**SRC Technical Note**
**1996 - 001**

**December 18, 1996**

# Refinement in State-Based Formalisms

Leslie Lamport

**Systems Research Center**
130 Lytton Avenue
Palo Alto, California 94301

http://www.research.digital.com/SRC/

Copyright ©Digital Equipment Corporation 1997. All rights reserved

# Refinement in State-Based Formalisms

Leslie Lamport
Digital Equipment Corporation
lamport@pa.dec.com

18 December 1996

## 1 A Tale of Two Clocks

A friend of ours, who was a brilliant mathematician, has been hospitalized because of long-term abuse of hallucinogenic drugs. We decide to give him a digital clock for his room. However, his doctor suggests that the hour and minute displays together might be too confusing. So, we put tape over the minute display, turning our gift into a digital hour clock.

We present the clock to our friend, who asks what it is supposed to do. Although he has lost his memory of everyday objects, his mathematical abilities are undiminished. So, we give him a mathematical explanation. Since the drugs have destroyed his sense of time, we do not have to relate the behavior of the clock to real time; we need only explain the sequence of numbers displayed by the clock. We describe the clock by saying that this sequence must be some sequence of values for the variable $hr$ allowed by the temporal-logic formula $S_H$ defined as follows.

$$\begin{aligned} Init_H &\triangleq hr \in \{1, \ldots, 12\} \\ N_H &\triangleq hr' = (hr \bmod 12) + 1 \\ S_H &\triangleq Init_H \wedge \Box N_H \end{aligned} \tag{1}$$

Being an expert in temporal logic, our friend understands that the formula $Init_H$ describes the initial state, and the formula $\Box N_H$ asserts that every successive pair of states satisfies the relation $N_H$, where unprimed values refer to the first state and primed values refer to the second. We end our visit, leaving him happily watching the clock and checking that it satisfies $S_H$.

When we next visit the hospital, the doctor tells us that our friend has improved considerably, so we can remove the tape and make the clock's minute display visible. We do so, and our friend requests a new specification that describes both the minute and hour displays. We choose the variable $min$ to represent the minute display, and we

give him the obvious formula $S_M$ defined as follows.[1]

$$
\begin{array}{lcl}
Init_M & \triangleq & (hr \in \{1, \ldots, 12\}) \ \wedge \ (min \in \{0, \ldots, 59\}) \\
N_M & \triangleq & \wedge \ min' = min + 1 \bmod 60 \\
 & & \wedge \ hr' = \textbf{if } min = 59 \textbf{ then } (hr \bmod 12) + 1 \textbf{ else } hr \\
S_M & \triangleq & Init_M \ \wedge \ \Box N_M
\end{array}
\tag{2}
$$

Our friend is puzzled and asks how removing the tape could change the behavior of the hour display. He explains that if removing the tape didn't change the hour display's behavior, then its sequence of values should still satisfy the original specification $S_H$. After all, $S_H$ does not imply the absence of a minute display, so it should be true for any sequence of values for $hr$ allowed by the new specification $S_M$. But it isn't. Formula $S_H$ asserts that the value of $hr$ should change in every successive state, while formula $S_M$ asserts that there are fifty-nine successive states with the same value of $hr$.

We realize that he is right; our specification $S_H$ is wrong. We must modify it to allow steps that leave $H$ unchanged. Because we might one day give him a new clock with a second display, we must also modify $S_M$ to allow steps that leave both $hr$ and $min$ unchanged. To save a bit of writing, we introduce the notation that $[N]_f$ is an abbreviation for $N \vee (f' = f)$, where $f'$ is the expression obtained by priming all the free variables in $f$. We now redefine $S_H$ and $S_M$ by

$$
\begin{array}{lcl}
S_H & \triangleq & Init_H \wedge \Box[N_H]_{hr} \\
S_M & \triangleq & Init_M \wedge \Box[N_M]_{\langle hr, min \rangle}
\end{array}
$$

Our mathematician friend understands that $\langle hr, min \rangle' = \langle hr, min \rangle$ iff $hr' = hr$ and $min' = min$, so the subscript $\langle hr, min \rangle$ means that $S_M$ allows steps that leave both $hr$ and $min$ unchanged. He knows that the new specification $S_H$ describes the behavior that he has already observed in the clock, since by watching only the hour display, he couldn't possibly tell whether or not there were steps that didn't change the hour display.

However, our friend is sad. The new specifications allow behaviors in which, after a while, the display stops changing, and he so enjoys watching the numbers change. We tell him not to worry because the actual specifications are

$$
\begin{array}{lcl}
S_H & \triangleq & Init_H \ \wedge \ \Box[N_H]_{hr} \ \wedge \ \mathrm{WF}_{hr}(N_H) \\
S_M & \triangleq & Init_M \ \wedge \ \Box[N_M]_{\langle hr, min \rangle} \ \wedge \ \mathrm{WF}_{\langle hr, min \rangle}(N_M)
\end{array}
\tag{3}
$$

where the WF formulas assert that the clock keeps advancing forever. Once we explain the precise meaning of WF, our friend is quite happy. He easily deduces the simple theorem

$$S_M \Rightarrow S_H \tag{4}$$

and he knows that this theorem means that if the clock satisfies the specification $S_M$, then it also satisfies the specification $S_H$, so the hour display does indeed continue

[1] He is already familiar with our notation of representing conjunctions as lists of formulas bulleted by $\wedge$.

to behave as before. We remind him that (4) is, by definition, what it means for the specification $S_M$ to implement the specification $S_H$.

Our friend enjoys his clock, and spends days doing little but watching it and verifying that it satisfies the specification $S_M$. Then, one day, a violent patient at the hospital breaks it. So, we purchase a brand new clock and send it to him. On our next visit, we find our friend distraught. The new clock does not satisfy the specification $S_M$. He explains that one time the clock changed from 7:59 to 7:00, and later it jumped directly from 7:00 to 8:00. Examining the clock, we discover that, on the hour, both displays do not change simultaneously. There can be a noticeable instant between when the minute and hour displays change. To our friend, with his distorted sense of time, that instant seems very long.

To pacify him, we must specify the new clock. We realize that, if we want to write the same kind of specification as before, we must add another variable to distinguish whether the clock reads 7:00 because it really is 7:00, or because it is in the middle of changing from 7:59 to 8:00. We use the boolean variable $chg$ that will be true iff the clock is in the middle of changing the hour, and we present our friend with the following specification $IS_L$:

$$
\begin{array}{lcl}
Init_L & \triangleq & Init_M \wedge \neg chg \\
Min & \triangleq & \wedge\ \neg chg \vee (min = 59) \\
 & & \wedge\ min' = (min + 1) \bmod 60 \\
 & & \wedge\ chg' = \neg chg \wedge (min = 59) \\
 & & \wedge\ hr' = hr \\
Hr & \triangleq & \wedge\ (\neg chg \wedge (min = 59)) \ \vee\ (chg \wedge (min = 0)) \\
 & & \wedge\ hr' = (hr \bmod 12) + 1 \\
 & & \wedge\ chg' = (min = 59) \\
 & & \wedge\ min' = min \\
N_L & \triangleq & Min \vee Hr \\
IS_L & \triangleq & Init_L \wedge \Box[N_L]_{\langle hr, min, chg \rangle} \wedge \mathrm{WF}_{\langle hr, min, chg \rangle}(N_L)
\end{array}
$$

Our friend looks puzzled and asks us where he should look for the value of $chg$. We tell him that it's just a specification variable, and doesn't necessarily correspond to anything on the actual clock. "Well, why doesn't your specification say so?" he demands. So, we define a new specification $S_L$ by

$$S_L \triangleq \exists\, chg : IS_L \tag{5}$$

Formula $S_L$ asserts that the clock acts as if there were a sequence of values for $chg$ for which $IS_L$ is satisfied; it says nothing about the actual values of $chg$. (We write $\exists$ instead of $\exists$ to indicate that we are asserting the existence of a sequence of values, not just a single value.) A clock satisfies $S_L$ iff it acts exactly the same as the clock you would get by building a device with $hr$, $min$, and $chg$ displays that satisfies $IS_L$, and then putting tape over the $chg$ display.

Reminded of the definition of $\exists$, our friend observes that $S_H$ is equivalent to $\exists\, min : S_M$. Formula $S_H$ is indeed the specification of our first gift—the old clock with tape over the minute display.

Our new specification does not make him happy. He liked the orderly progression of times on his old clock, and a sequence like 7:59, 8:59, 8:00 seems bizarre to him. He wishes he had his old clock back. Although we can't replace the old clock, we can do something almost as good: we can explain how to interpret any behavior of the new clock as a "virtual" behavior of the old clock. We offer to do this by writing a formula $R_{LH}$ that contains the variables $hr$ and $min$ as well as a new variable $\widehat{min}$. To interpret a sequence of values of $hr$ and $min$ produced by the new clock as a virtual behavior of the old clock, he just has to find a sequence of values of $\widehat{min}$ so that $R_{LH}$ is satisfied. He can then consider the sequence of $hr$ and $\widehat{min}$ values to be a behavior of the old clock.

Our friend agrees that this would make him happy if formula $R_{LH}$ satisfies two conditions. The first is that, if the sequence of $hr$ and $min$ values satisfies $S_L$, then the resulting sequence of $hr$ and $\widehat{min}$ values should satisfy $S_M$. Mathematically, this is expressed by

$$S_L \wedge R_{LH} \;\Rightarrow\; \widehat{S_M} \tag{6}$$

where $\widehat{S_M}$ is the formula obtained by substituting $\widehat{min}$ for $min$ in $S_M$. But this condition by itself would allow us to cheat—for example, by letting $R_{LH}$ be FALSE. Formula $R_{LH}$ must also satisfy the condition that, for any sequence of values of $hr$ and $min$ satisfying $S_L$, there must be some sequence of values of $\widehat{min}$ satisfying $R_{LH}$. In other words, the following must be true.

$$S_L \;\Rightarrow\; \exists\, \widehat{min} \;:\; R_{LH} \tag{7}$$

We tell him he is right, and that when (6) and (7) hold, we say that $S_L$ implies $S_M$ under the refinement $R_{LH}$. We then define $R_{LH}$ by

$$\begin{array}{rcl} R_{LH} & \triangleq & \wedge\; \widehat{min} = min \\ & & \wedge\; \Box \left[ \begin{array}{l} \widehat{min}' = \textbf{if } hr' \neq hr \\ \qquad\qquad \textbf{then } 0 \\ \qquad\qquad \textbf{else } \textbf{ if } min' = 0 \textbf{ then } \widehat{min} \\ \qquad\qquad\qquad\qquad\qquad \textbf{else } \; min' \end{array} \right] \langle hr, min, \widehat{min} \rangle \end{array}$$

Being a brilliant mathematician, he quickly verifies (6) and (7). In fact, he points out that the following result, which implies (7), is true.

$$\exists\, \widehat{min} \;:\; R_{LH} \tag{8}$$

Now that he understands what the refinement is, he realizes that there is another way to obtain it. Formula $S_L$ implies the existence of a sequence of values for $chg$ satisfying $IS_L$. We can therefore use $chg$ to define $\widehat{min}$. First, define $\overline{min}$ by

$$\overline{min} \;\triangleq\; \textbf{if } (min = 0) \wedge chg \textbf{ then } 59 \textbf{ else } min)$$

and then define the virtual variable $\widehat{min}$ to equal the expression $\overline{min}$, which is a simple function of $min$ and $chg$. This is an easier way to define the refinement because it requires writing no new temporal formulas.

Our friend is actually using the alternative refinement $R_{LH}$ defined by

$$R_{LH} \triangleq (\exists\, chg : IS_L \wedge \Box(\widehat{min} = \overline{min})) \tag{9}$$

Since $S_L$ equals $\exists\, chg : IS_L$, and the variable $\widehat{min}$ does not occur in $IS_L$, the proof of (7) with this definition of $R_{LH}$ requires no mathematical brilliance. Moreover, to prove (6), it suffices to prove the simpler theorem

$$IS_L \Rightarrow \overline{S_M} \tag{10}$$

where $\overline{S_M}$ is the formula obtained by substituting the expression $\overline{min}$ for the variable $min$ in $S_M$. (Observe that $\overline{S_M}$ contains the variables $hr$, $min$, and $chg$, while formula $\widehat{S_M}$ in (6) contains the variables $hr$, $min$, and $\widehat{min}$.)

Finally, our friend notes that since $chg$ does not occur in $S_M$, (10) implies

$$(\exists\, chg : IS_L) \Rightarrow (\exists\, min : S_M) \tag{11}$$

But $S_L$ is defined to equal $\exists\, chg : IS_L$, and he already observed that $\exists\, min : S_M$ is equivalent to $S_H$, so (11) implies that $S_L$ implements $S_M$.

## 2 In General

Let us now leave our friend to his recovery and generalize from his clocks. Although we have used TLA [4] to specify the clocks, everything generalizes to any method in which the meaning of a specification is a set of sequences of states.

We are given a higher-level specification $S_H$ and a lower-level specification $S_L$ defined by

$$\begin{aligned} S_H &\triangleq \exists\, h : IS_H \\ S_L &\triangleq \exists\, l : IS_L \end{aligned}$$

where $h$ and $l$ are sequences of variables.[2] In the clock example, $IS_H$ was called $S_M$. We say that $S_L$ implements $S_H$ iff $S_L$ implies $S_H$. For this to make sense, a specification needs to be invariant under stuttering, meaning that it is satisfied by a behavior $\sigma$ iff it is satisfied by any behavior obtained from $\sigma$ by adding or deleting steps that leave the state unchanged.

To prove that $S_L$ implies $S_H$, we first rename bound variables if necessary so that no variable of $h$ occurs free in $IS_L$. (In our example, we renamed the bound variable $min$ of $S_H$ to $\widehat{min}$.) Then $S_L$ implies $S_H$ iff the following formula is true.

$$IS_L \Rightarrow \exists\, h : IS_H \tag{12}$$

To prove (12), we must show that $IS_L$ implements $IS_H$ under some refinement $R_{LH}$, which by definition means proving

$$IS_L \wedge R_{LH} \Rightarrow IS_H \tag{13}$$

$$IS_L \Rightarrow \exists\, h : R_{LH} \tag{14}$$

[2] If $h$ is the sequence $h_1, \ldots, h_m$ of variables, then $\exists\, h : IS_H$ is an abbreviation for $\exists\, h_1, \ldots, h_n : IS_H$, which is in turn an abbreviation for $\exists\, h_1 : \ldots \exists\, h_n : IS_H$.

Since no variable in $h$ occurs free in $IS_L$, (14) is equivalent to $\exists\, h : (IS_L \Rightarrow R_{LH})$. Hence, by replacing $R_{LH}$ with $IS_L \Rightarrow R_{LH}$, we can always replace (14) by the stronger condition

$$\exists\, h \;:\; R_{LH} \tag{15}$$

Proving that (13) and (14) imply (12) is a simple exercise in predicate logic. To prove the converse, that (12) implies the existence of a refinement $R_{LH}$ satisfying (13) and (14), we simply take $R_{LH}$ to be $IS_H$.

The simplest type of refinement is a refinement mapping, in which $R_{LH}$ equals $\Box(h = \overline{h})$, where $\overline{h}$ is some state function not containing the variables $h$. In that case, (15) obviously holds, and (13) is equivalent to

$$IS_L \Rightarrow \overline{IS_H} \tag{16}$$

where $\overline{IS_H}$ is the formula obtained from $IS_H$ by replacing the variables $h$ with the state function $\overline{h}$. All the free variables of $\overline{IS_H}$ will be free variables of $IS_L$.

The validity of (12) implies the existence of a refinement $R_{LH}$, but not necessarily the existence of a refinement mapping. The existence of an arbitrary refinement $R_{LH}$ is of no help, since (14) has exactly the same form as the formula (12) that we have to prove. So, we need a method that is more general than refinement mappings but easier than using an arbitrary refinement. We generalize the alternative clock refinement defined in (9), replacing $chg$ by an arbitrary sequence $a$ of new variables and $IS_L$ by some formula $IS_L^a$ which implements $IS_H$ under a refinement mapping. It must be easy to prove

$$IS_L \Rightarrow \exists\, a \;:\; IS_L^a \tag{17}$$

and we must be able to find a refinement mapping for which we can prove

$$IS_L^a \Rightarrow \overline{IS_H} \tag{18}$$

(Even though no refinement mapping exists under which $IS_L$ implements $IS_H$, a refinement mapping satisfying (18) can exist because it can mention the variables $a$.) Predicate logic reasoning shows that, if the variables $a$ do not occur free in $IS_L$, then (17) and (18) imply (12). This approach is equivalent to defining

$$R_{LH} \;\triangleq\; (\exists\, a \;:\; IS_L^a \wedge \Box(h = \overline{h}))$$

If $\exists\, a : IS_L^a$ is equivalent to $IS_L$ instead of just implied by it, then we say that $IS_L^a$ is obtained from $IS_L$ by adding the auxiliary variables $a$.

The most common type of auxiliary variable is a history variable, in which the value of $a$ at any point in a behavior depends only on the current and previous values of the free variables of $IS_L$. Simple rules for adding auxiliary variables—that is, rules for constructing $IS_L^a$ from $IS_L$ so that $\exists\, a : IS_L^a$ is equivalent to $IS_L$—have been used for decades [5]. Another kind of auxiliary variable is a stuttering variable, which adds steps that do not change any of the variables of $IS_L$. In our clock example, we would use a stuttering variable to prove that $S_H$ implies $\exists\, min : S_M$. A more complicated type of auxiliary variable is a prophecy variable, whose value can depend on future values of the variables of $IS_L$.

A theorem of [1] asserts that, under certain conditions, fairly simple rules for adding auxiliary variables suffice to guarantee the existence of a refinement mapping for proving any valid formula of the form (12). More complicated rules allow those conditions to be weakened [2, 3]. However, there do not yet exist rules for adding auxiliary variables that are both simple and sufficiently general to handle all known examples.


### Acknowledgments

Stephan Merz found errors in an earlier version.


## References


[1] Martín Abadi and Leslie Lamport. The existence of refinement mappings. *Theoretical Computer Science*, 82(2):253–284, May 1991.

[2] Kai Engelhardt and Willem-Paul de Roever. Generalizing Abadi & Lamport's method to solve a problem posed by A. Pnueli. In J. C. P. Woodcock and P. G. Larsen, editors, *FME '93: Industrial Strength Formal Methods*, volume 670 of *Lecture Notes in Computer Science*, pages 294–313. Springer-Verlag, 1993.

[3] Bengt Jonsson. Simulations between specifications of distributed systems. In Jos C. M. Baeten and Jan Frisco Groote, editors, *CONCUR '91, 2nd International Conference on Concurrency Theory*, volume 527 of *Lecture Notes in Computer Science*, pages 346–360. Springer-Verlag, 1991.

[4] Leslie Lamport. The temporal logic of actions. *ACM Transactions on Programming Languages and Systems*, 16(3):872–923, May 1994.

[5] Susan Owicki and David Gries. Verifying properties of parallel programs: An axiomatic approach. *Communications of the ACM*, 19(5):279–284, May 1976.