normalization

The *normalization* command extracts the property that is implicit in a declaration, reducing the declaration to be in terms of the carrier set of the underlying type (unlike *predication*).

$$
\begin{array}{rcl}
\ldots;\ i : e;\ \ldots \mid p & \Longrightarrow & \ldots;\ i : carrier(\tau\ e);\ \ldots \mid i \in e \land p \\
\ldots;\ i == e;\ \ldots \mid p & \Longrightarrow & \ldots;\ i : carrier(\mathbb{P}(\tau\ e));\ \ldots \mid i = e \land p \\
\ldots;\ e;\ \ldots \mid p & \Longrightarrow & \ldots;\ carrier(\tau\ e);\ \ldots \mid e \land p
\end{array}
$$

In the first two rules, it must be the name of the declaration that is inspected. In the last rule, that of a schema declaration, it must be the schema inclusion declaration that is inspected, not the schema expression within it—an extra click of button 1 is needed. The extracted property is that schema expression used as a predicate.

The *normalization* command can also be applied to declarations in the hypothesis part of a goal, the new predicate being generated as the first antecedent.

The *normalization* command can also be applied to a quantified predicate with a schema text whose | part is not *true*: the predicate of that | part is logically combined with the predicate in the • part.

$$
\begin{array}{rcl}
\forall\ ds \mid p_1 \bullet p_2 & \Longrightarrow & \forall\ ds \bullet p_1 \Rightarrow p_2 \\
\exists\ ds \mid p_1 \bullet p_2 & \Longrightarrow & \exists\ ds \bullet p_1 \land p_2 \\
\exists_1\ ds \mid p_1 \bullet p_2 & \Longrightarrow & \exists_1\ ds \bullet p_1 \land p_2
\end{array}
$$

See also *denormalization*.

1. Tactic example

"normalization" d_1 d_2

This example applies the *normalization* command to declarations d_1 and d_2.

IT 7-Apr-2000