External and Internal Choice with Event Groups in Event-B

Michael Butler

University of Southampton

March 8, 2012

Outline

1. Motivating examples: internal and external choice
2. Traces and refusal sets (based on CSP)
3. Existing definition of machine failures
4. New definition of machine failures using event groups
5. Refinement rule with groups

Simple vending machines with external (VM1) and internal (VM2) choice

machine $VM1$
variables $m1 \in \{idle, vend\}$
initialisation
$m1 := idle$
events

Coin $\widehat{=}$ **when**
$m1 = idle$
then
$m1 := vend$
end

Tea $\widehat{=}$ **when**
$m1 = vend$
then
$m1 := idle$
end

Coffee $\widehat{=}$ **when**
$m1 = vend$
then
$m1 := idle$
end

Simple vending machines with external (VM1) and internal (VM2) choice

machine $VM1$
variables $m1 \in \{idle, vend\}$
initialisation
$m1 := idle$
events
Coin $\widehat{=}$ **when** $m1 = idle$ **then** $m1 := vend$ **end**
Tea $\widehat{=}$ **when** $m1 = vend$ **then** $m1 := idle$ **end**
Coffee $\widehat{=}$ **when** $m1 = vend$ **then** $m1 := idle$ **end**

machine $VM2$
variables $m2 \in \{idle, tea, coffee\}$
initialisation
$m2 := idle$
events
Coin $\widehat{=}$ **when** $m2 = idle$ **then** $m2 :\in \{tea, coffee\}$ **end**
Tea $\widehat{=}$ **when** $m2 = tea$ **then** $m2 := idle$ **end**
Coffee $\widehat{=}$ **when** $m2 = coffee$ **then** $m2 := idle$ **end**

In Rodin VM2 refines VM1 but this is not really satisfactory.

Simple transaction system and its refinement

machine $Transaction1$
variables ts, db
invariants

$ts \in \{pending, success, abort\}$
$db \in DataBase$

initialisation

$ts := pending \quad || \quad db := DB_0$

events

Update $\widehat{=}$ **when**
  $ts = pending$
then
  $ts := success$
  $db := update(db)$
end

Abort $\widehat{=}$ **when**
  $ts = pending$
then
  $ts := abort$
end

Simple transaction system and its refinement

machine $\mathit{Transaction1}$
variables $ts,\ db$
invariants
$ts \in \{pending, success, abort\}$
$db \in DataBase$
initialisation
$ts := pending \ \ ||\ \ db := DB_0$
events
Update $\widehat{=}$ **when**
$ts = pending$
then
$ts := success$
$db := update(db)$
end
Abort $\widehat{=}$ **when**
$ts = pending$
then
$ts := abort$
end

machine $\mathit{Transaction2}$
variables $ts,\ db,\ f$
invariants
$ts \in \{pending, success, abort\}$
$db \in DataBase$
$f \in Bool$
initialisation
$ts := pending \ \ ||\ \ db := DB_0 \ \ ||\ \ f :\in Bool$
events
Update $\widehat{=}$ **when**
$ts = pending \wedge f = false$
then
$ts := success$
$db := update(db)$
end
Abort $\widehat{=}$ **when**
$ts = pending \wedge f = true$
then
$ts := abort$
end

In Rodin Transaction2 refines Transaction1 but this is satisfactory.

CSP-like Traces and failures of vending machines

VM1 and VM2 have the same event traces:

$$\langle Coin, Tea\rangle, \langle Coin, Tea, Coin, Coffee\rangle, \langle Coin, Tea, Coin, Tea\rangle,$$
$$\langle Coin, Coffee\rangle, \langle Coin, Coffee, Coin, Tea\rangle, \langle Coin, Coffee, Coin, Coffee\rangle, \cdot$$

CSP-like Traces and failures of vending machines

VM1 and VM2 have the same event traces:

$$\langle Coin, Tea\rangle, \langle Coin, Tea, Coin, Coffee\rangle, \langle Coin, Tea, Coin, Tea\rangle,$$
$$\langle Coin, Coffee\rangle, \langle Coin, Coffee, Coin, Tea\rangle, \langle Coin, Coffee, Coin, Coffee\rangle, \cdot$$

Failures of VM1 - choice in VM1 is external:

$$(\ \langle\rangle, \{Tea, Coffee\}\ )$$
$$(\ \langle Coin\rangle, \{Coin\}\ )$$
$$(\ \langle Coin, Tea\rangle, \{Tea, Coffee\}\ )\ \cdots$$

CSP-like Traces and failures of vending machines

VM1 and VM2 have the same event traces:

$$\langle Coin, Tea\rangle, \langle Coin, Tea, Coin, Coffee\rangle, \langle Coin, Tea, Coin, Tea\rangle,$$
$$\langle Coin, Coffee\rangle, \langle Coin, Coffee, Coin, Tea\rangle, \langle Coin, Coffee, Coin, Coffee\rangle, \cdot$$

Failures of VM1 - choice in VM1 is external:

$$(\ \langle\rangle, \{Tea, Coffee\}\ )$$
$$(\ \langle Coin\rangle, \{Coin\}\ )$$
$$(\ \langle Coin, Tea\rangle, \{Tea, Coffee\}\ )\ \cdots$$

Failures of VM2 - choice in VM1 is internal:

$$(\ \langle\rangle, \{Tea, Coffee\}\ )$$
$$(\ \langle Coin\rangle, \{Coin, Tea\}\ ),\quad (\ \langle Coin\rangle, \{Coin, Coffee\}\ )\quad \cdots$$

CSP-like Traces and failures of vending machines

VM1 and VM2 have the same event traces:

$$\langle Coin, Tea \rangle, \langle Coin, Tea, Coin, Coffee \rangle, \langle Coin, Tea, Coin, Tea \rangle,$$
$$\langle Coin, Coffee \rangle, \langle Coin, Coffee, Coin, Tea \rangle, \langle Coin, Coffee, Coin, Coffee \rangle, \cdot$$

Failures of VM1 - choice in VM1 is external:

$$(\ \langle\rangle, \{Tea, Coffee\} \ )$$
$$(\ \langle Coin \rangle, \{Coin\} \ )$$
$$(\ \langle Coin, Tea \rangle, \{Tea, Coffee\} \ ) \ \cdots$$

Failures of VM2 - choice in VM1 is internal:

$$(\ \langle\rangle, \{Tea, Coffee\} \ )$$
$$(\ \langle Coin \rangle, \{Coin, Tea\} \ ), \quad (\ \langle Coin \rangle, \{Coin, Coffee\} \ ) \quad \cdots$$

But VM2 cannot refuse both:

$$(\ \langle Coin \rangle, \{Tea, Coffee\} \ ) \quad \text{is not a refusal}$$

Weakest preconditions

$wp_M(a, Q)$ is the weakest precondition under which event a of M is *guaranteed* to establish postcondition Q

For concatenation fo traces $s; t$ we have

$$wp_M(\ s; t,\ \ Q) \ = \ wp_M(s, wp_M(t, Q))$$

Weakest preconditions

$wp_M(a, Q)$ is the weakest precondition under which event a of M is *guaranteed* to establish postcondition Q

For concatenation fo traces $s; t$ we have

$$wp_M(\ s; t,\ \ Q) \ = \ wp_M(s, wp_M(t, Q))$$

$\overline{wp}$ weakest precondition under which execution of s *might* lead to a state satisfing Q:

$$\overline{wp}_M(s, Q) \ \ \hat{=} \ \ \neg wp_M(s, \neg Q)$$

Weakest preconditions

$wp_M(a, Q)$ is the weakest precondition under which event a of M is *guaranteed* to establish postcondition Q

For concatenation fo traces $s; t$ we have

$$wp_M(\ s; t,\ \ Q) \ = \ wp_M(s, wp_M(t, Q))$$

$\overline{wp}$ weakest precondition under which execution of s *might* lead to a state satisfing Q:

$$\overline{wp}_M(s, Q) \ \ \hat{=} \ \ \neg wp_M(s, \neg Q)$$

a is enabled whenever it is possible to reach some state by executing a:

$$grd_M(a) \ \ \hat{=} \ \ \overline{wp}(a, true)$$

Existing failures semantics

The set of failures of machine M are pairs of the form

$$(s, X)$$

M may engage in trace s after which in may refuse all events in set X

Definition in terms of $\overline{wp}$:

$$(s, X) \in F_M \quad \hat{=} \quad \overline{wp}_M(\ i; s,\ \neg grd_M(X))$$

Existing failures semantics

The set of failures of machine M are pairs of the form

$$(s, X)$$

M may engage in trace s after which in may refuse all events in set X

Definition in terms of $\overline{wp}$:

$$(s, X) \in F_M \quad \hat{=} \quad \overline{wp}_M(\ i; s,\ \neg grd_M(X))$$

Choice between enabled events is external with this definition

From C.C. Morgan. *Of wp and CSP*, 1990. (In terms of action systems rather than Event-B)

Grouping events

In vending machine we want choice between *Tea* and *Coffee* to be external.
In transaction example, we want the choice between *Update* and *Abort* to be internal.

Proposed approach: group events to specify that
- choice between enabled events within a group is internal
- choice between events of different groups is external.

Put *Update* and *Abort* in the same group.
Put *Tea* and *Coffee* is different groups

Group refusal

Assume a is part of an event group G.
a can be refused when a is not enabled or when some other event in G is enabled:

$$\neg grd_M(a) \;\vee\; grd_M(G \setminus \{a\}).$$

Group refusal

Assume a is part of an event group G.
a can be refused when a is not enabled or when some other event in G is enabled:

$$\neg grd_M(a) \;\vee\; grd_M(G \setminus \{a\}).$$

To define the refusal condition for set X, we factor X into its groups.
For group G, the set $X \cap G$ is the events of X in group G.
This is refused when

$$\neg grd_M(X \cap G) \;\vee\; grd_M(G \setminus X)$$

This is simplified to:

$$grd_M(G) \implies grd_M(G \setminus X)$$

New definition

grp_M is the set of groups of M

For $g \in grp_M$, let $evt_M(g)$ be the set of events in g

$$ref_M(X) \quad \hat{=} \quad \bigwedge_{g \in grp_M} grd_M(evts_M(g)) \implies grd_M(evts_M(g) \setminus X)$$

New definition of failures:

$$(s, X) \in F_M \quad \hat{=} \quad \overline{wp}_M(\ i; s,\ ref_M(X)\ )$$

With this definition choice between enabled events within a group is internal and choice between groups is external

Example calculations

Assume groups $\{Coin\}$ and $\{Tea, Coffee\}$ in $VM1$

$$ref_{VM1}(\{Tea\}) \quad = \quad ?$$

$$ref_{VM1}(\{Tea, Coffee\}) \quad = \quad ?$$

Example calculations

Assume groups $\{Coin\}$ and $\{Tea, Coffee\}$ in $VM1$

$$
\begin{aligned}
& ref_{VM1}(\{Tea\}) \\
= \quad & (\ grd(\{Coin\}) \implies grd(\{Coin\})\ ) \quad \wedge \\
& (\ grd(\{Tea, Coffee\}) \implies grd(Coffee)\ ) \\
= \quad & true \ \wedge\ (\ (m = vend \vee m = vend) \implies m = vend\ ) \\
= \quad & true
\end{aligned}
$$

Example calculations

Assume groups $\{Coin\}$ and $\{Tea, Coffee\}$ in $VM1$

$$
\begin{aligned}
& ref_{VM1}(\{Tea\}) \\
= \quad & (\ grd(\{Coin\}) \implies grd(\{Coin\})\ ) \quad \wedge \\
& (\ grd(\{Tea, Coffee\}) \implies grd(Coffee)\ ) \\
= \quad & true \ \wedge\ (\ (m = vend \vee m = vend) \implies m = vend\ ) \\
= \quad & true
\end{aligned}
$$

$$
\begin{aligned}
& ref_{VM1}(\{Tea, Coffee\}) \\
= \quad & (\ grd(\{Coin\}) \implies grd(\{Coin\})\ ) \quad \wedge \\
& (\ grd(\{Tea, Coffee\}) \implies grd(\{\})\ ) \\
= \quad & true \ \wedge\ (\ (m = vend \vee m = vend) \implies false\ ) \\
= \quad & m \neq vend
\end{aligned}
$$

Well-formedness conditions for failures

In CSP, the failures set of a process satisfies the following conditions:

$$
\begin{array}{l}
(\langle\rangle, \{\}) \in F \\
(s;t, X) \in F \quad \Longrightarrow \quad (s, \{\}) \in F \\
(s, X) \in F \ \wedge \ Y \subseteq X \quad \Longrightarrow \quad (s, Y) \in F \\
(s, X) \in F \ \wedge \ a \in A \ \wedge \ a \notin X \quad \Longrightarrow \\
\qquad\qquad (s, X \cup \{a\}) \in F \ \vee \ (s;a, \{\}) \in F
\end{array}
$$

We can prove that the definition of F_M for Event-B machines satisfies these

Data refinement with groups

Refine event M_a by event N_a
General predicate transformer definition

$$rep(wp_M(a, Q)) \implies wp_N(a, rep(Q))$$

$rep(Q)$ is typically defined as $(\exists v \cdot I(v, w) \wedge Q)$
I is the gluing invariant, v are the abstract variables

Data refinement with groups

Refine event M_a by event N_a
General predicate transformer definition

$$rep(wp_M(a, Q)) \implies wp_N(a, rep(Q))$$

$rep(Q)$ is typically defined as $(\exists v \cdot I(v, w) \wedge Q)$
I is the gluing invariant, v are the abstract variables

Machine refinement:
(i) $wp_M(i, Q) \implies wp_N(i, rep(Q))$
(ii) $rep(wp_M(a, Q)) \implies wp_N(a, rep(Q))$, each $a \in A$
(iii) $rep(grd_M(\ evt_M(g)\ )) \implies grd_N(\ evt_M(g)\ )$, each $g \in grp_M$

$grd_M(\ evt_M(g)\ )$ is the disjunction of the guards of events in g

Refinement by group subsetting

Splitting an event group means we are converting internal choice to external choice

Suppose *VM1* has 2 groups:

$$G1 = \{Coin\}, \quad G2 = \{Tea, Coffee\}$$

We can change the grouping to be:

$$H1 = \{Coin\}, \quad H2 = \{Tea\}, \quad H3 = \{Coffee\}$$

This kind of event splitting is always is a valid refinement step