```
ma-init-sub  THM

⊢  ∀V:(Id→Type), M1,M2:MsgA.
|  (∀x:Id. V(x) ⊆r M2.ds(x))
|  ⇒
|  M1 ⊆ M2 ⇒ (∀x:Id, v:V(x). M2.init(x,v) ⇒ M1.init(x,v))
| by Repeat (Analyze 0 THENA Complete Auto) THEN Unfold_MsgA -6
|    THEN
|    Unfold_MsgA -5
|    THEN
|    SplitAndHyps
|    THEN
|    Unfold ‘fpf-val‘ 0
|    THEN
|    Try (DoSubsume THEN BackThruSomeHyp)
|    THEN
|    AllHyps
|    (λh.Unfold ‘fpf-sub‘ h THEN InstHyp [x] h THENA Complete Auto)
|    THEN
|    ExRepD
|    THEN
|    ThinTrivial
|
|\
| 1. V : Id→Type
| 2. ds : x:Id fp-> Type
| 3. da : a:Knd fp-> Type
| 4. init : x:Id fp-> ds(x)?Void
| 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
| 6. ef :
| |  kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
| |  ds(2of(kx))?Void
| 7. send :
| |  kl:Knd×IdLnk fp-> (tg:Id
| |                     ×State(ds)→ma-valtype(da; 1of(kl))→
| |                     (da(rcv(2of(kl); tg))?Void List)) List
| 8. frame : x:Id fp-> Knd List
| 9. sframe : ltg:IdLnk×Id fp-> Knd List
| 10. Top
| 11. d1 : x:Id fp-> Type
| 12. d2 : a:Knd fp-> Type
| 13. i1 : x:Id fp-> d1(x)?Void
| 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
| 15. e1 :
| |   kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
| |   d1(2of(kx))?Void
| 16. s1 :
| |   kl:Knd×IdLnk fp-> (tg:Id
| |                      ×State(d1)→ma-valtype(d2; 1of(kl))→
| |                      (d2(rcv(2of(kl); tg))?Void List)) List
| 17. f1 : x:Id fp-> Knd List
| 18. s2 : ltg:IdLnk×Id fp-> Knd List
| 19. Top
| 20. ∀x:Id. V(x) ⊆r d1(x)?Top
| 21. ds ⊆ d1
```

```
| 22. da ⊆ d2
| 23. ∀x:Id. x ∈ dom(init) ⇒ x ∈ dom(i1) & init(x) = i1(x)
| 24. pre ⊆ p1
| 25. ef ⊆ e1
| 26. send ⊆ s1
| 27. frame ⊆ f1
| 28. sframe ⊆ s2
| 29. x : Id
| 30. v : V(x)
| 31. x ∈ dom(init)
| 32. x ∈ dom(i1)
| 33. init(x) = i1(x)
| 34. v = i1(x)
| ⊢  v = init(x)  by Assert (ds(x)?Void = d1(x)?Void)
| |
| .....assertion .....
| 1. V : Id→Type
| 2. ds : x:Id fp-> Type
| 3. da : a:Knd fp-> Type
| 4. init : x:Id fp-> ds(x)?Void
| 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
| 6. ef :
| |  kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
| |  ds(2of(kx))?Void
| 7. send :
| |  kl:Knd×IdLnk fp-> (tg:Id
| |                     ×State(ds)→ma-valtype(da; 1of(kl))→
| |                     (da(rcv(2of(kl); tg))?Void List)) List
| 8. frame : x:Id fp-> Knd List
| 9. sframe : ltg:IdLnk×Id fp-> Knd List
| 10. Top
| 11. d1 : x:Id fp-> Type
| 12. d2 : a:Knd fp-> Type
| 13. i1 : x:Id fp-> d1(x)?Void
| 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
| 15. e1 :
| |   kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
| |   d1(2of(kx))?Void
| 16. s1 :
| |   kl:Knd×IdLnk fp-> (tg:Id
| |                      ×State(d1)→ma-valtype(d2; 1of(kl))→
| |                      (d2(rcv(2of(kl); tg))?Void List)) List
| 17. f1 : x:Id fp-> Knd List
| 18. s2 : ltg:IdLnk×Id fp-> Knd List
| 19. Top
| 20. ∀x:Id. V(x) ⊆r d1(x)?Top
| 21. ds ⊆ d1
| 22. da ⊆ d2
| 23. ∀x:Id. x ∈ dom(init) ⇒ x ∈ dom(i1) & init(x) = i1(x)
| 24. pre ⊆ p1
| 25. ef ⊆ e1
| 26. send ⊆ s1
| 27. frame ⊆ f1
| 28. sframe ⊆ s2
| 29. x : Id
```

```
| 30. v : V(x)
| 31. x ∈ dom(init)
| 32. x ∈ dom(i1)
| 33. init(x) = i1(x)
| 34. v = i1(x)
| ⊢  ds(x)?Void = d1(x)?Void
| | by Unfold ‘fpf-cap‘ 0 THEN SplitOnConclITE
| |
| |\
| | .....truecase .....
| | 1. V : Id→Type
| | 2. ds : x:Id fp-> Type
| | 3. da : a:Knd fp-> Type
| | 4. init : x:Id fp-> ds(x)?Void
| | 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
| | 6. ef :
| | |  kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
| | |  ds(2of(kx))?Void
| | 7. send :
| | |  kl:Knd×IdLnk fp-> (tg:Id
| | |                     ×State(ds)→ma-valtype(da; 1of(kl))→
| | |                     (da(rcv(2of(kl); tg))?Void List)) List
| | 8. frame : x:Id fp-> Knd List
| | 9. sframe : ltg:IdLnk×Id fp-> Knd List
| | 10. Top
| | 11. d1 : x:Id fp-> Type
| | 12. d2 : a:Knd fp-> Type
| | 13. i1 : x:Id fp-> d1(x)?Void
| | 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
| | 15. e1 :
| | |   kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
| | |   d1(2of(kx))?Void
| | 16. s1 :
| | |   kl:Knd×IdLnk fp-> (tg:Id
| | |                      ×State(d1)→ma-valtype(d2; 1of(kl))→
| | |                      (d2(rcv(2of(kl); tg))?Void List)) List
| | 17. f1 : x:Id fp-> Knd List
| | 18. s2 : ltg:IdLnk×Id fp-> Knd List
| | 19. Top
| | 20. ∀x:Id. V(x) ⊆r d1(x)?Top
| | 21. ds ⊆ d1
| | 22. da ⊆ d2
| | 23. ∀x:Id. x ∈ dom(init) ⇒ x ∈ dom(i1) & init(x) = i1(x)
| | 24. pre ⊆ p1
| | 25. ef ⊆ e1
| | 26. send ⊆ s1
| | 27. frame ⊆ f1
| | 28. sframe ⊆ s2
| | 29. x : Id
| | 30. v : V(x)
| | 31. x ∈ dom(init)
| | 32. x ∈ dom(i1)
| | 33. init(x) = i1(x)
| | 34. v = i1(x)
| | 35. x ∈ dom(ds)
```

```
| | ⊢  ds(x) = if x ∈ dom(d1)→ d1(x) else Void fi  by SplitOnConclITE
| | |
| | |\
| | | .....truecase .....
| | | 1. V : Id→Type
| | | 2. ds : x:Id fp-> Type
| | | 3. da : a:Knd fp-> Type
| | | 4. init : x:Id fp-> ds(x)?Void
| | | 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
| | | 6. ef :
| | | |  kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
| | | |  ds(2of(kx))?Void
| | | 7. send :
| | | |  kl:Knd×IdLnk fp-> (tg:Id
| | | |                     ×State(ds)→ma-valtype(da; 1of(kl))→
| | | |                     (da(rcv(2of(kl); tg))?Void List)) List
| | | 8. frame : x:Id fp-> Knd List
| | | 9. sframe : ltg:IdLnk×Id fp-> Knd List
| | | 10. Top
| | | 11. d1 : x:Id fp-> Type
| | | 12. d2 : a:Knd fp-> Type
| | | 13. i1 : x:Id fp-> d1(x)?Void
| | | 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
| | | 15. e1 :
| | | |   kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
| | | |   d1(2of(kx))?Void
| | | 16. s1 :
| | | |   kl:Knd×IdLnk fp-> (tg:Id
| | | |                      ×State(d1)→ma-valtype(d2; 1of(kl))→
| | | |                      (d2(rcv(2of(kl); tg))?Void List)) List
| | | 17. f1 : x:Id fp-> Knd List
| | | 18. s2 : ltg:IdLnk×Id fp-> Knd List
| | | 19. Top
| | | 20. ∀x:Id. V(x) ⊆r d1(x)?Top
| | | 21. ds ⊆ d1
| | | 22. da ⊆ d2
| | | 23. ∀x:Id. x ∈ dom(init) ⇒ x ∈ dom(i1) & init(x) = i1(x)
| | | 24. pre ⊆ p1
| | | 25. ef ⊆ e1
| | | 26. send ⊆ s1
| | | 27. frame ⊆ f1
| | | 28. sframe ⊆ s2
| | | 29. x : Id
| | | 30. v : V(x)
| | | 31. x ∈ dom(init)
| | | 32. x ∈ dom(i1)
| | | 33. init(x) = i1(x)
| | | 34. v = i1(x)
| | | 35. x ∈ dom(ds)
| | | 36. x ∈ dom(d1)
| | | ⊢  ds(x) = d1(x)
| | | | by AllHyps
| | | |    (λh.
| | | |     Unfold ‘fpf-sub‘ h THEN InstHyp [x] h
| | | |     THENA
```

```
| | | |     Complete Auto)
| | | |
| | | 1. V : Id→Type
| | | 2. ds : x:Id fp-> Type
| | | 3. da : a:Knd fp-> Type
| | | 4. init : x:Id fp-> ds(x)?Void
| | | 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
| | | 6. ef :
| | |    kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
| | |    ds(2of(kx))?Void
| | | 7. send :
| | |    kl:Knd×IdLnk fp-> (tg:Id
| | |                       ×State(ds)→ma-valtype(da; 1of(kl))→
| | |                       (da(rcv(2of(kl); tg))?Void List)) List
| | | 8. frame : x:Id fp-> Knd List
| | | 9. sframe : ltg:IdLnk×Id fp-> Knd List
| | | 10. Top
| | | 11. d1 : x:Id fp-> Type
| | | 12. d2 : a:Knd fp-> Type
| | | 13. i1 : x:Id fp-> d1(x)?Void
| | | 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
| | | 15. e1 :
| | |     kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
| | |     d1(2of(kx))?Void
| | | 16. s1 :
| | |     kl:Knd×IdLnk fp-> (tg:Id
| | |                        ×State(d1)→ma-valtype(d2; 1of(kl))→
| | |                        (d2(rcv(2of(kl); tg))?Void List)) List
| | | 17. f1 : x:Id fp-> Knd List
| | | 18. s2 : ltg:IdLnk×Id fp-> Knd List
| | | 19. Top
| | | 20. ∀x:Id. V(x) ⊆r d1(x)?Top
| | | 21. ∀x:Id. x ∈ dom(ds) ⇒ x ∈ dom(d1) & ds(x) = d1(x)
| | | 22. da ⊆ d2
| | | 23. ∀x:Id. x ∈ dom(init) ⇒ x ∈ dom(i1) & init(x) = i1(x)
| | | 24. pre ⊆ p1
| | | 25. ef ⊆ e1
| | | 26. send ⊆ s1
| | | 27. frame ⊆ f1
| | | 28. sframe ⊆ s2
| | | 29. x : Id
| | | 30. v : V(x)
| | | 31. x ∈ dom(init)
| | | 32. x ∈ dom(i1)
| | | 33. init(x) = i1(x)
| | | 34. v = i1(x)
| | | 35. x ∈ dom(ds)
| | | 36. x ∈ dom(d1)
| | | 37. x ∈ dom(d1) & ds(x) = d1(x)
| | | ⊢  ds(x) = d1(x)  by ExRepD
| | |
| | .\
| | . .....falsecase .....
| | . 1. V : Id→Type
| | . 2. ds : x:Id fp-> Type
```

```
| | . 3. da : a:Knd fp-> Type
| | . 4. init : x:Id fp-> ds(x)?Void
| | . 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
| | . 6. ef :
| | .    kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
| | .    ds(2of(kx))?Void
| | . 7. send :
| | .    kl:Knd×IdLnk fp-> (tg:Id
| | .                       ×State(ds)→ma-valtype(da; 1of(kl))→
| | .                       (da(rcv(2of(kl); tg))?Void List)) List
| | . 8. frame : x:Id fp-> Knd List
| | . 9. sframe : ltg:IdLnk×Id fp-> Knd List
| | . 10. Top
| | . 11. d1 : x:Id fp-> Type
| | . 12. d2 : a:Knd fp-> Type
| | . 13. i1 : x:Id fp-> d1(x)?Void
| | . 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
| | . 15. e1 :
| | .     kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
| | .     d1(2of(kx))?Void
| | . 16. s1 :
| | .     kl:Knd×IdLnk fp-> (tg:Id
| | .                        ×State(d1)→ma-valtype(d2; 1of(kl))→
| | .                        (d2(rcv(2of(kl); tg))?Void List)) List
| | . 17. f1 : x:Id fp-> Knd List
| | . 18. s2 : ltg:IdLnk×Id fp-> Knd List
| | . 19. Top
| | . 20. ∀x:Id. V(x) ⊆r d1(x)?Top
| | . 21. ds ⊆ d1
| | . 22. da ⊆ d2
| | . 23. ∀x:Id. x ∈ dom(init) ⇒ x ∈ dom(i1) & init(x) = i1(x)
| | . 24. pre ⊆ p1
| | . 25. ef ⊆ e1
| | . 26. send ⊆ s1
| | . 27. frame ⊆ f1
| | . 28. sframe ⊆ s2
| | . 29. x : Id
| | . 30. v : V(x)
| | . 31. x ∈ dom(init)
| | . 32. x ∈ dom(i1)
| | . 33. init(x) = i1(x)
| | . 34. v = i1(x)
| | . 35. x ∈ dom(ds)
| | . 36. ¬x ∈ dom(d1)
| | . ⊢  ds(x) = Void
| | .   by Assert Void THEN UseWitness i1(x) THEN DoSubsume
| | .      THEN
| | .      Unfold ‘fpf-cap‘ 0
| | .      THEN
| | .      SplitOnConclITE
| |
| .\
| . .....falsecase .....
| . 1. V : Id→Type
| . 2. ds : x:Id fp-> Type
```

```
| . 3. da : a:Knd fp-> Type
| . 4. init : x:Id fp-> ds(x)?Void
| . 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
| . 6. ef :
| .    kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
| .    ds(2of(kx))?Void
| . 7. send :
| .    kl:Knd×IdLnk fp-> (tg:Id
| .                       ×State(ds)→ma-valtype(da; 1of(kl))→
| .                       (da(rcv(2of(kl); tg))?Void List)) List
| . 8. frame : x:Id fp-> Knd List
| . 9. sframe : ltg:IdLnk×Id fp-> Knd List
| . 10. Top
| . 11. d1 : x:Id fp-> Type
| . 12. d2 : a:Knd fp-> Type
| . 13. i1 : x:Id fp-> d1(x)?Void
| . 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
| . 15. e1 :
| .     kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
| .     d1(2of(kx))?Void
| . 16. s1 :
| .     kl:Knd×IdLnk fp-> (tg:Id
| .                        ×State(d1)→ma-valtype(d2; 1of(kl))→
| .                        (d2(rcv(2of(kl); tg))?Void List)) List
| . 17. f1 : x:Id fp-> Knd List
| . 18. s2 : ltg:IdLnk×Id fp-> Knd List
| . 19. Top
| . 20. ∀x:Id. V(x) ⊆r d1(x)?Top
| . 21. ds ⊆ d1
| . 22. da ⊆ d2
| . 23. ∀x:Id. x ∈ dom(init) ⇒ x ∈ dom(i1) & init(x) = i1(x)
| . 24. pre ⊆ p1
| . 25. ef ⊆ e1
| . 26. send ⊆ s1
| . 27. frame ⊆ f1
| . 28. sframe ⊆ s2
| . 29. x : Id
| . 30. v : V(x)
| . 31. x ∈ dom(init)
| . 32. x ∈ dom(i1)
| . 33. init(x) = i1(x)
| . 34. v = i1(x)
| . 35. ¬x ∈ dom(ds)
| . ⊢  Void = if x ∈ dom(d1)→ d1(x) else Void fi
| .   by Assert Void THEN UseWitness init(x) THEN DoSubsume
| .      THEN
| .      Unfold ‘fpf-cap‘ 0
| .      THEN
| .      SplitOnConclITE
|
.\
. 1. V : Id→Type
. 2. ds : x:Id fp-> Type
. 3. da : a:Knd fp-> Type
. 4. init : x:Id fp-> ds(x)?Void
```

```
. 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
. 6. ef :
. |  kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
. |  ds(2of(kx))?Void
. 7. send :
. |  kl:Knd×IdLnk fp-> (tg:Id
. |                      ×State(ds)→ma-valtype(da; 1of(kl))→
. |                      (da(rcv(2of(kl); tg))?Void List)) List
. 8. frame : x:Id fp-> Knd List
. 9. sframe : ltg:IdLnk×Id fp-> Knd List
. 10. Top
. 11. d1 : x:Id fp-> Type
. 12. d2 : a:Knd fp-> Type
. 13. i1 : x:Id fp-> d1(x)?Void
. 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
. 15. e1 :
. |   kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
. |   d1(2of(kx))?Void
. 16. s1 :
. |   kl:Knd×IdLnk fp-> (tg:Id
. |                       ×State(d1)→ma-valtype(d2; 1of(kl))→
. |                       (d2(rcv(2of(kl); tg))?Void List)) List
. 17. f1 : x:Id fp-> Knd List
. 18. s2 : ltg:IdLnk×Id fp-> Knd List
. 19. Top
. 20. ∀x:Id. V(x) ⊆r d1(x)?Top
. 21. ds ⊆ d1
. 22. da ⊆ d2
. 23. init ⊆ i1
. 24. pre ⊆ p1
. 25. ef ⊆ e1
. 26. send ⊆ s1
. 27. frame ⊆ f1
. 28. sframe ⊆ s2
. 29. x : Id
. 30. v : V(x)
. 31. x ∈ dom(i1)
. ⊢  v ∈ d1(x)?Void  by SubsumeC d1(x)?Top
. |
. |\
. | 1. V : Id→Type
. | 2. ds : x:Id fp-> Type
. | 3. da : a:Knd fp-> Type
. | 4. init : x:Id fp-> ds(x)?Void
. | 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
. | 6. ef :
. |    kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
. |    ds(2of(kx))?Void
. | 7. send :
. |    kl:Knd×IdLnk fp-> (tg:Id
. |                         ×State(ds)→ma-valtype(da; 1of(kl))→
. |                         (da(rcv(2of(kl); tg))?Void List)) List
. | 8. frame : x:Id fp-> Knd List
. | 9. sframe : ltg:IdLnk×Id fp-> Knd List
. | 10. Top
```

```
. | 11. d1 : x:Id fp-> Type
. | 12. d2 : a:Knd fp-> Type
. | 13. i1 : x:Id fp-> d1(x)?Void
. | 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
. | 15. e1 :
. |     kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
. |     d1(2of(kx))?Void
. | 16. s1 :
. |     kl:Knd×IdLnk fp-> (tg:Id
. |                         ×State(d1)→ma-valtype(d2; 1of(kl))→
. |                         (d2(rcv(2of(kl); tg))?Void List)) List
. | 17. f1 : x:Id fp-> Knd List
. | 18. s2 : ltg:IdLnk×Id fp-> Knd List
. | 19. Top
. | 20. ∀x:Id. V(x) ⊆r d1(x)?Top
. | 21. ds ⊆ d1
. | 22. da ⊆ d2
. | 23. init ⊆ i1
. | 24. pre ⊆ p1
. | 25. ef ⊆ e1
. | 26. send ⊆ s1
. | 27. frame ⊆ f1
. | 28. sframe ⊆ s2
. | 29. x : Id
. | 30. v : V(x)
. | 31. x ∈ dom(i1)
. | ⊢  v ∈ d1(x)?Top  by DoSubsume THEN BackThruSomeHyp
. |
. .\
. . .....subtype_rel .....
. . 1. V : Id→Type
. . 2. ds : x:Id fp-> Type
. . 3. da : a:Knd fp-> Type
. . 4. init : x:Id fp-> ds(x)?Void
. . 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
. . 6. ef :
. . |  kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
. . |  ds(2of(kx))?Void
. . 7. send :
. . |  kl:Knd×IdLnk fp-> (tg:Id
. . |                     ×State(ds)→ma-valtype(da; 1of(kl))→
. . |                     (da(rcv(2of(kl); tg))?Void List)) List
. . 8. frame : x:Id fp-> Knd List
. . 9. sframe : ltg:IdLnk×Id fp-> Knd List
. . 10. Top
. . 11. d1 : x:Id fp-> Type
. . 12. d2 : a:Knd fp-> Type
. . 13. i1 : x:Id fp-> d1(x)?Void
. . 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
. . 15. e1 :
. . |   kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
. . |   d1(2of(kx))?Void
. . 16. s1 :
. . |   kl:Knd×IdLnk fp-> (tg:Id
. . |                     ×State(d1)→ma-valtype(d2; 1of(kl))→
```

```
. . |                        (d2(rcv(2of(kl); tg))?Void List)) List
. . 17. f1 : x:Id fp-> Knd List
. . 18. s2 : ltg:IdLnk×Id fp-> Knd List
. . 19. Top
. . 20. ∀x:Id. V(x) ⊆r d1(x)?Top
. . 21. ds ⊆ d1
. . 22. da ⊆ d2
. . 23. init ⊆ i1
. . 24. pre ⊆ p1
. . 25. ef ⊆ e1
. . 26. send ⊆ s1
. . 27. frame ⊆ f1
. . 28. sframe ⊆ s2
. . 29. x : Id
. . 30. V(x)
. . 31. x ∈ dom(i1)
. . ⊢  d1(x)?Top ⊆r d1(x)?Void
. . | by Unfold `fpf-cap` 0 THEN SplitOnConclITE
. . |
. . .....falsecase .....
. . 1. V : Id→Type
. . 2. ds : x:Id fp-> Type
. . 3. da : a:Knd fp-> Type
. . 4. init : x:Id fp-> ds(x)?Void
. . 5. pre : a:Id fp-> State(ds)→ma-valtype(da; locl(a))→Prop
. . 6. ef :
. .    kx:Knd×Id fp-> State(ds)→ma-valtype(da; 1of(kx))→
. .    ds(2of(kx))?Void
. . 7. send :
. .    kl:Knd×IdLnk fp-> (tg:Id
. .                       ×State(ds)→ma-valtype(da; 1of(kl))→
. .                       (da(rcv(2of(kl); tg))?Void List)) List
. . 8. frame : x:Id fp-> Knd List
. . 9. sframe : ltg:IdLnk×Id fp-> Knd List
. . 10. Top
. . 11. d1 : x:Id fp-> Type
. . 12. d2 : a:Knd fp-> Type
. . 13. i1 : x:Id fp-> d1(x)?Void
. . 14. p1 : a:Id fp-> State(d1)→ma-valtype(d2; locl(a))→Prop
. . 15. e1 :
. .     kx:Knd×Id fp-> State(d1)→ma-valtype(d2; 1of(kx))→
. .     d1(2of(kx))?Void
. . 16. s1 :
. .     kl:Knd×IdLnk fp-> (tg:Id
. .                        ×State(d1)→ma-valtype(d2; 1of(kl))→
. .                        (d2(rcv(2of(kl); tg))?Void List)) List
. . 17. f1 : x:Id fp-> Knd List
. . 18. s2 : ltg:IdLnk×Id fp-> Knd List
. . 19. Top
. . 20. ∀x:Id. V(x) ⊆r d1(x)?Top
. . 21. ds ⊆ d1
. . 22. da ⊆ d2
. . 23. init ⊆ i1
. . 24. pre ⊆ p1
. . 25. ef ⊆ e1
```

```
. . 26. send ⊆ s1
. . 27. frame ⊆ f1
. . 28. sframe ⊆ s2
. . 29. x : Id
. . 30. V(x)
. . 31. x ∈ dom(i1)
. . 32. ¬x ∈ dom(d1)
. . ⊢  Top ⊆r Void
. .   by Assert Void THEN UseWitness i1(x) THEN DoSubsume
. .      THEN
. .      Unfold ‘fpf-cap‘ 0
. .      THEN
. .      SplitOnConclITE
END
```