## Rationale for SIMF Relations & Situations

Relations are at the core if conceptual and systems modeling and there are different theories of and approaches to the definition of relations and how to model them. This section is intended to define the goals and then to explain the rationale behind the structure and semantics of SIMF relations and situations. This can then serve to evaluate other options for representation as well as other goals.

### Goals

#### Relations involving multiple entities and values

Relations and similar concepts can be “unary”, like OWL properties, Binary with exactly 2 ends or n-ary with any number of involved entities and values. Our goal is to be able to represent all of these variations and understand how they are related.

#### Relationships that may only be true in specific timeframes or context

While many information systems only consider “snapshots” of the world – the world and relationships between things in it are ever changing. We want to be able to understand when a relation holds – which includes its timeframe as well as other context that could impact its interpretation. This is sometimes known as “4D”. We want to be able to represent both “3D” and “4D” concepts in a way that they can both be understood and related. Some 4D models are very complex or hard to comprehend.

#### Relationships as first-class entities

In that relationships may be true in different times and context and may have properties and relationships relative to their interpretation – relationships should be “first class” entities that can participate in other relations.

#### Relations have identity

As “first class” entities, relations have identity and may therefore have identifiers.

#### Unification of properties and relations

Many modeling paradigms have both properties and relations. At times it can be somewhat arbitrary as to representing some concept as a property or relation that has down-stream impact on what can be done with the concept. If there are different concepts of properties and relations, the difference should be clear and the mapping between them defined.

#### Unification of verb and noun centric relations and predicates

Some modeling paradigms will model the “ends” of relations as verbs where as others model them as nouns – yet these seem to represent the same or almost the same concepts. We should be able to admit both noun and verb concepts and understand how they are connected.

#### Clarification of roles, variables, ends and properties

There are various interpretations of “role” as well as the other descriptions of the “things related” sometimes known as “ends”, “variables” or “properties”. In OWL there are no ends, but fixed roles of “domain” and “range”.

We should have clear definitions of these concepts and how they are related.

#### Support for situations that are not relationships

Relationships connect various entities and values but there are larger situations (or states of affairs) that do so as well. There should be a clear demarcation for what makes something a relation vs. a situation and, where applicable, reuse of the concepts to define them.

#### Representation of speech acts and data structures

While conceptual modeling may focus more on relations, implementation technologies may focus on more hierarchical “data structures” that, at least, need to be able to be represented in the model such that mappings can be performed.

Communications can be thought of in terms of “speech acts” that may combine multiple facts and/or relations – the representation of and mapping to such speech acts should be supported.

It should be clear if/when these structures are “relationships”.

#### Intuitiveness & keeping the simple, simple

The representation of relationships and their type definitions (relations or relators) should be intuitive to the general public when viewed through an appropriate notation. Likewise, they should be intuitive to technology professionals when viewed through an appropriate notation.

While there are complex relations – the representation and views of simple relations should be as simple as practical.

#### Precision

The definition of relations should be able to capture the precise meaning and constraints of the concept. However, the level of precision should be optional.

#### Types and instances

The connection between definitions and instances of relationships as well as the “ends” of relationships should be clear.

The following has been proposed (Jim-L):

One instance of a relation being true is represented by a single n-tuple, called a relationship; the relation itself is the totality of all relationships of the same type. A marriage, for example, is a relationship between two individuals, but the relation called marriage is the totality of all marriages.

#### Generality and generalization

Where a relation concept is applicable across a wide range of views, use cases or communities – the definition of the relation should be able to be used across them and, where needed, specialized for more specific interpretations. All concepts and sub-concepts of a relation should be able to be specialized.

#### Definition, use and reuse of named concepts

The definition of a concept and the terms used to identify those concepts should be able to be defined once, in one place, and used wherever that concept is applicable.

Note that models such as UML tend to make “association ends” hard to reuse in other context.

On the other hand, where the same concept is defined in more than one place, those different definitions should be able to be connected.

#### Ability to represent bad architectures

While we should be able to represent good architectural practices it should be recognized that we will be federating with poor architectures and need to be able to represent them as well.

#### Multiple names

Any named concept, including Relationships and their “ends”, should be able to have multiple names in the same or different context.

#### Decomposability

A relation concept should be able to be defined based on a composition of more primitive concepts.

#### Multiple notations

Relations should be able to be represented by diagrams with “lines”, tables and text.

#### Semantic grounding

Relation concepts should be grounded in a formal logic.

#### Modal commitment of verb terms

There is stylistic variety in the use of nouns Vs. verbs on the end of relations. In general, the noun forms work better for reified relationships with roles whereas the verbs make sense as directional “paths”. In many languages, including English, Verbs also tend to convey a “Mode” such as timeframe or preference. For example, if we consider the weight of Jill we could say:

* Jill weighs (present tense)
* Jill weighed (past tense)
* Jill will weigh (future tense – prediction)
* Jill should weigh (preference or obligation)

So it seems like there are 2 kinds of concepts here: the weight of a person and the mode of the assertion – past, present, obligatory, prediction, etc.

Where verb phrases commit to a mode (e.g. present perfect), that mode should be specific and in some way connected to the non-modal concept that unifies them (weight in this example).

Some styles assume that present tense does not imply a mode – that assumption should be specific in the logic. In this case the mode would be implied by the context of the verb phrase, such as a separate modal operator on a “snapshot”.

#### Cardinalities / Multiplicities

It should be clear what the general requirements of cardinalities are for all relations (e.g. can a role be populated more than once in the same relation or are relationships n-tuples). Rules should be provided to specify relation cardinalities in all directions (e.g. to and from the relation).

#### Support for mapping

Relations and relationships should be able to support mapping to multiple representations of the same concepts.

#### Open world definitions, closed in a context.

All concepts data structures should be open world by default – that is multiple sources of “truth” should be able to be combined but then interpreted in a fixed (closed) context.

#### Do not require noise terms

In many modeling paradigms it is required to create “named elements” that are not really representative of concepts in the domain of discourse. Conceptual modeling should not require the introduction of such “noise” terms.

#### Conceptual models

Relations should be able to be used to represent concepts of real and possible worlds without requiring any information system or communications pattern.

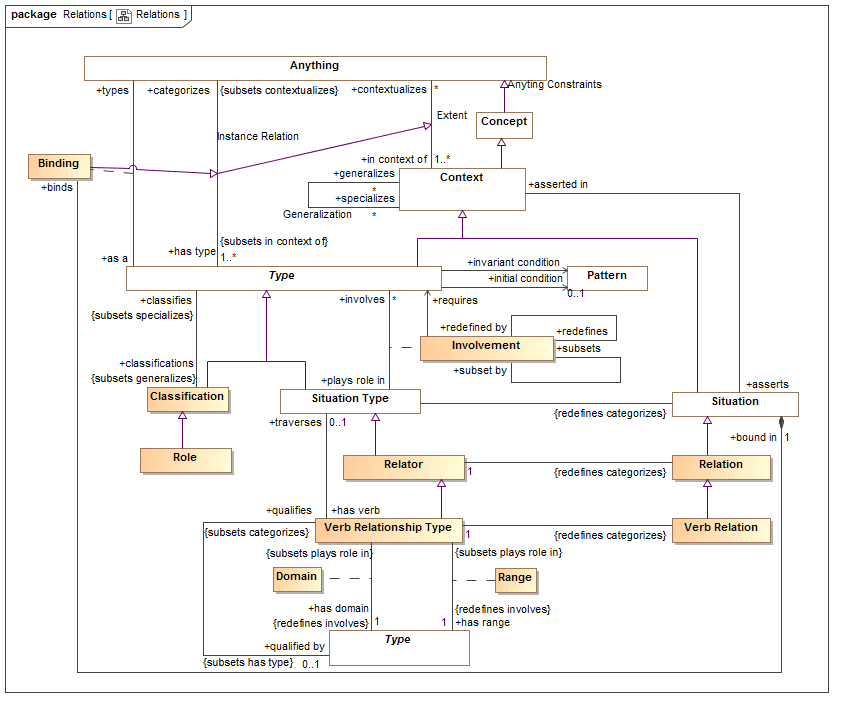
#### Understanding relations as “truthmakers”

The connection between relations and the “truth” they assert about a real or possible world should be clear. (See “We need to discuss the Relationship” – Guizzardi ).

### Model Design Choices

#### Various design choices will reference these models

Note: Specific definitions for each concept are in the model file.



#### Terms

Relation is used as the instance of a relationship, type relationship type is called a “Relator” to correspond with [Guizzardi 2015]. Note these terms are in conflict with those proposed in 1.1.2.11.

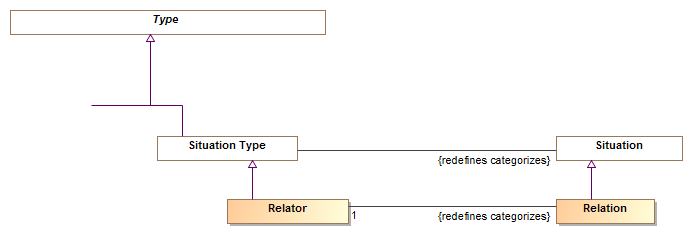
#### Relations as first-class

Relations are just another kind of “thing” – things that connect things. As such relations can have properties and participate in other relations.

**Supports or satisfies: 1.1.2.3**

#### Relations as type/instance

Like all other kinds of entities, relations and relators form a type-instance pair and are related by a type/instance relationship.



Note: While less explicit in the model to avoid complexity – there is also a type/instance relation between “Involvement” and “Binding” , see below in : 1.1.2.8.

**Supports or satisfies:** 1.1.2.3, 1.1.2.11.

#### Roles as types not bound to a single relation

Discussion

Role concepts are used in different ways. For example the role “Teacher” can be used as a general non-rigid classification (e.g. Jill is a teacher) as the end of other relations (e.g. Jill is a good teacher) as a part of a specific relation (e.g. Jill:Teacher teaching John:Student or Jill:Teacher teaching at Oakton High:School) or as a qualification (e.g. Jill has her teacher certification in Florida).

In UML “roles” can be at the end of associations or are sometimes reified as classes. If a noun based association style is being used there is sometimes redundancy between the association end name and the reified role. There are multiple choices for the same idea in UML:

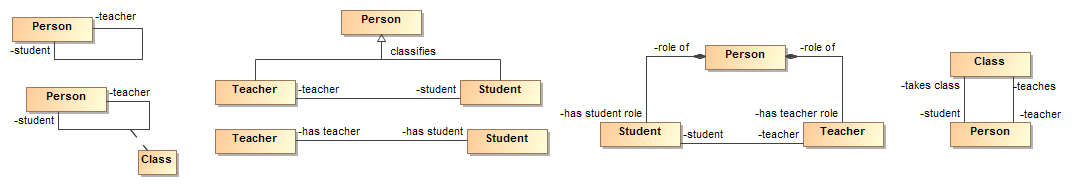


Figure 1-Examples of UML styles for roles

Note that the “reified” treatments would allow other properties and relations of the role and/or class, so for example we could have properties of teacher like “is good at job” or associations like “qualifications”. The reified roles adhere to the person where as the association and roles adhere only to the relationship.

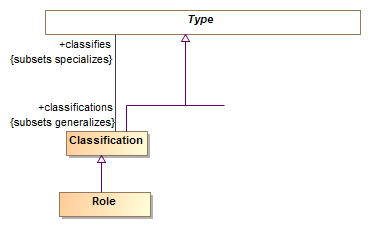
**Fundamental question**: Is there an overall concept of “Teacher” that serves all of these uses for the role? Is there a fundamental concept of the “occurrence”, in this case a class.

**Design Choices**

In support of 1.1.2.13 we define a “Role” as a “classification” type (non-rigid type). It is then possible to have roles participate in multiple relationships. These roles have names and are assumed to classify another type.

In that roles are named and can be reused, they are not necessarily defined within the context of a relationship and regardless of where defined, could be “reused” in any number of situations or relations.

So, for example, the concept of “Teacher” can be defined as a role of a person and can then be used in a “teaches” relation, a “as qualified” relation, “is good” property – etc.



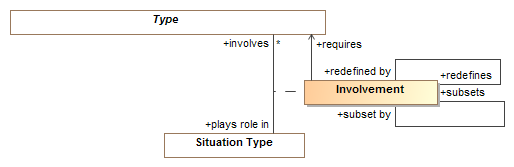
Note that the same person could also have different “Teacher” roles defined in different context if necessary – it is TBD if this is a good idea.

Note: “Phase” is the other subtype of classification.

**Supports or satisfies:** 1.1.2.41.1.2.61.1.2.71.1.2.13

#### Involvement as an un-named “end”

In support of 1.1.2.1 there must be a way to define all the “things related” in a relation. These are sometimes called properties, association ends, variables or roles. We are defining this connection as “Involvement” – that is the things involved in the relation or situation (some of these concepts are defined with respect to more generic situations, that design choice is discussed separately).



The quandary is: if the role and its term is defined in a role type that is reusable (see 1.1.3.4) the name of the “end” seems redundant. If sometimes you name the end and sometimes you name a reified role you have locked in a design choice that may be incompatible with other viewpoints.

**Design Choices**

**Involvement as a link:** Involvement is one of the few elements in our model that is not a named concept – it is simply a link between the relationship and the things it involves – it is expected that the name of the thing involved is the name applicable as the end name. If it isn’t, the modeler is always free to make another role type as the end.

**Involvement of any type**: Involvement of non-roles: For some relations – particularly those more at the level of properties, there is no role that is clearly different from the fundamental type of the thing. For example, weight of a physical object – what is the role of the physical object different than being physical? So as not to force the creation of “noise” terms or concepts (1.1.2.21) we allow any type (including roles) to be involved in a relation.

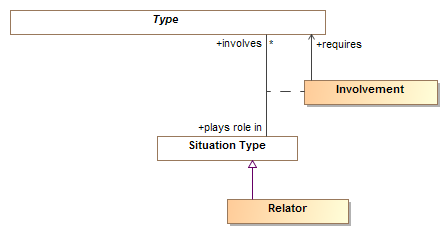
**Refinement of involvement**: To support refinement within the context of a relation “involvement” has specific relations corresponding to the UML concepts of redefined and subsets. Subsets is seen as equivalent to OWL sub-property.

**Supports or satisfies:** 1.1.2.1,1.1.2.4,1.1.2.6,1.1.2.12,1.1.2.13, 1.1.2.21

#### Relations as situations

Relations assert some “truth” between the involved things that is frequently only valid for some period of time. Truths between involved things can be very small (e.g. Jill’s weight on Jan 3rd 2010) or very large (e.g. the earth for all time and everything on it). They can be physical or purely conceptual – like a fish or an obligation. Some situations are fixed such that they can only be “true or false” and are otherwise atomic or can change over time. Further, there can be roles of very large things (the President of the United States) and small things (the object supported by a supporting object).

To represent the very general arrangement of most anything we use “Situation” defined as: “An identifiable arrangement of individuals, assertions and the relations and assertions between them over a timespan. Any "condition" that exists is a situation”, as is any occurrence.



Situations are then the defining element for “Involvement” as situations involve other things.

Other kinds of situations that involve roles include data structures and speech acts – which we see as other kinds of situations.

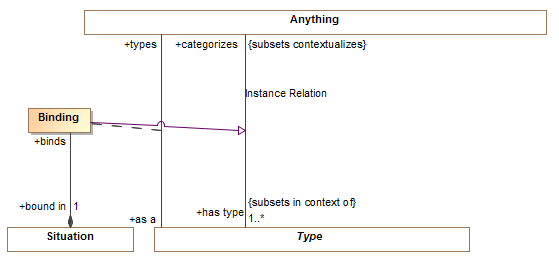
Relators (the type of relations) are a subtype of situation that is expected to be atomic – not change over time where as general situations may be dynamic (such as the contents of my desk or the position of a hurricane). So if Jill weighs something else today, that is a different relationship – not a change in the relationship. Each relationship (and situation) has a timeframe where by it is true (start and end time being possible properties of all situations).

**Supports or satisfies:** 1.1.2.2, 1.1.2.3, 1.1.2.7, 1.1.2.8, 1.1.2.23

**Issue**: The precise boundary between situation and relation needs to be better defined.

#### Binding

The type/instance relationship between a type and it’s “instances” (Things it classifies) seems clear. A relator is defined in terms of the types (or roles) it involves. An instance of a relation is bound to the things that play those roles (or fill those slots) with a “**binding**”. A Binding is an assertion that something is playing some role (as defined by an involvement) within a situation (or, an instance of a type in a context). Conceptually a binding is an instance of an involvement – but showing that way introduces complexities and relations we don’t want. Like an involvement, a binding is not named. It is referenced by the name of the type or role involved in a particular situation or relation (e.g. President of the U.S.). So, within the context of the “teacher teaches student” relation, Jill is the teacher and John is the student. These bindings are only asserted within this restricted context so we are calling the end “as a” (Jill as a teacher) type instead of “is a type”.



**Supports or satisfies:** 1.1.2.6, 1.1.2.7, 1.1.2.8, 1.1.2.11, 1.1.2.21

#### Verb Relationship Types

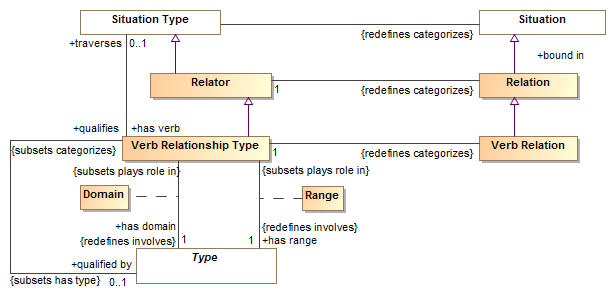
Domain concepts include both nouns and verbs. In many cases nouns and verbs used in relations assert the same facts, in different ways. To address this issue we are building on [Guizzardi 2015] and the concept of “Relators”. Relators are the fully reified, non-modal noun based representative of a concept.

If a relator is “read” from the perspective of one “end” to another the verb phrases make sense. In the relation Jill:Teacher teaching John:Student we can also say “Jill teaches John”, At least in English “Teaches” implies present tense. Some models may also include predicates like “Jill should-teach Physics”, which is clearly modal (this may or may not be bad form but it is done).

In that we will be importing some models from other forms, such as OWL, we may only have verb forms and not fully understand the "“Relator” – or may add it later.

To enable direct representation of verb concepts we introduce “Verb Relationship Type” and specialize the “Involves” concept with “Domain” and “Range” (as used in OWL). For a Verb Relationship the name of the relationship serves as the verb. It is permissible to just define verb relationships using domain and range.

A verb relation can be understood as a directed “reading” across a more general Relator. Where there are verbs that are inverses of the other there is an implicit Relator that is being “read” in each ways. A verb relation can be defined as traversing a relator – in other words it is a reading across that relator. In this way we understand all the verbs that may be different readings for the same relator.

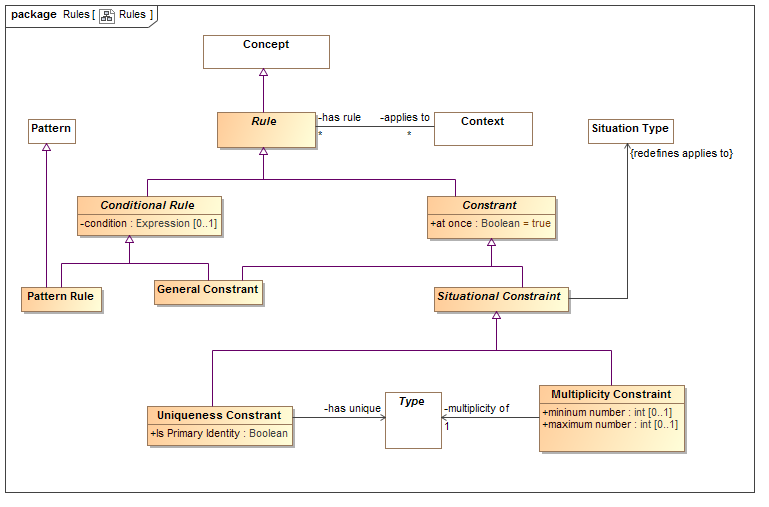


Where the verb implies some kind of mode, that mode is referenced by “qualifies”. Qualifies implies that the verb relation is always within the context of a mode relation of the type referenced by “qualifies”.

The combination of “Traverses” and “qualifies” allow all the “senses” and “directions” of the same fundamental “fact” to be understood.

Where a verb phrase traverses a relator the relator and all verb phrases that traverse it (for the same mode) can be inferred. This allows mapping between the different verb/noun styles without being confused by concepts that differ in their mode.

**Supports or satisfies: 1.1.1.18**, 1.1.1.24, 1.1.1.17, 1.1.1.10



#### Relation identity

Where a relation (or any type) involves a set of other types, a set of those types may be nominated as unique identifiers for the relations. “Uniqueness Constraint” is a rule that applies to a situation type (which includes relations) and says that type “has unique” set of involved other types/roles. To capture some paradigm, a uniqueness constraint may be specified as “primary”.

Note that the “applies to” relation is the situation or relation type the constraint is constraining.

“At once” (Inherited from constraint) specified that the rule constraints a snapshot, if false it is true for all time.

Issue: Have to consider if this is sufficient for identity derived from or across other relationships.

Constraint: The referenced “has unique” types must be involved in the “applies to” situation.

**Supports or satisfies: 1.1.1.4**

#### Multiplicity Constraints

Within a situation there may be constrains on the number of individuals that can “populate” a role. For example, the U.S. may have at most one President at a time.

Note that the “applies to” relation is the situation or relation type the constraint is constraining.

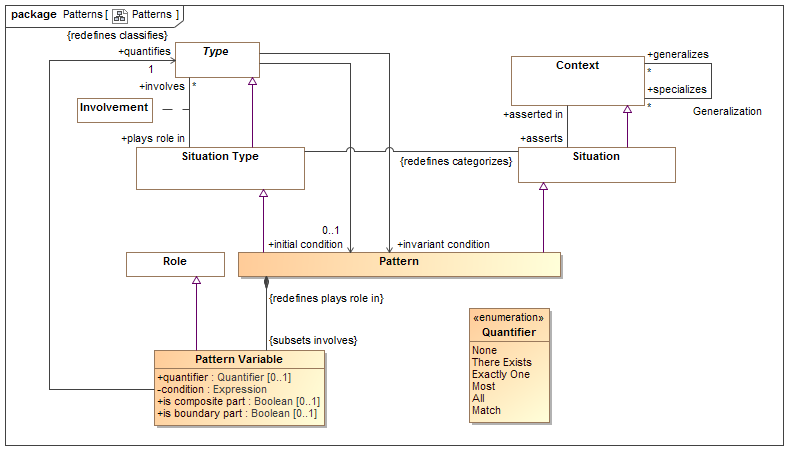
“At once” (Inherited from constraint) specified that the rule constraints a snapshot, if false it is true for all time.

**Issue**: If Relations are constrained to “Tuples” then it is implicit that the cardinality is at most one – this choice is still open.

**Supports or satisfies: 1.1.1.20**

**Issue**: Need to also support the “inverse” multiplicity – e.g. that a president can be a president of at most one country (this capability dropped out on the last simplification and needs to be fixed – how it is done depends on the “Tuples” choice).

### Patterns



#### Patterns invariant conditions

Patterns can have “Pattern Variables” that serve the purpose of quantification variables in some logics (e.g. for all X, X is on earth). The pattern holds the pattern variable and, as a context, can “assert” other situations – which include other relations. This is similar to a “composite structure” in UML. Such a pattern can be specified as invariant for a type – thus allowing a type to be “explained” in terms of other concepts.

**Supports or satisfies:** 1.1.1.16, 1.1.1.18

**Issue**: May want to make some specialized associations so the pattern model isn’t so abstract.

### Identifiers

Relations, like any concept, can have multiple identifiers.



**Supports or satisfies:** 1.1.1.16