What is wrong with defining local variable declaration as follows:
\[
\text{\texttt{var } } x: T \cdot P \quad \equiv \quad \forall x: T \cdot \exists x': T \cdot P
\]

§ Programs are implementable. Consider the program
\[
\text{\texttt{var } } x: \text{int} \cdot y:= x
\]
with global integer variables \( y \) and \( z \). Using the suggested definition,
\[
\text{\texttt{var } } x \cdot y := x
\]
\[
\equiv \quad \forall x \cdot \exists x'. x' = x \land y' = x \land z' = z
\]
\[
\equiv \quad \bot
\]
we get something unimplementable. If we had used the proper definition
\[
\text{\texttt{var } } x: \text{int} \cdot y := x
\]
\[
\equiv \quad \exists x, x'. x' = x \land y' = x \land z' = z
\]
\[
\equiv \quad z' = z
\]
we get something implementable, as we should. Less importantly, with the new
definition, a \textit{null} type is implementable:
\[
\text{\texttt{var } } x: \text{null} \cdot x := 0
\]
\[
\equiv \quad \forall x: \text{null} \cdot \exists x': \text{null} \cdot x' = 0 \land y' = y \land z' = z
\]
\[
\equiv \quad \top
\]
If we had used the proper definition
\[
\text{\texttt{var } } x: \text{null} \cdot x := 0
\]
\[
\equiv \quad \exists x: \text{null} \cdot \exists x': \text{null} \cdot x' = 0 \land y' = y \land z' = z
\]
\[
\equiv \quad \bot
\]
the \textit{null} type is unimplementable, as it should be. Note that the assignment is out-of-range.