University of Applied Sciences Northwestern Switzerland
School of Business

Semantic and Logical Foundations for Business Vocabulary and Rules

http://www.omg.org/spec/SBVR/1.0

SBVR Clause 10: Semantic and Logical Foundations

- The SBVR initiative is intended to capture business facts and business rules that may be expressed either informally or formally.
  - Formal statements of rules may be transformed into logical formulations that are used for exchange with other rule-based software tools.
  - Informal statements of rules may be exchanged as uninterpreted comments.
- Business rule expressions are classified as formal only if they are expressed purely in terms of
  - fact types in the pre-declared schema for the business domain, and
  - logical/ mathematical operators, quantifiers, etc.
- The following discussion of business rule semantics is confined to formal statements of business rules.

An Excursion into Logic

- The semantics of SBVR is defined by a mapping to predicate logic
- A predicate calculus consists of
  - formation rules (i.e. definitions for forming well-formed formulae).
  - a proof theory, made of
    - axioms (logical formulae).
    - transformation rules (i.e. inference rules for deriving new formulae).
  - a semantics, telling which interpretation of the symbols make the formulae true.
- Predicate Logic vs Propositional Logic
  - Propositional Logic is a formal system in which formulae
    - represent atomic propositions (having truth values true or false) or
    - are formed by combining propositions using logical connectives (and, or, not, …)
  - Predicate Logic considers the deeper structure of propositions

First-order Predicate Logic – Logical Symbols

The logical symbols include variables, logical operators and quantifiers.

- Variables are usually denoted by lowercase letters at the end of the alphabet *x*, *y*, *z*,..
- Symbols denoting logical operators are usually denoted as
  - ¬ (negation – logical not)
  - ∧ (conjunction – logical and)
  - ∨ (disjunction – logical or)
  - → (implication – logical condition)
  - ↔ (equivalence – logical equivalence)
- Symbols denoting quantifiers
  - ∀ (universal quantification, typically read as "for all")
  - ∃ (existential quantification, typically read as "there exists")

First-order Predicate Logic – Nonlogical Symbols

The nonlogical symbols include predicate symbols and function symbols

- The predicate symbols (or relation symbols) are often denoted by uppercase letters P, Q, R,... .
  - each predicate symbol has some arity ≥ 0
  - relations of arity 0 can be identified with propositional variables.
- The function symbols are often denoted by lowercase letters f, g, h,...
  - each predicate symbol has some arity ≥ 0
  - function symbols of arity 0 are called constant symbols, and are often denoted by lowercase letters at the beginning of the alphabet a, b, c,...

arity is the number of arguments

First-order Predicate Logic – Syntax of Terms

The set of terms is recursively defined by the following rules:

1. Any variable is a term,
2. Any constant symbol is a term,
3. If *f* is a function symbol of arity *n* and $t_1,\ldots,t_n$ *are terms, then* $(t_1,\ldots,t_n)$ is a term
4. nothing else is a term

First-order Predicate Logic – Syntax of Formulas

The set of formulae is recursively defined by the following rules:

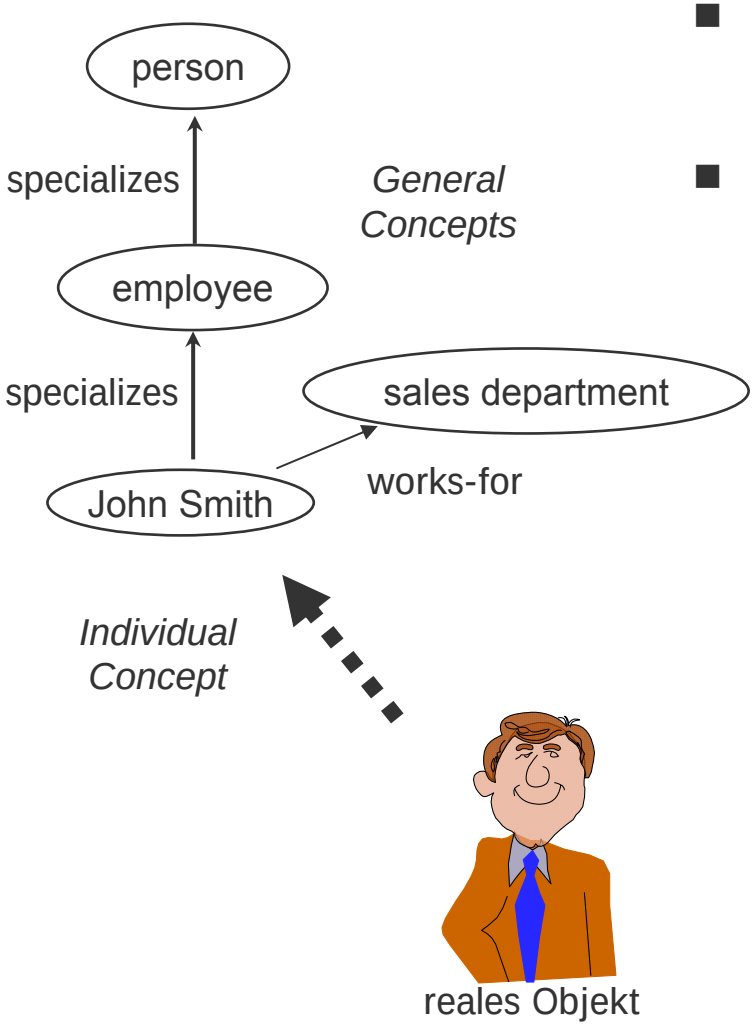
1. If P is a predicate symbol of arity n and $t_1,...,t_n$ are terms, then $P(t_1,...,t_n)$ is a formula (all these formulae are called atomic formula or atoms).
2. If A and B are formulae and the set of logical operators is $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$, then (A), $\neg A$, $A \wedge B$, $A \vee B$, $A \rightarrow B$, $A \leftrightarrow B$ are formulae.
3. If x is a variable, A is a formula and the set of quantifiers is $\{\forall,\exists\}$, then $\forall x\ A$ und $\exists x\ A$ are formulae.
4. Nothing else is a formula

In the formulae $\exists x{:}A$ and $\forall x{:}A$ the quantifiers bind the variable x. If a variable is not bound in a formula it is called a free variable.

SBVR: Conceptual Schema

- For any given business, the “universe of discourse” indicates those aspects of the business that are of interest.
- A “model,” in the sense used here, is a structure intended to describe a business domain, and is composed of
  - a conceptual *schema* (fact structure) and
  - a *population* of ground facts
- A *fact* is a proposition taken to be true by the business.
  - Instantiated roles of facts refer to individuals (such as “Employee 123”, "John Smith" or “the sales department”).
  - Individuals are considered as being of a particular type (such as “Employee” or “Department”) where *type* denotes “set of possible individuals.”
- The conceptual schema declares the *fact types* (kinds of facts, such as “Employee works for Department”) and *rules* relevant to the business domain.

The Conceptual Schema as a Semantic Net



- Two types of concepts:
  - **general concept** (class)
  - **individual concept** (instance)
- From this distinction we have at least three kinds of relations (binary **fact types**)
  - structural relations:
    - Relation between individual and general concepts (also called instance-of)

      John Smith *specializes* employee
    - Relation between individual and general concepts (also called is-a or SubClass-Of)

      employee *specializes* person

    Unfortunately, SBVR uses the same name for both kinds of structural relations
  - non-structural relations
    - arbitrary relations, e.g.

      John Smith *works-for* sales department

Conceptual Schema in Predicate Logic

- Concepts:
  - The general concepts correspond to unary predicates
  - The individual concepts correspond to constants
- Structural Relations
  - Instance-of is an atomic formula with the general concept as Predicate and the individual concept as term

    Employee(john_smith) **John Smith *specializes* employee**
  - Subclass relationship corresponds to an implication

    $\forall x$ Employee(x) $\rightarrow$ Person(x) **employee *specializes* person**
  - Binary fact types correspond to binary predicates

    works_for(john_smith, sales_department)

    John Smith *works-for* sales department

Facts

- Any fact model passes through a sequence of *states*, each of which includes a set of *ground facts.*
- *Facts* are either elementary or existential.
  - Elementary fact: declaration that an individual has a property
    - Example: Country(australia) $\wedge$ Large(australia)

      "The Country named 'Australia' is large "
  - Existential fact: assert the existence of an individual
    - Example: $\exists$x Country(x) $\wedge$ Country_code(x, us)

      "There is a Country that has the Country Code 'US' "

Static Constraints

- *Constraints* are used to define bounds, borders, or limits on fact populations, and may be static or dynamic.
- A *static constraint* imposes a restriction on what fact populations are possible or permitted, for each fact population taken individually.
- Example:
  - $\forall x \exists y \text{ Employee}(x) \wedge \text{Date}(y) \wedge \text{born\_on}(x,y)$
  - **Each** Employee was born on **at exactly one** Date

Reality model vs. in-practice model

- A *reality model* of a business domain is intended to reflect the constraints that actually apply to the business domain in the real world.
- An *in-practice model* of a business domain reflects the constraints that the business chooses in practice to impose on its knowledge of the business domain.

Suppose the following two fact types are of interest: Employee was born on Date; Employee has PhoneNumber. In the real world, each employee is born, and may have more than one phone number. Hence the reality model includes the constraint "**Each** Employee was born on **at least one** Date" and allows that "**It is possible that the same** Employee has **more than one** PhoneNumber." Now suppose that the business decides to make it optional whether it knows an employee's birth date. Suppose also that the business is interested in knowing at most one phone number for any given employee. In this case, the in-practice model excludes the reality constraint "**Each** Employee was born on **at least one** Date," but it includes the following constraint that doesn't apply in the reality model: **Each** Employee has **at most one** PhoneNumber.

First-order Predicate Logic – Interpretation

- To say whether a formula is true, we have to decide what the non-logical symbols mean.
- Interpretation: Let L be a language, i.e. the set of non-logical symbols. An interpretation consists of
  - a non-empty set D called the domain
  - for each constant in L the assignment of an element in D
  - for each n-ary function symbol in L the assignment of a mappling from D^n to D
  - for each n-ary predicate symbol in L the assignment of a relation in D^n
    (this is equivalent to a mapping of D^n into {true, false})

First-order Predicate Logic – Interpretation

- The operators and quantifiers have the following meaning
  - The truth values of the logical operators are defined by the usual truth tables
  - The formula $\exists x:F$ is true in the interpretation, if there is an assignment of x with an individual such that F is true
  - The formula $\forall x:F$ is true in an interpretation, if for every assignment of x the formula F

F	G	¬A	$A \wedge B$	$A \vee B$	$A \rightarrow B$	$A \leftrightarrow B$
true	true	false	true	true	true	true
true	false	false	false	true	false	false
false	true	true	false	true	true	false
false	false	true	false	false	true	true

First-order Predicate Logic – Models and Consequences

- An interpretation that makes a formula F true is called a model of F
- Logical consequence:
  - A formula G is a logical consequence of a formula G, if all models of F are also models of G. This is written as

$$F \models G$$

First-order Predicate Logic – Calculus

- A calculus consists of
  - a set of **axioms** (formulae representing the knowledge base)
  - a set of **infererence rules**: *syntactic* transformations which derive from a set of formulas a new formula
- Examples of Inference Rules:

Modus Ponens $\dfrac{A,\ A \rightarrow B}{B}$

Substitution $\dfrac{\forall x\ \ P}{P\ \{x/a\}}$

Modus Tolens $\dfrac{\neg B,\ A \rightarrow B}{\neg A}$

Modus Ponens with substitution $\dfrac{P\ (a),\ \forall x\colon (P(x) \rightarrow Q(x))}{Q\ (a)}$

First-order Predicate Logic – Calculus

- If a formula F can be derived from a set of formulas $F_1,\ldots,F_n$ by a sequence of inference rule applications, we write

$$F_1, \ldots, F_n \vdash F$$

(One says that there is a proof for F from $F_1,\ldots,F_n$)

- The control system that selects and applies the inference rules is called **inference procedure**.
- An inference procedure should be
  - **correct**, i.e. if $F_1, \ldots, F_n \vdash F$ then $F_1, \ldots, F_n \models F$
  - **complete**, i.e. if $F_1, \ldots, F_n \models F$ then $F_1, \ldots, F_n \vdash F$

Derivation Rules

- *Derivation rules* indicate how the population of a fact type may be derived from the populations of one or more fact types or how a type of an individual may be defined in terms of other types of individuals and fact types.
- Example 1:
  - Person1 is an uncle of Person2 **if** Person1 is a brother of **some** Person3 **who** is a parent of Person2,

$$\forall x,y,z \;\; Brother(x,y) \wedge Parent(y,z) \rightarrow Uncle(x,z)$$

- Example 2:
  - **Each** person **is a** employee **if the** person works for **a** company

$$\forall x,y \;\; Person(x) \wedge Company(y) \wedge works\_for(x,y) \rightarrow Employee(x)$$

Open/Closed World Semantics

- The closed world assumption (CWA) is the presumption that what is not currently known to be true is false.
  - Under the CAW, if a proposition cannot be proved true, it is false.
- The open world assumption (OWA) states that lack of knowledge does not imply falsity.
  - Under the OWA, if a proposition cannot be proved true and its negation cannot be proved true, the truth of the proposition is *unknown*

Open/Closed World and Negation

- The open or closed world semantics is important for negation
  - The CWA entails negation as failure: a failure to find a fact implies its negation
  - In the OWA entails full negation: lack of knowledge does not imply falsity. A proposition is false only if its negation can be proved.

Database Example for Open/Closed World

Suppose we have the following sample database with the employee number and name of each employee, as well as the cars they drive (if any):

Does it include all employees?

Employee	
empNr	*empName*
1	John Smith
2	Ann Jones
3	John Smith

Drives	
empNr	*carRegNr*
1	ABC123
2	AAA246
2	DEF001

Does it include all drivers?

- Users typically adopt the closed world assumption when interpreting data in databases.
- To decide how complete or correct the query results are, users have to take that into account knowledge how complete the data is.
- Example: Select employee number of each employee who does not drive a car

  select empNr **from** Employee **where** empNr **not in** (**select** empNr **from** Drives).
- Correctness of the result depends on whether all employees with their car registry are in the database.

Open and Closed World in a Business Domain

- The distinction between open and closed world assumption can be made on the level of the fact model, based on different criteria:
- **Domain of interest:** In modeling any given business domain, attention can be restricted to propositions of interest to that domain. If a proposition is not relevant to that domain, it is not included as a fact there.
  - In this case we do not assume missing information as false; rather we simply dismiss it from consideration.
  - Example: We decide what information about customers shall be stored in a CRM system. We can decide not to store information about his/her marital status.
- **Incomplete information:** In a given business domain we might be unable to collect all information.
  - In this case, missing information does not imply falsity.
  - Example: Assume we collect the phone number of our customers in a CRM system. If we do not find the phone number of a customer, it does not mean, that he does not have a phone number

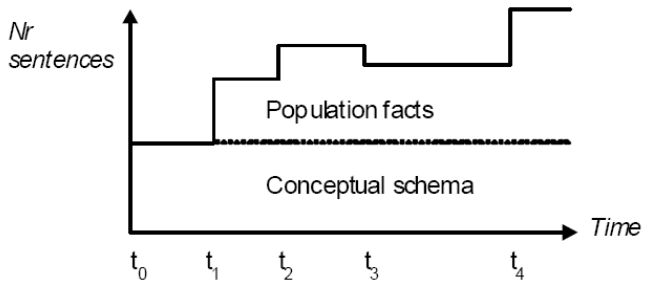
Open or closed world?

- A business might have complete knowledge about some parts and incomplete knowledge about other parts
- Thus, in practice a mixture of open and close world assumption may applied
- To cope with this situation, one might, for example,
  - assume open world semantic by default and
  - apply local closure to specific parts

  (or vice versa)
- Local closure means that for some parts of the overall DB schema the closed world assumption applies.
- Open closure can be asserted explicitly for individual and fact types, e.g.
  - **employee** ***is closed*** (all employees are known)
  - **has-name** ***is closed*** (all names of employees are known).

Changes of the Ground Fact Population

- The fact model includes both
  - the conceptual schema and
  - the ground fact population
    (set of fact instances that instantiate the fact types in the schema)
- In contrast to the conceptual schema, the (domain-specific) fact population is typically highly variable.
- In treating a fact model as a set of facts that typically changes over time, we allow facts to be added or deleted



- We might delete a fact
  - because we revise our decision on whether it is (taken to be) true (e.g. correcting a failure)
  - or because we decide that a fact is no longer of interest

Static Constraints and Changing Populations

- If we allow deletion or changes of knowledge, it may occur that previous inferences become invalid.
- To avoid this problem, static constraints are true for each state of the fact model.
- Similar, a derivation rule is applied at a single state.
- If the fact model changes there is a new state, for which the constraints and derivation rules are applied again without regard of previous states.
- Example:
  - Assume that customers get a discount if their shopping exceeds 1'000 Fr. within 12 months
  - The calculation of the discount changes as soon as a shopping is made such that 1'000 Fr. are reached and may be reduced again if the customer does not buy enough within 12 months.

Dynamic Constraints

- A *dynamic constraint* imposes a restriction on transitions between fact populations.
  - A person's marital status may change from single to married, but not from divorced to single

Dynamic constraints compare one state to another state. The formal semantics of dynamic constraints is not defined in SBVR 1.0

Quantifiers in SBVR

In addition to ∀ and ∃ SBVR supports numeric quantifiers:

Symbol	*Example*	*Name*	*Meaning*
∀	$\forall x$	Universal Quantifier	For each and every *x*, taken one at a time
∃	$\exists x$	Existential Quantifier	At least one *x*
$\exists^1$	$\exists^1 x$	Exactly-one quantifier	There is exactly one (at least one and at most one) *x*
$\exists^{0..1}$	$\exists^{0..1} x$	At-most-one quantifier	There is at most one *x*
$\exists^{0..n}$ ($n \geq 1$)	$\exists^{0..2} x$	At-most-*n* quantifier	There is at most *n x* *Note: n* is always instantiated by a number ≥ 1. So this is really a set of quantifiers (*n* = 1, etc.)
$\exists^{n..}$ ($n \geq 1$)	$\exists^{2..} x$	At-least-*n* quantifier	There is at least *n x* *Note: n* is always instantiated by a number ≥ 1. So this is really a set of quantifiers (*n* = 1, etc.)
$\exists^{n}$ ($n \geq 1$)	$\exists^{2} x$	Exactly-*n* quantifier	There is at exactly (at least and at most) *n x* *Note: n* is always instantiated by a number ≥ 1. So this is really a set of quantifiers (*n* = 1, etc.)
$\exists^{n..m}$ ($n \geq 1, m \geq 2$)	$\exists^{2..5} x$	Numeric range quantifier	There is at least *n* and at most *m x*

Definition of Additional Quantifiers

- The additional existential quantifiers can easily be defined in terms of the standard quantifiers
- Example:

$$\forall y\ \exists^2 x\ \ Parent(x,y)$$

is equivalent to

$$\forall y\ \exists x_1\ \exists x_2\ \ (Parent(x_1,y) \wedge Parent(x_2,y) \wedge x_1 \neq x_2 \wedge \forall x\ (Parent(x,y) \rightarrow (x = x_1 \vee x = x_2)))$$

Modalities

- In SBVR every constraint has an associated modality
- Alethic modality

□	necessity
◊	possibility

- Deontic modality

O	obligation
P	permission
F	forbidden

Interpretation of Alethic Modality

- If no modality is explicitly specified, an alethic modality of necessity is often assumed:

  C1 Each **Person** ***was born in*** at most one **Country**

- may be explicitly verbalized with an alethic modality

  C1' It is necessary that each **Person** ***was born in*** at most one **Country**

- For the model theory, we omit the necessity operator from the formula. The version without modal operator can be represented in standard predicate logic

Interpretation of Deontic Modality

- SBVR recommends that deontic modality is always declared explictly

  It is obligatory that each **Person** ***is a husband of*** at most one **Person.**

  or It is forbidden that a **Person** ***is a husband of*** more than one **Person**

- Deontic Modality can be represented in Predicate Logic:
  1. Normalize the formula by moving the modal operator to the front
  2. Replace the modal operators by predicates at the business domain level (e.g. forbidden).

- Example:

  It is forbidden that a **car driver** ***is less than*** **18 years**

  can be represented as

  $$\forall x\ \forall y\ (\text{Car\_driver}(x) \wedge \text{Age}(x,y) \wedge y < 18 \rightarrow \text{forbidden}$$

- In the Structured Englisch verbalization, forbidden is a modal operator while in the logic representation forbidden is a predicate. This predicates is treated like any other predicate, except that it has a reserved name.

Negation Rules for Alethic Modalities

Modality		Modal Formula		applying modal negation rules ... = (Logically Equivalent) Modal Formula	
		Formula	Reading (Verbalized as):	Formula	Reading (Verbalized as):
alethic	necessity	$\Box p$	It is necessary that *p*	$\sim\Diamond\sim p$	It is not possible that not *p*
	the negation of necessity: **non-necessity**	$\sim\Box p$	It is not necessary that *p*	$\Diamond\sim p$	It is possible that not *p*
	possibility	$\Diamond p$	It is possible that *p*	$\sim\Box\sim p$	It is not necessary that not *p*
	the negation of possibility: impossibility	$\sim\Diamond p$	It is not possible that *p* It is impossible that *p*	$\Box\sim p$	It is necessary that not *p*
	contingency	$\Diamond p \ \& \sim\Box p$	It is possible but not necessary that *p*	$\sim(\sim \Diamond p \ \vee \Box p)$	It is neither impossible nor necessary that *p*

Other transformation rules: $\forall x \Box Fx \equiv \Box \forall x Fx$

$\exists x \Diamond Fx \equiv \Diamond \exists x Fx$

Negation Rules for Deontic Modalities

Modality		Modal Formula		applying modal negation rules ... = (Logically Equivalent) Modal Formula	
		Formula	Reading (Verbalized as):	Formula	Reading (Verbalized as):
deontic	obligation	**O**p	It is obligatory that *p*	~***P***~*p*	It is not permitted that not *p*
	the negation of obligation: non-obligation	~**O**p	It is not obligatory that *p*	***P***~*p*	It is permitted that not *p*
	permission	**P**p	It is permitted that *p*	~***O***~*p*	It is not obligatory that not *p*
	the negation of permission: prohibition	~**P**p **F**p	It is not permitted that *p* It is prohibited that *p* It is forbidden that *p*	***O***~*p*	It is obligatory that not *p*
	optionality	***P****p* & ~***O****p*	It is permitted but not obligatory that *p*	~ (~***P****p* ∨ ***O****p*)	It is neither prohibited nor obligatory that *p*

Modalities and Rule Enforcement

- Rules often have just one modal operator.
- These rules can be transformed so that the modal operator is moved to the front using negation and tranformation rules
  - The rule is "tagged" with the modality of the main operator
- The impact of tagging a rule as a necessity or obligation is on the rule enforcement policy.
  - Enforcement of a necessity rule should never allow the necessity rule to be violated.
  - Enforcement of an obligation rule should allow states that do not satisfy the obligation rule
    (the precise action to be taken in that case is not specified in SBVR, as it is out of scope)

Semantics of Modal Theories

- We interpret formulae with modal operators in terms of *possible world semantics*.
- With respect to a *static constraint* declared for a given business domain, a possible world corresponds to a *state of the fact model* that might exist at some point in time
- Alethic Modality
  - A proposition is necessarily true if and only if it is true in all possible worlds.
  - A proposition is possible if and only if it is true in at least one possible world. A proposition is impossible if and only if it is true in no possible world (i.e., it is false in all possible worlds).
- Deontic Modality
- A model is an interpretation where each *non-deontic* formula evaluates to true, and the model is classified as a *permitted model* if the *p* in each deontic formula (of the form ***O****p*) evaluates to true, otherwise the model is a *forbidden model* (though it is still a model).