ANNEX 2 Traceability

1 Introduction

1.1 What is traceability?

ISO 8402 definition:

> *"The ability to retrieve*
> - *history*
> - *usage*
> - *localization of item(s) or similar activity(ies)*
>
> *by means of a registered identification."*

TMWG definition:

> *"Traceability is a mechanism to link all the deliverables in the various stages of the methodology. It enables a modeller to clearly identify at what stage in the process he has reached and where a particular deliverable was created."*

1.2 Why is traceability needed?

Traceability is necessary to enable modellers and developers to demonstrate to their clients which of their business requirements have been met.

Traceability is one the major quality criteria of any modelling/software development.

Traceability is needed by the modellers:
- to help them to find their way in the modelling process,
- to facilitates queries in a modeling approach.

In the context of message development, traceability is useful to facilitate future modifications of the messages.

Within the UN/CEFACT Unified Modeling Methodology, the requirement list is the formalisation of what the clients want and each of the requirements has a unique reference number and status which enables developers to indicate if the requirement has been met or not.

1.3 Traceability Items

Definition of Traceability Item

> *"Any artifact, whether it is a textual or model item, which needs to be explicitly traced from another artifact in order to keep track of the dependencies between them."*

In addition to items defined as deliverables of Business Process Modelling, it is necessary to capture and track the attributes of, and traceability between, many other kinds of item. These other traceability items include issues, assumptions, requests etc.

Capturing and tracking these other kinds of traceability item will help the effective management of the Business Process Modelling project.

1.4 Implicit and Explicit Traceability

There is a certain amount of traceability implicit in any modelling process. This is supplied by the formal relationships between the artifacts in the process. It provides a level of traceability which allows impact analysis to be undertaken using the information held in the models.

Implicit relationships are fundamental to the modelling process and benefit from being built as a natural part of the modeller's work. These relationships are central to the modelling process and are constructed, and maintained, as the models are matured.

Implicit traceability is limited to the relationships available in the modelling method.

Some materials used in the modelling process are related items which may be directly related to one of the artifacts, to more than one artifact, or to a workflow, in the Business Process Model that need to consider them. It is therefore necessary to have an explicit traceability to link all these items into a single hierarchy, such as is proposed below.

1.5 Possible Cost Impact

A major decision in setting up a traceability process is the level of traceability that is required and how much explicit traceability is necessary to meet this goal. The level chosen shall facilitate the modelling process, not complicate and restrict it.

The addition of explicit traceability to Business Process Modelling could have a significant cost impact, especially when the long-term cost of populating and maintaining this additional information is considered. It is essential to establish an appropriate level of explicit traceability, one which gives a justifiable added value advantage in balance with the cost.

2 Traceability Identifiers

To answer the modeller's needs for traceability, TMWG is proposing that all workflows and artifacts produced should be identified by a traceability identifier. This explicit mechanism is one of many mechanisms required in a complete traceability strategy. It is independent of the modelling method used and is of low cost to implement. It includes a basic facility to trace related items.

2.1 Naming/numbering convention

A naming/numbering convention is proposed that should ensure bottom-up and top-down traceability of the artefacts produced in the various workflows in the UMM.

The following chart summarises the convention used for the different types of traceability item. These are explained in the further, more detailed, text.

WORKFLOW ITEMS			which may contain
Business Domain Modelling	e-Business Requirements	Analysis	
D-<name>-<number>	D-<name>-<number> .E-<name>-<number>	D-<name>-<number> .E-<name>-<number> .Z-<number>	
is followed by…, for…	*is followed by…, for…*	*is followed by…, for…*	ARTIFACT
.R	.R		Requirement List
.R-<number>	.R-<number>		Requirement
.G	.G		Glossary
.U-<name>-<level>-<number>	.U-<name>-<level>-<number>		Use Case
.U-<name>-<level>-<number> .A-<number>	.U-<name>-<level>-<number> .A-<number>		Activity Diagram
.U-<name>-<level>-<number> .S-<number>	.U-<name>-<level>-<number> .S-<number>		Sequence Diagram
	.C -<number>	.C -<number>	Class Diagram
.C+ -<number>	.C+ -<number>	.C+ -<number>	Class Diagram (whole system)
followed by…, when there is…	*followed by…, when there is…*	*followed by…, when there is…*	SUPPLEMENTARY
.T -<number>	.T -<number>	.T -<number>	Related Item

3 Glossary of Traceability Types

3.1 WorkFlow Items

Type D: Workflow i: Business Domain Modelling
A business domain modelling is identified by the key letter D followed by a name and a unique number.

D-<name>-<number>

note: <number> is a unique number

An example is

D-purchasing-1

Type E: Workflow ii: e-Business Requirement
An e-business area is included into a business domain.
It is identified by the key letter E followed by a name and a unique number within the domain. So an e-business area which is part of business domain D-<name>-<number> will be referred to as

D-<name>-<number>.E-<name>-<number>

note: <number> is a sequential number within a domain

An example of an e-business area is:

D-purchasing-1.E-order from catalog-1

Type Z: Workflow iii: Analysis
An analysis workflow relates to an e-business area within a business domain.
It is identified by the key letter Z followed by a sequence number. So an analysis is in an e-business area which is part of a business domain will be referred to as

D-<name>-<number>.E-<name>-<number>.Z-<number>

note: <number> is a sequential number

An example of an Analysis is:

D-purchasing-1.E-order from catalog-1.Z-2

3.2 Artifact Items

Type R (i): Requirement list

A requirement list is identified by the key letter R.

A requirement list from the business domain modeling workflow will be referred to as:

D-<name>-<number>.R

A requirement list developed in an e-Business Requirements workflow within a business domain will be referred to as:

D-<name>-<number>.E-<name>-<number>.R

Type R (ii): Requirement

A requirement within a requirement list has a unique requirement number and will be referred to as:

D-<name>-<number>.R-<number>

or

D-<name>-<number>.E-<name>-<number>.R-<number>

An example is:

D-purchasing-1.R-10 or D.purchasing-1.E-order from catalogue-1.R-14

Type G: Glossary

A glossary is identified by the key letter G.

A glossary from a business domain modeling workflow will be referred to as:

D-<name>-<number>.G

A glossary from an e-business area within a domain will be referred to as:

D-<name>-<number>.E-<name>-<number>.G

Type U: Use case

A Use Case is identified by the key letter U followed by a name, a level (defined by an integer, in general less than ten) and a unique contextual number.

U-<name>-<level>-<number>

note: <number> is a unique number

So a use case attached to a business domain will be referred to as:

D-<name>-<number>.U-<name>-<level>-<number>

Examples are:

D-purchasing-1.U-purchasing-1-1

D-purchasing-1.U-specify product-2-1

D-purchasing-1.U-source potential suppliers-2-2

A Use Case attached to an e-business area will be referred to as:

D-<name>-<number>.E-<name>-<number>.U-<name>-<level>-<number>

Examples are:

D-purchasing-1.E-order from catalog.1.U-purchasing-1-1

D-purchasing-1.E-order from catalog.1.U-specify product-2-1
D-purchasing-1.E-order from catalog.1.U-source potential suppliers-2-2

A use case is described by a diagram and a text description.The same identifier should be used in both the use case diagram and the completed use case template.

Type A: Activity diagram
In general there is only one activity diagram associated with a Use case. In some complex cases, there may be more than one.
An Activity Diagram is identified by the key letter A followed by a unique contextual number.

A-<number>

note: <number> is a unique number

So an Activity Diagram attached to a use case will be referred to as:

D-<name>-<number>.U-<name>-<level>-<number>.A-<number>
in the business modeling workflow
or
D-<name>-<number>.E-<name>-<number>.U-<name>
-<level>-<number>.A-<number>
in an e-business requirements workflow

Examples are:
D-purchasing-1.U-purchasing-1-1.A-1
D-purchasing-1.E-order for catalog-1.U-purchasing-1-1.A-1
D-purchasing-1.E-order for catalog-1.U-purchasing-1-1.A-2

Type S: Sequence diagram
A Sequence Diagram is identified by the key letter S followed by a unique contextual number.

S-<number>

note: <number> is a unique number

Several sequence diagrams may be developed for each use case. So a Sequence Diagram attached to a use case will be referred to as:

D-<name>-<number>.U-<name>-<level>-<number>.S-<number>
in the business modeling workflow
or
D-<name>-<number>.E-<name>-<number>.U-<name>
-<level>-<number>.S-<number>
in an e-business workflow.

Examples are:
D-purchasing-1.U-purchasing-1-1.S-1
D-purchasing-1.E-order for catalog-1.U-specify product-2-1.S-1
D-purchasing-1.E-order for catalog-1.U-specify product-2-1.S-2

Type C: Class diagram
A Class Diagram can be related either to a use case in an e-business requirement or analysis workflow, or alternatively to an e-business system in the analysis workflow.
A Class Diagram related to a use case is identified by the key letter C followed by a unique contextual number.

A class diagram related to the whole system is identified by C+ followed by a unique number.

C-<number> or C+-1

note: <number> is a unique number

A Class Diagram will be referred to as:

D-<name>-<number>.E-<name>-<number>.U-<name>-<level>
-<number>.C-<number>

or

D-<name>-<number>.E-<name>-<number>.Z-<number>
.U-<name>-<level>-<number>.C-<number>

or

D-<name>-<number>.E-<name>-<number>.Z-<numbe>.C+
for the system use case.

Examples are:

D-purchasing-1.E-order for catalog-1.U-specify product-2-1.C-1
D-purchasing-1.E-order for catalog-1.Z-2.C+-1

3.3 Supplementary Items

Type T: Related Item

A Related Item is any piece of relevant material that is not a direct ouput of the modelling process but which can be related to any workflow or artifact of the modelling process.
A Related Item is identified by the key letter T followed by a unique contextual number.

T-<number>

note: <number> is a unique number

This identifier is attached to the end of the appropriate identification of the model item with which the Related Item is associated. If the Related Item is associated with more than one model item, it is associated with the 'parent' model.item

So a related item associated with a use case attached to a business domain will be referred to as:

D-<name>-<number>.U-<name>-<level>-<number>.T-<number

4 Alternative presentation

Rather than to use long names with separators, it may be more efficient to embed in each diagram a traceability table.

Examples

Example of a requirement list attached to a business domain.

	Name	Level	Number
WORKFLOWS			
i:Business Domain modeling	Purchasing		4
ii:e-business requirements			
iii:Analysis			
ARTIFACTS			
Requirement List	R		
Glossary			
Use case			
Activity diagram			
Sequence Diagram			
Class diagram			

Example of a sequence diagram attached to a use case, it-self attached to an Analysis within an e-business area.

	Name	Level	Number
WORKFLOWS			
i:Business Domain modeling	Purchasing		4
ii:e-business requirements	Order from catalog		5
iii:Analysis	Order from catalog		2
ARTIFACTS			
Requirement List			
Glossary			
Use case	Get quote	2	3
Activity diagram			
Sequence Diagram	S		2
Class diagram			

Example of a class diagram attached to a use case from Analysis within an e-business area.

	Name	Level	Number
WORKFLOWS			
i:Business Domain modeling	Purchasing		4
ii:e-business requirements	Order from catalog		5
iii:Analysis	Order from catalog		2
ARTIFACTS			
Requirement List			
Glossary			
Use case	Get quote	2	3
Activity diagram			
Sequence Diagram			
Class diagram	C		1

Example of a system class diagram from an Analysis within an e-business area.

	Name	Level	Number
WORKFLOWS			
i:Business Domain modeling	Purchasing		4
ii:e-business requirements	Order from catalog		5
iii:Analysis	Order from catalog		2
ARTIFACTS			
Requirement List			
Glossary			
Use case			
Activity diagram			
Sequence Diagram			
Class diagram	C+		1

C+: System Class Diagram

ANNEX 3 Glossary and Requirements List Template

A business modelling project Glossary captures any terms and acronyms the reader might need to understand about the business domain. The Glossary is maintained in a running list by the facilitator throughout the requirements gathering/modelling process. This document is used to define terminology associated with business process modelling as well as terminology specific to the business domain, explaining terms (or groups of terms from a sub-business domain) that may be unfamiliar to the reader of the use-case descriptions or other project documents. Often, this document can be used as an informal data dictionary, capturing data definitions so that use-case descriptions and other project documents can focus on what the system shall do with the information. Reference may be made to external documents that give such details.

In the adoption of the Unified Process to meet the business process modelling needs of UN/CEFACT, several terms considered to be more appropriate for UN/CEFACT are substituted for the terms typically used in the Unified Process. Annex 1, Modelling methodology Glossary, identifies such terms and defines them in the context of this document. These terms should be included in the Glossary of every business modelling project. Annex 1 also illustrates the format of the business modelling project Glossary.

The Requirements List provides an artifact for storing discrete, measurable business requirements and constraints. As requirements and constraints are discovered in performing the modelling steps they are added to this running list by the facilitator of the corresponding workflow. Note: requirements shall be referenced in all modelling artifacts, and if necessary, each requirement should reference modelling artifact(s) that are based on it.

Requirements List Identifier
D-<name>-<number>.R or D-<name>-<number>.E-<name>-<number>.R

Req. #	Statement	Source	Date	Status
Sequential number. Unique to this domain or this e-business requirement area.		Where this requirement came from example: Congressional Order #245 Domain Business needs etc		The current state. Example: open, discontin-ued, active

ANNEX 4 Use Case Specification Template

Use Case Name	The name of this use case.
Traceability Identifier	See Annex 2 for details
Use Case Description	The description should briefly convey the role and purpose of the use case. A single paragraph should suffice for this description.
Actors	List the Actors who participate in the use case.
Performance Goals	A specification of the metrics relevant to the use case and a definition of their goals. (Non-functional requirements may be a source of performance goals) *Name of performance goal* *A brief description of the performance goal.*
Preconditions	State the conditions which shall be met prior to commencement of the use case.
Postconditions	State the conditions which now apply as a result of completion of the use case scenario.
Scenario	A textual description of the scenario the use case represents. The scenario should describe *what* the business does to deliver value to a business actor, not *how* the business solves its problems. *Begins when, i.e., name of the first step in the scenario* *A brief description of the step* *Name of next step in the scenario* *A brief description of the step. etc.* *Ends when, i.e., name of the last step in the scenario* *A brief description of the step*
Alternative Scenario	Describe any alternatives to the above scenario
Special Requirements	The special requirements of the use case are included here. These are requirements not covered by the scenario as it has been described in the sections above. (Non-functional requirements may be a source for special requirements.) *Name of special requirement* *A brief description of the special requirement*
Extension Points	Extension points of the use case. *Name of extension point* *Definition of the location of the extension point in the flow of events*
Requirements Covered	References to all requirements that relate to this use case, by Req. #

ANNEX 5 Use Case checklist

Checkpoints for the Use-Case Model

• Consider if use cases should be factored out into separate use cases.
• Consider developing a new use case for alternative flows/exceptions.
• Consider generalising a use case to enhance reusability.
• Consider defining a new Actor/Role to enlarge the domain of application/reusability.
• Consider combining use cases where similar.

• Do the use cases conform to the business you want them to describe?
• Have all the use cases been found? The use cases should together perform all activities within the business.
• Are all activities within the business included in at least one use case?
• Is there a balance between the number and the size of the use cases?
• Is each use case unique? If not, consider merging it with a similar use case.
• Do the diagrams appear to be well-structured?
• Are the diagrams so large and complex that they should be broken down into several smaller diagrams?

Checkpoints for Actors

• Have all actors been found?
• Does each (human) actor express a role, not a person? Try to name at least two people that can act as the actor.
• Does each actor model something outside the business?
• Is each actor involved with at least one use case? If not, remove it.
• Does each actor represent one role? If not, you should probably split the actor into several actors, each expressing a different role.
• Does each actor have an explanatory name and description that is understandable to people outside the business modelling team?

Checkpoints for Use Cases

• Is its name and brief description clear and easy to understand, even to people outside the business-engineering team?
• Is each use case complete from an outside (actor's) perspective?
• Is each use case involved with at least one actor?
• Is each supporting use case involved with at least one actor? If not, it has to be initiated by an internal event, and does not have to interact with an actor to perform its activities.

For abstract use cases, you may add:

• Is the use case substantial enough to be an abstract use case on its own?
• Does it contain logically related activities?
• Is there a reason for the use case to exist?

Checkpoints for Use-Case Reports

• Is the use-case workflow clear and understandable?

• Is the wording informal enough to be understood by people outside the project team?
• Does it describe the workflow, and not just the purpose of the use case?
• Does it describe the workflow from an external viewpoint?
• Does the use case perform only activities inside the business?
• Are all possible activities that belong to the use case described?
• Are only actors that interact with the use case mentioned?
• Are only activities that belong to the use case described?
• Does it mention only use cases with which it is connected?
• Does it clearly indicate when the order of activities is not fixed?
• Is the workflow well-structured?
• Are the start and end of the workflow clearly described?
• Is each extends-relationship described clearly so that it is obvious how and when the use case is inserted?

Checkpoints for Supplementary Business Specifications

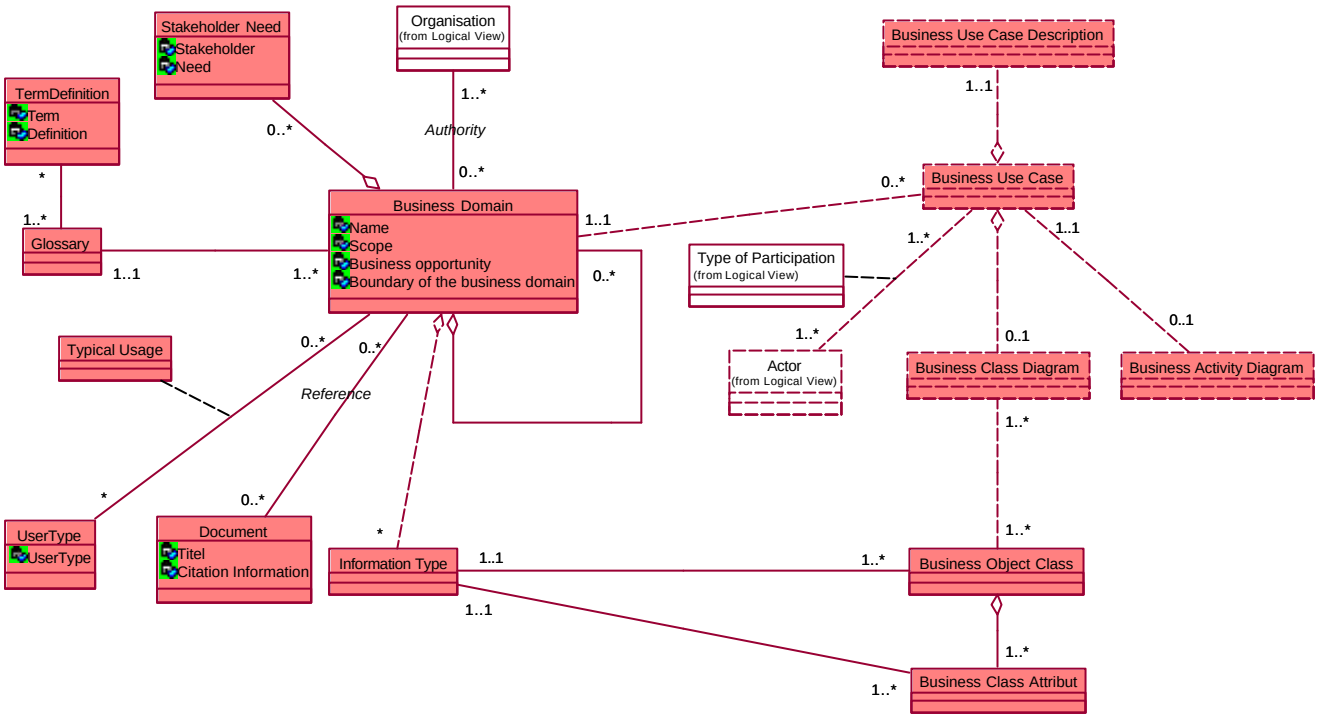
• Are all supplementary business definitions listed in the document general, in the sense that none of them should pertain to one single use case or business entity?
• Are all relevant business rules listed or referred to?
• If the business rules are not in the document, are the references correct and easily accessible for project members?

Checkpoints for the Glossary

• Does each term have a clear and concise definition?
• Is each glossary term included somewhere in the descriptions of the use cases? If not, it may imply that a use case is missing or that the existing use cases are not complete. It is more likely, though, that the term is not included because it is not needed. In that case, you should remove it.
• Are terms used consistently in the brief descriptions of actors and use cases?
• Does a term represent the same thing in all use cases?

ANNEX 6 Metamodel of the UN/CEFACT Modelling Methodology

Business Domain Meta Model Class Diagaram



Annex 6.2 e-Business Requirements Meta Model Class Diagram

Annex 6.3 Analysis Meta Model Class Diagram

Annex 6.4 Design Meta Model Class Diagram

ANNEX 7 Naming & Style Guide

Introduction

ThisA style guide tells you what the Business models you are creating should look like in a physical sense. ~~Business models are no different from other documents in this sense.~~ It begins with the general rules and syntax for naming and description of typical modeling elements. It follows with a section on the conventions applied to typically used diagrams. Finally it concludes with a description of the rules that apply specifically to the model of an individual commercial transaction as a UML activity model.
~~The style guide is complete in the sense that it covers rules for layout and naming that will work as stated. It is incomplete, at this point, in that all examples have not been exhaustively covered. Additionally, processes themselves change over time to adjust to changing requirements and newly discovered insights. This style guide will also adapt to such changes. Following the style guide will help to make any changes to defined processes, less capricious reasoning and more consistency in application.~~

Scope This annex specifies style conventions for layout and documentation of business process models built in accordance with the *e-Business Collaboration Modeling Metamodel.Ref XXX*

Style Conventions

Typographical and language conventions are used to convey specific meanings.

Typographical Conventions

The use of a ***bold/italic font*** indicates a UML or business process metamodel entity name.

Language Conventions

This specification adopts the conventions expressed in the IETF's[1] RFC 2119 "Key Words for Use in RFCs to Indicate Requirement Levels." The key words "SHALL", "SHALL NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

General Naming and Syntax Rules

The following general rules apply to all business modeling products:

Font

- All lettering on UML diagrams shall be formatted to use the Arial 8pts, black font.
- example: **Arial 8pts, black font**

General Syntax

[1] http://www.ietf.org

- All specification names shall be syntactically compliant with the OCL version 1.0 specification. This rule may be relaxed for readability and user understanding for the Business Domain Modelling and e-Business Requirements workflows.
- Abbreviations (including acronyms) shall not be used for naming any elements of the model except for well-known standards (e.g., ID).

Roles and Role Names

- A role name shall not be a partner type.
- A role description shall be the approved description corresponding to the selected role name.
- The role description cannot be changed or extended without first being approved by the technical modeler .
- Roles shall be defined to be one of the following types:
  1. Organizational: Authority to perform on a company level. If you want to authenticate the entire business document.
  2. Employee: Authority to perform on an individual level. If you want to authenticate a role (digital signature). If you want to authorize specific data.
  3. Functional: Can be either an organizational or an employee role.

Naming of Classes and Properties

- No use of "type" or "class" for names of entities and properties.
- No properties shall have "yes", "no" values.
- Use the word "Identifier" to prefix all fundamental business data entities that are used to identify real world concepts e.g. ProprietaryProductIdentifier. Identifiers are not enumerated as values for a fundamental business data entity.
- Use the word "Code" to prefix all fundamental business data entities that are used to identify real world concepts with a literal (symbol) e.g. GlobalPartnerClassificationCode. A codes list is enumerated as the values for a Fundamental Business Data Entity.
- Use the word "Global" to precede globally administered entities.
- Use the world "National" to precede nationally administered entities.
- Use the word "Proprietary" to precede proprietary administered entities.
- Use "is" to prefix business properties that are boolean (True/False) valued, e.g. ***isOnSale***.
- Keep tense out of business property names e.g. use "priceChangeDate" and not "PriceChangedDate" as the property can then be used for both past and future business events.
- All non-global e.g. proprietary, national, regional, geographic, identifiers and codes shall be business data entities. They shall comprise two fundamental business data entities. One specifying the administering authority and one specifying the identifier.
- Do not use "yes", "no", "true", "false" etc. as state transitions out of conditional states. Use active verbs e.g. accepted, not accepted, verified, not verified etc.
- Only name business properties if they are used to differentiate between associations to the same target class, or when they provide context but no attributes to a more general class.

- Using the name of a Class with a lower case first letter to reference objects that have un-named roles. This is specified in section 2.3, page 15 of the book titled "The Object Constraint Language."
- All business properties of type “CountableAmount” shall start with the words “numberOf” e.g. “numberOfProducts”.
- Class names shall comprise compacted proper names (i.e. no white space or special character delimiters) e.g. ***BusinessRole***. The first letter of each name shall be capitalized. This rule does not apply to association class stereotypes in the metamodel. These class names shall follow the syntax for associations. (Applies to BOV, FSV, and ISV. May be relaxed for BOM and BRV.)

Associations

- The first letter of an association name is small. The rest of the name is compacted proper names (i.e. no white space or special character delimiters) e.g. ***businessProcessFlow***. (Applies to BOV, FSV, and ISV. May be relaxed for BOM and BRV.)
- Use singular names for associations with singular cardinality.
- Use plural names for associations with multiple cardinality.
- Association naming phrases. If a name is associated with a specific UML role then the phrase shall start with the word “the”. If the name is associated with the general role then do not use the word “the” to precede the phrase.
- All associations shall have cardinality specified on the role of the association.

Operations

- The first letter of an operation name is small. The rest of the name is compacted proper names (i.e. no white space or special character delimiters) e.g. ***createQuickerly***. (Applies to BOV, FSV, and ISV.)
- Operation names shall be verbs e.g. ***add()***.
- Operation argument names (when needed) shall be nouns e.g. ***PurchaseOrder***.

Stereotypes

- Stereotype names shall syntactically match the e-business process metamodel stereotype names or the UML stereotype names. The UML stereotype names are all lower case. The e-business metamodel stereotype names are compacted proper names. (Applies at all model levels.)

Conditional statements

- Preconditions, post-conditions and invariant constraints shall be expressed using text and the OCL. (Applies at all Models levels. May be relaxed for BOM. Should be applied for BRV.)

Diagram Layout Rules and Conventions

Use Case Diagrams

- U…[Section to be written]……..

Class Diagrams

- For unidirectional navigational aggregation roles the name of the role shall be the name of the “A” role. Do not name the general role. This is particularly true for Business Data Entities.
- Business data entity attributes shall not be specified in attribute compartments. All attributes shall be specified as unidirectional aggregation associations. (Note: Modeling as association facilitates management of optional properties and the automatic generation of specifications.)
- Hide operation compartments when there are no operations defined for a class.
- Hide attribute compartments since there will be no attributes defined for a class.

Activity Diagrams

- Activity diagrams may span multiple vertical pages.
- Layout activity diagrams on portrait oriented pages.
- It is recommended that multiple transitions with guard conditions be shown from activities rather than using decision activities. This reduces clutter in the diagrams.
- Activity diagrams shall have one initial state and can have multiple final states.
- Asynchronous commercial transactions in activity diagrams shall be represented a single object flow.

Sequence Diagrams

- Sequence diagrams may span multiple pages. It is recommended that they do not span multiple vertical pages.
- Layout sequence diagrams on landscape oriented pages.
- Interactions shall be indicated as simple interactions.
- Response interactions shall be indicated as simple interactions.
- A sequence diagram shall have all the network component interactions that implement one commercial transaction. This includes all agent and service interactions.

Commercial Transactions and Commercial Transaction Activity Diagrams

Rules in sections two and three apply for all commercial transaction activity models. There are several other specific rules and conventions that make consistent commercial transaction modeling possible. These are outlined below.

Initiator / Responder (applied to roles):

- The initiator is the role that is responsible for managing the start state.
- The initiator is on the left by convention for readability and diagramming.

Activity and Business Document Names

- The activity names shall always be in the form <Verb><Noun>
- Business Document names shall be in the form <Noun><Verb>

The verbs should be selected from the following table. A new verb can be used if the business requirement is semantically different from any of the existing verbs.

	Initiating Business Activity	Responding Business Activity	Business Document Request	Business Document Response
Business Transaction	1. Cancel 2. Change 3. Create	Accept	1. Cancellation 2. Request 3. Change	Acceptance
Request / Confirm	Request	Confirm	Request	Confirmation
Query / Response	Query	Process	Query	Response
Notification	1. Notify 2. Transfer	Process	1. Notification 2. Notification	NA
Information Distribution	Distribution	Receive	Notification	NA

Process Flow Patterns for Commercial Transactions

- A Business Document shall always be in the same swimlane as the Business Activity that creates it.
- The process flow is always counter clockwise. The design rational for this is so that it more closely models the flow of a transaction as opposed to an interaction.
- There shall always be exactly two swimlanes.
- There shall always be exactly one initial state named START.
- There shall always be exactly two final states named FAILED and END.
- The guards on the two end states are always named SUCCESS and FAILED.
- The names of Activities, Business Documents, and Role Names shall be in proper case. This allows us to easily read OCL compliant names when we use them to create business constraints.
- Guards shall be in uppercase so that they can be used as values in OCL syntax.

Initial and Final States

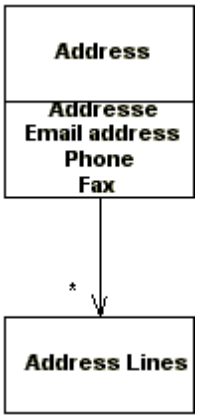
- The initial and final states describe the state of the business process support system
- States conditions are named in the form <Noun><Property><Verb>
- The <Noun> can be a Business Data Entity and the property is named “Status” in the form BDE Status <Verb>. Purchase Order Status Exists
- The <Noun> can be a Business Data Entity with no named property in the form BDE <verb>. Purchase Order Exists

- The <Property> can be the name of a business process support system with no <Noun> in the form <Property><Verb>. Signature Authorized.
- All states on the diagrams shall be in the same swimelane as the initiating activity. This is because a process cannot start or terminate on the responding role's swimlane.
- Every FAILED end state shall have the Notification of Failure as one of the conditions.

Annex 8 Describing addresses

Addresses are basic to most business systems. We need to record location information and how to contact our customers, suppliers, and employees. By recording this information, we are able to send letters, invoices, and purchase orders, and to contact them by phone, fax, or e-mail. One way to model this is shown in the figure below. Here we have an Address with the basic contact information, addressee's name, email address, and phone numbers. It also owns a group of address lines to give us a flexible way to record all the different kinds of addresses we may need.

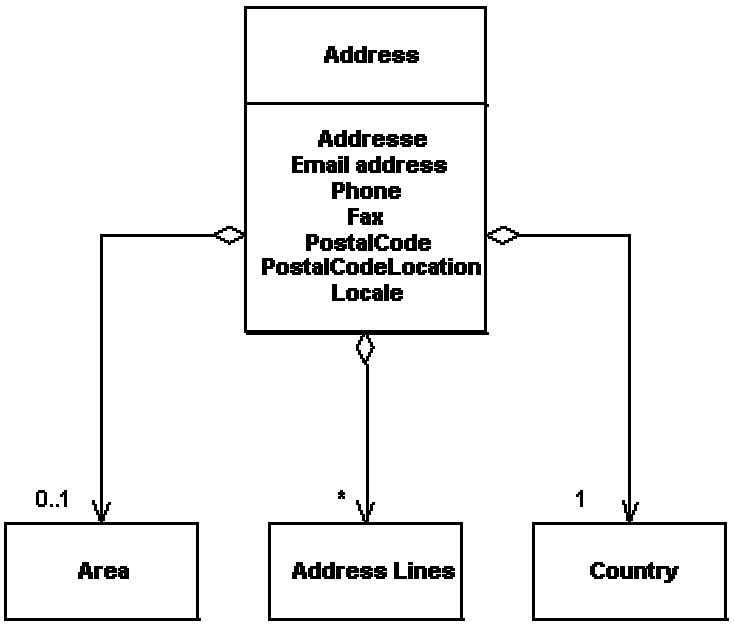
Basic address diagram



Using addresses in SanFrancisco

The SanFrancisco class that records this information is called Address. In SanFrancisco the same kind of address object is used both for the company's addresses and for the addresses of the company's business partners. Address contains the basic contact information, a collection of free form address lines and references to a country, and optionally a user-defined "area." (You can think of area as an application-defined equivalent to a region.) The following diagram shows the structure of a SanFrancisco address.

SanFrancisco address implementation model



Working with address information

The following table shows the names and types of the various attributes of an address. Most are Strings. The exceptions are Country and Area which are references to persistent objects. SanFrancisco does not impose any required format on the Strings, although LocaleInformation should conform to the standard for Java locales ("en_US", for example).

Address attributes table

Attribute Name	Type	Can Update?	Purpose
Addressee	String	Y	The person this location is associated with
PostalCode	String	Y	The Zip Code / PostalCode for this location
PostalCode Location	String	Y	The city for this location
PhoneNumber	String	Y	The phone number for this location
FaxNumber	String	Y	The fax number for this location
EMailAddress	String	Y	The email address for the addressee
Locale Information	String	Y	The locale code for this location
Country	Country	Y	A reference to the country for this location
Area	Area	Y	A reference to a user-defined area (sales, shipping, etc.)

At first glance you might think that the use of a single PhoneNumber is a limitation, but you will see that, in places where addresses are used, an open-ended collection is maintained. Therefore, you can have a separate address object for each phone number you need to maintain for a company or business partner. This could also be handled by using the generic address line support to add address lines that represent additional phone numbers.

Working with address lines

In addition to the String attributes, a SanFrancisco address has a collection of address lines. Each address line is a String. Each address line has a key, which is defined by the user or application. If all you want to do is store an image of an address label, you might key your address lines with sequential numbers; if you want more meaningful access to the address information, use keys like "street address", "city", "state", and so on.

```
Class Address
public void addAddressLineBy(String addressLine, String key)
public boolean containsAddressLine(String addressline)
public boolean containsAddressLineKey(String key)
public Iterator createAddressLineIterator()
public String getAddressLineAt(Iterator position)
public String getAddressLineBy(String key)
public String getAddressLineKey(String addressLine)
public String getAddressLineKeyAt(Iterator position)
public DMap getAddressLines()
```

The next example shows how to construct a display address from an Address object.

Example -- Display address

```
public void displayAddress() throws SFException {
// display a typically formatted U.S. address
        Address theAddress = CompanyContext.getActiveCompany().getPrimaryAddress();
        System.out.println(theAddress.getAddressee());
        System.out.println(theAddress.getAddressLineBy("addr1"));
        String addr2 = theAddress.getAddressLineBy("addr2");
        if (addr2 != null) {
                System.out.println (addr2);
        }
        System.out.println(theAddress.getAddressLineBy("city")
                + ", "
                + theAddress.getAddressLineBy("state")
                + "  "
                + theAddress.getPostalCode());
}
```

Finding addresses

Unlike many persistent Entities in SanFrancisco, addresses are not maintained by controllers. Instead, addresses are explicitly associated with the business object whose address it is. A company object will own its addresses and a business partner will own its address(es) and each provides methods to access and maintain addresses.

For more information about configuring a new address, see Configuring an address. For more information about extending the address object to provide additional data or functionality, see Extending an address.

ANNEX 9 References

ISO/IEC IS 14662 Information Technologies - Open-edi reference model

The Unified Modeling Language User Guide. Jacobson, Booch, Rumbaugh, 1998, Addison Wesley-ISBN 0-201-57168-4

The Unified Software Development Process. Jacobson, Booch, Rumbaugh, 1999, Addison Wesley Longman, ISBN 0-201-57169-2

Use Cases - Requirements in Context, Daryl Kulak and Eamonn Guiney, Addison-Wesley, 2000, ISBN 0-201-65767-8

The Unified Modeling Language Reference Guide, Rumbaugh, Jacobson, Booch, 1999, Addison-Wesley,

Applying Use Cases: A Practical Guide, Schneider and Winters, 1998, Addison Wesley Longman, ISBN0-201-30981-5

Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design, C. Larman, 1997, Prentice Hall

UML in a Nutshell: A Desktop Quick Reference, Sinan Si Albir, 1998, O'Reilly & Associates, Inc., ISBN 1-56592-448-7

e-Business Collaboration Modelling Metamodel, Draft 2.0, BCF#7, Edifecs Commerce.

e-Business Collaboration Design patterns, draft 2.0, BCF#8, Edifecs Commerce

FRENCH REFERENCES

Modélisation Objet avec UML, Pierre Alain Müller, 1997, Wrox Press

UML dans vos projets, Nathalie Lopez, Jorge Migueis, Emmanuel Pichon, 1998, Eyrolles

UML en action, Pascal Roques, Frank Vallée - Eyrolles

Web - http://www.essaim.univ-mulhouse.fr/uml