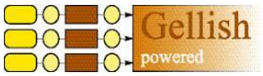


Gellish Modeling Method

Part 2
Creation and use of Domain Taxonomies

dr. ir. Andries van Renssen

December 2008

Table of content

1. Introduction

Interoperability of systems as well as integration of data from different sources requires the use of a common language. Gellish English is designed to act as such a common language. The Gellish Dictionary/Taxonomy with its various domains forms a general technical dictionary with definitions of concepts as are also defined in ordinary dictionaries and standards, extended with further specialized subtypes of those concepts.

However, it might be that the Gellish English Dictionary does not contain all the concept definitions that are required in a particular application domain or it may use different names, codes and abbreviations than are required in a particular context. For that reason the Gellish Modeling Method enables to extend the Gellish Dictionary/Taxonomy with definitions of additional concepts or with complete Domain Dictionaries/Taxonomies on various levels of detail, as is illustrated in Figure 1.

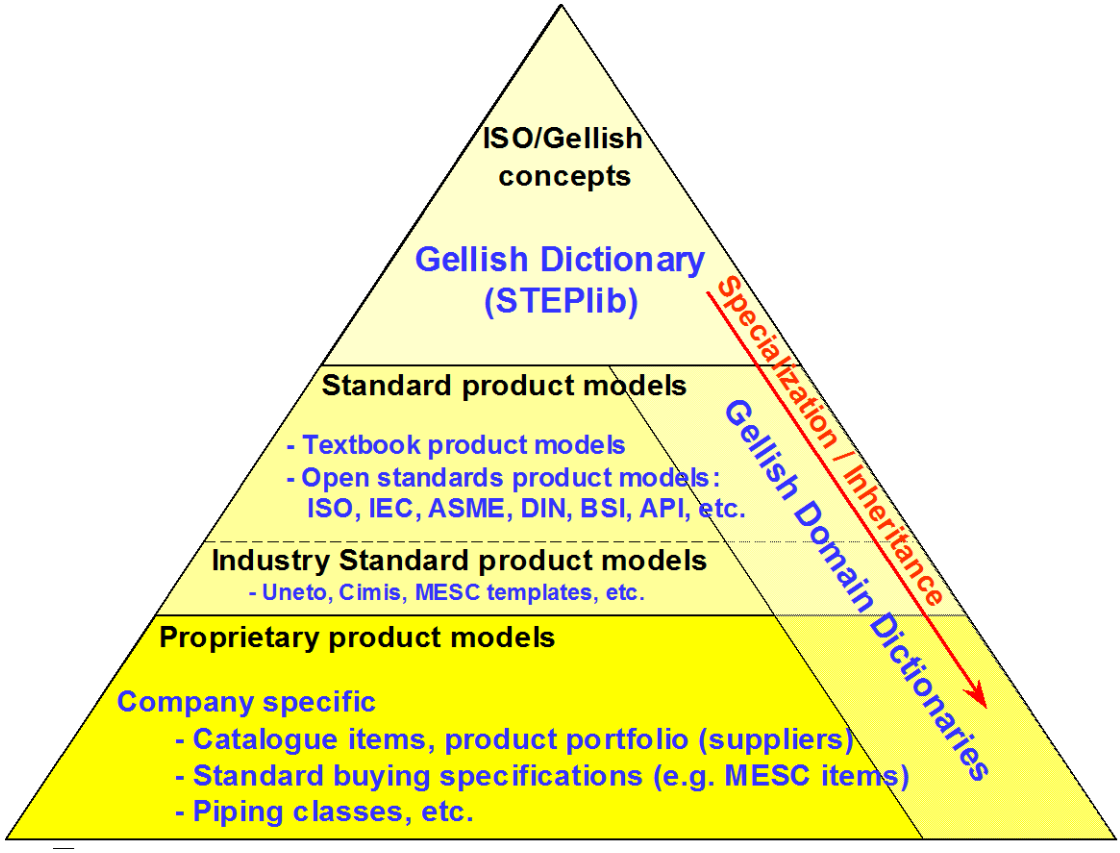


Figure 1, Domain Dictionaries as extensions of General Dictionary

Examples of reasons for creation of a Domain Dictionary/Taxonomy are:

- To create a dedicated electronic Gellish Dictionary/Taxonomy for a particular application domain. For example for civil engineering.
- To extent the Gellish English Dictionary/Taxonomy with definitions of additional concepts. For example as required for the mapping of equipment types and property types in a system.
- To create a Gellish Dictionary in another language than English.
- To map the names and codes used in a particular context to the names of the concepts in the Gellish Dictionary/Taxonomy.

This part of the Gellish Modeling Method is about the creation of electronic Smart Domain Dictionaries (Taxonomies) in Gellish. A smart dictionary is more powerful than an ordinary dictionary

because of the additional knowledge that it contains and because of the extended usage capabilities as will be explained in more detail below.

This document is intended to provide guidance on how to define such a Smart Domain Dictionary or Dictionary extensions, and how to define mappings between a application specific terminology and a Gellish Dictionary. The document also provides guidance on the use of Gellish Dictionaries.

1.1 The Gellish Modeling Method

A Gellish Dictionary is typically a component of an architecture for application systems that use the Gellish Dictionary and that also express information and/or knowledge in Gellish. This paragraph describes how a Gellish Dictionary fits in the overall Gellish Modeling Method Architecture.

The Gellish Modeling Method consists of a number of parts, as is described in the overview document Gellish Modeling Method – part 1, Architecture. The position of a Gellish Dictionary in the overall Architecture is illustrated in Figure 2.

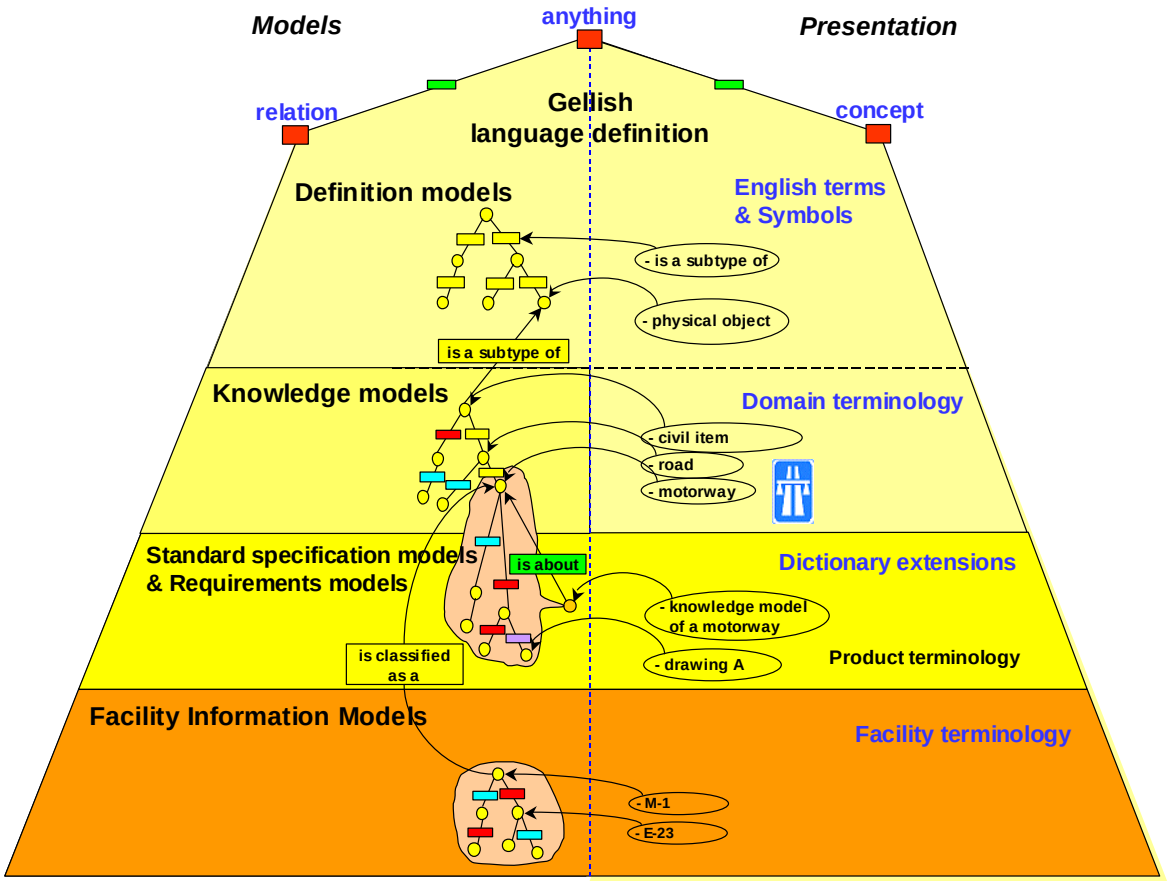


Figure 2, Definition Models integrated in the Gellish Modeling Method Architecture

The top plus the whole right hand side of Figure 2 illustrates the dictionary domain of the Architecture. The top part forms the standard Gellish English Dictionary, which includes the Gellish language definition. That language definition is composed of a definition of the grammar (with formal definitions of relation types) and a number of definition models that define the basic concept as far as needed to define the relation types. Those relation types as well as those concepts are arranged in the dictionary in a subtype-supertype hierarchy. The yellow rectangles between the concepts illustrate those specialization relations (also called subtype-supertype relations). Examples of defined basic concepts are: physical object, aspect, activity, person, etc.

Below that top part, on the right hand side, the domain terminology illustrates the part of the standard Gellish English Dictionary that defines concepts that belong to a general technical dictionary.

Examples in that part of the Gellish English Dictionary are: road, ship, pump, nut and bolt and also a large variety of other concepts such as: maintenance, risk, diameter, green and mm.

That same part also illustrates that specialized (public or proprietary) Gellish domain dictionaries or Gellish dictionaries in other languages can be developed that are complementary to the standard Gellish English Dictionary.

The standard Gellish English Dictionary as well as domain dictionaries can be extended with dictionary extensions for particular applications, as is illustrated in the third layer of the Architecture. The names of individual things that are illustrated at the bottom layer on the right hand side of Figure 2 can also be regarded as extensions of a Gellish dictionary.

The various parts of the Gellish Modeling Method in this Architecture provide guidelines for the creation of different elements of standardized system independent databases. Those parts are:

1. Domain Dictionaries, and Gellish Dictionary extensions. This part is concerned with smart definitions of concepts that are computer interpretable and define a common language for application in the other parts.
2. Knowledge Models, which specify generally valid knowledge about kinds of things. Knowledge models in Gellish contain expressions of knowledge that consist of relations between concepts that are defined in a Gellish Dictionary.
3. Requirements and Standard Specifications. Both consist of expressions of facts that are required to be the case in a particular context or when a particular standard is applicable. These kinds of facts are also expressed in Gellish as relations between concepts that are defined in a Gellish Dictionary. These specifications include:

   A) Requirements about kinds of things, which are applicable in a particular context for things of particular kinds.

   B) Standard Specifications, which specify the dimensions and other properties of standardized products as described in national or international standards or the manufacturer's models that are referenced in company product catalogues.
4. Facility Information Models that contain information about individual facilities and their components. Each individual facility or component, and each aspect of such a component, including an activity or process in which it is involved is defined by a classification relation that relates the individual thing with a concept that is defined in a Gellish Dictionary.

The first three parts of the Architecture are concerned with models with definitions, knowledge and requirements about kinds of things. The last part, about Facility Information Models, is about models of individual things.

The part of the Architecture that is described in this document (part 2) is concerned with the creation of *Domain Dictionaries and Gellish Dictionary extensions and their use*.

1.2 What is an electronic Smart Dictionary?

An electronic Smart Dictionary has the following characteristics:

- It contains definitions per *concept*, whereas ordinary dictionaries usually provide various definitions that are associated with a *term*, where it is often unclear whether those definitions are alternative definitions of the same concept or whether they are definitions of different concepts. Thus a smart dictionary explicitly distinguishes homonyms (the same term for different concepts) and explicitly specifies which terms are used as synonyms. To enable this, a Gellish Dictionary uses unique identifiers (UID's) to denote concepts, which UID's are distinguished from terms, because terms may refer to different concepts in different contexts.

- It is completely arranged as a taxonomy, which is a subtype-supertype hierarchy of concepts. This means that each concept is defined as an explicit subtype of one or more supertype

concepts by specialization relations (*A* is a specialization of *B* and *B* is a specialization of *C*, etc.).

- It includes also concepts with names that consists of multiple terms, whereas ordinary dictionaries usually limit themselves to single term concepts. For example, the concept: 'line shaft centrifugal pump' will be found in a smart dictionary, but an ordinary dictionary will usually only define line, shaft, centrifugal and pump, but the combination of such terms is usually insufficient to define the concept with the multiple term name.

- It defines relation types (as a special kind of concepts). These relation types enable to make computer interpretable expressions that express facts, including the expression of knowledge and information. This means that a smart English dictionary defines a subset of the English language. Those relation types have names and synonyms that consist of standardised phrases. For example: the phrase 'is a part of' can be used to express the fact that *A* is a part of *B*, whereas the phrase 'can have as part a' can be used to express the knowledge that a whole of a particular kind can have as a part a component of a particular kind. For example, the fact that a pump can have a bearing is expressed in a structured subset of English as:
  pump <can have as part a> bearing.

  - Note: All three elements in this expression are names of standard concepts that are typically defined in a smart English dictionary.

- Concepts are related to other concepts in various ways. Those relations are explicit and are of standardised relation types. Those relations express additional knowledge about the concepts that are by definition true. This makes a smart dictionary an ontology. An ontology can extent these relations beyond the dictionary facts to encyclopedic knowledge about the concept. Both kinds of knowledge about the concepts (dictionary definitions and encyclopedic knowledge) can be used by computers, because it is explicitly modelled. Furthermore, such knowledge about a concept is inherited by all the subtypes of the concepts in the subtype hierarchy (taxonomy).

- It uses one language independent unique identifier (a natural number) to represent each concept. This enables that facts that are expressed in one language can be automatically presented by a computer in any other language for which a smart dictionary is available.

- It can be extended by private and proprietary concepts and terms. For example, company specific codes and proprietary knowledge. Some instructions are given in 'Proper definition of a concept'.

- It is computer interpretable and system independent.

1.3 Objective and Scope of this document

The *objective* of this part of the Gellish Modeling Method is:

- To provide guidance on how to create a electronic smart dictionary as a common language for a language community (an application domain) that can be used as a basis for storage and exchange of knowledge, requirements and information about objects, aspects, processes and activities in that application domain in multiple systems.

- To provide guidance on how to define computer interpretable definitions of concepts, together with their names, synonyms, abbreviations and possibly symbols and references as an extension of the general Gellish language definition.

- To clarify how to specify definitions in one language and to display them in another language.

- To provide guidance on how a Gellish Smart Dictionary can be used.

The *scope* of this part is the specification of Smart Dictionaries. The scope of a smart dictionary consists of Definition Models that include:

1. Smart definitions, which are computer interpretable expressions of facts that are always and by definition true for the kind of things that are defined.
2. Names, codes, abbreviations, synonym names, symbols, etc. to denote the concepts.
3. A unique identifier (UID) for each concept to represent it.
4. References to documents that are used to define the concepts.

Out of scope of this part are:

- The specification of definitions, information and knowledge about individual things.
- The specification of knowledge that is not by definition true for all members of the defined kind.

1.4 Implementation of Gellish Dictionaries

Gellish Dictionaries that are created according to the Gellish Modeling Method can be documented in a system independent way in a Gellish Database that consists of one or more standard tables (see ref 1). This has as consequence that the dictionary definitions and identifiers can be imported in any system that is able to read Gellish Database tables. That is the reason why this document does not discuss any particular software system, but only deals with their data and document content.

2. What is a Definition Model?

Ordinary dictionaries usually provide for each term one or more definitions. The user of the dictionary is then left with the task to determine whether those definitions are different definitions for the same concept or that the definitions describe multiple concepts. In other words ordinary dictionaries are usually not explicit in the distinction between synonyms and homonyms. Furthermore, the definitions in ordinary dictionaries are sentences in which other terms are used, but the dictionary does not provide explicit relations to the definitions of those other terms.

In a Smart Gellish Dictionary a definition is created in the form of a computer interpretable Definition Model, which means that the components of the model are explicitly related to each other by relations in a data structure.

Each Definition Model defines a concepts and not a term, whereas each concept is represented by a unique identifier (UID) in the form of a whole number. Each concept may be denoted by various terms, such as names, codes, abbreviations, etc. in different languages, but those terms are all distinguished from the concept itself and they all refer to the same concept UID. This means that a Smart Gellish Dictionary makes an explicit distinction between homonyms (the same term for different concepts) and allows many alias terms for the same concept.

A Definition Model may contain also a sentence with a textual definition, but in addition to that, a Definition Model contains explicit relations between the defined concept and other concepts that are used to make the definition. In that way a Definition Model consist of a collection of facts, which forms a network of related concepts. Each relation with another concept in a model is classified by a standard relation type, which shall be selected from the Gellish Dictionary (the standard relation types are also defined in the Smart Gellish Dictionary).

3. Definition Model Creation Procedure

The process to create an electronic Definition Model for a concept typically consists of the following steps:

1. Allocate a unique identifier to the concept.

2. Specify a preferred name of the concept
3. Specify a direct supertype of the concept.
4. Provide a textual definition of the concept.
5. Specify a discriminating aspect of a physical object.
6. Provide an intended function of the concept.
7. Specify defining parts of the concept.
8. Specify synonyms, codes, abbreviated terms and translations for the same concept.
9. Specify pictures and drawings about the concept.

Such a specification of a concept can be specified by filling-in a template, from which a Gellish Database table can be generated. Such a template may look like the following example:

Definition of a new concept	
UID:	520,243
Language:	English
Language community:	engineering
Preferred name:	vessel
Synonym, abbreviated name or code:	V
…	
Supertype concept UID:	130,108
Supertype concept name:	container
Textual definition:	is a container that is intended
Discriminating function UID:	?
Discriminating function name:	storage of fluid
Discriminating aspect UID:	
Discriminating aspect name:	
Discriminating aspect value UID:	
Discriminating aspect value name/number:	
Discriminating aspect value Unit of Measure:	
…	
Obligatory part UID:	10161
Obligatory part name:	inlet / outlet
…	

Table 1, Template for the definition of a new concept

3.1 Step 1, Allocate a unique identifier to the concept

Each new concept or fact shall be represented in the Gellish language by a unique identifier (UID). Such a UID is basically an arbitrary whole number and is meaningless.

To enable the integration of data from different sources it is essential to ensure that UID's of concepts and facts in the Gellish Dictionary and in various Domain Dictionaries and UID's of facilities, knowledge and requirements models are globally unique. Therefore, Gellish Dictionary manager provides a service to reserve ranges of UID's for your application domains.

> Note that translation of existing concept names and expressions of facts do not need new UID's as the concepts as well as the facts are language independent and thus can be used for the same concept or fact in various languages.

The UID's for new concepts in a Domain Dictionary should be selected from a range that is reserved for the new domain dictionary concepts and facts. A range of UID's can be reserved via the

SourceForge website through a request via the FORUM HELP see: http://sourceforge.net/forum/forum.php?forum_id=89287.

The UID's 1 – 100 are reserved for the specification of unknown variables in queries.

The UID's 101 – 999 are reserved for general temporal use and testing.

The UID's 1.000 - 14.999.999 are reserved for the Gellish Dictionary.

The UID's 15.000.000 - 30.000.000 are reserved for the identification of dates in Gellish.

It is recommended to propose new concepts as extensions of the public domain Gellish Dictionary via the proposals mailing list on SourceForge:

gellish-proposals@lists.sourceforge.net

In that case the Gellish language definition manager will allocate UID's for the new concepts and facts within the above-mentioned range and the concept will become available for general use in the public domain Gellish English Dictionary.

In order to avoid overlap between the objects in a Domain Dictionary and the Gellish Dictionary it is important to ensure that the allocation of UID's is a managed process.

3.2 Step 2, Specify the preferred name of the concept

The specification of a new kind of thing that is represented by a UID requires at least one term (character string) that has the role of the name of the concept. It is required to specify in which language that name is expressed and also from which language community (or discipline) within that language the name originates. This is also the community in which the definition should be discussed and agreed. Both, the language and the language community, are the contexts that can be used later to judge which concept is meant in case of homonyms.

For example, the next new concept will have UID 520243. The concept will be called in English <vessel> and the language community where the name and definition originates is <static equipment engineering>, with its own UID 193264 which is selected from the Dictionary. Note that this language and language community are sufficient to distinguish this concept from another concept that is also called vessel in English, but in the discipline of transportation and which is a kind of ship.

In Gellish this is expressed as follows:

Language	UID of language community (discipline)	Name of anguage community (discipline)	UID of left hand object	Name of left hand object
English	193,264	static equipment engineering	520,243	vessel

Table 2, Allocation of a name to a concept

Note that the concept in the table is called the left hand object, because there will also be a right hand object as will be explained below.

3.3 Step 3, Specify the direct supertype of the concept

It shall be specified in the Definition Model of each concept what is its direct supertype concept.

This specification requires the expression of a fact about a subtype-supertype relation that uses the relation type (1146) <is a specialization of>. That fact is represented by unique fact id (the UID of the fact).

The specification of the name of the concept (as described in the previous paragraph) can be combined with this specialization relation on one line in a Gellish Database table as follows:

Language	Language community (discipline)	UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object	Textual definition
English	static equipment engineering	520,243	vessel	1,130,209	1,446	is a specialization of	130,108	container	is a container that is intended for storage of fluids.

Table 3, Specification of a fact about a subtype-supertype relation

Note 1: In this table, the UID of the right hand object indicates the supertype concept, whereas the name (or one of the names) of that concept is added in the database table for human readability.

Note 2: The character string (name) that is used in the expression of the fact that specifies the subtype-supertype relation will be the preferred name of the concept in the indicated language.

The fact that the subtype-supertype relation will be specified for each concept means that all the concepts together will be arranged in one subtype-supertype hierarchy, which is also called a taxonomy. The relation between the defined concept and its supertype concept forms a branch in this 'tree' of concepts.

One concept may have more than one supertype concept and a supertype usually will have more than one subtype concept. This means that the total hierarchy will not have a true tree-structure, but it will be a hierarchical network structure. The top structure of this hierarchy is defined in the Gellish Dictionary. The concept at the absolute top of the hierarchy is called 'anything'.

This means that each concept in a Domain Dictionary should be a direct or indirect subtype of the total hierarchy. It also means that all concepts together will form one integrated pyramid as is illustrated in Figure 3.

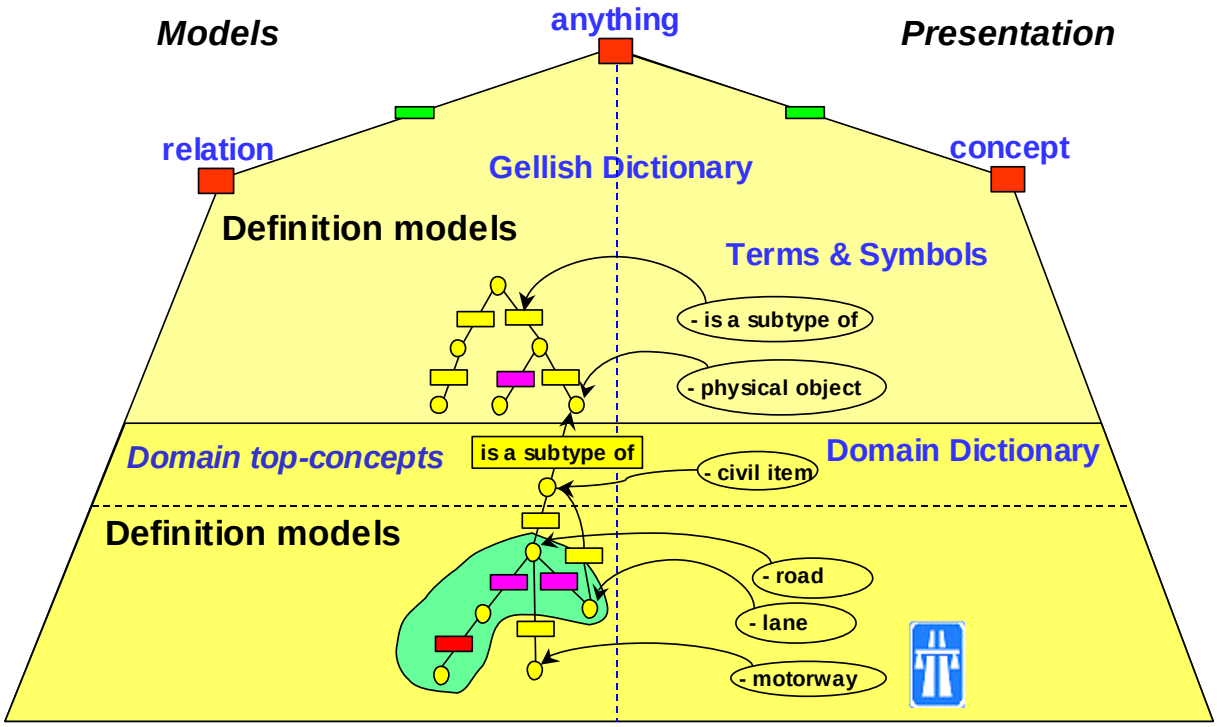


Figure 3, Domain Dictionary as an extension of a Gellish Dictionary

Figure 3 also illustrates that the construction of a Domain Dictionary might be aided by the pre-selection of a number of concepts that together form the top-concepts of the Domain Dictionary. This will simplify the search for proper supertype concepts, because the pre-selection will reduce the search to a limited number of concepts that are already in the domain of the Domain Dictionary. This work method requires that during the preparation for the construction of the Domain Dictionary those

selected domain top-concepts shall be defined as subtypes of higher level concepts in the Gellish Dictionary. It means that the Domain Dictionary becomes a truncated pyramid under the Gellish Dictionary concepts.

Note that a defined concept can have a supertype concept that is either a concept from the list of domain top-concepts or it is a not yet defined supertype concept. Such a not yet defined supertype concept is itself a new concept that shall be defined in the same way as the defined concept.

During the creation of the Domain Dictionary the hierarchy will be subject to changes that may influence other parts of the hierarchy. A regular check on the consistency of the whole hierarchy is therefore important to achieve a high quality taxonomy. The use of tools to regenerate the hierarchy is therefore recommended.

The following rule may be helpful to judge whether a concept is a subtype of another concept:

- if the potential subtype concept has the same function and is defined by the same aspects as the potential supertype concept, then it might well be a true subtype concept.

A subtype concept inherits all aspects that the supertype concept has, so that the definition of the supertype concept does not need to be repeated for its subtype concepts. The only things that need to be defined for the subtype concept is its differences from the supertype concept and the differences from its neighbour concepts that are also subtype of the same supertype concept.

3.4 Step 4, Provide a textual definition

Each concept shall be related to a textual definition (although in future those definitions might be generated from the modeled definition). A textual definition shall consist of one or more sentences in the natural language that is indicated as the context for its preferred name. The textual definition shall begin with a reference to the direct supertype concept and shall then specify in which respect the defined concept is distinguished from its direct supertype and from its neighboring concepts that are also subtype of the same supertype.

A textual definition may be extended with a description of what is typically but not necessarily the case. Such a description shall be expressed in a separate sentence and shall begin with the word ‘Typically’. It may also be extended with examples.

For example:

vessel is a container that is intended for storage and/or processing of fluids. Typically it is closed.

Such a textual definition can be added on the same line in a Gellish Database table as the specification of the name and the supertype of the concept, as is illustrated in Table 3 in the previous paragraph.

3.5 Step 5, Specify a discriminating aspect of a physical object

The fact that a concept is a subtype of its supertype concept implies that there is at least one difference with the definition of the higher level supertype concept.

When the defined concept is a kind of physical object, then in most cases the subtype concept is distinguished from its supertype by a particular discriminating aspect. The supertype concept has freedom (or more freedom) for the values of the aspect than the subtype, whereas the subtype concept is defined by a particular value or a narrower value range for that aspect.

For example, the concept rose (flower) is not defined by a particular colour, although each rose has a colour. However, the subtype concept 'red rose' is defined by the fact that its colour is by definition red.

This means that the discriminating aspect shall be specified for the supertype. This can be done in Gellish as follows:

UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object	UID of UoM	UoM
101	rose	201	5,652	has subtypes that have as discriminating aspect a	550,003	colour		

Table 4, Specification of a discriminating aspect

Then the subtype red rose is defined by its red colour value as follows:

UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object	UID of UoM	UoM
102	red rose	202	5,317	is by definition a possessor of a	103	colour of red rose		
103	colour of red rose	203	5,283	is by definition qualified as	551,759	red		

Table 5, Specification of a discriminating aspect value

Note 1: The value of the discriminating aspect is the real discriminator that defines the subtype concept.

Note 2: The aspect value red is a qualifies the possessed aspect ‘colour of red rose’ and does not qualify the red rose directly, nor does it qualify the concept colour in general.

The *concept values*, such as red in the above example, are concepts that also need to be defined in the dictionary or need to be selected from the Gellish Dictionary. A definition of such a concept value is done in a similar way as the definition of other concepts, except that the a concept value is defined by a qualification relation (<is a qualification of>) instead of a specialization relation, whereas no discriminating aspects need to be defined. The definition of an aspect value (also called a qualitative aspect) can be specified in Gellish as follows:

UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object	Definition
551,759	red	204	1726	is a qualification of	550,003	colour	is a colour that is chromatic resembling the hue of blood.

Table 6, Definition of an aspect value

Sometimes the discriminating aspect is quantified by a number on a scale, indicated by a unit of measure. The relation type that indicates the quantification on a scale can also indicate whether the value is equal, greater or less than the number of the scale. For example:

UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object	UID of UoM	UoM
104	M6 bolt	205	5,317	is by definition a possessor of a	105	diameter of a shaft of an M6 bolt		
105	diameter of a shaft of an M6 bolt	206	5,492	shall be quantified on scale as	920,106	6	570,423	mm

Table 7, Specification of a discriminating quantitative aspect value

The other relation types are:

5,280 can be quantified as greater than

5,281 can be quantified as less than

Typical kinds of aspects (subtypes of aspect) that need to be considered to determine discriminating aspects are:

- Aspect
    - Technology (operating principle)
    - Shape
    - Size
    - Topolology (structure)
    - Material of construction

A discriminator is not necessarily an aspect, it can also be one (or more) of the following kinds of dicriminators:

- Function (role in an activity or process)
    - Application (purpose?)
- Composition
    - Obligatory parts
    - Configuration (arrangement?)
- Organisation (owner, custodian, etc.)

The specification of such other discriminators is discussed in the next paragraphs.

3.6 Step 6, Specify an intended function

A concept can be defined by a discriminator that is its intended functions. Such a function is a role for which objects of this kind are designed. It means that a well-formed object of such a kind is intended to be suitable to perform or enable an activity or process to take place. The kind of object therefore has intrinsic aspects that enable to perform such a function, although such aspects might not be specified explicitly.

Apart from the function for which such a kind of object is made suitable, it is possible that objects of such a kind may also have other functions.

The intended function can be specified as follows:

UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object
130,069	compressor	207	4,717	has as an intended function a	191,936	compression

Note that the right hand object is an occurrence (activity, process or event), which is the activity that is intended to be performed by the fulfiller of the function (the player of the role).

3.7 Step 7, Specify defining parts

A concept may be defined by the fact that it has by definition particular kinds of other objects as its part(s). Typically such parts enable the capabilities of the assembly to perform its intended function. Such parts are discriminators that define the whole assembly. Therefore we call such kinds of part, defining parts.

Such defining parts can be distinguished from components that are part of a well-formed whole, but that do not define the whole. For example, a well-formed person has two legs, but a person without legs is still a person. So a leg is not by defining part of a person and that composition relation is not part of a Definition Model of the concept person.

For example, the concept bike is defined by its obligatory parts: frame, wheel, saddle and handlebars. Without any of such a part the object would not be able to function and is therefore not a bike. This is specified as follows in Gellish:

UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object
670,171	bike	213	5,519	has by definition as part a	40,080	frame
670,171	bike	214	5,519	has by definition as part a	130,679	wheel
670,171	bike	215	5,519	has by definition as part a	520,195	saddle
670,171	bike	216	5,519	has by definition as part a	109	handlebars

Table 8, Specification of defining parts

Note, a part can also by definition be part of a whole. This is specified by the inverse expression of the relation type 5519 <is by definition a possible part of a>.

The cardinality constraints determine the minimum number of objects of the same kind that can be part of the same whole at the same time or the minimum number of wholes for one part. Those cardinality constraints are described in more detail in paragraph XX.

The various parts of different instances of the whole can be different from each other, because there may be subtypes of the parts. Those subtypes of the parts can be the cause that also subtypes of the assembly can be defined. For example, there are different types of frames for man and for woman. So a bike has options for the kinds of frames, but if we define that a man's bike has by definition a man's frame, then for man's bike has no option for its frame type.

3.8 Step 8, Specify aliases and synonyms

Aliases, synonyms, codes and translations specify alternative terms ('names') for the same concept. Therefore, in the expression of such facts the UID of the left hand objects and the UID of the right hand object are the same.

Language	UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object
English	670,171	bike	217	1,981	is a synonym of	670,171	bicycle
Dutch	670,171	fiets	218	4,691	is a translation of	670,171	bicycle
International	430,266	H2O	219	1,983	is a code for	430,266	water
English	430,465	AC	220	1,982	is an abbreviation of	430,465	alternating current

Table 9, Specification of aliases and translations

There are several other relation types available to specify other kinds of strings that are used as names of things, such as <is a …> long name, serial number, title, location code, identifier, etc. They all are subtypes of relation type 1980.

3.9 Step 9, Specify pictures and drawings about a concept

Definitions of concepts may partly be provided in graphical form instead of in data and textual form.

The concept definitions may also be illustrated by examples or typical exemplars. Such examples or typical exemplars are individual objects that can be used as illustrations of instances of the concept or as a template to copy them for the creation of individual exemplars that comply with the definition of the concept. However, objects that differ from those examples may still belong to the kind that is represented by the concept, because the examples and typical objects are not normative.

This means that a Gellish Dictionary may contain normative drawings and illustrative drawings and pictures. Normative drawings are drawings that specify aspects that are by definition the case for the concept. Illustrative drawings and pictures present examples or typical examples of instances that nevertheless are related to the concept (kind of thing) for which it is an illustration.

A picture or drawing is an individual physical object. For example, some ink on a piece of paper or bits arranged in a pattern on an electronic or optical device. In a Gellish Dictionary we only deal with electronic pictures and drawings such as files in various formats. For example files in jpg, tiff, pdf, dwg, etc. formats.

Such an individual picture or drawing file is defined in Gellish by a classification relation as an electronic data file and is specified to be an element of a directory with as name a URL address. Gellish makes an explicit distinction between individual physical drawings (e.g. three drawings: one in pdf format, one in dwg format and one on paper) and the (common) qualitative information that is presented on each of those drawings. The files are presentations of the same qualitative information (being the drawing or picture content). Thus, such qualitative information is a qualitative concept that can classify the contents of several individual physical drawings or pictures.

Assume that "picture-1 of a rose" is a representative picture that illustrates what a rose is and assume we have (among others) a rose.jpg file with that picture. Then, altogether the picture is documented in Gellish as a representative picture of the concept 'rose' as follows:

UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object
106	rose.jpg	208	1,225	is classified as a	490,533	electronic data file
106	rose.jpg	209	1,227	is an element of	107	http://
107	http://	210	1,225	is classified as a	970,274	URL address
108	picture-1 of a rose	211	4,996	is presented on	106	rose.jpg
108	picture-1 of a rose	212	1,726	is a qualification of	490,120	picture
101	rose	213	1,911	is a concept that is described by	108	picture-1 of a rose

Table 10, Specification of drawings and pictures

Note: Relation type 2072 expresses that the concept rose is graphically represented by annotation (a line or area or an assembly of lines and/or area's). It may be that the annotation is a part of a bigger drawing, whereas the total drawing is presented on the file. In that case the relation between the picture and the total drawing is as follows:

picture-1 is a part of picture-2

The fact that annotation is defined as any arbitrary figure and is not restricted to symbols means that graphical representation is the same as visualization.

3.10 Information about a concept

Strictly speaking information about a concept does not belong to a definition model. However, one might want to capture informative text or references for clarification. This can be done in two ways:

1. By referencing a document
2. By inclusion of a piece of text in the model.

3.10.1 Referencing a document

The definition of a document file is similar to the definition of a drawing or picture as described in the previous paragraph. Relation type 1911 expresses that the concept is described by the information, which is the document content that is contained in one or more physical files. This is illustrated by the example below.

UID of left hand object	Name of left hand object	UID of Fact	UID of Relation type	Name of relation type	UID of right hand object	Name of right hand object
110	API 617.pdf	220	1,225	is classified as a	490,533	electronic data file
110	API 617.pdf	221	1,227	is an element of	107	http://
107	http://	210	1,225	is classified as a	970,274	URL address
111	API 617	223	1,726	is a qualification of	910,137	standard specification
111	API 617	224	4,996	is presented on	110	API 617.pdf
130,069	compressor	225	1,911	is a concept that is described by	108	API 617

3.10.2 Inclusion of text in the model

Instead of a reference to a document it is also possible to include the text directly in the model. This is done by inclusion of the text in the 'full description' column on the line where the information is defined by a qualification as information or as a subtype of information (in the example below the text is a qualification of 'requirement').

UID of left hand object	Name of left hand object	UID of relation type	Name of relation type	UID of right hand object	Name of right hand object	Full description
130,069	compressor	5,298	shall be compliant with	5	API 617	
5	API 617	1,726	is a qualification of	970,002	information	
130,069	compressor	5,298	shall be compliant with	6	Par. 5.3.1	
6	Par. 5.3.1	1,726	is a qualification of	970,007	requirement	The purchaser and the vendor shall mutually determine the measures that must be taken to comply with any governmental codes, regulations, ordinances, or rules that are applicable to the equipment.

4. References

1. Gellish Database Table Definition, http://sourceforge.net/project/showfiles.php?group_id=28353.
2. Gellish Modeling Method – Part 1 – Architecture.
3. Gellish Modeling Method – Part 2 – Creation of Domain Dictionaries and Dictionary Extensions.
4. Gellish Modeling Method – Part 3 – Specification of requirements and verification of deliverables.
5. Gellish Modeling Method – Part 4 – Creation of Facility Information Models.

5. Appendix A, Summary of used relation types

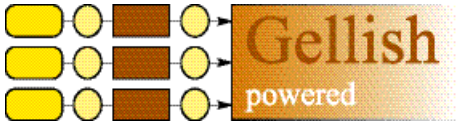
The following relation types are used to create a core of Requirements Models as described in this document. Additional relation types will be needed to add other kinds of requirements in the form of data or when the content of documents is modeled.

Note that each relation type has an inverse expression. When the inverse expression is used the left hand object and right hand object shall be exchanged.

UID of Relation type	Name of relation type	Description
1,446	is a specialization of	is a relation between two concepts that specifies that the first concept is a subtype of the related second concept (the supertype). This means: what is true about the supertype concept is also true (by inheritance) for the subtype concept. The subtype concept is distinguished from the supertype concept by additional constraints, which are also called discriminators. Example: motorway <is a specialization of> road
5,652	has subtypes that have as discriminating aspect a	is a relation between a concept and a conceptual aspect that specifies that the concept has subtypes, which subtypes are distinguished from the concept because they have by definition a particular value for the conceptual aspect. Example: instrument <has subtypes that have as discriminating aspect a> measured property
5,317	is by definition a possessor of a	is a relation between a concept and a kind of aspect that specifies that the concept is defined by having that kind of aspect (by definition). Example: pipe <is by definition possessor of a> pipe diameter
5,283	is by definition qualified as	is a relation between a possessed aspect concept and a qualitative aspect that specifies that the possessed aspect is qualified by the qualitative aspect (by definition). Example: construction material of a copper bolt <is by definition qualified as> copper
1726	is a qualification of	is a relation between a conceptual aspect and a qualitative aspect that specifies that the qualitative aspect has the conceptual aspect as its nature. Example 1: red <is a qualification of> colour Example 2: this text <is a qualification of> information
5,492	shall be quantified on scale as	is a relation between a possessed aspect concept and a numeric value that specifies that the aspect has a value that is equal to the numeric value when being quantified on a particular scale (which scale is specified separately) Example: diameter of an M6 bolt <can be quantified on scale as> 30 mm
5,493	shall be quantified as greater than	is a relation between an aspect and a numeric value that specifies that the aspect has a value that is greater than the numeric value when being quantified on a particular scale (which scale is specified separately)
5,494	shall be quantified as less than	is a relation between an aspect and a numeric value that specifies that the aspect has a value that is less than the numeric value when being quantified on a particular scale (which scale is specified separately)
4,717	has as an intended function a	is a relation between a concept and a kind of activity or process that specifies that the concept is defined so that instances of that kind are intended to be suitable to be performer of activities or processes of the specified kind. Example: compressor <has as an intended function a> compression
5,519	has by definition as part a	is a relation between two concepts that specifies that a member of the first concept has by definition one or more parts that are member of the second concept. Example: car <has by definition as part a> engine
1,981	is a synonym of	is a relation between two text strings (names) that both denote the same concept, that specifies that the first text string is a synonym of the second text string, whereas that first text string originates in the context for the first text string.
4,691	is a translation of	is a relation between two text strings (names) that both denote same concept, that specifies that the first text string is a translation of the second text string, whereas that first text string originates in the language (and context) for the first text string.
1,983	is a code for	is a relation between two text strings that both denote the same concept, that specifies that the first text string is a code for the second text string, whereas that code originates in the context for the code.
1,982	is an abbreviation of	is a relation between two text strings that both denote the same concept, that specifies that the first text string is an abbreviation of the second text string, whereas that abbreviation originates in the context for the abbreviation.
1,225	is classified as a	is a relation between an individual thing and a kind of thing (concept) that specifies that the individual thing is a member of the kind. Example: file-1.doc <is classified as a> electronic file
1,227	is an element of	is a relation between an individual thing and a collection of individual things that specifies that the individual thing is an element of the collection. Example: file-1 <is an element of> directory-1
4996	is presented on	is a relation between information and an individual physical object that specifies that the information is expressed on the physical object. It is expressed as an aspect of the physical object. For example as a spatial (shape and pattern) aspect of ink on paper. Example: pictogram-1 <is presented on>file-1.dwg
1,911	is a concept that is described by	is a relation between a concept and qualitative information that specifies that the concept is described by the qualitative information. Note that the qualitative information can be graphical or textual. Example: motorway <is a concept that is described by> pictogram-1

6. Certification criteria

The Gellish@Work organization is accredited to certify software as being able to import and/or export data files with a Gellish compliant electronic Dictionary and to certify Gellish Dictionaries themselves. Gellish compliant software or data sets can be recognized by the following Gellish logo 'Gellish powered' in the following layout:



This chapter describes the certification criteria for a Gellish Dictionary.

6.1 What is a correct Gellish Dictionary

Correct Gellish is a collection of Gellish expressions of facts (main facts with auxiliary facts) that are presented in one or more Gellish Database tables or a representation thereof and that are compliant with the following rules.

Rule 1: Only use of properly defined concepts and relation types.

In each relation between concepts (kinds of things / classes) the concepts shall be:

- Either selected from the Gellish dictionary,

or

- Properly defined as a subtype of a concept in the Gellish dictionary,

whereas the used concepts are related by relations that are classified by kinds of relations (relation types) that are also selected from the Gellish dictionary.

How to verify rule 1:

1. Verify for each used concept whether the concept (UID) has a sequence of specialization relations (including also subtypes of specialization relations) with more general concepts, which sequence, when following the hierarchy, terminates at the concept "anything". If the top concept is not "anything", then an error shall be generated including a report about the concepts that actually terminates the sequence.

   This can be verified using the built-in verification rules of the Gellish Browser in combination with the Gellish Dictionary.

2. Verify for each used relation type whether the relation type exists in the Gellish Dictionary (TOPini part). Proprietary extensions of relation types shall be verified separately.

   This can be verified using the built-in verification rules of the Gellish Browser in combination with the Gellish Dictionary.

Rule 2: Proper definition.

A subtype of a concept is properly defined if it is described by a textual description that is a text string that expresses that the subtype is a specialization of its direct supertype concept and by which aspects and aspect values it is distinguished from its supertype and from its 'sister' subtypes[1]. Instead of such an explicit textual definition it is also allowed (and in particular business contexts required) to specify modeled definitions. In a modeled definition the supertype is specified to have subtypes that are defined by a particular kind of aspect, whereas the value of that aspect is specified for each subtype. An ideal modeled definition can be used to generate a textual definition.

How to verify rule 2:

Verify the text in the full description column by manual inspection.

[1] For an extensive discussion of the determination of proper classes, see the 'Gellish English Dictionary Extension Manual' at http://sourceforge.net/project/showfiles.php?group_id=28353.

Rule 4: Grammatical correctness.

Each relations between things is classified by a kind of relation (relation type) as specified in column 60 and 3 of a Gellish Database table. Such a kind of relation defines the kinds of roles that are played by the related things in a relation of such a kind. Each role player shall be able to play the role that it is required to play according to the kind of relation. The (required or specified) nature of a role player shall not be in conflict with the nature required by other relation types in which the role player plays a role.

The Gellish Dictionary (in TOPini) includes "facts about conceptual relations" that define the relation types of the Gellish language. These definitions also specify the required kinds of roles and the kinds of things that can play those kinds of roles (which implies that also their subtypes can play those roles).

Example 1: the Gellish expression C is a specialization of D is only correct if C is a concept, because the <is a specialization of> relation requires as first role a subtype, wheras a subtype role can only be played by a concept (or by a subtype of concept). The relation type requires as second role a supertype, which can also only be played by a concept. So, D must also be a concept.

Example 2: the Gellish expression "A is by definition a performer of a B" is only correct if A is a subtype of physical object, because the facts in the Gellish Dictionary specify that an <is by definition a performer of a> relation requires as first role a performer and it also specifies that a performer role can only be played by (a subtype of) physical object. The relation type requires as second role a "performed", which is a role that can only be played by (a subtype of) occurrence. So B must be (a subtype of) occurrence. These conclusions about the nature of A and B shall not be in conflict with the nature required by other relation types in which A or B plays a role.

How to verify rule 4:

Verify the consistency of the object type of the left hand and of the right hand object according to various relations in which those objects are involved as follows: When something is related to another thing by a relation of a particular kind, then it can be derived from the definition of the relation type in the Gellish Dictionary what kind of thing it must be. This can be determined as follows: according to the Gellish Dictionary (TOPini part) the relation type requires a first kind of role and a second kind of role. This is either directly specified for the relation type or it is inherited from its supertype relation type. The Gellish Dictionary also specifies which kind of thing can play such a kind of role. This can also be specified either directly or by inheritance from its supertype kind of role. When the object is a left hand object and the Gellish phrase for the relation type is a normal phrase (or both the opposite), then the thing shall be of the kind of thing that can play the first kind of role. When it is a right hand object or an inverse phrase then it shall be of the kind of thing that can play the second kind of role, etc.

Conclusions about the kind of thing that are derived from various relations for something and its explicit classification relations or specialization relations (or their subtypes) shall not be in conflict with each other. There is no conflict if the conclusion about the kind of thing drawn from the classification or specialization relation(s) is the same or a subtype of the kind that is required by a relation type.

This should be verified by the built-in rules in the Gellish Browser.

If the specialization hierarchy of concepts is incomplete (rule 1 is not satisfied), then this verification is impossible.

Rule 5: Semantic consistency.

The relation types that classify the relations in Gellish expressions determine the nature of the things that play the roles in the relation. Those natures shall be consistent with the natures that are required by other relations in which the same thing plays a role.

Example 1: from the Gellish expression C is a specialization of D it can be concluded that C is a concept. If there is another Gellish expression that states that C is a part of E, then from that relation it can be concluded that C is an individual thing. These two expressions are thus inconsistent as C cannot be a concept and at the same time be an individual thing.

Example 2: from the Gellish expression A is a performer of B it can be concluded that B is an occurrence. If there exists another Gellish expression that states that B is a performer of F then the two expressions are inconsistent, because B cannot be an occurrence as well as a physical object.

How to verify rule 5:

There is no conflict between a conclusion about the kind of thing drawn from one relation type and a conclusion drawing from another relation type for the same thing, when the required kind of thing is the same or is a subtype or supertype of the kind that is required by the other relation type. In other words they are in conflict when the conclusions about the kinds of things belong to different branches of the specialization hierarchy.

This should be verified by the built-in rules in the Gellish Browser.

Rule 6: Inheritance of constraints.

All constraints (facts that are by definition the case) that are valid for members of a supertype concept shall also be valid constraints for any of the members of its subtype concepts.

Example 1: If a car has by definition a motor, then there may not be subtypes of car that do not have by definition a motor and there may not be individual (well-formed) cars without a motor.

How to verify rule 6:

Whether a relation is a constraint is be determined by the relation type that classifies the relation. The following list of relations types or their subtypes specify constraints: xxxx, yyyy.

If a constraint (or more generally a fact) is redefined for a subtype, without or with identical cardinality constraints (see rule 7), then the redefinition shall be reported and its status shall be changed into 'inherited'. This shall imply that the fact shall be ignored, as the constraint shall be inherited from its supertype concept. However, the Gellish Browser shall verify whether facts with the status inherited are indeed inherited from a specification for a supertype, as facts about the supertypes might have been changed so that the inheritance is not the case anymore.

Rule 7: Inheritance of cardinality constraints.

The cardinality constraints of a subtype concept shall be equal or narrower than the cardinality constraints of its supertype concept.

How to verify rule 7:

The Gellish Browser shall verify whether minimum and maximum cardinality constraints that are defined for a concept are narrower than possible definitions of cardinality constraints for the same relation type for its supertypes. It is incorrect if a cardinality constraint of a subtype is wider than the cardinality constraint of its supertype.

Rule 8: Completeness of auxiliary facts.

A Gellish Database table shall comply with the column definitions according to one of the subsets that are defined in the "Gellish Database Definition" document. Each field on each row in such a table shall have values that comply with the rules defined in that document.

How to verify rule 8:

The Gellish Browser shall verify whether a value in a field is present when that is obligatory and whether the value satisfies the format constraints for the field.

- NTA8611 guidelines that are applicable for dictionaries

- Relevant rules from CHEOBS
- Checks as performed by Paulus Kooiman

