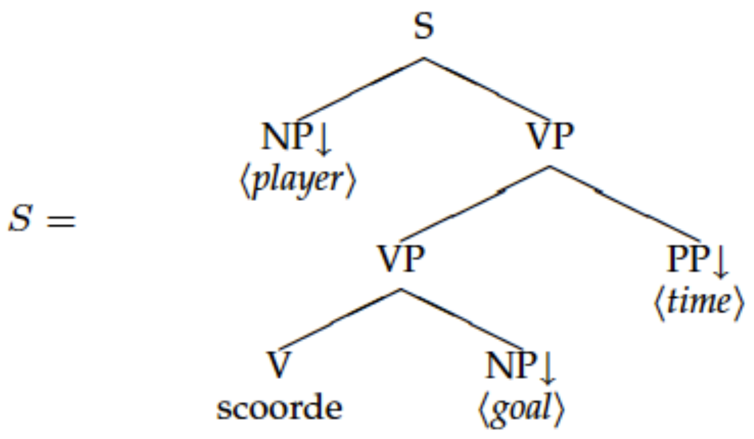
Generally speaking, template-based NLG models map a semantic representation (like slot-value pairs) directly onto surface form (natural language). To give a formal description, we choose one kind descripted in this article.

Formally a syntactic template $\sigma =< S, E, C, T >$. S represents the template syntactic tree with *open slots*, which are to be substituted by real words. E specifies what values could be fitted into each slot in S. C shows the condition when rule S could be selected. T is a topic set related to S. The above concepts are illustrated in the figure:



$$
\begin{aligned}
E = \; & \mathit{player} \leftarrow \text{ExpressObject}\ (\mathit{currentgoal.player}, P, \mathit{nom}) \\
& \mathit{goal} \leftarrow \text{ExpressObject}\ (\mathit{currentgoal}, P, \mathit{gen}) \\
& \mathit{time} \leftarrow \text{ExpressTime}\ (\mathit{currentgoal.time}) \\
C = \; & \text{Known}\ (\mathit{currentmatch.result}) \wedge \mathit{currentgoal} = \text{First}\ (\mathit{notknown,goallist}) \wedge \\
& \mathit{currentgoal.type} \neq \mathit{owngoal} \\
T = \; & \text{goalscoring}
\end{aligned}
$$

The algorithm to fill real words into open slots could be descripted as follows: first we iterate over all felicitous value combinations specified by the expression functions in E, and then select from them all the filled trees that match the input semantic (e.g. the requested slots and informable values) and follow Chomskyan binding rules i.e. an anaphor must be bound in the binding domain, a pronoun must be free in its binding domain and a R-expression must be free.

After the filtering process the remaining trees are valid ones. Randomly select one from the set and that's the surface form we generate.

Next we discuss how C and T take effect. For C we check all the current requirements of the semantic state and if a template σ passes the check we can choose it. For T we can define other rules to apply additional restrictions.