INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center St. · Suite 600 · Berkeley, California 94704-1198 · (510) 666-2900 · FAX (510) 666-2956

A Proposed Formalism for ECG Schemas, Constructions, Mental Spaces, and Maps

Jerome A. Feldman

TR-02-010

September 2002

Abstract

The traditional view has been that Cognitive Linguistics (CL) is incompatible with formalization. Cognitive linguistics is serious about embodiment and grounding, including imagery and image-schemas, force-dynamics, real-time processing, discourse considerations, mental spaces, context, and so on. It remains true that some properties of embodied language, such as context sensitivity, can not be fully captured in a static formalism, but a great deal of CL can be stated formally in a way that is compatible with a full treatment. It appears that we can specify rather complete embodied construction grammars (ECG) using only four types of formal structures: *schemas, constructions, maps,* and *spaces.* The purpose of this note is to specify these structures and present simple examples of their use.

A Proposed Formalism for ECG Schemas, Constructions, Mental Spaces, and Maps

The traditional view has been that Cognitive Linguistics (CL) is incompatible with formalization. Cognitive linguistics is serious about embodiment and grounding, including imagery and image-schemas, force-dynamics, real-time processing, discourse considerations, mental spaces, context, and so on. In traditional formal approaches, little of this could be discussed and many CL workers gave up on formalization altogether. This has had many unfortunate effects, the most serious of which has been a lack of precision in CL.

It remains true that some properties of embodied language, such as context sensitivity, can not be fully captured in a static formalism, but a great deal of CL can be stated formally in a way that is compatible with a full treatment. In terms of the Neural Theory of Language (NTL) we can view a formal grammar as specifying the connections that exist in a neural realization of a grammar, without specifying the weights of these connections or the dynamics of how the system behaves in context.

Within a Neural Theory of Language (NTL), the precision of formal approaches becomes consistent with the traditional concerns of Cognitive Linguistics. In NTL, dynamic embodied semantics, discourse, and phonology can be modeled via what is called "enactment" or "imaginative simulation." Each such enactment, in a neural system requires a control mechanism consisting of neural parameters -- minimal information structures guiding the embodied enactment. In NTL, these can be modeled precisely, that is, formally. In other words, we can precisely specify the parameterizations of semantics and phonology, the formalism shaped by the embodied semantics and phonology. Grammar, morphology, and the lexicon can then be specified with equal precision as the pairing of semantics (including discourse) with phonology (including the order of phonological elements in speech).

This working note incorporates ideas from several members of the NTL group and has been fairly stable for about a year. It assumes a paradigm for language understanding comprised of two distinct phases. The first, *analysis*, phase takes an utterance in context and produces a semantic specification, the *SemSpec*, which is used by the second, *enactment*, phase in understanding the utterance. This is all described in various papers of which [BC 2002] is the most recent. Within this paradigm, it appears that we can specify rather complete grammars using only four types of formal structures: *schemas, constructions, maps*, and *spaces*. The purpose of this note is to specify these structures and present simple examples of their use.

In addition to the grammar, we assume that there will be one or more external ontologies involved, with the obvious links between lexical items and ontology items (ExItem) and between ontology relations (ExRel) and the relations used in the grammar. In the grammar, category constraints (ExCat) from the ontology can be used to specify role restrictions. External predicates in the grammar will be restricted to those that are expressed in the associated external ontologies.

We are following the general linguistic paradigm that a grammar of e.g., English, can be independent of much of the our detailed world knowledge and that people can learn new words and fields without changing the basic grammar. From an applied perspective, this means that we can build a core NLU system that can be used with novel applications by specifying interfaces to the ontology and Enactment modules for that domain. From the neural/psychological perspective, this says that only part of human knowledge is schematized for language

The immediate consequence of this stance is that we will NOT recreate all world knowledge as a collection of schemas and relations. Only the categories and schemas needed for Analysis must be defined. It is not obvious that this separation of grammar and detailed meaning can be achieved, but that is our goal, for the reasons described just above. Some grammatical features (case, gender, etc.) will be quite like those of unification grammars such as HPSG [HPSG]. But there is an additional novel idea being explored in ECG.

Grammars in ECG are deeply cognitive, with meaning being expressed in terms of cognitive primitives such as image schemas, force dynamics, etc. The hypothesis is that a modest number of universal primitives will suffice to provide the core meaning component for the grammar. Specific knowledge about specialized items, categories and relations will be captured in the external ontology as ExItem, ExCat, and ExRel respectively. External items, etc. can appear in an ECG grammar and new ones can be freely added provided only that they are well defined in an external ontology. More details on this will be given at appropriate places in this note.

In addition to general knowledge represented in the ontology, there will be an evolving belief structure capturing the understander's beliefs about the discourse situation. In this note, we will not specify more about these components nor about the X-schemas needed for Enactment. The focus here is on formalism for representing the knowledge structures needed for the SemSpec and for constructions that map from linguistic form to these meaning structures.

The key to scalability in any paradigm is compositionality; our goal in modeling language understanding is to systematically combine the heterogeneous structures posited in cognitive linguistics to yield overall interpretations. We

have identified four conceptual primitives that we believe capture the proposed structures and thus suffice for building scalable language understanding systems: SCHEMA, MAP, MENTAL SPACE, and CONSTRUCTION. We will describe each primitive using a common formalism based on that used in the Embodied Construction Grammar (ECG) framework. The unified representation of these four primitives provides an overarching computational framework for identifying the underlying conceptual relations between diverse linguistic phenomena.

The various formal types that we will define each has a lattice structure induced by the SUBCASE OF relation. These should not be viewed as part of the external ontology, but as separate ECG lattices – namely the SCHEMA, MAP, MENTAL SPACE, and CONSTRUCTION lattices. We will give some examples of each of the four basic types after each is defined.

SCHEMAS

Schemas are the basic building block of ECG semantics and are intended to model image schemas, active X-schemas, Fillmore Frames, etc. A schema description is constituted of optional elements as follows:

SCHEMA <name>
 SUBCASE OF <schema>
 EVOKES <schema> AS <local name>
 ROLES
  < local role >: <role restriction>
  < local role > <-> <role>
 CONSTRAINTS
  <role> <- <value>
  <role> <-> <role>
  < setting> :: <role> <-> <role>
  <predicate>
  < setting name> :: <predicate>

Local roles can be names inherited or introduced in the current definition. Keith Sanders has suggested allowing the bracketed repetition of inherited roles and that seems fine. Role restrictions consist of a type (another schema name or a category of an external ontology) and an optional cardinality restriction. The double arrow <-> notation specifies that the two roles are to be unified. This expression can appear in either the ROLES or CONSTRAINTS section for convenience. If a local role name ends in *, that role can take multiple values. More generally, a <role> can be either a local role or a dotted slot chain in the standard way.

Values can include numbers and strings for now. These include the fixed values for conventional roles such as PLURAL, 2PERSON, etc. The setting names will come from a fixed set of roles in control schemas, e.g., before, happen. The :: notation specifies that the following condition holds when simulation is in the designated state or transition. This is intended to capture the fact that some schemas model dynamic situations. It also seems to be good for capturing the distinction between permanent constraints and ones that are variously called stage, transitory, or episodic.

The predicates model particular semantic relations that hold in a given schema (and later in a construction, etc.). These are restricted to a fixed set that can be evaluated wrt the external ontology (ExRel) and internal belief structure. These include situational calculations like bigger(box6,pen7). Predicates can either be persistent (individual) properties or, when marked by a :: prefix, transitory or stage properties.

The special identifier SELF refers to the schema (and later the mental space, map or construction) being defined. One of the innovations of ECG is the EVOKES primitive. A use of EVOKES brings into the analysis (activates at the neural level) a schema that is related to the one being defined and deliberately under specifies the relation between the two schemas; any relation between the two schemas must be specified by explicit role binding.

For example:

```
SCHEMA hypotenuse
  SUBCASE OF line-segment
  EVOKES right-triangle AS rt
  ROLES  Comment inherited from line-segment
  CONSTRAINTS
        SELF <-> rt.long-side
```

In this classical Langacker example, the hypotenuse schema is a special case of line-segment and inherits its roles, not given here. The very idea of hypotenuse involves the notion of a right triangle and this is a standard use of EVOKES. Under specification is crucial here because the triangle might be mentioned before or after its hypotenuse in discourse. The CONSTRAINTS section specifies that each instance of a hypotenuse is to be bound to (unified with) the long-side role of its parent triangle. Our convention is that an instance of any schema is specified by the schema name followed by an integer, e.g., hypotenuse47.

The most important difference between SUBCASE OF and EVOKES is that in the former case, the new schema can act as a specialization of its parent and

inherits all of the parental roles, while in the latter case the new schema just uses evoked schemas as auxiliaries. Evokes introduces a crucial mechanism of under specification – when one schema evokes another, there is no commitment on which appears first and also no implied subcase relation in either direction.

A more typical example use would be the following two related schemas:

```
SCHEMA SPG
  ROLES
    source: Place
    path: Directed –Curve
    goal: Place
    trajector: Entity

SCHEMA Translational-motion
  SUBCASE of Motion
  EVOKES SPG AS s
  ROLES
     mover <-> s.trajector
     source <-> s.source
     goal <-> s.goal
  CONSTRAINTS
    before:: mover.loc <-> source
    after:: mover.loc <-> goal
```

Here the SPG (source, path, goal) schema is simple and primitive, reflecting the belief that goals are a cognitive building block. The Translational-motion schema is more involved. It evokes (activates) an instance of SPG and is also a subcase of Motion in general. The roles before and after are inherited from Motion and refer to states of the general X-schema controller. The constraints specify that the location of the moving entity is the same as source before the motion and is bound to the goal after the motion. The `::' notation thus captures the distinction between permanent constraints and ones that are more transitory or episodic. More generally, the (<->) binding of roles is quite like standard unification and is the basic operation of ECG.

CONSTRUCTIONS

Constructions are parings of form and meaning. The meaning pole of a construction is quite like a SCHEMA and we will use essentially the same formalism as above to describe the MEANING part of constructions. In the current design, the construction specification has three subparts. CONSTRUCTIONAL elements and constraints entail both form and meaning; FORM and MEANING sections obviously do not. The full specification is:

CONSTRUCTION <name>
 SUBCASE OF <constructions>
 CONSTRUCTIONAL
    EVOKES < construction > AS <local name>
    CONSTITUENTS <local name>: < construction >
    ROLES Comment same as for schemas
    CONSTRAINTS
  FORM
    ELEMENTS <form elements>
    CONSTRAINTS <order constraints and others>
 MEANING
   SUBCASE OF <named schema>
   EVOKES <schema> AS <local name>
   ROLES Comment same as for schemas
   CONSTRAINTS Comment same as for schemas

Again, SUBCASE OF denotes inheritance with all parental roles being available. Constructions, as opposed to schemas, do have CONSTITUENTS and these are themselves constructions. The Constructional section has the full range of possibilities. EVOKES, as with schemas, can bring in other constructions that are related in a variety of ways to SELF. The most common use seems to be to activate containing or parallel constructions that fit with SELF. Constructional roles and constraints are used to capture agreement relations, among other things.

Form constraints act upon both the pure form ELEMENTS specified and upon the form poles of CONSTITUENTS and their own CONSTITUENTS, etc. through dotted names. A few examples will follow.

The meaning section of a construction can evoke a named schema as well as additional roles and constraints. The meaning constraints will do most of the work of integrating this construct with the evolving SemSpec. The current design includes a convention whereby agreement roles in the meaning pole of a construction are also considered to be constructional roles unless there is an

explicit blocking role value. For example, the German lexical entry for Maedchen (young girl) might be something like:

```
CONSTRUCTION maedchen
 SUBCASE OF common-noun
 CONSTRUCTIONAL
    ROLES
      Gender <- neuter
 FORM
       "Maedchen"
 MEANING
        SUBCASE OF Referent-schema
        CONSTRAINTS:
        Gender <- feminine
        Number <- singular
        Category <- human
        Attributes* <-  Age(SELF, young)
```

Here the grammatical neuter of the word blocks the percolation of its semantically feminine character for agreement purposes. But its other semantic role values are also grammatical agreement features. The notation:

Age(SELF, young)

is a shorthand for expressing the fact that one attribute of this lexeme is a relation schema of the form:

```
SCHEMA age_scale54
 SUBCASE OF linear scale
 ROLES
        { Subject:  Entity }
        { Value:    Age-Value }
 CONSTRAINTS:
        Subject <-> Maedchen
        Value <- young
```

Notice that this schema has one role unified with the Maedchen construction and is also itself identified as an Attribute of Maedchen. This two way linking will be common in ECG and reflects bi-directional neural connectivity.

Some other construction examples include:

CONSTRUCTION spatial-predication
 SUBCASE OF phrase
 CONSTRUCTIONAL
    CONSTITUENTS
       rel : spatial-relation
       lm : referring-exp
    CONSTRAINTS
       rel.case <-> lm.case
       rel.number <-> lm.number
 FORM
      CONSTRAINTS
       rel < lm
 MEANING Comment – this construction just adds one binding
       rel. landmark <-> lm

Even in English, spatial prepositions select for case and some also for number features. So we don't get:
       *among a cow, *with they/their.
The constructional constraints capture this.

CONSTRUCTS and INSTANCES

We will retain the standard distinction between constructions, which define relations between form and meaning and CONSTRUCTS, which are instances of constructions with specific bindings. It seems that we will not normally need to use special terminology to distinguish instances from definitions for SCHEMAS or MAPS. Recall that the result of analysis is a collection of interlinked schema instances (prominently of REFERENT and PREDICATE schemas) called the SemSpec. If the need arises, we can explicitly say "schema-instance" and similarly "map-instance" and possibly "mental-space-instance", but this shouldn't be required often.

People will normally write down the definitions of schemas, constructions, maps, and mental spaces, not instances of them. But we will sometimes want to write specific instances as examples. In these cases we will use the convention that a schema (construction, map or mental space) name followed by an integer will denote an instance of that schema. For example: SCHEMA hypotenuse42

MAPS

By now it should be no surprise that we will want maps to also be described in way similar to those used for schemas and constructions. The crucial distinction is that <-> bindings unify two entities to be the same while maps explicitly connect two different kinds of things. Maps appear in several distinct roles in ECG, but the same formal apparatus appears to handle them all.

MAPS should have all the parts of SCHEMAS, plus one additional section, PAIRS, which specifies the mapping between the roles of the various schemas involved. Formally, this becomes:

MAP <name>
SUBCASE OF <map>
EVOKES <map> AS <local name>
ROLES Comment – these are roles of the map itself
<local role>: <role restriction>
<local role> <-> <role>
CONSTRAINTS Comment same as for schemas
PAIRS
<schema>.<role> ~ <schema>.<role>

Where, as always, the role names can be dotted slot chains. We will want to add more elaborate pairings with value mappings as well, but that can wait.

The roles and constraints apply to the map as a whole and not to any particular pairing. For maps, the current design does not have SUBCASE OF cause the new map to inherit all the PAIRS of its parent maps - just their roles. The pair inheritance effect can be achieved with the notation for copying pairings from EVOKED maps, as described later in this section. We obviously can add a notation for inheriting all parental pairs if that proves useful.

For example, a simplified time-money metaphor map might be:

MAP timeISmoney
SUBCASE OF timeAS resource
ROLES // inherited from timeAS resource
{ map_type <- Metaphor }
{ source: Domain }
{ target: Domain }
CONSTRAINTS
map_type<- METAPHOR
source <- money

target <- time

PAIRS

target.interval ~ source.amount

target.availability ~ source.possession

target.allocate ~ source.spend

We also need a notation for accessing the elements of a map. There is a similar need for accessing the elements of a mental space and all of the notations must cohere. The current design uses a prefix operator @ for map elements, for example in the case above:

@timeISmoney. target.interval

would be one way to specify money.amount.

There is a potential ambiguity in the expression above – taken alone it could mean either:

(@timeISmoney. target).interval

or

@timeISmoney. (target.interval)

The suggested design is that any unresolvable ambiguity be an error in the grammar; in fact only the second reading makes sense for our example.

This example is a general metaphor map, but the same notation would be used for specific maps such as those between mental spaces, to be discussed below. It appears that MAPs will also be useful for describing morphological transformations, but that has not been fully worked out.

For specific MAPs, such as those between mental spaces, it will be necessary to have operations for adding and removing PAIRS from a map. For example, a discourse about a movie, will often introduce new relations between actors and their parts. This is a bit different from other operations in the formalism and it is not clear which syntax would be best for this. One notation that is consistent with our current assignment syntax (<-) in meaning constraints is:

<map> <- PAIR(<role>, <role>) for addition

and <map> <- - PAIR(<role>, <role>) for subtraction.

There could also be notation for copying pairs from another map mp, like:

<map> <- mp6.PAIR(<role>, <role>)

where either role could be left blank and filled in automatically. So,

<map> <- mp6.PAIR(,)

would copy all the pairs of mp6 to <map>.

We will see some more example of maps in action in the section on mental spaces.

MENTAL SPACES and their MAPS

The term *mental spaces* (Fauconnier) refers to a conceptual domain built up during discourse, in its most general form simply a set of entities and relations among them. In our language understanding framework, a mental space is a major partition of the overarching conceptual space that characterizes the (speaker's or hearer's) representation of the current discourse; it functions as a domain of reference and predication, such that the referents and predications built up by linguistic expressions must be assigned to or associated with some particular space for enactment to occur.

Mental Spaces play an important role in the formalism. A full or expanded mental space will have its own history, enactment and belief structure. This is a lot of mechanism and it is often not all needed for what are traditionally called mental space phenomena. Each type of mental space (time, counterfactual, depiction, etc.) will have an associated ordinary schema that can be used as a *skeleton* for expanding to a full mental space, examples will follow.

Following Keith Sander's suggestion we will have a parallel ordinary schema for each kind of mental space. For example, depiction can often be handled by a simple schema like:

```
SCHEMA  depiction
  ROLES
     model: SITUATION
     artifact: ARTWORK
     author: PERSON
     medium:  Comment painting, story, etc.
     elements* : entity    // this is a role for the named elements
```

This is enough structure to enable us to say a lot about the picture of Paris on my wall. But if someone went on to say that in this picture the Eiffel tower is only half finished, a mental space and a map is needed. An example of the

proposed method for expanding a schema into a full mental space will be given below.

We will follow a naming convention for mental spaces because each represents a major partition of the Enactment. Recall that each mental space has its own belief context, history, inferences, etc. As before mental spaces have a lattice structure and have the features parallel to the other ECG types. The major addition is that mental spaces will have at least one MAP, often to the discourse space D. If we do adopt a naming convention, <name>_ SP, then we can use dot to access through mental spaces as well without confusion. The syntax is:

SPACE <name_SP>
 SUBCASE OF <mental space>
 EVOKES <mental space> AS <local name>
 MAPS
 ROLES
 < role name>: <role restriction>
 < role name> <-> <role name>
 CONSTRAINTS Comment same as for schemas

As a first example, the mental space corresponding to the depiction frame above might be:

SPACE depiction_SP
 MAPS
 dm: depiction-map
 ROLES
 model: SITUATION
 artifact: ARTWORK
 author: PERSON
 medium: Comment painting, story, etc.
 CONSTRAINTS
 SELF <-> dm.artifact

Where the depiction MAP is defied as:

MAP depiction-map
ROLES
 model: SITUATION
 artifact: ARTWORK
 PAIRS
 model.entity ~ artifact.entity // each instance has specific pairings
 model.relation ~ artifact.relation

For another example of a mental space and its maps, we could have:

SPACE timewarp_SP
 SUBCASE OF spacetime_SP
 ROLES
        status: undetermined
        time_gap:temporal_interval
 MAPS
        entity_map

And as a specific case:

SPACE timewarp_SP6
 ROLES
        status: focus
        time_gap <- 5 years
 MAPS
        entity_map <- Keith_map

Where the Keith_map would consist of pairs linking various people and other significant entities of his current life with what is known about them five years back. This is obviously very different for Keith than it is for most of us.
But in any case, we could refer to Keith's favorite food as an undergrad by something like:
        timewarp_SP6. favorite_food

An First Example

To see how all this fits together, I will sketch out a preliminary version of how we might analyze the sentence:

When he was an undergraduate, Keith ate mostly tofu.

Let's assume that there is a multi-clausal construction like the following:

CONSTRUCTION temporal-pred
 SUBCASE OF multi-clause
 CONSTRUCTIONAL
    CONSTITUENTS
        tc: temporal-clause
        pred: predication
 FORM

```
      CONSTRAINTS
        tc  <  "," < pred
MEANING
      ROLES:
        othertime_MS: timewarp_SP
        time-map:   map
      CONSTRAINTS:
        EXTENDS( D, othertime_MS, time-map)
        othertime_MS <- predication
        time-map  <- PAIR( tc.protag, pred.protag)
```

This construction matches a temporal clause followed by a comma followed by some predication. The meaning part will need to have roles for at least the mental space representing the designated time and for a map linking that to D, the discourse space. Let's look at the three constraints. The first constraint uses a relational primitive EXTENDS, following Fauconnier. It says that the new space, othertime_MS, extends D using the MAP time-map. There might be a need for more than one kind of extension, but that will become obvious as we do examples.

The second constraint says that the predicate expressed by the predicate clause should be asserted in the mental space: othertime_MS. There is some trickiness with tense here – the assertion should be retensed to fit the space. Finally, the third constraint establishes the pairing of the protagonist of the time clause with that of the second clause. This should allow access to other facts about the two referent descriptors.

For the next example, let's look at how a depiction schema might get expanded (opened) into a mental space using the picture of Paris example. Recall that the simplified schema version would be something like:

```
SCHEMA  painting33
 SUBCASE OF : depiction
 ROLES
      medium:  painting
      model: ref 44   Comment Paris in 1885
      artifact:  ref55    Comment the painting on my wall
      author:
```

Now suppose that a new clause is processed that requires the schema to be expanded to a mental space. It could be:

In the painting, the Eiffel Tower is only half finished.

Assuming that a referent descriptor for painting33 (the painting on my wall) is accessible, the appropriate construction should build a mental space:

SPACE depiction_SP22
ROLES
depiction_type <- painting
MAPS
picture_map66

where picture_map66 has a pairing for ref44(Paris) and ref55(the picture) and another pairing for the Eiffel Tower referent and some region of the picture. As more entities (a boat on the Seine) were mentioned in the discourse, additional pairs would be added to the picture_map66 depiction map. For example, if the region of paint called Region77 depicted the boat, the construction that analyzed a referring expression with in a depiction would do three things. It would bind Region77 (computed by, e.g., vision) to its role <depicting region> and bind boat to its role <depicted ref>. Then it would execute a constraint statement like:

<map> <- PAIR(<depicted thing>, <depicting region>)

This is fairly messy, but it will be hard to find anything simpler. If the picture were of Paris in 1885, it should be possible to capture this with a single depiction map where the thing depicted is bound to what the hearer knows about Paris of 1885. Some follow on sentences might have another map to Paris at present, etc.

A Second Example involving Depiction

Here is a simplified version of ECG SemSpec for:
"In the picture, the girl with blue eyes has green eyes."

Suppose we have the depiction schema as above.

SCHEMA depiction
ROLES
model: Situation
artifact: Artwork
author: Person
medium: {book, painting, etc.}
elements* : entity // this is a role for the named elements

We will later expand this into a mental space with the same roles .We will also need three Referent Descriptors. These are not fully described in this note, but are discussed in a companion paper.

Referent Descriptor G1
Category: human
Distribution: individual
Gender: Female
Accessibility: Given
Restriction: EyeColor(blue)
Reified Referent: Olya G.

Referent Descriptor G2
Category: human
Distribution: individual
Gender: Female
Accessibility: Given
Restriction: EyeColor(green)
Reified Referent: generic girl

Referent Descriptor P14
Category: depiction // cf. the depiction schema above
Distribution: individual
Accessibility: Given
Attributes*: size(20,40)
Reified Referent: print on the wall

Space Pic32_MS
SUBCASE OF mental space
MAP: M22
ROLES:
model: Scene33
artifact: P14
author: Picasso
medium: painting
elements: G2, ...

Now, assuming that we refer to the current discourse space as base, we have.

Map M22
SUBCASE OF element map
PAIRS:
base.G1 ~ Pic32_MS.G2

SUBCASE OF painting map
PAIRS:
Map M44
base.P14.region(x,y) ~Pic32_MS.G2

Here, we have two maps. M22 (an element map) maps elements of a base space to their depictions in any kind of depiction space. M44 maps an element in a picture to a region of paint. This should allow a system to Enact the input sentence and also follow on sentences that refer to either G1 or G2. The constructions for all this would need to be worked out in a general compositional way; we can do this if there is public demand.

Discourse Spaces

Our final example illustrates one of the most important uses of mental spaces – capturing the structure of discourse. Any software system we might build for

natural language is, ipso facto, engaged in some kind of discourse. The system might be trying to understand news stories, give tourist advice or tutor a child. The information about the goals and context of a given system, other discourse participants and their interrelationships and goals, and the common ground are all elements of the main discourse space. Importantly, additional discourse spaces for embedded content (e.g., a story about a conversation) as well as models of various participants are all spaces of the same structure. In cognitive modeling terms, this suggest that people have standard ways of organizing different types of discourse and that these nest recursively.

The following extended example is our current way of parameterizing discourse. It relies crucially on mental spaces and is the most complete example of the uses of the formalism space primitives. A full exposition of these structures is beyond the scope of this note, but most of it should be understandable. This is the compilation by Keith Sanders of ideas from several NTL researchers. Note that the notation is slightly different from the formalism as described above – just try to get the general ideas.

space Mental-Space
  roles:
    type: Mental-Space-Type // "discourse", "action" etc.

    entities: *set of* Entity
    spaces: *set of* Mental-Space // MSp's that SELF maps to or from

    entityMaps: *set of* Entity-Map
    spaceRelns: *set of* Space-Relation

    history: Enactment-History
    currentState: Enactment-State

semantic schema Enactment-State
  roles:
    xSchemas: *set of* Marked-X-Schema
    imgSchemas: *set of* Parameterized-Image-Schema

semantic schema Enactment-History
  roles:
    states: *set of* Enactment-State
    semSpecs: *set of* Semantic-Specification

space Discourse-Space
subcase of Mental-Space
  roles:

{ type: Discourse-Space-Type } // a subset of all MSp-Types
genre: Discourse-Genre
register: Register
subjectMatter: Domain

{ entities: *set of* Entity }
participants: *set of* Discourse-Participant // subset of *entities*

currentSpeaker: Discourse-Participant // member of *partcpts*
currentAddressee: *set of* Discourse-Participant // subset of *partcpts*

{ spaces: *set of* Mental-Space } // MSp's that SELF maps to or from
viewpointSpace: Mental-Space // member of *spaces*
focusSpace: Mental-Space // " "

commonGroundSpace: Mental-Space

{ entityMaps: *set of* Entity-Map }
{ spaceRelns: *set of* Space-Relation }

{history: Enactment-History }
{currentState: Enactment-State } // captures current partcpt relations...

segments: *list of* (?) Discourse-Segments
currentSeg: Discourse-Segment // member of *segments*

semantic schema Discourse-Segment
roles:
speaker: Discourse-Participant
addressee: *set of* Discourse-Participant

gestureContent: *set of* Sem-Spec // from co-speech gesture(s)
cxnContent: *set of* Sem-Spec // i.e. semantic poles of constructions

semantic schema Discourse-Participant
subcase of Human
roles:
{ age: Age }
{ sex: Sex } // to be distinguished from gramm'l. gender
{ race: Race }

{ socStatus: Social-Status }
{ grpIdentity: *set of* Social-Group }

// I assume that all Humans maintain various MSp's (belief, desire etc.)...

{ spaces: *set of* Mental-Space }

// ...and when Humans are Discourse-Participants, one of those spaces
// will be a discourse model
discourseModel: Discourse-Space // member of *spaces*

// We want a role here for the participant's overall intent in the discourse,
// and maybe one for their intent in the "current discourse segment".
discourseIntent: Semantic-Specification // (type??)
currSegIntent: Semantic-Specification // (type??)

REFERENCES

BC 2002 B.K. Bergen and N. Chang (2002). Embodied Construction Grammar in Simulation-Based Language Understanding. To appear in Ostman and Fried (eds*) Construction Grammar(s) Cognitive and Cross Linguistic Dimensions*. John Benjamins, to appear.

HPSG Ivan Sag and Tom Wasow; *Syntactic Theory: A Formal Introduction*, CSLI Press, 1999.

CFPS N. Chang, J. Feldman, R. Porzel, and K. Sanders (2002). *Scaling Cognitive Linguistics: Formalisms for Language Understanding*. Proc. First ScaNaLU Workshop, Heidelberg, June 2002