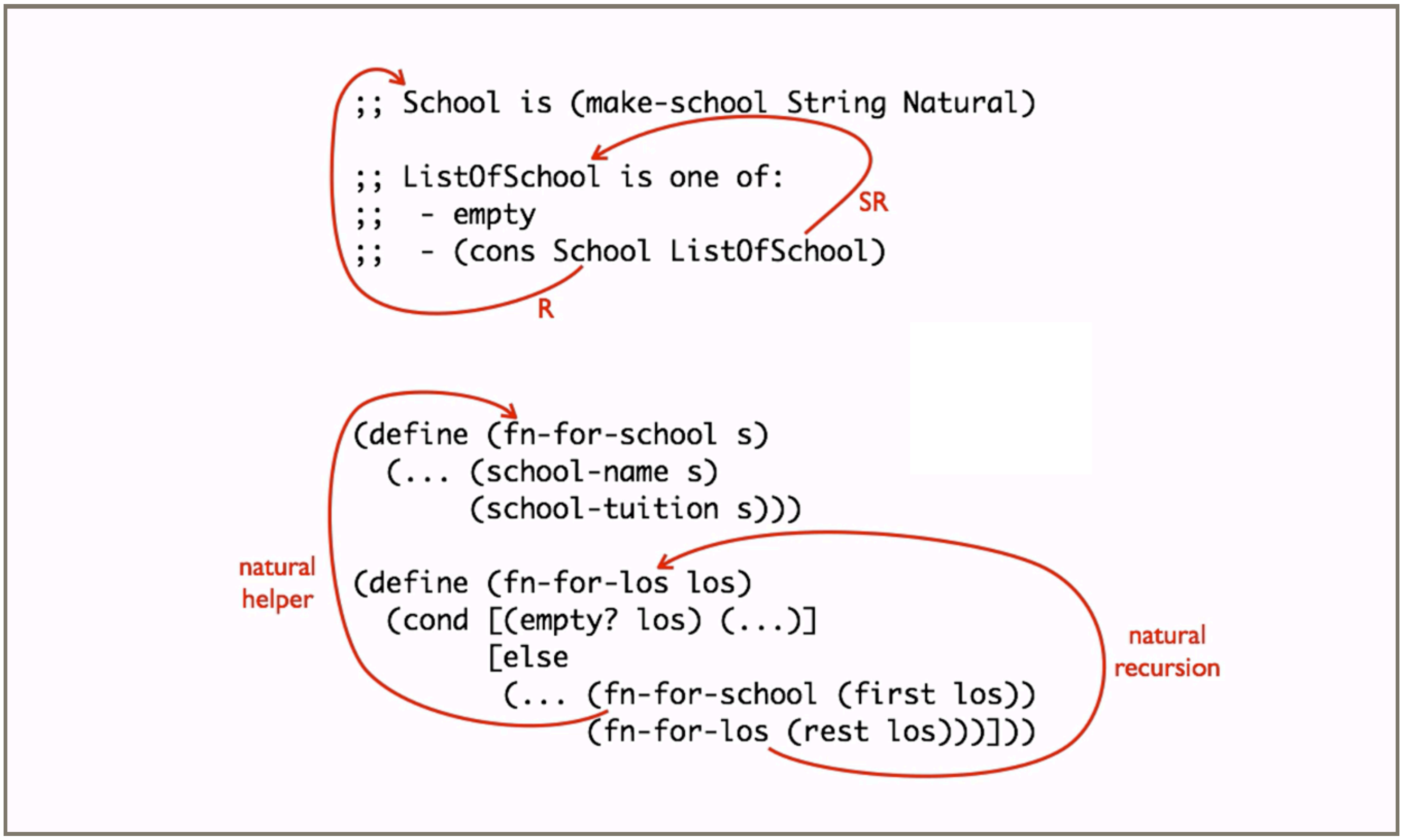
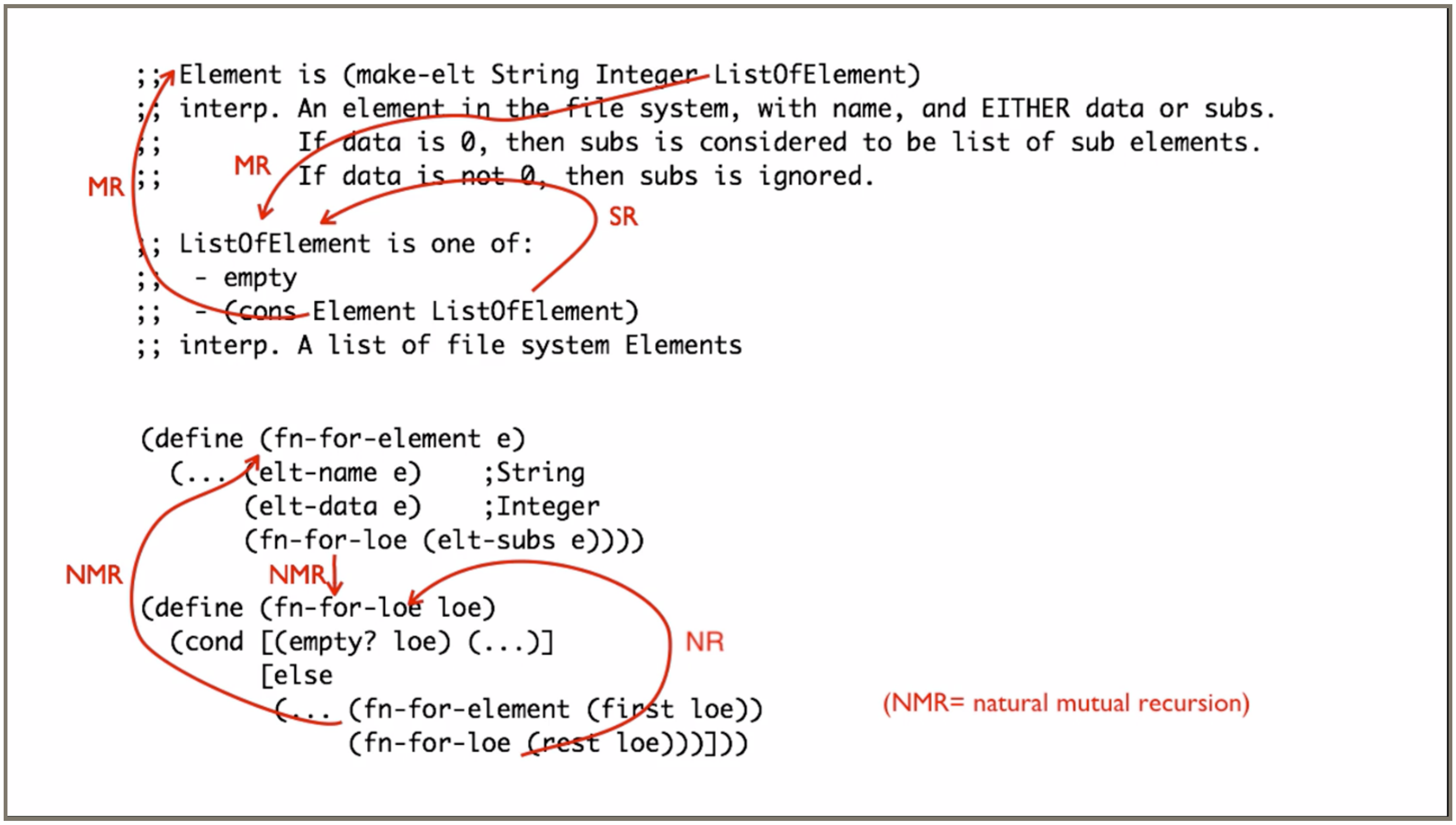


```
;; School is (make-school String Natural)

;; ListOfSchool is one of:
;;  - empty
;;  - (cons School ListOfSchool)


(define (fn-for-school s)
  (... (school-name s)
       (school-tuition s)))

(define (fn-for-los los)
  (cond [(empty? los) (...)]
        [else
         (... (fn-for-school (first los))
              (fn-for-los (rest los)))]))
```



```
;; Element is (make-elt String Integer ListOfElement)
;; interp. An element in the file system, with name, and EITHER data or subs.
;;         If data is 0, then subs is considered to be list of sub elements.
;;         If data is not 0, then subs is ignored.

;; ListOfElement is one of:
;;  - empty
;;  - (cons Element ListOfElement)
;; interp. A list of file system Elements


(define (fn-for-element e)
  (... (elt-name e)    ;String
       (elt-data e)    ;Integer
       (fn-for-loe (elt-subs e))))

(define (fn-for-loe loe)
  (cond [(empty? loe) (...)]
        [else
         (... (fn-for-element (first loe))
              (fn-for-loe (rest loe)))]))
```