# 1. Matters of Method

In science, the method of verification is the outermost line of defense against error. It may be crude as long as it can be made objective. Designed for a particular theory (or kind of theories), the verification method should interact with its theory in such a way that there constantly arise new questions of a kind (i) which can be decided more or less conclusively by the method of verification, and (ii) the answers to which are relevant to the theory's further development.

In natural science, verification consists in experiments which are (i) specified exactly in quantitative terms and (ii) which can be repeated by anybody anywhere. This requires that the notions and structures of the theory are so precise that they are suitable for the scientific setup of experiments. For the grammatical analysis of language, however, the quantitative verification method happens to be unsuitable.

The method we propose instead consists in building a functional model of natural language communication. This requires (i) a declarative specification in combination with an efficiently running implementation (prototype of a talking robot), (ii) establishing objective channels of observation, and (iii) equating the adequacy of the robot's behavior with the correctness of the theory – which means that the robot must have (iv) the same kinds of external interfaces as humans, and process language in a way which is (v) input/output-equivalent with the language processing of humans.

## 1.1 Sign- or Agent-Oriented Analysis of Language?

A natural language manifests itself in the form of signs, the structures of which have evolved as conventions within a language community. Produced by cognitive agents in the speaker mode and interpreted by agents in the hearer mode, these signs are used for the transfer of content from the speaker to the hearer. Depending on whether the scientific analysis concentrates on the isolated signs or on the communicating agents, we may distinguish between *sign-oriented* and *agent-oriented* approaches.[1]

Sign-oriented approaches like Generative Grammar, Truth-Conditional Semantics, and Text Linguistics analyze expressions of natural language as objects, fixed on paper, magnetic tape, or by electronic means. They abstract away from the aspect of communication and are therefore neither intended nor suitable to model the speaker

[1] Clark (1996) distinguishes between the *language-as-product* and *language-as-action* traditions.

and the hearer mode. Instead, linguistic examples, isolated from the communicating agents, are analyzed as hierarchical structures which are formally based on the principle of *possible substitutions*.

The agent-oriented approach of Database Semantics (DBS), in contrast, analyzes signs as the result of the speaker's language production and as the starting point of the hearer's language interpretation. Inclusion of the agents' production and interpretation procedures requires a time-linear analysis which is formally based on the principle of *possible continuations*.

The goal of Database Semantics is a theory of natural language communication which is complete with respect to function and data coverage, of low mathematical complexity, and is suitable for an efficient implementation on the computer. The central question of Database Semantics is:

**How does communicating with natural language work?**

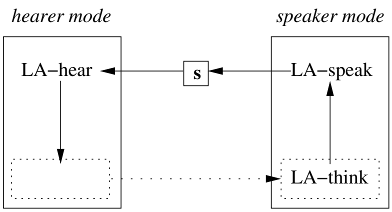
In the most simple form, this question is answered as follows.

Natural language communication takes place between cognitive agents. They have real bodies "out there" in the world with external interfaces for nonverbal recognition and action at the context level, and verbal recognition and action at the language level. Each agent contains a database in which contents are stored. These contents consist of the agent's knowledge, its memories, current recognition, intentions, plans, etc.

The cognitive agents can switch between the speaker and the hearer mode (turn-taking).[2] In a communication procedure, an agent in the speaker mode codes content from its database into signs of language which are realized externally via the language output interface. These signs are recognized by another agent in the hearer mode via the language input interface, their content is decoded, and is then stored in the second agent's database. This procedure is successful if the content coded by the speaker is decoded and stored equivalently by the hearer.

In Database Semantics, the modeling of turn-taking is based on a special data structure in combination with the time-linear algorithm of Left-Associative Grammar (LA-grammar).[3] The algorithm is used in three variants, called LA-hear, LA-think, and LA-speak. In communication, these three LA-grammars cooperate as follows:

1.1.1 THE BASIC MODEL OF TURN-TAKING




[2] For a study of turn-taking see Thórisson (2002).

[3] For the formal definition and complexity analysis of LA-grammar as well as a detailed comparison with Phrase Structure Grammar and Categorial Grammar see FoCL'99, Part II.

In the agent shown on the right (speaker mode), LA-think selectively activates content stored in the agent's database. The activated content is mapped into surfaces of a natural language by LA-speak, which are realized as external signs (represented by the small box containing **s**). In the agent shown on the left (hearer mode), LA-hear interprets the signs, which are stored in the agent's database.

The representation of turn-taking shown in 1.1.1 may be interpreted in two ways:

1.1.2 TWO VIEWS OF TURN-TAKING

1. *Viewed from the outside:*
   Two communicating agents are observed as they are taking turns. This is represented by 1.1.1 when the two boxes are taken to be two different agents, one in the hearer and the other in the speaker mode.
2. *Viewed from the inside:*
   One communicating agent is observed as it switches between being the speaker and the hearer. This is represented by 1.1.1 when the two boxes are taken to be the same agent switching between the speaker and the hearer mode (with the dotted right-hand arrow indicating the switch).

In DBS, turn-taking is regarded as a well-defined, well-motivated computational problem, which is central to the linguistic analysis of natural language: all syntactic and semantic analysis must be integrated into turn-taking as the most basic mechanism of communication. Without it, there is only one-sided monologue as the limiting case.

## 1.2 Verification Principle

Our theory of natural language communication is developed as a functional model, presented as a *declarative specification* for an efficient computer program with associated hardware. A declarative specification describes the necessary properties of a software, such as the external interfaces, the data structure, and the algorithm. Thereby, the accidental[4] properties of an implementation, such as the choice of programming language or the stylistic idiosyncrasies of the programmer, are abstracted away from.

In contrast to an algebraic definition[5] in logic, a declarative specification is not based purely on set theory. Instead, it takes a procedural point of view, specifying the general architecture in terms of components with input and output conditions as well as the functional flow through the system. A declarative specification must be general enough to provide a solid mathematical foundation and structure, and detailed enough to permit easy programming in different environments.

A declarative specification is needed because machine code is not easily read by humans. Even programs written in a higher level programming language such as Lisp

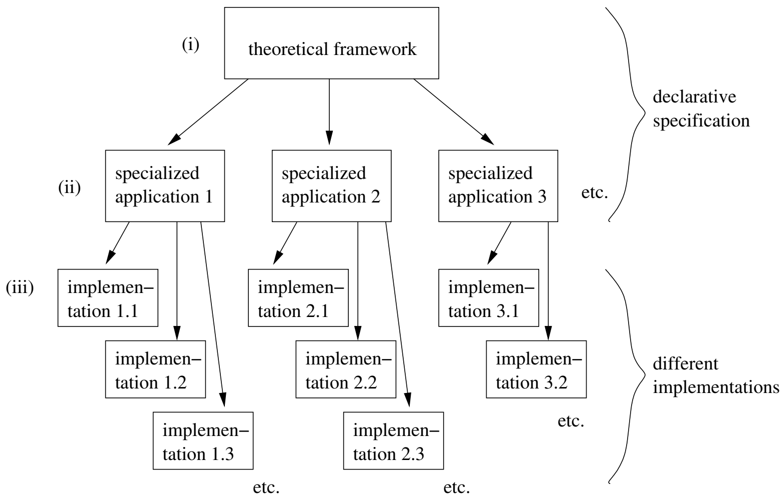
[4] The term accidental is used here in the philosophical tradition of Aristotle, who distinguishes between the necessary and the accidental (or incidental – kata sumbebêkos).

[5] The algebraic definition of LA-grammar in CoL'89 benefited greatly from help by Dana Scott.

are meaningful only to experts. What one would like to see in a piece of software is the *abstract functional solution* to the task it was designed to perform.

The declarative specification for a certain application consists of two levels: (i) a general theoretical framework (e.g., a functional system of natural language communication) and (ii) a specialization of the general framework to a specific application (e.g., English, German, Korean, or any other natural language). The theoretical framework in combination with a specialized application may in turn be realized (iii) in various different implementations, written in Lisp, C, or Java, for example.

### 1.2.1 CORRELATION OF DECLARATIVE SPECIFICATION AND IMPLEMENTATIONS




A declarative specification may have many different implementations which are equivalent with respect to the necessary properties. In Database Semantics, the evolving declarative specification must always be accompanied by at least one up-to-date implementation in order to automatically demonstrate the functioning of the theory in its current stage, and to test it with respect to an ever-widening range of various tasks. In this way, errors, incompletenesses, and other weaknesses of the current stage may be determined (explicit hypothesis formation, cf. FoCL'99, 7.2.3), which is a precondition for developing the next improved stage of the declarative specification.

The cycle of theory development and automatic testing is the *verification method* of Database Semantics. It differs from the quantitative methods of the natural sciences (repeatability of experiments) as well as the logical-axiomatic methods of mathematics (proof of consistency), though it is compatible with them.

The verification method[6] of Database Semantics is important for the following reasons. First, the signs of natural language are based on conventions which are not sus-

[6] See also FoCL'99, Introduction VIII–X.

ceptible to the quantitative methods of the natural sciences. Second, the analysis of the natural languages in linguistics and neighboring fields such as the philosophy of language is fragmented into a very large number of different schools and subschools, which raises the question of their comparative evaluation.

## 1.3 Equation Principle

Database Semantics aims at modeling the language communication of artificial agents as naturally as possible for two reasons. First, maximal *user-friendliness* should be provided in practical applications. User-friendliness in man–machine communication means that the human and the robot can understand each other (i) correctly and (ii) without the human having to adapt to the machine.[7]

Second, long-term *upscalability* in theory development should be ensured. Upscaling in the construction of a talking robot means that one can proceed without difficulty from the current prototype to one of greater functional completeness and/or data coverage.[8] In the history of science, difficulties in upscaling have practically always indicated a fundamental problem with the theory in question.[9]

To ensure user-friendliness and upscalability in the long run, Database Semantics must strive to approximate at the various levels of abstraction what has been called "psychological reality." For this purpose, we propose the following principle, which equates the correctness of the theoretical description with the behavioral adequacy of the electronic model (prototype of a talking robot).

### 1.3.1 THE EQUATION PRINCIPLE OF DATABASE SEMANTICS

1. The more realistic the reconstruction of cognition, the better the functioning of the model.

[7] For this, the robot must be designed to have procedural counterparts of human notions. For example, in order to understand the word red, the robot must be capable of physically selecting the red objects from a set; in order to understand the notion of being happily surprised, the robot must be capable of experiencing this emotion itself; etc.

Given that the technical preconditions for this kind of user-friendliness will not become available for some time, Liu (2001) proposes to integrate current robotic capabilities with practical tasks guided by humans. This is a positive example of a smart solution, like the use of restricted language in machine translation (cf. FoCL'99, p. 47).

[8] For example, functional completeness requires the ability of automatic word form recognition in principle. Extending the data coverage means that more and more word forms of the language can be recognized; similarly, functional completeness requires the ability of contextual action in principle. Extending the data coverage means that more and more contextual action types such as different kinds of locomotion, manipulation, etc., become available.

[9] Problems with upscaling in Truth-Conditional Semantics arise in the attempts to handle the Epimenides Paradox (cf. FoCL'99, Sect. 19.5), propositional attitudes (cf. ibid, Sect. 20.3), and vagueness (ibid, Sect. 20.5). Problems with upscaling in Generative Grammar arise in the attempts to handle the constituent structure paradox (ibid, Sect. 8.5) and gapping constructions (cf. Chaps. 8 and 9 below).

2. The better the functioning of the model, the more realistic the reconstruction of cognition.

The first part of the Equation Principle looks for support from and convergence with the neighboring sciences in order to improve the performance of the prototype. This means, for example, that we avoid conflicts with established facts or strong conjectures regarding the phylogenetical and the ontogenetical development as provided by ethology and developmental psychology, include the functional explanations of anatomy and physiology, and take seriously the results of mathematical complexity theory (no undecidable or exponential algorithms).

The second part of the equation principle provides a heuristic strategy in light of the fact that the "real" software structures of cognition (at their various levels of abstraction) are not accessable to direct observation.[10] Our strategy tries to achieve a realistic reconstruction indirectly by aiming for functional completeness and completeness of data coverage in the incremental upscaling of an artificial cognitive agent.

## 1.4 Objectivation Principle

For a functional reconstruction of cognition in general and natural language communication in particular, different kinds of data are available. The differences stem in part from alternative constellations in which the data originate, and in part from alternative channels which are used in the respective constellations.

The constellations regard the interaction between (i) the user, (ii) the scientist, and (iii) the electronic model (robot). They are distinguished as follows:

### 1.4.1 Constellations providing different kinds of data

1. Interaction between (i) the user and (iii) the robot
2. Interaction between (i) the user and (ii) the scientist
3. Interaction between (ii) the scientist and (iii) the robot

Depending on the constellation, data can be transmitted via the following channels:

### 1.4.2 Data channels of communicative interaction

1. The *auto-channel* processes input automatically and produces output autonomously, at the context as well as the language level. In natural cognitive agents, i.e., the user and the scientist, the auto-channel is present from the very beginning in its

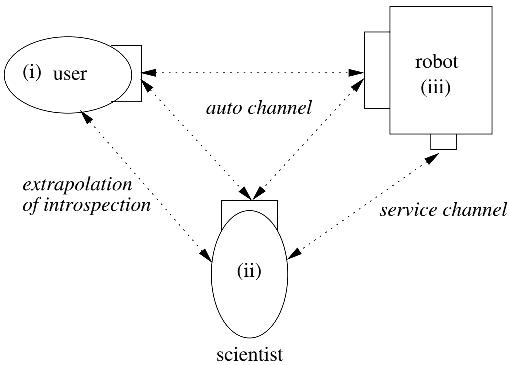
[10] A notable exception is the direct study of central cognition in neurology, especially fMRI or functional magnetic resonance imaging (cf. Matthews et al. 2003; Jezzard et al. 2001). Currently, however, these data leave room for widely differing interpretations, and are used to support conflicting theories.

full functionality. In artificial agents, in contrast, the auto-channel must be reconstructed – and it is the goal of Database Semantics to reconstruct it as realistically as possible.

2. The *extrapolation of introspection* is a specialization of the auto-channel and results from the scientists' effort to improve man–machine communication by taking the view of the human user. This is possible because the scientist and the user are natural agents.
3. The *service channel* is designed by the scientist for the observation and control of the artificial agent. It allows direct access to the robot's cognition because its cognitive architecture and functioning is a construct which in principle may be understood completely by the scientist.

The three constellations and the role of the three data channels in the interaction between user, scientist, and robot may be summarized graphically as follows:

1.4.3 INTERACTION BETWEEN USER, ROBOT, AND SCIENTIST




The scientist observes the external behavior of the user and the robot via the auto-channel, i.e., the scientist sees what they do and can also interview them about it. In addition, the scientist observes the cognitive states of (a) the user indirectly via a scientifically founded extrapolation of introspection and (b) the robot directly via the service channel. For the scientist, the user and the robot are equally real agents "out there" in the world, and their cognitive states have the same ontological status.

Of the three channels, the auto-channel is available to the user, the robot, and the scientist. It is the channel used most, but it is also most prone to error: At the level of context there are the visual illusions, for example, and at the level of language there are the misunderstandings. In addition, one has to take into account the possibility that the partner of discourse might deviate from the truth, either consciously or unconsciously.

As long as everyday access to the partner of discourse is restricted to the auto-channel, we can never be completely certain whether what we said was really understood as intended by us, or whether we really understood what was intended by the other, or whether what was said was really true. In philosophy, this is the much discussed problem known as *solipsism* (Wittgenstein 1921).

For a scientific analysis of natural language communication, however, there are the priviledged accesses of (i) the extrapolation of introspection and (ii) the service channel. In the extrapolation of introspection, the discourse between the scientist and the user is restricted to the domain of user–robot interaction. Therefore, misunderstandings between the scientist and the user are much less likely than in free communication, though they are still possible. The direct access to the robot via the service channel, furthermore, allows the scientist to determine objectively whether or not the cognition of the artificial agent is functioning properly. Thus, artificial cognitive agents are special insofar as they are not subject to the problem of solipsism.

## 1.5 Equivalence Principles for Interfaces and for Input/Output

The methodological principles of Database Semantics presented so far, namely

1. the *Verification Principle*
   i.e., the development of the theory in the form of a declarative specification which is continuously verified by means of an implemented prototype (cf. Sect. 1.2),
2. the *Equation Principle*
   i.e., the equating of theoretical correctness with the behavioral adequacy of the prototype during long-term upscaling (cf. Sect. 1.3), and
3. the *Objectivation Principle*
   i.e., the establishing of objective channels for observing language communication between natural and artificial agents (cf. Sect. 1.4),

are constrained by

4. the *Interface Equivalence Principle*, and
5. the *Input/Output Equivalence Principle.*

According to the Principle of Interface Equivalence (4), the artificial surrogate must be equipped with the same interfaces to the external world as the natural original. At the highest level of abstraction, this requires the external interfaces of recognition and action at the context and the language level (cf. 2.1.3). At lower levels of abstraction, the interfaces in question split up into the different modalities (cf. Sect. 2.2) of vision, audio, tactile, etc., for recognition, and locomotion, manipulation, etc., for action.

The Interface Equivalence between the model and the natural original is crucial for the automatic reconstruction of reference, i.e., the relation between language and the world. For example, if the robot cannot perceive, it cannot understand the human's reference to a new object in their joint task environment. The Interface Equivalence

Principle has fundamental consequences on the theory of semantics for natural language, especially the ontological foundations (cf. 2.3.1).

The Principle of Input/Output Equivalence (5) presupposes Interface Equivalence (4). Input/Output Equivalence requires that the artificial agent (i) takes the same input and produces the same output as the natural original, (ii) disassembles input and output in the same way into parts, and (iii) orders the parts in the same way during intake and discharge. The input and output data, like the external interfaces, are concretely given and therefore are susceptible to an objective structural analysis.

The Input/Output Equivalence between the model and the natural original is especially relevant for the automatic interpretation and production of the signs used in natural language communication. Therefore, this principle has fundamental consequences on the theory of grammar for natural language.

The two Equivalence Principles constitute a *minimal requirement* for any scientific reconstruction of cognition in general and the mechanism of natural language communication in particular. The reason is as follows: If we had direct access to the architecture and the functioning of cognition, comparable to the investigation of the physical structures and functions of the bodily organs in the natural sciences (anatomy, physiology, chemistry, physics), the resulting model would certainly have to satisfy the Principles of Interface Equivalence and Input/Output Equivalence.

If, due to the absence of direct access, the nature of the cognitive system must be inferred indirectly, namely in an incremental process of upscaling the functional completeness and the data coverage of an artificial surrogate, this does not diminish the importance of the external interfaces and the input/output data. On the contrary, as concretely given, directly observable structures they constitute the external fixpoints for any reconstruction of the internal cognition procedures which is scientifically well-founded.

## 1.6 Surface Compositionality and Time-Linearity

The general principles of Interface Equivalence and Input/Output Equivalence require a careful analysis and reconstruction (i) of the natural agent's recognition and action components and (ii) of the data being passed through these components. One important kind of data are the expressions of natural language produced in the speaker mode and interpreted in the hearer mode.

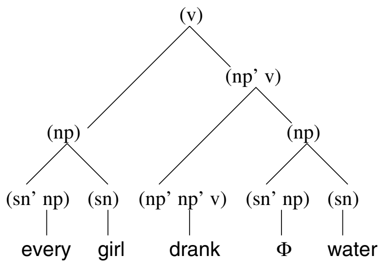
Externally, these data are objects in a certain medium, represented by sounds, handwritten or printed letters, or gestures of a sign language, which can be recorded on film, tape, or disc, and measured and described with the methods of the natural sciences. Given that these objects are concretely given, they constitute the empirical basis which linguistic analysis should neither add to nor subtract from. This elementary methodological principle is known as Surface Compositionality (SCG'84):

### 1.6.1 Surface compositionality

> A grammatical analysis is surface compositional if it uses only the concrete word forms as the building blocks of composition, such that all syntactic and semantic properties of a complex expression derive systematically from the syntactic category and the literal meaning of the lexical items.

Surface Compositionality is best illustrated by examples which violate it, such as the following grammatical analysis:

### 1.6.2 Analysis violating surface compositionality




In order to treat the noun phrases every girl and water alike, this analysis postulates the zero element Φ. The presumed "linguistic generalization" is illegitimate, however, because the postulated determiner Φ of water is not concretely given in the surface.

Nevertheless, the categories of 1.6.2 are well-motivated and defined as follows:

### 1.6.3 The categories of 1.6.2

(sn' np) = *determiner*, takes a singular noun sn' and makes a noun phrase np.
(sn) = *singular noun*, fills a valency position sn' in the determiner.
(np' np' v)= *transitive verb*, takes a noun phrase np and makes an intransitive verb (np' v).
(np) = *noun phrase*, fills a valency position np' in the verb.
(np' v) = *intransitive verb*, takes a noun phrase np and makes a (v).
(v) = *verb* with no open valency positions (sentence).

The rules generating Example 1.6.2 are based on the principle of possible substitutions, and are defined as follows:

### 1.6.4 Rules computing possible substitutions for deriving 1.6.2

(v) → (np) (np' v)
(np) → (sn' np) (sn)
(np' v) → (np' np' v) (np)
(sn' np) → every, Φ
(sn) → girl, water
(np' np' v) → drank

Each rule replaces the category on the left-hand side of the arrow by the categories on the right-hand side (top-down derivation). It is also conceivable to replace the categories on the right-hand side by the one on the left-hand side (bottom-up derivation).

Without the zero determiner postulated in 1.6.2, at least one additional rule would have to be defined. However, according to the Principle of Surface Compositionality, it is methodologically unsound to simply postulate the existence of something that is absent, but considered necessary or desirable.[11] Failure to maintain Surface Compositionality leads directly to high mathematical complexity and computational intractability.

Having determined the basic elements of linguistic analysis, i.e., the surfaces in the concretely given sign and their standard lexical analysis, let us turn to the proper grammatical relations between these basic items. The most elementary relation between the words in a sentence is their time-linear order. Time-linear means linear like time and in the direction of time (cf. Sect. 3.4).

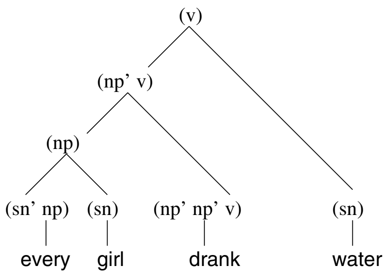
The time-linear structure of natural language is so fundamental that a speaker cannot but utter a text sentence by sentence, and a sentence word form by word form. Thereby the time-linear principle suffuses the process of utterance to such a degree that the speaker may decide in the middle of a sentence on how to continue.

Correspondingly, the hearer need not wait until the utterance of a text or sentence has been finished before his or her interpretation can begin. Instead the hearer will interpret the beginning of the sentence without having to know how it will be continued.

Example 1.6.2 violates not only Surface Compositionality, but also Time-Linearity. The grammatical analysis is not time-linear because it fails to combine every girl with drank directly. Instead, based on the principle of possible substitutions, the complex expression drank water must be derived first.

A time-linear analysis, in contrast, is based on the principle of possible continuations. As an example, consider the following time-linear derivation, which uses the same categories (cf. 1.6.3) as the non-time-linear derivation 1.6.2:

### 1.6.5 SATISFYING SURFACE COMPOSITIONALITY AND TIME-LINEARITY




This bottom-up derivation always combines a sentence start with a next word into a new sentence start, using the following (simplified) rules of Left-Associative Grammar:

[11] The inverse kind of violating Surface Compositionality consists in treating words which are concretely given in the surface as if they weren't there, simply because they are considered unnecessary or undesirable for one's "linguistic generalization." For a more detailed discussion see SCG'84 and FoCL'99, Sects. 4.5, 17.2, 18.2, and 21.3.

1.6.6 RULES COMPUTING THE POSSIBLE CONTINUATIONS FOR DERIVING 1.6.5

(VAR' X) (VAR) $\Rightarrow$ (X)
(VAR) (VAR' X) $\Rightarrow$ (X)

Each rule consists of three patterns. The patterns are built from the variables VAR, VAR', and X.[12]

The first pattern of a rule, e.g., (VAR' X), represents the sentence start *ss*, the second pattern, e.g., (VAR), the next word *nw*, and the third pattern, e.g., (X) the resulting sentence start *ss'*. The variables VAR and VAR' are restricted to a single category segment, while X is a variable for a sequence of category segments consisting of zero or more elements.

Rules computing possible continuations are based on matching their patterns with the input expressions, thereby binding their variables:

1.6.7 APPLICATION OF A RULE COMPUTING A POSSIBLE CONTINUATION

| | *ss* | | *nw* | | *ss'* | |
|---|---|---|---|---|---|---|
| *rule patterns* | (VAR' | X) | (VAR) | $\Rightarrow$ | (X) | |
| | \| | \| | \| | | \| | *matching and binding* |
| *categories* | (sn' | np) | (sn) | | (np) | |
| *surfaces* | every | | girl | | every girl | |

During matching, the variable VAR' is "vertically" bound to sn', the variable X to np, and the variable VAR to sn. In the result, the valency position sn' of the determiner category (sn' np) has been filled (or canceled), producing the *ss'* category (np), and the input surfaces every and girl are concatenated into every girl.

To handle the combination between a verb and object nouns with or without a determiner, e.g., ...drank + a coke versus ...drank + water, in a surface compositional manner, the possible values of the variables VAR and VAR' are restricted[13] and correlated as follows:

1.6.8 VARIABLE DEFINITION OF THE TIME-LINEAR RULES FOR DERIVING 1.6.5

If VAR' is sn', then VAR is sn. (*identity-based agreement*)
If VAR' is np', then VAR is np, sn, or pn. (*definition-based agreement*)

The formalism of a time-linear derivation sketched in 1.6.5–1.6.8 is of a preliminary kind. It was used in NEWCAT'86 for the automatic time-linear analysis of 221 syntactic constructions of German and 114 of English, complete with LISP source code. It was also used in CoL'89 for 421 syntactic–semantic constructions of English with a sign-oriented, hierarchical semantic analysis.

[12] There is a convention in Database Semantics that constants are written in lowercase Roman letters, while variables are written in uppercase Roman letters or in lowercase Greek letters. Cf. Appendix C, Sect. C.3.

[13] The variable restrictions for handling agreement in English are summarized in the Appendix C, C.3.4.