Chapter 19

Flight Phases and System Modes

Flights are divided into phases, in which the state of the system is differentiated from that of other phases. The phases distinguished by, for example, the Boeing statistics [Gro96] are

- Load, taxi, unload
- Takeoff
- Initial climb (with flaps)
- Climb (after flaps retracted)
- Cruise
- Descent
- Initial approach (between initial approach fix and outer marker)
- Final approach (after outer marker)
- Landing

The use (and abuse) of machinery modes, in particular autopilot modes, as well as the so-called *mode confusion* syndrome they engender, was studied in [Deg96]. The significance of modes in WBA lies in the system architecture they entail and the special way they give priority in certain circumstances to particular responses to problems.

A mode has special entry and exit conditions. Furthermore, when this mode is under human control, it is provided with particular mechanisms for entering and exiting. Although in the array of flight phases, all phases may in principle be aborted by the crew or ATC, in practice the phases intuitively most susceptible to abort are Takeoff and Final approach.

Modes and phases are formally very similar; they have entry and exit conditions and actions, and being in a particular mode or phase is part of the overall system state. We therefore do not distinguish them, and treat them as one type of feature. We include a TLA module to describe an arbitrary mode. The mode has entry and exit actions, as well as an explicit action to remain within the mode, even though logically this is not necessary in TLA. The significance for complex systems with human parts is the axiom *Decision* (Figure 19.1). which states that when an agent has reasons both for remaining within a mode and exiting from it, that implies an obligation to decide explicitly to remain within the mode or to exit from it.

This method of handling modes diverges from the concerns of PARDIA, which aims to build a simple classificatory scheme for agency in complex systems. The mode scheme rather aims to dictate what shall be done when a system of norms changes, either for normal reasons or because there is a conflict and a resolution is required. An explicit decision of a certain sort shall be made. This is behaviorally normative rather than classificatory. It plays a crucial role in our incident explanation; when the pilots perceive that the layout of the airport is different from that at Frankfurt and finally realise they're aiming to land at the wrong airport, the informal commentaries say that they chose to continue with the landing. Had the commentaries said nothing, and had the pilots never remarked the difference, the flight would have proceeded substantially as it did proceed, with no changes other than in the mental images of the pilots. We may take it that their realisation and execution of the appropriate action (namely, to change nothing) is mentioned to portray explicit conformance to a norm: this norm is what we aim to represent in the module *Mode* in Figure 19.1.

In application of this module to the example, we shall designate as the *landing* mode that phase of flight which starts with the crew accepting a clearance to land and landing. This in general encompasses the three phases *Initial approach*, *Final approach*, and *Landing*, although depending on the traffic levels, one may be given (and accept) clearance to land also during the Final approach phase. This is immaterial to our current example, although one can foresee circumstances under which this may make a difference.

We instantiate the *Modes* module inside a module called *Phases*, with one instantiation per phase. *Phases* as we present it here contains instances of just three phases: *LandingPhase* that we shall use in the example, and the two Boeing phases *InitialApproachPhase* and *FinalApproachPhase*. Since the narratives for the example do not specify in which of the Boeing phases the aircraft was in when the crew remarked that the situation was untoward, we do not restrict *InitialApproachPhase* and *FinalApproachPhase* with the *InMode* predicate *(AC)in_landing_phase* of *LandingPhase*. The point of *Phases* is to collect all the actual instantiations used for an example. The *Mode* module is a general part of WBA, whereas *Phases* is intended to be broader, with some general parts (for example, all phases from the Boeing document above) as well as some instantiations particular to the example (for autopilot modes, for example). In general, a *Phases* module will be defined for each individual example, instantiat-

module *Mode*

DECLARATIONS

CONSTANT X

VARIABLES $\mathit{InMode}, \mathit{OutOfMode}$

Definitions

$$\mathit{InMode} \triangleq \neg \mathit{OutOfMode}$$

Actions

$$\mathit{EnterMode} \triangleq \mathit{OutOfMode} \wedge \mathit{InMode}'$$

$$\mathit{ExitMode} \triangleq \mathit{InMode} \wedge \mathit{OutOfMode}'$$

$$\mathit{RemainInMode} \triangleq \mathit{InMode} \wedge \mathit{InMode}'$$

Axiom

$$\begin{aligned} \mathit{Decision} \triangleq\ & \wedge\ \mathit{InMode} \\ & \wedge\ \mathit{Reason}(X, \mathit{InMode}') \\ & \wedge\ \mathit{Reason}(X, \mathit{OutOfMode}') \\ & \quad \Rightarrow O(\mathit{Decide}(X, \mathit{RemainInMode}) \vee \mathit{Decide}(X, \mathit{ExitMode})) \end{aligned}$$

Figure 19.1: Mode Handling

module *Phases*

DECLARATIONS

CONSTANTS CRW, AC

VARIABLES $(AC)\_in\_landing\_phase$

DEFINITION

$$
\begin{aligned}
&LandingPhase \triangleq \textbf{instance } Modes\\
&\qquad \textbf{with } X \leftarrow CRW, InMode \leftarrow (AC)in\_landing\_phase\\
&InitialApproachPhase \triangleq \textbf{instance } Modes\\
&\qquad \textbf{with } X \leftarrow CRW,\\
&\qquad\qquad InMode \leftarrow \land\ (AC)in\_approach\\
&\qquad\qquad\qquad \land\ (AC)before\_outer\_marker\\
&FinalApproachPhase \triangleq \textbf{instance } Modes\\
&\qquad \textbf{with } X \leftarrow CRW,\\
&\qquad\qquad InMode \leftarrow \land\ (AC)in\_approach\\
&\qquad\qquad\qquad \land\ (AC)\_after\_outer\_marker\\
&Decision \triangleq \land\ InitialApproachPhase.Decision\\
&\qquad \land\ FinalApproachPhase.Decision\\
&\qquad \land\ LandingPhase.Decision
\end{aligned}
$$

Two of the Boeing phases are included as *InitialApproachPhase* and *FinalApproachPhase*. Although these two are disjoint and adjacent, the phase *LandingPhase* may overlap with them. When a decision is required, it is likely to be required for all phases the aircraft is currently in.

Figure 19.2: Phases Module with Three Phases

ing those modes and phases necessary for the analysis, and containing large parts in common with other examples, corresponding to similar flight phases and similar autopilots. Various parts of *Phases* play crucial roles in rules of inference, for example in Rule 19.1 below. Thus *Phases* itself plays the role of a distinguished module, and its name that of a keyword, whose shape is assumed to be correct by EL. Although *Phases* will change from example to example, corresponding to different categorisations of phases of flight, and different autopilots with different modes, its form must remain identical to that in Figure 19.2, else the Rule 19.1 will become nonsense. That means, in essence, that *Phases* is a logical constant of the appropriate sort, as also its name, and that of its definitions.

LandingPhase is instantiated in *Phases* in Figure 19.2 with $(AC)in\_landing\_phase$ as *InMode*, and thus $\neg(AC)in\_landing\_phase$ as *OutOfMode*. It follows that the only actions (or nonaction) in *Landing_Spec* that can effect these changes are *Accept_Landing* for and *CRW_breakoff*. This entails that the sentences

$$\begin{pmatrix} \wedge & Landing\_Phase.EnterMode \\ \wedge & Landing\_Specs.Spec \end{pmatrix} \Rightarrow Accept\_Landing(ATC, CRW, APT)$$

and

$$\begin{pmatrix} \wedge & Landing\_Phase.ExitMode \\ \wedge & Landing\_Specs.Spec \end{pmatrix} \Rightarrow CRW\_breakoff(CRW)$$

are both provable. Any one of a number of *Landing_Specs*-actions are compatible with remaining in the landing phase, as is to be expected. It is reasonable to interpret the crew's decision '*to continue the landing*' quite simply as *Landing_Phase.RemainInMode*. Were we to pursue detailed reasoning concerning their reasons and intentions, we would be trying to explain the whole business of coming to this conclusion rather more deeply than seems to be needed, and we would be stretching the boundaries between rational reconstruction of a rational process, and psychological explanation, which WBA is not (yet?) equipped to perform.

We may so far demonstrate from the circumstances that

$$O\begin{pmatrix} \vee & Decide(CRW, LandingPhase.RemainInMode) \\ \vee & Decide(CRW, LandingPhase.ExitMode) \end{pmatrix}$$

which is equivalent to

$$O\begin{pmatrix} \vee & Decide\left(CRW, \begin{matrix} \wedge & (AC)\_in\_landing\_phase \\ \wedge & (AC)\_in\_landing\_phase' \end{matrix}\right) \\ \vee & Decide\left(CRW, \begin{matrix} \wedge & (AC)\_in\_landing\_phase \\ \wedge & \neg(AC)\_in\_landing\_phase' \end{matrix}\right) \end{pmatrix}$$

We need to explain

$$Decide\left(CRW, \begin{matrix} \wedge & (AC)\_in\_landing\_phase \\ \wedge & (AC)\_in\_landing\_phase' \end{matrix}\right)$$

The intuitive reason for this decision is that the crew had reasons both for continuing and for breaking off the landing, and a decision was called for, so they made one. This may be formulated more generally:

> The *agent* had good reasons both for and against a specific *course of action*, and a demonstrable obligation to choose one or the other; so chose one.

This constitutes an explanation, in so far as one needs one, of the behavior. Such an explanation is not yet derivable from EL. It rests crucially on the contrasting reasons. One common situation in which one has contrasting reasons is one in which the *Hypotheses* (namely, the actual situation) in concert with the *Procedures* do in fact lead to a contradiction (which could be taken to indicate a weakness in the procedures). Since other EL rules require that *Hypotheses* and *Procedures* are true (i.e., they both occur as hypotheses of the rules), they cannot be contradictory in any application of this rule. (An exception is the rule that translates $\succ$ into $\vdash_{TLA}$.) A rule which allows an inference to be drawn from contradictory *Hypotheses* and *Procedures* cannot be derivable from such rules whose hypotheses require their joint truth.

The EL inferece rule formulated in Section 19.1 enables us to turn this contradiction into reasons for and against a course of action.

19.1 An EL Rule for Required Decisions

We introduce a somewhat complicated EL rule which codifies this procedure. This rule makes essential use of the *Decision* rules in *Phases.* The rule is only intended to be applied when *Phases.Decision* is used, and furthermore when this statement is correct and appropriate. It is not the case, as with normal inference rules, that any sentence may be substituted and validity preserved. Logically speaking, then, EL with the agency rules and axioms is a logic with certain *constants*: modules or statements with distinguished, privileged status. It may also make sense to distinguish other types of statements, for example *Axioms* from *Norms*. *Axioms* would be statements that are universally valid throughout, for example, laws of physics. They cannot be violated; whereas *Norms* may be violated during the course of a behavior (and we would expect their violation to carry causal-explanatory significance). We don't choose formally to distinguish these categories here, except for use of the constant *Phases.Decision* in the rule below, and a requirement that the *Phases* module be appropriate.

$$\begin{array}{l}
\mathit{Hypotheses} \\
\mathit{Hypotheses} \wedge \Box \mathit{Procedures} \succ \left(\begin{array}{ll} \wedge & \mathit{Reason}(X, A) \\ \wedge & \mathit{Reason}(X, \neg A) \end{array} \right) \\
\left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \mathit{Reason}(X, A) \\ \wedge & \mathit{Reason}(X, \neg A) \\ \wedge & \Box \mathit{Phases.Decision} \end{array} \right) \succ O(\mathit{Decide}(X, A) \vee \mathit{Decide}(X, \neg A)) \\
\mathit{Decide}(X, A) \\
\hline
\left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \Box \mathit{Procedures} \\ \wedge & \Box \mathit{Phases.Decision} \\ \wedge & \mathit{Decide}(X, A) \end{array} \right) \Box\!\!\Rightarrow \mathit{Decide}(X, A)
\end{array} \tag{19.1}$$

The presence of *Decide(X,A)* in both the antecedent and consequent of the explanatory connective $\Box\!\!\Rightarrow$ may seem peculiar. But it is appropriate. It enables the rule to remain monotone. We consider what would happen if *Decide(X,A)* were not to appear in the antecedent.

19.2 An Unsatisfactory Rule

Consider the following argument. Suppose the rule were to be formulated without *Decide(X,A)* in the antecedent of the conclusion. Consider a situation in which both *Decide(X,A)* and *Decide(X, ¬A)*. This is obviously a peculiar situation in that contradictory decisions were made. Let us also assume the following rule:

$$\begin{array}{c} X \Box\!\!\Rightarrow A \\ X \Box\!\!\Rightarrow B \\ \hline X \Box\!\!\Rightarrow (A \wedge B) \end{array} \tag{19.2}$$

This is to say that if a collection of facts sufficiently-causally-explains a fact A, and also a fact B, that it so does their conjunction. If we were to have formulated the inference rule without the *Decide* predicate in the antecedent of the conclusion, the following inferences could both be made:

$$
\begin{array}{l}
\mathit{Hypotheses} \\
\mathit{Hypotheses} \wedge \Box \mathit{Procedures} \succ \left(\begin{array}{ll} \wedge & \mathit{Reason}(X, A) \\ \wedge & \mathit{Reason}(X, \neg A) \end{array} \right) \\
\left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \mathit{Reason}(X, A) \\ \wedge & \mathit{Reason}(X, \neg A) \\ \wedge & \Box \mathit{Phases.Decision} \end{array} \right) \succ O(\mathit{Decide}(X, A) \vee \mathit{Decide}(X, \neg A)) \\
\mathit{Decide}(X, A) \\
\hline
\left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \Box \mathit{Procedures} \\ \wedge & \Box \mathit{Phases.Decision} \end{array} \right) \Box\!\!\Rightarrow \mathit{Decide}(X, A)
\end{array}
\tag{19.3}
$$

and

$$
\begin{array}{l}
\mathit{Hypotheses} \\
\mathit{Hypotheses} \wedge \Box \mathit{Procedures} \succ \left(\begin{array}{ll} \wedge & \mathit{Reason}(X, A) \\ \wedge & \mathit{Reason}(X, \neg A) \end{array} \right) \\
\left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \mathit{Reason}(X, A) \\ \wedge & \mathit{Reason}(X, \neg A) \\ \wedge & \Box \mathit{Phases.Decision} \end{array} \right) \succ O(\mathit{Decide}(X, A) \vee \mathit{Decide}(X, \neg A)) \\
\mathit{Decide}(X, \neg A) \\
\hline
\left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \Box \mathit{Procedures} \\ \wedge & \Box \mathit{Phases.Decision} \end{array} \right) \Box\!\!\Rightarrow \mathit{Decide}(X, \neg A)
\end{array}
\tag{19.4}
$$

We can now derive the following rule from Rules 19.3, 19.4 and 19.2:

$$\begin{array}{l} Hypotheses \\ Hypotheses \wedge \Box Procedures \succ \begin{pmatrix} \wedge & Reason(X,A) \\ \wedge & Reason(X,\neg A) \end{pmatrix} \\ \\ \begin{pmatrix} \wedge & Hypotheses \\ \wedge & Reason(X,A) \\ \wedge & Reason(X,\neg A) \\ \wedge & \Box Phases.Decision \end{pmatrix} \succ O(Decide(X,A) \vee Decide(X,\neg A)) \\ \\ Decide(X,A) \\ Decide(X,\neg A) \\ \hline \\ \begin{pmatrix} \wedge & Hypotheses \\ \wedge & \Box Procedures \\ \wedge & \Box Phases.Decision \end{pmatrix} \Box\!\!\Rightarrow (Decide(X,A) \wedge Decide(X,\neg A)) \end{array} \quad (19.5)$$

This would suggest that if two contradictory decisions were made, that the situation, which was sufficient to explain either decision individually, would also have been sufficient to explain a pair of contradictory decisions, and this seems intuitively implausible. It should be sufficient to explain either, but surely more is required to explain why *both* were made. Thus the weakening of Rule 19.1 by omitting *Decide(X,A)* from the antecedent of the conclusion is not appropriate.

19.3 Justification for the Behavioral Rule

Justification for Rule 19.1 is as follows. It must be shown that the relation of antecedent to consequent is causal; also that the antecedent contains an array of causal factors that render it sufficient.

We argue first that it is a causal relation. We must therefore argue, under the hypotheses of the rule, that

$$\begin{array}{ll} \neg(\wedge\ Hypotheses & \Box\!\!\rightarrow \neg Decide(X,A) \\ \quad \wedge\ \Box Procedures & \\ \quad \wedge\ \Box Phases.Decision & \\ \quad \wedge\ Decide(X,A)) & \end{array}$$

This is straightforward. The nearest possible world to that in which the antecedent is true; that is, the antecedent of the conclusion of Rule 19.1 is false, is the world in which the situation and procedures were identical, but instead the *Decide(A, ¬A)* was made. That is, simply the opposite decision. After all, this decision is supposed to be relatively free. Furthermore, we would argue that the world in which *both* decisions *Decide(X,A)* and *Decide(X,* $\neg$*A)* were made is stranger, thus further away, than the world in which just the opposite decision was made. This nearest world is a world in which, then, *Decide(X,A)* was not made; thus one in which $\neg$*Decide(X,A)* is true. QED.

Second, we argue that it provides a sufficient explanation. The *Hypotheses* describe the actual situation as it pertains. From this situation, along with operating procedures, one can obtain reasons for A as well as reasons for $\neg A$. From the fact that one has reasons both for and against A, and one is in a particular flight phase (stated in the Hypotheses), it follows from the decision axiom of *Phases* that one should decide between A and $\neg A$. These facts alone are sufficient to explain one's decision, whether it be A or $\neg A$. (Notice that the rule covers both decisions, since deciding for a negation, $\neg B$, is achieved by taking B to be $\neg A$ and observing that $\neg B \equiv \neg\neg A$.)

19.4 Separating Two Steps

It may be thought that, with transitivity of $\succ$, the two complex hypotheses could be compressed into

$$\left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \Box \mathit{Procedures} \\ \wedge & \Box \mathit{Phases.Decision} \end{array}\right) \succ O(\mathit{Decide}(X, A) \vee \mathit{Decide}(X, \neg A))$$

but this is not to be wished. When the *Hypotheses* and *Procedures* contradict each other, then anything follows in TLA (in general, in any logic incorporating the rule *ex falso quodlibet*), in particular:

$$\left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \Box \mathit{Procedures} \end{array}\right) \succ O(\mathit{Decide}(X, A) \vee \mathit{Decide}(X, \neg A))$$

without the hypothesis $\Box$*Phases.Decision*, and for any A at all. Thus proof structure is lost: if *Hypotheses* contradict *Procedures* it does indeed follow that the crew ought to decide how many stars there are in the night sky, as well as everything else they can think of, if *ex falso quodlibet* holds. But that doesn't explain what they actually need to decide or whether they decided it. It follows from *Hypotheses* and *Phases.Decision* that if they have reasons both for remaining in and exiting their current flight phase (that's included in *Hypotheses*), that they ought to decide that. That they have reasons both for remaining in and exiting their current flight phase would indeed follow using *ex falso quodlibet* in this case. But separating the two conditions ensures that the remaining-or-exiting decision is entailed by the situation of being in whatever flight phase they're in along with other statements, and not because its hypotheses are inconsistent.

19.5 Procedural Conflicts

In certain cases, procedures may conflict with the actual situation. In such a situation, we hypothesise that a decision is required, whether to stay in the current mode or to do something else. This entails that if the actual situation continues to be in conflict with procedure, a continual evaluation of the current mode is required. This reasoning is encapsulated in the following rule

$$\frac{\begin{array}{l} \mathit{Hypotheses} \\ \mathit{Hypotheses} \wedge \Box \mathit{Procedures} \succ \bot \\ \mathit{Hypotheses} \succ \mathit{Phases.Mode.InMode} \end{array}}{\begin{array}{l} O(\vee\ \mathit{Decide}(X, \mathit{Phases.Mode.ExitMode}) \\ \quad \vee\ \mathit{Decide}(X, \mathit{Phases.Mode.RemainInMode})) \end{array}} \tag{19.6}$$

The second hypothesis of this rule, ($\mathit{Hypotheses} \wedge \Box \mathit{Procedures} \succ \bot$), is a formulation of the conflict between actual situation and procedure: they contradict each other. The rule derived directly from Rule 19.1 for the case in which hypotheses and procedures conflict, ($\mathit{Hypotheses} \wedge \Box \mathit{Procedures} \succ \bot$), is:

$$\frac{\begin{array}{l} \mathit{Hypotheses} \\ \mathit{Hypotheses} \wedge \Box \mathit{Procedures} \succ \bot \\ \left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \mathit{Reason}(X, A) \\ \wedge & \mathit{Reason}(X, \neg A) \\ \wedge & \Box \mathit{Phases.Decision} \end{array} \right) \succ O(\mathit{Decide}(X, A) \vee \mathit{Decide}(X, \neg A)) \\ \mathit{Decide}(X, A) \end{array}}{\left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \Box \mathit{Procedures} \\ \wedge & \Box \mathit{Phases.Decision} \\ \wedge & \mathit{Decide}(X, A) \end{array} \right) \Box\!\!\Rightarrow \mathit{Decide}(X, A)} \tag{19.7}$$

One might be tempted by

$$\mathit{Hypotheses} \wedge \Box \mathit{Procedures} \succ \bot$$

along with Rule $\bot_{\mathsf{I}}$, in Chapter 20:

$$\frac{\bot}{A}$$

to conclude in such a case in which hypotheses and procedures contradict that

$$Hypotheses \wedge \Box Procedures \succ \begin{pmatrix} \wedge & Reason(X, A) \\ \wedge & Reason(X, \neg A) \end{pmatrix}$$

and so that Rule 19.7 would follow directly from Rule 19.1. However, this line of reasoning is incorrect, because it normally requires the Deduction Theorem, which in this logic is not a valid rule. The Deduction Theorem says that whenever

$$\frac{A}{B}$$

then in fact

$$\vdash A \Rightarrow B$$

The conclusion we would desire would follow, were the Deduction Theorem to be available, by converting Rule $\perp_{\mathsf{I}}$ into the implication

$$\vdash \perp \Rightarrow \begin{pmatrix} \wedge & Reason(X, A) \\ \wedge & Reason(X, \neg A) \end{pmatrix}$$

and then using the transitivity of $\Rightarrow$. Since the Deduction Theorem does not hold for intensional logics such as TLA or EL, this way is not open to us. Furthermore, it's not clear that Rule 19.7 would allow us the kind of inferences we need to make without addition, since in the circumstances in which the hypotheses and procedures contradict, it would not necessarily be reasonable to suppose the crew would automatically have reason both for and against remaining in mode, and therefore the antecedent of the third hypothesis might not necessarily be true, and so the third hypothesis must obtain its justification through other means.

An alternative way to obtain rules for inconsistent hypotheses and procedures is to consider the case in which, rather than determining if the consequent of the third hypothesis of Rule 19.7 is a logical consequence of a specific antecedent, we were simply to determine if the consequent were true: that is, if the agent X ought to have made one decision or the other, regardless of the reasons why heshe ought to have done:

$$\frac{\begin{array}{l} Hypotheses \\ Hypotheses \wedge \Box Procedures \succ \perp \\ O(Decide(X, A) \vee Decide(X, \neg A)) \\ Decide(X, A) \end{array}}{\begin{pmatrix} \wedge & Hypotheses \\ \wedge & \Box Procedures \\ \wedge & \Box Phases.Decision \\ \wedge & Decide(X, A) \end{pmatrix} \Box\!\!\Rightarrow Decide(X, A)} \tag{19.8}$$

Use of such a rule avoids deciding whether a crew has reasons both for and against remaining in mode. The third hypothesis $O(\mathit{Decide}(X, A) \vee \mathit{Decide}(X, \neg A))$ can in specific cases of interest be proved with the help of Rule 19.6. Coupling the hypotheses of Rule 19.6 together with the hypotheses of Rule 19.8 leads to the following two specific rules – the first for *ExitMode*:

$$\frac{\begin{array}{l} \mathit{Hypotheses} \\ \mathit{Hypotheses} \wedge \Box\mathit{Procedures} \succ \bot \\ \mathit{Hypotheses} \succ \mathit{Phases.Mode.InMode} \\ \mathit{Decide}(X, X, \mathit{Phases.Mode.ExitMode}) \end{array}}{\begin{array}{l} \left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \Box\mathit{Procedures} \\ \wedge & \Box\mathit{Phases.Decision} \\ \wedge & \mathit{Decide}(X, \mathit{Phases.Mode.ExitMode}) \end{array}\right) \\ \Box\!\!\Rightarrow \mathit{Decide}(X, X, \mathit{Phases.Mode.ExitMode}) \end{array}} \tag{19.9}$$

and the second for *RemainInMode*

$$\frac{\begin{array}{l} \mathit{Hypotheses} \\ \mathit{Hypotheses} \wedge \Box\mathit{Procedures} \succ \bot \\ \mathit{Hypotheses} \succ \mathit{Phases.Mode.InMode} \\ \mathit{Decide}(X, X, \mathit{Phases.Mode.RemainInMode}) \end{array}}{\begin{array}{l} \left(\begin{array}{ll} \wedge & \mathit{Hypotheses} \\ \wedge & \Box\mathit{Procedures} \\ \wedge & \Box\mathit{Phases.Decision} \\ \wedge & \mathit{Decide}(X, \mathit{Phases.Mode.RemainInMode}) \end{array}\right) \\ \Box\!\!\Rightarrow \mathit{Decide}(X, \mathit{Phases.Mode.RemainInMode}) \end{array}} \tag{19.10}$$

In other words, if the actual situation and procedures are in conflict, then any decision to remain in or exit from a mode is fully explained by that decision (rather than its contrary), along with the modes of the activity, the actual situation, and the operating procedures.

19.5.1 Determining Rules for Behavior

Dealing with behavior, either of autopilots designed for human use, or of pilots themselves, does not lend itself to the kind of absolute formulation of rules of inference of the sort one finds in formal logic. The criterion for a rule of inference to be valid is that if the premisses are all true, the conclusion is guaranteed to be also. How are we to determine whether a rule such as Rule 19.10 satisfies this exacting standard?

Rule 19.10, like the others in this section, is a rule which determines when we have a satisfactory explanation of an action (likely) involving human agency. An argument can be made, as above, that such a rule holds for the domain of aviation, but it may very well not for the domain of, say, eating candy. How can it then be a *rule of inference*, one might ask, since such rules have universal validity, and therefore apply, if they apply at all, also to candy eating?

The question can be answered by classifying the rules of inference of EL into two:

- the *basis rules* are valid, in the usual sense of validity under all interpretations;
- the *behavioral rules* are formulated on the basis of a restricted domain of activity (aviation rather than candy eating), and are used to indicate what counts as a sufficient explanation of behavior in this domain.

It may very well be the case that one must generate new behavioral rules for aspects of behavior in individual accidents. This is in order; human behavior, even when circumscribed so precisely as in aviation, is a very complex subject, so we should expect that sufficient explanations of that behavior are also complex. One could imagine that a few hundred to a thousand rules might encompass all behavior of relevance for explanatory purposes in this domain. One could well imagine it could be even more. Such questions properly belong to studies of the cognitive psychology and organisational behavior involved.

This should not discourage us from formulating and using behavioral rules, however. A rule is used to explain certain behavior. In our example, the rules above concerning modes were prompted by the specific need to generate an explanation of [1.1]: *CRW opts to continue landing*, given that the crew had noticed that the airport wasn't Frankfurt and that they had '*safety reasons*' for continuing. Does this mean that the rules are ad hoc? In one sense, yes, but in another sense no. It is to be expected that different accidents will expose different features of agency. These features are discussed and reasoned over in any case by investigators. Attempting to formulate them as rules, preferably given previously-formulated behavioral rules, enforces a discipline on the discussion of how the behavior is justified. A rule is formulated, and the discussion of the applicability of the rule includes its validity in the entire domain, not just its sufficiency for the one instance under consideration. This lends precision to the business of deciding whether agent actions were appropriate, and whether procedures need to be amended.

Formulating and discussing behavioral rules, then, brings advantages of formal methods, known from software and hardware analysis, namely their precision and level of generality, into the domain of agency within complex systems. It may be argued that such approaches cannot terminate - that more and more rules will need to be brought in, and that one will never have enough to form a complete theory of agency in a given domain. That may be so, or it may be false - it is not necessary for one to develop a complete theory of agency in order to figure out the role that specific agents' actions played in a specific instance, or to discuss what counts as a good explanation of that role. It is, however, necessary both to be precise and to attain some level of generality; and that is accomplished by formulating rules expressing what counts as sufficient explanations of certain forms of behavior, as we have done above.