Proof obligations for specification and refinement of liveness properties under weak fairness

Hector Ruiz Barradas, Didier Bert

IMAG

**Institut d'Informatique et de
Mathématiques Appliquées
de Grenoble**

LSR

Laboratoire Logiciels, Systèmes, Réseaux

RAPPORT DE RECHERCHE

Proof Obligations for Specification and Refinement of Liveness Properties under Weak Fairness

Héctor Ruíz Barradas[1,2] and Didier Bert[2]

[1] Universidad Autónoma Metropolitana Azcapotzalco, México D. F., México
`hrb@correo.azc.uam.mx, Hector.Ruiz@imag.fr`
[2] Laboratoire Logiciels, Systèmes, Réseaux - LSR-IMAG - Grenoble, France
`Didier.Bert@imag.fr`

RR 1071-I LSR 20 Février 2005

B.P. 72 - 38402 SAINT MARTIN D'HERES CEDEX - France

Centre National de la Recherche Scientifique
Institut National Polytechnique de Grenoble
Université Joseph Fourier Grenoble I

Proof Obligations for Specification and Refinement of Liveness Properties under Weak Fairness

Abstract

In this report we present a formal model of fair iteration of events in a B event system. The model is used to justify proof obligations for basic liveness properties and preservation under refinement of general liveness properties. The model of fair iteration of events uses the dovetail operator, an operator proposed in [4] to model fair choice. The proofs are mainly founded in fixpoint calculations of fair iteration of events and weakest precondition calculus.

Keywords

Liveness properties, Event systems, B method, Unity logic, Refinement, Fairness, Weak Fairness.

Résumé

Dans ce rapport nous présentons un modèle formel d'itération équitable d'événements dans un système B événementiel. Le modèle est utilisé pour justifier des obligations de preuve des propriétés de vivacité de base et de la préservation dans les raffinements des propriétés de vivacité générales. Le modèle d'itération équitable d'événements utilise l'opérateur *dovetail*, un opérateur proposé dans [4] pour modéliser une sélection équitable. Les preuves sont fondées principalement sur des calculs de point fixe de l'itération équitable d'événements et le calcul des plus faibles préconditions.

Mots-clés

Propriétés de vivacité, système d'événements, méthode B, logique Unity, rafinement, équité, équité faible.

Table of Contents

1 Introduction

In [9] we proposed the specification and proof of liveness properties under a weak fairness assumption in B events systems [2]. The syntax and semantic of liveness properties that we adopted are similar to the ones used in UNITY [5].

Liveness properties are divided in two classes: basic liveness properties and general liveness properties. Basic properties are specified by the *ensures* relation $\gg_w$. General liveness properties are specified by the *leads to* relation $\leadsto$. $\gg_w$ and $\leadsto$ are relations between predicates on the system state.

We proposed two proof obligations for basic liveness properties founded on weakest precondition calculus. The proof of general liveness properties is made by applying inference rules of the UNITY logic.

Following the B method, an abstract model can be refined in a more concrete one. To preserve through refinement liveness properties specified in abstract models, we proposed two other proof obligations. One proof obligation is discharged by applying weakest precondition calculus, an the other one need to identify basic liveness properties in the refinement and to apply the UNITY logic.

The goal of this report is to justify the proof obligations concerning proofs of basic liveness properties and preservation of general liveness properties under refinement, by a reasoning on the set theoretic formulation of event systems. Our approach was inspired by [2], where proof obligations concerning modalities are justified by fixpoints of iteration of events, instead of a reasoning over the set of traces in a system, as it is done in [3]. However, our approach uses a model including a fair choice operator, which allows us to model our weak fairness assumption over the iteration of events.

This report is structured as follows. In section 2 we present the main definitions used in this work. In particular we define the liberal set transformer for events in a B system and we present the dovetail operator which is used to model our fairness assumption. In section 3 we introduce the proof obligations for basic liveness properties and we prove that they are sufficient conditions to guarantee that fair iteration of events in the system, terminates in a state satisfying the postcondition established by the basic liveness property. In section 4 we present how to specify and prove general liveness properties. Moreover, we give two proof obligations to guarantee preservation of general liveness properties under refinement, and we demonstrate they are sufficient conditions to ensure that fair iteration of refined events terminates into a state satisfying the predicate established by the general liveness property. In section 5 we give the conclusions of this report and some comments about the future work.

2 The dovetail operator

In [4] the dovetail operator, a fair nondeterministic choice operator, is introduced. In this section we give the definition of this operator by its weakest liberal transformer. In the first part of this section we define the weakest liberal set transformer of events in a B event system. In the second part we give the formal definition of the dovetail operator by definition of its weakest liberal set transformer and its termination set.

2.1 The Liberal Set Transformer

In [1], each generalized substitution S has associated a set transformer $\mathsf{str}(S)$ of type $\mathbb{P}(u) \rightarrow \mathbb{P}(u)$, where u is the state space of a machine or refinement. For any r in $\mathbb{P}(u)$, $\mathsf{str}(S)(r)$

denotes the largest subset of states where the execution of S must begin in order for the substitution S to terminate in a state belonging to r. In [2], the events of a B system are formalized by conjunctive set transformers, but instead of identifying the set transformer associated with an event F by $\mathsf{str}(F)$, it is denoted by its name F. In this way $F(r)$ denotes the set $\mathsf{str}(F)(r)$, where F is an event of a B system and r a subset of the state space u. In what follows, we use this notation.

In order to deal with the notion of the weakest liberal precondition of an event F we define *the liberal set transformer* of an event F as $\mathcal{L}(F)$.

Definition 1. *The Liberal Set Transformer*

$$\mathcal{L}(F) = \lambda r \cdot (r \in \mathbb{P}(u) \mid \{x \mid x \in u \wedge wlp(F, x \in r)\})$$

The set $\mathcal{L}(F)(r)$ denotes the largest subset of states where the execution of event F must begin in order for F to terminate in a state belonging to r or loop. The liberal set transformer of the events in a B system are defined as follows:

$$\begin{array}{ll} \mathcal{L}(skip)(r) = r & \\ \mathcal{L}(F \mathbin{[\!]} G)(r) = \mathcal{L}(F)(r) \cap \mathcal{L}(G)(r) & \mathcal{L}(F\ ;\ G)(r) = \mathcal{L}(F)(\mathcal{L}(G)(r)) \\ \mathcal{L}(p \mid F)(r) = (p \cup \{x \mid x \in u \wedge u \subseteq r\}) \cap \mathcal{L}(F)(r) & \mathcal{L}(p \Longrightarrow F)(r) = \overline{p} \cup \mathcal{L}(F)(r) \end{array}$$

where r and p are subsets of u and $\overline{p}$ is $u - p$. We note, that set $\{x \mid x \in u \wedge u \subseteq r\}$ in the liberal set transformer of the preconditioned event may have only two values: $\varnothing$ for $r \neq u$ or u for $r = u$. In the guarded command $p \Longrightarrow F$ and the preconditioned event $p \mid F$, we follow the notation introduced in [2] where the guard or the precondition of the commands is a set instead of a predicate. Definitions of liberal set transformers presented here are the set counterpart of definitions in [7].

We remark that definitions of liberal set transformer of any set transformer S, made up of set transformers F or G, such that $\mathcal{L}(F)(u) = u$ and $\mathcal{L}(G)(u) = u$, and operators $[\!]$, $|$, $\Longrightarrow$ and $;$, respect $\mathcal{L}(S)(u) = u$ [6].

The set transformers $F(r)$ and $\mathcal{L}(F)(r)$ for event F and postcondition r are related by the pairing condition:

$$F(r) = \mathcal{L}(F)(r) \cap \mathsf{pre}(F) \tag{1}$$

where $\mathsf{pre}(F)$, the termination set of F is equal to $F(u)$. From the pairing condition we conclude the implication:

$$F(u) = u \Rightarrow F(r) = \mathcal{L}(F)(r) \quad \text{for any } r \text{ in } \mathbb{P}(u) \tag{2}$$

which indicates that the set transformer F and $\mathcal{L}(F)$ are the same provided the event F always terminates. When $F(r)$ or $\mathcal{L}(F)(r)$ are recursively defined:

$$F(r) = \mathcal{F}(F(r)) \quad \text{or} \quad \mathcal{L}(F)(r) = \mathcal{G}(\mathcal{L}(F)(r))$$

for monotonic functions $\mathcal{F}$ and $\mathcal{G}$, according to [8] we take $F(r)$ as the strongest solution of the equation $X = \mathcal{F}(X)$ and $\mathcal{L}(F)(r)$ as the weakest solution of the equation $X = \mathcal{G}(X)$. As these solutions are fixpoints, we take $F(r)$ as the least fixpoint of $\mathcal{F}$ ($\mathsf{fix}(\mathcal{F})$) and $\mathcal{L}(F)(r)$ as the greatest fixpoint of $\mathcal{G}$ ($\mathsf{FIX}(\mathcal{G})$).

2.2 Definition of Dovetail Operator

The dovetail operator is used to model the notion of fair scheduling of two activities. Let A and B be these activities, then the operational meaning of the construct $A \triangledown B$ denotes the execution of commands A and B fairly in parallel, on separate copies of the state, accepting as an outcome any proper, nonlooping, outcome of either A or B. The fair execution of A and B means that neither computation is permanently neglected if favor of the other.

A motivating example of the use of the dovetail operator is given in [4]. In that example the recursive definition: $X = (n := 0 \triangledown (X \ ; n := n + 1))$ which has as solution "set n to any natural number", is contrasted with the recursion $Y = (n := 0 \, [\!] \, (Y \ ; n := n + 1))$ which has as solution "set n to any natural number or loop". The possibility of loop in X is excluded with the dovetail operator because the fair choice of statement $n := 0$ will certainly occur. In Y the execution of that statement is not ensured.

The semantic definition for dovetail operator in [4] is given by definition of its weakest liberal precondition predicate transformer (wlp) and its termination predicate hlt. We give an equivalent definition using the weakest liberal set transformer $\mathcal{L}$ and its termination set pre:

Definition 2. *The Dovetail Operator*

$$\begin{aligned}
&\mathcal{L}(F \triangledown G)(r) = \mathcal{L}(F)(r) \cap \mathcal{L}(G)(r)\\
&\mathsf{pre}(F \triangledown G) = (F(u) \cup G(u)) \cap (\overline{F(\varnothing)} \cup G(u)) \cap (\overline{G(\varnothing)} \cup F(u))\\
&\mathsf{pre}(F \triangledown G) = (F(u) \cap G(u)) \cup (\overline{F(\varnothing)} \cap F(u)) \cup (\overline{G(\varnothing)} \cap G(u))
\end{aligned}$$

The two definitions of the termination set $\mathsf{pre}(F \triangledown G)$ are equivalents; it can be proved by distribution of union over intersection. In another hand we remember that $\mathsf{grd}(F) = \overline{F(\varnothing)}$.

The set transformer $(F \triangledown G)(r)$, for any r in $\mathbb{P}(u)$ associated with the dovetail operator is obtained from the pairing condition (1):

$$(F \triangledown G)(r) = \mathcal{L}(F \triangledown G)(r) \cap \mathsf{pre}(F \triangledown G) \tag{3}$$

We note that as far as the liberal set transformed is concerned, the dovetail operator is equal to the choice operator. It differs by having a more liberal pairing condition: to ensures that $F \triangledown G$ halts, it suffices to forbid F and G from both looping and to forbid either from looping in a state where the other fails.

As in [4], we can prove: $grd(F \triangledown G) = grd(F) \vee grd(G)$, but we give a shorter proof than [4] in terms of sets. We prove:

$$\overline{(F \triangledown G)(\varnothing)} = \overline{F(\varnothing)} \cup \overline{G(\varnothing)} \tag{4}$$

Proof

$$\overline{F(\varnothing)} \cup \overline{G(\varnothing)}$$

$$=$$

$$\overline{F(\varnothing) \cap G(\varnothing)}$$

$$= \qquad \{ \text{ Pairing Condition } \}$$

$$\overline{\mathcal{L}(F)(\varnothing) \cap \mathcal{L}(G)(\varnothing) \cap F(u) \cap G(u)}$$

$$= \qquad \{ \; F(\varnothing) \cap \overline{F(\varnothing)} = \varnothing \; \}$$

$$\overline{\mathcal{L}(F)(\varnothing) \cap \mathcal{L}(G)(\varnothing) \cap (F(u) \cap G(u) \cup F(\varnothing) \cap \overline{F(\varnothing)})}$$

$$= \qquad \{ \; \mathcal{L}(F)(\varnothing) \cap F(\varnothing) = \mathcal{L}(F)(\varnothing) \cap F(u) \text{ See note below } \}$$

$$\overline{\mathcal{L}(F)(\varnothing) \cap \mathcal{L}(G)(\varnothing) \cap (F(u) \cap G(u) \cup F(u) \cap \overline{F(\varnothing)})}$$

$$= \qquad \{ \text{ Similar to two last steps } \}$$

$$\overline{\mathcal{L}(F)(\varnothing) \cap \mathcal{L}(G)(\varnothing) \cap (F(u) \cap G(u) \cup F(u) \cap \overline{F(\varnothing)} \cup G(u) \cap \overline{G(\varnothing)})}$$

$$= \qquad \{ \text{ Definition of } (F \triangledown G)(\varnothing) \text{ (2) and (3) } \}$$

$$\overline{(F \triangledown G)(\varnothing)}$$

□*Note* In [4] this step requires the proof of *wlp.F.false* $\Rightarrow$ *grd.F* $= \neg$*hlt.F*. We denote this implication as a set expression: $\mathcal{L}(F)(\varnothing) \cap F(\varnothing) = \mathcal{L}(F)(\varnothing) \cap F(u)$. However the proof of this expression is easily given by the pairing condition. We finally note that the sets $F(\varnothing)$ and $F(u)$ are not equals as we can think from the given equality; only the intersection of these sets with $\mathcal{L}(F)(\varnothing)$ is equal.

The dovetail operator is in general non monotonic for the approximation order in commands as defined in [4]. Therefore the existence of least fixed points of recursive equations cannot be proved generally. However, the existence of least fixed points in a restricted class of recursive definitions containing the dovetail operator, is proved in [4]. In this report we only use the dovetail operator to model fair iteration of events. We do not propose the use of this operator to model or refine B event systems. The set transformer modeling fair iteration of events with the dovetail operator is monotonic in the set inclusion order.

3 Basic Liveness Properties

Let S be a B event system with state variable x and invariant I, made up of a family of events indexed by a certain index set L:

$$S \mathrel{\widehat{=}} [\!]_{i \in L} \; F_i$$

where $[\!]_{i \in L} \; F_i$ denotes the choice of events F_i over a set L. Since we cannot ensure the execution of continuously enabled events with an infinite set of events in a system, as required by the weak fairness assumption, the set of labels L must be finite. Let P and Q be two predicates on the state of S. A basic liveness property, specified by the relation *ensures* ($\gg_w$) as:

$$G \cdot P \gg_w Q$$

(pronounce “by event G, P ensures Q”), where $G = [\!]_{i \in K} \; F_i$ and K is a non empty subset of L, indicates that by the execution of event G in a state where the state variable x satisfies P, the system goes to another state where the state variable satisfies Q, under a weak fairness assumption.

The sufficient conditions to guarantee that system S satisfies the property $G \cdot P \gg_w Q$ are:

	ANTECEDENT	CONSEQUENT
WF0	$I \wedge P \wedge \neg Q \Rightarrow [S]\, P \vee Q$	$G \cdot P \gg_w Q$
WF1	$I \wedge P \wedge \neg Q \Rightarrow \mathit{grd}(G) \wedge [G]\, Q$	

If we consider the choice of events which does not establish postcondition Q, we can restate the proof obligations as follows:

	ANTECEDENT	CONSEQUENT
WF0'	$I \wedge P \wedge \neg Q \Rightarrow [F]\, P \vee Q$	$G \cdot P \gg_w Q$
WF1'	$I \wedge P \wedge \neg Q \Rightarrow \mathit{grd}(G) \wedge [G]\, Q$	

where $F = [\!]_{i \in L-K}\, F_i$. As we have $S = F \;[\!]\; G$, we can prove the equivalence between the antecedents of WF0 and WF1 with WF0' and WF1'.

In the following section we proof that WF0 and WF1 are indeed sufficient conditions to guarantee that by the execution of event G in a state satisfying P, the system goes to another state satisfying Q, under a weak fairness assumption. However, as we prove our rules in a set theoretical framework, we give an equivalent definition of proof obligations WF0 and WF1 in term of set transformers. In this way, each event F_i in B system S, is considered as a set transformer of type $\mathbb{P}(u) \rightarrow \mathbb{P}(u)$, where $u = \{z \mid I\}$ is the set of states satisfying invariant I. According to [1], the set transformer $\mathsf{str}(F_i)$ is defined as follows:

$$\mathsf{str}(F_i) = \lambda r \cdot (r \in \mathbb{P}(u) \mid \{z \mid z \in u \wedge [F_i]\, z \in r\})$$

Following the notation introduced in [2], we use names of events to denote set transformers. In this way $F_i(r)$ denotes the largest subset of u, where the execution of event F_i must start in order to terminate in a state belonging to r. Now, considering the sets:

$$p = \{z \mid z \in u \wedge P\}$$
$$q = \{z \mid z \in u \wedge Q\}$$

the inclusions

$$p \cap \overline{q} \subseteq S(p \cup q) \tag{5}$$
$$p \cap \overline{q} \subseteq \mathsf{grd}(G) \cap G(q) \tag{6}$$

are equivalent to WF0 and WF1 respectively. To prove the equivalences we assume that $I \Rightarrow [S]\, I$ holds. Then we have:

Proof

$\quad WF0$

$\Rightarrow$ $\qquad$ { Def. of WF0 and assumption }

$\quad \forall x \cdot (I \wedge P \wedge \neg Q \Rightarrow [S]\,(P \vee Q)) \wedge \forall x \cdot (I \Rightarrow [S]\,I)$

$\Rightarrow$ $\qquad$ { S is conjunctive }

$\quad \forall x \cdot (I \wedge P \wedge \neg Q \Rightarrow [S]\,((P \vee Q) \wedge I))$

$\equiv$ $\qquad$ { def. p, q and set. trans. }

$\quad \forall x \cdot (x \in p \cap \overline{q} \Rightarrow x \in S(p \cup q))$

$\equiv$

$\quad p \cap \overline{q} \subseteq S(p \cup q)$

$\equiv$

$\quad \forall x \cdot (I \wedge P \wedge \neg Q \Rightarrow [S]\,((P \vee Q) \wedge I))$

$\Rightarrow$ $\qquad$ { weakening }

$\quad WF0$

$\quad WF1$

$\Rightarrow$ $\qquad$ { Def. of WF1 and assumption }

$\quad \forall x \cdot (I \wedge P \wedge \neg Q \Rightarrow grd(G) \wedge [G]\,Q) \wedge \forall x \cdot (I \Rightarrow [S]\,I)$

$\Rightarrow$ $\qquad$ { G is conjunctive }

$\quad \forall x \cdot (I \wedge P \wedge \neg Q \Rightarrow grd(G) \wedge [G]\,(Q \wedge I))$

$\equiv$ $\qquad$ { def. p, q }

$\quad \forall x \cdot (x \in p \cap \overline{q} \Rightarrow \neg([G]\,\mathit{false}) \wedge [G]\,x \in q)$

$\equiv$ $\qquad$ { Def. set. trans. }

$\quad \forall x \cdot (x \in p \cap \overline{q} \Rightarrow x \in \overline{G(\varnothing)} \wedge x \in G(q))$

$\equiv$

$\quad p \cap \overline{q} \subseteq \mathsf{grd}(G) \cap G(q)$

$\Rightarrow$ $\qquad$ { Weakening }

$\quad WF1$

□

3.1 Termination of Fair Iteration

The general strategy in the proof of a basic liveness properties $P \gg_w Q$ is to divide the events of S into two groups: one for the events that establish Q and another one for the events that maintain P or establish Q. The first group is characterized by event G, and the second one by an event F, where $F = [\!]_{i \in L-K}\, F_i$. Events F and G are modeled by conjunctive set transformers of type $\mathbb{P}(u) \to \mathbb{P}(u)$, and the B event system S can be seen as:

$$S \mathrel{\widehat{=}} F \mathbin{[\!]} G \tag{7}$$

As we know, most of the time, an abstract system like S does not terminate. For this reason we cannot speak about the establishment of a certain postcondition Q when S terminates. In [2], this situation is managed by translating the problem of reachability of a certain postcondition Q in a system S to the problem of termination of the iteration $(\neg Q \Longrightarrow S)^{\frown}$. We follow a similar approach, but we consider a fair iteration with the help of the dovetail operator.

Let q be a subset of u and $X(q)$ be the following iteration:

$$X(q) = \overline{q} \Longrightarrow ((F \ ; X(q)) \ \triangledown \ G) \tag{8}$$

Since all events in S always terminate, we conclude that F and G always terminate. Therefore we expect that $X(q)$ eventually terminates when it is executed in any state of $\overline{G(\varnothing)} \cap \overline{q}$. This expectation is ensured with the semantic of the dovetail operator, which guarantees that G will be eventually executed. On the other hand, if $X(q)$ starts its execution in any state of q, the guard of $X(q)$ is not enabled and the state of the system is not changed. This is formally stated in the following lemma:

Lemma 1. (Termination)
Let $X(q)$ be a fair iteration, $X(q) = \overline{q} \Longrightarrow (F \ ; X) \ \triangledown \ G$, where F and G are conjunctive set transformers of type $\mathbb{P}(u) \to \mathbb{P}(u)$, $\mathsf{pre}(F) = u$ and $\mathsf{pre}(G) = u$. Then the inclusion $\mathsf{grd}(G) \cup q \subseteq \mathsf{pre}(X(q))$ holds.

Proof

$$\begin{array}{lr}
\quad \overline{G(\varnothing)} \cup q & \\
= & \{ \ \ G(u) = u \ \ \} \\
\quad \overline{G(\varnothing)} \cap G(u) \cup q & \\
\subseteq & \{ \ \ Z = (F \ ; X(q)) \ \ \} \\
\quad Z(u) \cup \overline{G(\varnothing)} \cap G(u) \cup q & \\
= & \{ \ \ \text{absorption} \ \ \} \\
\quad Z(u) \cup (Z(u) \cap \overline{Z(\varnothing)}) \cup (\overline{G(\varnothing)} \cap G(u)) \cup q & \\
= & \{ \ \ \text{def. dovetail (2), } Z \text{ and } G(u) = u \ \ \} \\
\quad \mathsf{pre}((F \ ; X(q)) \ \triangledown \ G) \cup q & \\
= & \{ \ \ \text{def. termination set} \ \ \} \\
\quad ((F \ ; X(q)) \ \triangledown \ G)(u) \cup q & \\
= & \{ \ \ \text{def. set transformer} \ \ \} \\
\quad (\overline{q} \Longrightarrow ((F \ ; X(q)) \ \triangledown \ G))(u) & \\
= & \{ \ \ \text{def. termination set and (8)} \ \} \\
\quad \mathsf{pre}(X(q)) &
\end{array}$$

□

3.2 Total Correctness of Fair Iteration

From lemma 1, we assert that the fair iteration $X(q)$ always terminates when it is executed in a state where $grd(G)$ holds. Informally, this fact results from the operational meaning of the dovetail operator. As it was indicated in section 2, the two operands in the dovetail operator, $F \ ; X(q)$ and G, are executed fairly in parallel on separate copies of the state, accepting as an outcome any proper, nonlooping outcome of either operand. If F does not preserve $grd(G)$, the sequence $F \ ; X(q)$ may loop forever depending on the guard of F; however in this case the semantic of the dovetail operator guarantees that $X(q)$ terminate because G do so when it is executed in a state of $\mathsf{grd}(G)$. This behavior is not exactly the same as in the B event system S because of the guards: event G cannot be executed in a state where $\neg grd(G)$ holds.

In order to improve our model of fair iteration among events, we add the constraint that F must preserve the guard of G . In this way, the behaviors of S under the weak fairness assumption and $X(q)$ are similar. Furthermore, if G is able to establish q when it starts its execution in a state in $p \cap \overline{q}$, for a certain subset p of u, if $p \cap \overline{q}$ is a subset of $\mathsf{grd}(G)$ and F preserves p or establishes q when it is executed in any state of $p \cap \overline{q}$, then we can assert that $X(q)$ terminates in a state of q when it is executed in any state of $p \cap \overline{q}$. This reasoning is formalized in the following lemma:

Lemma 2. (Total Correctness)
Under assumptions of lemma 1, and for any p and q in $\mathbb{P}(u)$, such that $p \cap \overline{q} \subseteq F(p \cup q)$, $p \cap \overline{q} \subseteq$ $\mathsf{grd}(G)$, and $p \cap \overline{q} \subseteq G(q)$ then $p \cup q \subseteq X(q)(q)$ holds.

Proof
According to the pairing condition (3), the goal of lemma 2 becomes:

$$p \cup q \subseteq \mathcal{L}(X(q))(q) \tag{9}$$
$$p \cup q \subseteq \mathsf{pre}(X(q)) \tag{10}$$

In order to prove subgoal (9), we note the following equality for any r in $\mathbb{P}(u)$:

$$\mathcal{L}(X(q))(r) = \mathcal{F}(q)(r)(\mathcal{L}(X(q))(r)) \tag{11}$$

where $\mathcal{F}(q)(r)$, for any subset q and r of u, is the set transformer:

$$\mathcal{F}(q)(r) = \overline{q} \Longrightarrow (G(r) \mid F) \tag{12}$$

Equality (11) is proved as follows:

$$\begin{array}{ll}
& \mathcal{L}(X(q))(r) \\
= & \qquad \{ \ (8) \ \} \\
& \mathcal{L}(\overline{q} \Longrightarrow ((F \,;\, X(q)) \bigtriangledown G))(r) \\
= & \qquad \{ \ \text{def. Liberal of guard} \ \} \\
& q \cup \mathcal{L}((F \,;\, X(q)) \bigtriangledown G)(r) \\
= & \qquad \{ \ \text{def. of dovetail 2} \ \} \\
& q \cup \mathcal{L}(F \,;\, X(q))(r) \cap \mathcal{L}(G)(r) \\
= & \qquad \{ \ G(u) = u \text{ and property (2)} \ \} \\
& q \cup \mathcal{L}(F \,;\, X(q))(r) \cap G(r) \\
= & \qquad \{ \ \text{def. Liberal of sequencing} \ \} \\
& q \cup \mathcal{L}(F)(\mathcal{L}(X(q))(r)) \cap G(r) \\
= & \qquad \{ \ F(u) = u \text{ and property (2)} \ \} \\
& q \cup F(\mathcal{L}(X(q))(r)) \cap G(r) \\
= & \qquad \{ \ \text{def. preconditioned set transformer} \ \} \\
& q \cup (G(r) \mid F)(\mathcal{L}(X(q))(r)) \\
= & \qquad \{ \ \text{def. guarded set transformer} \ \} \\
& (\overline{q} \Longrightarrow (G(r) \mid F))(\mathcal{L}(X(q))(r)) \\
= & \qquad \{ \ \text{def. of } \mathcal{F}(q)(r) \text{ (12)} \ \} \\
& \mathcal{F}(q)(r)(\mathcal{L}(X(q))(r))
\end{array}$$

We note that $\mathcal{F}(q)(r)$ is a monotonic set transformer, that is for any subset s and t of u, such that $s \subseteq t$ we have:

$$
\begin{array}{ll}
\quad s \subseteq t & \\
\Rightarrow & \{\text{ monotonic } F\ \} \\
\quad F(s) \subseteq F(t) & \\
\Rightarrow & \\
\quad G(r) \cap F(s) \subseteq G(r) \cap F(t) & \\
\Rightarrow & \\
\quad q \cup (G(r) \cap F(s)) \subseteq q \cup (G(r) \cap F(t)) & \\
\Rightarrow & \{\text{ def. set transformer }\} \\
\quad (\overline{q} \Longrightarrow (G(r) \mid F))(s) \subseteq (\overline{q} \Longrightarrow (G(r) \mid F))(t) & \\
= & \{\ (12)\ \} \\
\quad \mathcal{F}(q)(r)(s) \subseteq \mathcal{F}(q)(r)(t) &
\end{array}
$$

Therefore, as indicated in section 2.1, a recursive definition of a liberal set transformer $\mathcal{L}(X(q))(r) = \mathcal{F}(q)(r)(\mathcal{L}(X(q))(r))$, with monotonic $\mathcal{F}(q)(r)$, allow us to state:

$$\mathcal{L}(X(q))(r) = \mathsf{FIX}(\mathcal{F}(q)(r)) \tag{13}$$

Furthermore, we note

$$\mathsf{FIX}(\mathcal{F}(q)(r)) = \bigcup \mathit{\Phi}_r^q \tag{14}$$

where $\mathit{\Phi}_r^q = \{x \mid x \in \mathbb{P}(u) \wedge x \subseteq \mathcal{F}(q)(r)(x)\}$.

Finally, the proof of subgoal 9 is as follows:

1. $p \cap \overline{q} \subseteq G(q) \cap F(p \cup q) \cup q$; From Hyp.
2. $p \cap \overline{q} \subseteq (\overline{q} \Longrightarrow (G(q) \mid F))(p \cup q)$; 1 and set trans.
3. $p \cap q \subseteq G(q) \cap F(p \cup q) \cup q$; Trivial
4. $p \subseteq (\overline{q} \Longrightarrow (G(q) \mid F))(p \cup q)$; 3 and 2
5. $q \subseteq q \cup G(q) \cap F(p \cup q)$; trivial
6. $q \subseteq (\overline{q} \Longrightarrow (G(q) \mid F))(p \cup q)$; 5 and set trans.
7. $p \cup q \in (\overline{q} \Longrightarrow (G(q) \mid F))(p \cup q)$; 6 and 4
8. $p \cup q \in \mathcal{F}(q)(q)(p \cup q)$; 7 and (12)
9. $p \cup q \in \mathit{\Phi}_q^q$; 8 and def. $\mathit{\Phi}_q^q$
10. $p \cup q \subseteq \bigcup \mathit{\Phi}_q^q$; 9
11. $p \cup q \subseteq \mathsf{FIX}(\mathcal{F}(q)(q))$; 10 and (14)
12. $p \cup q \subseteq \mathcal{L}(X(q))(q)$; 11 and (13)

The proof of subgoal 10 which terminates the proof of lemma 2 is:

1. $p \cap \overline{q} \subseteq \overline{G'(\varnothing)}$; Hyp.
2. $p \cap q \subseteq q$; trivial
3. $p \subseteq \overline{G'(\varnothing)} \cup q$; 2 and 1
4. $p \subseteq \mathsf{pre}(X(q))$; 3 and lemma 1
5. $q \subseteq \overline{G'(\varnothing)} \cup q$; trivial
6. $q \subseteq \mathsf{pre}(X(q))$; 5 and lemma 1
7. $p \cup q \subseteq \mathsf{pre}(X(q))$; 6 and 3

□

As we can see, the hypothesis in lemma (2): $p \cap \overline{q} \subseteq (F \;[\!]\; G)(p \cup q)$, $p \cap \overline{q} \subseteq \overline{G(\varnothing)}$ and $p \cap \overline{q} \subseteq G(q)$ are the corresponding proof obligations (5) and (6) for a basic liveness property $G \cdot P \gg_w Q$ of system S. These inclusions, the implicit assumption that all events in S always terminate and the fairness assumption, are the guarantee that iteration of events in S, starting at any state in $P \wedge \neg Q$ will certainly terminate in a state into Q, which is the intended meaning of the basic liveness property.

3.3 Guard of the Fair Loop

As we know, for any monotonic set transformer S, the complement of the guard of S, $S(\varnothing)$, denotes the set of states where the execution of S is impossible. In the other hand, S becomes a miraculous statement when its execution "starts" in any state of $S(\varnothing)$, and it is able to establishes any postcondition q, because S is monotonic and then $S(\varnothing) \subseteq S(q)$ holds for any subset q of $\mathsf{dom}(S)$.

Before we calculate the guard of the fair loop, we prove the following lemma indicating that $X(q)$ is a monotonic set transformer:

Lemma 3. (Monotony of the Fair Loop)
For any subset s and t of u, such that $s \subseteq t$ we have $X(q)(s) \subseteq X(q)(t)$

Proof

$$
\begin{array}{ll}
\quad s \subseteq t & \\
\Rightarrow & \{\ \text{Monotony of } G\ \} \\
\quad G(s) \subseteq G(t) & \\
\Rightarrow & \{\ \text{Fact of sets for any } y \in \mathbb{P}(u)\ \} \\
\quad q \cup (G(s) \cap F(y)) \subseteq q \cup (G(t) \cap F(y)) & \\
\equiv & \{\ \text{def. } \mathcal{F}(q)(s) \text{ and } \mathcal{F}(q)(t)\ \} \\
\quad \mathcal{F}(q)(s)(y) \subseteq \mathcal{F}(q)(t)(y) & \\
\Rightarrow & \{\ \text{Fact of sets for any } y \in \mathbb{P}(u)\ \} \\
\quad \{\, y \mid y \subseteq u \wedge y \subseteq \mathcal{F}(q)(s)(y)\,\} \subseteq \{\, y \mid y \subseteq u \wedge y \subseteq \mathcal{F}(q)(t)(y)\,\} & \\
\Rightarrow & \{\ \text{from (14)}\ \} \\
\quad \mathsf{FIX}(\mathcal{F}(q)(s)) \subseteq \mathsf{FIX}(\mathcal{F}(q)(t)) & \\
\equiv & \{\ \text{from (13)}\ \} \\
\quad \mathcal{L}(X(q))(s) \subseteq \mathcal{L}(X(q))(t) & \\
\Rightarrow & \\
\quad \mathcal{L}(X(q))(s) \cap \mathsf{pre}(X(q)) \subseteq \mathcal{L}(X(q))(t) \cap \mathsf{pre}(X(q)) & \\
\equiv & \{\ \text{Pairing condition (1)}\ \} \\
\quad X(q)(s) \subseteq X(q)(t) & \square
\end{array}
$$

Now we state the following lemma:

Lemma 4. (Guard of Fair Loop)
The guard of $X(q)$ is the complement of the least fixpoint of $\mathcal{F}(q)(\varnothing)$ $(\overline{X(q)(\varnothing)} = \overline{\mathit{fix}(\mathcal{F}(q)(\varnothing))})$

In order to prove lemma 4, we prove

$$X(q)(\varnothing) = \mathsf{fix}(\mathcal{F}(q)(\varnothing))$$

as follows:

Proof

$$\begin{array}{ll}
 & X(q)(\varnothing)\\
= & \hfill \{\ \text{Def. } X(q)\ \}\\
 & q \cup ((F \mathbin{;} X(q)) \triangledown G)(\varnothing)\\
= & \hfill \{\ \text{From (4)}\ \}\\
 & q \cup ((F \mathbin{;} X(q))(\varnothing) \cap G(\varnothing))\\
= & \hfill \{\ \text{Set Transformers}\ \}\\
 & (\overline{q} \Longrightarrow (G(\varnothing) \mid F))(X(q)(\varnothing))\\
= & \hfill \{\ \text{Def. } \mathcal{F}(q)(\varnothing)\ (12)\ \}\\
 & \mathcal{F}(q)(\varnothing)(X(q)(\varnothing))\\
= & \hfill \{\ \mathcal{F}(q)(\varnothing) \text{ is a monotonic function}\ \}\\
 & \mathsf{fix}(\mathcal{F}(q)(\varnothing))
\end{array}$$

□

As the complement of the guard of $X(q)$ is the least fixpoint of $\mathcal{F}(q)(\varnothing)$, we know that $\mathsf{fix}(\mathcal{F}(q)(\varnothing))$ contains all finite chains terminating out of the guard of $\mathcal{F}(q)(\varnothing)$, that is, in the set $q \cup (G(\varnothing) \cap F(\varnothing))$. Formally, this fact is stated as follows:

$$\forall i \cdot (i \in \mathbb{N} \Rightarrow \mathcal{F}(q)(\varnothing)^{i}(\mathcal{F}(q)(\varnothing)(\varnothing)) \subseteq \mathsf{fix}(\mathcal{F}(q)(\varnothing)) \qquad (15)$$

Proof

Let r be any set in $\{z \mid z \subseteq u \wedge \mathcal{F}(q)(\varnothing)(z) \subseteq z\}$. We prove by induction:

$$\forall i \cdot (i \in \mathbb{N} \Rightarrow \mathcal{F}(q)(\varnothing)^{i+1}(\varnothing) \subseteq r) \qquad (16)$$

Base Case:

$$\begin{array}{ll}
 & \mathcal{F}(q)(\varnothing)^{0+1}(\varnothing)\\
= & \\
 & \mathcal{F}(q)(\varnothing)(\varnothing)\\
\subseteq & \hfill \{\ \text{Monotony of } \mathcal{F}(q)(\varnothing)\ \}\\
 & \mathcal{F}(q)(\varnothing)(r)\\
\subseteq & \hfill \{\ \text{Hyp. } \mathcal{F}(q)(\varnothing)(r) \subseteq r\ \}\\
 & r
\end{array}$$

Inductive Step:

$$\begin{array}{ll}
1.\ \mathcal{F}(q)(\varnothing)^{i+1}(\varnothing) \subseteq r & ;\ \text{Ind. Hyp.}\\
2.\ \mathcal{F}(q)(\varnothing)(\mathcal{F}(q)(\varnothing)^{i+1}(\varnothing)) \subseteq \mathcal{F}(q)(\varnothing)(r) & ;\ \text{1, Mon. of } \mathcal{F}(q)(\varnothing)\\
3.\ \mathcal{F}(q)(\varnothing)^{i+2}(\varnothing) \subseteq r & ;\ \text{2 and Hyp.}
\end{array}$$

Now, (15) follows from (16), considering that $\mathsf{fix}(\mathcal{F}(q)(\varnothing))$ is the generalized intersection of all subsets in $\{z \mid z \subseteq u \wedge \mathcal{F}(q)(\varnothing)(z) \subseteq z\}$. □

From monotony of $X(q)$ (lemma 3), the guard of $X(q)$ (lemma 4) and (15), it follows, for any $r \in \mathbb{P}(u)$:

$$\forall i \cdot (i \in \mathbb{N} \Rightarrow \mathcal{F}(q)(\varnothing)^{i+1}(\varnothing) \subseteq X(q)(r))$$

This last inclusion indicates that the set of states that guarantees termination of $X(q)$ in any state of r, contains all states where any iteration of $(\overline{q} \Longrightarrow G(\varnothing) \mid F)$ terminates in $q \cup (G(\varnothing) \cap F(\varnothing))$. Moreover, if $X(q)$ starts execution in any state of $\mathcal{F}(q)(\varnothing)^{i+1}(\varnothing)$ for any $i \in \mathbb{N}$, $X(q)$ becomes a miraculous statement, able to establish any postcondition.

4 General Liveness Properties

In B event system S, with state variable x and invariant X, general liveness properties are specified by formulae $P \leadsto Q$, where P and Q are predicates on the system state. This property specifies that the system eventually reaches a state satisfying Q whenever it reaches any state in P. There are three basic differences between a $\leadsto$ relation and a $\gg_w$ relation. The first difference is the number of steps involved in the transition from P to Q. With $\gg_w$, the helpful transition is done by the execution of an atomic event, while with $\leadsto$, the number of atomic transitions is not specified. The second difference is that we can assert with $G \cdot P \gg_w Q$ that the system maintains P while Q is not established. We do not have this guarantee when we specify $P \leadsto Q$. Finally, the third difference is that a general liveness property does not directly depend on any fairness assumption while a basic liveness property do.

A property $P \leadsto Q$ holds in a B event system if it is derived by a finite number of applications of the rules defined by the UNITY theory:

	ANTECEDENT	CONSEQUENT
BRL	$G \cdot P \gg_w Q$	$P \leadsto Q$
TRA	$P \leadsto R,\ R \leadsto Q$	$P \leadsto Q$
DSJ	$\forall m \cdot (m \in M \Rightarrow P(m) \leadsto Q)$	$\exists m \cdot (m \in M \wedge P(m)) \leadsto Q$

So as to reason about liveness properties, we incorporate the proof system in UNITY in the framework of B event systems. We can use all theorems in [5] concerning *ensures* and *leads to* relations in the rules of proof of several properties of B event systems.

4.1 Refining Liveness Properties

If abstract system S is refined into another one T we need to assert that any abstract property $\mathcal{P}$ is preserved in T. As property $\mathcal{P}$ depends on basic properties $\mathcal{Q}$, we only need to demonstrate that each basic property $\mathcal{Q}$ is preserved in T. We can establish the validity of each property $\mathcal{Q}$ in the refinement T by the proof of WF0 and WF1 proof obligations. However, if we do these proofs, we would repeat the proofs done in the abstraction S because WF0 is completely preserved by refinement and WF1 is partially preserved. So as to reduce the number and complexity of proofs, we propose two new proof obligations that the refinement T must satisfy in order to preserve a basic liveness property. We present these proof obligations for a certain basic liveness property $\mathcal{Q}$.

Let $\mathcal{Q}$ be the property $G \cdot P \gg_w Q$ which holds in abstract system S. From WF0' and WF1' in section 3, we know that S can be considered as an event system $F \parallel G$, where $F = \parallel_{i \in L-K} F_i$, such that $P \wedge \neg Q \Rightarrow [F]\,(P \vee Q)$ and $P \wedge \neg Q \Rightarrow [G]\,Q$ holds under the assumption of I. If S is refined to T, the refinement is considered as an event system $F' \parallel G' \parallel H$, where F' and G' are the refinements of F and G respectively and H are new events that refine *skip* [2]. We consider that the abstract state is refined by a concrete one, and these states are related by the gluing invariant J. Under the assumptions $I \wedge J$, and according to the very definition of refinement, we conclude that $P \wedge \neg Q \Rightarrow [F' \parallel H]\,(P \vee Q)$ and $P \wedge \neg Q \Rightarrow [G']\,Q$ hold in T. However we cannot assert that $P \wedge \neg Q \Rightarrow grd(G')$ holds under the same assumptions, because the refined event G' has a guard stronger than $grd(G)$. Then, in order to guarantee the preservation of $\mathcal{Q}$ we need to prove that T reaches a state in the guard of G' when it is in a state out of the guard (rule LIP: Liveness Preservation), and that the guard of G' is preserved by F' and H (rule SAP: Safety Preservation). Formally the proof obligations in T are:

LIP	$I \wedge J \wedge P \wedge \neg Q \wedge \neg grd(G') \leadsto grd(G')$
SAP	$I \wedge J \wedge P \wedge \neg Q \wedge grd(G') \Rightarrow [F' \mathbin{[\!]} H]\, grd(G')$

We conclude this section by a summary of our approach to the specification and refinement of a general liveness property $\mathcal{P}$. Property $\mathcal{P}$ is proved in the abstract system S by identifying basic liveness properties $\mathcal{Q}$, such that $\mathcal{P}$ is derived from $\mathcal{Q}$ by application of rules given in section 4. Each property $\mathcal{Q}$ is then proved by WF0 and WF1 proof obligations. When system S is refined to system T, each property $\mathcal{Q}$ in S generates new proof obligations $\mathcal{P}'_1$ and $\mathcal{P}'_2$ in T as stated by LIP and SAP rules. In turn, in order to prove each general liveness property $\mathcal{P}'_1$, we need to identify other basic liveness properties $\mathcal{Q}'$. We continue this process at each step of refinement. We observe that properties $\mathcal{Q}$ and $\mathcal{Q}'$ specify an atomic transition at each step of refinement. However, the transition at step $i+1$ of a refinement is "shorter" than the transition at step i. That is, at level i a certain basic property $\mathcal{Q}$ specifies an atomic transition from a state in P to another one in Q. At level $i+1$ we do not need to prove the (concrete) transition from P to Q, we are only concerned with the transition specified in $\mathcal{Q}'$ which is necessary in the proof of the transition from a state in $r^{-1}[p \cap \overline{q}] \cap \overline{\mathsf{grd}(G')}$ to another one in $\mathsf{grd}(G')$, where G' is the refinement of the helpful event related to $\mathcal{Q}$. In this way, our method of specification and proof of liveness properties becomes a guide that serves to specify and prove the dynamic behavior of a system at each level of refinement.

In the next subsection we give a justification of proof obligations LIP and SAP, as sufficient conditions to ensures the preservation of liveness properties under refinement.

4.2 Proving Refinement of Basic Liveness Properties under Weak Fairness

When abstract system S (7) is refined, the abstract events F and G are refined by concrete events F' and G' respectively and new events H appear. In this way, the abstract system S is refined by the system S':

$$S' \mathrel{\widehat{=}} F' \mathbin{[\!]} G' \mathbin{[\!]} H \tag{17}$$

Let y be the concrete state variable of S' and v the concrete state space, where $v = \{y \mid \exists x \cdot (I(x) \wedge J(x,y))\}$, I is the abstract invariant of S and J the gluing invariant of T. The events F', G' and H are modeled by conjunctive set transformers of type $\mathbb{P}(v) \to \mathbb{P}(v)$. The abstract and concrete events are related by the refinement relation: $F \sqsubseteq F'$ and $G \sqsubseteq G'$ and new events refine *skip*: $\mathit{skip} \sqsubseteq H$. These relations among events are defined by the following proof obligations [1]:

$$F(\overline{r[\overline{s}]}) \subseteq \overline{r[\overline{F'(s)}]} \tag{18}$$

$$G(\overline{r[\overline{s}]}) \subseteq \overline{r[\overline{G'(s)}]} \tag{19}$$

$$\mathit{skip}(\overline{r[\overline{s}]}) \subseteq \overline{r[\overline{H(s)}]} \tag{20}$$

where s is universally quantified over $\mathbb{P}(v)$ and r is a total relation from v to u defined as follows $r = \{y \mapsto x \mid I(x) \wedge J(x,y)\}$.

From conditions stated in lemma 2, we know that abstract system S eventually reaches a state in q when its execution arrives at any state of p. In order to preserve this abstract transition, we need to observe a concrete transition from a state in p' to another one in q', where p' and q' are the corresponding concrete states $r^{-1}[p]$ and $r^{-1}[q]$ respectively. In the following paragraphs we analyze sufficient conditions for this concrete transition.

We consider the fair iteration $X'(q')$ made up of events in S':

$$X'(q') = \overline{q'} \Longrightarrow (((F' \mathbin{[\!]} H) \mathbin{;} X'(q')) \triangledown G') \tag{21}$$

This recursion models the iteration of events F' and H in the concrete system. Now we state the following lemma:

Lemma 5. (Partial Correctness)
Under the assumptions of lemma 2 and refinement conditions (18), (19) and (20), the inclusion $p' \cup q' \subseteq \mathcal{L}(X'(q'))(q')$ holds.

Proof
A brief outline of the proof is as follows. From assumptions of lemma 2: $F(u) = u$, $G(u) = u$, $p \cap \overline{q} \subseteq F(p \cup q)$ and $p \cap \overline{q} \subseteq G(q)$ and refinement conditions (18), (19) and (20), the following inclusions follow:

$$\begin{aligned}
&F'(v) = v && (22) \qquad && r^{-1}[p \cap \overline{q}] \subseteq F'(p' \cup q') && (25)\\
&G'(v) = v && (23) \qquad && r^{-1}[p \cap \overline{q}] \subseteq G'(q') && (26)\\
&H(v) = v && (24) \qquad && r^{-1}[p \cap \overline{q}] \subseteq H(p' \cup q') && (27)
\end{aligned}$$

We prove (22) and (25). The other proofs are done in a similar way. First, we prove the following inclusion for any s in $\mathbb{P}(u)$:

$$r^{-1}[F(s)] \subseteq F'(r^{-1}[s]) \tag{28}$$

The proof of this inclusion is based on equivalence $r[a] \subseteq b \equiv r^{-1}[\overline{b}] \subseteq \overline{a}$, where a and b are universally quantified over $\mathbb{P}(v)$ and $\mathbb{P}(u)$ respectively. The reference to this equivalence in the proof is given as "(Equ)". The proof of (28) is:

1. $F(\overline{r[\overline{r^{-1}[s]}]}) \subseteq \overline{r[\overline{F'(r^{-1}[s])}]}$; From (18)
2. $r[\overline{F'(r^{-1}[s])}] \subseteq \overline{F(\overline{r[\overline{r^{-1}[s]}]})}$; 1
3. $r^{-1}[F(\overline{r[\overline{r^{-1}[s]}]})] \subseteq F'(r^{-1}[s])$; 2 and Equ
4. $r^{-1}[s] \subseteq r^{-1}[s]$; trivial
5. $r[\overline{r^{-1}[s]}] \subseteq \overline{s}$; 4 and Equ
6. $s \subseteq \overline{r[\overline{r^{-1}[s]}]}$; 5
7. $r^{-1}[F(s)] \subseteq r^{-1}[F(\overline{r[\overline{r^{-1}[s]}]})]$; 6 and monotony
8. $r^{-1}[F(s)] \subseteq F'(r^{-1}[s])$; 7 and 3

The proof of (22) is as follows:

1. $F'(v) \subseteq v$; $F \in \mathbb{P}(v) \to \mathbb{P}(v)$
2. $r^{-1}[F(u)] \subseteq F'(r^{-1}[u])$; (28) and $s = u$
3. $r^{-1}[u] \subseteq F'(r^{-1}[u])$; 2 and $F(u) = u$
4. $v \subseteq F'(v)$; r total: $r^{-1}[u]{=}v$
5. $F'(v) = v$; 4 and 1

The proof of (25) is as follows:

1. $r^{-1}[p \cap \overline{q}] \subseteq r^{-1}[F(p \cup q)]$; From Hyp.
2. $r^{-1}[F(p \cup q)] \subseteq F'(r^{-1}[p \cup q])$; (28) and $s = p \cup q$
3. $r^{-1}[p \cap \overline{q}] \subseteq F'(p' \cup q')$; 2, 1, def. p' and q'

With inclusions (22)–(27) we make a calculus similar to the proof of subgoal $p \cup q \subseteq \mathcal{L}(X(q))(q)$ of lemma 2. That is, we derive the equality $\mathcal{L}(X'(q'))(q') = \mathsf{FIX}(\overline{q} \Longrightarrow (G'(q) \mid (F' \mathbin{[\!]} H)))$ in a way similar to calculation of (13). Then, using the equality $\mathsf{FIX}(\overline{q} \Longrightarrow (G'(q') \mid (F' \mathbin{[\!]} H))) = \bigcup \Phi'$, where $\Phi' = \{x \mid x \in \mathbb{P}(v) \wedge x \subseteq (\overline{q} \Longrightarrow (G'(q') \mid (F' \mathbin{[\!]} H)))(x)\}$ we conclude $p' \cup q' \in \Phi'$ from refinement conditions. Finally, from $p' \cup q' \in \Phi$ and the last two equalities we conclude the goal of lemma (5): $p' \cup q' \subseteq \mathcal{L}(X'(q'))(q')$.

□

The inclusion $q' \cup \mathsf{grd}(G') \subseteq \mathsf{pre}(X'(q'))$ follows from a calculus similar to the proof of lemma 1. As we know in S, $p \cap \overline{q}$ is included in the guard of G. Unfortunately this inclusion is not preserved by refinement, because the guard of G' is stronger than the guard of G ($\mathsf{grd}(G') \subseteq r^{-1}[\mathsf{grd}(G)]$). Therefore the set $r^{-1}[p \cap \overline{q}]$ is not included in the termination set of $X'(q')$. From lemma 5, inclusion $\mathsf{grd}(G') \subseteq \mathsf{pre}(X'(q'))$ and pairing condition, we conclude $p' \cap \mathsf{grd}(G') \subseteq X'(q')(q')$. Furthermore, if the guard of G' is preserved by F' and H, we can assert that concrete system S' has a transition to a state into q' whenever it arrives at any state into $p' \cap \mathsf{grd}(G')$. This is formally stated in the following lemma:

Lemma 6. *Under the assumptions of lemma 2 and refinement conditions (18), (19) and (20) as well as the following condition:*

$$r^{-1}[p \cap \overline{q}] \cap grd(G') \subseteq (F' \mathbin{[\!]} H)(grd(G')) \tag{29}$$

the property $G' \cdot y \in p' \wedge grd(G') \gg_w y \in q'$ holds in S'.

Proof

We apply WF0 and WF1 proof obligations in order to prove this lemma. First, we prove the following inclusion:

$$p' \cap \overline{q'} \subseteq r^{-1}[p \cap \overline{q}] \tag{30}$$

We take $y \in p' \cap \overline{q'}$ as premise and we prove $y \in r^{-1}[p \cap \overline{q}]$:

1. $y \in p' \cap \overline{q'}$; premise
2. $y \in p' \wedge \neg(y \in q')$; 1
3. $y \in r^{-1}[p] \wedge \neg(y \in r^{-1}[q])$; 2 and def p' and q'
4. $\exists x \cdot (x \in p \wedge x \mapsto y \in r^{-1}) \wedge \neg(\exists x \cdot (x \in q \wedge x \mapsto y \in r^{-1}))$; 3
5. $\exists x \cdot (x \in p \wedge x \mapsto y \in r^{-1}) \wedge \forall x \cdot (x \mapsto y \in r^{-1} \Rightarrow x \notin q)$; 4
6. $\exists x \cdot (x \in p \wedge x \notin q \wedge x \mapsto y \in r^{-1})$; 5
7. $y \in r^{-1}[p \cap \overline{q}]$; 6

Proof of WF0: $y \in p' \wedge grd(G') \wedge y \notin q' \Rightarrow [F' \mathbin{[\!]} H]\,(y \in p' \wedge grd(G') \vee y \in q')$:

1. $r^{-1}[p \cap \overline{q}] \subseteq F'(p' \cup q')$; (25)
2. $r^{-1}[p \cap \overline{q}] \subseteq H(p' \cup q')$; (27)
3. $r^{-1}[p \cap \overline{q}] \cap \mathsf{grd}(G') \subseteq (F' \mathbin{[\!]} H)(\mathsf{grd}(G'))$; (29)
4. $r^{-1}[p \cap \overline{q}] \cap \mathsf{grd}(G') \subseteq (F' \mathbin{[\!]} H)(p' \cap \mathsf{grd}(G') \cup q')$; 3, 2 and 1
5. $p' \cap \overline{q'} \subseteq r^{-1}[p \cap \overline{q}]$; (30)
6. $p' \cap \mathsf{grd}(G') \cap \overline{q'} \subseteq (F' \mathbin{[\!]} H)(p' \cap \mathsf{grd}(G') \cup q')$; 5 and 4
7. $y \in p' \wedge grd(G') \wedge y \notin q' \Rightarrow [F' \mathbin{[\!]} H]\,(y \in p' \wedge grd(G') \vee y \in q')$; 6, set trans.

Proof of WF1: $y \in p' \wedge grd(G') \wedge y \notin q' \Rightarrow grd(G') \wedge [G']\, y \in q'$:

1. $r^{-1}[p \cap \overline{q}] \subseteq G'(q')$; (26)
2. $p' \cap \overline{q'} \subseteq r^{-1}[p \cap \overline{q}]$; (30)
3. $p' \cap \overline{q'} \subseteq G'(q')$; 2 and 1
4. $p' \cap \mathsf{grd}(G') \cap \overline{q'} \subseteq \mathsf{grd}(G') \cap G'(q')$; 3
5. $y \in p' \wedge grd(G') \wedge y \notin q' \Rightarrow grd(G') \wedge [G]\, y \in q'$; 4

□

In fact, a calculus of the termination set of $X'(q')$, using the definition of dovetail operator (2), allows us to conclude $\mathsf{pre}(X'(q')) = \mathsf{fix}(\overline{q'} \cap \overline{\mathsf{grd}(G')} \Longrightarrow (F' \mathbin{[\!]} H))$. According to [1], $\mathsf{pre}(X'(q'))$ is the same set as the termination set of $(\overline{q'} \cap \overline{\mathsf{grd}(G')} \Longrightarrow (F' \mathbin{[\!]} H))^{\frown}$. From the equality between the two sets, we conclude that the iteration of F' and H will stop when S' arrives into a state in $\mathsf{grd}(G') \cup q'$. This reasoning allow us to propose the following lemma:

Lemma 7. *Under the assumptions of lemma 6 and the condition:*

$$y \in r^{-1}[p \cap \overline{q}] \wedge \neg grd(G') \leadsto grd(G') \tag{31}$$

the property $y \in p' \leadsto y \in q'$ holds in S'

Proof

The proof of lemma (7) requires the proof of the following property:

$$y \in p' \cap \overline{q'} \wedge \neg grd(G') \text{ UNLESS } y \in p' \cap \overline{q'} \wedge grd(G') \vee y \in q' \tag{32}$$

Let *lhs* be the left hand side of the *unless* property and *rhs* its right hand side. The *unless* property follows from $lhs \wedge \neg rhs \Rightarrow [S']\,(lhs \vee rhs)$. In this case, the property follows from the following implication:

$$y \in p' \cap \overline{q'} \wedge \neg grd(G') \Rightarrow [S']\,(y \in p' \cap \overline{q'} \vee y \in q')$$

1. $r^{-1}[p \cap \overline{q}] \subseteq (F' \mathbin{[\!]} H)(p' \cup q')$; (25) and (27)
2. $r^{-1}[p \cap \overline{q}] \subseteq (F' \mathbin{[\!]} H)(p' \cap \overline{q'} \cup q')$; 1 and absorption
3. $p' \cap \overline{q'} \subseteq r^{-1}[p \cap \overline{q}]$; (30)
4. $p' \cap \overline{q'} \subseteq (F' \mathbin{[\!]} H)(p' \cap \overline{q'} \cup q')$; 3 and 2
5. $p' \cap \overline{q'} \cap \overline{\mathsf{grd}(G')} \subseteq G'(p' \cap \overline{q'} \cup q')$; def. $grd(G')$
6. $p' \cap \overline{q'} \cap \overline{\mathsf{grd}(G')} \subseteq S'(p' \cap \overline{q'} \cup q')$; 5 and 4
7. $y \in p' \cap \overline{q'} \wedge \neg grd(G') \Rightarrow [S']\,(y \in p' \cap \overline{q'} \vee y \in q')$; 6

The proof uses the PSP theorem:

$$\frac{\mathcal{P} \leadsto \mathcal{Q} \ \ , \mathcal{R} \text{ UNLESS } \mathcal{S}}{\mathcal{P} \wedge \mathcal{R} \leadsto \mathcal{Q} \wedge \mathcal{R} \vee \mathcal{S}}$$

and the cancellation (CAN) theorem:

$$\frac{P \leadsto Q \vee R, R \leadsto R'}{P \leadsto Q \vee R'}$$

1. $y \in r^{-1}[p \cap \overline{q}] \wedge \neg grd(G') \leadsto grd(G')$; (31)
2. $p' \cap \overline{q'} \subseteq r^{-1}[p \cap \overline{q}]$; (30)
3. $p' \cap \overline{q'} \cap \overline{\mathsf{grd}(G')} \subseteq r^{-1}[p \cap \overline{q}] \cap \overline{\mathsf{grd}(G')}$; 2
4. $y \in p' \cap \overline{q'} \wedge \neg grd(G') \Rightarrow y \in r^{-1}[p \cap \overline{q}] \wedge \neg(grd(G'))$; 3
5. $y \in p' \cap \overline{q'} \wedge \neg grd(G') \leadsto y \in r^{-1}[p \cap \overline{q}] \wedge \neg(grd(G'))$; 4
6. $y \in p' \cap \overline{q'} \wedge \neg grd(G') \leadsto grd(G')$; TRA 5, 1
7. $y \in p' \cap \overline{q'} \wedge \neg grd(G') \leadsto y \in p' \cap \overline{q'} \wedge grd(G') \vee y \in q'$; 6, PSP and (32)
8. $y \in p' \cap \overline{q'} \wedge \neg grd(G') \leadsto y \in p' \wedge grd(G') \vee y \in q'$; 7
9. $y \in p' \wedge grd(G') \leadsto y \in q'$; lemma 6 and BRL
10. $y \in p' \cap \overline{q'} \wedge \neg grd(G') \leadsto y \in q$; CAN 9 and 8
11. $y \in p' \cap \overline{q'} \wedge grd(G') \Rightarrow y \in p \wedge grd(G')$; trivial
12. $y \in p' \cap \overline{q'} \wedge grd(G') \leadsto y \in q'$; 11, BRL 9, TRA
13. $y \in p' \cap \overline{q'} \leadsto y \in q'$; DSJ 12 and 10
14. $y \in p' \cap q' \Rightarrow y \in q'$; trivial
15. $y \in p' \leadsto y \in q'$; 14,BRL, DSJ 13

□

Our last step in our proofs is to demonstrate that premises (29) and (31) of theorems 6 and 7 are equivalent to SAP and LIP proof obligations.

As we can see, the premises (29) and (31) of theorems 6 and 7 are the set theoretical counterpart of SAP and LIP proof obligations which are needed to guarantee the preservation of basic liveness properties in a refinement.

In order to prove the equivalence between (29) and SAP rule, we demonstrate the following equivalences:

$$y \in r^{-1}[p \cap \overline{q}] \equiv \exists x \cdot (P(x) \wedge \neg Q(x) \wedge I(x) \wedge J(y,x)) \tag{33}$$

$$y \in \mathsf{grd}(G') \equiv y \in v \wedge grd(G') \tag{34}$$

$$y \in (F' \mathbin{[\!]} H)(\overline{G'(\varnothing)}) \equiv y \in v \wedge [F' \mathbin{[\!]} H]\, grd(G') \tag{35}$$

Proof of (33)

$$\begin{array}{ll}
\quad y \in r^{-1}[p \cap \overline{q}] & \\
\equiv & \\
\quad \exists x \cdot (x \in p \cap \overline{q} \wedge y \mapsto x \in r) & \\
\equiv & \{ \text{ def. } p \text{ and } q \ \} \\
\quad \exists x \cdot (P(x) \wedge \neg Q(x) \wedge x \in u \wedge y \mapsto x \in r) & \\
\equiv & \{ \text{ def. } u \text{ and } r \ \} \\
\quad \exists x \cdot (P(x) \wedge \neg Q(x) \wedge I(x) \wedge J(y,x)) &
\end{array}$$

□

Proof of (34)

$$
\begin{aligned}
& y \in \mathsf{grd}(G') \\
\equiv & \qquad\qquad \{\ \text{def. } \mathsf{grd}(G')\ \} \\
& y \in \overline{G'(\varnothing)} \\
\equiv & \qquad\qquad \{\ \text{def. } G'(\varnothing)\ \} \\
& y \in \overline{\{\, z \mid z \in v \wedge [G']\, z \in \varnothing \,\}} \\
\equiv & \qquad\qquad \{\ \text{set theory}\ \} \\
& y \in \{\, z \mid z \in v \wedge \neg[G']\, z \in \varnothing \,\} \\
\equiv & \qquad\qquad \{\ \text{set theory}\ \} \\
& y \in \{\, z \mid z \in v \wedge \neg[G']\, \mathit{false} \,\} \\
\equiv & \qquad\qquad \{\ \text{set theory}\ \} \\
& y \in \{\, z \mid z \in v \wedge \mathit{grd}(G') \,\} \\
\equiv & \\
& y \in v \wedge \mathit{grd}(G')
\end{aligned}
$$

□

Proof of (35)

$$
\begin{aligned}
& y \in (F' \,[\!]\, H)(\overline{G'(\varnothing)}) \\
\equiv & \qquad\qquad \{\ \text{def. set transformer}\ \} \\
& y \in \{\, z \mid z \in v \wedge [F' \,[\!]\, H]\, z \in \overline{G'(\varnothing)} \,\} \\
\equiv & \\
& y \in \{\, z \mid z \in v \wedge [F' \,[\!]\, H]\, (z \in v \wedge \mathit{grd}(G')) \,\} \\
\equiv & \qquad\qquad \{\ [F' \,[\!]\, H]\, z \in v \equiv z \in v\ \} \\
& y \in \{\, z \mid z \in v \wedge [F' \,[\!]\, H]\, \mathit{grd}(G') \,\} \\
\equiv & \qquad\qquad \{\ \text{set theory}\ \} \\
& y \in v \wedge [F' \,[\!]\, H]\, \mathit{grd}(G')
\end{aligned}
$$

□

The equivalence between (29) and SAP rule is as follows:

Proof

$$
\begin{aligned}
& r^{-1}[p \cap \overline{q}] \cap \mathsf{grd}(G') \subseteq (F' \,[\!]\, H)(\mathsf{grd}(G')) \\
\equiv & \\
& \forall y \cdot (y \in r^{-1}[p \cap \overline{q}] \cap \mathsf{grd}(G') \Rightarrow y \in (F' \,[\!]\, H)(\mathsf{grd}(G'))) \\
\equiv & \\
& \forall y \cdot (y \in r^{-1}[p \cap \overline{q}] \wedge y \in \mathsf{grd}(G') \Rightarrow y \in (F' \,[\!]\, H)(\mathsf{grd}(G'))) \\
\equiv & \qquad\qquad \{\ (33),\ (34) \text{ and } (35)\ \} \\
& \forall y \cdot (\exists x \cdot (P(x) \wedge \neg Q(x) \wedge I(x) \wedge J(y,x)) \wedge y \in v \wedge \mathit{grd}(G') \Rightarrow y \in v \wedge [F' \,[\!]\, H]\, \mathit{grd}(G')) \\
\equiv & \\
& \forall y \cdot (\exists x \cdot (P(x) \wedge \neg Q(x) \wedge I(x) \wedge J(y,x)) \wedge y \in v \wedge \mathit{grd}(G') \Rightarrow [F' \,[\!]\, H]\, \mathit{grd}(G')) \\
\equiv & \qquad\qquad \{\ \exists x \cdot (P(x) \wedge I(x) \wedge J(y,x)) \Rightarrow y \in v\ \} \\
& \forall y \cdot (\exists x \cdot (P(x) \wedge \neg Q(x) \wedge I(x) \wedge J(y,x)) \wedge \mathit{grd}(G') \Rightarrow [F' \,[\!]\, H]\, \mathit{grd}(G')) \\
\equiv & \\
& \forall (x,y) \cdot (P(x) \wedge \neg Q(x) \wedge I(x) \wedge J(y,x) \wedge \mathit{grd}(G') \Rightarrow [F' \,[\!]\, H]\, \mathit{grd}(G'))
\end{aligned}
$$

□

In order to prove the equivalence between (31) and LIP proof obligation, we need the following theorem about *leads to*

$$\frac{(\exists x \cdot (P(x)) \wedge Q) \leadsto R\ ,\ x \backslash Q\ ,\ x \backslash R}{(P(x) \wedge Q) \leadsto R} \tag{36}$$

Proof

1. $(\exists x \cdot (P(x)) \wedge Q) \leadsto R$; premise
2. $(\exists y \cdot (P(y)) \wedge Q) \leadsto R$; 1
3. $P(x) \Rightarrow \exists y \cdot (P(y))$; for any x
4. $P(x) \wedge Q \Rightarrow (\exists y \cdot (P(y)) \wedge Q)$; 3
5. $(P(x) \wedge Q) \leadsto (\exists y \cdot (P(y)) \wedge Q)$; 4 and BRL
6. $(P(x) \wedge Q) \leadsto R$; TRA 5 and 2

□

Now, the equivalence between (31) and LIP proof obligation is as follows:

$$\begin{array}{ll} & y \in r^{-1}[p \cap \overline{q}] \wedge \neg grd(G') \leadsto grd(G') \\ \equiv & \hfill \{\ \text{using (33)}\ \} \\ & \exists x \cdot (P(x) \wedge \neg Q(x) \wedge I(x) \wedge J(y,x)) \wedge \neg grd(G') \leadsto grd(G') \\ \equiv & \hfill \{\ x \backslash grd(G'),\ \text{(36) and DSJ}\ \} \\ & P(x) \wedge \neg Q(x) \wedge I(x) \wedge J(y,x) \wedge \neg grd(G') \leadsto grd(G') \end{array}$$

□

5 Conclusions

In this report we present a formal model of fair iteration of events in a B event system. Moreover we use the model to justify our proof obligations for basic liveness properties and preservation under refinement of general liveness properties. The model of fair iteration of events uses the dovetail operator, an operator proposed in [4] to model fair choice. Our proofs are mainly founded in fixpoint calculations of fair iteration of events and weakest precondition calculus.

Our approach to justify our proof obligations was inspired by [2]. The approach, founded in fixpoint calculations and weakest precondition calculus, to justify proof obligations about liveness properties is not classical. It is common to justify proof obligations of this kind of properties by operational reasoning about state traces in the system [3], and the justifications are not so formal as expected. The approach taken in this report allows us to make axiomatic proofs and verify it with the prover of atelier B.

As a future work, we investigate the relationship between general liveness properties and the iteration of events under weak fairness or minimal progress assumptions. We are mainly interested in sufficient conditions to guarantee preservation of liveness properties when a system with weak fairness assumptions is refined in a system with minimal progress assumptions.

References

1. J.-R. Abrial. *The B-Book, Assigning Programs to Meanings.* Cambridge University Press, 1996.
2. J.-R. Abrial and L. Mussat. Introducing Dynamic Constraints in B. In *B'98: Recent Advances in the Development and Use of the B Method, LNCS 1393*, pages 83–128. Springer-Verlag, april 1998.

3. Ralph J.R. Back and Qiwen Xu. Refinement of Fair Action Systems. *Acta Informatica*, 35:131–165, 1998.
4. Manfred Broy and Greg Nelson. Adding Fair Choice to Dijkstra's Calculus. *ACM Transactions on Programming Languages and Systems*, 16(3):924–938, May 1994.
5. K. Mani Chandy and Jayadev Misra. *Parallel Program Design A Foundation.* Addison-Wesley, 1988.
6. Edsger W. Dijkstra and Carel S. Scholten. *Predicate Calculus and Program Semantics.* Springer-Verlag, 1990.
7. Steve Dune. Introducing Backward Refinement into B . In *ZB 2003: Formal Specification and Development in Z an B, LNCS 2651*, pages 178–196. Springer-Verlag, June 2003.
8. Eric C.R. Hehner. do Considere od: A Contribution to the Programming Calculus. *Acta Informatica*, 11:287–304, 1979.
9. Hector Ruíz-Barradas and Didier Bert. Propriétés dynamiques avec hypothèses d'équité en B événementiel. In *Approches Formelles dans l'Assitance au Développement de Logiciels, AFADL'2004*, pages 299–313. Besançon, France, june 2004.