MODULE *Agent*

EXTENDS *FiniteSets*, *Naturals*, *Sequences*, *TLC*

CONSTANTS	*Emails*	Set of incoming *Emails*
VARIABLES	*Archived*,	Set of archived *Emails*
	Arrived,	Queue of incoming *Emails*
	Completed,	Queue of completion responses
	RemoteOutbox,	Set of outgoing *Emails*
	Parsed,	Set of parsed *Emails*
	Abandoned	Set of failed *Emails*

$vars \triangleq \langle Abandoned, Archived, Arrived, Completed, Parsed, RemoteOutbox \rangle$
$EmailsInQueue \triangleq Abandoned \cup Archived \cup Arrived \cup Completed \cup Parsed$

$TypeOK \triangleq$
$\land\ Abandoned \subseteq Emails$
$\land\ Archived \subseteq Emails$
$\land\ Arrived \subseteq Emails$
$\land\ Completed \subseteq Emails$
$\land\ Parsed \subseteq Emails$
$\land\ RemoteOutbox \in Seq(Emails)$

$Range(S) \triangleq \{S[n] : n \in \text{DOMAIN } S\}$

$Invariants \triangleq$

$\land\ \forall\, email \in Completed : email \notin Parsed \Rightarrow email \notin Arrived$

Don't parse e-mails more than once.

$\land\ \forall\, email \in Range(RemoteOutbox) : email \notin Completed \Rightarrow email \notin Parsed$

Don't complete e-mails more than once.

$\land\ \forall\, email \in Abandoned : email \notin Arrived \cup Completed \cup Parsed$

Abandoned e-mails not to appear anywhere else, as *Abandoned* is a general queue state separate from e-mail processing state.

$\land\ \forall\, email \in Archived : email \notin Arrived \cup Completed \cup Parsed$

Same with archived emails.

$\land\ Len(RemoteOutbox) = Cardinality(Range(RemoteOutbox))$

Don't send e-mails more than once.

$ReceiveEmailOK(email) \triangleq$

Enqueues an *Email* from *Inbox* to *Arrived*.

$\land\ Arrived' = Arrived \cup \{email\}$
$\land\ \text{UNCHANGED } \langle Abandoned, Archived, Completed, Parsed, RemoteOutbox \rangle$

$ReceiveEmailError(email) \triangleq$

Fails reading an *email* from *Inbox*. Logs it, marks it and moves it to *RemoteArchived* folder. Support engineer can move the *email* back to *Inbox* after addressing the issue.

$\land\ Abandoned' = Abandoned \cup \{email\}$

$\quad \land \text{UNCHANGED } \langle \mathit{Archived}, \mathit{Arrived}, \mathit{Completed}, \mathit{Parsed}, \mathit{RemoteOutbox} \rangle$

$\mathit{ReceiveEmail} \triangleq \land \exists\, \mathit{email} \in \mathit{Emails} \setminus \mathit{EmailsInQueue} :$
$\qquad\qquad\qquad \lor \mathit{ReceiveEmailOK}(\mathit{email})$
$\qquad\qquad\qquad \lor \mathit{ReceiveEmailError}(\mathit{email})$

$\mathit{ParseEmail1OK}(\mathit{email}) \triangleq$

The first step of parsing an e-mail response stores the parsed content in the queue.

$\quad \land \mathit{email} \notin \mathit{Parsed}$
$\quad \land \mathit{Parsed}' = \mathit{Parsed} \cup \{\mathit{email}\}$
$\quad \land \text{UNCHANGED } \langle \mathit{Abandoned}, \mathit{Archived}, \mathit{Arrived}, \mathit{Completed}, \mathit{RemoteOutbox} \rangle$

$\mathit{ParseEmail2OK}(\mathit{email}) \triangleq$

The second step of parsing removes the e-mail response from the queue only after the parsing is successful. This ensures we don't lose any e-mails in case of a failure.

$\quad \land \mathit{email} \in \mathit{Parsed}$
$\quad \land \mathit{Arrived}' = \mathit{Arrived} \setminus \{\mathit{email}\}$
$\quad \land \text{UNCHANGED } \langle \mathit{Abandoned}, \mathit{Archived}, \mathit{Completed}, \mathit{Parsed}, \mathit{RemoteOutbox} \rangle$

$\mathit{ParseEmailOK}(\mathit{email}) \triangleq$

Parses an *email*. The sub-operations occur over distributed settings and may fail. Each sub-operation is atomic, and their order of execution is important.

$\quad \lor \mathit{ParseEmail1OK}(\mathit{email})$
$\quad \lor \mathit{ParseEmail2OK}(\mathit{email})$

$\mathit{ParseEmail1Error}(\mathit{email}) \triangleq$

Fails parsing an *email*.

$\quad \land \mathit{email} \notin \mathit{Parsed}$
$\quad \land \mathit{Abandoned}' = \mathit{Abandoned} \cup \{\mathit{email}\}$
$\quad \land \mathit{Arrived}' = \mathit{Arrived} \setminus \{\mathit{email}\}$
$\quad \land \text{UNCHANGED } \langle \mathit{Archived}, \mathit{Completed}, \mathit{Parsed}, \mathit{RemoteOutbox} \rangle$

$\mathit{ParseEmail} \triangleq$
$\quad \exists\, \mathit{email} \in \mathit{Arrived} \setminus \mathit{Abandoned} :$
$\qquad \lor \mathit{ParseEmailOK}(\mathit{email})$
$\qquad \lor \mathit{ParseEmail1Error}(\mathit{email})$

$\mathit{CompleteMessage1OK}(\mathit{email}) \triangleq$
$\quad \land \mathit{email} \notin \mathit{Completed}$
$\quad \land \mathit{Completed}' = \mathit{Completed} \cup \{\mathit{email}\}$
$\quad \land \text{UNCHANGED } \langle \mathit{Abandoned}, \mathit{Archived}, \mathit{Arrived}, \mathit{Parsed}, \mathit{RemoteOutbox} \rangle$

$\mathit{CompleteMessage2OK}(\mathit{email}) \triangleq$
$\quad \land \mathit{email} \in \mathit{Completed}$
$\quad \land \mathit{Parsed}' = \mathit{Parsed} \setminus \{\mathit{email}\}$
$\quad \land \text{UNCHANGED } \langle \mathit{Abandoned}, \mathit{Archived}, \mathit{Arrived}, \mathit{Completed}, \mathit{RemoteOutbox} \rangle$

$$\begin{array}{l}
CompleteMessageOK(email) \triangleq \\
\quad \lor CompleteMessage1OK(email) \\
\quad \lor CompleteMessage2OK(email)
\end{array}$$

$$\begin{array}{l}
CompleteMessage1Error(email) \triangleq \\
\quad \land email \notin Completed \\
\quad \land Abandoned' = Abandoned \cup \{email\} \\
\quad \land Parsed' = Parsed \setminus \{email\} \\
\quad \land \text{UNCHANGED } \langle Archived, Arrived, Completed, RemoteOutbox \rangle
\end{array}$$

$$\begin{array}{l}
CompleteMessage \triangleq \\
\quad \exists\, email \in Parsed \setminus (Arrived \cup Abandoned) : \\
\qquad \lor CompleteMessageOK(email) \\
\qquad \lor CompleteMessage1Error(email)
\end{array}$$

$SendOutCompletion1OK(email) \triangleq$

Sends out a completion response e-mail.

$$\begin{array}{ll}
\quad \land email \notin Range(RemoteOutbox) & \text{We haven't already sent this e-mail} \\
\quad \land RemoteOutbox' = Append(RemoteOutbox, email) & \\
\quad \land \text{UNCHANGED } \langle Abandoned, Archived, Arrived, Completed, Parsed \rangle &
\end{array}$$

$SendOutCompletion2OK(email) \triangleq$

Marks an *email* as sent.

$$\begin{array}{ll}
\quad \land email \in Range(RemoteOutbox) & \text{Previous step to send this e-mail succeeded.} \\
\quad \land Archived' = Archived \cup \{email\} & \\
\quad \land Completed' = Completed \setminus \{email\} & \\
\quad \land \text{UNCHANGED } \langle Abandoned, Arrived, Parsed, RemoteOutbox \rangle &
\end{array}$$

$SendOutCompletion1Error(email) \triangleq$

Fails sending the e-mail.

$$\begin{array}{ll}
\quad \land email \notin Range(RemoteOutbox) & \text{We haven't already sent this e-mail} \\
\quad \land Abandoned' = Abandoned \cup \{email\} & \\
\quad \land Completed' = Completed \setminus \{email\} & \\
\quad \land \text{UNCHANGED } \langle Archived, Arrived, Parsed, RemoteOutbox \rangle &
\end{array}$$

$$\begin{array}{l}
SendOutCompletion \triangleq \\
\quad \exists\, email \in Completed \setminus (Abandoned \cup Parsed) : \\
\qquad \lor SendOutCompletion1OK(email) \\
\qquad \lor SendOutCompletion2OK(email) \\
\qquad \lor SendOutCompletion1Error(email)
\end{array}$$

$AllDone \triangleq$

All done and system comes to equilibrium.

$$\begin{array}{l}
\quad \land Archived \cup Abandoned = Emails \\
\quad \land Parsed \setminus Abandoned = \{\}
\end{array}$$

$\quad \land$ UNCHANGED $vars$

$$
\begin{aligned}
Init \triangleq\ & \land Abandoned = \{\} \\
& \land Archived = \{\} \\
& \land Arrived = \{\} \\
& \land Completed = \{\} \\
& \land Parsed = \{\} \\
& \land RemoteOutbox = \langle\rangle
\end{aligned}
$$

$$
\begin{aligned}
Next \triangleq\ & \lor ReceiveEmail \\
& \lor ParseEmail \\
& \lor CompleteMessage \\
& \lor SendOutCompletion \\
& \lor AllDone
\end{aligned}
$$

$$Spec \triangleq Init \land \Box[Next]_{vars} \land \mathrm{WF}_{vars}(Next)$$

Temporal properties for verification

$NoLostEmails \triangleq$

No e-mails should be lost. This is a safety property.

$$
\begin{aligned}
& \forall\, email \in Emails : \\
& \quad \Box(email \in EmailsInQueue \Rightarrow \Diamond\Box(email \in Abandoned \cup Range(RemoteOutbox)))
\end{aligned}
$$

THEOREM $Spec \Rightarrow \Box\, TypeOK$

THEOREM $Spec \Rightarrow \Box Invariants$

THEOREM $Spec \Rightarrow NoLostEmails$

\* Modification History

\* Last modified *Wed* May 03 16:27:57 *KST* 2023 by *hcs*

\* Created *Fri Apr* 28 13:04:37 *KST* 2023 by *hcs*