sqequal_com[12,41]

Hereare the essentials:

1. For any two terms a and b, we intend 'a $\sim$ b' to be a type (in U{1})
2. For any term a, 'a $\sim$ a'
3. For two terms a and b, 'a $\sim$ b' if b computes to a (normal direct computation)
4. For any of the following types T, and two terms a, b in T, we
   have 'a = b in T => a $\sim$ b'
   a. int
   b. Atom
   c. any equality type (such as Unit)
5. Two terms 'opid1{params1}(a1, ..., an)' and 'opid2{params2}(b1, ..., bn)' are squiggle equal if opid1 and opid2 are the same, params1 and params2 are the same, the arity of the terms is the same, and 'a1 $\sim$ b1', ..., 'an $\sim$ bn'
   -- This rule is not strictly necessary, given the substitution rule, and the reflexive rule, but it is pretty useful in any case.
6. Substitution: if 'T[a]' is a hypothesis or conclusion in a sequent, and 'a $\sim$ b', then 'T[a]' can be replaced by 'T[b]' (and there is no need to prove functionality).

We don't have a rule for proving when two squiggle types are equal, so we can't use the squiggle type as a hypothesis. If we had one, the rulewould be something like this:

'a $\sim$ b = c $\sim$ d in U1'

if 'a $\sim$ b <=> c $\sim$ d' and all free variables in a, b, c, and d belong to "canonical" types (T is a canonical type if 'all a, b: T. a = b in T => a $\sim$ b').