The Basic Data Program

Model Rules for Basic Data

Version 1.1.0

January 2015

Foreword

The model rules were prepared as part of establishing the basic data model: a common data model for all basic data.

The establishment of the basic data model involves a number of other products and deliveries to which it must be possible to apply the model rules:

- **Modelling tools**
  Tools that support the establishment and maintenance of the basic data model in compliance with the model rules.

- **Commissioning plan**
  The plan for commissioning the model rules throughout the entire Basic Data Initiative. Publication expected alongside model rules version 1.0.0.

- **Distribution platform**
  The platform for distributing the basic data model for data users.

- **Management framework for the basic data model**
  Agreement on the maintenance of the model rules and basic data model, including the definition of the organisation and distribution of responsibility, rules for versions and establishment of decision processes for making amendments. Publication expected before the end of 2013.

The basic data model must be included in the combined documentation of basic data, which will also include documentation from the data distributor.

Version history

Version	Date	Status	Notes
0.1	14/03/2013	Draft	First draft for general properties on the basis of workshop discussions on 14 January, 5 February and 13 March 2013.
0.2	22/03/2013	Draft	Workshop discussions on 20 March 2013 written up.
0.3	17/04/2013	Draft	Outline created for entire document, introduction added, outline for chapter on general model rules, revision of general properties and notes for missing sections.
0.4	15/05/2013	Draft	Workshop discussion on 23 April 2013 incorporated. All chapters edited. New document structure.
0.5	21/05/2013	Draft	Incorporation of the first comments from the external review (all chapters).
0.6	24/05/2013	Draft	Incorporation of the remaining comments from the external review (all chapters). Amendments to chapters 1, 2 and 3. New rules in chapter 5. Chapters 6 and 7 combined into a new chapter 6. Corrections and more detailed descriptions in all chapters.
0.7	28/05/2013	Draft	Incorporation of comments from workshop on 27 May 2013.
0.8	30/05/2013	Draft	Incorporation of comments from meetings with ERST and MBBL.
0.9	31/05/2013	Draft	Ready to be sent to the steering committee.
1.0.0	24/09/2013	Draft	Amendments as a result of comments and POC implemented. Ready to be sent to the project committee. This version has tracked changes.
1.0.0	02/10/2013	Draft	Version without tracked changes. An example of an application of the rule is inserted in some cases as per the rules. Ready for the steering committee.
1.0.0	10/10/2013	Draft	Corrections.
1.0.5	10/01/2015	Draft	English version – small translation issues. NOTE: intermediate version between 1.0.0 and 1.1.0
1.1	23/01/2015	Vers.	English version – final edition – should mirror version 1.1

Table of contents

1 Introduction

1.1 Objective

The Basic Data Initiative was initiated with a view to ensuring that public basic data about individuals, businesses, property, addresses and geographic conditions is updated in one location and used by everyone. A more detailed background to the Basic Data Initiative can be found here [Basic Data Initiative].

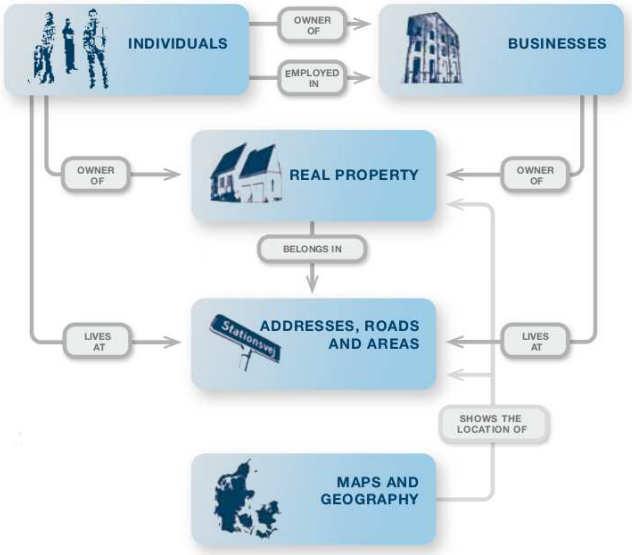


Figure 1 – Conceptual overview of the Basic Data Initiative

Public basic data is maintained and used by many different authorities, and that is why there is a need for all data to be combined in one model in order to ensure a comprehensive overview and thus avoid redundant data maintenance.

The Basic Data Initiative contains different business domains that are related to each other and, in certain areas, overlap. To create a data model for basic data that interested parties find consistent, it is important to ensure that there is common access to the modelling task. The model rules ensure that the modelling of data objects takes place on the basis of a common set of guidelines, and that the entire model is based on common fundamental properties.

The *objective* of the model rules is therefore to ensure a collective and consistent basic data model in a distributed management environment.

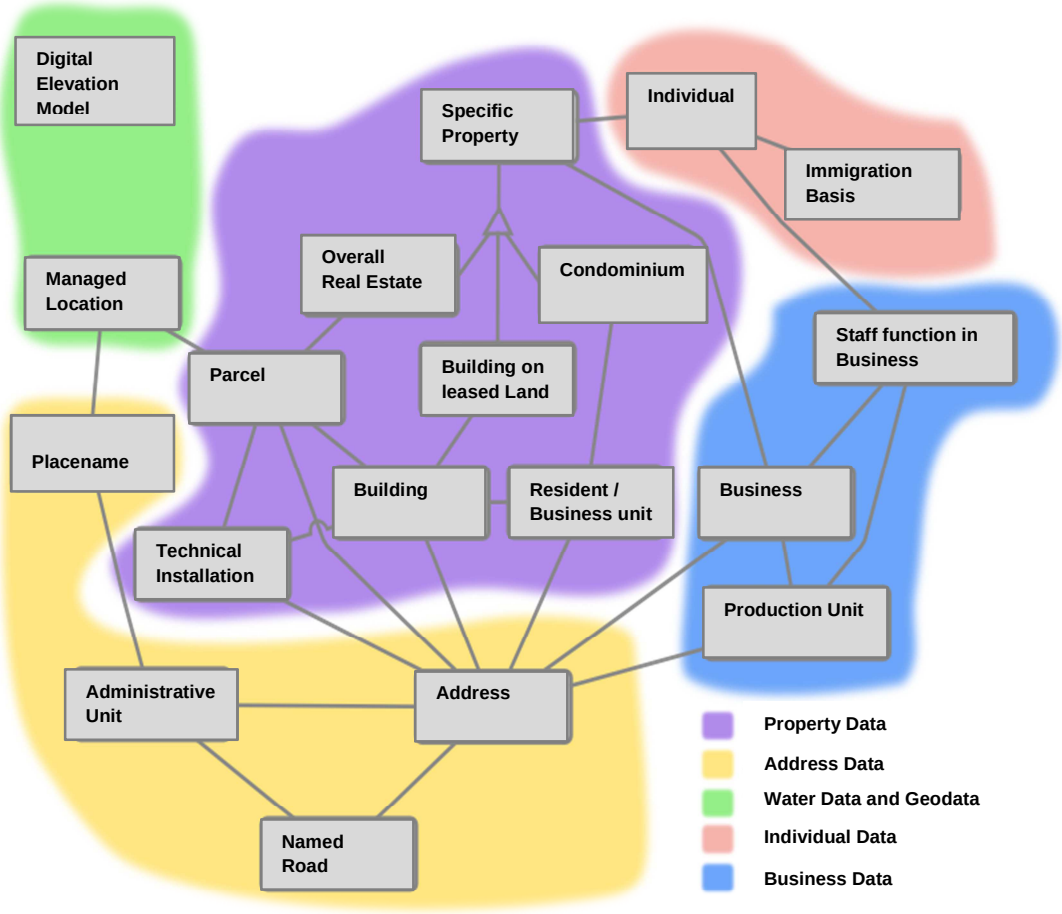


Figure 2 – Overview of business domains in basic data based on [Conceptual data model version 0.8] from 2011. A business domain is e.g. "Business". The correct demarcation of business domains is defined by the basic data administrators.

1.1.1 What is a collective and consistent basic data model?

We want to provide end users (authorities, businesses and private operators) with a collective and consistent data model. This means that users will experience consistency across business domains and uniform concept application as well as uniform modelling and general properties for model entities in the basic data model. This experience is sustained and maintained even though the data in the model are maintained by different authorities.

1.1.2 Objectives

The primary aim of the model rules is to create a common approach to modelling basic data that are necessary for creating a collective and consistent data model.

In concrete terms, the model rules must satisfy the following objectives:

- The model rules must form a basis for the uniform modelling of basic data.
- The model rules must ensure the necessary abstraction level required to satisfy the needs of all interested parties.
- The model rules must ensure the reuse of existing standards where possible.

- The model rules must make it simple for data users to build applications that use basic data and to submit uniform queries spanning the breadth of the basic data.

1.1.3 Advantages

Basic data authorities will experience a number of advantages to using the common model rules:

- It is easier to ensure shared guidelines for data modelling in-house in the organisation.
- The data model spans all business domains and thus provides the opportunity to reuse data.
- It is simpler to exchange data objects.
- It is simpler to ensure high data quality.
- There is less concept confusion.
- There will be less redundant data spanning business domains.

1.2 Target audience

The model rules have four primary interested parties:

Data users

Data users are end users who, through the Basic Data Initiative's use of the model rules, will experience a collective, consistent and effective way of accessing and using basic data.

Data owners

The data owners operate from within the individual registration authorities that store, maintain and distribute basic data. The data owners are very interested in a collective, consistent data model with options for reuse and effective governance. The model rules constitute an important part of the basis for realising this advantage.

Developers

In this document, developers shall refer to the managers, project managers, business experts and systems suppliers among data owners and data users who shall deliver solutions in the Basic Data Initiative for the distribute and use of data. They are very interested in model rules that will ensure a collective, consistent data model. This group will require the collective data model to be presented at several different abstraction levels: conceptually, logically and physically.

Data modellers

The data modellers who will design the basic data model though their work modelling business domains are dependent on the model rules being unambiguous, well-defined and meaningful

1.3 Reading instructions

1.3.1 Content

The content of this document is as follows:

- **Chapter 2 – Focus and demarcation of the model rules**
  The focus and demarcation of the basic data model and model rules are described here.

- **Chapter 3 – Architecture conditions**
  The architecture and infrastructure conditions that affect the design of the model rules are described here.

- **Chapter 4 – Using the model rules**
  This chapter explains how the model rules are constructed and how they should be observed. The rules themselves follow in chapters 5 and 6.

- **Chapter 5 – General model rules**
  This chapter sets out general model rules, focusing on the data model design and diagramming. Here the rules are set out for e.g. modelling language, naming of elements, language, documentation, etc.

- **Chapter 6 – Rules for general properties**
  This chapter sets out rules focusing on the content of the data model and provides the framework for data content in the management objects. It is here that rules significant to e.g. management objects' identification and history are set out. The rules are utilised in the specification of general properties for all model entities.

- **Chapter 7 – References**
  References in the text are specified within square brackets: [...] and refer to chapter 7.

The following appendices are attached as aids for practical modelling work:

- **Appendix 1 – Table view of rules**
  Appendix 1 provides a summary of all the rules from chapters 5 and 6 in table form. The table can be used as a "checklist" in connection with preparing a data model.

- **Appendix 2 – Table view of general properties**
  Appendix 2 provides an overview of the general properties from chapter 6 in table form. The table can be used as a "checklist" in connection with system design.

- **Appendix 3 – Documentation of the data model**
  Appendix 3 provides an overview of how the data model should be documented. This Appendix is an elaboration on rule 5.9. The overview can be used as a "checklist" in connection with documenting the data model.

1.3.2 Definitions

Definitions of concepts used in the model rules – words in bold refer to other definitions. Where possible, definitions that do not build on other definitions in the list are retrieved from external definitional works.

Concept	Definition
Operator	An object that participates in an activity. Not all objects can be operators because not all objects can be said to participate. Some objects will instead be included e.g. as tools in an activity. [Den Fællesoffentlige Topontologi] 3 Operator
Architecture building block	A demarcated part of the IT architecture that specifies a set of business capabilities. The building block may be considered a reusable and replaceable part of the architecture, and can be described in more or less detail. TOGAF 3.21 Building block http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap03.html#tag_03_21
Event	An activity that constitutes an isolated whole. An event takes place at a specific time and in a specific location. [Den Fællesoffentlige Topontologi] 1.2.1 Event
Event distributor	A **computer system** that exchanges **event information** between **computer systems**. The eventdistributor creates an **architecture building block** with corresponding business capabilities. A eventdistributor is established as part of the [Joint Municipal Framework Architecture].
Computer system	One or more computers, peripheral equipment, and software that perform data processing. ISO/IEC 2382-1:1993, 01.01.20
Data	Re-interpretable representation of information in a formalized manner suitable for communication, interpretation, or processing. ISO/IEC 11179-4:2004(en), 3.2.6 data
Data distributor	A **computer system** that effectively and reliably distributes **data** from the basic data registers. The data distributor creates an **architecture building block** with corresponding business capabilities. The joint municipal data distributor is established as part of the [Basic Data Initiative].
Data event	An amendment to **data** – as opposed to the **event** in "reality" that gave rise to the amendment to **data**.
Data model	Graphical and/or lexical representation of **data**, specifying their properties, structure and inter-relationships . ISO/IEC 11179-1:2004, 3.2.7 data model
Data object	A concrete data instance of a **model entity**. One example of this is a person object: Person ("Jens Hansen", "010101-8881", "01-01-2001"). See also Figure 3 below.
Data interfaces	The specifications (e.g. XML Schema, JSON Schema, WSDL) that establish the framework for the data format in a **service**.

Domain model	A **data model** for a business domain (e.g. “Address”, “Place name”, “Business”). All the domain models combined constitute the **basic data model**.
Business event	An **event** in “reality” that has given rise to an amendment to **data**.
Management	The consistent, **data**-based practice of public authorities in Denmark.
Management object	The **management**’s representation of the concrete – physical or conceptual – existing object (address, watercourse, business) over which authority has been exerted and on which data are therefore being collected. The management object is an independent whole that is easy to describe and has associated information. E.g. the management object “Person” might have the following associated information: “Name”, “Civil registration number” and “Date of birth”. See [Architecture Guide – business object] and Figure 3 below.
Basic data	The **data** stored and managed by **basic data registers**.
Basic data model	The collective and consistent **data model** for **basic data**. The basic data model comprises **domain models**.
Basic data register	A data collection that aims to collect and forward **data** relating to **management objects** and that participates in the [Basic Data Initiative].
Event notification	A document prepared by a **computer system** in connection with a **data event** intended to be forwarded to other **computer systems** to advise these of **data events** to which they might react.
Information	Characters that convey meaning. [Den Fællesoffentlige Topontologi] 1.6.1 Information
Information model	A **data model** where the focus is on the description of the specific names, properties and **relations** of **model entities** as well as their multiplicity and cardinality. Corresponds with the Logical Data Model, see [Architecture Guide – information model]
Classification component	A **computer system** that creates an **architecture building block** that aims to store, synchronise and distribute classification systems.
Conceptual data model	A **data model** where the focus is on the description of the **management objects**’ overall **relations**.
Model	An object that represents an entity by possessing a genuine subset of its properties A model can resemble the original to the extent that it is mistaken for it, but it is not the original. A model of Vor Frue Kirke may have the same form as the original, but deviates in terms of e.g. materials and size. However, a copy has the same properties as the original. [Den Fællesoffentlige Topontologi] 1.5.5 Model

Model owner	The authority that owns and is accountablefor a **domain model** at any given time.
Model entity	The **modelling** of a **management object** where its properties are expressed as classes and attributes. See also Figure 3 below.
Modelling	The action of making a **model** of something.
Organisation component	A **computer system** that creates an **architecture building block** that aims to store, synchronise and distribute **information** about organisations – their contact details, employees and internal organisation.
Relation	An entity that connects entities. A relation is the relationship between two or more entities. [Den Fællesoffentlige Topontologi] 1.3 Relation
Service	A business, **architecture building block** or **computer system**'s ability to provide services to internal or external consumers.

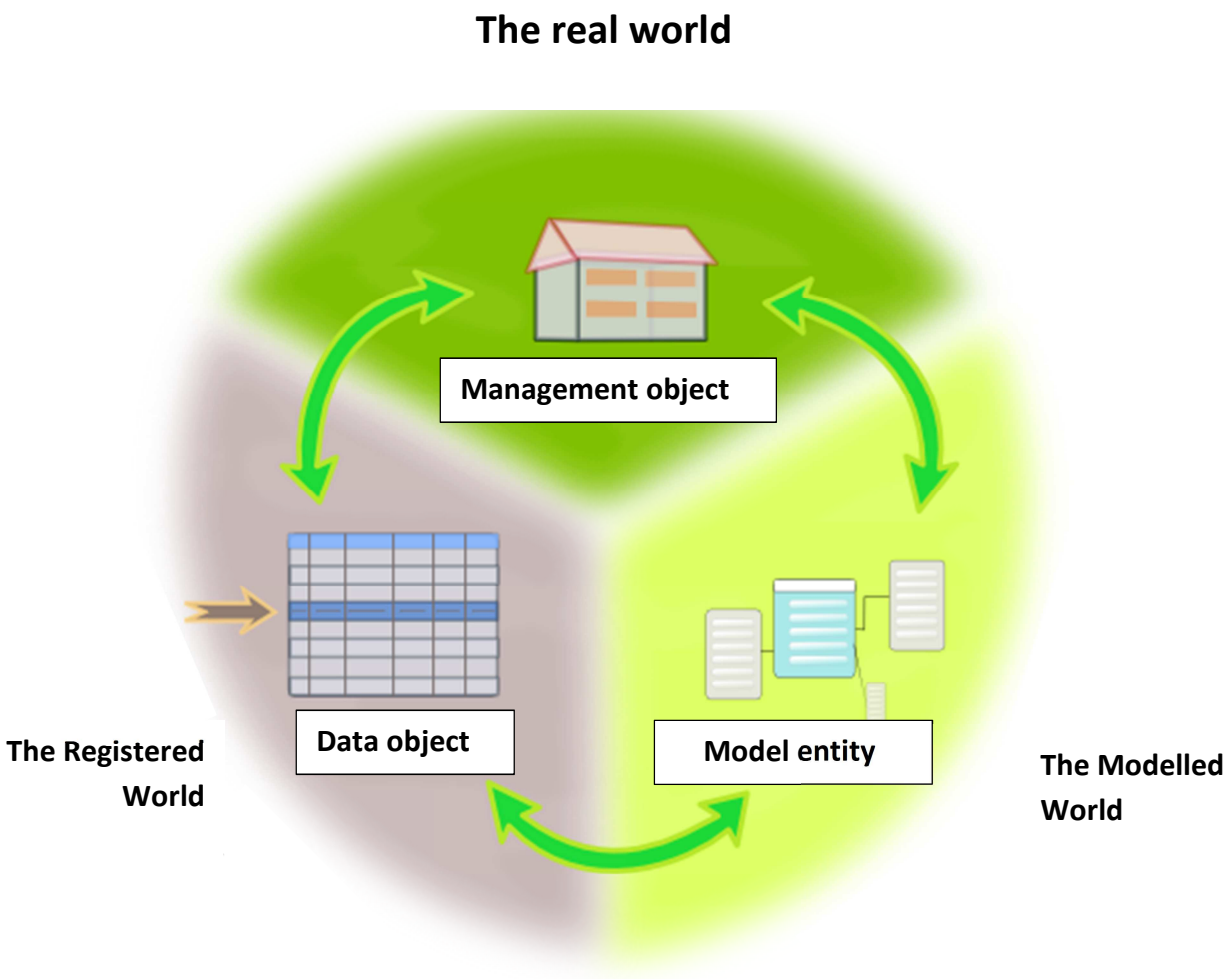


Figure 3 – Illustration of the relationship between the concepts "Management object", "Model entity" and "Data object".

1.3.3 Abbreviations

Abbreviation	Description
UML	Unified Modelling Language, www.uml.org
XML	Extensible Markup Language, www.w3.org/xml

2 Focus and demarcation of the model rules

The focus and demarcation of the model rules are described here.

2.1 The basic data model comprises domain models

The basic data model comprises domain models – i.e. data models for all the business domains in the Basic Data Initiative (e.g. “Person”, “Address”, “Business”). See also Figure 2.

Each domain model has a model owner – i.e. the basic data authority that owns and is responsible for a domain model at any given time. It is up to the model owner to establish the scope and demarcation of the domain.

The model owner is accountable for ensuring that the domain model reflects the domain data and is modelled in compliance with the business requirement and the model rules.

The collective basic data model is stored and distributed in a central location1.

2.2 Model for distributing basic data

The basic data model must be the model used to distributing basic data from the data distributor.

The focus of the model rules is therefore the distribution and communication of basic data to data users who retrieve data via the data distributor.

Data users constitute both external data users and the authorities participating in the Basic Data Initiative. The basic data authorities have undertaken to use each other’s data and will thus also retrieve data from the data distributor. The basic data model will thus also constitute the collective program’s overview and understanding of basic data.

The model rules do not apply to data models for the storage or updating of basic data in-house in the basic data registers. The model rules do not apply to data models for internal storage in the data distributor or for the data flow between basic data registers and the data distributor.

Please note that the model rules in chapters 5 and 6 require certain information to be contained within the basic data.

---

1 This is described in more detail in the document “Distribution platform”, which is expected to be published before the end of 2013: Domain models are submitted by the model managers in XML format. XMI is a standardised XML-based exchange format for UML models. The UML model is organised in packages, where the package name (see section 5.2) corresponds with the data domain, e.g.: Person, Building, Specific Property.
The basic data secretariat puts these models on a version-controlled distribution platform – most likely a subversion server (http://subversion.apache.org/) at Digitaliser.dk (https://svn.softwareborsen.dk/) – from which they can be reused by everyone. Most modelling tools can import XMI from a subversion server so that modellers can reuse elements in their own modelling. The basic data secretariat shall also ensure that the modelling is distributed in diagram form in a central location in the basic data register so that modellers and others can orientate themselves in the model without having to import it into their own tool.

2.3 Information model

The model rules focus on logical information modelling of the data distributed as basic data for data users. See the Architecture Guide for a more detailed definition of logical information modelling: [Architecture Guide – information model].

The information model must describe all the information distributed as basic data.

One of the aims of the basic data model is a model-driven architecture that collates the maintenance of data models and documentation in one location. In this context this means that the information model is the central data model maintained by the model owner. Data models on other abstraction levels may be derived from the information model: *conceptual data model* and *data interfaces*.

The *conceptual data model* must provide a general overview of basic data used by decision-makers and for communication, see e.g. [Conceptual data model version 0.8]. The conceptual data model must be maintained alongside the information model.

Data interfaces (understood as e.g. physical schemas in the form of XML Schema) aim to make the data model operational for system developers. One of the aims of the model rules is to allow automatic generation of data interfaces from the information model2.

2 The aim of many of this document's rules is for the information model to be automatically converted into data interfaces formulated e.g. in XML Schema or JSON Schema. An investigation of whether the model can be converted for other modelling methods such as RDF/OWL will also be carried out in order to form a basis for Linked Data/Semantic Wed-related data exchange methods.
The methodology for these conversions and the resultant derivative conditions in the domain models are not fully detailed at present. The role that models and interfaces play for the data distributor has also not been clarified (see section 3.2). Technological pilot investigations and business-oriented investigations shall clarify these conditions, generate processes for conversions and give insight into what this will require of the domain models.
The "Modelling tools" and "Distribution platform" documents that the basic data secretariat is expected to publish will contain details about methodology and processes. Upcoming versions of the model rules will be affected by the processes' requirements of the model.

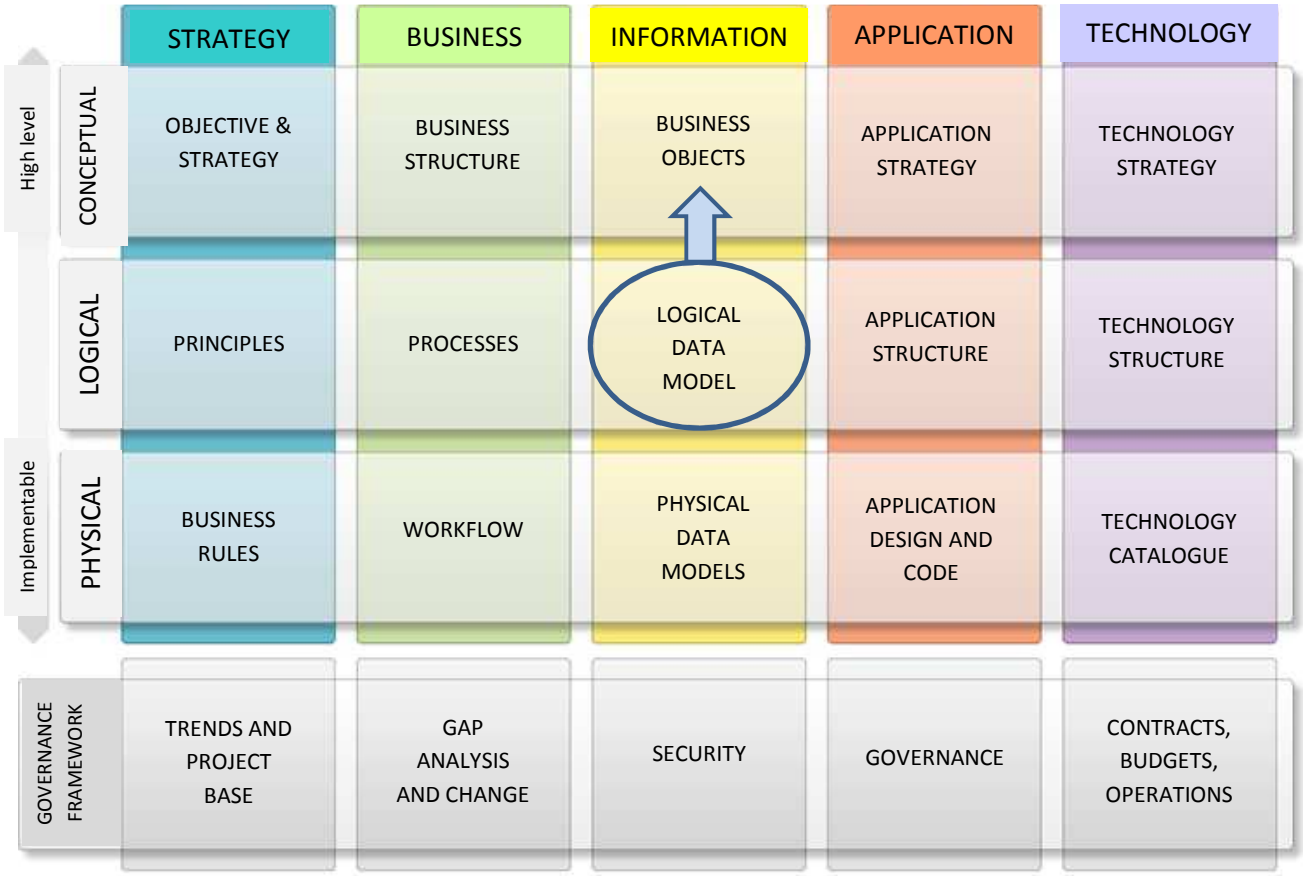


Figure 4 – The focus of the model rules illustrated in relation to the OIO EA table: The main focus is on information modelling at the logical level. Conceptual data models and data interfaces can be derived from this at the physical level.

2.4 Object level

The model rules focus on the modelling of management objects, i.e. “Person” or “Address” etc. This means that the model rules, documentation requirements and general properties apply to the basic data model’s model entities. Rules are not set out at data set level, neither is metadata for the data set dealt with.

2.5 Demarcation

A number of technical specifications lie outside of the scope of this document – primarily because they concern development projects within the Basic Data Initiative that have not yet been fully clarified. The model rules therefore do not cover the data distributor’s service specification including data formats and protocols for access to basic data.

The model rules do not contain requirements for a specific approach to preparing data models. Information models submitted as part of the basic data model must comply with the model rules, but there is freedom of choice as regards the modelling process for the domains[3].

[3] The following recommendations based on publications from the DANTERMcentre (www.danterm.dk/) and the [OIO work model] can be provided for the process behind domain modelling:
Information models may be e.g. developed "top-down" from a high-level model such as a concept model, "bottom-up" from a physical data model or through systematic review of the content of the data register that shall distribute data.
The top-down approach involves the following steps:

1. Terminological concept modelling
2. Preparation of a terminological ontology containing information about concepts in the form of characteristic features and conceptual relations.
3. Conceptual data modelling
4. Preparation of a data model that reflects types of entities and their mutual relations, and that constitutes an abstract representation of data.
5. Logical data modelling
6. Preparation of a data model that specifies the organisation of data in a manner that reflects the logical structure in an IT system.
7. Physical data modelling
8. Preparation of a data model that reflects the physical structure in an IT system.

3 Architectural conditions

The basic data model has interfaces for other projects and planned components in the public IT architecture. These interfaces help to create a framework and objectives for how the model should be designed. The common IT architecture is not necessarily established or reliable, and similarly the model rules for the basic data model must have a certain degree of dynamism to be able to support integration with the surrounding architecture.

A number of the elements in the public IT architecture that have a direct impact on the model rules are reviewed below – primarily to create a reference framework for discussing the purpose of individual rules. Many of the elements are not fully developed and operational and thus do not constitute concrete architecture components that can set precise objectives for basic data modelling. That is why the following sections contain a number of assumptions that form the basis of the model rules' design. These assumptions are based on dialogues with the organisations responsible for implementing components, and are as faithful to the aims as possible. If these assumptions are proven not to reflect the future landscape, the model rules must be adjusted.

3.1 Existing standards

When designing the model rules a lot of emphasis was put on adapting existing international and national standards with a particular focus on INSPIRE, ISO and Sag og Dokument:

INSPIRE: A lot of basic data are covered by the EU's INSPIRE Directive [INSPIRE]. This document must therefore not include rules that prevent basic data from living up to INSPIRE's standards and guidelines. INSPIRE also has a well-founded modelling basis based on e.g. ISO standards and a well-tested method basis where EU member states have collaborated to develop data models for INSPIRE data. That is why there is an emphasis on reusing INSPIRE's standards and guidelines. Further information about INSPIRE's modelling basis can be found here [INSPIRE GCM].

ISO: The model rules emphasise the reuse of ISO standards for e.g. data types. In addition to the ISO standards constituting an internationally recognised reference framework, ISO has also worked with modelling the standards in UML, which is the selected modelling language for the basic data model. In this way, ISO is able to make a number of reusable elements available to the domain modellers. See rule 5.5 and rule 5.6.

Sag og Dokument: Comprises a number of public standards in the Sag og Dokument area, which was developed with a view to supporting digital working methods and exchanging information between organisations [Sag og Dokument]. The standards cover e.g. a specification of general properties in the Sag og Dokument area, see [S&D General Properties], which have been reused to the greatest extent possible in the rules for general properties in chapter 6. In addition to this we recommend the use of the standards Classification [Classification] and Organisation [Organisation]. See also section 3.3.

3.2 Data distributor

A data distributor is established under the auspices of the Basic Data Initiative to effectively and safely distribute data from the basic data registers, read more here [Data distributor]. Data sent via the data distributor are also the data modelled in the basic data model. Since the data distributor is not yet established, this means that model rules intended to support a specific service interface on the data distributor cannot be established. This is why this document must be based on assumptions about how the basic data model best supports data access on the data distributor.

Assumptions:

- It is assumed that at some point the data distributor will be included in a model-driven architecture where the specification of data is handled under the auspices of the data model rather than the documentation, which is separate from the data.
- It is assumed that the data distributor will be able to support the distribution of event notifications – that it will send data event notifications to a prospective event-driven architecture [EDA].

That is why the document contains rules that make it easier to convert the developed UML models for other types of data modelling. The first round is intended to make it possible to auto-generate XML Schema based on the UML model. Thus, it may be possible in the long term to develop interfaces where the data users compile extracts and data interfaces based on the model.

The auto-conversion of UML models to XML Schema is currently being performed under the auspices of INSPIRE.
It will also be possible to reinterpret the UML model for RDF [http://en.wikipedia.org/wiki/Resource_Description_Framework]. The RDF model can form a basis for a number of different representations of data and can also be used to distribute data in a Linked Data/Semantic Web context.

The document also contains rules to ensure that data necessary for creating event notifications are present in the basic data. Read more below.

3.3 The Joint Municipal Framework Architecture

Work is being done under the auspices of KL and KOMBIT to implement infrastructure components as the creation of the framework architecture's architecture building blocks that attend to general functions in connection with storing and distributing organisation structures and other types of reference data [Joint Municipal Framework Architecture]. These components and a notification distributor are available. The modelling and use of basic data can be made much more effective by using functionality from this architecture, which is why the model rules were prepared with due regard to the assumptions below about the building blocks and components.

3.3.1 Organisation component

The standards for the *Sag og Dokument* area describe a logical structure for how the structure of a given organisation can be distributed in a general format that allows external systems to refer to parts or wholes within the organisation [Organisation]. The operators referred to in the organisation may be both specific (person, IT system, function) or general (section, office, organisation). Certain municipalities have already implemented system support for the organisation component, and KOMBIT is preparing a supply of a generally usable component [Joint Municipal Framework Architecture – support systems].

Assumptions:

- It is assumed that the organisation component will be a permanent and clear part of the public IT architecture.
- It is assumed that the state will use a corresponding component to distribute organisation.
- It is assumed that data relating to operators in the organisation component have identities and bitemporal properties allowing them to be used distributed.

For these reasons the rules below recommend e.g. that information about the operator responsible for the data object's registration in the database and its validity is stored and distributed as a reference for an organisation distributed in an organisation component.

3.3.2 Classification component

Similarly, KOMBIT [Joint Municipal Framework Architecture – support systems] is planning for the joint municipal framework architecture to contain a classification component – also described in the Sag og Document standards [Classification]. The classification component must store and distribute structured data – taxonomies and graphs – that reflects the business knowledge with an internal structure and which can be used to contextualise the management objects. Examples include FORM [FORM] and KLE [KLE], which are structured descriptions of the authorities' business. In connection with basic data, the structured data can be used to contextualise the management objects and ensure consistency – e.g. specification of the business area responsible for updating data. Data stored in the classification component are allocated a bitemporality and audit trails, just as the framework architecture is intended for the distribution and updating of content. That is why the component is ideal for storing a large amount of value domains, also simple lists.

Assumptions:

- It is assumed that the classification component will be a permanent and clear part of the public IT architecture.
- It is assumed that the state will use a corresponding component to distribute structured data.
- It is assumed that individual classes in the classification component have an identity and bitemporal properties allowing them to be used distributed.

For these reasons the rules below recommend e.g. that a management object's business area, business process and business event (see below) are stored and distributed as a reference to structured data, distributed in a classification component. The management object's status can also be distributed using the classification component.

3.3.3 Notification distributor/Event-driven architecture

The aim of notification distribution (see [EDA]) is to link administrative processes across authorities. When an administrative business event results in a data event, the event often triggers a number of other administrative processes – usually for another authority. Relevant authorities must therefore be advised of the data event change in the data object.

Event-driven architecture makes it possible for a user system to subscribe to data events – in the form of a notification. The aim is for the recipient of the notification, e.g. an IT system or case officer, to react to the notification and initiate the resultant administrative processes. E.g. a change to a person's address may trigger a changed service.[4] This presupposes that data change notifications can be distributed in a meaningful manner where it is possible to set up filters (subscriptions) that can distribute notifications to the relevant recipients.

If the notification only contains information about the data event itself, the recipient will analyse the extent to which the event is relevant to their business. In the event of a change to a person's address, a case officer e.g. has to know whether the change is a result of the person actually having moved or a road being renamed. Much better quality is therefore achieved if the notification contains information about the *business event* (in reality) that triggered the change and the *business area* and *business process* that were involved in the change to the data. The notification can thereby be distributed to the relevant user and included in a relevant process in the recipient organisation. Automated notification distribution may therefore presuppose that this information is present in the correct form so that it can support the distribution of notifications.

[4] Effektiv Sagsbehandling og Kontrol [ESK] presupposes completely automatic ongoing case processing where a system-generated notification triggers an automatic recalculation of e.g. a service.

Assumptions:

- It is assumed that data events in basic data shall give rise to event notifications distributed by the data distributor.
- It is assumed that these notifications shall accrue to the Joint Municipal Framework Architecture/service platform.
- There are no assumptions as regards the infrastructure sending and receiving notifications, which is why the technical protocols and agreement basis for notification distribution are not discussed in more detail in this document.

This is why rule 6.4 presents a number of suggestions as to how the data model could support the systemisation and distribution of properties in the event of a data event (business event, area and process) that could support automated notification distribution.

4 Using the model rules

This chapter explains how the model rules are constructed and how they should be observed.

4.1 The rules are either requirements or recommendations

The model rules are specified as either requirements or recommendations:

- Rules specified with "**must**" are requirements that must be observed.
- Rules specified with "**should**" are recommendations that should be observed, but that are not strictly necessary.

See also Appendix 1, which provides an overview of all the rules with specification of whether they are requirements or recommendations.

The rules are not exhaustive – where a domain is bound by rules specified in other contexts, it may follow these as long as there is no conflict with the rules in this document.

4.2 The rules may be developed within the business domains

The model owner may choose to make the rules more stringent or develop properties as required. Similarly, the model owner may need to specify additional rules or properties applicable to the business domain(s) in question. E.g. the basic data covered by the INSPIRE Directive should contain properties that are standard for INSPIRE data, but not for other basic data.

4.3 Pattern for rules

The model rules are described using the following pattern:

Name	*Specify name of rule*
Rule	*Describe the rule clearly and precisely*
Rationale	*Describe the business value of following the rule*
Implication	*Describe the impact the rule has on the business and technical implementation*

The rules generally concern actual modelling – they comment on the model entities, their attributes and the model as it is.

The rationale for the rule is usually found in an argument for the data objects – modelling must ensure that a specific data content can be distributed.

After the rule there is sometimes a short, practical example demonstrating how to apply the rule.

5 General model rules

The general model rules concern the data model design. The aim of the rules is to ensure the level of uniformity in the domain models required for diagramming, which is necessary for establishing a collective basic data model.

5.1 Data models must be prepared as UML class diagrams

Rule
Data models for basic data must be described in UML (Unified Modelling Language) version 2.4.1 as UML class diagrams.

Rationale
A collective and consistent model presupposes the use of a common modelling language. UML is chosen because it is an internationally recognised modelling language which establishes a general understanding of the method of relating elements to each other.

Implication
UML class diagrams must be prepared for all basic data with classes, attributes, relations and cardinalities as well as associated documentation. There is an attribute completeness requirement so that all the information distributed as basic data can be found in the basic data model.
See [UML] for more information. General UML symbols and notations are not described in greater detail in this document.

5.2 The UML model must be organised in packages

Rule
The UML model must be organised in packages with one package for each business domain.

Rationale
A logical organisation of the model's elements in packages makes it easier to name and reference elements.

Implication
Each domain model is placed in an UML package. A package may have subpackages. Where relevant, elements are made public so that elements in other packages can refer to them. More details about this rule can be found in [INSPIRE GCM] section 9.6.3.

The basic data secretariat shall also ensure that UML packages with elements that are general properties (see chapter 6) and packages developed by ISO, e.g. standardised data types, can be referred to (see rule 5.5). See also the footnote to section 2.1

5.3 Model entities must be reused

Version:

1.1.0

Rule

Model entities must be reused across the basic data model.

Rationale

The Basic Data Initiative presupposes that model entities will be reused across basic data to ensure consistency and avoid redundant data maintenance.

Implications

Model owners need to model the model entities for their own business domains and try to relate to model entities in other business domain's UML packages. See also the footnote to section 2.1

Examples

If e.g. the person domain needs to model "Address", the modelling must refer/reuse («use») the correct model entity within the address domain.

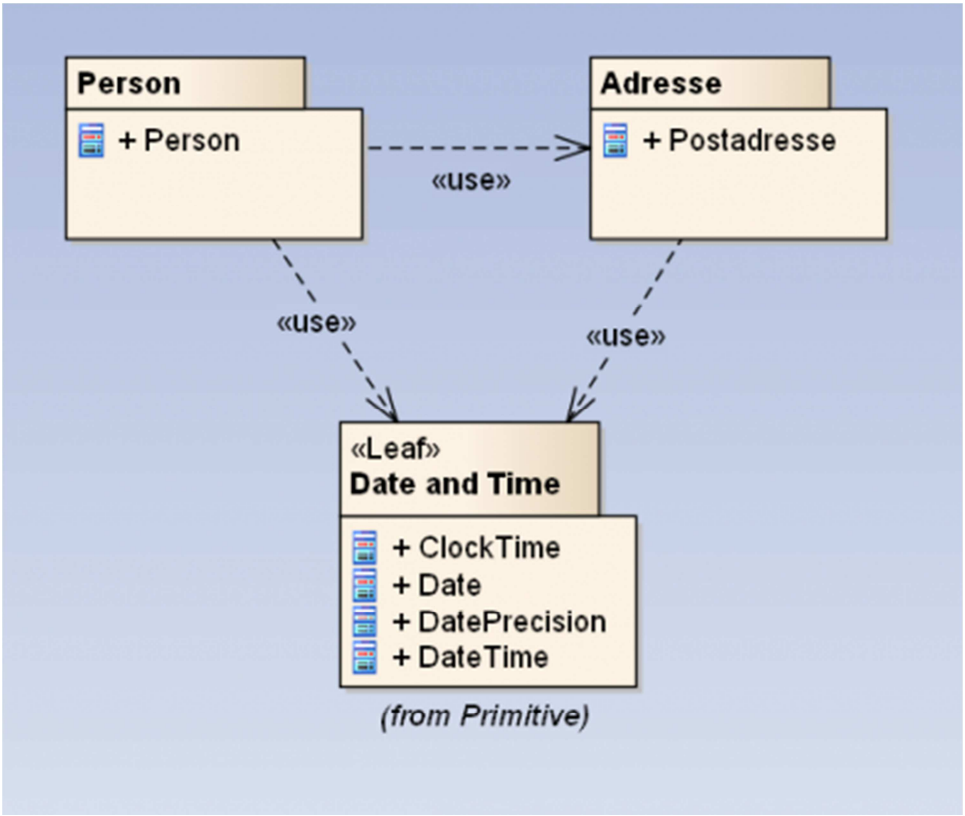


Figure 5 – Package "Person" uses («use») the packages "Address" and "Date and Time" (from ISO/TC 211 Harmonised Model) to use the classes PostalAddress and DateTime.

5.4 Attributes and relations must be fully modelled

Version:

1.1.0

Rule

Attributes and relations between UML elements must be satisfactorily modelled – i.e. with attributes modelled using multiplicity and data type, and relations modelled using multiplicity, navigabilityand roles.

Rationale

For the model to be clear and form a basis for the semantic understanding of data, it is necessary for the attributes and relations to be described in detail.

Implications

Attributes must be modelled using an explicit type – see rules 5.5 Standardised data types must be reused – and using multiplicity, i.e. number rules for the number of values that the attribute can include in each instance of the class. Relations (apart from generalisation/specialisation) between UML elements (associations, compositions, aggregations) must be modelled using name, reading order, navigability, designated role/relation end and multiplicity (multiplicity must not be specified for generalisation/specialisation). See also the footnote to section 2.3

Examples

The Figure shows how roles, reading order, navigability and multiplicity express a complex association between two data objects that would not otherwise be able to communicate under the auspices of the model.

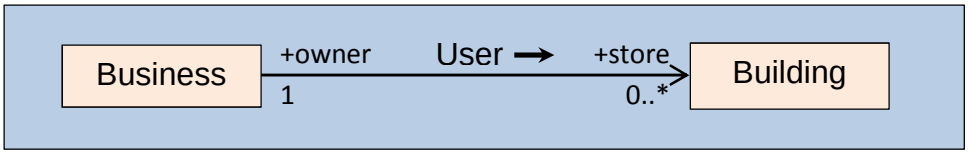


Figure 6 – Business with User role uses Building as Storage. A business can be everything from store-less to a store monopoly (multiple stores) – a Building can only have one owner.

5.5 Standardised data types must be reused

Rule

All attributes must either be allocated a standardised data type or a data type modelled as a UML element in the same or another package.

When using a standardised data type, reference must be made to ISO 19103, where a number of standardised data types are collected and modelled in UML. ISO 19107 is used for geography.

Rationale
ISO provides a recognised framework for standardised data types. The use of a standard for data types makes it easier to build interfaces that distribute basic data as easy-to-use services.

Implication
Standardised data types in the model must be retrieved from ISO/TC 211 Harmonised Model.
This is a collection of data types that are primarily used by geography-related modelling projects – e.g. INSPIRE. Nevertheless, ISO/TC 211 Harmonised Mode also contains generally usable data types (CharacterString, Integer, DateTime, etc.) modelled as UML[5].

If the domain does not find the types in ISO/TC 211 Harmonised Model available, it can construct its own type library to be published with the domain model.

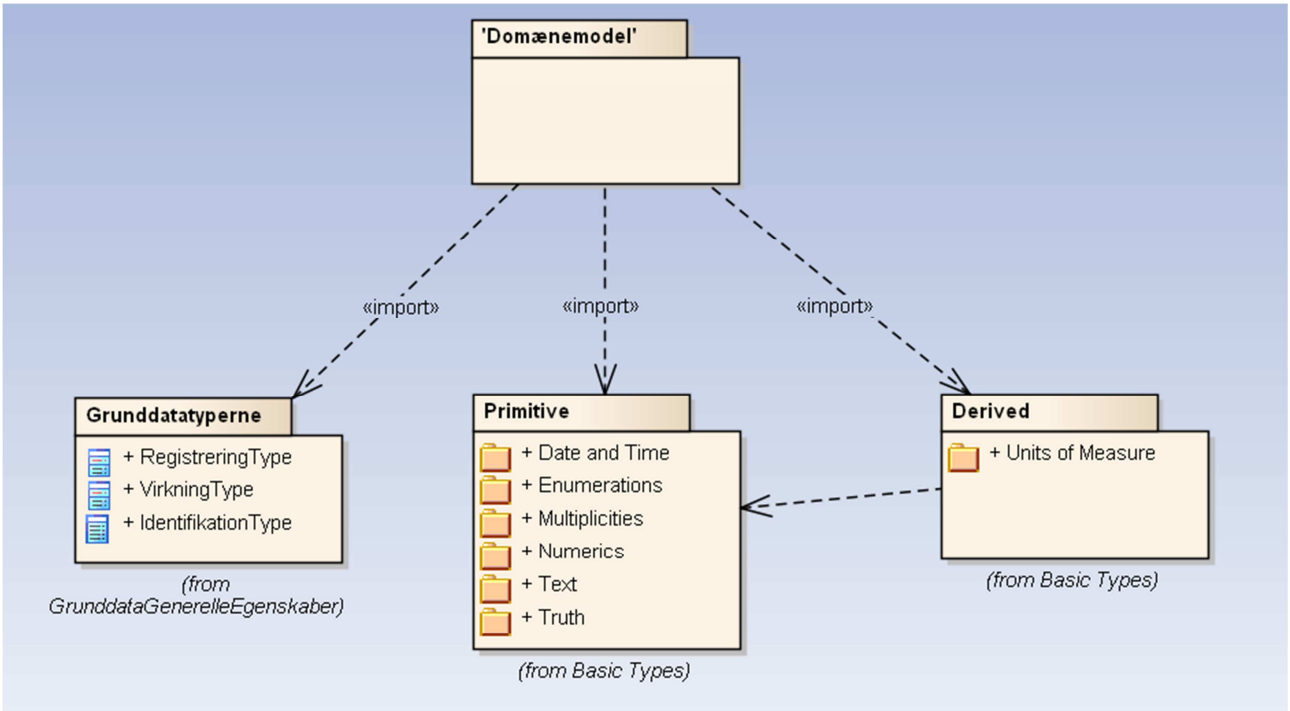


Figure 7 – The domain models reuse types from ISO TC211 Harmonised Model and types from the basic data type library – see rules 6.1, 6.3 and 6.4

5.6 UML stereotypes must be used

Version:

1.1.0

[5] ISO/TC 211 Harmonised Model is available at https://inspire-twg.jrc.it/svn/iso/, from where it can be imported into a modelling tool. The "Modelling tools" document that the basic data secretariat is expected to publish shall describe the use of ISO/TC 211 Harmonised Model in detail.

Rule

All UML elements are allocated a UML stereotype.

Rationale

In the long term it must be possible to interpret the model for other model types and e.g. data interfaces. UML cannot in itself designate the elements' role (model entity, data type, enumeration) in the model, which is why it is necessary to extend the model with these roles. UML stereotypes are extensions of the modelling language that make it possible to specify other properties and categorise the model elements. Using stereotypes you can designate specific classes that have specific roles in the data model, which also makes it possible for an external tool to convert the model to e.g. interfaces and ontologies. The stereotypes add roles to the elements and structure the tagged values that contain the documentation of the model entities (see Note about tagged values).

Implications

The following stereotypes are used in the basic data model*:

- **«DKObjekttype»**: All management objects must be modelled as UML classes with the stereotype «DKObjekttype».
- **«DKEnumeration», «DKKodeliste», «DKKlassifikation»**: Attributes with coded values must be modelled using UML elements with the stereotypes «DKEnumeration», «DKKodeliste» or «DKKlassifikation» as a set of values – see Noteabout coded values.
- **«DKDatatype»**: Attributes that are not coded values and not allocated a standardised data type from ISO (seerule 5.5) must be modelled using a data type in the same or another package with the stereotype «DKDatatype» as a set of values. See also the footnote to section 2.3
- **«DKEgenskab»**: Attributes and association ends in the basic data model have the stereotype «DKEgenskab».
- **«DKDomænemodel»**: "Root" package in a delivered domain model marked with «DKDomænemodel»

The basic data secretariat ensures that an UML profile containing the stereotypes is specified.

NOTE about tagged values

UML elements may contain extra attributes called tagged values. If you use tagged values on the stereotypes, they can be added to all classes with that stereotype at the same time. These tagged values may contain specific instructions for the software that interprets the model for e.g. XML Schema – see [INSPIRE GCM] section 9.6.3. There is not currently an overview of the instructions required. These can potentially be added later with minimal changes to the approved models. The stereotypes contain tagged values for the documentation of the model's elements. These documentation-tagged values (Definition, Note, Alternative name, Legislative basis, Example) are described in Appendix 3

NOTE about coded values

Coded values can be modelled in three ways:

«DKEnumeration», where the values are explicitly found as a list in the model. Typically used where the value domain is well understood and where changes are not expected quicker than the model's overall version cycle allows. E.g. days of the week: {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday}

«DKKodeliste», where the values are stored externally, but available on the Internet. See [INSPIRE GCM] sections 9.4.9, 9.5.2 and Appendix G. This allows for the dynamic adaption and extension of the code list as required, while also ensuring a degree of control of the history and provenance of the values.

«DKKlassifikation», where the values are stored in a taxonomy – e.g. in a classification component following [Classification]. The classification component may equip the values with metadata, update metadata and bitemporality. A classification component is included in a prospective KL/KOMBIT framework architecture. See section 3.3.2.

For certain domains there are rules on how external data is controlled and handled – e.g. the INSPIRE rules that apply to some geographic information that is also basic data. These rules can be observed while also observing the model rules.

The domain must ensure that the external data to which the domain data refers is available and of the required quality. The domain must also observe the rules that apply for archiving and that may require external data to be delivered to archive along with domain data.

* The model can be extended using domain-specific stereotypes just as basic data stereotypes can be extended in the domain, e.g. geographic information's DKFeaturetype, which extends DKObjekttype

5.7 Naming rules must be followed

Version:

1.1.0

Rule

Elements in the UML model must be named uniformly throughout the entire basic data model. Naming must be clear within the domain – in practice within the UML package.

Rationale

A uniform naming convention gives the data model a uniform expression and makes it easier to identify and differentiate the different classes of model elements from each other.

Implications

Classes and associations must be clearly named within their package.

Classes, attributes and associations are named in accordance with the following schema:

- **Elements representing management objects (with the stereotype «DKObjekttype») that are data types («DKDatatype»), classifications («DKKlassifikation»), enumerations («DKEnumeration») and code lists («DKKodeliste»)** are named using "UpperCamelCase" – i.e. using uppercase letters to start both the first word and all subsequent words and without using spaces in the name.
- **Attributes, associations and relation ends** are named using "lowerCamelCase" – i.e. using lowercase letters to start the first word and uppercase letters to start all subsequent words in the name and without using spaces in the name*.

*Where the names comprise acronyms you may deviate from this rule; e.g. "NUTS1value", "CPRnumber"

NOTE: With regard to the model's use and conversion in software that cannot handle the Danish characters Æ, æ, Ø, ø, and Å, å, these may be transliterated as "Ae", "ae", "Oe", "oe", "Aa" and "aa".

5.8 Language rules must be used

Rule
Danish must be used for naming elements included in general properties.

The model owner determines the language used for naming the domain's elements.

ISO standards follow their English designations (e.g. "Integer" and "codeList").

The data model is documented in Danish, see rule 5.9.

Rationale
As basic data forms the authorities' management basis, it is primarily described in Danish, which is the management language. However, some basic data may be subject to international obligations that require the use of other languages. E.g. some basic data are covered by the EU's INSPIRE Directive and must, as a result of this, reuse UML elements where the language is English. It is therefore up to the model owner for the domain to decide on the language for domain-specific elements.

Implication
It is up to the model owner to choose the language for the domain's model elements. Models may therefore exist in which there is a mix of Danish and other languages.

5.9 The data model must be documented

Version:

1.1.0

Rule

The data model must be documented by describing elements in the UML model. The descriptions are established and maintained together with the model described in Appendix 3.

Rationale

The documentation makes it possible for the model's users to understand the model's elements. When you both develop and use the model it is essential to communicate and understand the significant content of individual parts of the model. As the documentation is embedded in the data model, the automatic creation of a catalogue of classes, attributes and relations is possible. The documentation can also be included in the data interfaces if so desired.

Implications

The data model's classes, attributes and relations must be documented as described in Appendix 3.

NOTE: Some domains use large amounts of metadata that is intended for inclusion in user-oriented information about data (explanations for reports, help text on websites, etc.) It would not be appropriate to embed this metadata in the model itself, but it can be modelled and distributed like all other data and distributed through the general data architecture.

5.10 References to classifications, business models and organisation models should be used

Rule
As far as possible, references to external information should be references to published classifications, business models and organisation models.

Rationale
Most data objects must be related to classifications and other types of structures in order to give the object context and consistency. To reinforce reusability and searching, these references should point to existing, published and structured data sets such as FORM. It may also be relevant to point at specific organisational units – it may be advantageous to reuse these from published organisation models. See also section 3.3.

There is currently no infrastructure to support data structured in this way. It may therefore be necessary – possibly as a migration strategy – to make data consistent in simple ways, e.g. by reference to a list of organisational units. Such lists may later be imported into infrastructure components and the references adapted.

Implication
The area is under development, which means that an exhaustive implication cannot be established. Similarly, this rule cannot clearly specify concrete modelling or form a basis for the development of concrete infrastructure.

Recommendations:

When specifying:	Use:	
Business area:	**FORM**:	Generally recommended to be used where a public business area is specified. Specifically recommended to be used as value domain for the attribute 'business area' in rule 6.4.
Organisational unit:	**ORGANISATION**:	Generally recommended to be used for references to organisational units. Specifically recommended to be used as value domain for

		the attributes registration operator and validity operator in rule 6.3.
Structured data	**CLASSIFICATION**:	Generally recommended to be used for references that require a more complex structure than enumeration or codeList can support. Specifically recommended to be used as value domain for the attributes status in rule 6.2 and business process in rule 6.4.

References:
[FORM], [Organisation], [Classification]

6 Rules about general properties

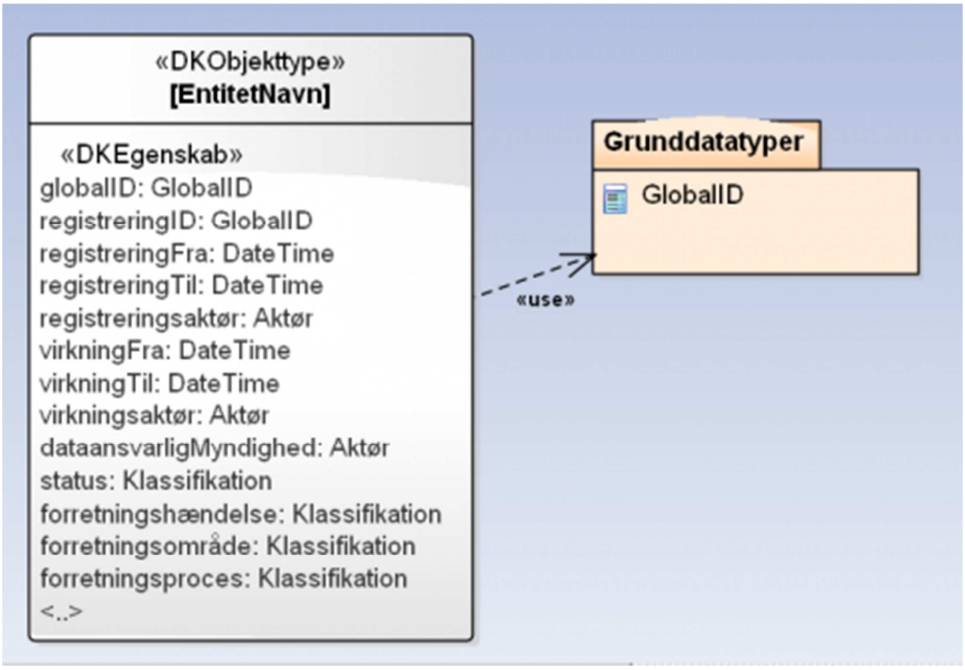
Version:

1.1.0

Rule

The general properties must support basic data being used in context in a distributed management environment. The properties must also ensure that the users experience increased data quality which includes data being distributed in a uniform manner so that properties used often have the same form across basic data. The rule must also facilitate a strong, intelligent, event-driven architecture. In this chapter the rules are primarily formulated at a business level and followed by specifications of general properties that all model entities must have.

The general properties give rise to a number of attributes and a data type included in the basic pattern for a model entity. These will be made available by the basic data secretariat.



A template for modelling a model entity containing the general properties that are given rules in the chapter. Model entities in the domain models will of course contain additional attributes.

The general properties and their attributes are explained in the following rules.

The rule's implication is specified for each general property, regardless of whether it is Mandatory or optional:

- **Mandatory property:** Must be present in the domain model and must be modelled as specified in this chapter.
- **Optional property**: Must only be present in the domain model if it makes sense within the domain. If the property is present, it must be modelled as specified in this chapter.

In some cases the attribute value does not need to be entered. Whether the attribute value needs to be left blank is specified for each attribute.

6.1 All model entities must be modelled using a persistent, unique identification

Version:

1.1.0

Rule

All entities must be modelled using a persistent, unique identification.

Rationale

All data objects, data object attributes and relations must be clearly identifiable across the collective basic data model and the many submodels used as inputs for the collective model.

The data object and unique identification often have one or more business keys, e.g. a land register or land register number. But the business keys cannot stand alone as the basic data model generally supports history, which means that the data object may have different business keys over time, just as the same business key may be included in several management objects at any given time.

That is why it is essential that the object's identification is consistent throughout the entirety of the data object's life cycle.

The use of HTTP-URI has shown its value for clear and constant identification through many years of practical use, e.g. in XML Schema.

An HTTP-URI also has the benefit of being able to be used as URL, Uniform Resource Locator. When the HTTP-URI is used as URL, and thus as a link, it is possible to receive more information about what the HTTP-URI identifies. The URI has become "de-referenceable" (http://en.wikipedia.org/wiki/Dereferenceable_Uniform_Resource_Identifier)

This gives the global ID two functions:

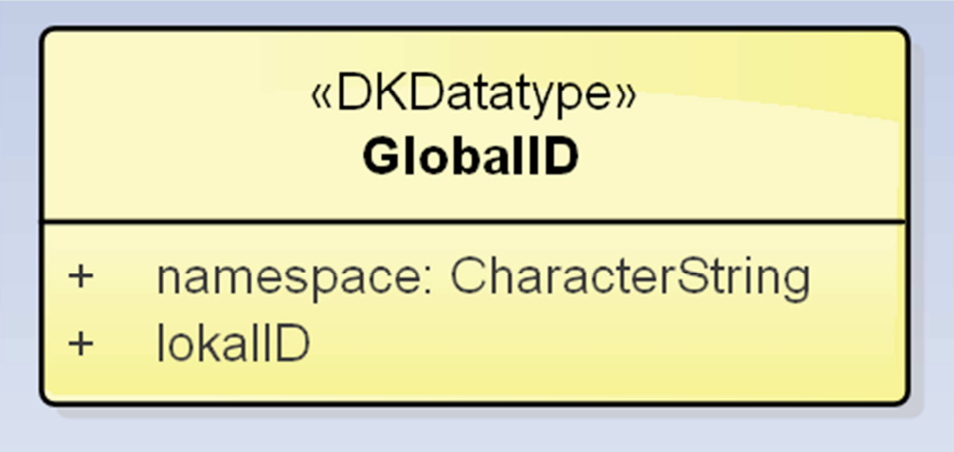
- Global and clear identification of an entity across the basic data model and across other models.
- Possibility of referring directly to the data object such that a given data object may potentially be addressed directly – also using a generic web application.

Implications

All entities must be modelled using the attribute 'globalID' with the type GlobalID – the data type has been made available by the model secretariat – see http://digitaliser.dk/resource/2742281

Significance	Unique identification of the object
Value	Object's unique ID
Data type	GlobalID
Requirements	Mandatory

Example:



Data type GlobalID

One instance of GlobalID constitutes one instance of the attribute namespace and one instance of the attribute lokalID. Both of these are specialisations of the standardised type CharacterString. This also means that namespace and lokalID must be able to be combined into a HTTP-URI as specified in IETF RFC 3986 (http scheme). This must be universally unique and will be de-referenceable.

An HTTP-URI may be put together as follows – based on IETF RFC 3986, section 3, Syntax Components:

Schema specification + ":// + Authority + "/" + Path + "#" + Object identifier

Content:

Component	Value
Schema specification	"http"
Authority	"data.gov.dk"

Path	Optional, but see below
Object identifier	Value, which must at least be unique within the domain

The HTTP-URI must as a whole be universally unique.

This may be achieved in a number of ways:

- Missing or generic Path, universally unique Object identifier
- Example: http://data.gov.dk#e1f9a650-21c7-11e3-8224-0800200c9a66
- Uniquely restrictive Path, locally unique Object identifier
- Example: http://data.gov.dk/lufthavnskoder/IATA#CPH
- Missing or generic Path, locally unique Object identifier
- Example: http://data.gov.dk/person#08f88d52-4add-4ec6-9a80-18caaf0f0e7b

It is the domain's responsibility to ensure that the collective, identified HTTP-URI is universally unique.

It is important that the collective identifier is reliable over time. That is why the elements Authority and Path are not considered reflective of an applicable administrative context for the data object that would possibly have to be changed if the data's ownership or content was changed. The elements only distribute semantic information about data, i.e. its logic structure and significance.

The components Schema specification, Authority and Path can constitute a Namespace modelled using the attribute namespace. The Object identifier is modelled using the attribute lokalID

Significance	Identification of a namespace within which the lokalID is unique
Value	A HTTP-URI without Object identifier (Schema specification + "://" + Authority + "/" + Path)

Data type	CharacterString
Requirements	Mandatory

Significance	Identification of object
Value	Object ID
Data type	CharacterString
Requirements	Mandatory

6.2 All model entities must be modelled using a status

Rule
All model entities must be modelled using a status that clearly specifies where a management object is in its life cycle.

Rationale
Management objects typically go through one life cycle. A life cycle for a building could be: “Proposal > Planning > Construction > In use > Demolition > History”. The business domain may set out rules for the statuses applicable to a given object and for how the management object goes through these.

These conditions must be located and distributed in an approved value domain[6] to which the model entity shall refer.

[6] A valid specification of statuses ideally presupposes work with processes and concepts in the domain.

The data object's status expresses the data object's relevance to the data user. Business-wise it is useful to distribute a data object's status explicitly instead of letting the data user analyse their way to the information. The use of status is included to ensure high data quality and potentially reduce development costs as less business and troubleshooting logic would be implemented.

Implication
All model entities must have the attribute 'status'

status	
Significance	Management object status
Value	A status
Data type	enumeration, code list, classification
Value domain	Domain-specific list, value must not be blank
Requirements	Mandatory

Status conditions must be defined for all management objects in the domain models. These status conditions are defined by the model owner and published in the common model. The conditions may be modelled as enumeration, code list or classification (see Note about coded values in section 5.6).

NOTE: Ideally statuses should be designated with the actual conditions, not the last reached milestone. E.g. "in use" rather than "taken into use" – the object will still have been taken into use.

6.3 All model entities must support bitemporality and operator specification

Version:

1.1.0

Rule

All model entities must be modelled using specification of registration, validity and operators.

Rationale

Bitemporality:

Across basic data there is a need to implement bitemporality in order to support a collective requirement for audit trails. It must in other words be possible to reconstruct data objects so that there is control of the object's composition or condition at any given time. Among other things, the purpose of this is to document the concrete historical decision basis in connection with e.g. case processing.

Bitemporality is modelled using bitemporal properties. The "bi" part comes from there being two time aspects – "valid time" and "registration time" – that are handled together.

Registration time:

The time from when the version is registered in the database until it is either replaced by a new version or deregistered.

Valid time:

The time when a given version of data corresponds to the conditions in reality that the version depicts.

Operators:

Information about the operators responsible for the data content, ensuring traceability in connection with audits and use of data. The operator may be one of a number of different types, e.g. an organisation, an IT system, a work function or a defined user.

Implications

A data object may comprise of a number (1–*) of versions (if an attribute value is changed, the data object is considered changed and thus to have different versions). All versions are considered parts of a "master data object" and have the same Identification.

All versions must be characterised using their registration time and valid time. These time aspects are modelled using the attributes registreringFra (registrationFrom), registreringTil (registrationTo), virkningFra (validFrom) and virkningTil (validTo).

The valid time must be interpreted such that the valid interval includes the start time but not the end time.

Individual data domains are responsible for deciding and documenting the extent to which there may be "holes" in the valid time – periods within which reality is not reflected in data.

Operators must be associated with each version of a data object, involving:

Reference to the operator that initiated the validity period

Reference to the operator that carried out registration

The value may be a reference to e.g. an organisation, a system or a case officer, see section 3.3.1 and rule 5.10.

A full temporal reconstruction of data objects requires that all changes to the data object be hidden with the necessary time stamps. That is why this rule contains a requirement for data being stored in registers implemented in a way that makes this possible. Such an implementation partly comprises a physical data model that adds a new row to the data table for each new version of data in such a way that all changes can be found again and partly business rules for how time stamps are linked. A document explaining how this can be done in a RDB/SQL environment can be downloaded here

Attributes:

Each model entity using the stereotype DKObjekttype – corresponding to a business object – is modelled using the following attributes:

Registration:

SIGNIFICANCE	Identification of a specific registration (version) of the data object

VALUE	Unique ID
DATA TYPE	GlobalID
REQUIREMENTS	Optional

SIGNIFICANCE	Time when registration took place
VALUE	Time
DATA TYPE	dateTime (ISO 8601), value must not be blank
REQUIREMENTS	Mandatory

SIGNIFICANCE	Time when a new registration took place for the data object and thus when this version was no longer the most recent.

VALUE	Time
DATA TYPE	dateTime (ISO 8601), value may be blank
REQUIREMENTS	Mandatory

SIGNIFICANCE	The operator who carried out the registration
VALUE	Name of an operator or reference to an organisational model (see rule 5.10)
DATA TYPE	Domain–specific operator, value must not be blank
REQUIREMENTS	Mandatory

Validity:

SIGNIFICANCE	Time from when the management object became valid

VALUE	Time – validity period includes this time
DATA TYPE	dateTime (ISO 8601), value must not be blank
REQUIREMENTS	Mandatory

SIGNIFICANCE	Time when the management object becomes invalid
VALUE	Time – the validity period ceases immediately before this time
DATA TYPE	dateTime (ISO 8601), value may be blank
REQUIREMENTS	Mandatory

SIGNIFICANCE	The operator that made the management object valid

VALUE	The name of an operator or reference to an organisational model (see rule 5.10)
DATA TYPE	Domain–specific operator, value must not be blank
REQUIREMENTS	Mandatory

6.4 All model entities should support notification distribution

Version:

1.1.0

Rule

The model entities in basic data should be modelled so that the data object contains information that may improve the quality of event notifications sent in connection with updating the data object. This information covers the business-related context within which the data object is updated and the underlying business-related reasons for the update.

Rationale

Automated notification distribution is described in chapter 3.3.3. At present there is no comprehensive overview of how notification distribution should take place, which is why this rule is worded with the aim of describing the framework for the data necessary for notification distribution. Further work with public notification distribution might very well result in an adjustment of this rule.

Automated notification distribution presupposes that for each change to data, the business context that gave rise to the change is registered.

The business context is described using three parameters:

- business event (forretningshændelse) that describes the event in reality (see section 1.3.2) which triggered the change to the data
- business area (forretningsområde) – the part of public business that handles the event and thus brings about the change to the data
- business process (forretningsproces) – the manual or IT-supported process in which the business area handles the event

Event, area and process constitute value domains that are typically modelled by the domain

- Business events are typically (such as status – see rule 6.2) specific to each management object – a data structure must be constructed that reflects the business' knowledge of the events that might affect a management object
- The business area may be specified on the basis of FORM
- Business processes require mapping of the domain's processes

The three value domains may be modelled to be as complex as desired – as simple lists, embedded in the model or as reference to external data either in the form of code lists or classification components – see NOTE about coded values in section 5.6 and section 3.3.2.

The model owner must decide the level on which modelling most corresponds to the business requirements.

Implications

Each model entity using the stereotype DKObjekttype – corresponding to a business object – may be modelled using the following attributes:

SIGNIFICANCE	The business area that updated the data object
VALUE	Specification of a business area. Domain-specific list, value may be blank
DATA TYPE	enumeration, codeList, klassifikation

REQUIREMENTS	Optional

SIGNIFICANCE	The business process that updated the data object
VALUE	Specification of a business process. Domain-specific list, value may be blank
DATA TYPE	enumeration, codeList, klassifikation
REQUIREMENTS	Optional

SIGNIFICANCE	The business event that initiated the update
VALUE	Specification of a business event. Domain-specific list, value may be blank
DATA TYPE	enumeration, codeList, klassifikation

REQUIREMENTS	

7 References

Reference	Title	Link
[Architecture Guide]	OIO Architecture Guide	Link
[Architecture Guide – management object]	OIO Architecture Guide, B1 Business objects	Link
[Architecture Guide – information model]	OIO Architecture Guide, C1 Information objects in logical data model	Link
[Basic Data Initiative]	Basic Data Initiative	Link
[Classification]	Specification of service interface for classification – OIO-approved [v. 1.1]	Link
[Data distributor]	Description of the public data distributor on the Danish Authority for Digitisation's website	Link
[EDA]	Event-driven Architecture	Link
[ESK]	Effektiv Sagsbehandling og Kontrol	Link
[FORM]	FORM online	Link
[Joint Municipal Framework Architecture]	KL's website on the Joint Municipal Framework Architecture	Link
[Joint Municipal Framework Architecture – support systems]	KOMBIT's website on the provision of support systems in the framework architecture	Link
[Den Fællesoffentlige Topontologi]	Den Fællesoffentlige Topontologi – Link to Public Administration association FORVIR's website	Link
[INSPIRE]	The INSPIRE Directive	Link
[INSPIRE GCM]	INSPIRE Generic Conceptual Model	Link
[INSPIRE documentation]	Connecting to the public INSPIRE UML repository using Enterprise Architect	Link
[KLE]	KLE online [Municipality Association Topics]	Link
[Conceptual data model version 0.8]	Conceptual data model for basic data version 0.8	Link
[Notification distributor]	Notification distributor – scope document for joint municipal implementation	Link
[OIO work model]	Publications concerning the OIO work model for	Link

	data standardisation in the sectors	
[Organisation]	Specification of service interface for organisation – OIO-approved [v. 1.1]	Link
[Sag og Dokument]	Sag og Dokument standards	Link
[S&D General Properties]	General properties for services in the Sag og Dokument area – OIO-approved [v. 1.1]	Link
[UML]	Unified Modelling Language	Link

Appendix 3: Documentation of the data model

The data model must be documented using elements in the UML model. The model's classes and their attributes describe business objects and their properties as well as the roles that the objects have in relation to each other described as roles/relation ends at each end of an association. These business objects, properties and roles must be described in compliance with the concept modelling in the domain.

Documentation in UML with tagged values

Classes, attributes and roles/relation ends are documented using tagged values (see Note about tagged values) as specified in the table below. Individual tags in the documentation correspond to the greatest extent possible to the simple knowledge organisation system [SKOS]. This structured way of preparing documentation allows for the flexible generation of text documentation based on the model, just as great precision can be used to convert the documentation for interfaces and derived models.

Tag definition	SKOS equivalent	Requirements	Content
definition	skos:definition	Required	Definition of the object/property/role. Short text that clearly describes the object/property/role – may also contain a section with a longer description, e.g. aim, references and source.
note	skos:note	Optional	Detailed description of the object/property/role

alternativtNavn	skos:altLabel	Optional	Other names that the object/property/role may have
lovgrundlag	--	Optional	Specification of the legislative basis that sanctions the collection of data for the object/property/role
eksempel	skos:example	Optional	Examples of the use of the object/property/role

The values are specified in clean text

The tagged values mentioned in the schema are included in the basic data model's stereotypes as defined in rule 5.6 UML stereotypes must be used.

The tool support in the MDG and base project distributed by the model secretariat (http://digitaliser.dk/resource/2742281) is the relevant tagged values added to the stereotypes in a separate group – Basic data:documentation

[Figure showing Class properties with tagged values] – will be added later

[Examples of completed tagged values – either graphically in EA or as tables for class, attribute and relation end]