6.1. Delimiting the problem

The TRANSLATOR machine translation (MT) project explores the knowledge-based approach to MT. The basic translation strategy is to extract meaning from the input text in a source language, SL, represent this meaning in a language-independent semantic representation and then render this meaning in a target language, TL. The knowledge representation language used in such a set-up is called, for historical reasons, interlingua (henceforth, IL).

TRANSLATOR's ultimate aim is achieving good quality automatic translation in a non-trivial subworld and its corresponding sublanguage. The philosophy of TRANSLATOR aims at the independence of the process of translation from human intervention in the form of the traditional pre- and/or post-editing. Interaction during the process of translation can be accommodated by this philosophy, but only as a temporary measure. Interactive modules will be plugged into the system pending the development of automatic modules for performing the various tasks as well as more powerful inference engines and representation schemata. This methodology facilitates early testing of a system (even before all the modules are actually built) and enhances its shorter-term feasibility.

This strategy is an extension of one of the approaches discussed, for example, by Carbonell and Tomita in Chapter 5 of this volume since it implies knowledge acquisition during the exploitation stage and also involves a broader class of texts as its input. Johnson and Whitelock (Chapter 8) are also proponents of the interactive approach, but their motivation is different, in that they perceive the human to be an integral part of their system even in its final incarnation. In any case, interactivity is not the central design feature of TRANSLATOR. The background of the TRANSLATOR MT project is presented in Tucker and Nirenburg (1984).

6.2. Configuration of TRANSLATOR

There are three procedural modules in TRANSLATOR: the analyzer, the augmentor and the synthesizer. The analyzer obtains the input text and produces a representation of it as a set of IL frames. A number of slots in these frames will remain unfilled, because a text practically never has all the
information about objects and events it describes spelled out — it is intended that the rest be inferred by the hearer (reader). In a computer model these inferences are made through the operation of the augmentor. This additional information is instrumental in narrowing the choice of TL correlates for an input text (see Chapter 1 for a discussion of choices). Finally, the synthesizer obtains the augmented IL text and produces a TL text.

Each of the procedural modules is supported by a number of TRANSLATOR's static knowledge clusters, as shown in Figure 1.

![Figure 1. Knowledge and processes in TRANSLATOR.](image)

In this chapter we will describe the design of TRANSLATOR's IL and illustrate how its grammar is used to represent the meaning of SL texts with the help of the knowledge of the ontology of the TRANSLATOR subworld, stored in the IL dictionary. In other words, the illustration will concern only the knowledge structures for the analyzer. The augmentor and the synthesizer will necessarily require significant additions to the IL dictionary (including representations of typical groupings of objects and events in a subworld). We see the compilation of the subworld knowledge base as a prolonged and incremental empirical process.\(^1\)

---

\(^1\) It is clear that in order for this process to succeed one has to come up with an appropriate methodology for the effort. It is common knowledge that people tend to be idiosyncratic in the ways they perform semantic analysis (which is exactly the type of work involved in IL dictionary compilation). Interactive aids, as described in Chapter 9, must be developed, and the experience of working with large dictionaries, even of a different type, as discussed in Chapter 14 must be taken into account.
IL is, in fact, a set of two knowledge representation languages: the Interlingua Dictionary Representation Language (DRL) and the Interlingua Grammar Representation Language (GRL), respectively. DRL is a language for describing the types of concepts that can appear in the subworld of translation. GRL is a language for representing the assertions about tokens of those types that actually appear in texts. DRL is used for writing the IL dictionary, while GRL is used for writing the IL grammar. The distinction between these two languages is similar to that between the description and assertion languages in KL-ONE (cf. Brachman and Schmolzoe, 1985). The ontology of the translation subworld is expressed in DRL, while the results of analyzing an SL text take the form of a set of GRL frames that incorporate instances of DRL frame types. Note that GRL is not used for representing the grammar of any SL or TL.

After discussing these languages we will very briefly sketch the structure the SL — IL dictionary to help us through an illustration of how the IL dictionary is used during the analysis stage.

6.3. DRL

The IL dictionary is a source of information for representing the meanings of SL texts. In it, one does not find any information pertaining to any particular SL or TL. Thus, it is only for reader convenience that most of DRL concepts as well as most of the members of the property sets used to describe properties of IL concepts were made to look like English words. This choice was made with the dictionary writers in mind. The other possibility would have been to assign non-suggestive identifiers to DRL concepts. This would have slowed down the process of dictionary compilation. The dictionary writers must do their best not to mix the semantics of an entry head in the IL dictionary with that of an English word whose graphical form is identical.

There are two kinds of entities in DRL: concepts and properties. Concepts are, roughly speaking, IL 'nouns' (objects) and IL 'verbs' (events). IL 'adjectives,' 'adverbs' and 'numerals' are represented by properties. These are organized as sets of property values indexed both by the name of the property set (e.g., 'color,' 'time' or 'attitude') and by the individual values, to facilitate retrieval. Property values pertain to specific concept types. Their tokens do not appear on their own in IL texts, but only as fillers of slots in the frames for concept tokens. Thus, for example, 'red' will be a potential filler for the 'color' property of a token of every physical object. The proper choice of primitive concepts in a real subworld is a notoriously difficult problem (cf. Hayes, 1979). We approach it as an empirical task and are prepared to revise our system when new insights and information are gathered. An explanation of the relationship between IL word types and
tokens follows. An illustration of the ways members of various natural language lexical classes are treated in the SL-IL dictionary may be seen in Table 2.

The IL dictionary is organized as a set of entries (concept nodes) interconnected through a number of link types (properties). However, the structural backbone of the dictionary is the familiar isa hierarchy with property inheritance. Note that most of the time the translation system will be working with terminal nodes in this hierarchy. But the nonterminal nodes play a special role in it. By representing sets of entries, thereby providing a link among a number of (related) concepts, they serve as the basis for a variety of inference-making procedures. Even more importantly, these ‘nonterminal entries’ constitute, together with the sets of various property values, the schema of the dictionary, the set of terms that are used to describe the semantics of the rest of the dictionary entries.

Like all other nodes in the hierarchy, nonterminal nodes represent dictionary entries, which means that they can also have tokens. This device comes handy when, on analyzing a segment of input, we conclude that a certain slot filler is unavailable in the text. At the same time, if we know the identities of other slot fillers in the frame, we can come to certain conclusions about the nature of an absentee. For instance, if the Agent slot of a certain mental process is not filled, we, by consulting the ‘agent-of’ slot of the nonterminal node ‘mental-process,’ can infer (or, rather, abduce) that, whatever it is, it must be a ‘creature.’ This knowledge helps in finding referents for anaphoric phenomena.

The dictionary entries represent IL concept and property types; IL texts consist of IL concept tokens (as well as IL clause and sentence tokens), organized in accordance with the IL syntax postulated by GRL. Every token of an IL concept stands in the is-token-of relationship to its corresponding type. Structurally both IL concept types and IL concept tokens are represented as frames. The frame for a type and the frame for a corresponding token are not identical in structure, though the intersection of their slot names is obviously non-zero. One must note, however, that even in this case the semantics of the slots in the dictionary frames is different from that of the corresponding slots in the text frame.

Some of the slot names in the type frames refer to the paradigmatic relationships of this concept type with other concept types. These are the type parameters of an IL dictionary entry. The rest of the information in an entry describes syntagmatic relationships that tokens of this particular type have with tokens of other types in an IL text. These are the token parameters. Among the type parameters one finds, for example, pointers in the isa hierarchy and such relationships as part-of.
The token-parameter slots in the dictionary entries contain either default values for the properties (the 'no-value' nil value is among the possible default choices) or acceptable ranges of values, for the purpose of validity testing. IL concept tokens, which are components of IL text, not its dictionary, have their slots occupied by actual values of properties; if information about a property is not forthcoming, then the default value (if any) is inherited from the corresponding type representations.

We now turn to the actual description of DRL. We will do this by presenting the top levels of the isa hierarchy of concepts in our subworld and listing the frames for high-level nodes. Next, we'll present examples of IL dictionary frames, including one complete path in the isa hierarchy, from the root to a terminal node.

Figure 2. A fragment of the isa network.
The hierarchy of Figure 2 depends on the subworld for which it is designed. It may be overdeveloped in some of its branches and underdeveloped in many others. This state of affairs corresponds to the strategy of working within a subworld.

6.3.1. Frames

\[
\text{all ::= ('all' (subworld' subworld*))}
\]

This is the root of the isa hierarchy. The slots in this frame mean that every node in the tree represents a concept that belongs to one or more subworlds.

\[
\text{event ::= ('event' ('isa' all) ('patient' object))}
\]

At this level we meet the 'isa' slot for the first time. This is the pointer to a node's parent in the hierarchy. Events, as one can see from Figure 2, are subdivided into processes and states. The only property common to all events is the conceptual case of 'patient.'

\[
\text{process ::= ('process' ('isa' event) ('is' process-sequence) ('part-of' process*) ('agent' creature) ('object' object) ('instrument' object) ('source' object) ('destination' object) ('preconditions' state*) ('effects' state*))}
\]

In addition to the conceptual case slots, the process frame contains information about preconditions and effects. These are states that must typically hold before and after the process takes place, respectively. A process can also be a component of other processes. Thus, for instance, move is a component of travel, fetch, insert, etc. The 'is' slot of a process frame contains either the constant primitive, if the process is not further analyzable in DRL, or the description of the sequence of processes which comprise the given process. The process-sequence is a list of process names connected by the operators sequential, choice and shuffle. In other words, a process may be a sequence of subprocesses (sequential), a choice among several subprocesses (choice), a temporally unordered sequence of subprocesses (shuffle) or any recursive combination of the above. This treatment of processes is inspired by Nirenburg, Reynolds and Nirenburg (1985). For the purposes of

\[\text{footnote}2\] This reflects our opinion that in the sentence John is asleep John is not an agent, but rather a patient. Note that 'patient' in DRL subsumes the semantics of 'beneficiary.'
machine translation it seems unnecessary to introduce a more involved temporal logic into consideration for the ‘is’ slot.

\[
\text{physical-process} ::= (\text{'physical-process'} \\
\quad (\text{'isa'} \text{ process}) \\
\quad (\text{'object'} \text{ object}))
\]

\[
\text{mental-process} ::= (\text{'mental-process'} \\
\quad (\text{'isa'} \text{ process}) \\
\quad (\text{'agent'} \text{ creature}) \\
\quad (\text{'object'} \text{ object} | \text{event}))
\]

Only creatures can be fillers for the ‘agent’ slot. Mental processes classify into reaction processes (cf. the English ‘please’ or ‘like’), cognition processes (‘deduce’) and perception processes (‘see’). Objects of mental processes can be either objects, as in (1) or events, as in (2).

(1) I know John
(2) I know that John has traveled to Tibet

\[
\text{speech-process} ::= (\text{'speech-process'} \\
\quad (\text{'isa'} \text{ process}) \\
\quad (\text{'agent'} \text{ person}) \\
\quad (\text{'patient'} \text{ person}^* | \text{organization}^*) \\
\quad (\text{'object'} \text{ event} | \text{object}) \\
\quad (\text{'source'} \text{ 'agent'}) \\
\quad (\text{'destination'} \text{ 'patient'})
\]

Some of the varieties of speech processes recognized by DRL are listed in Figure 2. The ‘agent’ slot filler has the semantics of the speaker. The ‘patient’ is the hearer. Note that there is a possibility for the hearer to be a group or an organization, as in (3). The ‘agent’ is the ‘source’ and the ‘patient’ is the ‘destination’ of a speech process.

(3) I promised the band to let them have a ten-minute break every hour

\[
\text{state} ::= (\text{'state'} \\
\quad (\text{'isa'} \text{ event}) \\
\quad (\text{'part-of'} \text{ state}^*))
\]

The actant in states, which is the patient rather than the actor, is inherited from the event frame.

\[
\text{object} ::= (\text{'object'} \\
\quad (\text{'isa'} \text{ all}) \\
\quad (\text{'part-of'} \text{ object}^*) \\
\quad (\text{'consists-of'} \text{ object}^*) \\
\quad (\text{'belongs-to'} \text{ creature} | \text{organization}) \\
\quad (\text{'object-of'} \text{ process}) \\
\quad (\text{'patient-of'} \text{ event}) \\
\quad (\text{'instrument-of'} \text{ event}) \\
\quad (\text{'destination-of'} \text{ event}) \\
\quad (\text{'source-of'} \text{ event})
\]

The ‘...-of’ slots are used for consistency checks.
6.3.2. Properties

Property values are primitive concepts of IL used as values for slots in concept frames. We give here just an illustration of these. Many more exist and will be used in the implementation.

\[
\begin{align*}
\text{size-set} & : = \text{infinitesimal} \mid \ldots \mid \text{huge} \\
\text{color-set} & : = \text{black} \mid \ldots \mid \text{white} \\
\text{shape-set} & : = \text{flat} \mid \text{square} \mid \text{spherical} \ldots \\
\text{material-set} & : = (\text{gold (specific-gravity 81) (unit-value 228)}) \mid \ldots \\
\text{subworld-set} & : = \text{computer-world} \mid \text{business-world} \mid \text{everyday-world} \ldots \\
\text{boolean-set} & : = \text{yes} \mid \text{no} \\
\text{texture-set} & : = \text{smooth} \mid \ldots \mid \text{rough} \\
\end{align*}
\]

\[
\text{properties} ::= (\text{properties}' \\
\quad \text{'none'} \\
\quad (\text{'size'} \text{ size-set}) \\
\quad (\text{'color'} \text{ color-set}) \\
\quad (\text{'shape'} \text{ shape-set}) \\
\quad (\text{'texture'} \text{ texture-set}) \\
\quad (\text{'belongs-to'} \text{ creature} \mid \text{organization}) \\
\quad (\text{'part-of'} \text{ object} \mid \text{event}) \\
\quad (\text{'consists-of'} \text{ object} \mid \text{event}) \\
\quad (\text{'power'} \text{ real}) \\
\quad (\text{'speed'} \text{ real}) \\
\quad (\text{'mass'} \text{ real}) \\
\quad (\text{'edibility'} \text{ boolean-set}) \\
\quad (\text{'made-of'} \text{ material-set}) \\
\quad \ldots)
\]

6.3.3. From the root to a leaf

A path of concept representations from the root to a leaf node is presented below.

\text{all - object - pobject - + alive - creature - person - computer-user}

Frames for 'all' and 'object' see above.

\[
\text{pobject} ::= (\text{pobject}' \\
\quad (\text{'isa'} \text{ object}) \\
\quad (\text{'object-of'} \ (\text{Take Put})) \\
\quad (\text{'size'} \text{ size-set}) \\
\quad (\text{'shape'} \text{ shape-set}) \\
\quad (\text{'color'} \text{ color-set}) \\
\quad (\text{'mass'} \text{ integer}))
\]

The '+' sign in slots means all inherited information plus the contents of the current slot.

\[
\text{+ alive} ::= (\text{+ alive}' \\
\quad (\text{'isa'} \text{ pobject}) \\
\quad (\text{'edibility'} \text{ boolean-set}))
\]

\[
\text{creature} ::= (\text{creature}' \\
\quad (\text{'isa'} + \text{alive}) \\
\quad (\text{'agent-of'} \text{ (Eat Ingest Drink Move Attack)}) \\
\quad (\text{'consists-of'} \text{ (Head Body)}) \\
\quad (\text{'object-of'} \ (\text{+ (Attack)}) \\
\quad (\text{'power'} \text{ real}) \\
\quad (\text{'speed'} \text{ real}))
\]
person ::= ('person'
  ('isa' creature)
  ('agent-of' (+ (Take Put Find Speech-process Mental-Process)))
  ('source-of' Speech-process)
  ('destination-of' Speech-process)
  ('consists-of' (+ (Hand Foot ...)))
  ('size' medium)
  ('shape' oblong)
  ('power' 50)
  ('speed' 50)
  ('mass' 55))

computer-user ::= ('computer-user'
  ('isa' 'person')
  ('agent-of' (+ (Operate)))
  ('subworld' computer-world))

The complete frame of the leaf of this path, ‘computer-user’, including all inherited slots and default values is listed below. In reality frames like this do not exist, because the tokens of this type do not contain all the possible slot fillers.

(computer-user
  ('isa' 'person')
  ('agent-of' (Operate Take Put Find Speech-process Mental-Process
    Eat Ingest Drink Move Attack))
  ('object-of' (Find Mental-process Speech-process Attack Take Put))
  ('destination-of' Speech-process)
  ('size' size-set)
  ('shape' shape-set)
  ('color' color-set)
  ('edibility' boolean-set)
  ('source-of' Speech-process)
  ('consists-of' (Hand Foot Head Body))
  ('size' medium)
  ('shape' oblong)
  ('power' 50)
  ('speed' 50)
  ('mass' 55)
  ('subworld' computer-world))

6.4. GRL

In the previous section we dealt with the IL dictionary. This section is devoted to the syntax of IL ‘text.’ Unlike a natural language text, an IL text is not linear. It is a (potentially) complex network of IL sentences, interconnected by IL discourse markers. An IL sentence is represented as a frame with slots for each of any number clauses (that are represented as frames themselves) as well as for speech act and focus information. The IL clause is the place where event tokens are put into the modal and spatio-temporal context. The tokens for events and their actant objects that appear in IL texts are produced by obtaining tokens of the appropriate concept types in the dictionary and augmenting them by various property values identified during SL text analysis. It follows that the slots whose values depend on contextual meaning (e.g., negation and modality) appear only in GRL frames for event and object tokens, and not in DRL.
At the same time, there are regular correspondences between units of GRL and DRL. The values of the properties in entity tokens typically correspond to the data types listed as fillers for the corresponding slots in DRL frames. Thus, for instance, the color property slot in the DRL frame for 'flower' can be occupied by a list (white yellow blue red purple pink, etc.), the one for 'snow,' on the other hand, will contain only the one-element list (white). At the same time, 'rose11' will have the value 'red' as the contents of its 'color' slot. This underscores the difference in the semantics of similarly named slots in DRL and GRL.

6.4.1. Text

text ::= sentence | (discourse-structure-type text text+)

The above means that an IL text is either an empty string, a single sentence, or a number of sentences interconnected through discourse structure markers.

6.4.2. Sentence

sentence ::= ('sentence-token'
  ('clauses' clause*)
  ('subworld' subworld)
  ('modality' modality)
  ('focus' focus)
  ('speech-act' speech-act))

Every sentence is declared to contain a speech act. Thus, we will represent (4) as (5), provided we can infer the identities of the speaker and the hearer, as well as the identity of the process:

(4) I'd rather not do it.
(5) The boss ordered Employee X not to agree to the terms of Sales Offer Y.

Both direct and indirect speech acts are represented with the help of speech process tokens. With direct speech acts, the information to be put into the sentence frame is present in the text, while with indirect speech acts it has to be inferred.

Thematic information about the sentence includes the values for the 'given' and the 'new' (or focus) slots in the sentence frame. Both values can be pointers to a concept, a property of a concept, or an entire clause. The value of the modality slot for the IL sentence is chosen from the set of modalities. The subworld slot is a marker that shows that the sentence belongs to a certain 'semantic field.' In TRANSLATOR the designated topic for translations is the world of technical texts about computers. In broader environments the subworld information will be helpful to prune unneeded inference paths.
6.4.3. Clause

('clause-token'
  ('discourse-structure' discourse-structure)
  ('focus' focus)
  ('modality' modality)
  ('time' time)
  ('space' space)
  ('event' event)
  ('quantifier' quantifier2)
  ('subworld' subworld))

The major difference between the interlingua clauses and events is that clauses contain information that actually appears in the input text (augmented by anaphora resolution), while events can be either contained in the input or inferred from it.

A clause may be connected discourse-wise not only with another clause but also with an object or an event, as well as with a sentence, a paragraph or even a whole text; also note that discourse structure assigns the given clause as one of the two arguments in the discourse structure; one clause can be an argument in more than one discourse-structure expression.

6.4.4. Process

('physical-process-token-id'
  ('is-token-of' string)
  ('agent' object-token)
  ('object' object-token)
  ('patient' object-token)
  ('instrument' object-token)
  ('source' object-token)
  ('destination' object-token)
  ('negation' negation)
  ('quantifier' quantifier2)
  ('phase' phase-set)
  ('manner' manner-set)
  ('space' space)
  ('time' time)
  ('subworld' subworld-token))

An actual process token is represented as follows:

(move21
  ('is-token-of' move)
  ('is' primitive)
  ('agent' person12)
  ('object' person12)
  ('source' (in house2))
  ('destination' (in house3))
  ('negation' nil)
  ('quantifier' nil)
  ('phase' static)
  ('manner' easily)
  ('part-of' travel5)
  ('time' (before 1700))
  ('subworld' everyday-world))
6.4.5. State

Events in IL have a property of ‘phase:’ they are either ‘static,’ ‘beginning’ or ‘end.’ This device is needed to represent changes of state. Changes of state are sometimes represented as a separate class of processes. The solution in IL may be more economical.

6.4.6. Object

A typical frame for an object token in GRL is as follows. The ‘string’ in the ‘is-token-of’ slot stands for the name of the corresponding object type.

An example object token follows:

(person23
 ('is-token-of' person)
 ('subworld' everyday-world)
 ('negation' no)
 ('quantifier' any)
 ('power' 50)
 ('speed' 50)
 ('mass' 55))

Note the difference from DRL object frames. No ‘...-of’ slots here. More emphasis on syntagmatic relationships and default overriding.

6.4.7. Time

\[
time : = \text{absolute-time} \mid \text{relative-time}
\]

\[
\text{absolute-time} : = \left( \text{‘time’} \right) \\
\quad \left( \text{‘quantifier’ quantifier2} \right) \\
\quad \left( \text{‘point’ integer} \right) \mid \\
\quad \left( \text{‘interval-begin’ integer} \right) \\
\quad \left( \text{‘interval-end’ integer} \right)
\]

\[
\text{relative-time} : = \left( \text{‘time’} \right) \\
\quad \left( \text{temporal-operator event} \right) \\
\quad \left( \text{‘quantifier’ quantifier2} \right)
\]

\[
\text{temporal-operator} : = \text{simultaneous} \mid \text{before} \mid \text{during} \mid \text{around} \mid \text{always} \mid \text{none}
\]

Relative time markers will predominantly appear in texts.
6.4.8. Space

space ::= absolute-space | relative-space
absolute-space ::= ('space
  ('quantifier' quantifier2)
  ('coordinate1' real)
  ('coordinate2' real)
  ('coordinate3' real))

relative-space ::= ('space'
  (spatial-operator object)
  ('quantifier' quantifier2))
spatial-operator ::= left-of | equal | between | in | above | near | none

As in the case of time, relative (topological) space specifications will predominate in texts.

6.4.9. Slot operators

quantifier1 ::= all | any | most | many | some | few | 1 | 2 | ...
quantifier2 ::= hardly | half | almost | completely

6.4.10. Modality

modality ::= ('modality' modality-set)
modality-set ::= real | desirable | undesirable | conditional | possible | impossible | necessary

6.4.11. Focus

focus ::= ('given'
  ('object' obj) |
  ('event' event) |
  ('clause' clause) |
  ('quantifier' event-quantifier | quantifier))
  ('new'
  ('object' obj) |
  ('event' event) |
  ('clause' clause) |
  ('quantifier' event-quantifier | quantifier))

The thematic information, together with the discourse structure and speech act information, explicitly represents the rhetorical force of SL texts. It is the lack of this type of knowledge in most IL-oriented systems that led many MT researchers to declare that SL traces are necessary in the internal representation. Table 1 illustrates the influence of the focus information on distinguishing between sentences (all of them paraphrases of the sample sentence (6), see below) that have the same propositional content and would have to be judged indistinguishable by an analysis system that did not tackle rhetorical meaning.
<table>
<thead>
<tr>
<th>Entity in Focus</th>
<th>Form of Input Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object: 'Database'</td>
<td>Data such as the above, that is stored more or less permanently in a computer, we term a DATABASE</td>
</tr>
<tr>
<td>Object: 'Data'</td>
<td>What is stored more or less permanently in a computer and what we term a database is DATA; for an example of such data see above</td>
</tr>
<tr>
<td>Clause: '[data that is] stored more or less permanently in a computer'</td>
<td>Data, such as the above, which we term a database, is stored more or less permanently in a computer</td>
</tr>
<tr>
<td>Clause: '[data is] such as the above'</td>
<td>It is data such as the above, stored more or less permanently in a computer, that we term a database</td>
</tr>
</tbody>
</table>

Table 1. An illustration of focus distinctions.

6.4.12. Discourse structure

discourse-structure ::= (discourse-structure-type
  (clause clause-n | sentence | text) |
  (clause-n | sentence | text clause)* )

discourse-structure-type ::= none | temp | equiv | +expan | -expan |
  condi | +simil | -simil | choice

For a more detailed description of the discourse cohesion markers in TRANSLATOR see Tucker et al., 1986.

A clause may be connected discourse-wise not only with another clause but also with a sentence, a paragraph or even a whole text; also note that discourse structure assigns the given clause as one of the two arguments in the discourse structure; one clause can be an argument in more than one discourse-structure expression.

6.4.13. Speech act

speech-act ::= ('speech-act'
  ('type' speech-process)
  ('direct?' yes | no)
  ('speaker' object)
  ('hearer' object+)
  ('time' time)
  ('space' space))

Every IL sentence features a speech act, irrespective of whether it was overtly mentioned in the SL text. If it was, it is represented through a token of a speech process. Otherwise, it is inferred. The time and space of the
speech act can be quite different from that of the proposition which is the information transferred through this speech act.

6.4.14. Other slots and slot fillers

negation ::= boolean-set
referred-set ::= above | below | object-token
manner-set ::= difficulty | attitude
difficulty ::= easily | ... | difficulty
attitude ::= caring | ... | nonchalantly
phase-set ::= static | beginning | end

6.5. Knowledge about SL

A word in the SL text potentially contains clues for a number of types of meaning: syntactic, semantic or pragmatic. The analysis modules require a number of dictionaries for operation. One of them is syntactic and contains information necessary for the syntactic parsing modules to operate. Another is semantically-pragmatic and contains information to be used by the semantic, discourse, anaphora and speech act parsers. Table 2 illustrates what IL categories correspond to sample categories of English grammar.

<table>
<thead>
<tr>
<th>SL (English) Category</th>
<th>IL Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOUN</td>
<td>object frame</td>
</tr>
<tr>
<td>ADJECTIVE</td>
<td>slot in object frame; state frame</td>
</tr>
<tr>
<td>VERB</td>
<td>action or state frame</td>
</tr>
<tr>
<td>MODAL</td>
<td>modality marker in clause and sentence frames</td>
</tr>
<tr>
<td>ADVERB</td>
<td>slot in action or state frame (time, space, ...); state marker ('he moved nonchalantly'); may introduce a separate clause frame</td>
</tr>
<tr>
<td>DETERMINER</td>
<td>marker in given/new; directs generation of new instances of objects vs. reference to existing instances</td>
</tr>
<tr>
<td>CONJUNCTION</td>
<td>marker of sentence type; marker of cohesion</td>
</tr>
<tr>
<td>PREPOSITION</td>
<td>case marker; state marker; may introduce a separate clause frame</td>
</tr>
<tr>
<td>DEMONSTRATIVE</td>
<td>marker of deixis</td>
</tr>
<tr>
<td>NUMERAL</td>
<td>a quantifier operator on IL NPs</td>
</tr>
<tr>
<td>PRONOUN</td>
<td>marker of deixis; reference to object</td>
</tr>
</tbody>
</table>
6. The structure of interlingua in TRANSLATOR

<table>
<thead>
<tr>
<th>SL (English) Category</th>
<th>IL Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>marks boundaries of case slot values</td>
</tr>
<tr>
<td>VP</td>
<td>helps mark boundaries</td>
</tr>
<tr>
<td>PP</td>
<td>helps mark boundaries</td>
</tr>
<tr>
<td>CLAUSE</td>
<td>fills clause frame</td>
</tr>
<tr>
<td>S</td>
<td>fills sentence frame</td>
</tr>
<tr>
<td>PARENTHETICALS</td>
<td>discourse markers; connector clues for sentences</td>
</tr>
</tbody>
</table>

Table 2. Selected categories of English with their IL counterparts.

A number of analysis modules will be employed by TRANSLATOR for extracting the above meanings from SL texts. We will not discuss them in any detail in this chapter, but will assume their existence in order to perform a manual trace of TRANSLATOR's analysis.

6.6. A sample analysis

Given the IL dictionary and grammar, we can now sketch the process of using these knowledge structures for analyzing an SL text. The system makes use of a number of analysis knowledge sources, as shown in Table 3. The emphasis in the discussion that follows is not on a particular control structure for analysis, but rather on the identity of the sources of knowledge that are used to derive IL representations (texts). Therefore, the central point is the expressive power of the interlingua.

<table>
<thead>
<tr>
<th>KS name</th>
<th>input</th>
<th>output</th>
<th>background knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>morphological analyzer</td>
<td>SL word token</td>
<td>SL word type + form specification</td>
<td>SL form formation paradigms + exception lists</td>
</tr>
<tr>
<td>syntactic constituent detector</td>
<td>SL sentence</td>
<td>boundaries of constituents in SL sentence</td>
<td>SL grammar and syntactic dictionary</td>
</tr>
<tr>
<td>dependency structure detector</td>
<td>SL sentence</td>
<td>syntactic dependency structure of SL sentence</td>
<td>SL grammar and syntactic dictionary</td>
</tr>
<tr>
<td>functional structure detector</td>
<td>SL sentence</td>
<td>functional structure of SL sentence</td>
<td>SL grammar and syntactic dictionary</td>
</tr>
<tr>
<td>event detector</td>
<td>SL sentence</td>
<td>IL frame token</td>
<td>SL-IL dictionary + results of syntactic analysis + IL dictionary</td>
</tr>
<tr>
<td>KS name</td>
<td>input</td>
<td>output</td>
<td>background knowledge</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------</td>
<td>----------------------------------------------</td>
<td>--------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>case detector</td>
<td>SL sentence</td>
<td>IL frame token</td>
<td>SL-IL dictionary + results of syntactic analysis + IL dictionary</td>
</tr>
<tr>
<td>time detector</td>
<td>SL sentence + event token</td>
<td>filler for the time slot of event token</td>
<td>SL-IL dictionary + IL dictionary + IL grammar</td>
</tr>
<tr>
<td>space detector</td>
<td>SL sentence + event token</td>
<td>filler for the space slot of event token</td>
<td>SL-IL dictionary + IL dictionary + IL grammar</td>
</tr>
<tr>
<td>clause builder</td>
<td>event token</td>
<td>clause token</td>
<td>IL grammar</td>
</tr>
<tr>
<td>modality detector</td>
<td>SL sentence + clause token</td>
<td>filler for the modality slot of clause token</td>
<td>SL-IL dictionary + results of syntactic analysis + IL dictionary + IL grammar</td>
</tr>
<tr>
<td>focus detector</td>
<td>SL sentence + clause token</td>
<td>filler for the slots in the focus sub-frame of clause token</td>
<td>SL-IL dictionary + results of syntactic analysis + IL dictionary + IL grammar</td>
</tr>
<tr>
<td>discourse detector</td>
<td>SL sentence + clause token</td>
<td>filler for the discourse slot of clause token</td>
<td>SL-IL dictionary + results of syntactic analysis + IL dictionary + IL grammar + representations of previously translated sentences</td>
</tr>
<tr>
<td>sentence builder</td>
<td>a set of clause tokens</td>
<td>sentence token</td>
<td>IL grammar</td>
</tr>
<tr>
<td>speech act detector</td>
<td>SL sentence + clause tokens + sentence token</td>
<td>fillers for the subslots of the speech-act slot of a sentence token</td>
<td>IL grammar + IL representations of previously translated sentences</td>
</tr>
<tr>
<td>subworld detector</td>
<td>SL sentence + clause tokens + sentence token</td>
<td>filler for subworld slot of a sentence token</td>
<td>IL grammar + IL representations of previously translated sentences</td>
</tr>
<tr>
<td>ellipsis augmentor</td>
<td>SL sentence + clause tokens + sentence token + world knowledge base</td>
<td>inferred fillers for various clause, object or event slots reference to which was omitted in SL text</td>
<td>IL grammar + IL representations of previously translated sentences</td>
</tr>
</tbody>
</table>
Table 3. Analysis knowledge sources in TRANSATOR

<table>
<thead>
<tr>
<th>KS name</th>
<th>input</th>
<th>output</th>
<th>background knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>anaphora resolver</td>
<td>SL sentence + clause tokens +</td>
<td>inferred fillers for various clause,</td>
<td>IL grammar + IL representations of previously</td>
</tr>
<tr>
<td></td>
<td>sentence token + world knowledge</td>
<td>object or event slots reference to</td>
<td>translated sentences</td>
</tr>
<tr>
<td></td>
<td>base</td>
<td>which was indirect in SL text</td>
<td></td>
</tr>
<tr>
<td>augmentor</td>
<td>IL text with some frame slots</td>
<td>IL text with fewer slots unfilled</td>
<td>world knowledge base + IL grammar</td>
</tr>
<tr>
<td></td>
<td>unfilled</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The background knowledge for the analyzer includes 1) the IL dictionary; 2) the IL grammar; 3) the SL-IL dictionary and 4) the SL grammar.

The input to the analysis is an SL text; its output is an IL text. The aim of the processing is to make the IL text ‘mean’ the same as the SL text.

Consider the example sentence (6). We assume that this sentence is the first input to the system in a translation session.

(6) Data, such as the above, that is stored more-or-less permanently in a computer, we term a database.

The system starts by evoking a morphological analyzer for SL and a dictionary look-up knowledge source which is responsible for retrieving the entries for all the words in the input text. Only a fraction of the SL words in (6) will cause the instantiation of new tokens of IL concepts.

The syntactic constituent detector knowledge source determines boundaries of syntactic constituents in (6). Another knowledge source is entrusted with augmenting the representation by eliminating elliptical phenomena, so that the clauses acquire the fullest possible complement of arguments. A special knowledge source deals with the anaphoric reference, as, for instance, seen in the first of the relative clauses of (6). Thus, the three clauses are identified (cf. (7)).

(7) Data is such as the above.
Data is stored more or less permanently in a computer.
Data is a database.

Additional knowledge sources come into play at this point, in order to perform the following tasks:

- extract the event names from the SL text and, for each event
- translate it into IL
- augment the representation by inserting the IL translations of the event actants and other relevant property slots (thus bringing it up to the IL clause level)

3 The clauses are listed in their ‘reconstructed’ form, for simplicity.
embellish the representation by specifying the modality, focus, IL sentence-level discourse structure and speech act information (thus reaching the IL sentence level of representation) and, finally,
connect this representation through IL text-level discourse connectors (thus reaching the level of IL text).

The three events present in (7) are identified by the event-detector knowledge source, as the following state tokens.

(state1
  (is-token-of be-example-of)
  (phase ?)
  (patient ?)
  (counterpatient ?)
  (time ?)
  (space ?)
  (subworld ?))

(state2
  (is-token-of be-in)
  (phase ?)
  (patient ?)
  (counterpatient ?)
  (time ?)
  (space ?)
  (subworld ?))

(state3
  (is-token-of be-defined-as)
  (phase ?)
  (patient ?)
  (counterpatient ?)
  (time ?)
  (space ?)
  (subworld computer-world))

Independently of the event detector operation, the case detector knowledge source produces object tokens for all the candidates present in the input. Three objects are involved in sentence (6). These objects are: data, computer and database. The representations for the corresponding tokens are as follows. Note the difference in the information contained in the object type frame in DRL and the following GRL frames. No default values for DRL object slots are overruled in these particular tokens.

(object1
  (is-token-of data)
  (subworld computer-world)
  (negation nil)
  (quantifier nil)
  (properties none))

(object2
  (is-token-of computer)
  (subworld computer-world)
  (negation nil)
  (quantifier any)
  (properties none))

(object3
  (is-token-of database)
  (subworld computer-world)
  (negation nil)
The specialized knowledge sources dealing with additional clause properties (subworld, phase, time and space in our example) produce the necessary fillers for their slots. After the event representations are augmented by the translations of their actants and additional properties they look as follows.

(state1
  (is-token-of be-example-of)
  (phase static)
  (patient object1)
  (counterpatient (referent above))
  (time always)
  (space none)
  (subworld computer-world))

(state2
  (is-token-of be-in)
  (phase beginning)
  (patient object1)
  (counterpatient (computer any))
  (time always)
  (space none)
  (subworld computer-world))

(state3
  (is-token-of be-defined-as)
  (phase beginning)
  (patient object1)
  (counterpatient (object3 any))
  (time always)
  (space none)
  (subworld computer-world))

Note that since this is the first sentence in a text, there is no hope of restoring the referent of the counterpatient slot in the token of 'be-example-of.'

Once the events are analyzed, TRANSLATOR proceeds to assemble the clause representations. The knowledge sources active at this stage are the focus, modality and discourse structure detectors. The time and space values are inherited from the representation of the corresponding event. The three clauses are represented as follows.

(clause1
  (discourse-structure (+ expan clause1 clause3))
  (event state1)
  (focus state1 . counterpatient)
  (modality real)
  (subworld computer-world)
  (time always)
  (space none))

(clause2
  (discourse-structure (+ expan clause2 clause3))
  (event state2)
  (focus time)
  (modality conditional)
  (subworld computer-world)
  (time always)
  (space (in object1 object2)))
When the IL clauses are ready, the IL sentence knowledge sources can finish their activity. A number of slot fillers propagate to the level of the sentence from the clause level. The main clause detector and the speech act detector are the typical knowledge sources at the sentence level. Also note that the focus information at the sentence level is not necessarily inherited from the clause level. The sentence token produced by the TRANSLATOR analyzer for (6) is as follows.

(sentence
 (main-clause clause3)
 (clauses clause1 clause2)
 (subworld computer-world)
 (modality real)
 (focus object3)
 (speech-act (type definition)
  (direct? no)
  (speaker author)
  (hearer reader)))

The text frame is at this point trivial: the text is just one sentence long so far. Therefore, there are no discourse markers to be detected by the main text level knowledge source: the text discourse marker detector.

The sentence (6) has been represented in IL as a network of frames. The process will be repeated for all the rest of the sentences in an input text. Then the IL text will be given as input to the generator group of knowledge sources. The design of the generator will be reported elsewhere. Suffice it to say now that for the above IL representation of (6) we will allow a number of possible TL translations that we will consider equally acceptable. For English as TL consider the translations (8) in addition to the original SL sentence (6). All of these are, naturally, acceptable paraphrases.

(8) (a) Data of the type described above is called a database if it is stored for significant periods of time inside a computer.

(b) Such data, stored in a computer for a long time, is a database.

(c) A database is defined as data, similar to the above, stored relatively permanently in a computer.

(d) Databases are collections of such data stored for long periods of time in computer memory.

4 Or a section of the input text, when it is too extensive. The questions of engineering-oriented appropriateness, such as deciding where to put the cutoff point for accumulating analyzed text before going to the generator module, are postponed till implementation time.
6. The structure of interlingua in TRANSLATOR

(e) We define a database as data of the kind illustrated above, stored inside a computer.

Many more variants are possible.

6.7. Conclusion

IL-based MT projects seem to be the most scientifically challenging in the family of approaches to MT. They aim at fully automatic translation, and therefore do not promise immediate feasibility. We would like to conclude this chapter by presenting a number of critical opinions about IL-based projects, with our comments.

Opinion. It is unnecessary to extract the full meaning from the SL text in order to achieve adequate MT.

Comment. With a large portion of input sentences, especially in carefully selected subworlds, an MT system can do without an involved semantic analysis. Many sentences, however, will involve at least the ambiguities for word choice in synthesis, and this will necessitate either semantic analysis or post-editing. Machines, unlike humans, cannot on demand produce interpretations of an input text at an arbitrary depth exactly sufficient for understanding. Therefore, if one aims at fully automatic translation, one has to design the system so that it performs semantic analysis to the maximum necessary depth for all sentences involved. One can, of course, think of designing a system that can decide how deeply each sentence can be analyzed semantically in an attempt to minimize semantic analysis. We maintain that the decision making involved in such an enterprise is as complex as the initial problem of uniformly deep semantic analysis.

Opinion. It is not necessary to finish processing the input sentence before starting the translation. Indeed, people very often do this (consider interpreters) with very good results.

Comment. This opinion is based on introspection. The real thought processes that go on in the heads of translators and interpreters have not been studied scientifically. It seems highly probable that this behavior is knowledge-based. Also, the decision to start translating before finishing the processing of input is a function of the translator’s belief about the cost of a possible error. Indeed, the more ‘important’ the translation is, the more one would be disinclined to indulge in this type of activity.

What makes such preemptive moves possible is the translators’ extensive knowledge about the subject matter of the text (speech), about the speech situation as well as their expert knowledge about the process of translation in general. A subproject of TRANSLATOR is devoted to building an expert system that embodies the knowledge about translators’ craft, to be used as the main troubleshooter of the system.
Opinion. Approaches to MT based on AI (such as the interlingua-based approach described here) do not pay sufficient attention to the syntactic analysis of SL, while syntactic information is important for MT. Instead of translating one ends up with paraphrasing. The structure of the SL text, when used in addition to TL in MT, governs the choice of one of the paraphrases. Therefore, one needs to retain information about SL till the stage of TL synthesis.

Comment. Syntactic structure of texts in SL conveys meaning. This meaning is extracted by the analyzer in a knowledge-based system, with the help of knowledge about the syntax of SL. This is done, commensurately with the general scope of the effort, in all knowledge-based MT systems. The difference is that no results of syntactic analysis are stored by most of such systems (cf., however, Lytinen, 1984). This decision stems from the conviction that the syntactic structures of SL do not need to play a role in determining the syntactic structures to be used in generating a TL text.

If one accepts the premise that it is not reasonable to expect the syntactic structures of an SL sentence and its TL translation to be similar, then attempting to catalog the relations among the various syntactic structure trees in SL and TL seems to be redundant, if possible at all. Instead, one should devise a representation for meanings conveyed by the syntax of SL and then provide (SL-independent) rules for building syntactic structure of TL from this representation. The information about the thematic structure of SL text and its discourse parameters are good examples of what can be expressed by syntax in SL.

If we assume that the common (invariant) core of a set of paraphrases is their Θ-structure (set of conceptual case slots with values filled), together with such verbal properties as tense, aspect and modality, then, indeed, the current experimental knowledge-based MT systems are producing a sentence which is a(n unmarked) paraphrase of an SL sentence. If, however, one extracts from the SL text the information about speech acts, discourse and topic/focus, one obtains additional constraints that help distinguish between the paraphrases and choose the most appropriate one. We claim that the above constraints have at least the same power as the systems in which meanings in SL and TL are put in correspondence through extensive enumeration (in the bilingual dictionaries) of possible combinations, augmented by syntactic information. In other words, a system like TRANSLATOR does not produce paraphrases any more than a transfer system that uses the syntax of SL in translation proper.

Opinion. IL-based approaches lead to an overkill because no peculiarities of SL (and of the relationship between, or contrastive knowledge of, SL and TL) can be used in translation. Some languages have quite a lot in common in their syntax and meaning distribution. It is wasteful not to use this
additional information in translation.

Comment. While such insights can sometimes be detected and used, most of them come from human intuition, and cannot be taken advantage of in MT systems, which are not typically built as models of human performance. We also believe that it is wrong to imply that discovery, representation and manipulation of these pieces of contrastive knowledge can be simpler or, in fact, distinct from involved semantic analysis.

Opinion. Interlingua reflects the semantics of just one language, English. It will not be automatically extensible to describe the semantics of other languages, because they have different perceptions of the world.

Comment. Indeed, the interlingua dictionary will be updated and revised even during the exploitation phase of a knowledge-based machine translation system. It would be nice to be able to come up with a language of atomic units of sense, à la Hjelmlev's semes, but we cannot hope to. Therefore, IL will at any given moment provide a semantic analysis of limited grain size.

Opinion. Generation of TL is a relatively simple problem for which very little or no knowledge other than lexical or syntactic is needed.

Comment. Generation requires non-trivial decision making, for instance, in the light of the discussion in the previous paragraph, or, for that matter, as regards the computational stylistics, which will have to be a part of the choice-making mechanisms in building TL texts.

Acknowledgement

The authors wish to thank Irene Nirenburg for reading, discussing and criticizing the numerous successive versions of the manuscript. Needless to say, it's we who are to blame for the remaining errors. This chapter is based upon work supported by the National Science Foundation under Grant DCR-8407114.