Analysis of a Technical Description of the Airbus A320 Braking System

Peter B. Ladkin

CRIN-CNRS & INRIA Lorraine, Bâtiment LORIA, BP 239, 54506 Vandœuvre-Lès-Nancy, France.
Peter.Ladkin@loria.fr

Version of April 15, 1995

Abstract. We analyse the description of the operation of the Airbus A320 braking systems contained in the Flight Crew Operating Manual. We use the predicate-action diagrams of Lamport to express and to complete the description, and give reasons why such a more rigorous expression is preferable.

1. Introduction

On September 14th, 1993, an Airbus A320 landed at Warsaw Airport in Poland in a thunderstorm. It overran the end of the runway, surmounted an earth bank, and came to rest on the other side. Two people died and others were injured in this accident [FI.93a, FI.93b, FI.93c, FI.93d, FI.93e]. This paper analyses the specification of the A320 braking system contained in the Flight Crew Operating Manual [FCOM].

Airplanes are procedurally-oriented machines. Manufacturers devise ways in which they shall be flown as part of the certification process, and descriptions of these methods, as well as descriptions of the system design, are required documentation on every aircraft that flies [FAR, Part 91, Subpart A, Paragraph 91.9]. Crews progress through extensive training on 'approved procedures' for flying the aircraft.

The Airbus A320 aircraft is the first aircraft in commercial operation to use digital 'fly-by-wire' control, that is, in normal operation, the aircraft primary control is achieved by electronic activation from a computer or series of computers (the 'Electrical Flight Control System' [DM93]) which integrate inputs from sensors and from pilot 'requests' and compute outputs to the control surface actuators. The A330, A340 and the forthcoming Boeing 777

are also 'fly-by-wire'. The pilot is one more computer user, no longer in direct physical control of her airplane.

The software-based flight control system is specified, designed and tested on the ground. The aircraft is extensively tested in the air, although no reasonable amount of testing alone can guarantee the reliability that designers must aim for[1], unless the system design procedure is known in advance to guarantee that level of safety [BF93, LS92, LS93].

We analyse the FCOM description of an airplane's systems as a high-level specification. A high-level specification is a rigorous specification of a system in terms of states and predicates that are considered as atomic units at the high-level, which may be analysed into more complex actions, or more complex logical combinations of more-refined state predicates, called a 'lower-level'. The methodology of specification and verification using hierarchical decomposition was pioneered by Dijkstra [Dij68] and extensively developed by Parnas [Par72, Par74]. Milestone systems to use this approach were SRI's PSOS [NBF+80] and SIFT [Wen78]. See [Neu86] for a recent contribution to this area.

We use state-of-the-art logical methods, the *predicate-action diagrams* of Lamport [Lam94b], which despite having a logically rigorous meaning concerning the states and actions of a system, require little more of the user than what she must know and use already.

We analyse the FCOM description initially 'by hand' to identify infelicities in the English, and potential semantic confusions. We then identify state predicates and the system actions and write the description as predicate-action diagrams. Finally we criticise and complete the predicate-action diagrams. We do not assume much knowledge of logic other than an ability to read and understand Boolean logic in English, and a passing acquaintance with a couple of features of temporal logic. Formal details of Lamport's Temporal Logic of Actions (TLA), on which predicate-action diagrams are based, may be found in [Lam94a].

There are no specific mathematical verification steps required in the development of software for flight control systems [RTCA92, para 12]. 'Practical demonstration' may be used, despite fundamental problems with demonstrating these systems to the required level of reliability, as we note in Section 1.3. Nevertheless, the hierarchical design and specification methodologies have been employed and their advantages recognised for a quarter century. See [Rus93] for a detailed discussion of successes and failures, and prospects. We believe there are benefits from considering FCOM descriptions as high-level specifications of complex aircraft systems, subject to the same level of rigorous analysis as any other specification of a safety-critical system. This would ensure that the specification is correct at the high-level, and that the lower-level implementations indeed do what the Operating Manual says they should do. This confers more rigor on the system development process, as well as ensuring the considerable advantage to the crew that the description they have is relatively complete, correct, and not open to misinterpretation.

We suggest that flight crew, who are expected to understand Boolean logic anyway [FCOM, 1.27.10, P11, below], should have a complete, accurate specification of system operation at the high level from which to work. We show that such a specification may be provided using predicate-action diagrams, with little if any complexity over and above the Boolean logic ex-

[1] No reliability number for software alone is required in [RTCA92], since it is a process standard. There is a requirement of a failure rate of 10^{-9} per flight hour for the EFCS as a whole, including software

pressed in English that is currently used. This is in line with current industry thinking, as in the task force convened to study problems with controlled flight into terrain – "treating what happens on an aircraft flight deck as the next stage in a process that begins with the design and manufacture of an airliner and applying the same management techniques [...] a factor that often crops up when crashes are analysed is the failure of the pilot at the controls to stick to standard flight procedure. But that is not necessarily the pilot's fault: [...] Or perhaps poor descriptions mean the procedure is misunderstood." [Eco94]. Our work may help alleviate such problems, if they occur, in another phase of flight.

A method such as ours for ensuring a higher level of rigor in the FCOM, at little cost in readability, could be incorporated into certification techniques for the systems of future air transports.

1.1. The Braking System Design of the A320

The braking system design of the A320 is described in the A320 Flight Crew Operating Manual [FCOM]. There are four main components of this system, of which the two primary components (and the sole components according to which the stopping distances are calculated) are the brakes and anti-skid. The other two are the spoilers (which destroy lift from the wings) and the thrust reversers (which divert engine exhaust to thrust in a forward direction). The brakes and anti-skid system are described in [FCOM, 1.32.30: Landing Gear: Brakes and Anti-Skid]; the spoiler activation in [FCOM, Flight Controls Description]; the thrust reverser actuation in [FCOM, 1.70.70: Power Plant: Thrust Reverser System]. In Section 4 we criticise this description from the semantic point of view. In Section 5, we translate this English-language description into Predicate-Action Diagrams and identify further infelicities.

1.2. Predicate-Action Diagrams and TLA$^+$ Specifications

A predicate action diagram is a simple state diagram in which the states are given by the values of chosen *state predicates* (conditions which have values), and transitions between these states, indicated by labelled arrows, are actions of a given type. Predicate-Action Diagrams corresponding to the braking system descriptions are shown in Figures 1, 2 and 3.

Predicate-action diagrams have a formal, automatable translation into the temporal logic TLA. Predicate-action diagrams and TLA lend themselves naturally to hierarchical specification, and also automated verification and logical checking techniques [Eng94].

1.3. Related Work

Development and Design of Flight Control Computers in Transport Aircraft. The Airbus approach to using computers in civil aircraft design is presented in [Pot93]. A description of the Electrical Flight Control System of the A320 may be found in [DM93], independent reports of facts and findings related to the Warsaw accident in [FI.93a, FI.93b, FI.93c, FI.93d, FI.93e], and the official report on the Warsaw accident itself is [Inv94].

System Safety Analysis. An extensive introduction to engineering safety and reliability issues may be found in [HK81]. Fault trees were originally developed for use in the analysis of the design of nuclear power plants [WGNH81]. Engineering safety issues in general and for transport aircraft in particular are discussed in [LT82], and the story of DC-10 accidents and their connection with engineering and administrative procedure as well as moral lessons to be drawn are extensively discussed in [FB92]. Safety analyses of computer systems in general and software in particular are discussed in [Lev86, Lev91, PvK90], and a series of techniques based on Petri Nets are proposed in [LS87].

Formal Specification of On-Board Computer Systems. The Flight Warning Computer of the A330/340 Series of aircraft was specified in the standard formal description technique LOTOS in [GH94]. A critique of this particular approach appears in [Rus93, pp102-104]. A preliminary effort to analyse the A320 braking system specification using Boolean logic expressed in TLA, predicate-action diagrams, and fault trees may be found in [Sin94].

Verification of Algorithms for On-Board Flight Control Systems. The SIFT project at SRI was the first major project undertaken to design and verify a computational flight control system [Wen78]. During this project, the series of agreement problems now called *Byzantine Generals Problems* as well as algorithms for solving them were first discovered [PSL80, LSP82], as well as the first algorithms for synchronising clocks between computers [LMS85], whose descendants are called *interactive convergence algorithms* [Sch87, Sha92]. The rationale for synchronising clocks in on-board flight control systems, as well as an account of how the relevant algorithms are shown to be correct may be found in [Rv93]. Various interactive convergence algorithms were formally proved correct using the EHDM verification system developed at SRI International in [Sha92].

Testing of Flight Control Systems. The failure rate to be aimed for in safety-critical systems such as flight-control systems is 10^{-9} [RTCA92]. However, testing systems with such low failure-rate requirements cannot provide a sufficiently high level of confidence that such high levels of reliability have been achieved [LS93, BF93]. One must have prior confidence that the system satisfies the stringent safety criteria before testing is performed (to the joy, one would imagine, of the Society of Test Pilots).

Safety Analyses of Mishaps. An analysis of a particularly appalling series of accidents with a medical radiation machine are documented in gruesome detail in [LT93]. A fragment of a formal safety analysis based on the standard formal description technique LOTOS which could have discovered the culpable faults is presented in [Tho94b, Tho94a]. A very readable account of many major mishaps, along with analysis of all fatal A320 accidents to date, is [Mel94].

2. The Design of the A320 Braking System

The braking system design of the A320 is described in the A320 Flight Crew Operating Manual [FCOM]. We repeat here the description of the brakes, antiskid, spoilers and thrust-reversal systems from [FCOM]. We quote the relevant page contents and identification from

[Inv94]. (The page numbering system depends on the operating company, so may not be the same as in [Inv94] in arbitrary copies of the Manual.)

Each page is identified by a section number (three sets of digits separated by periods), a *REV*ision number, a *SEQ*uence number, and a Page number. Pages are updated dynamically, and a legally 'current' manual is required to have all its pages with a given series of these four identification codes. Although the reader may find it somewhat tedious going, it is essential to refer to these descriptions for the analysis which follows. The rest of this section consists in an exact quotation.

Landing Gear: Brakes and Anti-Skid (1.32.30)

1.32.30, REV 16, SEQ 001, P 1

Description

General

The main wheels are equipped with carbon multidisc brakes which can be actuated by either of two independent brake systems.
The normal system uses green hydraulic pressure whilst the alternate system uses the yellow hydraulic system backed up by hydraulic accumulator.
An anti-skid and autobrake system are also provided.
Braking commands come either from the brake pedals (pilot action) or the autobrake system (deceleration rate selected by the crew).
All braking functions (normal and alternate braking control, anti-skid control, autobraking, brake temperature indication) are controlled by a double channel Brake and Steering Control Unit (BSCU).
The main gear wheels are fitted with fusible plugs which protect against tire burst in the event of overheat.

▷ Main gear wheels are also equipped with brakes cooling fans which permit a high speed cooling of brakes.

Anti Skid System

The anti skid system provides maximum braking efficiency by maintaining the wheels at the limit of an impending skid.
At skid onset, brake release orders are sent to the normal and to the alternate servovalves as well as to the ECAM system which displays the released brakes.
The anti skid is deactivated when the speed is lower than 20 kts (ground speed).
An ON/OFF switch activates or deactivates the anti skid system and nose wheel steering.

Principle

The speed of each main gear wheel (given by a tachometer) is compared with the aircraft

speed (reference speed). When the speed of a wheel decreases below 0.87 time [*sic*] reference speed, brake release orders are given to maintain the wheel slip at that value (best braking efficiency).
In normal operation, the reference speed is determined by BSCU from the horizontal acceleration from ADIRU 1 or ADIRU 3.
In case of ADIRU 1 and ADIRU 3 failures, reference speed equals the maximum of either main landing gear wheel speeds. Deceleration is limited to 1.7 m/s^2 (5.6 ft/s^2).

1.32.30, REV 15, SEQ 001, P 3

Auto Brake

The aim of this system is: [....]

System arming

The crew may arm the system by depressing the LO, MED or MAX push button switches, provided all the following arming conditions are met:

- Green pressure available
- Anti-skid electrically powered
- No failure in the braking system

Note: *Auto brake may be armed with parking brake on.*

System Activation

Automatic braking is initiated by the ground spoiler extension command (refer to 1.27). Consequently in the event of an acceleration stop, if the deceleration is initiated with the speed below 72 KTS, the automatic braking will not be operative because the ground spoilers will not be extended.

System disarming

The system is disarmed by:

- Pressing the push-button switch or,
- Loss of one or more arming conditions or,
- Applying sufficient force to the rudder pedals when autobrake is operating:
  - In MAX mode both pedals must be depressed,
  - In MED or LO desarming [*sic*] may be accomplished by action on one pedal only,
- Ground spoiler retraction (refer to 1.27).
- Flight condition since 10 seconds.

1.32.30, REV 15, SEQ 001, P 4

Operation

There are four modes of operation:

- Normal braking.
- Alternate braking with anti-skid.
- Alternate braking without anti-skid.
- Parking brake.

Normal Braking

Braking is normal when:

- green hydraulic pressure is available
- A/SKID and N/W STRG is ON
- PARKING BRAKE is not ON.

Anti-skid is operative and autobrake is available.

The control is electrically achieved through the BSCU:

- either via the pedals
- or automatically
  - on ground by autobrake system
  - in flight by setting the landing gear lever to the up position

Anti-skid system is controlled by the BSCU via the normal servo valves.
No brake pressure indication is provided.

Alternate Braking With Anti-Skid

Active when green hydraulic pressure is insufficient and provided:

- yellow hydraulic pressure is available
- A/SKID and N/W STRG switch is ON
- PARKING BRAKE is not ON

The automatic switching between the green and yellow system is achieved by an automatic hydraulic selector.
Control is achieved by the pedals through the auxiliary low hydraulic pressure distribution line acting on the dual valves. The BSCU controls anti-skid system via the alternate servo valves.
The pressure delivered to the left and right brakes as well as the accumulator pressure are indicated on a triple indicator located on the center instruments panel.
Autobrake is inoperative.

1.32.30, REV 15, SEQ 001, P 5

Alternate Braking Without Anti-Skid

The anti-skid system is deactivated:

- electrically (A/SKID and N/W STRG sw OFF or power supply failure or BSCU failure)
- or hydraulically (Y + G sys lo press, the brakes are supplied by the brake accumulators only).

Control is achieved by the pedals (acting on the dual valves).
Alternate servo valves are fully open.
Brake pressure has to be limited by the pilot by refering [*sic*] to the triple indicator to avoid wheel locking.
The accumulator is dimensioned to supply at least seven full brakes applications.
Auto brake is inoperative.

Parking Brake

Operating the PARKING BRK control handle deactivates the other braking modes and the anti-skid system.
Brakes are supplied by yellow hydraulic system or accumulators pressure via the dual shuttle valves. Alternate servo valves are open allowing full pressure application.
Accumulators maintain the parking pressure for at least 12 hours.
Yellow accumulators can be pressurized by depressing the yellow elec pump sw.
Brake pressure indications are available on the triple indicator.

Flight Controls Description (1.27.10: REV 18, SEQ 106, P 11)

Ground Spoiler Control

Achieved by the spoilers 1 to 5.

- **Ground spoilers are armed** when the speed brakes control lever is pulled up into the armed position.
- Ground spoilers automatically extend:
  - (At MLG touch down) OR (During T.O run at speed greater than 72 KT)

    WHEN
    - (They are armed and all thrust levers are at idle) OR
    - (When reverse is selected on at least one engine (remaining engine at idle)
- Ground spoilers retraction is achieved when:
  - (All thrust levers are set at idle) AND (Speed brake control lever is pushed down)

    OR

- One thrust lever advanced
  - above 20°
  - at least 3 sec between 4° and 20°

Power Plant: Thrust Reverser System (1.70.70, REV 15, SEQ 005, P 2)

[...]

Actuation Logic

Deployment requires:

- one FADEC channel operating with its associated throttle reverse signal
- main gear compressed signal from at least one LGCIU
- TLA reverse signal from at least one SEC

Before the transit completion the FADEC sets reverse idle thrust.

3. Logical Analysis

Specifications are logical objects. They make assertions in a language (syntax), which is interpreted (semantics) to make assertions. When specifications are required to be precise, then a formal language with a precisely defined semantics may be used. The main advantages of non-formal languages are that they are readable and intuitively understandable by those without exposure to formal languages. The main disadvantage is that they are imprecise and specifications may be ambiguous in crucial ways. The main advantages of formal languages is that they are precise (whether a particular system behavior conforms or not is a question with a definite answer) and they are manipulable by computer, thus facilitating the use of computer-based tools such as verifiers, specification analysers, and automated system design aids. I shall refer to the FCOM Section 1.30.32: Landing Gear, Brakes and Anti-Skid here as LGBA followed by the page number; similarly 1.27.10: Flight Controls Description as FCD with the page number.

The simplest example of a formal language for specification is propositional or Boolean logic. An example of a formal method based on propositional logic is Fault Tree Analysis [WGNH81].

The specification of the braking system design includes some causal or temporal dependencies, thus ensuring that Boolean logic by itself is ultimately insufficient, and must be supplemented by a semantics that considers change over time. *At skid onset, brake release orders are given ..* (LGBA P1). *The system is disarmed by [...] applying sufficient force to the rudder pedals when [...]* (LGBA P3). *Ground spoilers retraction is achieved when .....* (FCD P11). Section 4 discusses whether some of these are assertions concerning the state of the system, or whether they are assertions concerning actions causing certain other actions. Either way, it is incontrovertible that the *state* of the system changes, and that some of these

changes may be appropriately described by a logic which also takes into account that Boolean variables may change value during operation of the system (for example, the ground spoilers are sometimes retracted, sometimes not). This point has been remarked in [BA93], where the inadequacy of simple fault trees with Boolean logic to describe such situations was also noted.

The simplest extension of Boolean logic to include the change of Boolean values of propositions with time is Temporal Logic. Unsurprisingly, temporal logic has found widespread use in computer science for describing situations in which interactions amongst components of a system occur over time. See e.g. [MP92]. In temporal logic semantics, a collection of Boolean values of the propositions describes a *state*. A proposition is now called a *state predicate*, since its Boolean value depends on the state in which it is evaluated.

How do we describe a change in state? A change of state is called a *transition*, and a cause of a change of state is called an *action*. We will briefly indicate why this ontology is necessary. *Transitions* transform one state into another, for example a transition which is an instance of the *action* `x` $\leftarrow$ `(x+1)` transforms a state in which the value of variable x is 2 to one in which the value of x is 3 (other actions can also yield this result). A transition is uniquely specified by the total of the effects it has on all the propositions. An *action* is that which causes a state transition. Actions may cause different transitions depending on what state they operate on, for example the action `x` $\leftarrow$ `(x+1)` has different results depending on the previous value of x. Actions are thus binary relations on states (the 'previous state' and the 'resultant' state). (One could maybe require actions to be state *functions*, but functions are just special types of relations, and also other parts of a state may 'spontaneously' change when the action is performed, so that the resultant state may not be uniquely determined. Hence the notion of relation is appropriately general.)

So, when we need to express change in Boolean values over time, we must pass from Boolean logic to a logic with state predicates, transitions and actions. How complex the state predicates must be depends on the problem, and on the ingenuity of the engineer analysing the system. The ontology has become somewhat richer, and at this point one has to choose how one is to express some general relations amongst the denizens. One approach which has shown its practical value in describing both digital and hybrid systems is the Temporal Logic of Actions (TLA) of Lamport [Lam94a]. Besides giving primacy to actions, TLA is able to incorporate arbitrary mathematics in describing the relations amongst states, and is able to handle assertions involving temporal measurement constraints ('real-time' constraints) without modification [AL94]. TLA incorporates modules, so the values of encapsulation and information-hiding are available. Nevertheless, our goal is to use as little apparatus as possible.

We have noted that some of the description in the A320 Flight Crew Operating Manual involves assertions concerning actions. We further note that real-time constraints also appear essentially: *The system is disarmed by: [...] flight condition since 10 seconds* (LGBA P3). *Ground spoilers retraction is achieved when: [...] at least 3 sec between* 4° *and* 20° (FCD P11).

4. A Critique of the Description

The description above serves as a specification for flight crew members of the states and transitions of the A320 brakes, including the antiskid and autobrake systems. Operation

of the autobrake is not described, although some of its modes are referred to. Similarly, one may reasonably infer from analysis of the description that antiskid has two modes, '*on*' and '*off*', and that the system transits from one mode to another through operation of the *A/SKID* switch. For this description to hold to the normal standards required of specification, these properties would need to be asserted explicitly. One cannot imagine that the handbook description would lose any coherence if they were, and the space is available to do so (only 3.5 pages of 4 are used). We shall assume these properties of A320 antiskid operation. Similarly, we shall assume that *autobrake* has two modes, '*on*' and '*off*', and that its operation is specified by the further parameter '*deceleration rate*' (the existence of this parameter may be inferred from the explicit statement in the specification that its value of *LO*, *MED* or *MAX* is selected by the crew).

The flight crew's understanding of the operation of the system comes from the FCOM. Thus, the FCOM serves as a specification of the A320 braking system operation at a high level which needs to mention only few of the engineering details. Precision is required. However, specifications in English can be ambiguous.[2] And ambiguity when precision is required can be dangerous. Our critique focuses on features of the description where precision is lacking.

The description describes *modes* and mode *changes* of subsystems of the braking system, and how transitions are accomplished between modes. Thus it adheres implicitly to the state-predicate/transition model of systems described in Section 3.

State-Predicate or Action? Given the state-predicate/action semantics of the description, there is a recurrent ambiguity between whether a state or an action is being described, that is caused by a certain grammatical construction in English. Given that the state-predicate/action type distinction is the foundation of the interpretation of the specification, this is a fundamental problem.

Here is an example: does *[X] is armed when [...]* reflect a state predicate or a predicate describing an action? Similarly *is activated*, *is pulled up*, *is set at idle*? English constructs a third-person state predicate by using the auxiliary verb 'is' with the past participle, as in 'is pulled up'. Equally, it constructs the present passive voice (which speaks of an action) with 'is' + past participle, as in 'is pulled up'! So, is *the speed brakes control lever is pulled up [...]* a phrase denoting an action of pulling the levers up, or rather is it a phrase denoting the position of the levers as *pulled up*? Maybe we can look at the preposition following the verb. Consider *[...] is pulled up* **into** *[..]* vs. *[...] is pulled up* **in** *[...]* - the former denotes an act, whereas the latter a state, providing we can trust the FCOM writers to use the appropriate prepositions. But this is a subtle distinction, and what evidence do we have to trust the writers to use this distinction appropriately? For example, there are obvious uncorrected spelling errors all over (see below).

For a more sensitive example, consider *Ground spoilers retraction is achieved WHEN (all thrust levers are set at idle AND speed brake control lever is pushed down) OR [...]*? The difference between state-predicate and action interpretation is operationally crucial. Suppose the ground spoilers are extended, but thrust levers are at idle and the speed brake control lever

[2] Robert Dorsett and others have noted that there is even more likelihood of problems when manuals are written by non-native speakers, or translated from one language into another.

is down (and the condition following the OR is unambiguously false). If the state interpretation is meant, then there is a fault in the system. If the action interpretation is meant, then there is not necessarily a fault - the pilot should cycle the levers, so that the *action* of moving them into the said positions causes the desired action of the ground spoilers.

Another possible state/action ambiguity is *The system is disarmed by [...] OR loss of one or more arming conditions OR [...]* (LGBA P3).

4.1. Other Semantic Ambiguities

- The Boolean connectives are not always clearly stated, and thus it is possible to interpret the specification in a way in which it was not intended:
  The system is disarmed by:
  - *Pressing [...] or,*
  - *loss [sic] [...] or,*
  - *Applying [...]* [no following conjunction],
  - *Ground spoiler retraction* [no following conjunction].
  - *Flight condition since 10 secs.*

  (LGBA P3).
  Two of these phrases have, and two have no, logical conjunction expression following them. Are the missing conjunctions logical `or` or logical `and`? If either of them were to be *and*, parentheses are also needed to disambiguate the expression obtained. The situation is complicated by the different choice of punctuation. There are commas following the first three phrases and a period (full stop) following the fourth. If this punctuation difference is meaningful, it would lead us to interpret the first four phrases as a disjunction, and the final phrase as conjoined with this (bulleted phrases terminated with periods often indicate logical conjunction). Thus, letting the conditions (which in the case in which actions are mentioned, we interpret as the postcondition of the actions) be represented by *X*, *Y*, *Z*, *U* and *W*, we regard the serious interpretation possibilities to be (*X or Y or Z or U or W*) or (*(X or Y or Z or U) and W*), and there are many others theoretically obtainable by varying the parentheses in the second expression, or by changing the third *or* to *and* and then varying parentheses.
  It would have been more helpful in the manual to give an unambiguous Boolean expression (in English or an Anglicised version of Boolean logic, both of which are otherwise used in the specification).
- Does *desarmed* [*sic*] *by action on one pedal only* mean that the pedal must be moved, or merely that holding one pedal down causes the condition?
- In *Auto Brake* Section *System arming*, it is stated that three '*arming conditions*' must be met for the autobrake system to be armed when certain switches are 'pushed'. In Section *System disarming*, one of the subexpressions in the expression giving the conditions under which the system may be disarmed is '*loss of one or more arming conditions*'. The specification does not define the phrase 'arming conditions'. We presume that the only arming conditions are the three mentioned under '*System arming*', but there is no explicit

or implicit indication (such as highlighting of the phrase, or an adverb such as '*above*' employed at the second occurrence of the phrase).

- The expression '*wheel slip*' is used when describing the operating principle of the anti-skid system. The value of wheel slip maintained by the anti-skid system when it is operating is indicated by the phrase '*that value*'. Whatever '*that value*' may be, it is apparently determined by the expression '*wheel speed decreases below* 0.87 *time* [*sic*] *reference speed*'. The dimensions of the *wheel slip* parameter are not given, nor is any value. We must infer that *wheel slip* is dimensionless, and that it is a constant of proportion between wheel speed and reference speed, but this inference, although likely, is by no means certain. There is no explicit expression of the constraint maintained by the anti-skid system, leaving us to suppose it is something like *wheel speed* = (0.87 × reference speed).
- We think it is unlikely that the autobrake system was intended to be in more than one mode at once. Can we presume there is an interlock on the LOW, MED , MAX switches, to ensure that there is no mode confusion, or can the system be in more than one mode at once? What happens when two or more of these buttons are 'pushed' at the same time? [The answer is that the switches are indeed interlocked and there's arbitration for cases of multiple buttons being pushed at the same time [Anon94].]
- There's a further infelicity in the autobrake arming/disarming procedure specification. Provided three 'arming conditions' are satisfied, the system is armed by pressing the push button switches marked *LO, MED, MAX*. The system is also disarmed by '*pressing the push button switch*'. Is this the same or a different switch? If the same switch (i.e. the three switches are toggles), then the action of pressing a switch with the three 'arming conditions' met entails as postcondition both that the system is armed and that the system is disarmed. We may presume that 'disarmed' is intended to be a mutually exclusive state with 'armed', and that therefore this specification is contradictory. [The three switches are toggles. The description of operation of toggles is logically infelicitous. It is necessary also for the *A/SKID* switch to be on, so that the antiskid is armed, for the autobrake switches to function as 'described'.]
- the 'speed' of the aircraft is mentioned in a number of places (*during T.O. run at a speed greater than 72 KT* (FCD P11), *[...] with the speed below 72 KTS, [...]* (LGBA P3). Also, the *speed of each main gear wheel* (LGBA P1) and the *aircraft speed (reference speed)* (LGBA P1). From the last quote, there is more than one 'speed' involved. Which is which? We could guess that 'speed' without a qualifier means 'reference speed'. In fact it must have a precise definition in terms of sensor measurements which actuate the logic. Why is this definition not used in the FCOM?
- Hydraulic pressure is described as 'available', 'insufficient', 'lo press'. What are these values, in particular is 'lo press' a synonym for 'insufficient'?
- Similarly, *autobrake* is described as 'available', 'armed' and 'inoperative' in various braking modes. What exactly are the states of *autobrake*, and how are they changed?
- *Operating the PB control handle deactivates the other br-modes and anti-skid.* This is another ambiguous description. If we are to take this literally, then it says that *any action* with the handle (the normal meaning of the word 'operating') will deactivate all other braking modes and anti-skid. Presumably, one action puts the parking brake on and a

corresponding but contrary action puts it off. According to the description, putting the brake off also deactivates the other braking modes. The brakes would thus be in none of the four modes, which is simply a contradiction with what is required of the modes (they must be pairwise exclusive and jointly include every possible state of the braking system). One suspects that what is meant is that *moving the PB control handle into the ON position activates parking-brake-mode and deactivates the other brake-modes and anti-skid mode.* [Note that *autobrake* is not one of the four braking modes. According to the specification it may be 'armed' while in parking brake mode, 'available' in normal-braking mode, and 'inoperative' in both alternate-braking modes.]

- The phrase *A/SKID and N/W STRG sw off or power supply failure or BSCU failure* is ambiguous.
  - If '*sw*' means '*switch*', then *A/SKID* is is a synonym for '*antiskid switch on*' and parentheses are needed to disambiguate the Boolean expression, which has the form '*X and Y or Z or W*', which may denote one of three non-equivalent Boolean expressions, with the Disjunctive Normal Forms *(X and Y) or (X and Z) or (X and W)*, *(X and Y) or (X and Z) or W*, and *(X and Y) or Z or W*, where *X* is *antiskid on*, *Y* is *N/W switch off*, *Z* is *power supply is failed*, and *W* is *BSCU is failed*.
  - If '*sw*' means '*switches*', then the first expression is a conjunction-noun-phrase *A/SKID and N/W STRG switches both off*, and the phrase is equivalent to *((not X) and Y) or Z or W*, which is different from any of the previous three.
  - A particularly notable example of inexact semantics is: *the anti-skid system is deactivated: [...] hydraulically (Y + G sys lo press, the brakes are supplied by the brake accumulators only).*
    1. '+' seems to mean 'and'. In Boolean notation, used also in engineering, it means 'or'!
    2. there is no verb, and no other potentially disambiguating words such as 'both';
    3. the phrase in parentheses before the comma appears to be a precondition of the action (hydraulic deactivation of the anti-skid system), and the phrase after appears to be a post-condition or consequence. But neither the layout of the conditions, nor use of disambiguating words such as 'afterward' or 'then' differentiate these two quite different semantic roles;
    4. what does it mean that the brakes are 'supplied' *by ....*? That they're *activated by ....* ? The verb chosen is semantically inappropriate to the context.
    5. One must guess that *Y + G sys lo press* means *low pressure in both yellow and green hydraulic systems*, given the lack of verb, or disambiguating words, and the ambiguity of the meaning of '+'. And one may guess wrong.
- There is another Boolean ambiguity in expression of the ground spoiler control (FCD P11). In an otherwise exemplary piece of Boolean logic expressed in English, the final phrase '*One thrust lever advanced [...]*' is followed by two bulleted items. The connective must be guessed. We would guess '*OR*' but it is possible (although unlikely) for it to be '*AND*'.

4.2. Spelling and syntactical non-uniformities

Writing code in a programming language or in a specification language forces one to ensure that keywords are spelled uniformly and correctly, and that attention is paid to punctuation. This is because of the way compilers and other syntax checkers must operate. Designers and builders of compilers and syntax checkers would have evolved the technology over the last thirty years to allow compilers a *DWIM* ('do what I mean') mode if it was easily possible. They have not. The most fruitful assumption to make of a programmer or specification engineer's code is that it follows the syntactic rules. Uniformity aids comprehension. The description we are considering, given in the FCOM, is a specification which does not adhere uniformly to the standards of syntax required of an engineer building the system.

Spelling non-uniformities : *speed brake***s** *control lever* vs. *speed brake control lever* (FCD P11); *desarming* (LGBA P3); *Anti skid* (LGBA P1) vs. *anti-skid* (LGBA P3,P4,P5); *[...] below 0.87 time reference speed* (LGBA P1); *switch* vs. *sw*; *Tachy.* (LGBA P2) for *tach***o***meter* (LGBA P1).

Other non-uniformities : Items in a list (or, inadvisedly, subclauses in some Boolean constructs) are nested. Both hyphens ('-') and bullets ('·') are used, and when nested they are used alternately. Sometimes, the hyphen is used as the outermost listifier (FCD P11, LGBA P4,P5), and sometimes bullets (LGBA P3). Sometimes, the levels of nesting are mixed hyphens and bullets (the disjunction in '*Ground spoilers retraction is achieved when [...]*' in FCD P11). As we have noted, these bullets sometimes have the meaning of Boolean connectives. They are not semantically neutral. It seems to us advisable to use them in a uniform manner.

4.3. Anti-Skid System Logic

The logic of the antiskid system is given in a logic diagram in [FCOM, Landing Gear: Brakes and Anti-Skid 1.32.30 REV 15 SEQ 001 P 2]. We describe it in words, and critique it.

[begin description in words] Let the linear speed computed from the wheel tachometer '*on impact*' be v_0, the aircraft longitudinal deceleration be γ_{ir} and the deceleration rate selected by the pilots be γ_{prog}. It is also indicated on the diagram that if autobrake is active there are precisely three deceleration rates selectable, depending on which of the *LO*, *MED* and *MAX* autobrake switches are *ON*.

The notation v_{ref} is used for the "reference speed". If *autobrake* is not active, v_{ref} is equal to $v_0 - \gamma_{ir}.t$. If *autobrake* is active, v_{ref} is equal to $max(v_0 - \gamma_{ir}.t, v_0 - \gamma_{prog}.t)$. These are compared with the wheel speed (the factor of 0.87 does not show on the diagram) to determine the state of the brake *release order*. Brake release is accomplished by two servo valves, one on the green hydraulic system and one on the yellow system, which receive the signal from the release order mechanism in sync.

[end description in words]

A major point is that the expression $v_0 - \gamma_{ir}.t$ must surely be incorrect. Supposing the speed v_0 is measured accurately, the current aircraft speed is not given from v_0 by taking *current*

acceleration multiplied by elapsed time, since the current acceleration might be higher or lower than the average. The correct expression is $v_0 - \int_0^{t_e} \gamma_{ir} dt$, where t_e is elapsed time from when v_0 is measured. Accelerometers measure these integrals accurately. We have no reason to believe that a cruder approximation than the integral is used. A critical point is that in order to determine the current value of the reference speed, the *point of impact* in time must have been determined, as well as the *velocity of the wheel at impact time*. Whether the velocity is obtained from one or from both wheel tachometers is not specified, neither is it specified how the velocity is determined if two wheel tachometers give divergent readings. There is currently reason to believe that the system did not accurately determine either parameter in the Warsaw accident (*op.cit.*).

One minor point is that the arithmetic expressions, while written correctly, do depend on the higher precedence of multiplication over subtraction. Furthermore, the '-' sign for subtraction on the diagram is short and potentially unclear. Parentheses would eliminate the chance of misinterpretation of the formulas.

5. The FCOM Specification as Predicate-Action Diagrams

In Section 4 we analysed the specification for ambiguous semantics. In this section, we analyse the logic of the specification. We use predicate-action diagrams to represent the information contained in the specification. Predicate-action diagrams are almost self-explanatory. The nodes are partial states, that is they are collections of values of selected state predicates. We call these partial states 'states'. 'Actions' change the values of the state predicates, and are represented by arrows between the 'states'. The 'actions' represented in the diagrams are collections of *all* the actions that can change the value of one of the predicates of the 'state'. It is required that the result of any action that changes the value of the 'state' belongs to one of the 'actions', and that all the 'states' that result from any of the actions belonging to an 'action' appear in the diagram. Thus, a predicate action diagram focuses on certain predicates, and shows how the values of those state predicates are changed by actions of the system, which are grouped into sets of actions that all have the same effect on the selected state predicates.

In the representation of the FCOM specification in predicate-action diagrams, we represent the 'actions' by logical disjunctions of its component actions. We also label the 'states' by certain useful indicative expressions. Thus, a braking-mode 'state' labelled with *normal* satisfies the state predicate that *braking-mode = normal*. However, these labels, while helpful, do not necessarily correspond to explicit state predicates. The state predicates explicitly asserted in the 'state' are given by conjunctions written in an ellipse attached by a line to the 'state'.

5.1. Braking Mode

The braking mode specification from the FCOM is shown in Figures 1 and 2. The state predicates asserted to hold in a given mode are shown as ellipses connected by an undirected line to the 'state'. Each 'state' is a mode. The state predicates defining how braking control

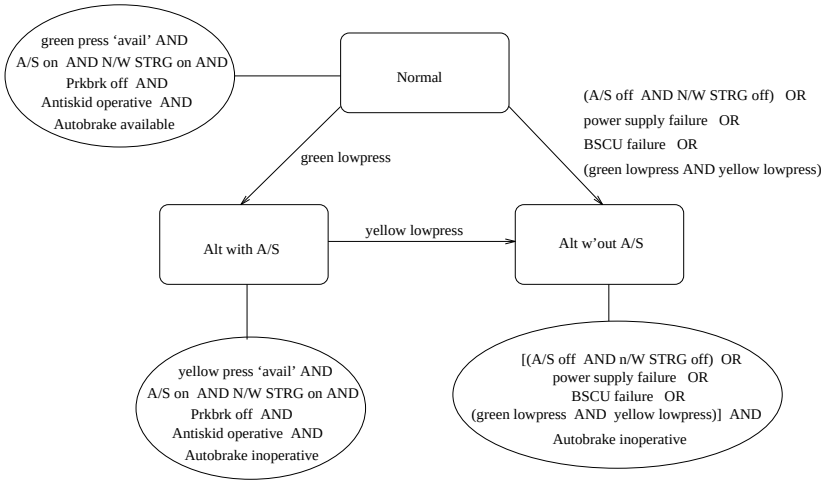

Fig. 1. The Braking Modes from the FCOM Specification

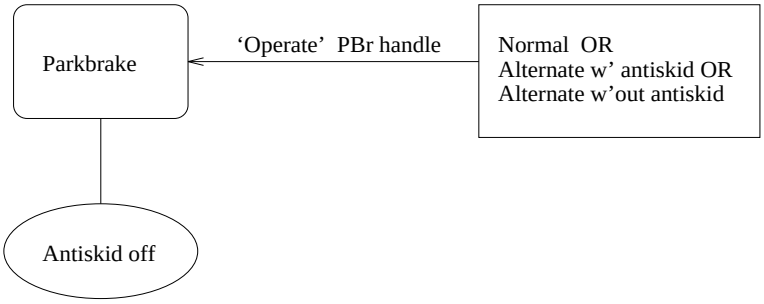

Fig. 2. The Parking Brake Mode from the FCOM Specification

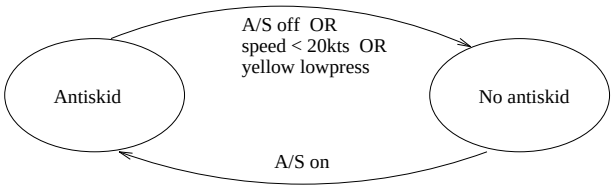

Fig. 3. The Antiskid Condition from the FCOM Specification

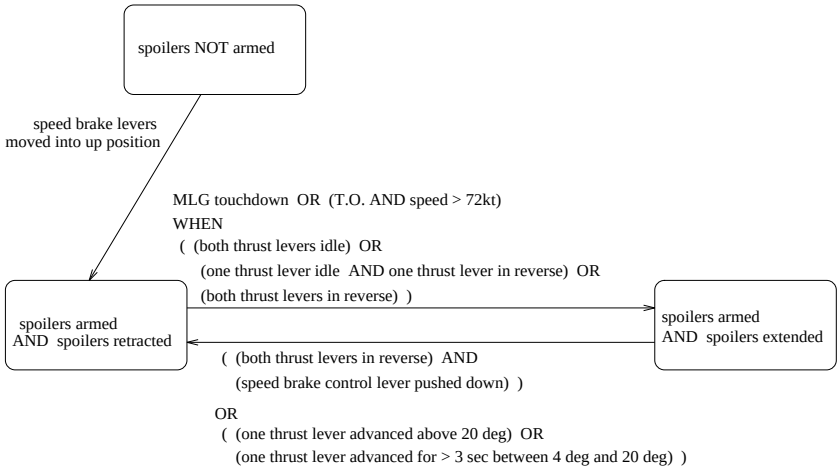

Fig. 4. Ground Spoiler Deployment from the FCOM Specification

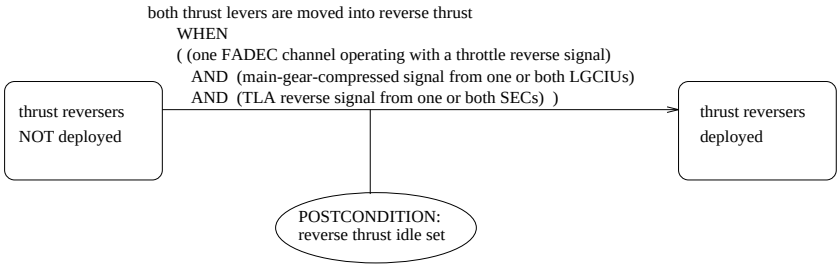

Fig. 5. Reverse Thrust Deployment from the FCOM Specification

is achieved, or how the BSCU operates, in each mode, are omitted. All the other predicates defined in the FCOM description are shown, with the exception of the predicate defining the mode, which is contained implicitly in the mode name in the 'state' node. The actions are shown in full as a Boolean disjunction of actions in the one case where it is applicable.

In Figure 2, we are able to coalesce the 'states' *normal*, *alternate with antiskid* and *alternate w'out antiskid* into one 'state' in order to simplify the diagram. The resulting pair of diagrams is much simpler than one diagram which includes *parkbrake* mode as a fourth 'state' and has three more transitions, all with the same label, from the three other modes. This facility is similar to that obtained by the use of hierarchical states in Statecharts [HLN+90]. For predicate-action diagrams, it comes 'for free' from the definition of 'state' and 'action'.

There is one case, the transition from *normal* to *alternate w'out antiskid* in which the description is ambiguous between an action description and a state description. We have shown the Boolean description both as an 'action' and as part of a 'state' description. It is thus obvious from the diagram that although this is an infelicity, the two interpretations are consistent. (We contrast this with the situation with the *speed brakes control lever*, in which the different interpretations have different semantics.)

We may now critique the specification further. According to the diagram, it is not possible to transit from any mode other than *normal* to *alternate with antiskid* or *alternate without*

antiskid, or from *alternate with antiskid* to *alternate without antiskid*. One can surmise that this situation does not correspond to the intended design. Thus, the specification is incomplete. Many state transitions are not defined, and the results of many actions (such as the pressure on either green or yellow systems returning to '*available*') are not defined. While this is not clear from the written specification, it becomes very clear when the information in the specification is represented as a predicate-action diagram.

5.2. Antiskid Operation

There are modes with *antiskid* and modes without *antiskid* shown on the braking mode diagram in Figure 1. Figure 3 shows the FCOM description of antiskid alone (omitting the logic diagram showing the operation of *antiskid*). This reflects an infelicity in the FCOM description, in which one has to look in two spatially separated locations to obtain the information relevant to understanding *antiskid*: some information is in the section on the *Antiskid System* (LGBA P1) and other, disjoint, information is in *Alternate Braking Without Anti-Skid* (LGBA P5). Both pieces of information are relevant to understanding the operation of *antiskid*, and a reader may be misled by either section into thinking that she had all the pertinent information. Thus, the predicate-action diagram representation induces one to inquire if there is a logically equivalent way of representing the information, so that it may be placed in one location in the FCOM.

For example, by comparing the *antiskid* predicate-action diagram with the braking mode diagram, one can inquire if there is a simple 'factoring' of the two diagrams, so that all the information appearing in the transition between *alternate with antiskid* and *alternate w'out antiskid* is contained in the antiskid transition diagram in Figure 3. We believe not. Thus, the specification may be criticised on further structural grounds that the antiskid mode is neither fully integrating into the braking system description, nor orthogonal to it.

5.3. Ground Spoilers

The description of ground spoiler operation in [FCOM, 1.27.10 P11] is a Boolean formula, and looks more precise and relatively complete than some of the other descriptions we have analysed. But it may be seen immediately from the predicate-action diagram in Figure 4 that the description is incomplete. It is not clear how the spoilers may be disarmed.

The design required that the system may not pass directly from spoilers-extended to spoilers-disarmed without retracting the spoilers. This is a non-trivial condition. Similarly, suppose that the condition defining the transition from spoilers-armed-and-retracted to spoilers-extended is true, and the speed brakes lever is moved into the *UP* position. The diagram asserts that the system must pass through the state of armed-but-retracted before the spoilers can be extended. Both these conditions are non-trivial consequences of the diagram. We may surmise that they are indeed true of the system.

Thus, the only incompleteness is the missing transition from spoilers-armed-and-retracted to spoilers-not-armed. The existence of this single omission lends credence to our assertions that (a) Boolean logic expressed in English as used in this description is often preferable

to pure English descriptions in specifications; and (b) the predicate-action diagrams help to detect incompleteness that may not be evident from the written description.

5.4. Reverse Thrust

The reverse thrust description is particularly incomplete. How does one move from reverse-thrust-deployed to reverse-thrust-not-deployed? One can surmise that one moves the thrust control levers into or through neutral detente, since this is the normal way on most airplanes. However, the A320 is sufficiently different in operation from its predecessors that one might also surmise certain other conditions must be true. The deployed-to-stowed condition should be explicitly stated.

The *WHEN* condition in the action description is intended to be a precondition for the state-transition action to occur. The transition action should be distinguished from the pilot action of moving the thrust levers into reverse thrust. If the preconditions are not fulfilled when the levers are moved, the transition does not occur and the thrust reversers remain not deployed.

There is also a misleading condition stated in the description that when the reversers are deployed, reverse idle thrust is set. We surmise that this is actually a postcondition of the deployment action, and not a state predicate (as it appears to be from the way it is expressed in the description). Reverse thrust would be pretty useless if it was always in idle when deployed!

5.5. Conclusions

We see that writing the FCOM description as predicate-action diagrams allows us to see the logical relations (or lack of them) between the various states, and gives clear guidance on how to complete the logical descriptions. All possible transitions between modes should be shown, and their corresponding actions described as required by the semantics of the diagrams. The result will be a complete logical specification of all mode changes relating to the brakes subsystem (defined by the set of state predicates which pertain to the brake system).

6. Revising the Predicate-Action Diagrams

The predicate-action diagram representation of the FCOM description can easily be completed, using the predicate-action representation, in the following steps:

- list all state predicates occurring in any diagram;
- determine the values of all the state predicates in every state denoted, and list them with the state;
- add transitions each way between all pairs of states;
- notate each transition with its action description (where possible);
- disambiguate the action descriptions from a given state (if necessary);

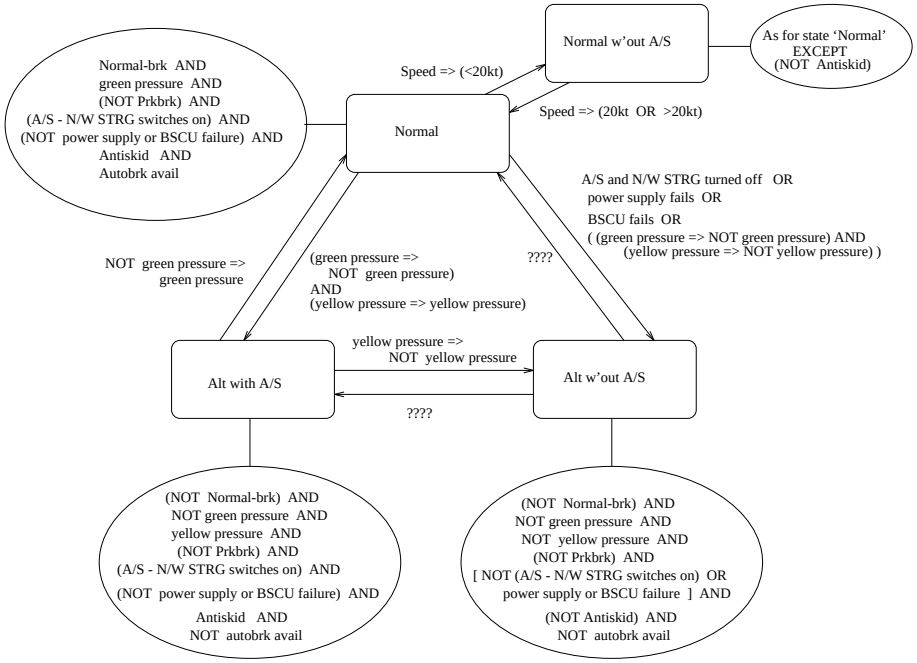

Fig. 6. The Revised Braking Modes

- note those transitions between pairs of states that cannot occur, and remove these transition arrows.

6.1. Braking Modes

The revised predicate-action diagram for the braking modes, obtained using the above steps, is shown in Figure 6. It was easily and speedily obtained from the original figure using the diagram editor `xfig`. The antiskid braking feature was included, since this seemed the easiest way of incorporating the information. To save space and condense the information, we use the notation `As for` *other-state* `EXCEPT` *predicate-change* to indicate that the state information is the same as another specified state, except for the value of a certain state predicate or predicates which have changed value to that given in the expression *predicate-change*.

The actions have been disambiguated in the revised diagram, so that they are more clearly actions rather than predicates. The symbol '`=>`' is used as a binary connective between state descriptions, to give a resulting expression which is an action description. Thus, state predicates joined by '`=>`' are unambiguously action descriptions.

The state description associated with state *Normal* shows no value for the Boolean variable *yellow pressure* which denotes that there is sufficient pressure in the yellow braking hydraulic system. This is because in this state the variable *yellow pressure* may take either value, given the values of all the other state predicates. We surmised this information, given the description of the green system as the normal system and the yellow system as the alternate. If green pressure is sufficient, as it is in the state *Normal*, then it is reasonable to surmise that the

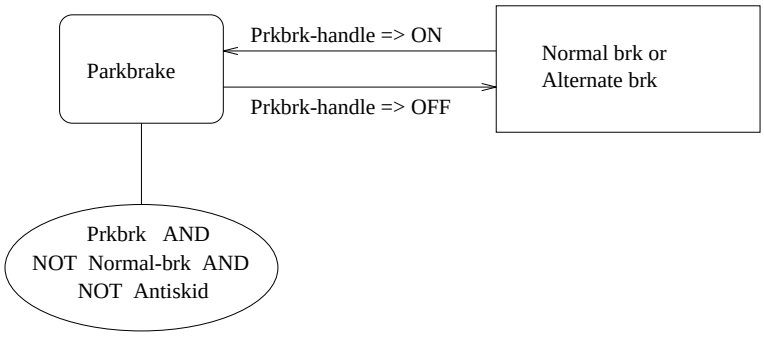

Fig. 7. The Revised Parking Brake Mode

state of the alternate braking system is not immediately relevant. We note that this is a supposition, not an inference, and thus may be incorrect. Nevertheless, a complete diagram must state what the possible values of all state predicates may be in this state.

We have also surmised that for braking purposes, the `A/S` and `N/W STRG` switches are operated in tandem. The original description considers only states in which they have the same values. A predicate-action diagram which considers what happens when these switches are not operated in tandem may be more complex.

We conclude that there are, in fact, five braking modes (not four, as declared in the Manual), since the antiskid is logically *tightly coupled* with the other information (*coupling* is the inverse of modularity, and the term is introduced and discussed in the context of system safety in [Per84]). In the original diagram there is an ambiguous transition out of state *Normal*. If *green pressure* transits from *normal* to *NOT normal*, then it is also necessary to note the status of the *yellow pressure* in order to determine whether the system transits into *Alternate with Antiskid* mode or into *Alternate w'out Antiskid* mode.

Finally, there are two transitions which could exist, but for which the actions are not easily determined from the original description. These actions have been denoted by labels '`????`' on the corresponding transition arrows. We can determine no obvious suppositions concerning these two transitions.

The revised parking-brake mode specification is shown in Figure 7. This mode is not tightly coupled with the others, as shown by the relatively small number of state predicates which must be defined in *Parkbrake* mode.

We have disambiguated the action required of the Parking Brake handle, in what we surmised was the most reasonable manner. Again, we may be wrong. However, in a complete specification, the actions that result in the system transiting into mode *Parkbrake* must be well-defined.

6.2. Ground Spoilers

Our intuition concerning the original description of the ground spoiler deployment suggested that the diagram would be easy to complete. The names of the states themselves in the original diagram correspond closely to the values of state predicates which distinguish the original states. The two state predicates which matter from the original diagram are *spoilers*

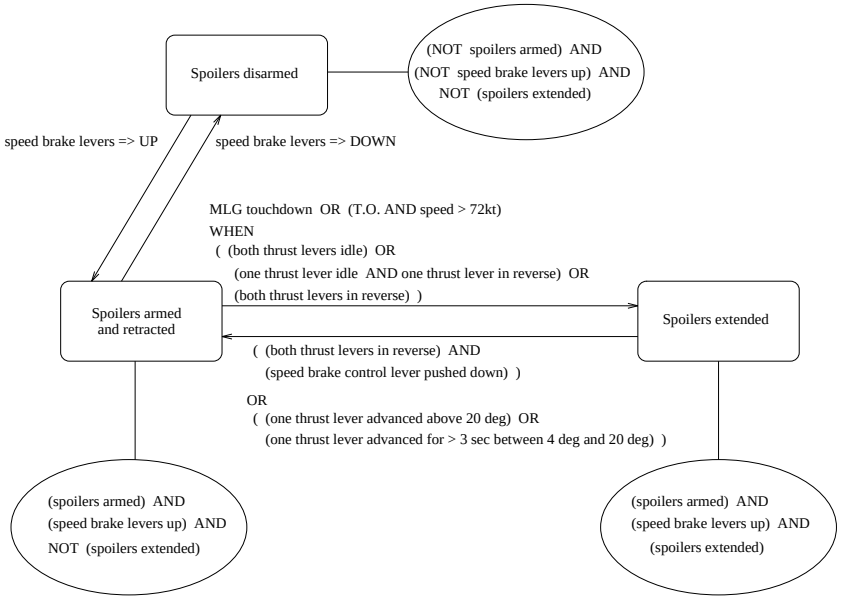

Fig. 8. Revised Ground Spoiler Deployment Diagram

armed and *spoilers extended*. Further, there is only one missing transition, from *Spoilers armed and retracted* to *Spoilers disarmed*, which we can surmise occurs when the speed brake levers are moved to the down position.

The specified transition of the speed brake levers is to the *UP* position, which the original description asserted caused the ground spoilers to become armed. This action is denoted again using the => notation in a slight different manner. This leads us to conclude that the position of the speed brake levers is important, hence we include the position as a state predicate, which forces us to define the position of the levers in all states. Defining the position of the levers in all states enables us to consider the two transitions missing from the original diagram. It is clear that in moving from a state in which the speed brake levers are *UP* to one in which the speed brake levers are down (i.e. not-*UP*, since the predicate is Boolean), the action must involve moving the speed brake levers from *UP* to *NOT UP*. Thus, we could introduce this action on the two transition arcs missing from the original diagram. However, we surmise that it is not intended that the spoilers can be extended while disarmed. Less obvious is the supposition that the spoilers cannot be disarmed while extended. However, we make both suppositions and do not include these transitions in the revised diagram. Notice that considering the features of the diagram, while attempting to complete it, led us specifically to ask the questions concerning direct transitions between disarmed and extended states, one of which is feasible and thus must be explicitly excluded (if our supposition is correct).

It is simple to indicate the values of the two relevant state predicates in the diagram, so we have done so explicitly. However, there is a technique which might aid in increasing readability in cases in which there are many state predicates, but in which there are logical connections between them. Consider the following example. Since we surmise that the spoilers cannot be extended unless they are armed, instead of including the state predicates explicitly, we might

wish to annotate the diagram with the Boolean expression `if` *(spoilers extended)* `then` *spoilers armed* and indicate only one Boolean value in the *Spoilers extended* and *Spoilers disarmed* states. This may be of value in cases in which the sets of state predicates become unwieldy but yet there are simple implications between them. Recalling that in fact a predicate-action diagram corresponds to a formula of logic, adding these implications as annotations to the diagram is defined to mean conjoining the logical formulas defining these implications to the formula defining the diagram. Thus, exactly the same logical information is contained in an annotated diagram as in a diagram with the values of all the state predicates explicit.

6.3. Revising the Reverse Thrust Deployment Description

We don't revise the description of the reverse thrust deployment, because (a) all the principles are already illustrated by the braking mode and ground spoiler deployment descriptions; and (b) we are not sure what the annotation concerning *reverse thrust idle set* is supposed to mean. If the pilots move the levers into the reverse thrust detente, then is it the case that the levers are moved automatically into the idle-reverse detente, and then must be moved again into full reverse-thrust to give reverse thrust? Or maybe there is an interlock which prevents the levers from being moved from forward thrust or forward-idle into any other reverse thrust detente than reverse-idle, or maybe something else is meant. Since this cannot be determined from the original description, it is clear that this will affect how the predicate-action diagram is modified. It may introduce more states, for example. We leave this resolution to the reader.

7. Further Analysis

We have revised the predicate-action diagrams for the original description, to fulfil certain criteria of completeness. Economy of expression leads us to decouple diagrams and thus specification where we can. The series of predicate-action diagrams we obtained are easy to understand, given the criterion of completeness, and may form the basis of a formally correct Flight Crew Operations Manual, that will be at least as understandable as the original version, and probably more so, given that infelicities have been ironed out. But is this all there is to analysing such a description?

7.1. Deeper Logical Dependencies

We have noted in Section 3 that some expressions refer to state predicates holding over some explicit period of time. For example, the assertion that a change in the ground spoiler state may occur if one thrust lever is advanced to between 4° and 20° for 3 seconds or longer. The simple version of a temporal logic semantics we have used, Boolean logic with state-changes, is insufficient to express the logical relations between this expression and other expressions of the positions of the thrust levers. For example, if a thrust lever is advanced to between 4° and 20° for a certain period of time, it cannot also at that time be in a position of less than 4° or greater than 20°. This follows by simple axioms for the data domain over which the variable *thrust-lever-position* ranges. But unless we can express these values and their exclusion, then

Variable name	Values
antiskid-mode	activated, deactivated
autobrake-mode	disarmed, armed, active
braking-on?	Boolean
braking-system-failure?	Boolean
sensed-skid-onset?	Boolean
anti-skid-switch-position	ON, OFF
main-gear-tachometer-left	real-number
main-gear-tachometer-right	real-number
sensed-reference-speed	real-number
brake-release-order-in-effect	Boolean
wheel-slip-value	real-number
green-hydraulic-pressure	available, insufficient
yellow-hydraulic-pressure	available, ??
antiskid-power-electric?	Boolean
autobrake-arming	lo, med, max
speedbrake-control-lever-position	armed, disarmed
sensed-MLG-touch-down?	Boolean
takeoff-run?	Boolean
speed->-72-kt?	Boolean
ground-spoilers-state	armed, extended, in-transit, retracted
thrust-lever-positions	$\{1,2\} \times$ { reverse, idle, $< 4°$, $[4°, 20°]$, $> 20°$ }
number-rudder-pedals-depressed	0,1,2
in-flight?	Boolean
speed-brakes-control-lever-position	up-armed, down-disarmed
sensed-speed	$<$ 20-kts, $[20kts, 72kts]$, $>$ 72-kts
A/SKID	ON, not-ON
N/W-STRG	ON, not-ON
PARKING-BRAKE	ON, not-ON
power-supply-failure?	Boolean
BSCU-failure?	Boolean

Table 1. The variables used in the actuation logic

the logical relations of the assertion with other statements of thrust lever position cannot be made explicit. Nevertheless, these might be (and we would surmise, are) crucial to correct operation of the aircraft.

A full logical expression of the description contained in the FCOM thus must include some logical means of handling assertions of thrust lever position and temporal duration. This can be done straightforwardly in TLA [AL94]. In fact, by using TLA, we believe the approach we are suggesting here is extendable to all forms the specification may take, since TLA allows arbitrary mathematics to be included. However, we will not further justify this assertion here.

When moving from indivisible Boolean state predicates to formulas which assert the values of certain variables, it is first necessary to list all the potential variables. The list of variables referred to in the FCOM description appears in Table 1. Although it is beyond the scope of this paper to demonstrate a full analysis of the logic of the specification, including the temporal durational dependencies, we describe what must be done with the variables, since this leads to identification of the interaction between system states representing certain environmental variables, and those variables themselves. We believe that it is advisable for safety

Variable name	Values
sensed-skid-onset?	Boolean
sensed-reference-speed	real-number
wheel-slip-value	real-number
sensed-MLG-touch-down?	Boolean
takeoff-run?	Boolean
speed->-72-kt?	Boolean
in-flight?	Boolean
sensed-speed	$<$ 20-kts, $[20kts, 72kts]$, $>$ 72-kts

Table 2. Internal variables corresponding to environmental situations

reasons for pilots and engineers to understand the potential sources of discrepancy between the environmental conditions, and the internal system states which are supposed faithfully to reflect those conditions. As we have noted with regard to *wheel speed upon impact*, such discrepancies have been noted in the Warsaw crash [FI.93c]. Amongst other things, the main landing gear strut sensors did not sense compression, and the wheels did not spin up, even though the airplane was in the process of landing on the runway. It is fairly straightforward explicitly to identify such variables. We show how to do so here by example.

7.2. Listing the Variables

Firstly, all the variables mentioned in the FCOM description should be listed, along with the types of values they can have, which may be determined from the description. These variables determine the possible states, given by their combinations of values. This may be a larger set than is actually needed, since there may be obvious logical dependencies between some variables. This list may then be surveyed to determine variables which are internal representations of environment variables. Variables which correspond to environmental properties are listed in Table 2 and a reduced set in which variables which are restatements of values of others variables, and thus logically dependent on them, are listed in Table 3.

The three steps involved in analysing the variables are:

- List all variables appearing in the description being considered;
- identify variables which are representations of environmental conditions, and for each such variable *[variable-name]*, add a new variable named *sensed-[variable-name]*;
- identify logical dependencies amongst the variables listed, and reduce the set of variables by elimating those whose values may be expressed as values of others (the choice of which variables to eliminate and which one to retain is arbitrary, subject only to the condition that the eliminated variables may be defined in terms of the retained ones).

8. Conclusions

We have suggested that the description in the Flight Crew Operating Manual of a complex fly-by-wire aircraft such as the A320 may be considered as a high-level system specification

Variable name	Values
sensed-skid-onset?	Boolean
sensed-reference-speed	real-number
wheel-slip-value	real-number
sensed-MLG-touch-down?	Boolean
takeoff-run?	Boolean
in-flight?	Boolean

Table 3. Reduced Set of Internal variables corresponding to environmental situations

according to the principles of hierarchical system specification and analysis. We have shown how predicate-action diagrams, a user-friendly technique based on rigorous logical methods, may be used to express and finally to analyse the specification for completeness and for coupling. We have noted that the resulting specification, expressed as predicate-action diagrams, is likely to be more readable to the flight crew, since it is complete, unambiguous and pictorial, as well as yielding a rigorous basis for the high-level specification of the aircraft systems. Such an approach could be incorporated in standards [RTCA92] for future transport aircraft systems development. Finally, we have indicated further analysis of the variables in the description in order to determine logical relations that are not expressed by simple Boolean state predicates, and have suggested without argument that TLA is also appropriate for expressing these further logical and physical relations.

References

[AL94] M. Abadi and L. Lamport. An old-fashioned recipe for real time. *ACM Transactions on Programming Languages and Systems*, 16(5):1543–1571, Sep 1994.

[Anon94] An A320 Captain, 1994. Personal Communication.

[BA93] G. Bruns and S. Anderson. Validating safety models with fault trees. In Janusz Gòrski, editor, *Proceedings of the 12th International Conference on Computer Safety, Reliability, and Security*, pages 21–30. Springer-Verlag, 1993.

[BF93] R.W. Butler and G.B. Finelli. The infeasibility of quantifying the reliability of life-critical real-time software. *IEEE Transactions on Software Engineering*, 19(1):3–12, January 1993.

[Dij68] E.J. Dijkstra. The structure of the THE multiprogramming system. *Communications of the ACM*, 11(5), May 1968.

[DM93] R. Dorsett and P. Mellor. The Airbus A320 electrical flight control system. Draft Version 0.14, January 1993.

[Eco94] Air crashes: But surely ... *The Economist*, 331(7866):92–93, June 4th - 10th 1994.

[Eng94] U. Engberg. *TLP, an automated prover for TLA*. PhD thesis, University of Aarhus, Aarhus, Denmark, 1994.

[FAR] Department of Transport, Federal Aviation Administration, Washington, D.C. *Federal Aviation Regulations*. Continually updated.

[FB92] J.H. Fielder and D. Birsch. *The DC-10 Case: A Study in Applied Ethics, Technology and Society*. State University of New York Press, 1992.

[FCOM] Airbus Industrie, Toulouse-Blagnac, France. *A320 Flight Crew Operating Manual*. Pages reproduced in [Inv94].

[FI.93a] Rain factor in loss of Lufthansa A320. *Flight International*, 144(4388):14, 22 - 28 September 1993.

[FI.93b] Aquaplaning 'an A320 crash factor'. *Flight International*, 144(4389):15, 29 September- 5 October 1993.

[FI.93c] Actuation delay was crucial at Warsaw. *Flight International*, 144(4391):10, 13 - 19 October 1993.

[FI.93d] Early Warsaw result provokes questions. *Flight International*, 144(4394):14, 3 - 9 November 1993. News report by A. Jeziorski.

[FI.93e] Warsaw over-run was preventable. *Flight International*, 144(4399):8, 8 - 14 December 1993.

[GH94] H. Garavel and R-P. Hautbois. An experience with the LOTOS formal description technique on the Flight Warning Computer of Airbus 330/340 Aircraft. In *Proceedings of the First AMAST International Workshop on Real-Time Systems*, Workshops on Computing. Springer-Verlag, 1994. Conference held in Iowa City, IA, November 1993.

[HK81] E.J. Henley and H. Kumamoto. *Reliability Engineering and Risk Assessment.* Prentice-Hall, 1981.

[HLN+90] D. Harel, H. Lachover, A. Naamad, A. Pnueli, M. Politi, R. Sherman, A. Shtull-Trauring, and M. Trakhtenbrot. STATEMATE: A working environment for the development of complex reactive systems. *IEEE Transactions on Software Engineering*, 16(4):403–414, APril 1990.

[Inv94] Main Commission Aircraft Accident Investigation. Report on the accident to Airbus A320-211 aircraft in Warsaw on 14 September 1993. Warsaw, March 1994.

[Lam94a] L. Lamport. The Temporal Logic of Actions. *ACM Transactions on Programming Languages and Systems*, 16(3):872–923, May 1994.

[Lam94b] L. Lamport. TLA in pictures. In `http://www.research.digital.com/SRC/tla/`, 1994.

[Lev86] N.G. Leveson. Software safety: Why, what and how. *Computing Surveys*, 18(2):125–163, June 1986.

[Lev91] N.G. Leveson. Software safety in embedded computer systems. *Communications of the ACM*, 34(2):34–46, February 1991.

[LMS85] L. Lamport and P.M. Melliar-Smith. Synchronizing clocks in the presence of faults. *Journal of the ACM*, 32(1):52–78, January 1985.

[LS87] N.G. Leveson and J.L. Stolzy. Safety analysis using Petri Nets. *IEEE Transactions on Software Engineering*, 13(3):386–397, March 1987.

[LS92] B. Littlewood and L. Strigini. The risks of software. *Scientific American*, pages 62–75, November 1992.

[LS93] B. Littlewood and L. Strigini. Validation of ultrahigh dependability for software-based systems. *Communications of the ACM*, 36(11):69–80, November 1993.

[LSP82] L. Lamport, R. Shostak, and M. Pease. The Byzantine Generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, July 1982.

[LT82] E. Lloyd and W. Tye. *Systematic Safety.* Civil Aviation Authority, London, 1982.

[LT93] N.G. Leveson and C.S. Turner. An investigation of the Therac-25 accidents. *IEEE Computer*, 26(7):18–41, July 1993.

[Mel94] P. Mellor. CAD: Computer-Aided Disaster! *High Integrity Systems*, 1(2), 1994.

[MP92] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer-Verlag, 1992.

[NBF+80] P.G. Neumann, R.S. Boyer, R.J. Feiertag, K.N. Levitt, and L. Robinson. A provably secure operating system: The system, its applications, and proofs. Technical Report CSL-116, SRI International, Menlo Park, CA, 1980. 2nd. edition.

[Neu86] P.G. Neumann. On hierarchical design of computer systems for critical applications. *IEEE Transactions on Software Engineering*, SE-12(9):905–920, September 1986.

[Par72] D.L. Parnas. On the criteria to be used in decomposing systems. *Communications of the ACM*, 15(12), December 1972.

[Par74] D.L. Parnas. On a 'buzzword': Hierarchical structure. In *Information Processing 1974 (Proc. IFIP Congress)*, pages 336–339. 1974.

[Per84] C. Perrow. *Normal Accidents: living with high-risk technologies.* Basic Books, New York, 1984.

[Pot93] J.P. Potocki de Montalk. Computer software in civil aircraft. *Microprocessors and Microsystems*, 17(1):17–23, 1993.

[PSL80] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, April 1980.

[PvK90] D. Parnas, A.J. van Schouwen, and S.P. Kwan. Evaluation of safety-critical software. *Commu-*

nications of the ACM, 33(6):636–648, June 1990.

[RTCA92] Radio and Technical Commission for Aeronautics, Washington, D.C. *DO-178B: Software Considerations in Airborne Systems and Equipment Certification*, December 1992. This document is known as EUROCAE ED-12B in Europe.

[Rus93] J.M. Rushby. Formal methods and the certification of critical systems. Technical Report CSL-93-7, SRI International Computer Science Laboratory, Menlo Park, CA, December 1993.

[Rv93] J.M. Rushby and F. von Henke. Formal verification of algorithms for critical systems. *IEEE Transactions on Software Engineering*, 19(1):13–23, January 1993.

[Sch87] F.B. Schneider. Understanding protocols for Byzantine clock synchronization. Technical Report 87-859, Cornell University Department of Computer Science, Ithaca, NY, August 1987.

[Sha92] N. Shankar. Mechanical verification of a generalized protocol for Byzantine fault-tolerant clock synchronization. In *[Vyt92]*. 1992.

[Sin94] Amanda J. Sinton. A safety analysis of the Airbus A320 braking system design. Master's thesis, University of Stirling Department of Computing Science and Mathematics, Stirling, Scotland, April 1994.

[Tho94a] Muffy Thomas. A proof of incorrectness using the LP Theorem Prover: the editing problem in the Therac-25. *High Integrity Systems*, 1(1):35–48, 1994.

[Tho94b] Muffy Thomas. The story of the Therac-25 in LOTOS. *High Integrity Systems*, 1(1):3–15, 1994.

[Vyt92] J. Vytopil, editor. *Formal Techniques in Real-Time and Fault-Tolerant Systems*, volume 571 of *Lecture Notes in Computer Science*. Springer-Verlag, 1992.

[Wen78] J. Wensley *et al.* SIFT: Design and analysis of a fault-tolerant computer for aircraft control. *Proceedings of the IEEE*, 66(10):1240–1255, October 1978.

[WGNH81] W.E.Vesely, F.F. Goldberg, N.H.Roberts, and D.F. Haasl. *Fault Tree Handbook*, volume NUREG-0492. U.S. Nuclear Regulatory Commission, Washington, D.C., January 1981.