Extended Discourse Representation Structures in Attempto Controlled English

Technical Report ifi-2005.8

Norbert E. Fuchs[1], Stefan Hoefler[2], Kaarel Kaljurand[1]
Gerold Schneider[1], Uta Schwertel[3]

[1] Department of Informatics & Institute of Computational Linguistics, University of Zurich
Email: {fuchs,kalju,gschneid}@ifi.unizh.ch
[2] Language Evolution and Computation Research Unit, University of Edinburgh
Email: stefan@ling.ed.ac.uk
[3] Institut für Informatik, Ludwig-Maximilians-Universität München
Email: uta.schwertel@ifi.lmu.de

Abstract

This technical report describes the extended discourse representation structures (DRS) derived from texts written in version 4 of Attempto Controlled English (ACE 4).

The need to extend the standard DRS representation arose from the two requirements to adequately represent plural nouns introduced into ACE 4, and to perform logical deductions on ACE texts.

An extended representation structure contains both the original ACE 4 text and its translation into a discourse representation structure – a closed logical formula using a syntactical variant of the language of first-order logic.

The discourse representation structure itself uses a reified, or 'flat' notation, meaning that its atomic conditions are built from a small number of predefined predicates that take constants standing for words of the ACE text as their arguments. Furthermore, each logical atom gets an index relating it to the sentence of the ACE text from which it was derived.

Contents

1 Introductory Notes

This technical report describes the extended representation of discourse representation structures (DRSs) derived from version 4 of Attempto Controlled English (ACE 4). It uses illustrative ACE 4 examples, but does not describe ACE 4 itself. For a complete description of the ACE 4 language please refer to the ACE 4 Language Manual found on the Attempto web site [1]. An abstract grammar for the syntax of ACE 4 – aimed to the linguist – is provided in [3].

Please note that the current extended DRS representation deviates from DRS representations found in previous publications of the Attempto project.

We expect the reader to be familiar with the basic notions of Discourse Representation Theory (DRT) [4] as, for instance, introduced in [2].

Section 2 introduces the notation used in this report. Sections 3 to 10 describe discourse representation structures derived from declarative ACE sentences, and section 11 those derived from ACE questions.

2 Notation

The Attempto system translates an ACE text unambiguously into an extended DRS representation.

The discourse representation structure derived from the ACE text is returned as

```
drs(Domain,Conditions)
```

The first argument of `drs/2` is a list of discourse referents, i.e. quantified variables naming objects of the domain of discourse. The second argument of `drs/2` is a list of simple and complex conditions for the discourse referents. The list separator ',' stands for logical conjunction. Simple conditions are logical atoms, while complex conditions are built from other discourse representation structures with the help of the logical connectors negation '`-`', disjunction '`v`', and implication '`=>`'.

A DRS like

```
drs([A,B],[condition(A),condition(B)])
```

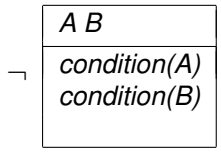
is usually pretty-printed as

A B
condition(A) *condition(B)*

The above DRS corresponds to the standard FOL representation

$$\exists AB : condition(A) \wedge condition(B)$$

Accordingly, a negated DRS like



corresponds to the standard FOL representation

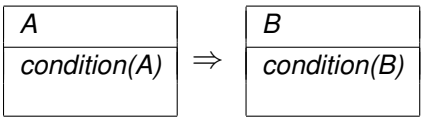
$$\neg\exists AB : condition(A) \wedge condition(B)$$

and is internally represented as

```
-drs([A,B],[condition(A),condition(B)])
```

in the Attempto system. We have defined `-/1` as a prefix operator which stands for the logical '¬'.

In a DRS, all variables are thus existentially quantified unless they stand in the restrictor of an implication. The implication



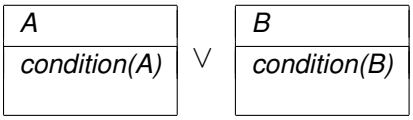
corresponding to the standard FOL representation

$$\forall A : condition(A) \rightarrow \exists B : condition(B)$$

is internally represented as

```
drs([A],[condition(A)]) => drs([B],[condition(B)])
```

The disjunction



corresponding to the standard FOL notation

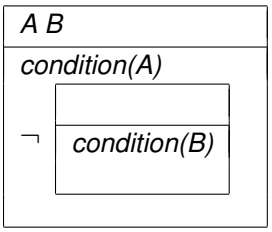
$$\exists A : condtion(A) \vee \exists B : condition(B)$$

is likewise internally represented as

```
drs([A],[condition(A)]) v drs([B],[condition(B)])
```

The predicates `=>/2` and `v/2` are defined as infix operators.

In nested discourse representation structures, a DRS can occur as an element of the conditions list of another DRS. Therefore



is represented as

```
drs([A,B],[condition(A),-drs([],[condition(B)])])
```

The discourse representation structure uses a reified, or 'flat' notation for logical atoms.

For example, the noun *a card* that customarily would be represented as

$$\exists A : card(A)$$

is represented here as

$$\exists A : object(A, card, object), ...$$

relegating the predicate 'card' to the constant 'card' used as an argument in the predefined predicate 'object'.

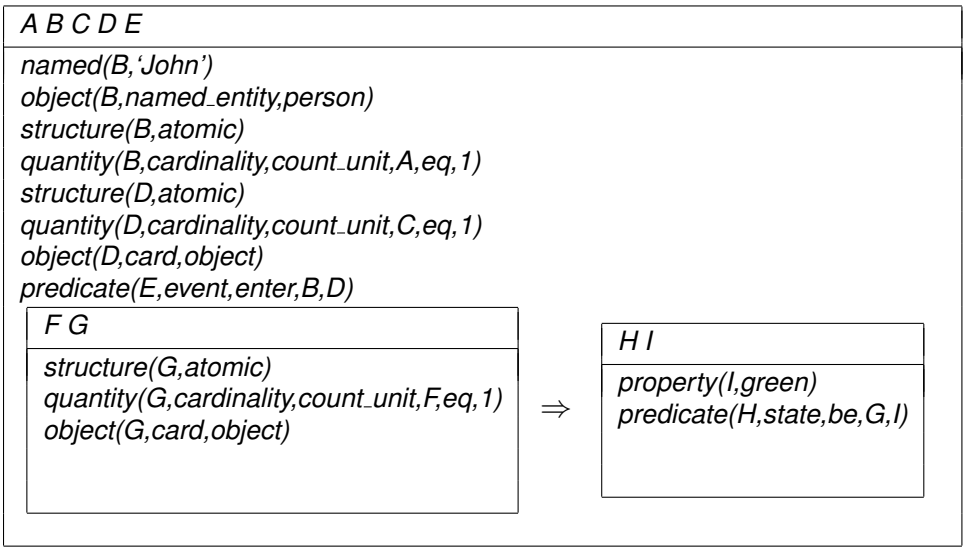
As a consequence, the large number of predicates in the customary representation is replaced by a small number of predefined predicates. This allows us to conveniently formulate axioms for the predefined predicates.

Logical atoms occurring in `drs/2` are actually written as `Atom-I` (using an infix operator `-/2`) where the number `I` refers to to the sentence from which `Atom` was derived.

The example text

```
John enters a card.  Every card is green.
```

the DRS of which is



will thus internally be represented as

```
drs([A,B,C,D,E],[named(B,'John')-1,
object(B,named_entity,person)-1,structure(B,atomic)-1,
quantity(B,cardinality,count_unit,A,eq,1)-1,structure(D,atomic)-1,
quantity(D,cardinality,count_unit,C,eq,1)-1,object(D,card,object)-1,
predicate(E,event,enter,B,D)-1,drs([F,G],[structure(G,atomic)-2,
quantity(G,cardinality,count_unit,F,eq,1)-2,
object(G,card,object)-2])=>drs([H,I],[property(I,green)-2,
predicate(H,state,be,G,I)-2])]).
```

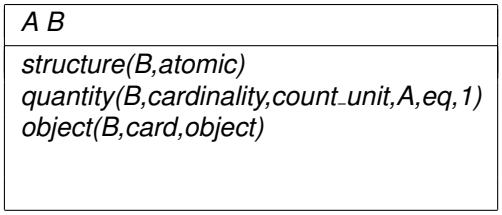
The following sections provide the discourse representation structures for a selected number of ACE 4 sentences in the form they will be output by the Attempto system, concretely by the Attempto Parsing Engine APE. Logical atoms, however, are represented without the number pointing to the sentence from which they were derived. Refer to the index at the end of this document if you want to find an explanatory DRS for a particular predicate that you saw in an ACE 4 DRS.

Using illustrative ACE 4 examples this report completely describes the language of extended DRSs derived from ACE texts. For a complete description of the ACE 4 language itself please refer to the ACE 4 Language Manual found on the Attempto web site [1].

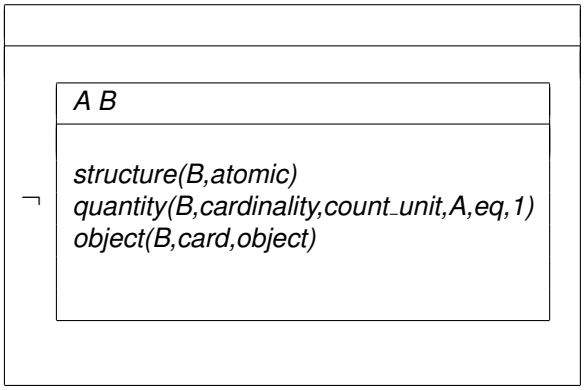
3 Noun Phrases

3.1 Singular Countable Noun Phrases

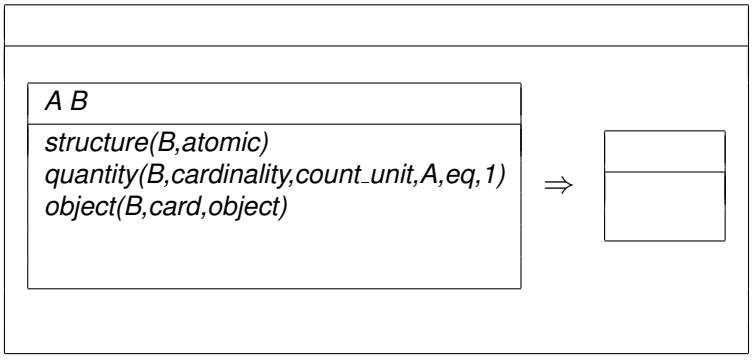
a card



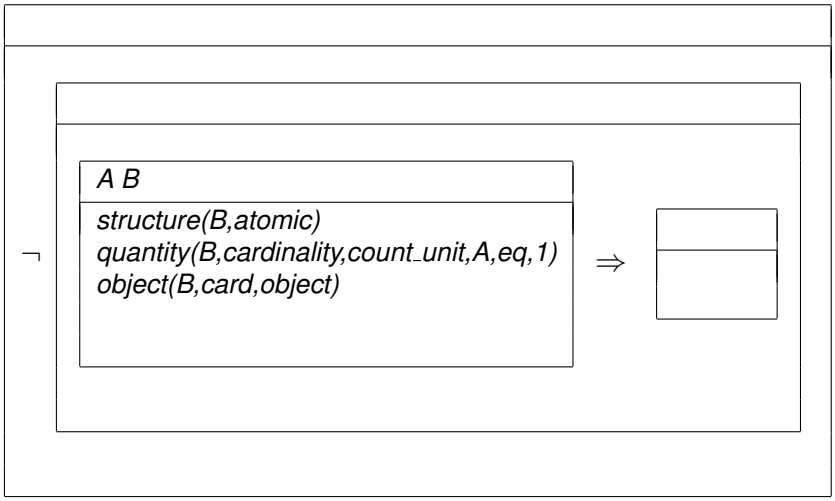
no card



every card

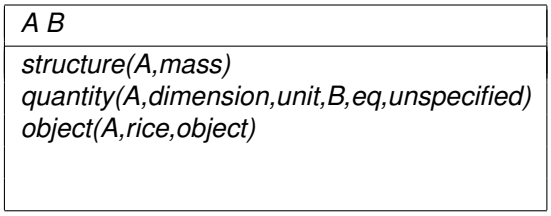


not every card

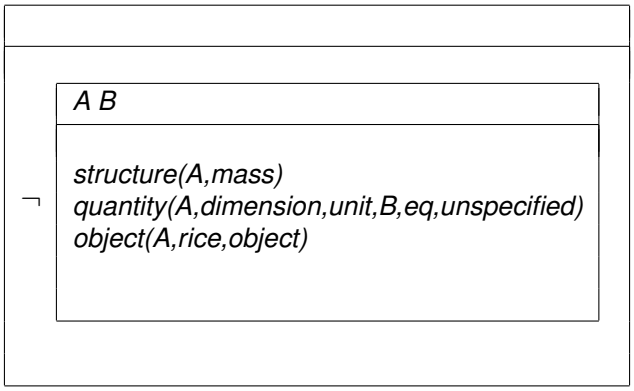


3.2 Mass Nouns

some rice

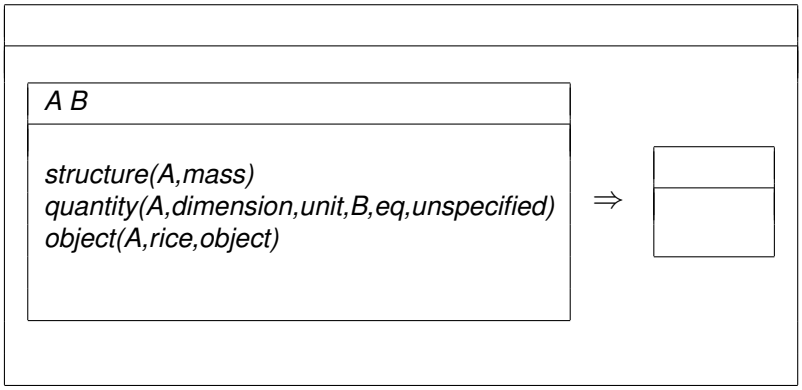


no rice

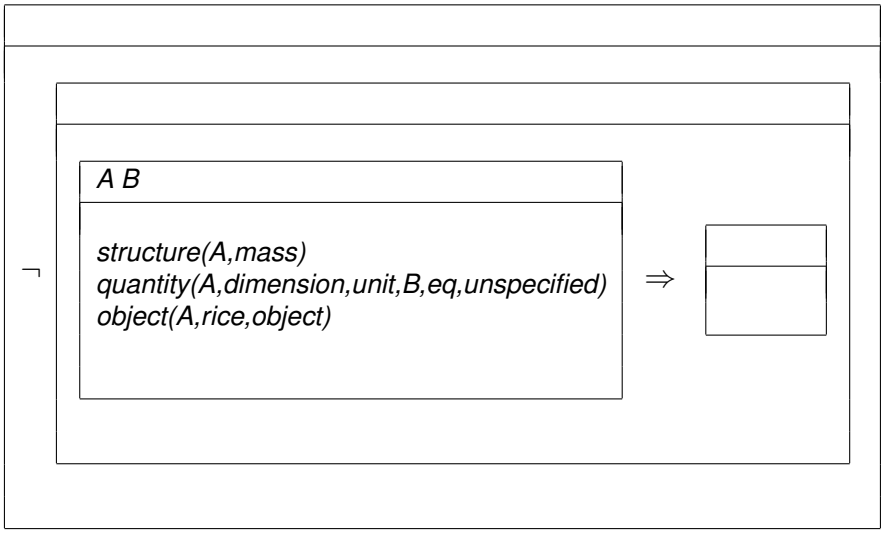


Note: the determiner *no* is ambiguous between countable and mass. For nouns that can be countable or mass, e.g. *money*, preference to countable is given. Mass reading can be forced by using sentential negation, e.g. *It is not the case that some money is omnipotent*.

all rice

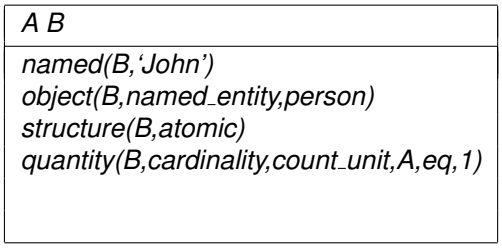


not all rice



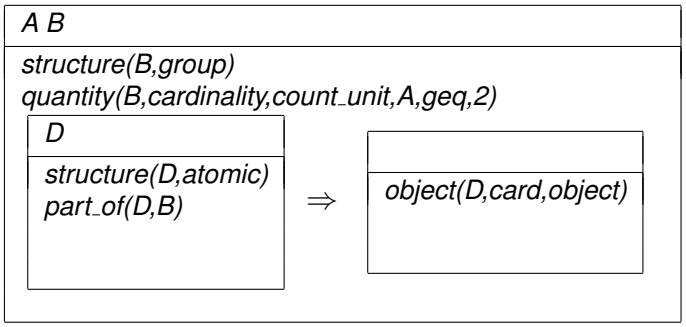
3.3 Proper Names

John

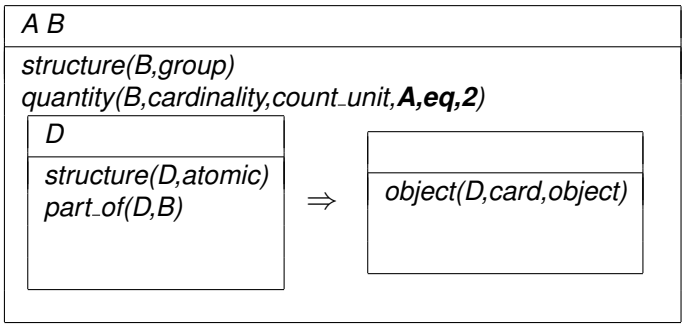


3.4 Plural Noun Phrases

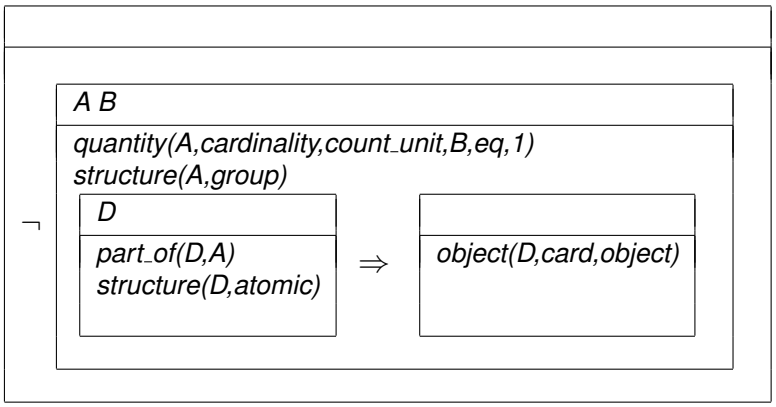
some cards



2 *cards*



no *cards*



3.5 Indefinite Pronouns

someone / somebody

A B
structure(A,dom) *quantity(A,unspecified,unspecified,B,eq,unspecified)* *object(A,unspecified,person)*

something

A B
structure(A,dom) *quantity(A,unspecified,unspecified,B,eq,unspecified)* *object(A,unspecified,object)*

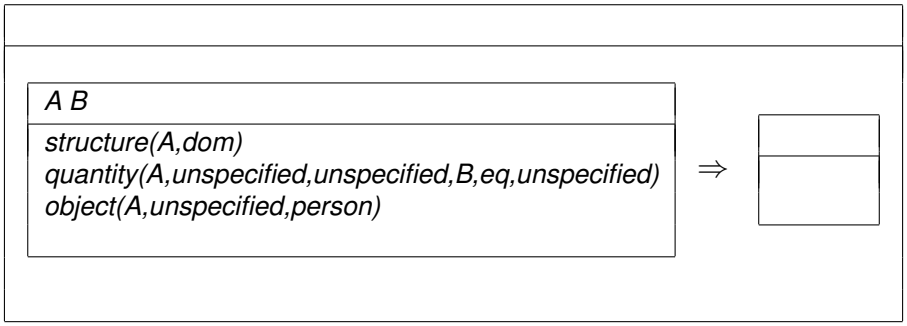
no one / nobody

¬ A B *structure(A,dom)* *quantity(A,unspecified,unspecified,B,eq,unspecified)* *object(A,unspecified,person)*

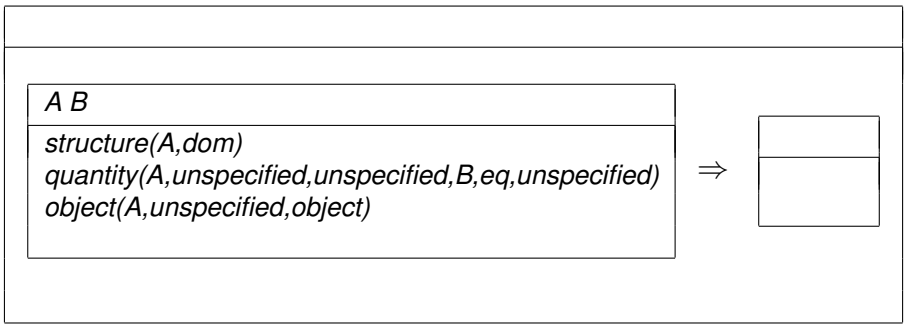
nothing

¬ A B *structure(A,dom)* *quantity(A,unspecified,unspecified,B,eq,unspecified)* *object(A,unspecified,object)*

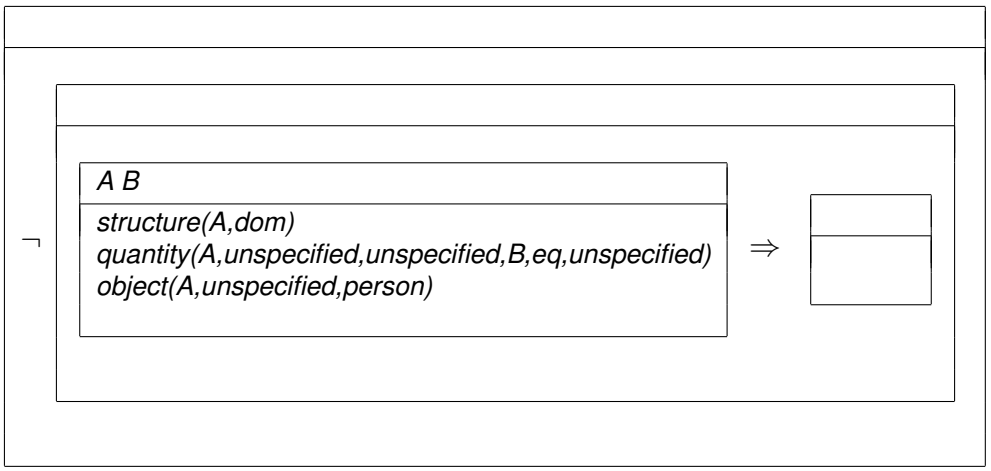
everyone / everybody



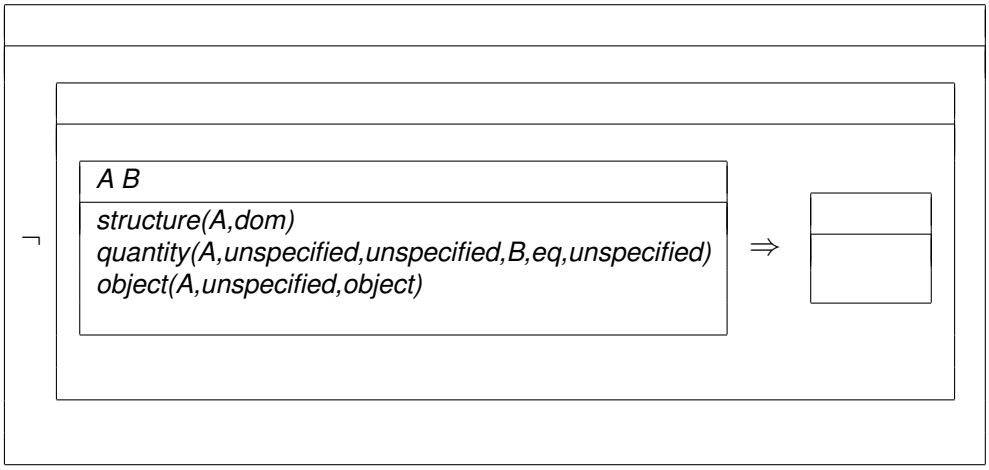
everything



not everyone / not everybody

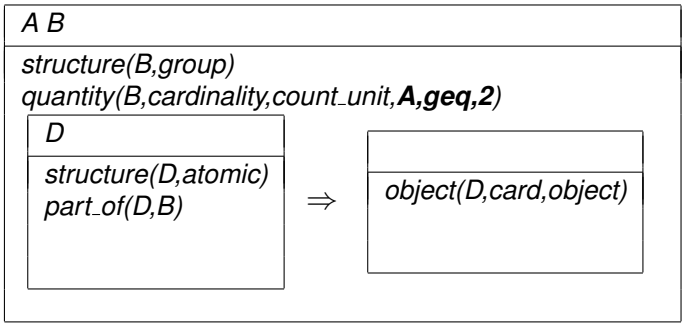


not everything

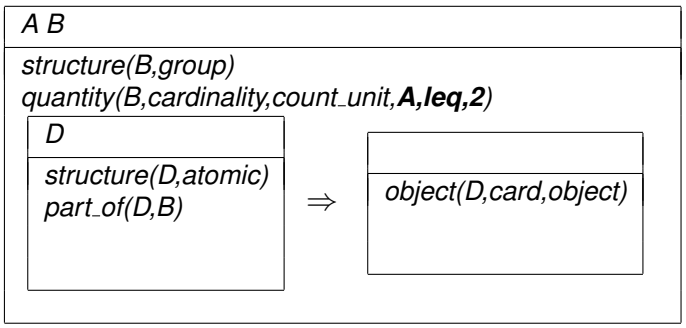


3.6 Generalised Quantors

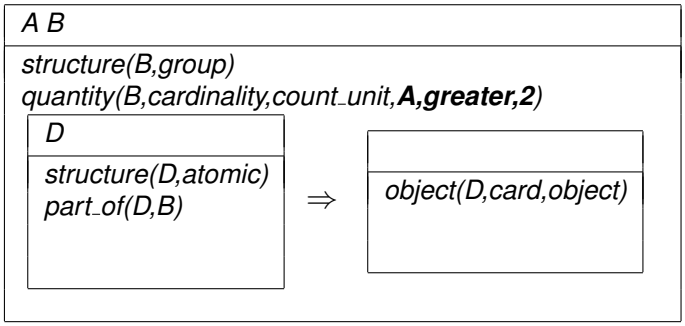
at least 2 *cards*



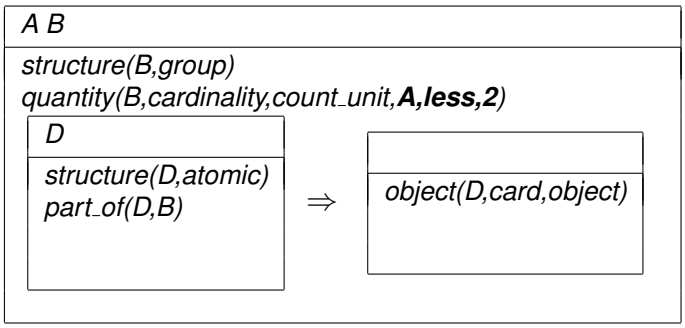
at most 2 *cards*



more than 2 *cards*

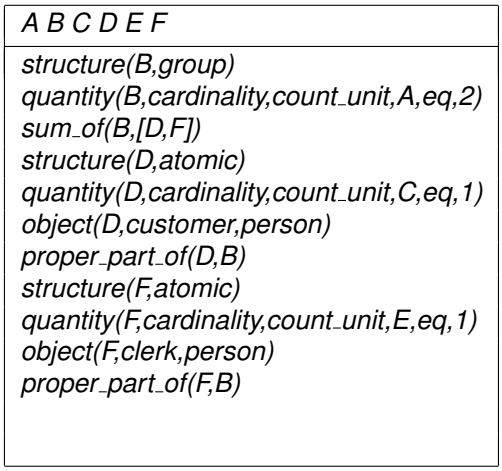


less than 2 *cards*



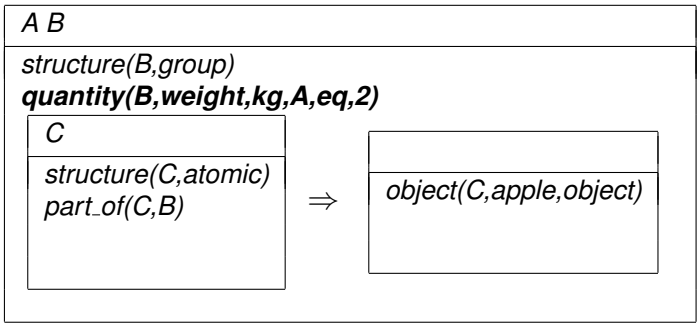
3.7 Noun Phrase Conjunction

a customer and a clerk

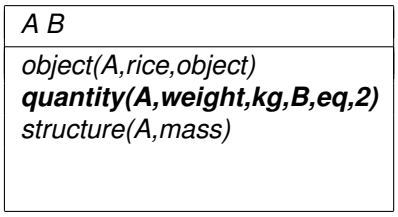


3.8 Measurement Noun Phrases

2 kg of *apples*



2 kg of *rice*

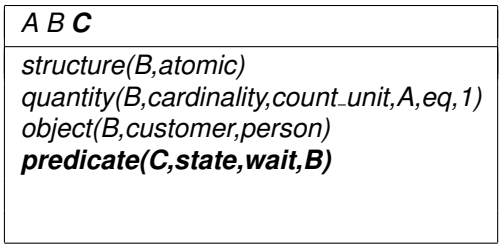


4 Verb Phrases

Note: the version of the currently used large lexicon, *clex*, does not distinguish event from state verbs. The distinction is thus unspecified.

4.1 Intransitive Verbs

A customer ***waits****.*



4.2 Transitive Verbs

*John **enters** a card.*

*A B C D **E***
named(B,'John') *object(B,named_entity,person)* *structure(B,atomic)* *quantity(B,cardinality,count_unit,A,eq,1)* *structure(D,atomic)* *quantity(D,cardinality,count_unit,C,eq,1)* *object(D,card,object)* ***predicate(E,event,enter,B,D)***

4.3 Ditransitive Verbs

Note: ditransitive verbs are a closed class.

A clerk gives a password to a customer and *A clerk gives a customer a password* lead to an identical DRS.

*A clerk **gives** a password **to** a customer.*

*A B C D E F **G***
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* *object(B,clerk,person)* *structure(D,atomic)* *quantity(D,cardinality,count_unit,C,eq,1)* *object(D,password,object)* *structure(F,atomic)* *quantity(F,cardinality,count_unit,E,eq,1)* *object(F,customer,person)* ***predicate(G,event,give_to,B,D,F)***

4.4 Copula

4.4.1 Copula and Predicative Adjectives

*A card **is valid**.*

*A B **C D***
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* *object(B,card,object)* ***predicate(C,state,be,B,D)*** ***property(D,valid)***

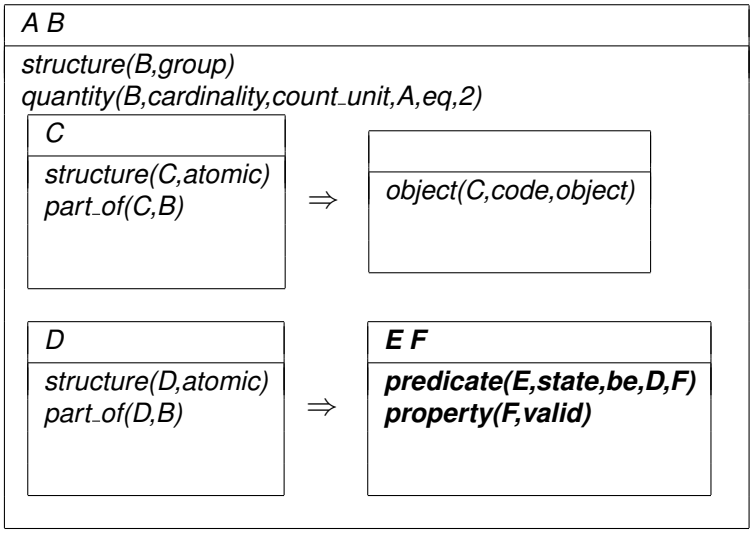
*A card **is valid and correct**.*

*A B **C D***
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* *object(B,card,object)* ***predicate(C,state,be,B,D)*** ***property(D,valid)*** ***property(D,correct)***

*2 codes **are valid**.*

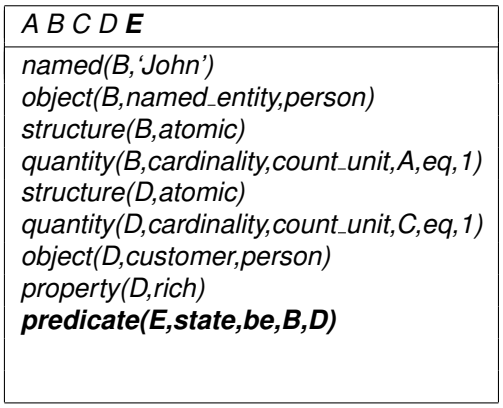
*A B **C D***
structure(B,group) *quantity(B,cardinality,count_unit,A,eq,2)* *D*: *structure(D,atomic)*, *part_of(D,B)* $\Rightarrow$ *object(D,code,object)* ***predicate(C,state,be,B,D)*** ***property(D,valid)***

*Each of 2 codes **is valid**.*



4.4.2 Copula and Noun Phrase

*John **is** a rich customer.*



4.4.3 Copula and Prepositional Phrase

*John **is** in the bank.*

*A B **C** D E*
named(B,'John') *object(B,named_entity,person)* *structure(B,atomic)* *quantity(B,cardinality,count_unit,A,eq,1)* ***predicate(C,state,be,B)*** *structure(D,atomic)* *quantity(D,cardinality,count_unit,E,eq,1)* *object(D,bank,object)* *modifier(C,location,in,D)*

4.5 Coordinated Verb Phrases

4.5.1 Verb Phrase Conjunction

A screen flashes and blinks.

A B C D
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* *object(B,screen,object)* *predicate(C,event,flash,B)* *predicate(D,state,blink,B)*

4.5.2 Verb Phrase Disjunction

A screen flashes or blinks.

A B
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* *object(B,screen,object)*

C		*D*
predicate(C,event,flash,B)	$\vee$	*predicate(D,state,blink,B)*

5 Modifying Nouns and Noun Phrases

5.1 Adjectives

5.1.1 Simple Adjectives

*A **rich** customer waits.*

A B C
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* ***property(B,rich)*** *object(B,customer,person)* *predicate(C,state,wait,B)*

5.1.2 Comparatives and Transitive Adjectives

*A customer is **richer than** John.*

A B C D E F
named(B,'John') *object(B,named_entity,person)* *structure(B,atomic)* *quantity(B,cardinality,count_unit,A,eq,1)* *structure(D,atomic)* *quantity(D,cardinality,count_unit,C,eq,1)* *object(D,customer,person)* ***property(F,richer_than,B)*** *predicate(E,state,be,D,F)*

5.1.3 Adjective Conjunction

*The **rich and old** customer waits.*

A B C
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* ***property(B,rich)*** ***property(B,old)*** *object(B,customer,person)* *predicate(C,state,wait,B)*

5.2 Appositions

5.2.1 Quoted Strings

*A customer enters the password **"Jabberwocky"**.*

A B C D E
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* *object(B,customer,person)* *structure(D,atomic)* *quantity(D,cardinality,count_unit,C,eq,1)* *object(D,password,object)* ***quoted_string(D,'Jabberwocky')*** *predicate(E,state,enter,B,D)*

5.2.2 Variables

A customer X greets a clerk. The clerk is happy. X is glad.

A *B C D E F G H I*
object(A,customer,person)
quantity(A,cardinality,count_unit,B,eq,1)
structure(A,atomic)
object(C,clerk,person)
quantity(C,cardinality,count_unit,D,eq,1)
structure(C,atomic)
predicate(E,unspecified,greet,A,C)
property(F,happy)
predicate(G,state,be,C,F)
property(H,glad)
predicate(I,state,be,A,H)

Note: Variables do not appear in the DRS. They only establish anaphoric references.

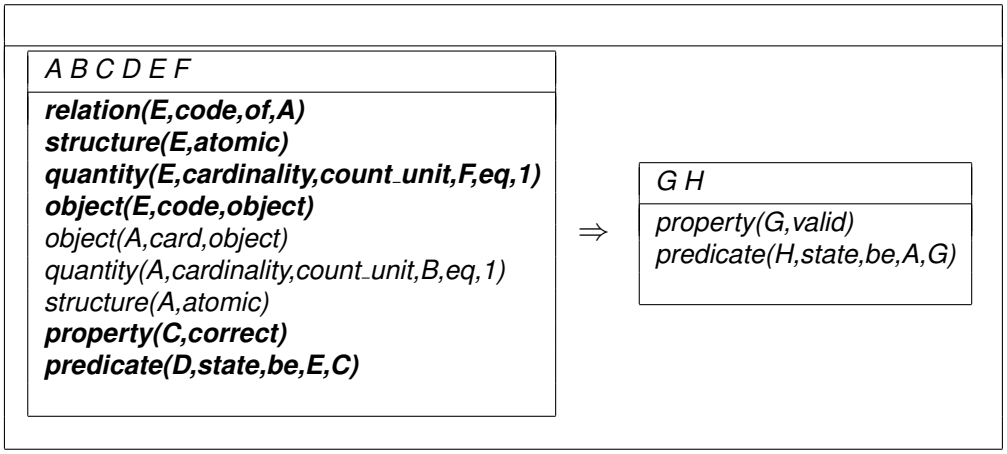
5.3 Relative Sentences

5.3.1 Simple Relative Sentences

A customer enters a card ***which is valid****.*

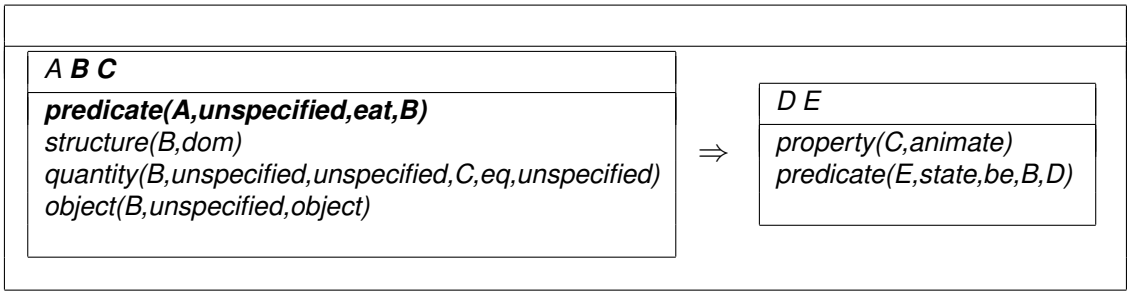
A B C D E ***F*** *G* ***H***
structure(B,atomic)
quantity(B,cardinality,count_unit,A,eq,1)
object(B,customer,person)
structure(E,atomic)
quantity(E,cardinality,count_unit,D,eq,1)
object(E,card,object)
property(H,valid)
predicate(F,state,be,E,H)
predicate(G,event,enter,B,E)

Every card ***the code of which is correct*** *is valid*



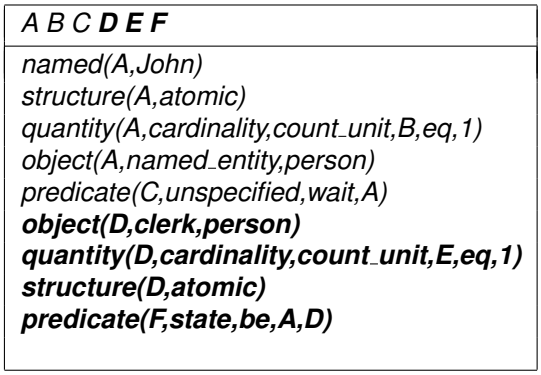
5.3.2 Relativized Indefinite Pronouns

Everything ***which eats*** *is animate.*



5.3.3 Relativized Personal Pronouns

John ***who is a clerk*** *waits.*



5.3.4 Relativized Variables

There is a card X. X ***which a customer possesses*** *is valid.*

A B C D E F G
object(A,card,object) *quantity(A,cardinality,count_unit,B,eq,1)* *structure(A,atomic)* ***object(C,customer,person)*** ***quantity(C,cardinality,count_unit,D,eq,1)*** ***structure(C,atomic)*** ***predicate(E,unspecified,possess,C,A)*** *property(F,valid)* *predicate(G,state,be,A,F)*

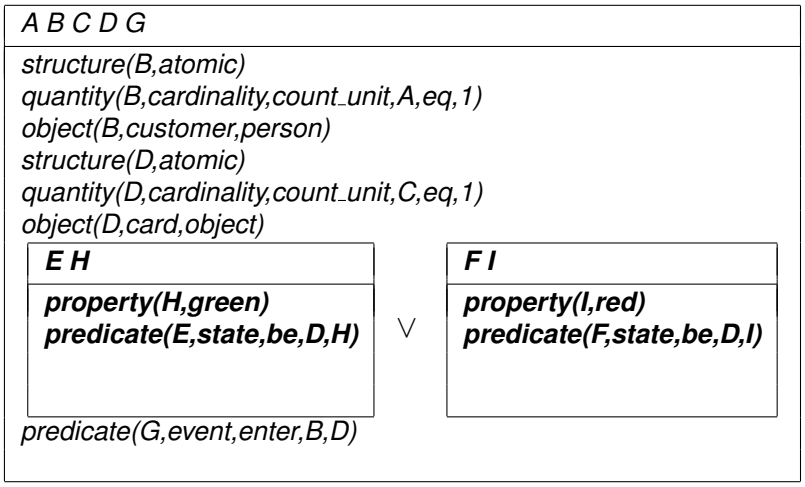
5.3.5 Relative Sentence Conjunction

A customer enters a card ***which is green and which is valid****.*

A B C D ***E F*** *G* ***H I***
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* *object(B,customer,person)* *structure(D,atomic)* *quantity(D,cardinality,count_unit,C,eq,1)* *object(D,card,object)* ***property(H,green)*** ***predicate(E,state,be,D,H)*** ***property(I,valid)*** ***predicate(F,state,be,D,I)*** *predicate(G,event,enter,B,D)*

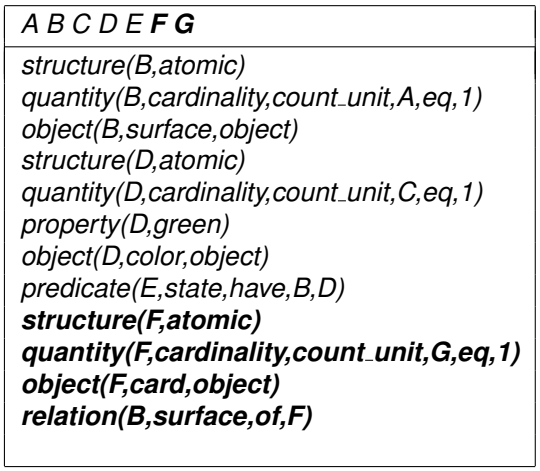
5.3.6 Relative Sentence Disjunction

A customer enters a card ***which is green or which is red****.*



5.4 *of*-Prepositional Phrases

The surface ***of the card*** *has a green color.*



5.5 Possessive Nouns

Possessive nouns are introduced by a possessive pronoun or a Saxon genitive. While possessive nouns are equivalent to *of* PPs, Saxon genitives in general are not because of the scoping rules of quantifiers:

- a man's dog (1 man with 1 dog) = a dog of a man (1 man with 1 dog)

- every man's dog (several men each with 1 dog) $\neq$ a dog of every man (1 dog of several men)

The customer's *card is valid.*

A B C ***D E*** *F*
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* *object(B,card,object)* *property(F,valid)* *predicate(C,state,be,B,F)* ***structure(D,atomic****)* ***quantity(D,cardinality,count_unit,E,eq,1)*** ***object(D,customer,object)*** ***relation(B,card,of,D)***

Note: There are no recursive Saxon genitives. "A customer's card" is in ACE, but "A customer's card's code" is not.

A customer enters a code. ***His*** *code is correct.*

A B C D E F G
object(A,customer,person) *quantity(A,cardinality,count_unit,B,eq,1)* *structure(A,atomic)* *object(C,code,object)* *quantity(C,cardinality,count_unit,D,eq,1)* *structure(C,atomic)* *predicate(E,unspecified,enter,A,C)* ***relation(C,code,of,A)*** *property(F,correct)* *predicate(G,state,be,C,F)*

6 Modifying Verb Phrases

6.1 Adverbs

*A customer **quickly** enters a card. = A customer enters a card **quickly**.*

A B C D E
structure(B,atomic) *quantity(B,cardinality,count_unit,A,eq,1)* *object(B,customer,person)* *structure(D,atomic)* *quantity(D,cardinality,count_unit,C,eq,1)* *object(D,card,object)* *predicate(E,event,enter,B,D)* ***modifier(E,manner,none,quickly)***

6.2 Prepositional Phrases

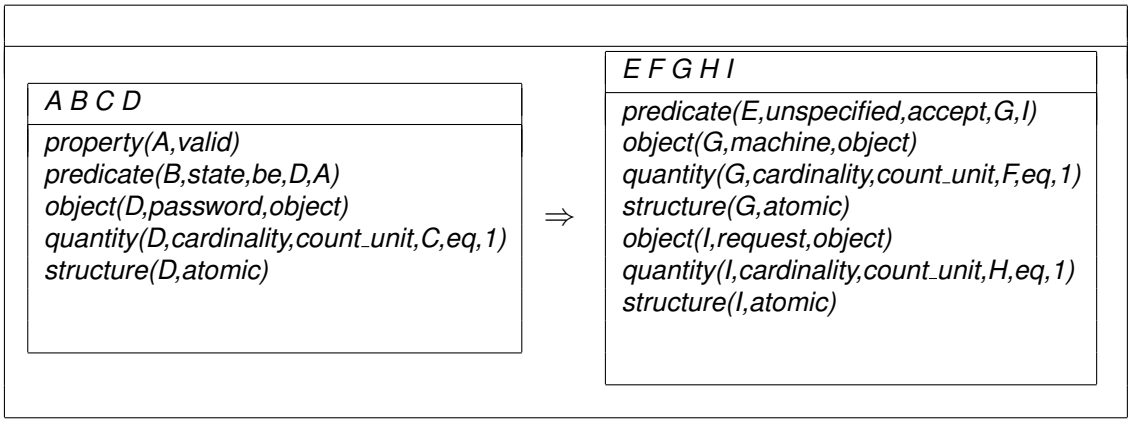
*John enters a card **in a bank**.*

*A B C D E **G F***
named(B,'John') *object(B,named_entity,person)* *structure(B,atomic)* *quantity(B,cardinality,count_unit,A,eq,1)* *structure(D,atomic)* *quantity(D,cardinality,count_unit,C,eq,1)* *object(D,card,object)* *predicate(E,event,enter,B,D)* ***structure(G,atomic)*** ***quantity(G,cardinality,count_unit,F,eq,1)*** ***object(G,bank,object)*** ***modifier(E,location,in,G)***

7 Composite Sentences

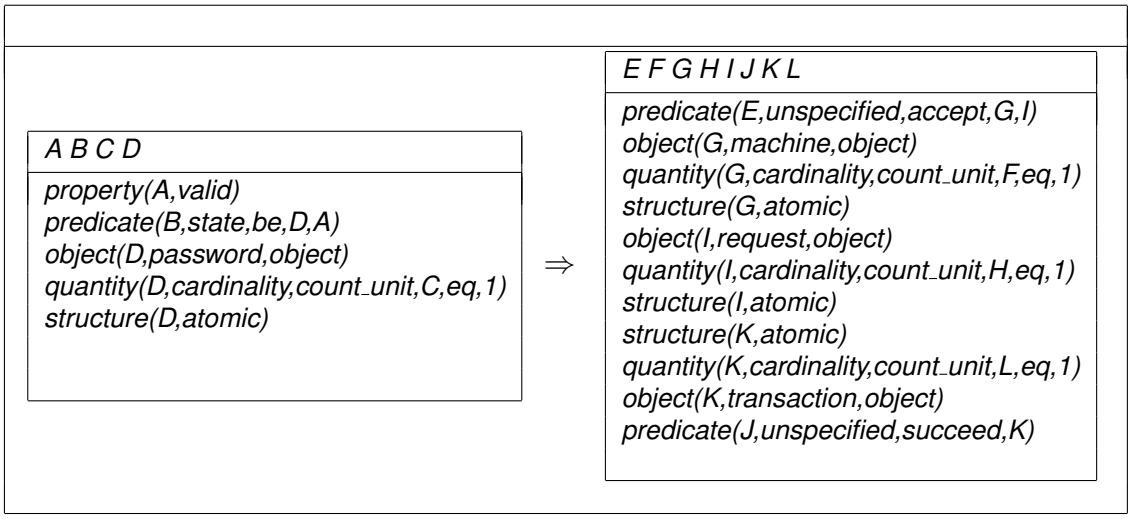
7.1 Conditional Sentences

If the password is valid then the machine accepts the request.

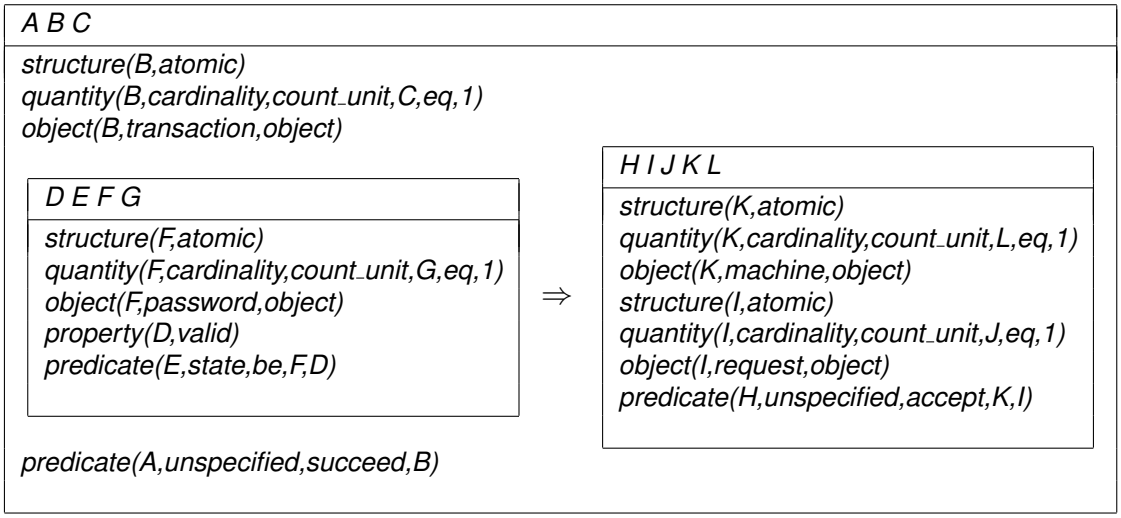


Conditional sentences always take wide scope. Narrow scope requires starting a new sentence.

If the password is valid then the machine accepts the request and the transaction succeeds.



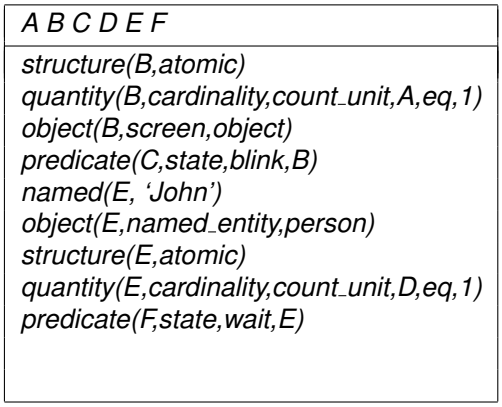
If the password is valid then the machine accepts the request. The transaction succeeds.



7.2 Coordinated Sentences

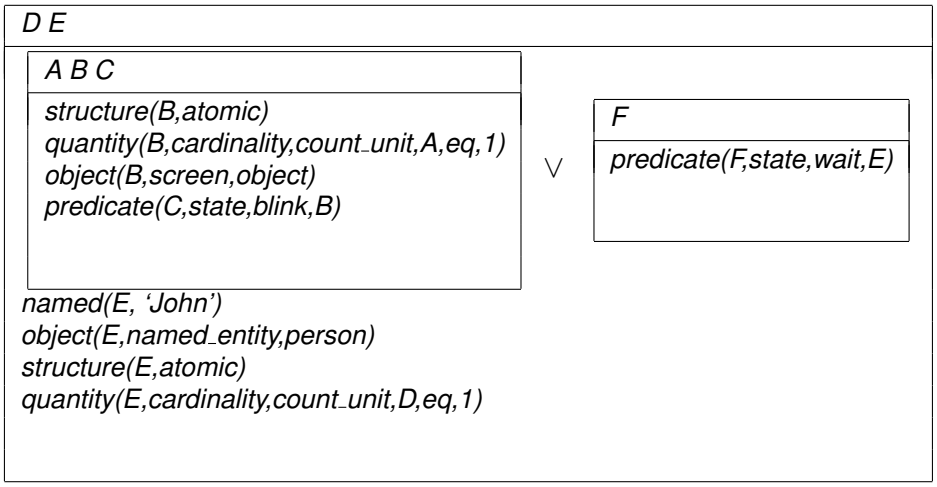
7.2.1 Sentence Conjunction

The screen blinks and John waits.



7.2.2 Sentence Disjunction

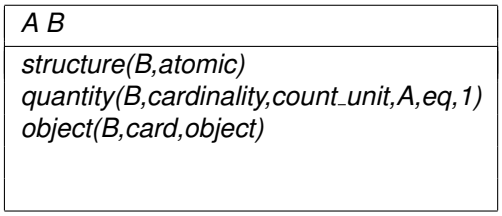
A screen blinks or John waits.



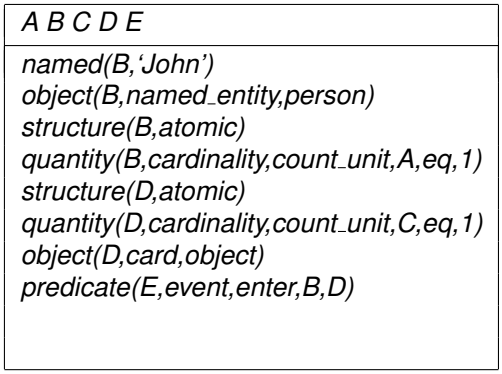
8 Quantified Sentences

8.1 Existential Quantification

A card ⇔ There is a card.

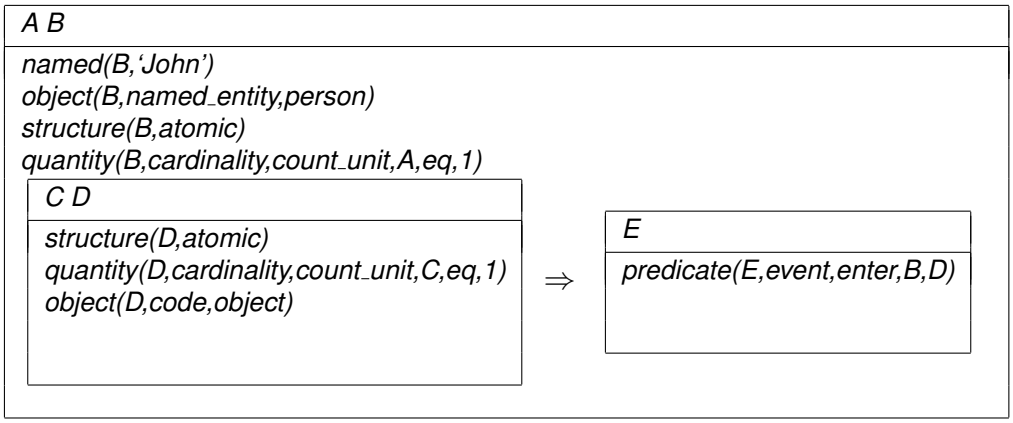


John enters a card.



8.2 Universal Quantification

John enters every code.

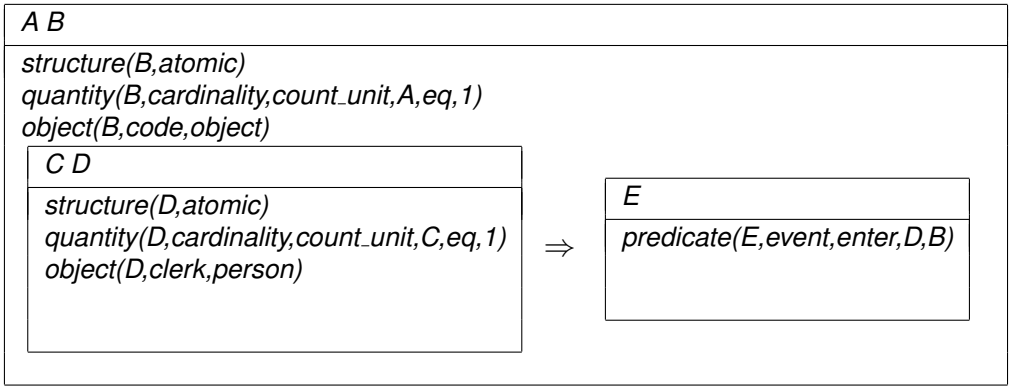


8.3 Global Quantification

8.3.1 Global Existential Quantification

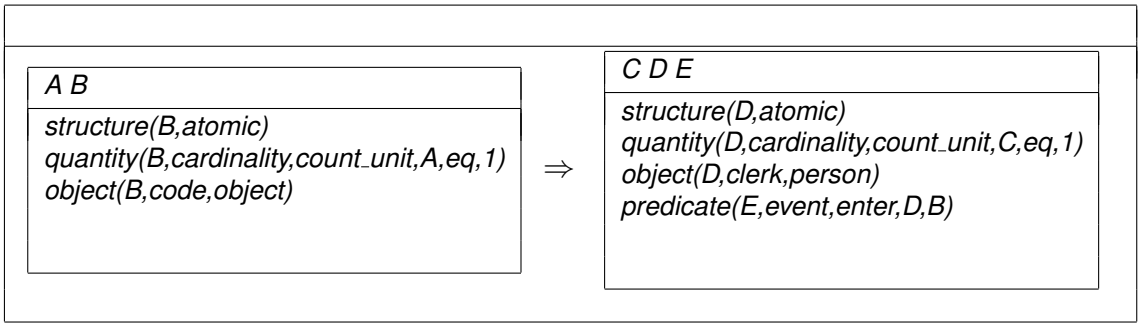
There is a code such that every clerk enters it and *There is a code that every clerk enters* lead to an identical DRS.

***There is** a code **such that** every clerk enters it.*



8.3.2 Global Universal Quantification

For every *code (there is) a clerk (such that he) enters it.*

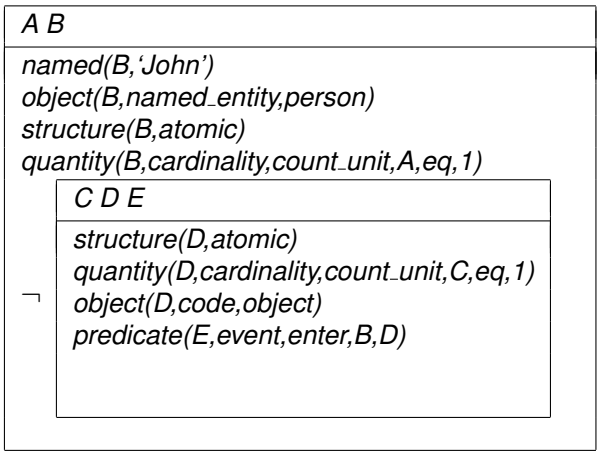


9 Negation

9.1 Quantor Negation

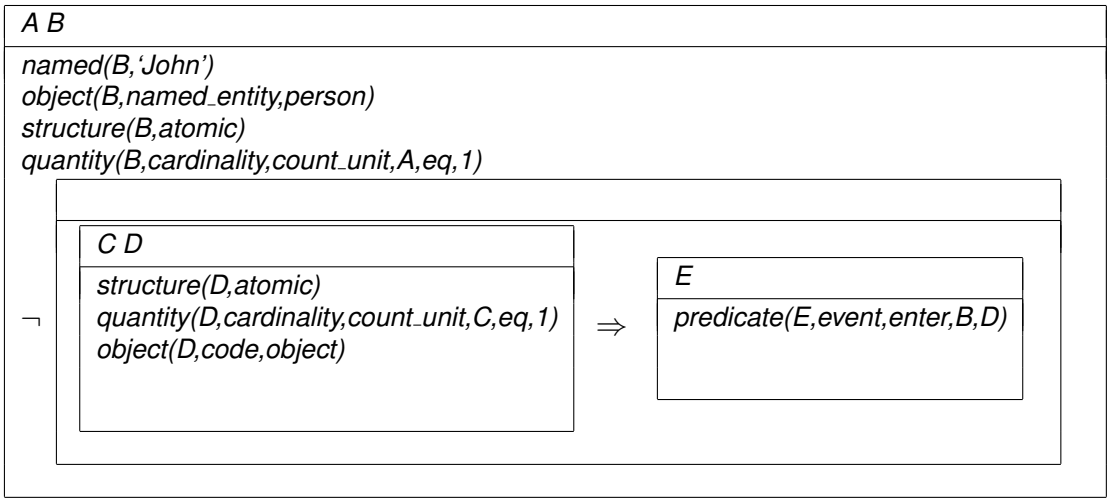
9.1.1 Negated Existential Quantor

John enters no code.



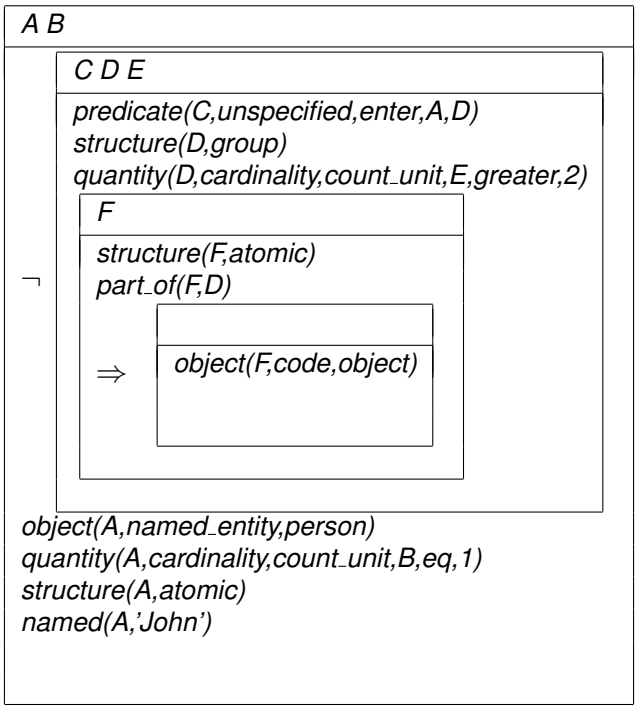
9.1.2 Negated Universal Quantor

John enters not every code.

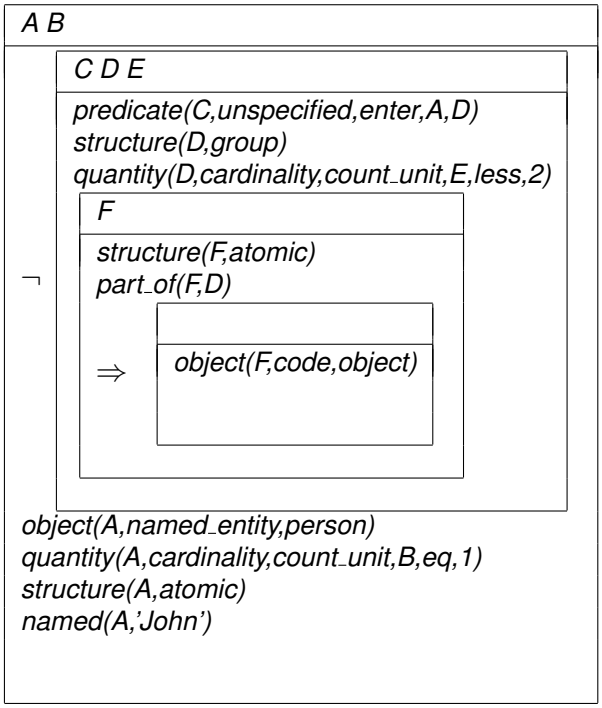


9.1.3 Negated Generalised Quantors

John enters ***not more than 2*** *codes.*



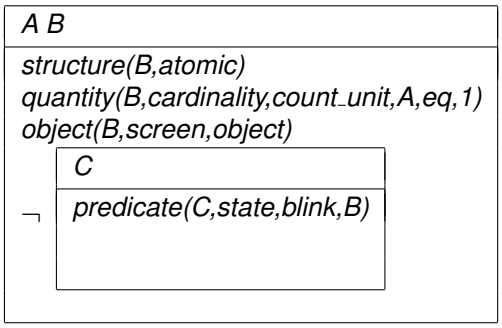
*John enters **not less than 2** codes.*



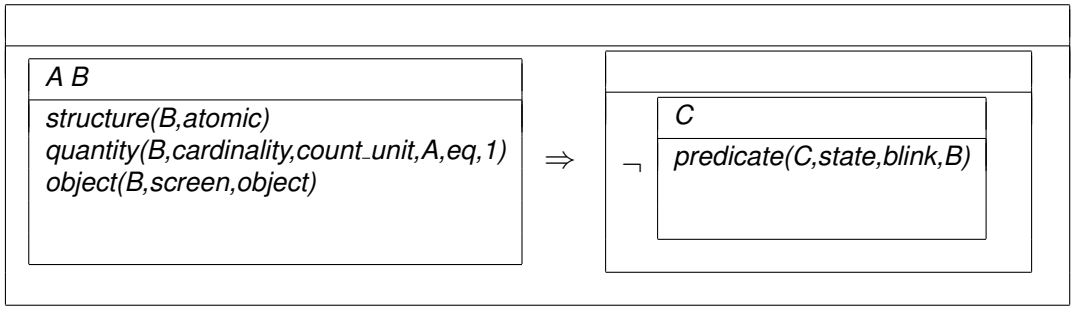
9.2 Verb Phrase Negation

9.2.1 Intransitive Verbs

A screen does not blink.

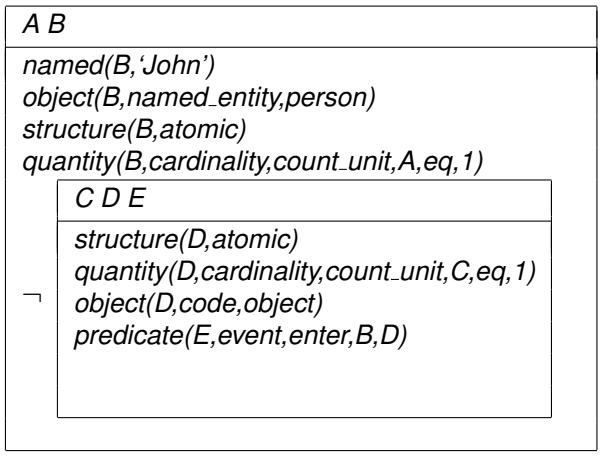


Every screen does not blink.

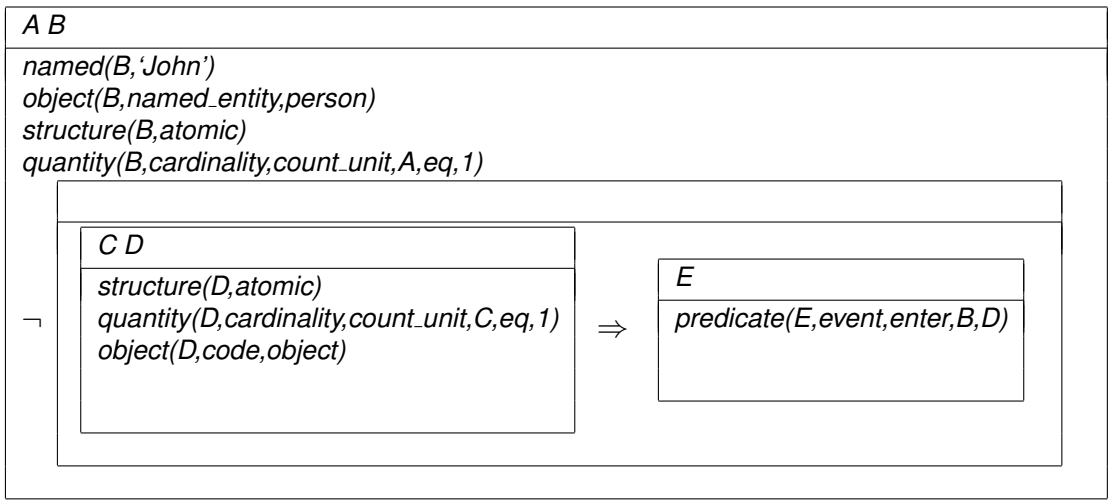


9.2.2 Transitive Verbs

John does not enter a code.

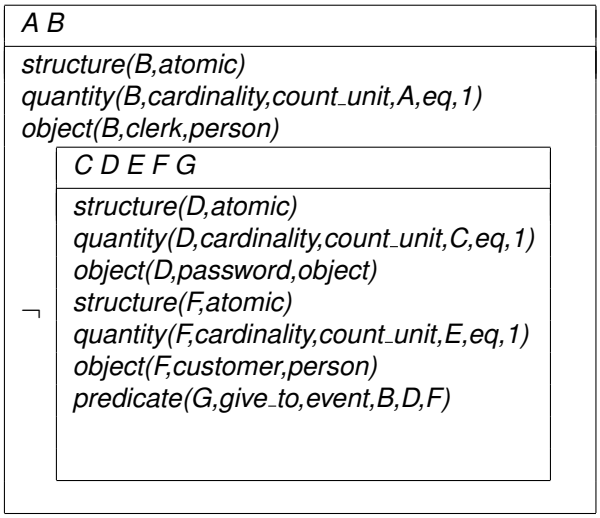


John does not enter every code.



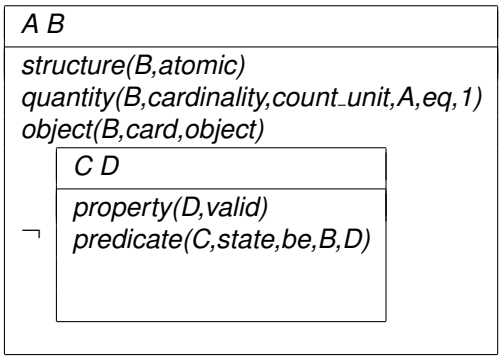
9.2.3 Ditransitive Verbs

A clerk does not give a password to a customer.



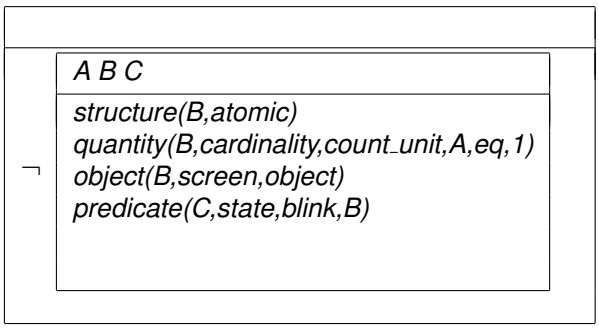
9.2.4 Copula

A card is not valid.

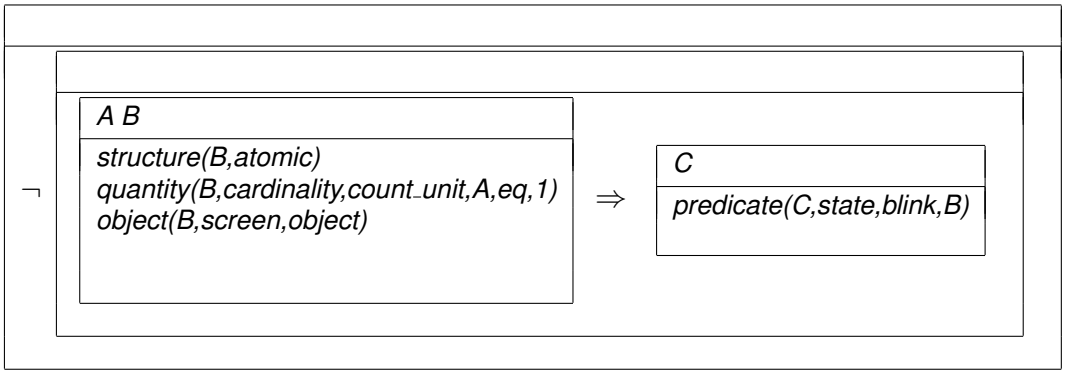


9.3 Sentence Negation

It is not the case that *a screen blinks.*

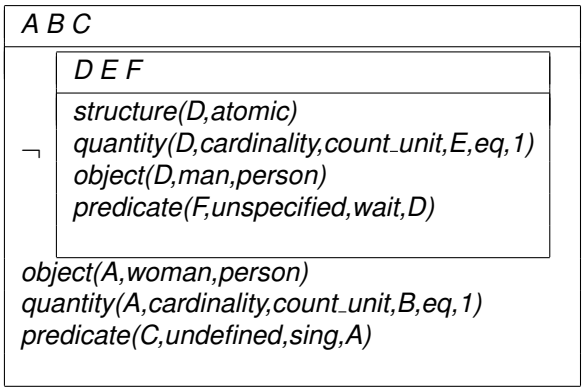


It is not the case that *every screen blinks.*

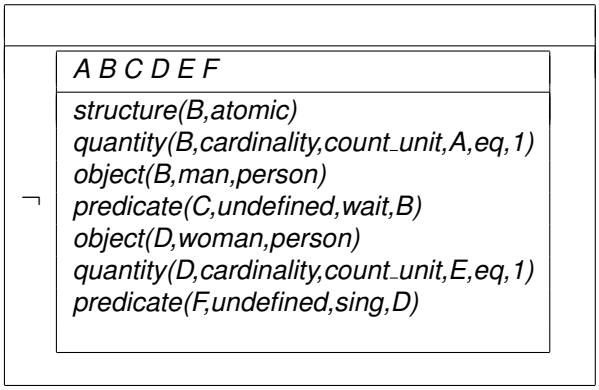


Sentence negation takes narrow scope, but wide scope can be triggered by repeating the *that* complementizer. Compare the following two examples.

It is not the case that *a man waits and a woman sings.*



It is not the case that *a man waits and* ***that*** *a woman sings.*



10 Plural Interpretations

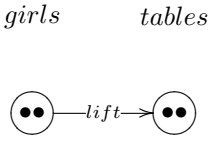
In this section, we present the eight readings of the natural English sentence

2 girls lift 2 tables.

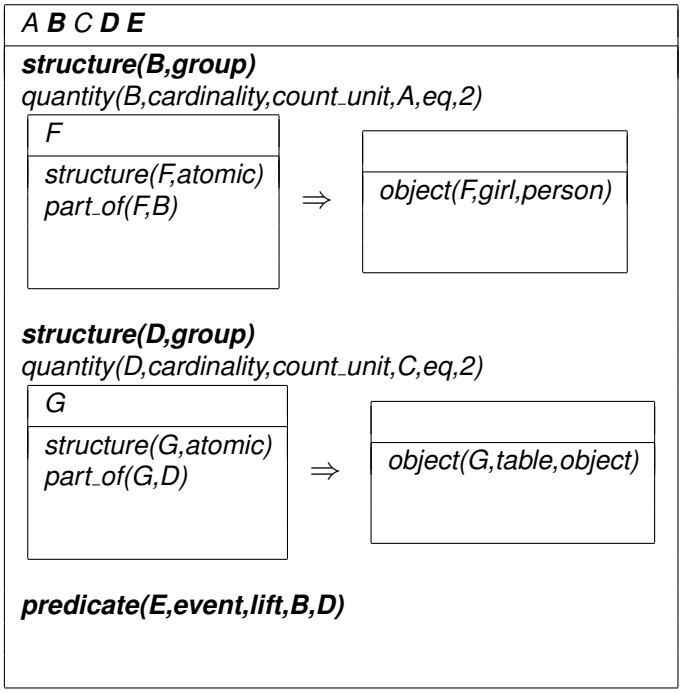
which can be expressed in ACE 4. Note that reading 4 has two interpretations, the second of which is given as reading 4' at the end of the section. For background information on the disambiguation of plurals consult [5] and [6].

In ACE, a plural noun phrase has a default collective reading. To express a distributive reading, a noun phrase has to be preceded by the marker *each of*. The relative scope of a quantifier corresponds to its surface position. We use *there is/are* and *for each of* to move a quantifier to the front of a sentence and thus widen its scope.

10.1 Reading 1

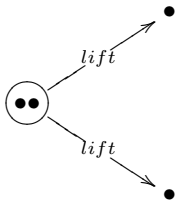


2 girls lift 2 tables.

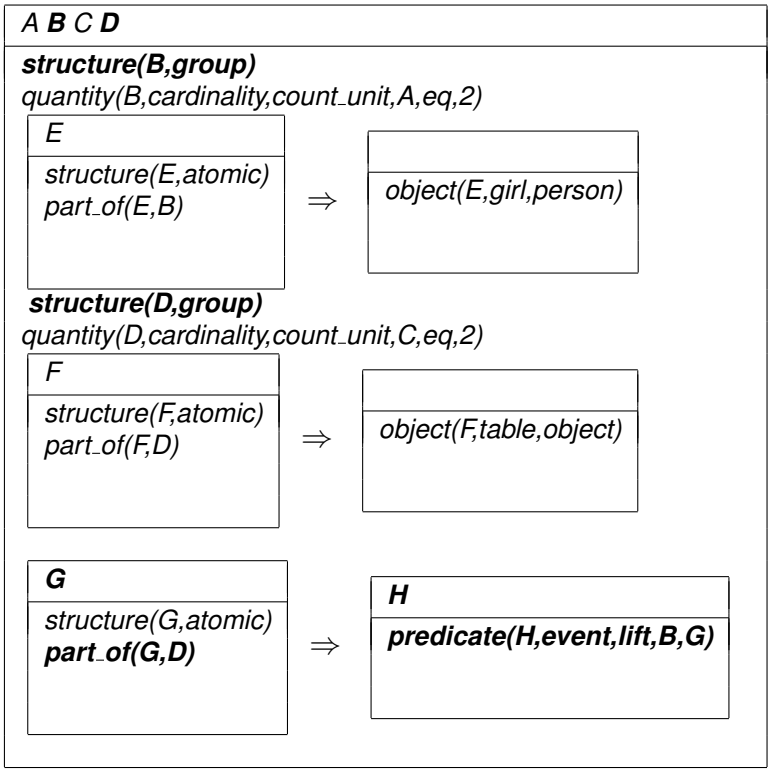


10.2 Reading 2

$girls$ $tables$

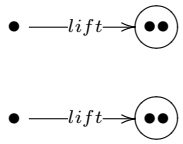


2 girls lift each of 2 tables.

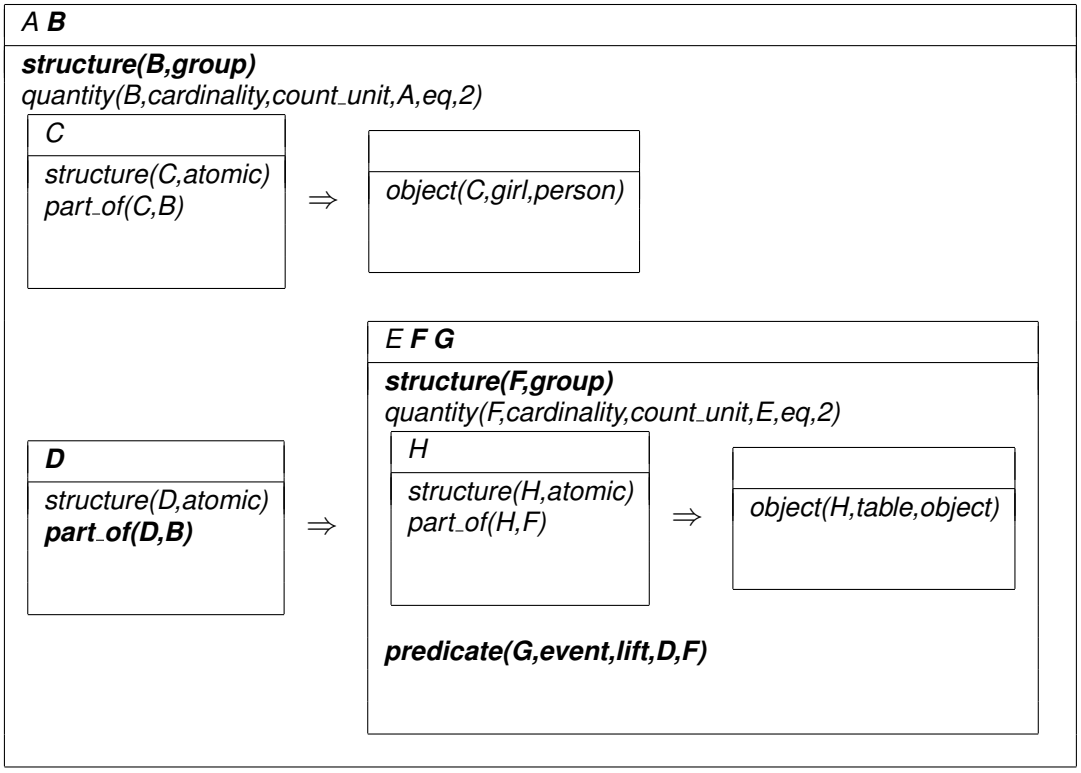


10.3 Reading 3

$girls$ $tables$

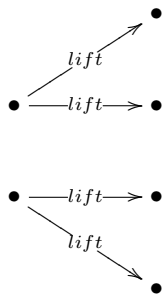


Each of 2 girls lifts 2 tables.

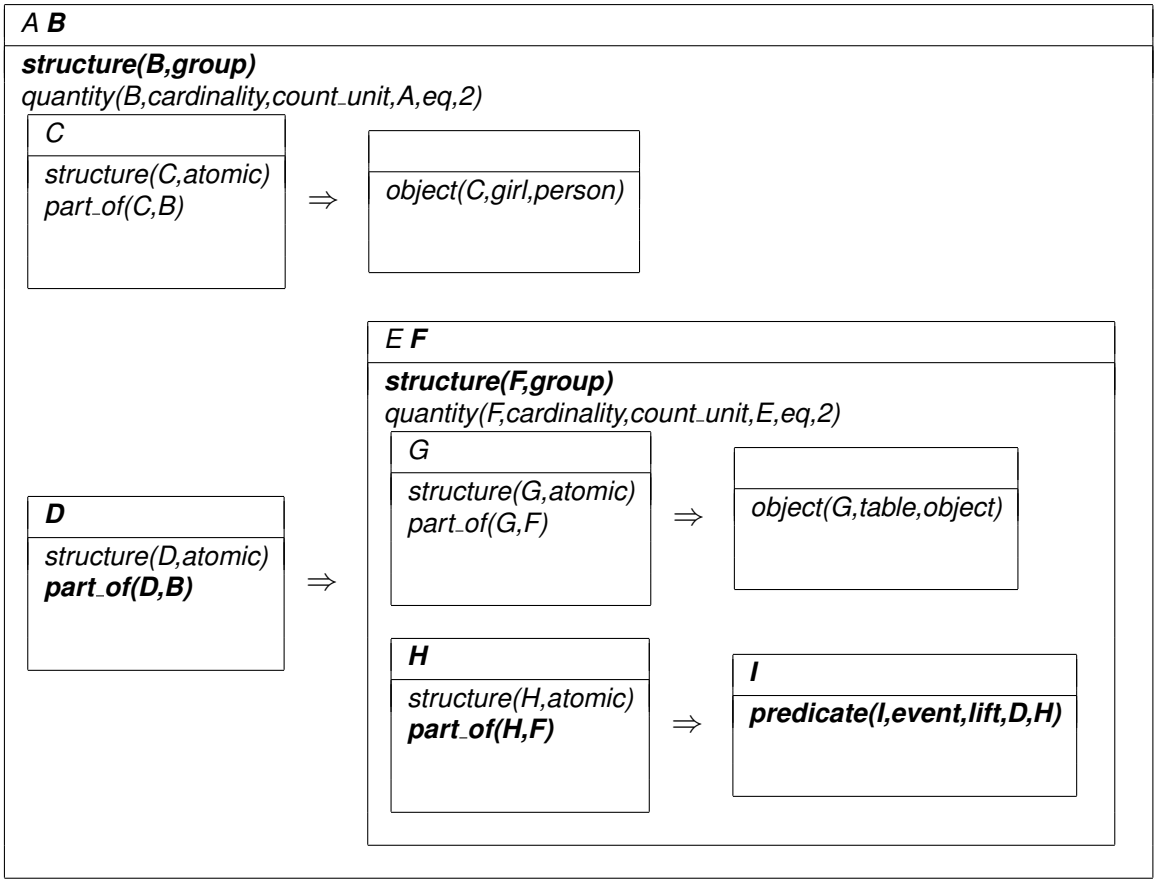


10.4 Reading 4

$girls$ $tables$



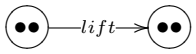
Each of 2 girls lifts each of 2 tables.



10.5 Reading 5

Reading 5 is identical to reading 1.

$girls$ $tables$



There are 2 tables such that 2 girls lift them.

*A **B** C **D** **E***

structure(B,group)
quantity(B,cardinality,count_unit,A,eq,2)

F	⇒	
structure(F,atomic) *part_of(F,B)*		*object(F,table,object)*

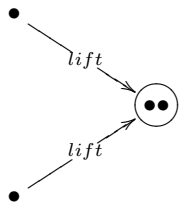
structure(D,group)
quantity(D,cardinality,count_unit,C,eq,2)

G	⇒	
structure(G,atomic) *part_of(G,D)*		*object(G,girl,person)*

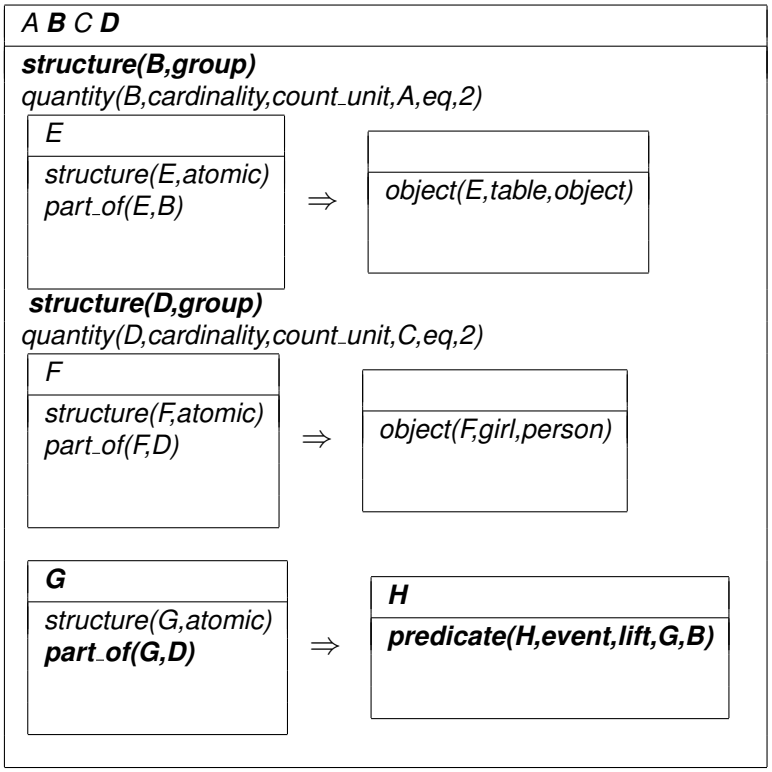
predicate(E,event,lift,D,B)

10.6 Reading 6

$girls$ $tables$

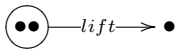


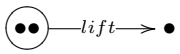
There are 2 tables such that each of 2 girls lifts them.



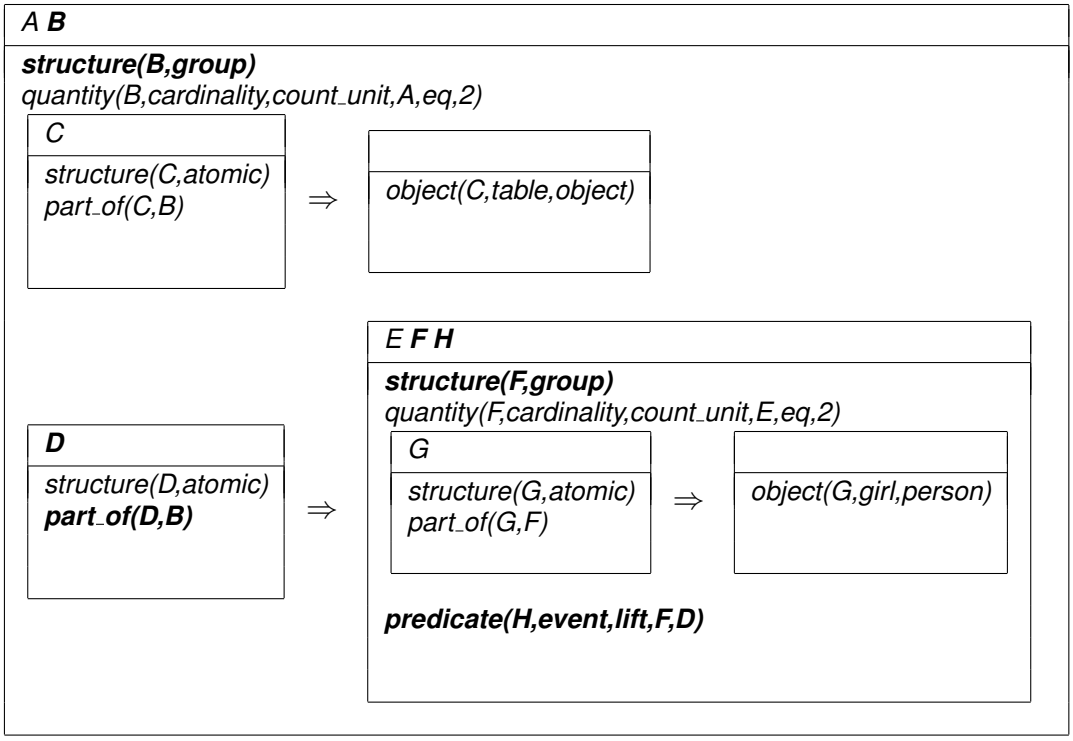
10.7 Reading 7

$girls$ $tables$



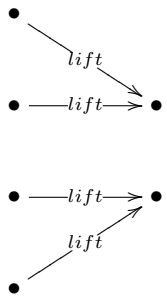


For each of 2 tables 2 girls lift it.

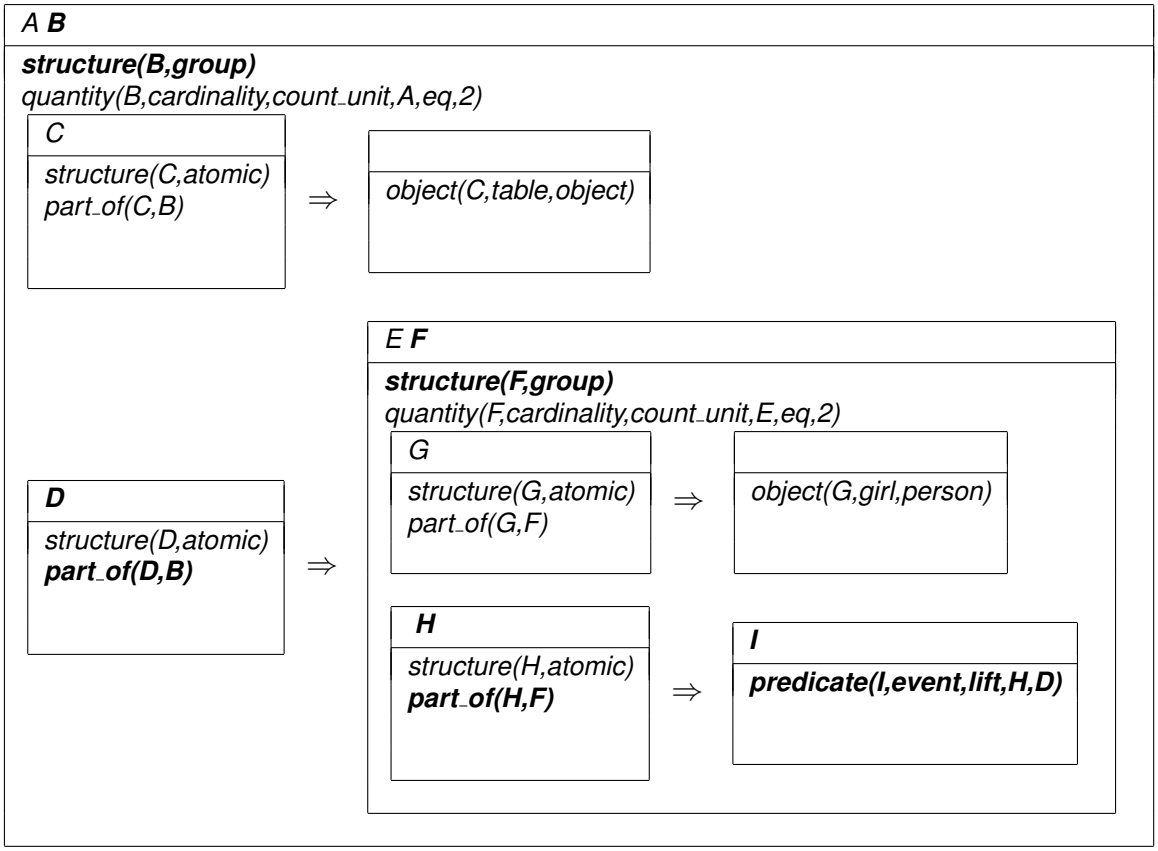


10.8 Reading 8

$girls$ $tables$

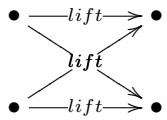


For each of 2 tables each of 2 girls lifts it.

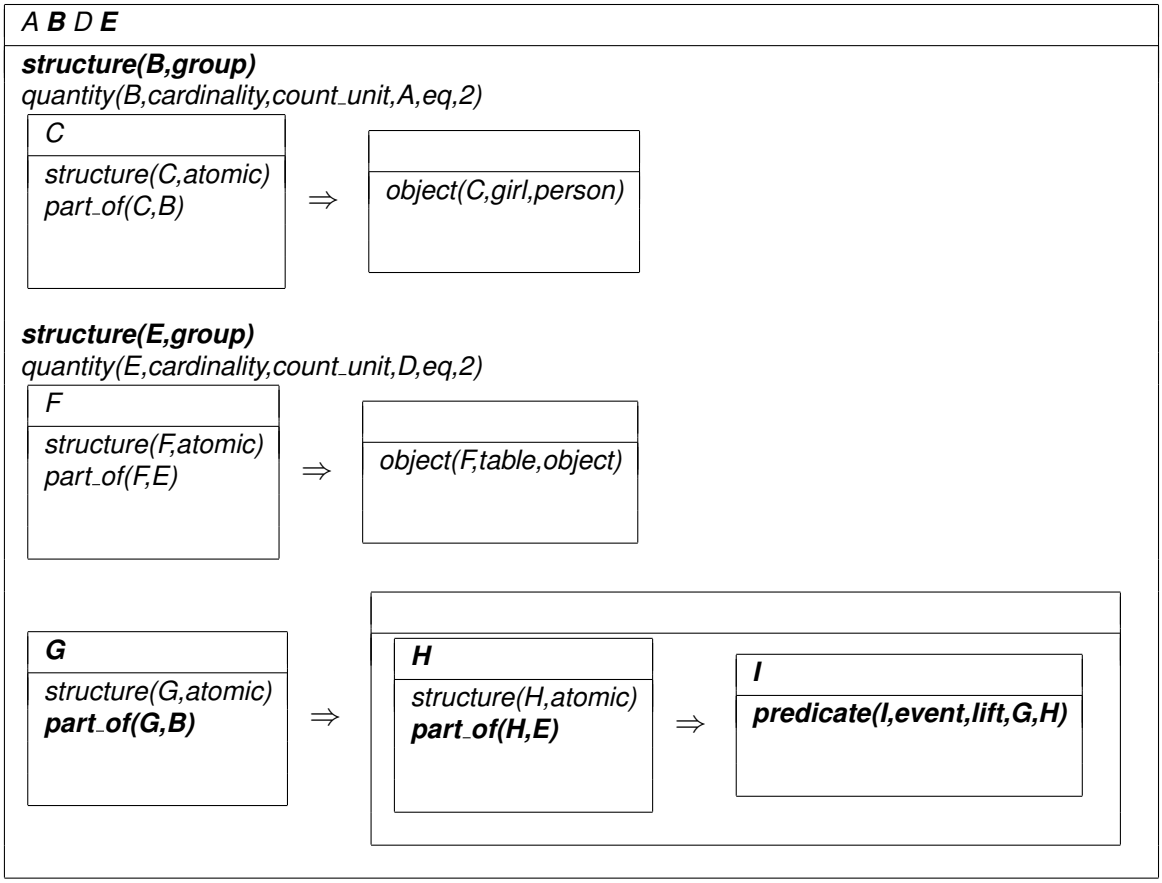


10.9 Reading 4'

$girls$ $tables$



There are 2 girls and there are 2 tables such that each of the girls lifts each of the tables.



11 ACE Questions

Yes/no-questions ask for the existence of a state of affairs. These questions are translated exactly as their declarative counterparts.

Does John enter a card?

A B C D E
object(C,card,object) *quantity(C,cardinality,count_unit,B,eq,1)* *structure(C,atomic)* *predicate(D,event,enter,A,C)* *object(A,named_entity,person)* *quantity(A,cardinality,count_unit,E,eq,1)* *structure(A,atomic)* *named(A,'John')*

Is the card valid?

A B C D
object(B,card,object) *quantity(B,cardinality,count_unit,A,eq,1)* *structure(B,atomic)* *property(C,valid)* *predicate(D,state,be,B,C)*

11.1 Who/What/Which-Questions

Who/what/which-questions ask for the subjects or the objects of sentences. These questions are translated as their declarative counterparts but contain additional conditions for the query words.

***Who** enters what?*

A B C D E
query(A,who) ***structure(A,dom)*** ***quantity(A,unspecified,unspecified,B,eq,unspecified)*** ***object(A,unspecified,person)*** *query(C,what)* *structure(C,dom)* *quantity(C,unspecified,unspecified,D,eq,unspecified)* *object(C,unspecified,object)* *predicate(E,event,enter,A,C)*

Which customer *enters a card?*

A B C D E
query(B,which)
structure(B,atomic)
quantity(B,cardinality,count_unit,A,eq,1)
object(B,customer,person)
structure(D,atomic)
quantity(D,cardinality,count_unit,C,eq,1)
object(D,card,object)
predicate(E,event,enter,B,D)

11.2 Where/When/How/...-Questions

Where/when/how/...-questions ask for details of an action. These questions are translated as their declarative counterparts but contain additional conditions for the query words.

Where *does John enter a card?*

A B C D E F G
named(G,'John')
structure(G,atomic)
quantity(G,cardinality,count_unit,A,eq,1)
object(G,named_entity,person)
query(F,E,where)
modifier(B,location,F,E)
predicate(B,event,enter,G,C)
structure(C,atomic)
quantity(C,cardinality,count_unit,D,eq,1)
object(C,card,object)

When *does John enter a card?*

A B C D E F G
named(G,'John')
structure(G,atomic)
quantity(G,cardinality,count_unit,A,eq,1)
object(G,named_entity,person)
query(F,E,when)
modifier(B,time,F,E)
predicate(B,event,enter,G,C)
structure(C,atomic)
quantity(C,cardinality,count_unit,D,eq,1)
object(C,card,object)

How *does John enter a card?*

A B C D E F G
named(G,'John') *structure(G,atomic)* *quantity(G,cardinality,count_unit,A,eq,1)* *object(G,named_entity,person)* ***query(F,E,how)*** ***modifier(B,manner,F,E)*** *predicate(B,event,enter,G,C)* *structure(C,atomic)* *quantity(C,cardinality,count_unit,D,eq,1)* *object(C,card,object)*

A Appendix: Predicate Declarations

modifier(X,K,Preposition,Y/Adverb)

X		discourse referent of the event or state that is modified
K	$\in$	{location, origin, direction, time, start, end, duration, instrument, comitative, manner, ...}
Y		discourse referent of an object, i.e. the NP of the modifying PP

named(X,ProperName)

X		discourse referent of the object that is named

object(X,Noun,K)

X		discourse referent of the object that is denoted by the noun
K	$\in$	{person, object, time}

part_of(X,Y)

X		discourse referent of an (atomic) object
Y		discourse referent of a (group) object

predicate(E,D,Verb,X)

E		discourse referent of the event or state that is denoted by the verb
D	$\in$	{event, state}
X		discourse referent of the subject

predicate(E,D,Verb,X,Y)

E		discourse referent of the event or state that is denoted by the verb
D	$\in$	{event, state}
X		discourse referent of the subject
Y		discourse referent of the direct object

predicate(E,D,Verb,X,Y,Z)

E		discourse referent of the event or state that is denoted by the verb
D	$\in$	{event, state}
X		discourse referent of the subject
Y		discourse referent of the direct object
Z		discourse referent of the indirect object

proper_part_of(X,Y)

X		discourse referent of an (atomic) object
Y		discourse referent of a (group) object

property(X,IntransitiveAdjective)

X		discourse referent of the object a property of which is described by the adjective

property(X,Comparative/TransitiveAdjective,Y)

X		discourse referent of the object that is described
Y		discourse referent of the object with which X is compared or the object of the adjective

property(X,TransitiveComparative,Y,Z)

X		discourse referent of the object that is described
Y		discourse referent of the object of the adjective
Z		discourse referent of the object with which X is compared

quantity(X,K,I,Q,J,N)

K	$\in$	{cardinality, weight, size, length, volume, dimension, ...}
I	$\in$	{count_unit, unit, kg, cm, liter, ...}
X		discourse referent of the object the quantity of which is indicated
Q		discourse referent of the (reified) quantity of X
J	$\in$	{eq, leq, geq, greater, less}
N		a number, or unspecified

query(X,Q)

X		discourse referent of the object that is asked for
Q	$\in$	{who, what, which}

query(P,Y,Q)

P		preposition
Y		discourse referent of an object, i.e. the NP of the modifying PP or an adverb
Q	$\in$	{where, when, how, ...}

quoted_string(X,QuotedString)

X		discourse referent of the object that is denoted by the quoted string

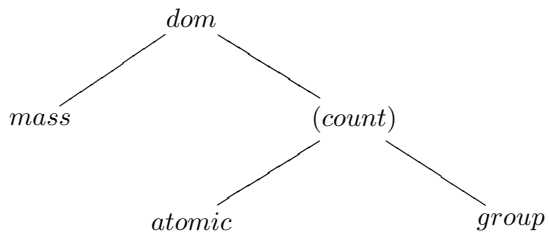
relation(X,Relation,of,Y)

X		discourse referent of the object that is related to Y
Y		discourse referent of the object that is related to X

$$X \xrightarrow{Relationof} Y$$

structure(X, D)

X		discourse referent of the object the structure of which is indicated
D	$\in$	{atomic, group, mass, dom}



sum_of(X,L)

X	discourse referent of a (group) object
L	list of discourse referents of objects that are a proper part of X

References

[1] Attempto Website. `http://www.ifi.unizh.ch/attempto`.

[2] Patrick Blackburn and Johan Bos. *Working with Discourse Representation Structures*, volume 2nd of *Representation and Inference for Natural Language: A First Course in Computational Linguistics*.

[3] Stefan Hoefler. The Syntax of Attempto Controlled English: An Abstract Grammar for ACE Version 4.0. Technical Report ifi-2004.03, Department of Informatics, University of Zurich, 2004.

[4] Hans Kamp and Uwe Reyle. *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer, Dordrecht/Boston/London, 1993.

[5] Uta Schwertel. Controlling Plural Ambiguities in Attempto Controlled English. In *Proceedings of the 3rd International Workshop on Controlled Language Applications*, Seattle, Washingtion, 2000.

[6] Uta Schwertel. *Plural Semantics for Natural Language Understanding – A Computational Proof-Theoretic Approach*. PhD thesis, University of Zurich, Zurich, 2003.

Index