1 VCG Correctness

theory *VCG-Correctness* = *VCG*:

locale *correctVCG* = *VCG* +

fixes *ReachablesAn*::*'prog* ⇒ (*'pos* × *'mem*) *set*
defines
ReachablesAn ≡ λ Π. *ReachableFromInv* (*effS* Π) (*initS* Π) ({*s*. Π,*s* ⊨ *aF* Π (*fst s*)})

assumes *correctInitF*:
⟦ *wf* Π; *s* ∈ *initS* Π ⟧ ⟹ *valid* Π *s* (*initF* Π)

assumes *correctWpF*:
⟦ *wf* Π; *s* ∈ (*ReachablesAn* Π); (*s*,*s'*)∈(*effS* Π); Π,*s* ⊨ (*wpF* Π (*fst s*) (*fst s'*) *Q*)
⟧ ⟹ Π,*s'* ⊨ *Q*

assumes *succsF-complete*:
⟦ *wf* Π; *s* ∈ (*ReachablesAn* Π); (*s*,*s'*) ∈ (*effS* Π) ⟧
⟹ (∃ *B*. (*fst s'*,*B*) ∈ *set* (*succsF* Π (*fst s*)) ∧ *valid* Π *s B*)

assumes *correctSafetyLogic*:
⟦ *wf* Π; Π ⊢ *f*; *s* ∈ (*ReachablesAn* Π) ⟧ ⟹ Π,*s* ⊨ *f*

lemma (**in** *correctVCG*) *isafeF-saF*:
Π,*s* ⊨ *isafeF* Π (*fst s*) ⟹ Π, *s* ⊨ *saF* Π (*fst s*)
lemma (**in** *correctVCG*) *vc-isafeF-ReachablesAn*:
assumes *wf-Pi*: *wf* Π
assumes *vc-provable*: Π ⊢ *vcg* Π
assumes *s-Reach*: *s* ∈ (*Reachables* Π)
shows (Π, *s* ⊨ *isafeF* Π (*fst s*) ∧ *s* ∈ *ReachablesAn* Π)
using *s-Reach*
proof (*induct rule*: *Reachables-induct*)

— induction base
case (*init s*)
show Π,*s* ⊨ *isafeF* Π (*fst s*) ∧ *s* ∈ *ReachablesAn* Π
proof (*rule conjI*)
from *wf-Pi vc-provable init*
show Π,*s* ⊨ *isafeF* Π (*fst s*)
by (*rule vc-init*)
next
show *s* ∈ *ReachablesAn* Π

```
      by (rule ReachablesAn-init)
  qed

next

case (step s s')

have s-Reach: s ∈ Reachables Π .

have s-isafeF: Π,s ⊨ isafeF Π (fst s) and

     s-ReachAn: s ∈ ReachablesAn Π
  using step by auto

have s-s'-effS: (s,s') ∈ effS Π .

from s-isafeF
have s-saF: Π,s ⊨ saF Π (fst s)
  by (rule isafeF-saF)

from s-saF
 have s-aF: Π,s ⊨ aF Π (fst s)
   by (simp add: saF-def semConj)

from wf-Pi s-ReachAn s-s'-effS
obtain B where B-succsF: (fst s',B) ∈ set (succsF Π (fst s)) and s-B: Π,s ⊨ B
 apply −
 apply (drule-tac s=s and s'=s' in succsF-complete)
 apply assumption
 apply assumption
 apply fastsimp
 done

from B-succsF
obtain su su'
where su-B-su'-succsF: succsF Π (fst s) = su@(fst s',B)#su'
 by (fastsimp simp add: in-set-conv-decomp)

from wf-Pi s-isafeF
have s-domC: fst s ∈ (set (domC Π))
  apply −
  apply (drule-tac  pc=fst s in isafeFdef)
  apply (rule classical)
  apply (simp add: isafeFdef semFalse)
  done
```

from *s-domC*
obtain *dC dC′* **where** *domC-dC-dC′*: *domC* Π = *dC*@(*fst s*)#*dC′*
by (*fastsimp simp add*: *in-set-conv-decomp*)

have *s′-isafeF*: Π,*s′* ⊨ *isafeF* Π (*fst s′*)
proof (*cases anF* Π (*fst s*))
case *None*

from *None wf-Pi s-domC s-isafeF su-B-su′-succsF*
have *B-imp-wpF*: Π,*s* ⊨ *B* ⌞⇒⌟ *wpF* Π (*fst s*) (*fst s′*) (*isafeF* Π (*fst s′*))
apply −
apply (*drule-tac pc*=*fst s* **in** *isafeFdef*)
apply (*simp add*: *semConj*)
done

from *s-B B-imp-wpF*
have *wpF-isafeF*: Π,*s* ⊨ *wpF* Π (*fst s*) (*fst s′*) (*isafeF* Π (*fst s′*))
apply −
apply (*drule semImpE*)
apply *simp*
done

from *wf-Pi s-ReachAn B-succsF s-s′-effS wpF-isafeF*
show Π,*s′* ⊨ *isafeF* Π (*fst s′*)
by (*rule-tac Q*=*isafeF* Π (*fst s′*) **and** *s*=*s* **in** *correctWpF*)

next

case (*Some A*)

from *Some domC-dC-dC′*
obtain *dA dA′* **where** *domA-dA-dA′*: *domA* Π = *dA*@(*fst s*)#*dA′*
by (*simp add*: *domA-def in-set-conv-decomp*)

from *wf-Pi vc-provable s-ReachAn*
have *s-vc*: Π,*s* ⊨ (*vcg* Π)
by (*rule correctSafetyLogic*)

from *wf-Pi Some s-isafeF s-domC*
have *isafeF-safeF-A*: *isafeF* Π (*fst s*) = ⌞⋀⌟ [*safeF* Π (*fst s*), *A*]
by (*drule-tac pc*=*fst s* **in** *isafeFdef*, *simp*)

from *wf-Pi domA-dA-dA′ s-vc su-B-su′-succsF*
have *isafeF-B-imp-wpF*: Π,*s* ⊨ ⌞⋀⌟ [*isafeF* Π (*fst s*),*B*] ⌞⇒⌟ *wpF* Π (*fst s*) (*fst s′*) (*isafeF* Π (*fst s′*))

```
    apply −
    by (drule vcgdef, simp add: semConj)

  from isafeF-B-imp-wpF s-isafeF s-B
  have s-wpF-isafeF: Π,s ⊨ wpF Π (fst s) (fst s′) (isafeF Π (fst s′))
    apply −
    apply (drule semImpE)
    apply (simp add: semConj)
    done

  from wf-Pi s-ReachAn B-succsF s-s′-effS s-wpF-isafeF
  show Π,s′ ⊨ isafeF Π (fst s′)
    by (rule-tac Q=isafeF Π (fst s′) and s=s in correctWpF)


qed

from s′-isafeF have s′-aF: Π,s′ ⊨ aF Π (fst s′)
  apply −
  apply (drule isafeF-saF)
  apply (simp add: saF-def semConj)
  done

show Π,s′ ⊨ isafeF Π (fst s′) ∧ s′ ∈ ReachablesAn Π
  proof (rule conjI)
    show Π,s′ ⊨ isafeF Π (fst s′)
      by (rule s′-isafeF)

   next
     from s-ReachAn s-aF s′-aF s-s′-effS
     show s′ ∈ ReachablesAn Π
       by (rule ReachablesAn-step)
  qed
qedThe Soundness proof of our VCGtheorem (in correctVCG) vcg-soundness:
assumes wf-Pi: wf Π
assumes vc-provable: Π ⊢ vcg Π
shows isSafe Π
proof −

have isSafePi: ∀ s ∈ Reachables Π. Π,s ⊨ safeF Π (fst s)
 proof (rule ballI)
 fix s
 assume s-Reach: s ∈ Reachables Π
 show Π,s ⊨ safeF Π (fst s)
   proof −
```

from *wf-Pi vc-provable s-Reach*
have *s-isafeF-ReachAn*: Π,$s \models$ *isafeF* Π (*fst s*) $\wedge$ $s \in$ *ReachablesAn* Π
by (*rule vc-isafeF-ReachablesAn*)

from *s-isafeF-ReachAn*
have *s-isafeF*: Π,$s \models$ *isafeF* Π (*fst s*)
by *simp*

from *s-isafeF*
show Π,$s \models$ *safeF* Π (*fst s*)
apply −
apply (*drule isafeF-saF*)
apply (*simp add*: *saF-def semConj*)
done
qed
qed
from *isSafePi* **show** *isSafe* Π
by (*simp add*: *isSafe-def*)
qedVerification Conditions also guarantee correct annotations.**lemma** (**in** *correctVCG*) *vcg-correctAn*:
assumes *wf-Pi*: *wf* Π
assumes *vc-provable*: $\Pi \vdash$ *vcg* Π
shows *correctAn* Π
proof −

have *correctAnPi*: $\forall$ $s \in$ *Reachables* Π. Π,$s \models$ *aF* Π (*fst s*)
proof (*rule ballI*)
fix *s*
assume *s-Reach*: $s \in$ *Reachables* Π
show Π,$s \models$ *aF* Π (*fst s*)
proof −
from *wf-Pi vc-provable s-Reach*
have *s-isafeF-ReachAn*: Π,$s \models$ *isafeF* Π (*fst s*) $\wedge$ $s \in$ *ReachablesAn* Π
by (*rule vc-isafeF-ReachablesAn*)

from *s-isafeF-ReachAn*
have *s-isafeF*: Π,$s \models$ *isafeF* Π (*fst s*)
by *simp*

from *s-isafeF*
show Π,$s \models$ *aF* Π (*fst s*)
apply −
apply (*drule isafeF-saF*)
apply (*simp add*: *saF-def semConj*)
done
qed

qed
from *correctAnPi* **show** *correctAn* Π
by (*simp add*: *correctAn-def*)
qed

end