工業製品と組込み Software の形式的要求仕様 —暦時計と自動航空 System を題材として—

Formal Method of Requirement Specification for Industrial Products with Embedded Software —As Applied to the Test Cases of a Calendar-Clock and an Automatic Flight System—

柳 生 孝 昭

要 約 暦時計と自動航空 System を題材として，要求仕様記述の形式的方法を工業製品の組込み software に適用する際の，特徴的な問題を考察する．記述のための言語は，部分的に解釈された 1 階述語論理を用いた．工業製品の振舞いは一般に，状態遷移の因果的連鎖として捉えられ，組込み software は言わば人工的な因果法則の役割を果す．従って状態遷移という動的な側面の，（通常，静的側面にのみ有効と見なされている）述語論理による記述と，古来難題とされて来た因果関係の定式化が，問題の核心に在る．本稿は適切な状態空間と，その上での 2 項または 3 項述語の定義，及び因果法則を現わす公理の設定によって，これらの問題の解決を試みた．更に，完成した仕様の分析を通して，因果連鎖の並列性や feedback の存在を顕にし得ることを示した．これらは仕様の実現（詳細設計）を考案する，また予め製品の異常な振舞いを見出すために，有効であろう．

Abstract Problems characteristic of applying formal method of requirement specification to industrial products with embedded software as seen in the test cases of a calendar-clock and an automatic flight system are studied. Partially interpreted 1 st order predicate logic serves as the specification language. Behaviors of industrial products are generally recognized as causal chains of state transitions where embedded software plays so to speak the role of artificial law of causality. Hence the central problems are how to describe the dynamism of state transitions in predicate logic (usually considered as suitable only for static aspects) and rigorously formulate the old notorious concept of causality. We have tried to solve these problems by defining an appropriate state space and binary or ternary predicates therein and postulating a set of axioms to represent cause-effect relations. We have also shown the possibility of detecting parallelism and feedback of causal chains through the analysis of completed specification, which may effectively serve for realizing the specification (detailed designing) and predicting undesirable behaviors of product.

およそ考え得るものは，悉く明晰に考え得る．
言い表し得るものは，悉く明晰に言い表し得る．
— L. Wittgenstein［ヴィト］

0. 背景と目的

［0］「設計ミス」とは何か？

今日我々の生活は無数の人工物，特に工業製品によって支えられているが，それらは時に，作り手も予期しないような異常な振舞いによって我々を驚かし，更には危険な状況に陥れる．携帯電話機の欠陥や人工衛星の作動不全は，製作者に莫大な金銭的損失を蒙らせ，原子炉の損傷や航空機の自動制御装置の暴走は，社会に深刻な害を及ぼす．事故や異常が起きる度に，世間は判で押したように「設計ミス」と言うが，これはどのような意味の言葉なのだろうか？形・寸法・諸属性を定め，部品を選んで配置し結合し，図面に表すという類の作業を，仮に

「設計」と呼ぶとして，その作業に間違いが有った，という意味なのか？　とすれば「設計」の正誤を判定する条件が，「設計」とは独立に，前以って示されている筈である．だが筆者の知る限り，そういう「条件」や「設計」との乖離は，およそ論じられることが無い．また「条件」が明示されていたとしても，「設計」はその「条件」に照らして正しいが，実は「条件」の方が適切ではなかったという，「設計」の正誤とは全く違う次元の問題が有り得る．しかしこの問題は，その存在すら認知されていないかのようである．「条件」が適切に定められ，「設計」がそれを満たす時にのみ，製品は正しく設計されたと言うべきだが，とすれば所謂「設計ミス」は，「条件」と「設計」の少なくとも一方が誤っていた時に生じ，従って三種に類別されなければならない．この別を曖昧にして置く限り，我々は，工業製品の持つ顕在的・潜在的危険性に正対しているとは，言えないであろう．

［1］　製品の複雑・巨大化に伴う「Babel」の塔的状況

複雑・巨大な工業製品の設計・製造・操作には，多くの異なる領域が関わり，それぞれに高度の専門性が求められるので，一人の設計者が全体に精通していることは，殆ど期待できない．これらの領域は言語を異にし，専門家の間の合理的対話は難しく，しばしば露にされるのは，或る領域に於いて常識とされる事柄が，別の領域の専門家によって気付かれもせずに見過ごされる，という事実である．近年の例で言えば高速増殖炉「もんじゅ」の温度計の破断（95.12），名古屋空港に於ける中華航空機の墜落（96.4），人工衛星「みどり」の太陽電池 paddle の破断（97.6），東海村の核燃料加工施設に於ける臨界事故（99.9）等には全て，この種の事実が認められる［柳生 2］．このような「Babel」の塔的状況が，工業製品を益々不透明・不可解なものにし，その危険性を深刻化させている．

［2］　形式的要求仕様の必要性

人の被造物である工業製品を，多様な領域の設計者や一般市民に取って理解可能・制御可能なものとするために，「設計」に先立つ「条件」の，領域中立的な言語による明示が求められていることは，明らかである．「条件の明示」とは詳しくは「開発者と使用者が合意する，当該成果物が満たすべき条件の，必要・十分，且つ正確な記述」であり，それを「要求仕様」または簡略に「仕様」と呼ぶ．［0］項に「設計」と書いたものは，仕様の（言語的）実現に他ならない．そこで仕様とその実現を併せたものを，改めて「設計」の定義としよう．

領域中立的言語による正確な記述という要求には，二律背反の嫌いが有る．正確を期すためには日常的語法を排し，専門用語に訴えざるを得ない，というのが通念だからである．だが正確と専門性は違う．科学，従ってそれに基づく工学・技術の知見は原理上，観測事実と数学的概念によって記述され，後者は究極的には，人の直接的な感覚与件と集合の概念に還元される．これらが全ての領域に共通な語彙の内であることは，言うまでもない．例えば熱伝導と金融派生商品の法則が共に，放物型の偏微分方程式によって記述されることを思い起そう．実務的には CAD の環境を前提とすれば，製品 model である database の schema は仕様に他ならないが，それを記述する言語（Data Description Language）は，汎用言語の一種として，領域中立的なのである．

自然言語に付き纏う多義性・曖昧さを払拭するには，形式的言語に依らざるを得ない．形式的言語とはどのような言語か，一般的な特徴付けと自然言語との違いについては，別稿

［柳生 2］に論じたので，本稿では触れない．後述のように，例示のためには，始めから特定の言語を決めて掛かる．形式的言語によって記述された仕様が即ち，形式的要求仕様である．

［3］ 抽象性，及び仮言命題について

仕様の影が薄ければ，製品の何たるかを知るためには，実現（図面や設計図書）に頼らざるを得ない．然るに後者は使用者に取っては過度に詳細な情報を含み，また普通は結果のみが示され，その前提は明らかにされないので，前提の認識に過ちが有っても，見過ごされる危険が大きい．実際，前掲の「もんじゅ」や「みどり」の場合はそうであった．従って仕様は，抽象的である（開発者のみに関わり，使用者は関知しない詳細を捨象する）と同時に，仮言命題の形の要求をも含むものでなければならない．これらの点に照らしても，仕様記述は実現とは全く次元を異にする営みなのである．抽象的な仕様記述のためには，言語もまた抽象的であるべきことは，言うまでもないであろう．

［4］ 組込み Software の特権的重要性

工業製品に組込まれた制御 software は上の事情に照らして，次のように三重の意味で，特権的な重要性を持つ．第 1 に抽象的・形式的仕様の必要性は，情報科学の分野では早くから指摘され，理論と実践の両面で豊かな蓄積が有ること，第 2 にしかしながらその関心は専ら，予め非物質化・記号化されている対象に向けられており，物質的存在である工業製品は埒外に在ること，第 3 に計算機制御の製品の設計には，機器の技術と情報技術という，少なくとも二種の専門領域が関わり，複雑・巨大な製品の場合には更に，操作もまた独自の領域の専門性を求めること，である．

以上の背景の下に本稿の目的は，工業製品と組込み　software の形式的仕様記述に於ける特徴的な問題を考察し，解決の方法を提案すると共に，手頃な題材への適用を通して，その有効性を確認することに在る．

1.　主要な問題，及び言語的枠組

1.0　問題の根源

物質的存在として，工業製品の属性や振舞いは，有限の記述によっては尽くし得ない．特に個物を，一群の固定的な属性の組によって特徴付けられる，一つの型（type）の実例（instance）として同定することは，視点（view）を定めて始めて可能である．視点が異なれば型や属性は全く違うし，個物が同一性を失うことにさえなりかねない．逆に個物は同一性を保ちつつ，型や属性を変えることができる．また全体を部分に分け，それを繰り返して，階層的に構造化するのが普通であるが，その仕方も視点に依存する．それぞれの視点の下での記述は，それぞれの model を形成し，それらが物理的に同一の対象の model であることは，それらの間の何らかの，それぞれの記述と整合する対応によってのみ，保証される．全ての model を導出し得る，単一の普遍的な model は，恐らく存在しない．

これも自明のことではあるが，工業製品の物理的な振舞いは，自然法則に従う．通念によれば，科学は自然現象に因果的法則性を認め，自然界の因果的な状態変化を，数学的に定式化し

た．従って工業製品の振舞いを科学的に捉えるには，それを因果的連鎖として数学的に表すことが基本である．一方 software の働きは，純粋に記号的・数学的な性質のものである．更に組込み software の場合は，制御対象である物理的製品が従うべき条件を定める，言わば人工的な因果法則として機能する．従って工業製品と組込み software の両者を共に数学的言語によって記述するのは，自然なことと思われるかも知れない．しかし事はそう単純ではない．数学は本質的に，対象の状態が如何なる条件の下に，如何なる原因によって，如何に変化して行くかというような動的な記述とは，相性が良くないのである．

1.1　仕様記述の言語的枠組

言語は部分解釈された 1 階述語論理とする．従って仕様は，1 階形式理論の形を取る．「部分解釈された」とは，標準的な data 型や問題向きに定義される抽象 data 型を始め，P. Martin-Löf［M-L 1，2］の意味で構成可能な型の全体を A とし，仕様の記述に於いて A の要素・関数・述語の自由な使用を許す，ということである．物質的対象を表すために，純粋な記号の集合＝型 ι を用意する（ι は identifier，識別子の集合の意）．本稿では詳細を省くが，対象の識別子は，仕様を満たす実現を構成する段階で，一つの対象を創成する度に，database 管理系によって自動的に与えられる［Yag 1］．それ自体は無意味な識別子による同定は，異なる視点を通して物質的対象の同一性を確保する，あるいは同一性を保ちつつ，自由に属性や他の対象との関係の追加・削除・変更を可能にするためである．

1.2　主要な問題

前二節から明らかに，問題の核心は論理的言語による状態，状態遷移，及び因果関係の定式化に在り，本稿はこれを主題とする．しかし下に述べるように，これらの概念の日常的な用法そのままを厳密化することは，難しい．

［0］「状態」を定式化する素朴な考えは，それを対象の属性と見なすこと，即ち状態値を対象に対応させる関数と見なすことである．しかしこの関数は，値（状態値）が変化し得るので，1 階述語論理で言う関数とは違う．では手続き型言語に於けるような変数として定義し，割当て文によって値を変えるというのはどうか．それは実現の手段としては良いかも知れないが，その変数が当の対象の状態を表すという記述を欠くならば，仕様とは言えず，そもそもこのような変数の概念も，1 階言語の枠内には収まらない．

［1］　状態値が時点ごとに定まることに注目し，状態を対象と時点の組の関数として定義する方法も，考えられる．状態遷移は，例えば時点 t の前後で状態値が変る

$$\mathrm{state}(\mathrm{obj}, t-0) = a \wedge \mathrm{state}(\mathrm{obj}, t+0) = b \wedge a \neq b$$

という形に，述べられるだろう．しかし t は，言わば Newton 的絶対時間の中の一点であり，工業製品の循環的な作動を目盛る時ではない．例えば日々の運転開始時点は後者に於いては特定の一点であり，しかもそこでの製品の状態は，一般に多価的なのである．それを絶対時刻の相異によって一価関数化しても，そのような関数は，我々が絶対時刻も，全ての時点での状態値も知り得ない以上，記述し得ない．

[2]　状態遷移図の矢印は，一つの状態から他の状態への遷移が許されることを示しており，それが可能であることや，まして実際に生じることを表してはいない．遷移の可能性は，自発性または何らかの原因の存在に依存する．前者は別の種類の矢印を，後者は因果関係の明示を必要とするが，何れも通常の状態遷移図に描かれることは無い．一方，前述の割当て文は，それが実行されるならば遷移の生起を，実行され得るならば可能性を表すが，遷移前の状態を示していないし，実行の可能性は，当の割り当て文のみを見ても分らない．

[3]　因果関係の厳密な定式化は，D. Hume 以来の難問である．詳しくは［柳生 3，5］に論じたように，通念に反して，近代科学は因果関係を捨象した．端的に言うと，科学法則の多くは等式の形を取っているが，その両辺の対称性が，原因と結果の非対称性を破壊するのである．また因果関係は二つの事象（原因と結果）の間の関係であるが，それを適切に捉える術が，1 階言語には無い．「事象」は命題，即ち論理式によって表現することになろうが，論理式の間の関係として唯一つ思い付きそうな含意（→）は，因果関係とは似て非なるものである．例えば［Bar 1］pp 62–4 も，その弊を免れていない．

2.　基礎概念の定式化

[0]　状態，状態変数，状態空間，状態関数，状態遷移

X を ι またはその直積の部分型とし，必要に応じ，幾つでも定義しておく．各 X に対し A の部分型 S と，X から 2^S(S の冪集合)への写像 state を定義し，それを「状態関数」，各 x ∈X に対応する state(x)を x の「状態空間」と呼ぶ．また state(x)は S の一つの部分型を定めると考え，その要素を x の「状態」，その型の変数を「状態変数」と呼ぶ．「状態遷移」は型 $\mathrm{state}(x)^2$ の上の 2 項述語 trans として定義する．

state(x)は対象 x が取り得る全ての状態から成る集合の心算である．それを値とする関数を定め，またそれを一つの型と考えることにより，対象と複数の状態の対応を，1 階言語の枠内で定式化するという問題（1.2[0]項）に応えようというのである．また述語 trans は，その状態遷移が許されるものであることのみを意味し（1.2[2]項），その可能性は，次々項に述べる因果関係によって保証される．

状態の時間への依存を明示的に扱うには，T を時間軸を表す型とし，state の定義域を X ×T とする．更に，各 x について T から state(x, T)への写像（時点の推移に伴う状態時系列）$\mathrm{instance}_x$ を選び，その集りを「世界線」と呼ぶ．当然，次の条件を要求する：

$$\mathrm{instance}_x(t) \in \mathrm{state}(x, t) \wedge \mathrm{trans}(\mathrm{instance}_x(t), \mathrm{instance}_x(t+\delta t)) \qquad (0)$$

δt は時間の刻みである．

[1]　事象

事象の生起は，何らかの対象の状態の変化によって，認識される．観測し得る限りでは何の状態変化も見られないということは，何の事象も生じていない，ということに等しいからである．逆に状態変化はそれ自体，一つの事象と見なし得る．そこで事象の概念と状態遷移のそれを同一視し，前項中段に述べたところに従って，trans と因果関係によってそれぞれ，事象生起の合法性と可能性を定式化する．

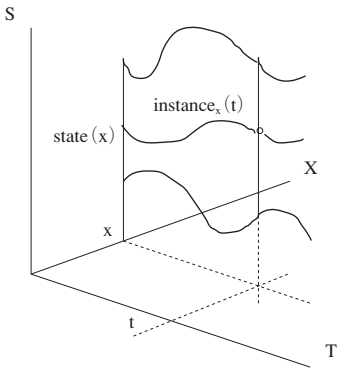


図 0　状態空間，及び状態時系列

［2］ 因果関係，状態遷移例

状態 s が t に，実際に遷移するという事象を仮に trans_a(s, t)と書く．2 階言語の記法を許すならば，因果関係は 1.2[3]項に述べたように，原因 trans_a(s_0, s_1)と結果 trans_a(s_1, s_2)の間の 2 項関係として表される．cause_effect(trans_a(s_0, s_1), trans_a(s_1, s_2))のように．この式を 1 階の述語

$$\text{cause\_effect}(s_0, s_1, s_2) \tag{1}$$

に書き換え，これを因果関係の形式的表現とする．次の公理の要請は，当然である．

$$\text{cause\_effect}(s_0, s_1, s_2) \rightarrow \text{trans}(s_0, s_1) \wedge \text{trans}(s_1, s_2) \tag{2}$$

原因事象の第 2 引数と結果事象の第 1 引数が同じであるのは，原因が生じてから結果を引き起すまでの時間を，捨象しているからである．その時間を明示するには，次のように考える．先ず時点 T に於ける s_0 から s_1 への遷移が，時間 t の後に s_1 から s_2 への遷移を引き起すということは，T〜T＋t を通して状態 s_1 が変らないことを前提とする，という点に注意する．これは因果性の概念が T〜T＋t の間の世界線，一般には部分状態空間に相対的であることを意味する．そこで部分状態空間 R に於ける因果関係を

$$\text{cause\_effect}(s, s', s''; R) \tag{3}$$

と記すと，時間 t を隔てた因果関係は，次の論理式によって表される：

$$\text{cause\_effect}((s_1, T-\delta t), (s_1, T), (s_2, T); \{(s, u) : T \leqq u < T+t \rightarrow s = s_1\}) \tag{4}$$

これを

$$\text{cause\_effect}(s_1[T, T+t), s_2) \tag{5}$$

と略記する．原因無しに生じ得る遷移を self(_, _)と書くと，特に重要な R である可能な状態遷移列は，次のように定義される：

$$\begin{aligned}\text{chain}(x) \equiv \{\sigma \in \text{state}(x)^N : \forall n \in N(&\text{self}(\sigma(n), \sigma(n+1)) \vee \\ &\text{cause\_effect}(\sigma(n-1), \sigma(n), \sigma(n+1); \sigma([0, n]))))\}\end{aligned} \tag{6}$$

相対的な因果関係の別の例として，遷移の循環を排除する場合は，

$$\text{cause\_effect}(s_0, s_1, s_2; \{s : \forall \sigma : \text{chain}(x)(s_2 = \sigma(0) \wedge s_1 \in \sigma(N) \rightarrow s \notin \sigma(N))\}) \tag{7}$$

可能な遷移とは，何れかの遷移列に項として現れるものである．即ち：

$$\text{trans\_p}(s, t; R) \equiv \exists \sigma : \text{chain}(x), n(s = \sigma(n) \wedge t = \sigma(n+1)) \tag{8}$$

次に，実際に生起する事象の列は明らかに，可能な状態遷移列を成す．逆に全ての可能な遷移列は，実際に生起する列と，存在資格に於いて同等と考えられる．つまり「実際の生起」

は一つの可能な遷移列の選択を含意し，個々の事象の生起は，遷移列に相対的な概念である．そこで「仮の」述語 trans_a を改めて，次のように定義する：

$$\mathrm{trans\_a}(s, t; \sigma) \equiv \sigma \in \mathrm{chain}(x) \wedge \exists n (s = \sigma(n) \wedge t = \sigma(n+1)) \tag{9}$$

明らかに，上の諸述語の間に下の関係が成立つ：

$$\mathrm{trans\_a}(s, t; \sigma) \rightarrow \mathrm{trans\_p}(s, t) \qquad \mathrm{trans\_p}(s, t) \rightarrow \mathrm{trans}(s, t)$$
$$\mathrm{cause\_effect}(s_0, s_1, s_2; \sigma) \wedge \mathrm{trans\_a}(s_0, s_1; \sigma) \rightarrow \mathrm{trans\_a}(s_1, s_2; \sigma) \tag{10}$$
$$\mathrm{cause\_effect}(s_0, s_1, s_2; \sigma) \wedge \mathrm{trans\_p}(s_0, s_1) \rightarrow \mathrm{trans\_p}(s_1, s_2)$$

これまでは簡単のために，議論を殆ど単一の対象の場合に限った．組立て品は対象の組から成るので，その状態は各要素の状態を決定する．即ち，組の状態空間から各要素の状態空間への射影が存在する．しかしこれらの射影は単射でも全射でもない．組立て品の状態には要素の状態によって決定されないものが有り，逆に要素の状態の組が全て可能とは限らないからである．同様に組立て品の状態遷移も，要素の状態遷移を引起すが，それによって決定はされない．

3. 例 1：暦時計の要求仕様

前節の定式化の応用として卓上型の暦時計を取り上げ，その形式的要求仕様記述を試みる．製品の外観は図 1 のようである[*1]．

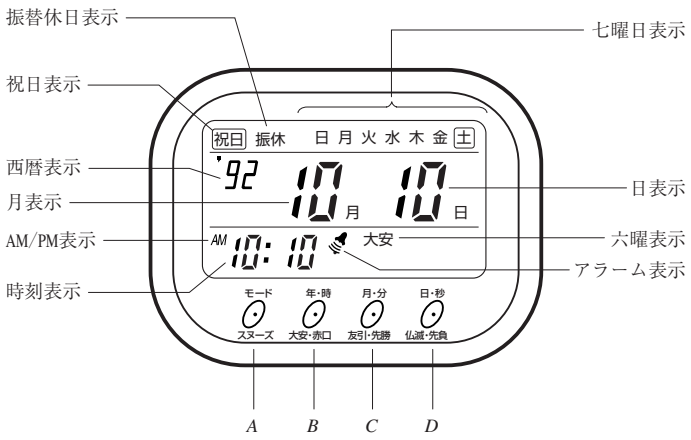


図 1　卓上暦時計の外観

3.0 記　法

概ね通常の論理・数学的な習慣に従うが，次の便法を用いる．

［0］ N mod n は巡回群 {0, 1, …, n − 1} を表す．

［1］ 一つの「型」の名によって，単項述語，またはその外延の何れをも指す．

［2］ 関数の宣言関数名（型 1）：型 2 は，型 1 が定義域，型 2 が値域であることを示す．

［3］ 述語名「遷移」,「因果」,「自発」,「可能」はそれぞれ，前節の trans, cause_effect, self, trans_p に対応する．

［4］ 関数または述語の宣言に続く｛｝内は，その関数または述語の解釈を定義する．
［5］ f*は関数 f の反復適用を表す．
［6］ 〈a｜b〉は a または b を指す．同一式中に複数個が現れる場合は，原則として同順のものを組合わせる．
［7］ s[a/b] は，状態変数 s の a–成分を b で置換えたもの，s[a/＋1] や s[a/後続] はそれぞれ，s[a/a(s)＋1] や s[a/後続(a(s))] の略記である．
［8］ [a/b, c/d] は，置換 a/b と c/d を同時に，[a/b；c/d] は，逐次的に行うことを表す．
［9］ 《 》内は読解を助けるための，非公式の注釈である．

制限：祝日，振替休日，及び六曜表示は省略した．また電池も装填されている状態のみを考えた．

3.1 要求仕様

理論 暦時計；
《この段落では属性や状態の値の型(1.1 節の A の部分型)と，そこでの関数や述語を定義する．》
型 分＝Nmod 60, 時＝｛‘*AM*’, ‘*PM*’｝×(1＋Nmod 12), 日＝1＋Nmod 31, 月＝1＋Nmod 12,
西暦＝(80＋Nmod 41) mod 100, 和暦＝1＋(54＋Nmod 41) mod 63, 年＝西暦∪和暦,
七曜＝｛‘月’, ‘火’, ‘水’, ‘木’, ‘金’, ‘土’, ‘日’｝,
明暗＝｛明, 暗, 点滅｝, 警報標識＝｛呼鈴印｝, 鳴黙＝｛鳴動, 一時停止, 黙止｝；
関数 西暦換算(和暦)：西暦｛西暦換算(y)＝(y≦32→y＋88)∧(y≧55→y＋25)｝；
述語 閏(年)｛閏(y)≡〈(西暦(y)→y)｜(和暦(y)→西暦換算(y))〉＝0 mod 4｝,
〈大｜小〉(月)｛〈大｜小〉(mo)≡mo∈〈｛1, 3, 5, 7, 8, 10, 12｝｜｛4, 6, 9, 11｝〉｝；
関数 月末日(年, 月)：日｛月末日(y, mo)＝((〈大｜小〉(mo)→〈31｜30〉)∧
(〈¬閏｜閏〉(y)∧mo＝2→〈28｜29〉))｝；
型 日付＝｛(y, mo, d)∈年×月×日：d≦月末日(y, mo)｝, 時刻＝時×分, 日時＝日付×時刻；
関数 後続(日付)：日付 ｛後続(y, mo, d)＝〈d＜月末日(y, mo)→(y, mo, d＋1)｜
d＝月末日(y, mo)∧mo<12→(y, mo＋1, 1)｜
d＝月末日(y, mo)∧mo＝12→(y＋1, 1, 1)〉｝,
真正(年×月×日)：日付｛真正(y, mo, d)＝〈d≦月末日(y, mo)→(y, mo, d)｜
d＞月末日(y, mo)→(y, mo＋1, 1)〉｝
《日付設定の操作（*D* 1*）によって，例えば 2 月 30 日や 11 月 31 日のような日付を，設定し得る．これらを 3 月 1 日や 12 月 1 日に正す．公理 *A* 3 及び *E* を参照.》,
＋1 h(時)：時｛＋1 h(ap, h)＝(〈h≠11→ap｜h＝11∧ap＝〈‘*AM*’｜‘*PM*’〉→
〈‘*PM*’｜‘*AM*’〉〉, h＋1)｝,
後続(時刻)：時刻｛後続(ap, h, m)＝(m<59→(ap, h, m＋1)｜m＝59→(＋1 h(ap, h), 0)〉｝,
後続(日時)：日時｛後続(e, t)＝〈t≠(‘*PM*’, 11, 59)→(e, 後続(t))｜
t＝(‘*PM*’, 11, 59)→(後続(e), 後続(t))〉｝,
後続(七曜)：七曜｛後続(〈‘月’｜…｜‘日’〉)＝(‘火’｜…｜‘月’〉｝,
曜日(日付)：七曜｛曜日(e)＝〈e＝(0：西暦, 1, 1)→‘土’｜e＝後続(f)→後続(曜日(f))｜
和暦(e[年])→曜日(西暦換算(e[年]))〉｝；

《次の段落は型 ι の物質的対象，状態空間，時間軸，及び因果関係を定義する.》

型　表示部 = {〈暦法 | 日付 | 七曜 | 警報標識 | 時刻〉表示部}, 警報音 = {音}, 警報時刻 = 時刻,
釦 = {*A*, *B*, *C*, *D*}, 位置 = {押下, 開放},
半分刻 = Nmod 2《「秒」を初期化しない限り自動的に進む, 1/2 分の目盛り》,
微分刻 = N《釦を押し続け, 状態遷移が続いて生ずる時の, 各遷移の間隔を表す目盛》;

関数　表示(表示部) {表示(〈暦法 | 日付 | 七曜 | 警報標識 | 時刻〉表示部)
= 〈〈{‘’’} | 日付 | 七曜 | 警報標識〉× 明暗〉 |
時刻 × 明暗 × 警報時刻 × {‘*ALARM*’} × 明暗〉},
警報(警報音) : {鳴黙}《警報(音) = 鳴黙 = {鳴動, 一時停止, 黙止} であることに注意.》,
位置(釦) : {位置}《同じく位置(*A*) = … = 位置(*D*) = 位置 = {押下, 開放}》;

型　状態空間 = $\Pi_{P\in 表示部}$表示(P) × 警報(警報音) × 位置(*A*) × … × 位置(*D*) × 微分刻 × 半分刻 =
({‘’’} × {明, 暗, 点滅}) × … × (時刻 × {明, 暗, 点滅}) × (警報時刻 × {‘*ALARM*’} ×
{明, 暗, 点滅}) × {鳴動, 一時停止, 黙止} × {押下, 開放}^{4} × N × Nmod 2;

変数　s, s′, s″ : 状態空間, h, h′, h″ : 半分刻, δ : 状態遷移列;

述語　遷移(状態空間, 状態空間), 自発(状態空間, 状態空間),
因果(状態空間, 状態空間, 状態空間), 可能(状態空間, 状態空間)
{可能(s, s′) ≡ 自発(s, s′) ∨ ∃s″(可能(s″, s) ∧ 因果(s″, s, s′))},
〈鳴動 | 一時停止 | 黙止〉(状態空間) {〈鳴動 | 一時停止 | 黙止〉(s) ≡
(鳴黙(s) = 〈鳴動 | 一時停止 | 黙止〉)};

型　状態遷移列 = {δ∈状態空間^N : 自発(δ(0), δ(1)) ∧ ∀i(自発(δ(i), δ(i + 1)) ∨
因果(δ(i − 1), δ(i), δ(i + 1)))};

述語　定常(状態遷移列) {定常(δ) ≡ ∀s∈δ(¬((日付 ∨ 時刻)明暗(s) = 点滅 ∧
位置((*B*) ∨ (*C*) ∨ (*D*))(s) = 押下))
《「定常」とは，釦操作による強制的な日時の変更を含まない状態遷移列を言う.
要素名「(日付 ∨ 時刻) 明暗」は,「日付」または「時刻」の直後の「明暗」を指す.》;

公理　《第 1 群の公理は一般則と時間の進行に関わる. δ = {s_0, …, s_n}, 半分刻(〈s | s_i〉) = 〈h | h_i〉とする.》

G 0 : ∀s, s′, s″(因果(s, s′, s″) → 遷移(s, s′) ∧ 遷移(s′, s″)),

G 1 : ∀s, s′(可能(s, s′) ↔ 遷移(s, s′))
《生起の可能性の無い状態遷移は, 始めから除いて置く.》,

H : ∀δ(∀i($h_i = h_0$) ∨ ∃j(自発(h_j, $h_{j+1} = h_j + 1$) ∨ 因果(s_{j-1}, $s_j = s_{j-1}$[位置(*D*)/押下],
$s_{j+1} = s_j$[半分刻/ + 1])))
《「秒」を初期化しない限り，如何なる状態遷移列にも，時間の自発的進行が割り込む.》,

T 0 : ∀s(h = 1 → 因果(s, s[半分刻/0], s[半分刻/0, 日時/後続]))
《如何なる状態に於いても,「半分刻」が 0 に戻ると瞬時に,「日時」が 1 分進む.》,

T 1 : ∀δ(定常(δ) → ∀i, k($h_i = 0 \wedge s_{i+1} = s_i$[日時/後続] ∧ ∀j(i + 1 < j < i + k → ¬($h_j = 1 \wedge h_{j+1} = 0$))
→ 日時(s_{i+k}) = 日時(s_{i+1})))
《「定常」状態遷移列に於いては,「半分刻」が 0 になり「日時」が 1 分進むと，再び 0 に戻るまでは,「日時」は不変に保たれる.》;

公理《第2群は釦 A の押下による表示と‘Mode’の変化に関わる．Mode は釦押下の意味を定める．》

$A\,0$：$\forall s, s'$（((黙止∨一時停止(s))∧(日時明暗, 位置(A), …, (D))(s) = (明, 開放[4])$_0$∧
(黙止∨一時停止(s′))∧位置(A)(s′) = 押下$_1$→
因果(s, s′, s′[日付明暗, 時刻明暗/点滅, 暗]$_2$))

《「日時明暗」＝「明」，即ち日時が表示されている状態（「日時表示 Mode」）で釦 A を押すと，時刻の表示は消え，日付の表示のみが点滅する「日付設定 Mode」に移る．論理式 $\_\__0$ を「日時表示 Mode」，$\_\__1$ をⒶ，$\_\__2$ を「日付点滅」と略記．以下も同様．》，

$A\,1$：$\forall s, s'$（日付設定 Mode∧Ⓐ→因果(s, s′, 時刻点滅)），

$A\,2$：$\forall s, s'$（時刻設定 Mode∧Ⓐ→因果(s, s′, 警報点滅)），

$A\,3$：$\forall s, s'$（警報設定 Mode∧Ⓐ→因果(s, s′, 日時表示[日付/真正；七曜/曜日(日付)]))

《日時表示 Mode に移ると共に，日付が正され，七曜が改めて計算される．》；

公理《第3群は各 Mode（釦名の後の数字，0〜3）の下での，釦 B〜D の押下による状態遷移を定める．日付，時刻，及び警報設定 Mode に於いては，これらの釦が押されている間，微分刻は自発的に進み，その度に，日時の進行が引き起こされる．》

$B\,0\langle 0 \mid 1\rangle$：$\forall s, s'$（日時表示 Mode∧(暦法明暗, 年)(s) = (⟨明, 西暦 | 暗, 和暦⟩)∧Ⓑ
→因果(s, s′, s′[暦法明暗, 年/⟨暗, 西暦換算$^{-1}$ | 明, 西暦換算⟩]))，

$D\,0\langle 0 \mid 1\rangle$：$\forall s, s'$（日時表示 Mode∧Ⓓ
→因果(s, s′, s′[警報標識⟨表示 | 非表示⟩/⟨非表示 | 表示⟩]))

《「警報標識⟨表示 | 非表示⟩」は「警報標識明暗＝⟨明 | 暗⟩」の略記．》，

$BCD\,10$：$\forall s, s'$（(日付設定 Mode∧(Ⓑ∨Ⓒ∨Ⓓ))→因果(s, s′, s′[微分刻/＋1]))，

$BC\,20$：$\forall s, s'$（(時刻設定 Mode∧(Ⓑ∨Ⓒ))→因果(s, s′, s′[微分刻/＋1]))，

$BC\,30$：$\forall s, s'$（(警報設定 Mode∧(Ⓑ∨Ⓒ))→因果(s, s′, s′[微分刻/＋1, 警報標識/表示]))，

BCD_1：$\forall s$((日付設定 Mode∧(Ⓑ∨Ⓒ∨Ⓓ))∨((時刻∨警報)設定 Mode∧(Ⓑ∨Ⓒ))
→自発(s, s[微分刻/＋1]))

《$__$は上の□の約束と異なり，第1引数である s に於いて，釦 B，C または D が「押下」であることを示すが，釦の状態は第2引数も変らず，誤解の恐れも無いので，同じ略記を用いる．以下も同様．》，

$\langle B \mid C \mid D\rangle 1^*$：$\forall s, s'$（日付設定 Mode∧⟨Ⓑ | Ⓒ | Ⓓ⟩∧s′ ＝s[微分刻/＋1]
→因果(s, s′, s′[⟨年 | 月 | 日⟩/＋1；七曜/曜日(日付)]))

《「日」は年月に拘らず 31 まで進むが，「月末日」を越えると，七曜は未定義となることに注意．》，

$\langle B \mid C\rangle 2^*$：$\forall s, s'$（時刻設定 Mode∧⟨Ⓑ | Ⓒ⟩∧s′ ＝s[微分刻/＋1]
→因果(s, s′, s′[⟨時/＋1 h | 分/＋1⟩]))，

$D\,2$：$\forall s, s'$（時刻設定 Mode∧Ⓓ→因果(s, s′, s′[半分刻/0]))，

$\langle B \mid C\rangle 3^*$：$\forall s, s'$（警報設定 Mode∧⟨Ⓑ | Ⓒ⟩∧s′ ＝s[微分刻/＋1]
→因果(s, s′, s′[⟨警報時/＋1 h | 警報分/＋1⟩]))；

公理《第4群は，午前 12 時と共に日付を正しいものに改める状態遷移と，釦の押下と開放が

自発的に生じることを保証する，二つの公理から成る.》

E：∀s, s′ (時刻(s) = ('*PM*', 11, 59) ∧ s′ = s[時刻/後続]
→因果(s, s′, s′[日付/真正；七曜/曜日(日付)])),

$P\langle A \mid B \mid C \mid D\rangle$：∀s(位置($A, \cdots, D$)(s) = 開放[4]
→自発(s, s′ = s[位置($\langle A \mid B \mid C \mid D\rangle$)/押下])),

R：∀s(自発(s, s′ = s[位置($A, \cdots, D$)/開放[4]]))；

公理《第 5 群は目覚まし音の鳴動，停止に関わる操作と，それに伴う状態遷移を述べる.》

F 0：∀s, s′ (警報標識表示(s) ∧ s′ = s[時刻/警報時刻]→因果(s, s′, s′[鳴黙/鳴動])),

F 1：∀s(因果(鳴動(s)[警報時刻, 後続[2](警報時刻)), 黙止(s)))
《(5)の記法を援用した．正確には次のように書くべきである：
∀s, s′, s″(鳴動(s) ∧ 時刻(s′) = 後続(警報時刻(s)) ∧ s″ = s′[時刻/後続]
→因果(s′, s″, s″[鳴黙/黙止]；{t：∃δ(δ(0) = s ∧ δ(n) = s′ ∧ t∈δ[0, n] ∧
∀i∈[0, n](鳴動(δ(i))))})》,

FA 0：∀s, s′ (鳴動(s) ∧ s′ = s[位置(A)/押下]→因果(s, s′, s′[鳴黙/一時停止])),

FA 1：∀s(因果(一時停止(s)[警報時刻, 後続[3](警報時刻)), 鳴動(s)))
《F 1 と同じく，(5)の記法を援用した.》,

$F\langle B \mid C \mid D\rangle$：∀s, s′ ((鳴動∨一時停止)(s) ∧ s′ = s[位置($\langle B \mid C \mid D\rangle$)/押下]
→因果(s, s′, s′[鳴黙/黙止]))；

理論了

4.　例 1 の分析

4.0　自発的事象と原因の有る事象，または状態決定の独立性と従属性

自発的事象の生起を示す公理には，次の 4 種が有る：

H：∀δ(∀i($h_i = h_0$) ∨ ∃j(自発($h_j, h_{j+1} = h_j + 1$) ∨ 因果($s_{j-1}, s_j = s_{j-1}$[位置(D)/押下],
$s_{j+1} = s_j$[半分刻/ + 1]))),

BCD_1：∀s((日付設定 Mode ∧ (Ⓑ∨Ⓒ∨Ⓓ)) ∨ ((時刻∨警報)設定 Mode ∧ (Ⓑ∨Ⓒ))
→自発(s, s[微分刻/ + 1])),

$P\langle A \mid B \mid C \mid D\rangle$：∀s(位置($A, \cdots, D$)(s) = 開放[4]
→自発(s, s′ = s[位置($\langle A \mid B \mid C \mid D\rangle$)/押下])),

R：∀s(自発(s, s′ = s[位置($A, \cdots, D$)/開放[4]])).

ここに現れている状態の内で，釦の位置は独立に定まる．半分刻と微分刻も，それぞれ後述の 4.1[3]と[4]の場合を除き，独立に定まる．その他の状態は現在の状態と，上の自発的事象を原因とする状態遷移から決定される．

4.1　状態遷移の並行性と非決定性

同一状態に於いて，$P\langle A \mid B \mid C \mid D\rangle$ と R は排反的，BCD_1 と R も排反的，H は何れとも共存可能である．このことと 1 節を踏まえ，公理群を詳しく見ると，次のことが分かる：

［0］ 一つの状態遷移列は 6 個の並行的な遷移列，即ち半分刻の交替(H)，微分刻の進行(D)，釦操作の連鎖（O)，これらの三者から因果的に定まる，表示/操作 Mode の変化（M)，鳴動（A)，及び表示の推移（T）から成る．

［1］ H は[3]項の場合を除き，独立に進行する．半分刻が 1 から 0 に移ると，H から T に向けて信号が送られ，それを原因事象とし，*T* 0 に従う結果事象 e(＝(s, s′))が生じ，次の状態 s′が定まる．

［2］ O は完全に独立に進行する．釦操作は信号として M，T または A に送られ，それを原因とし，*A* 0〜*A* 3，*B* 00〜*C* 3*（但し *D* 2 を除く）または *F*〈*A* 0｜*B*｜*C*｜*D*〉の何れかに従う結果事象 e(＝(s, s′))が生じ，次の状態 s′が定まる．

［3］ 操作が時刻設定 Mode に於ける釦 D の押下の場合には，信号は H に送られ，*D* 2 に従って，半分刻が 0 に戻る．続いて[1]の事象が生じる可能性が有る．

［4］ 操作が日付設定 Mode に於ける釦 *B*，*C*，*D* の押下，または時刻・警報設定 Mode に於ける *B*，*C* の押下の場合には，信号は D に送られ，*BCD* 10〜*BC* 30 に従って微分刻が 1 増す．これらの Mode の何れかと操作が保持されていれば，*BCD*_1 に従って，微分刻は自発的に増して行く．

［5］ 微分刻が 1 増すと信号が T に送られ，*B* 1*〜*C* 3*（但し *D* 2 を除く）に従って，日付，時刻または警報時刻が後続の状態に移る．

［6］ T に於いては，[1]または[2]に続き，前提条件が満たされていれば，*E* による更なる遷移が生じる．

［7］ それと同じ遷移が，M に於ける遷移を原因として，*A* 3 によって生じ得る．

［8］ T に於ける遷移を原因として，前提条件が満たされていれば，A に於ける遷移が，*F* 0，*F* 1，*FA* 1 の何れかによって生じる．

［9］ 状態遷移の非決定性は，釦操作の非決定性（恣意性）をそのまま反映している．各々の状態遷移列は一つの可能世界を表す．その意味で，状態時空間は全ての可能世界を含む，単一の世界を形成する．

4.2 一 般 化

上の分析を一般化すると，状態空間 $\Sigma=\Pi_{i\in I}S_i$ の部分積群 $\Sigma_n=\Pi_{i\in In}S_i$ $(I=I_0+\cdots+I_N)$ への分解に応ずる，状態遷移列 σ：T→Σ の Σ_n への射影群 $\{\sigma_n$：T→$\Sigma_n\}$ $(\sigma=\Pi\sigma_n)$ は，次の条件を満たす時，並行列を構成すると考えてよい：

［0］ Σ_n は，無原因の事象の生起を許すという意味で，他の Σ_m から独立である，または

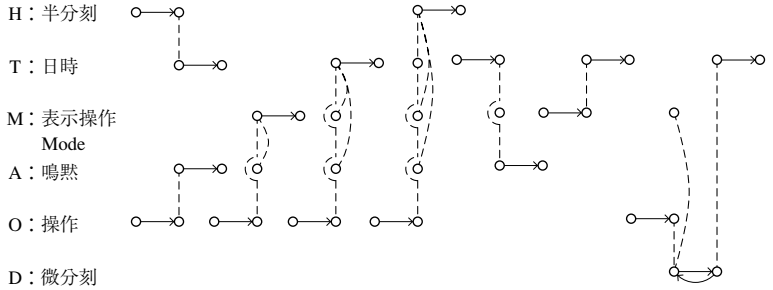


図 2　並行的な状態遷移列

［1］ Σ_n に於ける全ての事象は，他の幾つかの部分積 $\Pi_{p\in Pn}\Sigma_p$ に於ける事象を原因とするという意味で，従属的であるが，Σ_n が従属する，または従属される部分積の集合は，他の Σ_m とは異なる．

上の分析は，次のような仕様の実現を示唆する：並行的な状態遷移列の各々に対応する手続きを用意し，指定された前提条件の成立に応じて；

［a］ 自発的な状態遷移を生じさせる，または

［b］ 原因事象が生じているならば，結果の状態遷移を生じさせる，且つ

［c］ 状態遷移の生起を，それを原因として状態遷移が結果する可能性の有る手続きに，信号を送ることによって通知する．

5.　例 2：航空機の自動航空 System（Automatic Flight System,AFS）の要求仕様

次の題材は，1994 年 4 月 26 日名古屋空港に於いて着陸時に墜落した，中華航空公司 Airbus A 300 B 4–622 R 型 B 1816 機の自動航空 System である．例 1 と同様に，実際の製品の振舞いから出発して，それを説明する要求仕様を遡行的に推論するが，その第 1 段階は事故調査報告書［中華］の，飛行状況の推定と分析を形式化された日本語による記述に改め，第 2 段階に於いて，1 階理論を構成する．前者は［柳生 3］にて詳しく扱ったので，本稿では事故発生直前の事象列を要約した後，直ちに後者に進むことにする．

5.0　事故の概要：最後の約 1 分 40 秒間の操作と状態遷移列

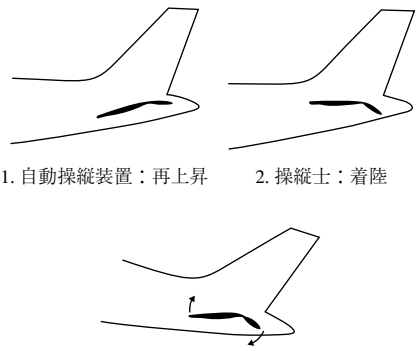
20 時 15 分 45 秒の墜落に至る最後の約 1 分 40 秒間の操作と状態遷移列は，次のように要約される：

［0］ 操縦士が誤って，または意図せずに，AFS を Go Around（GA，着陸中止）Mode に入れた．それは最後まで，解除されることが無かった．

［1］ 操縦士は着陸操作を続行したが，［0］の故に高度が下がらないので，Auto Pilot（AP）を起動させた．

［2］ 操縦士の機首下げの操作と AP による機首上げの動作が同時に進行し，水平尾翼は極端な「へ」字形を取るに至った（図 3）．

［3］ 機首上げ角度が所定の値を越えたため，α–floor（失速防止）機能が働き，出力が急激に増大した．その結果，速度と共に機首上げ角度が更に増し，機体は遂に失速状態に陥った．



1. 自動操縦装置：再上昇　　2. 操縦士：着陸

図 3　操縦士の機首下げ操作（右）と AP による機首上げ（左）の結果の状態

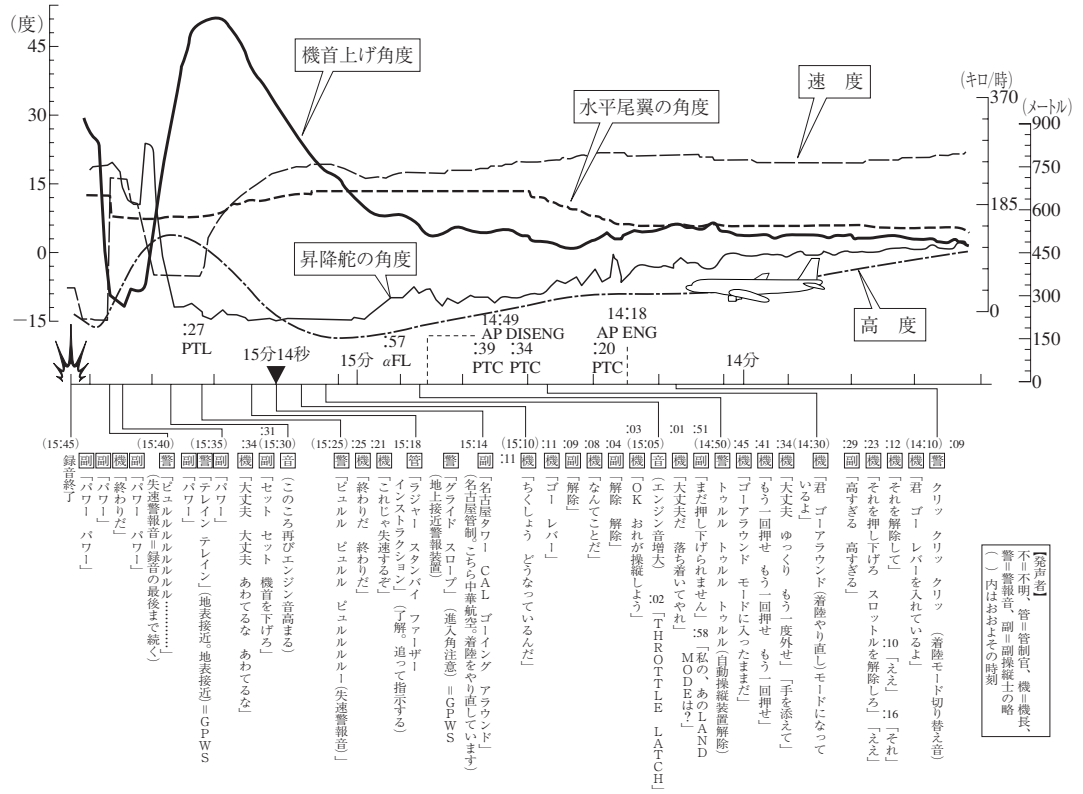


図 4　中華航空機の墜落に至る状態遷移（朝日新聞 94.5.11，［中華］に基づく追加・検証を含む）

5.1　機器の振舞い：状態遷移と因果関係

航空機の機器の振舞いを形式理論として述べるのは無論，膨大かつ緻密な作業を要することであり，以下は事故に直接的に関わる，しかも極く一部の例示に過ぎない．

型　直接制御不可能＝{A《機体》, W《主翼》,
THS《Trimmable Horizontal Stabilizer, 水平安定板》,
E《Elevator, 昇降舵》},
直接制御可能＝{CC《Control Column, 操縦輪》, GL《Go Lever》, THL《Thrust Lever》,
PTC《Pitch Trim Control Switch》},
自動制御＝{AFS《Automatic Flight System, 自動航空 System》,
AP《Auto Pilot, 自動操縦装置》},
対象＝直接制御不可能∪直接制御可能∪自動制御；
型　状態空間$_{X\in 対象}$＝状態(X)，可能状態空間：Π_X 状態空間$_X$＝速度×加速度×…×ptc；
変数　PS, PS$_1$, PS$_2$：可能状態空間，PS$_X$：状態空間$_X$；

関数　状態（対象）
{A　→速度《＝[0, v$_{max}$]》×加速度《＝[－, 0, ＋]》×
高度《＝[0, a$_{max}$]》×上昇率《＝[－, 0, ＋]》×推力《＝[0, thr$_{max}$]》,
W　→迎角《＝[aoa$_{min}$, aoa$_{max}$]》×迎角速度《＝[－, 0, ＋]》×
s[lap]/f[lap]《＝{0/0, 0/15, …, 30/40}》,
THS→角度$_{THS}$《＝[agl$_{Tmin}$, agl$_{Tmax}$]》×角速度$_{THS}$《＝{－, 0, ＋}》
×角加速度$_{THS}$《＝{－, 0, ＋}》,
E　→角度$_E$《＝[agl$_{Emin}$, agl$_{Emax}$]》×角速度$_E$《＝{－, 0, ＋}》
×角加速度$_E$《＝{－, 0, ＋}》,
AFS→mod[es]＝long[itudinal]_mod[es]《＝{SRS, V/S, ALT, LVL/CH, ALT*, PROFILE}》
×lat[eral]_mod[es]
《＝{HDG, HDG_SEL, VOR_CAPTURE, VOR_TRACK, NAV_TRACK}
《G(o_)A(round)＝
(SRS, ‘leveling the wings then maintaining the wings horizontal’)》,
AP　→ap《＝{CWS, CMD, DIS[engaged]}》,
CC　→操作$_{CC}$《＝{push[ed], rest, pull[ed]}》,
GL　→gl《＝{t[ake_]o[ff], ga, n[ot_]op[erated]}》,
THL→操作$_{THL}$《＝{push[ed], rest, pull[ed]}》,
PTC→ptc《＝{－, 0, ＋}》}；

述語　遷移(PS$_1$, PS$_2$)，因果(PS$_1$, PS$_2$, PS$_3$)
《因果(PS$_1$, PS$_2$, PS$_3$)∧c(PS$_1$, PS$_2$)∧e(PS$_2$, PS$_3$)を，原因事象と結果事象を見易くするために，次のように非公式的に書く：遷移(PS$_1$, PS$_2$)∧c(PS$_1$, PS$_2$)⇒遷移(PS$_2$, PS$_3$)∧e(PS$_2$, PS$_3$)．
述語 c, e はそれぞれ，原因に対する付加的条件，及び結果に伴う命題の成立を表す．》；

公理

［0］ ∀m∀t∃t′（遷移（nop：gl, ga：gl）⇒遷移（（m：mod, ga：gl）,（GA：mod, nop：gl））∧
遷移（t：推力, t′：推力）∧（t<t′））

《Go Lever を ga の位置に動かすと，Go Around mode に入り，推力が増す.》,

［1］ ∀t∀t′∀a∃a′∀v（遷移（t：推力, t′：推力）∧（t<t′）⇒遷移（a：加速度, a′：加速度）∧
（a<a′）
⇒遷移（v：速度, v＋a′：速度））

《このような自然法則の記述は通常，省略する.》,

［2］ ∀s∀f（遷移（s：操作$_{CC}$, push：操作$_{CC}$）⇒遷移（f：角加速度$_{E}$, ＋：角加速度$_{E}$））

《操縦輪を押すと昇降舵は正の向きに回転し，機首上げ角度は大きくなる.》,

［3］ PS_{AFS}＝GA∧PS_{AP}＝CMD→∀e∀g（遷移（e：角速度$_{E}$, ＋：角速度$_{E}$）
⇒遷移（g：角加速度$_{THS}$, －：角加速度$_{THS}$））

《「AP は昇降舵及び水平安定板を機首上げ方向に動かそうとした」（「中華」55 頁）とあるが，正確に如何なる事象が「」内を惹起するのか，同報告書からは読み取れない．上の式は，少なくとも操縦輪が押し下げられつつあれば，それは原因となるだろうという推測に基づいている．実際，同報告書には「AP の使用が開始された時から約 18 秒の間に，THS の角度は－5.3° から，機首上げ方向の限界に近い－12.3° まで徐々に大きくなり，その後 15 分 11 秒まで引き続き－12.3° のままであった．この間，昇降舵は連続して機首下げの方向へ操作されている」とある（同 5 頁）.》,

［4］ ∀p（PS_{AP}≠CMD→（遷移（0：ptc, p：ptc）⇒遷移（0：角加速度$_{THS}$, p：角加速度$_{THS}$））

《AP が CMD Mode になければ，ptc の操作により，水平安定板を機首上げ，または下げの向きに動かすことが可能.》,

［5］ ∀n∀n′（（n≠LAND∧n′＝HDG）∨（n＝SRS∧n′≠LAND）→
遷移（GA：mod,（n, n′）：mod））

《上に示す新たな Mode の設定により，GA Mode を解除することが可能.》,

［6］（遷移（CMD：ap, DIS：ap）⇒遷移（GA：mod, ？：mod））∧（？≠GA）

《「GA mode は AP Instinctive Disconnect Pushbutton を押すことにより解除される」（同 182 頁）とあるが，解除後の mode が何であるかは述べられていない.》,

［7］ ∀b∀b′∀t（遷移（b：迎角, b′：迎角）∧（b＜検知角（s/f）＜b′）∧（t＜設定値）
⇒∃t′（遷移（t：推力, t′：推力）∧（t<t′））

《迎角が slap/flap により定まる検知角を越え，推力が予め設定されている値より小さいと，推力も増す.》,

［8］ ∀t∀t′∀v（遷移（t：推力, t′：推力）∧（t<t′）⇒
∃v′（遷移（v：速度；v′：速度）∧（v<v′））

《推力が増すと，速度も増す.》,

［9］ ∀b∀v∀v′（（0<b：迎角）∧（尾翼迎角（角度$_{THS}$, 角度$_{E}$）＜0）∧（v<v′）
⇒∃b′（遷移（b：迎角；b′：迎角）∧（b<b′））

《機首が上昇，尾翼が降下の状態に在って，速度が増すと，迎角は更に増す.》；

［7］～［9］の下線部と□内については，6.2 節を参照.

6.　例 2 の分析

前節の記述の限りではあるが，要求仕様の欠陥と考えられる幾つかの問題点が指摘される．それらは GA　Mode 設定/解除の手順，AP の状態認識能力，及び α–floor の働きに関するものに，大別される．併せて，改善の可能性についても触れる．

6.0　Go–Around Mode の設定/解除の手順

[0]　AP が CMD に設定されており，且つ GA　Mode に在ると，AP の解除により GA も解除される（公理[6]）．にも拘らず，GA を直接的に解除する手段が無いのは何故か？

[1]　また，long. 及び lat. 両 modes の切り替えにより解除する場合も，LAND 以外の mode を選ばなければならない（[5]）．それは何故か？

[2]　もし GA の安易な解除は危険を引き起こすのであれば，AP の解除に伴って解除されるのは何故か？またその時，long. 及び lat. modes は何に設定されるのか？

[3]　安易に解除されるべきでないという考えであれば，設定も操縦士の意図を確認した上で為されるべきではないか？GL を ‘うっかり’ 操作しただけで GA を確定するような仕様は，理解し難い．

6.1　自動操縦装置（AP）の状態認識能力

[0]　GA mode に在る AP が，昇降舵については操縦士による操作を優先させながら，水平安定板をそれとは逆向きに動かすことにより，結果として操縦士の意図の遂行を妨げている（公理[3]）のは，矛盾ではないか？

[1]　しかもその矛盾が増大し，out–of–trim の状態に至る危険を認識できないのは，AP（AFS？）の機能の欠落ではないか？

[2]　更に，最終・最大の危険は失速に有り，その直接の原因は主翼迎角の過度の増大に在るのだから，AP は寧ろ主翼の状態により制御されるべきではないのか？　水平尾翼の状態のみを見ているとすれば，feedback loop が小さ過ぎないか？

6.2　主翼迎角増大と推力増大の正の Feedback

20 時 14 分 57 秒頃，水平安定板と昇降舵の角度差が最大になりつつも，機体は速度の減少と並行する形で，降下を続けていた．しかしその時，（α–floor 機能（主翼迎角が slap/flap に対応する限界角度を越えたことが検知されると，推力が設定値まで自動的に増大される）が働き，速度は増加に転じ，迎角は更に大きくなった．公理[7]～[9]の下線部を辿れば，失速を防止する筈の機能が逆に失速に導く状態を引き起こすという，この極めて危険な正の feedback を伴う因果連鎖が認められる．連鎖を絶つには，述語「因果」の前提条件の内で操作可能なもの — 今の場合は唯一□内のみであるが — を偽とする他は無い．実際，操縦 manual は操縦輪を押して機尾を上げることを指示しているが，事故機の昇降舵は既に限界に達していた．一方，水平安定板は機尾下げの状態に在るので，それを逆転することは可能だが，その状態は AP が操縦士の操作と競合する形で生じさせた結果であり，操縦士の認識の外であった．

6.3　要求仕様の改善の可能性

[0]　状態空間の整理と再構成

飛行の安全のためには，機体の状態の認識と望みの状態への遷移を，迅速・的確に為し得なくてはならない．それには以下の要求の充足が必須，ないし有効であろう：

・状態の各構成要素の意味・働きが，互に明瞭に分離されていること（独立性），

・状態空間の，基本的な状態とその細部への分岐による構造化（階層性），

・凡そ可能な状態遷移は，直接的に可能であること（$tr^* \equiv tr$）．

具体的には：

［0.0］ 飛行記録から推測すると，APが設定されなければ，機体はGA modeのまま着陸したであろう．とすれば，GAとLAND modesはどう違うのか？一方，APがCMDに設定されると差違は鮮明になり，手動での着陸はほぼ不可能となる．ここにはAFSのmodeの意味の曖昧さと，modeの軸とAPの設定の軸の干渉という二重の混乱が有る．Modeの意味は相互に，またAPの状態からも独立に，明確に定めるべきである．

［0.1］ 着陸体勢に入った後は，modeをLANDとGAに限ってはどうか？GAに移った時，long.及びlat.modesのdefault値を（SRS,‘leveling…horizontal’）に設定するのはよいとしても，下位の状態である後者を変えることにより，またそれのみによりGAが解除されるというのは奇怪である．操縦士は明らかに混乱したし，報告書も疑問を呈している（82頁）．GAとLANDは（恐らく操縦士もそう思い込んだように）互いに，直接的に遷移して然るべきであろう．

［0.2］ APの解除がGAの解除を伴うという仕様も，意味の類別の混乱以外の何ものでもない．APの設定がmodeを変えないとすれば（変えるならば，それも疑問である），解除の際も同じmodeのままであるのが，自然であろう．

［1］ 状態遷移の頑健化

前項に加えて，危険な状態が可能状態空間から排除され，たとえ誤操作があっても，そのような状態への遷移は決して生じないことの保証（頑健性）が求められる．特に，out-of-trimの危険（6.1［1］）と主翼迎角と（α-floor機能の正のfeedback（6.1［2］）は避けなければならない．それには例えば：

［1.0］ GA modeの下での操縦士による機首下げ方向への操作という状態そのものが矛盾なのだから，これを感知したらGAを解除するか，あるいは少なくとも警報を発すべきである．

［1.1］ あるいは（操縦士の意図に反する，機首上げ方向への）APによる水平安定板の操作を抑制すべきである（昇降舵についてはoverridingを許しつつ，それに反発する水平安定板の操作を続行するという中途半端な仕様は，正のfeedbackに至る，最悪の選択である）．

［1.2］（α-floorについては，（α-trim（公理［4］の操作の自動化）を同時に機能させることにより，失速防止と機体体勢の回復を共に計ることができないのか？

事故機の同型機については，APの改修を重ねているが，その論理的妥当性は疑わしい，またmanualに曖昧ないし矛盾した記述が少なくない，という問題も指摘される（［Lad］，［柳生2，3］）．Manualを仕様と同一視する立場を取れば，これらの問題も結局は，仕様の正確さ・首尾一貫性の欠如という問題に帰着する．

7. 結　語

工業製品とその組込み software の形式的要求仕様を，状態遷移と因果関係の定式化を中心に論じた．また本来の目的である正確な設計の確保に加え，重要な副産物として，並列的な状態遷移や feedback loop の顕在化等の可能性を示した．しかし設計対象が物理的存在であり，しかも人の手に成る人工物であるという特性に由来する，仕様記述上の問題は，本稿に論じたものの他にも無数に有る．それらの内で，特に重要と思われ，筆者自身が別項（[柳生 1，4，6]，[Yag 2]）にて断片的に触れて来たものを挙げると：[0]「設計ないし設計者の意図」の定式化，[1]設計に於ける‘Abduction’，即ち実現から遡行的に推論し，実現を説明する仮設としての仕様の形成に至る過程の定式化，[2]CAD や CAE に於いて不可欠な，浮動小数点型処理を含む program の，形式的要求仕様記述，等である．これらの問題を解決するためには，本稿が依拠して来た 1 階言語の枠組の拡張，例えば高階化が必要となるかも知れぬことを，指摘して置こう．

＊1　ツインバード工業(株)製 SESIOTIME，図は同製品の説明書中のものである．

参照文献
［朝日］ 朝日新聞，1994. 5. 11.
［Bar 1］ J. Barwise/J. Seligman：‘Information Flow’, Cambridge Univ. Press, 1997.
［Bar 2］ J. Barwise：‘Information and Impossibilities’, Notre Dame Journal. of Formal Logic Vol. 38 No. 4, 1997.
［中華］ 運輸省航空事故調査委員会：「航空事故調査報告書」
—中華航空公司所属エアバス・インダストリー式 A 300 B 4–622 R 型 B 1816 名古屋空港平成 6 年 4 月 26 日，1996. 7. 19.
［Lad］ P. B. Ladkin：‘Analysis of a Technical Description of the Airbus A 320 Braking System’, High Integrity Systems 1(4), 1995.
［M–L 1］ P. Martin–Löf：‘An Intuitionistic Theory of Types：Predicate Part’, in H. E. Rose/J. C. Shepherdson eds. ‘Logic Colloquium 1973’, North Holland, 1975.
［M–L 2］ P. Martin–Löf：‘Constructive Mathematics and Computer Programming’, L. J. Cohen et al. eds. Proc. 6 th Int’l Cong. for Logic, Methodology and Philosophy of Science, North Holland, 1979.
［ヴィト］ L. ヴィトゲンシュタイン/坂井秀寿訳：「論理哲学論考」，法政大学出版局，1975.
［柳生 1］ 柳生孝昭：「設計から見たアブダクション」，吉川弘之監修「技術知の本質」，東京大学出版会，1997. 5.
［柳生 2］ 同　：「要求仕様の形式的記述」，日本ユニシス「技報」第 60 号，1999. 2.
［柳生 3］ 同　：「人工物の因果的構造の情報 Model」，総合知学会誌，2000. 4.
［柳生 4］ 同　：「全体論的に捉えた機能の概念」，科学基礎論研究 97 号，2001. 12.
［柳生 5］ 同　：「近代科学・技術に於ける抽象化の原理とその功罪」，マスダ教育財団「知的文明研究会」報告書，2002. 4.
［柳生 6］ 同　：「幾何計算 Program の形式的要求仕様」，ライフサイクル工学コロキウム予稿集，東京大学人工物工学研究センタ，2002. 7.
［Yag 1］ Yagiu, T.：‘Modeling Design Objects and Processes’, Springer–Verlag, 1991.
［Yag 2］ Yagiu, T.：‘CAX Where X＝Clarification of Designer’s Intents’, Proc. 4 th CAX Work shop, 2000. 11.

執筆者紹介　柳　生　孝　昭（Takaaki Yagiu）

1957年東京大学理学部数学科卒業．58年，日本レミントン・ユニバック株式会社（現日本ユニシス株式会社）入社．米国駐在員，応用ソフトウェア部長，システム本部長，常務取締役を経て，現在，同社顧問．東京大学人工物工学研究センター客員研究員．著書：‘Modeling Design Objects and Processes’, Springer–Verlag, 1991.，訳書：「ある数学者の生涯と弁明」(G. H. Hardy)，シェプリンガー・フェアラーク東京，1994.，共著：「新工学知」（吉川弘之監修），東京大学出版会，1997.