9.6 SL, CSL und R6 als Spezialfälle von R6Sep

Ziel: Zeige, dass sich die bisherigen Programmlogiken in R6Sep einbetten lassen.

R6: Um R6 einzubetten, betrachte Programme ohne lokalen Stack.
Wenn alle Stack shared ist, gilt:

$\boxed{H} * \boxed{B}$ gdw. $\boxed{H \wedge B}$.

Damit folgt:

$c : (H, R, G, B)$ gilt in R6

gdw. $c : (\boxed{H}, R, G, \boxed{B})$ gilt in R6Sep.

SL: Um SL einzubetten, betrachte Programme ohne shared Stack.
Wenn alle Stack lokal ist, kann man

$R = G = \emptyset$ wählen.

Damit folgt:

$\{A\}\, c\, \{B\}$ gilt in SL

gdw. $c : (H, \emptyset, \emptyset, B)$ gilt in R6Sep.

CL: Füge die Ressourcen-Invariante der Spezifikation hinzu.
Wähle die Relies und Guarantees so,
dass sie die Ressourcen-Invariante unberührt lassen.
Damit folgt:

$I \vdash \{H\}\, c\, \{B\}$ gilt in CSL

gdw. $c : (H * \boxed{I}, \{I \leadsto I\}, \{I \leadsto I\}, B * \boxed{I})$

gilt in R6Sep.

Den Begriff der Gültigkeit hatten wir für RGIsp noch gar nicht definiert:

Definition:

$c : (P, R, G, Q)$ gilt, falls

- $modifies(c) \cap free(R,G) = \emptyset$
- $\neg locked(c)$ // innerhalb darf nicht vorkommen.
- $\forall s, h_L, h_S.\ [\![P]\!](s, h_L, h_S) \Rightarrow safe(c, s, h_L, h_S, R, G, Q)$

mit

- $safe_0(c, s, h_L, h_S, R, G, Q) := true$
- $safe_{n+1}(c, s, h_L, h_S, R, G, Q) :=$

(i) $c = skip \Rightarrow [\![Q]\!](s, h_L, h_S) = true$

// Bei Terminierung gilt die Postcondition.

(ii) $\forall h_F.\ (c, s, h_L \cdot h_S \cdot h_F) \not\to$ abort und

// Das Programm abortet nicht

(iii) $\forall c', s', h'.\ (c, s, h_L \cdot h_S \cdot h_F) \to (c', s', h')$

$\exists h_L', h_S'.$ • $h' = h_L' \cdot h_S' \cdot h_F$

• $(s, h_S, h_S') \in [\![G]\!]$

• $safe_n(c', s', h_L', h_S', R, G, Q)$

// Bei Transitionen sind wir für weitere n Schritte safe.

(iv) $\forall h_S'.\ (s, h_S, h_S') \in [\![R]\!] \wedge h_L \perp h_S' \Rightarrow safe_n(c, s, h_L, h_S', R, G, Q)$

// Wir sind trotz Interference safe.