The E-Framework: A Formalism for Natural Language Processing.

Annelise BECH and Anders NYGAARD,

EUROTRA-DK,
University of Copenhagen,
Njalsgade 80,
DK-2300 Kbh. S, Denmark.

Abstract.
This paper presents the most important characteristic of the new formalism used in Eurotra, the E-Framework. It is a formalism for natural language processing within a stratificational model.

In the E-Framework, mapping between levels of representation is performed on the basis of transitions between trees and partial descriptions of objects, called descriptors. These descriptors are completed using the definition of the target level.

The tree to descriptor strategy simplifies the expression of complex two-way relations between text and abstract representations when major structural changes have to be performed on the input. This is illustrated by way of a detailed example showing the interaction of the two formal devices of the E-Framework, the translator and the generator, the basic ideas of which are also briefly described.

The E-Framework has been implemented and forms the basis of the development of Eurotra's pre-industrial prototype of a transfer-based, multi-lingual machine translation system.

1. Introduction.

The E-Framework is the formalism used in Eurotra for implementing a pre-industrial prototype of a stratificational transfer-based, multi-lingual machine translation system.

The E-Framework has been developed on the basis of experiences with earlier Eurotra formalisms, e.g. <C,A>,T /Arnold, et al. 1986/. However, in at least two respects, it differs significantly from its predecessors and other formalisms which transform tree structures by mapping procedures, e.g. Rosetta (cf. /Appelo, et al. 1987/), GETA (cf. /Vauquois, et al. 1985/) or Mu (cf. /Nagao, et al. 1985/): In the E-Framework, there are just two formal devices, and the mapping from one level to the next is **not** performed by traditional tree to tree transducers. Instead, the mapping strategy is based on transitions from trees to partial descriptions.

2. Background.

The large number of transfer components in a multi-lingual translation system obviously makes it desirable to keep them as small and simple as possible. To achieve this, language specific phenomena are neutralized in Eurotra's interface representation of a text, and the main burden of work is shifted to the analysis/synthesis components. This means that every monolingual component expresses a rather complex two-way relation between abstract representation and actual text.

The E-Framework was developed because of problems in expressing this relation in a simple and perspicuous way in the earlier formalisms used in Eurotra: When grammars grew to have substantial linguistic coverage, the rules describing the mapping between levels became highly complex and numerous due to interdependence between the linguistic phenomena triggering structural changes in a representational tree.

The proliferation in the number of rules was mainly due to the fact that mapping rules identified specific target level rules to evaluate the new tree structures, and the complexity of rules was due to the tree to tree transducing strategy used which strictly required specification of full-fledged target level tree structures.

Therefore this kind of mapping strategy was abandoned, and a different one requiring only partial target level tree descriptions, has been adopted as the basic method for performing transitions within the E-Framework.

3. Overview of the E-Framework.

The E-Framework consists of just two formal devices, namely a **generator** and a **translator**. The generator and the translator are abstract devices which interpret **grammars** and **t-modules**, respectively.

A grammar defines a level of representation, and a t-module states the relation between source level trees and their corresponding partial descriptions to be completed by the generator at the target level.

Figure 1 below gives a schematic overview.

3.1. Representational object and descriptor.

Before we describe the translator and generator devices in greater detail, it is useful to have a look at the nature of the material they produce.

The generator produces **representational objects**. A representational object is a tree in which the nodes are **feature bundles**. A feature bundle is a set of simple features of the attribute-value type.

A representational object is fully described by the feature bundles and the dominance and precedence relations between them. By leaving out features and/or information about the relations between feature bundles, we obviously have only a partial description of the object. These partial descriptions we call **descriptors**, and they are what the translator feeds to the generator.

3.2. The Translator.

The translator uses a source level representational object and the specifications in a t-module to produce a descriptor for the target level.

A t-module is declarative; it consists of t-**rules** which describe the relation between source level representational objects and target level descriptors. A t-rule consists of a **left hand side** which describes a tree at the source level, and a **right hand side** which states the corresponding descriptor at the target level.

The left hand side of a t-rule is a tree description where the feature bundles to be included in the target descriptor are marked with identifiers. The right hand side is a definition of a descriptor in the form of identifiers, and dominance and precedence relations between them.

Feature bundles represented by identifiers on the right hand side are by default copies of the source level feature bundles, but t-rules may be stated to specify addition, change or deletion of features.

A single t-rule only defines the descriptor for the part of a source level tree matching the description on its left hand side. The set of t-rules to be used in the production of the descriptor for the full source level tree, is selected by the translator in a top-down driven match of the tree against the left hand sides. If some part of the tree does not match the left hand side of any t-rule in a t-module, the translator copies the dominance and precedence information from the source object.

Since the descriptors are only **partial** target object descriptions, they must be completed; this is done autonomously by the generator device. So, compared to tree to tree transducers, the translator device of the E-Framework can be characterized as quite weak: The translator can only provide guidance for the construction of target objects; it cannot build them itself.

3.3. The Generator.

The generator uses the level definition contained in a grammar and the descriptors provided by the translator to create representational objects.

The grammar itself is declarative; it consists of a set of **g-rules**, each of which is a description of legal parts of representational objects.

The generator completes a descriptor output from the translator by repeatedly applying g-rules. The application of rules is non-deterministic; it continues until no more information can be added and all information from the original descriptor has been validated by at least one g-rule. This results in one or more representational objects.

If some information is **not** validated, the generator cannot produce a legal representational object. Consequently, the attempt at completion fails.

The application of a g-rule is a customized version of unification which not only permits addition of features, but also of structural information and entire feature bundles.

This property of g-rule application is the key reason why a t-module need only specify very little; the E-Framework generator device has autonomous power to finish a partial description, and the descriptions in the grammars are not only used for checking structures, but also for constructing them.

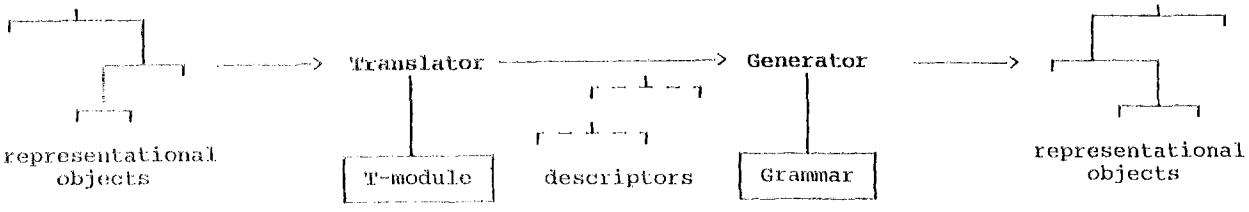
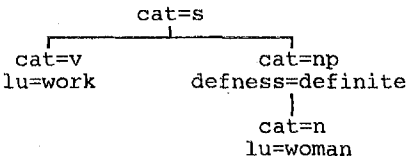


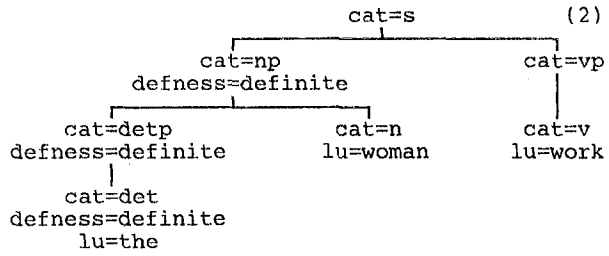
Figure 1.

4. An Example.
In this section, we illustrate how the devices described above interact to perform major structural changes with only a very simple t-module. For the sake of clarity, the example given is somewhat simplified and contains no more than the essential details.

Consider the source level representational object (1), representing the sentence "the woman works", and its target level representation (2).



(1)



(2)

Note that in the source object, the determiner "the" is represented only as a feature of the np-node, whereas in the target object, it has a structural representation. Note also that the constituents in the source objects appear in a canonicalized order. These are just some of the abstractions that are made in Eurotra in order to neutralize language specific behaviour, and which have to be undone in synthesis.

The only t-rule needed to produce the necessary descriptor is (3). Feature bundles are delimited by curly braces, square brackets denote immediate dominance, and angle brackets just dominance. Outside of parentheses, precedence is implicit in the sequence of identifiers. Precedence is unspecified for identifiers in parentheses. Identifiers are written in capital letters, attribute names and constant values in small letters. An arrow, '=>', separates the left and right hand sides of the t-rule.

```
S:{cat=s} [ V:{cat=v}, SUBJ:{cat=np} ]                    (3)
=> S < (V, SUBJ) >
```

From this rule and by copying information from the source level object, the translator produces the following descriptor:

```
{cat=s, ... }                                             (4)
    < ( {cat=v, lu=work, ... },
        {cat=np, defness=definite, ... }
            < {cat=n, lu=woman, ... } > ) >
```

The g-rules describing the target level are the following, where a '^' prefixed to a feature bundle means that it is optional, and an '!' prefix means that the feature bundle should be added if not present. Named variables are written in capitals.

```
{cat=s, ... }                                             (5)
    [ {cat=np, ... },
      {cat=vp, ... } ]
```

```
{cat=vp, ... }                                            (6)
    [ ^{cat=v, ... },
      ^{cat=np, ... } ]
```

```
{cat=np, defness=D, ... }                                 (7)
    [ !{cat=detp, defness=D, ... },
      {cat=n, ... },
      ^{cat=pp, ... } ]
```

```
{cat=detp, defness=D, ... }                               (8)
    [ {cat=det, defness=D, ... } ]
```

Let us first concentrate on the np part of the descriptor (4), and see how it is completed by the generator.

By unification with rule (7), we get the following structure where a node for the determiner phrase has been added:

```
{cat=np, defness=definite, ... }
    [ {cat=detp, defness=definite, ... },
      {cat=n, lu=woman, ... } ]
```

Now, rule (8) adds the determiner to the object:

```
{cat=np, defness=definite, ... }
    [ {cat=detp, defness=definite, ... }
      [ {cat=det, defness=definite, ... } ],
      {cat=n, lu=woman, ... } ]
```

The inserted node for the determiner contains the information which enables the generator to find and add the feature for the lexical unit "the".

A structural representation of the definiteness feature has been created, and as the original information in (4) has also been validated, the generator has finished its construction of the np.

As the dominance relation between the s-node and the v-node given by the descriptor in (4) does not necessarily imply immediate dominance in the finished object, the generator can create a vp-node by application of rule (6). The original descriptor is completed by rule (5), which also establishes the immediate precedence relation between the np-node and the vp-node.

This gives us the target object depicted in (2), which has a much richer structure than the one provided by the descriptor. As the structure was autonomously created by the generator applying g-rules, the example illustrates how the generator can add linear precedence information and introduce new nodes to complete the dominance and precedence relations given in a descriptor.

5. Conclusion.

The E-Framework itself does not put any restrictions on the number of representational levels, and it does not prescribe any specific distribution of linguistic phenomena over levels.

Expressing linguistic knowledge in a simple and modular way is obviously crucial for the perspicuity, extensibility and repairability of any large-scale natural language processing system.

The tree to descriptor strategy used in the E-Framework makes it possible to express the complex two-way relation between a text and its abstract representation as a set of simple, economical and non-procedural descriptions. This has been achieved by giving the generator device the power to use the grammar rules defining a level of representation for other than just checking purposes. As a consequence, the task of the translator has been diminished in that full structural specifications need no longer be stated in the t-modules as well as in the grammars.

In our application, the advantage of the tree to descriptor strategy is that even when aiming at simple bi-lingual t-modules in a stratificational translation model, which tends to demand a number of structural changes to be performed within monolingual components, the description of these changes is rather simple.

Acknowledgements:

The E-Framework in its present form is the result of the joint effort of Eurotra's framework group, of which the authors are members. If, however, there are any misrepresentations in this presentation, only we are to be blamed.

We are particularly indebted to Sergei Perschke, who put forth the basic ideas of the E-Framework.

We are grateful to Hanne Ruus, University of Copenhagen, for having scrutinized and criticized the draft versions of our paper. Also, we want to thank our colleague Bente Maegaard for some useful comments.

References:

Appelo, L., Fellinger, C., Landsbergen, J. (1987): "Subgrammars, Rule Classes and Control in the Rosetta Translation System", in **ACL Proceedings 87, (European Chapter)**, University of Copenhagen, pp. 118-133.

Arnold, D. J., Krauwer, S., Rosner, M., des Tombe, L., Varile, G. B. (1986): "The <C,A>,T Framework in Eurotra: A Theoretically Committed Notation for MT", in **Proceedings of Coling 86**, University of Bonn, pp. 297-303.

Boitet, C., Guillaume, P., Quézel-Ambrunaz, M. (1978): "Manipulations d'arborescences et parallélisme: le système ROBRA", in **Proceedings of Coling 78**, University of Bergen.

Nagao, M., Tsujii, J., Nakamura, J. (1985): "The Japanese Government Project for Machine Translation", in **Computational Linguistics**, vol. 11, no. 2-3, pp. 91-110.

Vauquois, B., Boitet, C. (1985): "Automated Translation at Grenoble University", in **Computational Linguistics**, vol. 11, no. 1, pp. 28-36.

For general information on the Eurotra Research and Development Project see e.g.

Multilingua, vol. 5, no. 3, 1986.