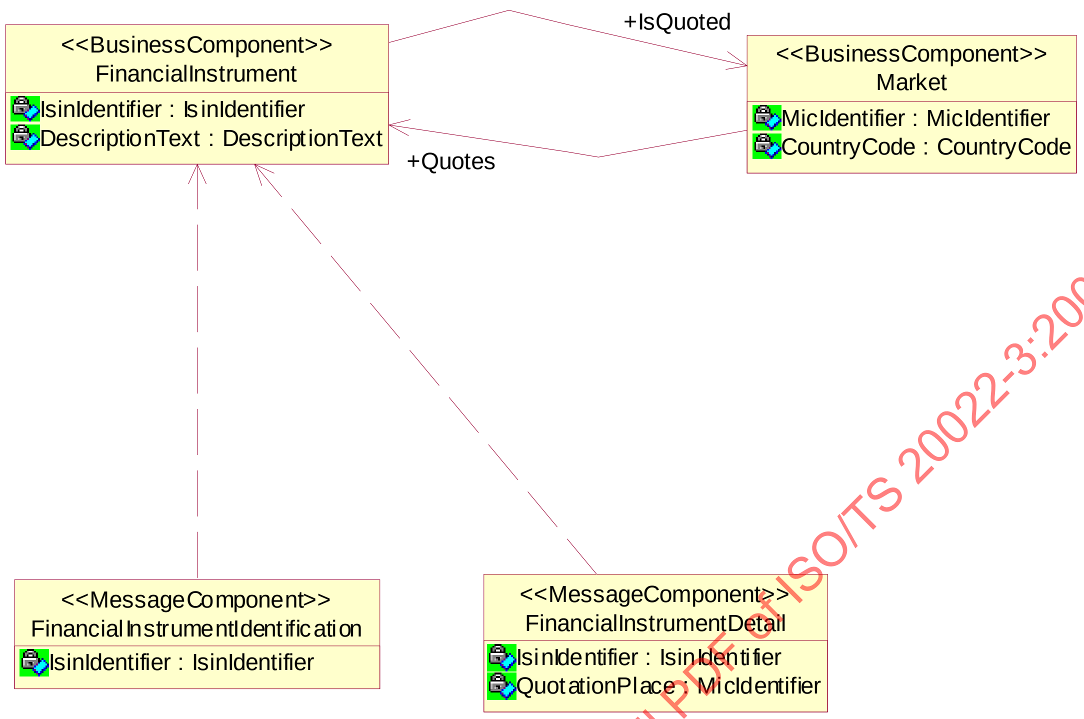
Consequently, since an abstract Business Component can never have a traceability link, the traceability will be on the Business Components specializing the abstract Business Component.



4.3.2.3 How to handle tables (bi-directional relations)

If a relationship between two Business Components is “bi-directional”, multiple Message Components will be introduced representing each direction of this association. This is a consequence of having hierarchical Message Definitions.

For example, a ISO 20022 compliant relation is represented by nesting the related Message Components. The nesting of two Message Components means that one component definition contains the other component definition.



<<BusinessComponent>>
FinancialInstrument
IsinIdentifier : IsinIdentifier
DescriptionText : DescriptionText
+IsQuoted
<<BusinessComponent>>
Market
MicIdentifier : MicIdentifier
CountryCode : CountryCode
+Quotes
<<MessageComponent>>
FinancialInstrumentIdentification
IsinIdentifier : IsinIdentifier
<<MessageComponent>>
FinancialInstrumentDetail
IsinIdentifier : IsinIdentifier
QuotationPlace : MicIdentifier

So if FinancialInstrument contains Market and Market contains FinancialInstrument, it would mean that message instances could contain an endless loop. In such a MessageComponent, we need a representation of only one direction of the relation “IsQuoted”. The solution is to define a MessageComponent called for example “FinancialInstrumentDetail” that would contain only the required information: the FinancialInstrument Identification and the identification of its quotation place.

4.3.2.4 How to handle recursion (relationship loops)

Assume for example three Business Components A, B and C, whereby A is linked to B, B is linked to C and C is linked to A; then multiple Message Components will need to be introduced to avoid introducing recursion. Again, the reason is that the Message Definitions essentially support hierarchical relations (tree representation) while modelling allows defining networks of relations.

4.3.2.5 Inheritance

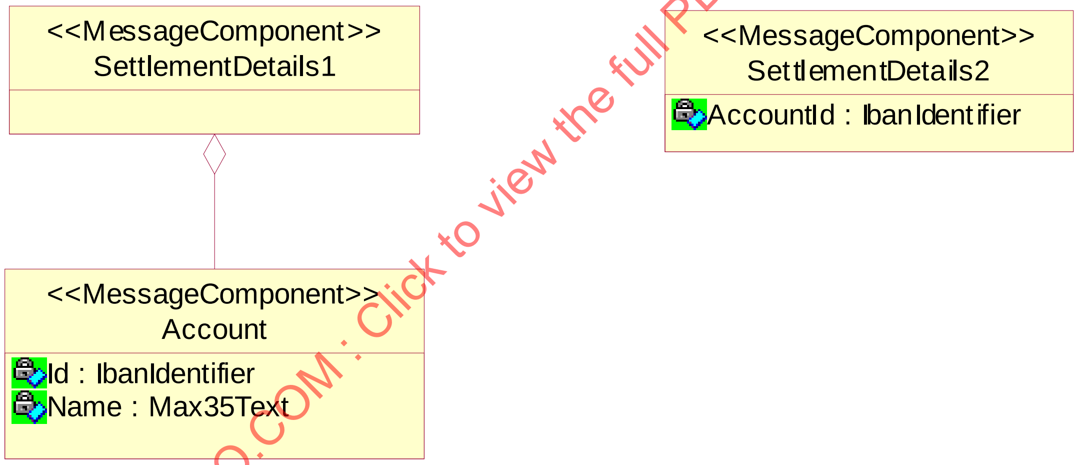
Inheritance in Business Components should not be duplicated in Message Components. Inheritance in business modelling allows modellers to classify Business Concepts. Message Components are only limited “views” developed to cope with implementation requirements.

4.3.2.6 How to optimize Message Components

Suppose a Business Component A is linked to a Business Component B, there are basically two ways to define Message Components:

1. Define a Message Component A and a Message Component B and represent the relation between A and B by defining an aggregation relation from A to B.
2. Add in the Message Component A the required Message Elements of B. This process is known as denormalization.

Be very careful about the **naming** of these 'moved' Message Elements.
Since Message Elements are 'context sensitive', it is strongly recommended to include the context from which the Message Element was 'removed' in the **new name** that is given to the Message Element.
Below example shows how the Message Element "Id" that belongs to the Message Component "Account" (that is linked to Message Component "SettlementDetails1") must be renamed to "AccountId" when it is 'moved out of its context into Message Component "SettlementDetails2".



4.4 Activity : compose Messages

For each identified Message in the sequence diagram, combine the selected (or new) Message Components to create the final structure of the corresponding Message in the Message Definition Diagram

An ISO 20022 compliant Message Definition is basically a tree data structure. Each branch of the tree is defined by its components. The principles defined for designing Message Components also apply to designing Messages.

To make sure all Messages are constructed in the same way, they have to follow the restrictions and rules imposed by the **message metamodel**. (see activity "4.3. define Message Components").

4.4.1 Common guidelines

When assembling Message Components to build Messages, the basic rule is that components should be linked in a "non-intrusive" way. If a component A and a component B need to be linked (because, for example, A "owns" B), the relation "owns" needs to be implemented without impacting the definition of A or B unless they always have to be used jointly.

A Message is modelled as a class with a stereotype <<Message>>.

A Message is composed of Message Components.

A Message Component is a class with stereotype <<MessageComponent>>. Message Components cannot be modified when assembling them into Messages. It means that no attributes and no aggregation links can be added to a <<MessageComponent>> class.

Where necessary, add information about multiplicity, choice, facets (e.g. the allowed structure of an element), operations (e.g. check that a currency code exists) and Rules (e.g. the Settlement Date must be later than the Order Date). (See ISO/TS 20022-4 Syntax Design Rules document).

4.4.1.1 Message granularity

Many of the decisions regarding Message granularity have already been taken when the Business Transactions (see Chapter 3.4.2 Message granularity) were developed. The same main guidelines should still be applied when composing the Messages.

4.4.1.1.1 Segregate the processing information from the data that is supporting these processes.

Messages should not contain Business Process indicators (Buy, Sell, Create, Delete, etc). Their identification will be done by including them in headers or in the Message name, or be part of the scope of the Message. Messages should only contain the data required to support the Business Processes, no identification of the Business Processes or refinements of Business Processes.

4.4.1.1.2 "Factorize"

Example:

Instead of having a currency code inside a repeating sequence and a rule saying "All currency codes must be identical", one can put the currency code only once out of the repeating sequence.

4.4.1.1.3 Avoid dependencies between code value and components.

Example:

Instead of having a code "PhysicalDelivery = yes or no", an address component and a rule saying "If the PhysicalDelivery = Yes THEN the address must be present", it is more efficient to create a component "PhysicalDeliveryAddress".

4.4.1.1.4 Be very careful when nesting components

Example:

By defining a list of Intermediaries in an InvestmentAccount component, we make it difficult to reuse this component in a context where the Intermediaries are not needed. The recommendation is then to keep the InvestmentAccount and the Intermediaries as two separate components and to create a third component representing the relation between the two.

5 Technical design

The technical design consists of a set of mapping specifications from the logical design model to a target technical implementation (such as an ISO 20022 XML Schema). During the technical design activity, the logical descriptions (of Messages, Collaborations, etc) are mapped to the target technology (e.g. ISO 20022 XML).

The technical design is based on a set of mapping specifications used to automatically generate a predictable representation of the Messages and Business Transactions.

This activity is said to be technical because it deals with the technical representation of the Message within the targeted technical environment (like for instance XML Schema for the Message syntax and programming code for some validation rules).

The mapping specifications can be found in “ISO/TS 20022-4 Syntax Design Rules”.

6 Naming conventions

The details of the naming conventions are described in the ISO 20022 submission templates and their related guidelines. Below are some generic rules that are applicable to most items in the ISO 20022 Repository.

1. Use the British English vocabulary.
2. Observe the (character) restrictions that are typical for most syntaxes when naming elements:
   - All names must start with an alphabetic character.
   - All characters following the first characters must be alphabetic characters or numeric characters.
3. Apply camel case convention:
   - A name for elements and attributes may be made up of multiple words, each consisting of alphanumeric characters.
   - Each word starts with a capital letter.
   - All white spaces between words are removed.

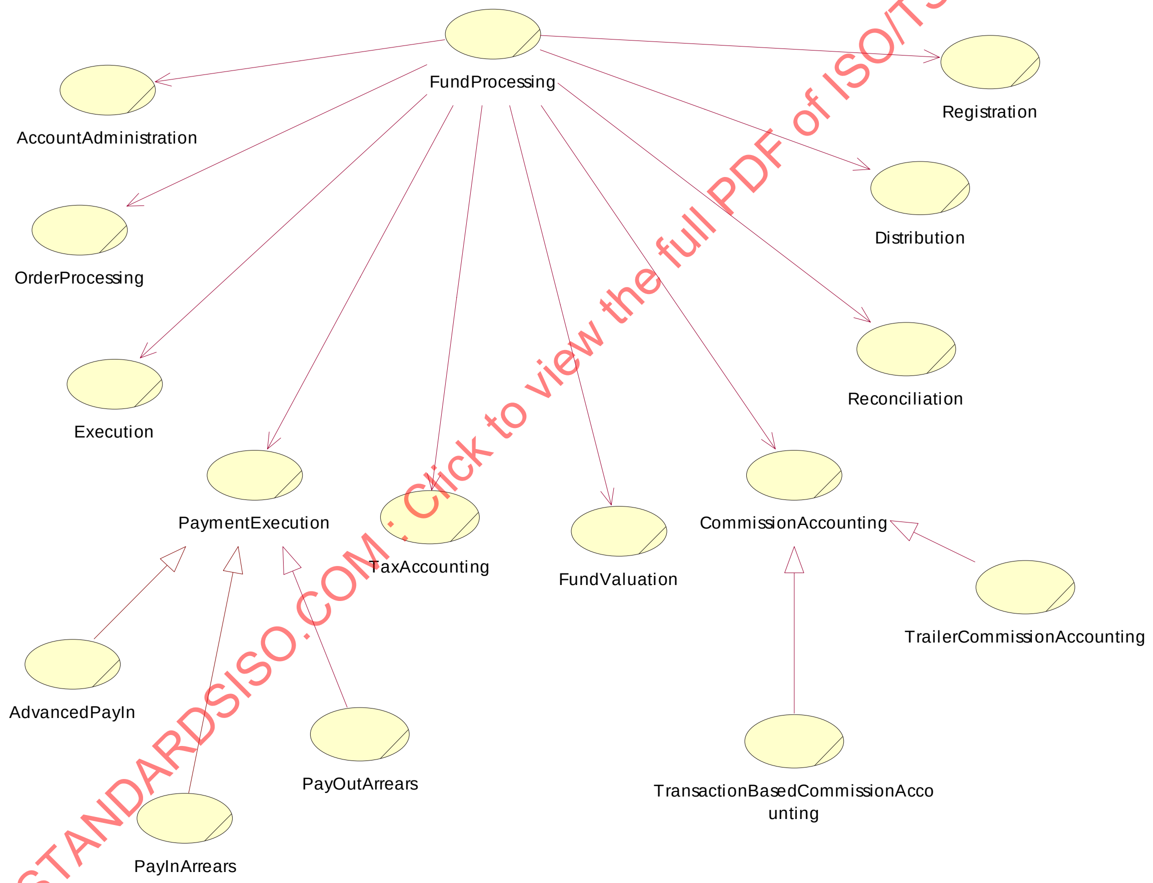
Annex A: Example

A.1 Business analysis: funds industry Business Processes

A.1.1 Define Business Model: OrderProcessing.

Business Process Diagram

This diagram shows all the Business Processes being considered in the Funds Industry:



Definition:

An Investor places an instruction to subscribe, redeem, transfer or switch between units in one or more funds.

Arguments:

Type of operation

Amount and currency or quantity of unit

Fund ID, type of share (share class)

Account ID and registration information

Trade date and time / Settlement date

Settlement method

Transfer Agent ID, Intermediary ID, Custodian ID

Compliance certification, Dividend preference

Commission information: type, amount and recipient

Requested discount (if different from the standard contractual discount)

Request for physical delivery

Tax information

Dealing currencies

Trigger:

Investor/intermediary decision.

A corporate action event.

A standing instruction (saving plans, saving schemes, etc)

Pre-condition:

Valid account exist

Fund exists.

Credit limit exists (for subscription), if required.

For switch, transfer or redemption, the units in the account must be settled/created.

The order complies with the terms set in the prospectus.

Intermediary and Investor are valid and verified.

Post-condition:

A valid instruction is generated.