1 Modeling

Formal specification of a buffer with an infinite number of states.

1.1 Rigid data types

Example 1. *Specification of lists of arbitrary elements:*

```
spec! LISTS
pr BOOL
sorts Elt List .
op err : -> Elt .
op empty : -> List .
op _;_ : Elt List -> List .
op _in_ : Elt List -> Bool .
eq-1 ∀L · L ; err = L
eq-2 ∀E · E in empty = false
eq-3 ∀E, E′ · (E in E′ ; L) if E = E′
eq-4 ∀E, E′ · E in E′ ; L = E in L if ¬(E = E′)
```

1.2 Nominals

Example 2. *Specification of nominals:*

```
spec! NOMINAL
sort Nominal .
op init : -> Nominal .
op next : Nominal -> Nominal .
```

1.3 Flexible data types

Example 3. *Specification of the attributes* `read` *and* `del`*:*

```
spec BUFFER[LISTS,NOMINAL]
op read : List -> Elt .
op del : List -> List .
eq-5 ∀Z · @Z read(empty) = err
eq-6 ∀Z · @Z del(empty) = empty
eq-7 ∀E, L · @init read(E ; L) = E
eq-8 ∀E, L · @init del(E ; L) = L
eq-9 ∀Z, E, L · @next(Z) read(E ; L) = @Z read(L)
eq-10 ∀Z, E, L · @next(Z) del(E ; L) = E; del(Z, L)
```

2 Formal verification