情報処理学会論文誌 Vol.54 No.6 1738–1750 (June 2013)

Event-Bによる列車監視システムのモニタリング要件の検証

佐藤 直人[1,a)] タイ ソン ホアン[2] デビッド ベイジン[2] 來間 啓伸[1]

受付日 2012年9月17日, 採録日 2013年3月1日

概要：本稿では，列車監視システムの仕様を形式手法 Event-B によって検証したケーススタディについて報告する．列車監視システムとは，信号機などの設備の状態データを受信し，それら状態データに基づいて列車運行状況をモニタ表示するシステムである．本ケーススタディでは，与えられた仕様に基づいて Event-B モデルを作成し，仕様が活性に関する性質を満たすことと，与えられた仕様に矛盾のないことを検証した．Event-B モデルを使った活性検証のためには，バリアントなど，特定の条件を満たす論理式を発見的に作成する必要がある．本稿では，対象の列車監視システムにおいて上記論理式を作成する方法を提案する．また仕様の矛盾検証では，リファイメントに基づく従来の手段では検証不可能な，外延的かつ部分的に与えられた仕様と内包的に与えらえた仕様との間の矛盾検証方法を提案する．これらの提案方法は，本稿で対象とした列車監視システムに限らず，任意のシステムに対して応用が見込める．

キーワード：形式手法，Event-B

Verification for Monitoring Requirements of Train Monitoring Systems in Event-B

Naoto Sato[1,a)] Thai Son Hoang[2] David Basin[2] Hironobu Kuruma[1]

Received: September 17, 2012, Accepted: March 1, 2013

Abstract: We report on a case study in specifying and verifying a train monitoring system. This system receives data from railway equipment, such as signals, and it displays their latest status on a monitor. We model the system in Event-B and establish invariants about its behavior. We also prove that the system satisfies liveness properties using our models. To prove the liveness properties, we find particular formulas such as variants in our models. Another challenge we address is proving the consistency of existing specifications with our models. Specifically we compare intensional specifications with partially defined extensional specifications. Our approach will be also useful to verify other systems.

Keywords: formal methods, Event-B

1. はじめに

本稿では，列車監視システムの仕様を，形式手法 Event-B を用いて検証したケーススタディについて報告する．

鉄道分野はシステムの高安全性・高信頼性が要求される分野であるため，従来からソフトウェアの高信頼化を目的とした形式手法の適用が推進されている．また近年では，鉄道向け機能安全規格 BSEN50128 の高レベル水準（Software Safety Integrity Level 3 および Level 4）において，開発上流から下流までの一貫した整合性保証の手段として形式手法が強く推奨されていることから，同規格認証取得のための手法としても適用が検討されている．

従来の鉄道分野における形式手法の適用は，モジュールの機能仕様とその機能実現のための手続き仕様との整合性検証など，開発下流工程が主な対象で，B メソッドや VDM の適用が主流であった [1], [2], [3], [4], [5], [6]．一方で開発上流工程に対しては，近年の Event-B の普及にともない，文献 [7], [8], [9], [10] に見られるようなケーススタディが報

[1] 株式会社日立製作所横浜研究所
Hitachi Yokohama Research Laboratory, Yokohama, Kanagawa 244–0817, Japan
[2] スイス連邦工科大学チューリッヒ校
Department of Computer Science, ETH, Zurich, Swiss
[a)] naoto.sato.je@hitachi.com

告されるようになってきた．Event-B とは，B メソッドを開発上流工程向きに発展させた手法である [10], [11].

Event-B による検証では，対象システムごとに異なるモデルを作成する必要がある．このモデルの作成方法は一般化されておらず．システムの機能や構造，あるいは求められる要件に応じて，どのようにモデルを構成するかが課題となる．現状は，検証者の持つノウハウに従ってモデル記述を行うため，検証者の経験量やセンスによっては何度も試行錯誤を繰り返すことになる．そのため，類似のシステムおよび要件を検証した成功事例が存在する場合は，その事例を模倣してモデルを作成するのが有効である．

本ケーススタディでは列車監視システムを対象に，Event-B による仕様検証を行った．列車監視システムは，路線に設置された信号機や軌道回路などの設備の状態を表す設備状態データを周期的に受信し，その演算結果をモニタに表示する機能を備える．このような監視機能を備えるシステムはコンソールシステムと呼ばれ，鉄道以外にも電力や水道など多くの分野において運用されている．また本ケーススタディでは，「入力データに応じて漏れなく，誤りなくモニタ表示を行う」という，多くのコンソールシステムに共通的に求められる要件を検証した．著者らの知る限り，Event-B によるコンソールシステムの検証事例は報告されていないため，本ケーススタディは，Event-B によってコンソールシステムを検証する際の先行事例として有用である．

対象の列車監視システムには，設備状態データを受信した後に必ずモニタ表示するという，活性（Liveness Property）に分類される性質が求められる．Event-B や B メソッドにおいて活性を検証する方法は，文献 [12], [13], [14], [15] などで提案されているものの，これらの方法で検証を実施するためには，特定の条件を満たす適切な論理式を発見する必要がある．著者らの知る限り，それらの論理式を発見する機械的な方法は提案されていない．そこで本稿では，上記論理式の発見方法を提案する．本提案方法は，列車監視システムをはじめとするコンソールシステムに限らず，同種の活性が求められる任意のシステムの検証に有効である．

また列車監視システムの仕様は，設備データに対する正しいモニタ表示結果を定義する外部仕様と，設備データ演算の手続きを定義する内部仕様によって定義される．このとき，内部仕様に基づくモニタ表示結果は，外部仕様として別途定義される結果と矛盾しないことが求められる．Event-B では，外部仕様をモデルに導入した後，リファイメントと呼ばれる方法によって内部仕様を導入することで，双方の仕様が一致し，矛盾しないことを示せる．ところが対象の列車監視システムでは，具体値によって外延的かつ部分的に外部仕様が与えられるため，外部仕様が仕様を定義する範囲（定義域）は内部仕様よりも狭い．つまり，仕様間に矛盾がない場合でも両者が一致することはないため，上記方法では矛盾を検証できないという問題がある．そこで本稿では，内部仕様と一致する仮想的な外部仕様を導入して従来のリファイメントを実施した後，実際の外部仕様が上記仮想的な外部仕様に含まれるかを判定することで，仕様の矛盾検証を行う方法を示す．本提案方法もコンソールシステムに限らず，外延的かつ部分的に外部仕様が与えられる任意のシステムに対して有効である．

以下，2 章では Event-B について簡単に説明する．3 章では，対象とする列車監視システムについて説明する．4 章と 5 章では，検証の課題と解決方法を述べる．6 章では解決方法に基づいて作成した Event-B モデルと，その検証方法を示す．7 章では，検証結果を示し，8 章では結果の評価を行う．9 章では関連研究について述べ，最後に 10 章で結論をまとめる．

2. Event-B の概要

Event-B は，B メソッドを開発上流工程に向けて発展させた形式手法であり，B メソッドの前提となるモジュールの機能仕様を，システムの要件に基づいて正しく導くことが主目的といえる．B メソッドと同様に集合論や一階述語論理を道具に用い，システムの詳細化の手続き（リファイメント）の正しさを証明することで，誤りのないシステムを開発するという考え方（Correctness by Construction）を採用している．

2.1 コンテクストとマシン

Event-B のモデルは，コンテクストとマシンから構成される．コンテクストでは，システムの静的な要素を定義する．データ構造を表現するために必要なキャリア集合（Carrier Sets）や関数などを表す定数（Constants）を宣言するとともに，宣言した定数の性質を公理（Axioms），あるいは定理（Theorems）として定義する．定理は証明が必要であるが，公理は前提として扱われるため証明は不要である．またコンテクストは他のコンテクストを拡張（Extends）しても作成できる．

一方マシンでは，システムの動的な要素を定義する．システムの状態などを表す変数（Variables）を宣言し，宣言した変数間に成立する条件をインバリアント（Invariants）あるいは定理（Theorems）として定義する．また，変数に値を代入するイベント（Events）でシステムの振舞いを定義する．また，マシンは他のマシンをリファイメント（Refines）することでも作成できる．マシンはコンテクストを参照（Sees）する．

イベントは，システムの離散的な状態遷移を表現しており，ガード条件とアクションから構成される．ガード条件は変数や定数を参照する式であり，ガード条件が成立する場合にイベントは発火可能である．アクションは変数への代入文であり，ガード条件が成立しイベントが発火した際に実行される．

2.2 リファイメント

リファイメントはマシンの詳細化を目的とした手続きであり，マシンの詳細化のために必要なコンテクストの拡張も含む．マシンにおけるリファイメントの対象はイベントであり，ガード条件の強化や，（新しく追加した変数に対する）アクションの追加などが可能である．このリファイメントを活用することで，イベントの追加や分割が可能になる．たとえばイベントの追加は，skip イベント（空イベント）のリファイメントに相当する．

2.3 証明責務

Event-B モデルの正しさは，証明責務の証明によって保証される．証明責務とは，変数間の関係を表すインバリアントの正しさや，リファイメントの正しさを表す論理式で，Event-B のモデル記述・証明環境である Rodin platform [16] によって Event-B モデルから一定の規則で生成される．生成された証明責務を証明することにより，Event-B モデルに関する性質を保証できる．つまり，Event-B における正しさの基準は証明責務によって与えられる．

ここでは，本稿で扱う以下の証明責務を紹介する．これらの証明責務の厳密な定義，およびその他の証明責務については，文献 [10] を参照されたい．

【マシンの定理に関する証明責務 **THM**】マシンの定理は，（それ以前に記述された）マシンの定理およびインバリアントと，参照可能なコンテクストの公理および定理から導出可能であること．
【インバリアントに関する証明責務 **INV**】イベントの実行前にインバリアントが成立するならば，同イベントの実行後にも同インバリアントは成立すること．
【イベントのシミュレーションに関する証明責務 **SIM**】リファイメント後のアクションは，リファイメント前のアクションを模倣する（矛盾しない）こと．

2.4 イベントの収束性

イベントには，収束性に関する属性として Convergent あるいは Anticipated の属性を追加できる．Convergent イベントとは，連続実行した場合にいずれ発火不可能になる（収束する）イベントである．より形式的には，その実行によって，バリアントと呼ばれるある自然数 V を減少させるイベントを意味する．バリアント V は自然数であるため，バリアント V が 0 まで減少すると，Convergent イベントは発火不可能になる．また Anticipated イベントとは，Convergent イベントの合間に実行しても，Convergent イベントの収束を妨げないイベントである．より形式的には，その実行によって，上記バリアント V を増加させないイベントを意味する．

Convergent あるいは Antitipated の属性を付してイベントを宣言した場合は，上記性質を満たすバリアント V を記述する必要がある．またその場合，以下の証明責務の証明が課される．

【**Convergent** イベントの証明責務 **VAR1**】当該 Convergent イベントの実行によってバリアント V が減少すること．
【**Convergent** イベントの証明責務 **NAT1**】バリアント V は自然数であること．
【**Anticipated** イベントの証明責務 **VAR2**】当該 Anticipated イベントの実行によって Convergent イベントのバリアント V が増加しないこと．
【**Anticipated** イベントの証明責務 **NAT2**】バリアント V は自然数であること．

3. 列車監視システムの監視機能

本研究で題材とした列車監視システムは，列車運行状況を示す信号機や軌道回路などの設備の状態をモニタ表示し，必要に応じて鉄道指令員からの制御操作を受け付けて列車集中制御装置へ送出するコンソールシステムである．鉄道指令員に対してモニタ表示を行う監視機能と，鉄道指令員から制御命令を受け付ける制御機能で構成されるが，本研究ではこのうち監視機能のみを対象とする．以降，対象システムという場合，この監視機能を意味する．

3.1 外部環境

対象システムの外部環境は，列車集中制御装置とその監視対象である設備，および鉄道指令員に関する定義として与えられる．

まずは，信号機などの設備に関する仮定を与える．

Assumption 1：設備の数は有限である．
Assumption 2：設備は，（信号機の表示など）可変な状態を表す物理状態と，（信号機の配置や他設備との関係など）不変な設定を表す物理設定を持つ．

設備の物理状態は，列車の運行によって変化する場合や，他装置からの入力によって変化する場合もある．本研究では，設備の物理状態を変化させる主体を限定せず，設備の物理状態は任意に変化すると仮定する．

Assumption 3：設備の物理状態は，任意のタイミングで任意の値に変化する．

次に，列車集中制御装置に関する仮定を与える．列車集中制御装置は，路線に設置された設備を 1 カ所で遠隔監視および制御するための装置である．設備の物理状態を取得して設備状態データを作成し，周期的に列車監視システムへ送信する．本研究ではより一般化し，任意のタイミングで設備状態データを作成・送信すると仮定する．また設備状態データは主に設備の物理状態に基づいて作成されるが，一部の設備状態データは，列車監視システムの制御機能が送信する制御命令に基づいて作成される．

Assumption 4：列車集中制御装置は任意のタイミングで物理状態を取得して設備状態データを作成し，列車監視システムへ送信する．設備状態データの送受信に際しては，データ改ざんや紛失の可能性はない．

Assumption 5：列車集中制御装置は，受信した設備の物理状態，および受信した制御命令に基づいて設備状態データを作成する．

最後に，制御命令を入力する鉄道指令員に関する仮定を与える．

Assumption 6：鉄道指令員は，（コンソールシステムの制御機能を介して）任意の設備に対する制御命令を，任意のタイミングで列車集中制御装置へ送信する．

3.2 システム仕様

次に，対象システムの内部構成を定義する．

Specification 1：列車監視システムの監視機能は，演算モジュールと表示モジュールから構成される．

Specification 2：演算モジュールは，列車集中制御装置から受信した設備状態データを読み出し，設備状態データと設備設定データを演算することで，表示データを作成する．表示データ演算の過程で，中間データを作成する場合もある（表示データ演算処理）．

Specification 3：表示モジュールは，作成した表示データに基づいて設備の状態をモニタ表示する（モニタ表示処理）．

Specification 4：表示データ演算処理の後は，必ずモニタ表示処理を実行する．いずれかの処理のみ連続で行うことはない．

設備設定データは設備に関する不変データであり，設備の物理設定に基づいて一意に決定される．表示データは設備のモニタ表示方法を示す可変データであり，表示データの値とモニタ表示結果は 1 対 1 に対応付く．

また設備設定データは，設備配置情報などを含む物理設定から，監視機能の観点で必要な情報を抽出しデータ化した結果である．よって本研究では，物理設定は設備設定データと同等の情報を持ち，設備設定データは物理設定に依存すると仮定する．

Assumption 7：物理設定は設備設定データと同等の情報を持ち，設備設定データは物理設定に依存する．

以降では簡単のため，システムの入力となる設備状態データと物理設定を合わせて，入力データと呼ぶ．

ここで，上記入力データと，出力であるモニタ表示結果の対応関係を表す仕様を外部仕様と呼ぶことにする．本対象システムにおける外部仕様は，具体値によって外延的に，かつ部分的に与えられる．この外延的かつ部分的な外部仕様は，テストケースとしても使用される．また，システムを実現する表示データ演算処理，およびモニタ表示処理の仕様を，内部仕様と呼ぶことにする．本対象システムでは，IF-THEN-ELSE 制御構造の手続き仕様として内包的に内部仕様が与えられる．この内部仕様に従ってシステムは実装される．

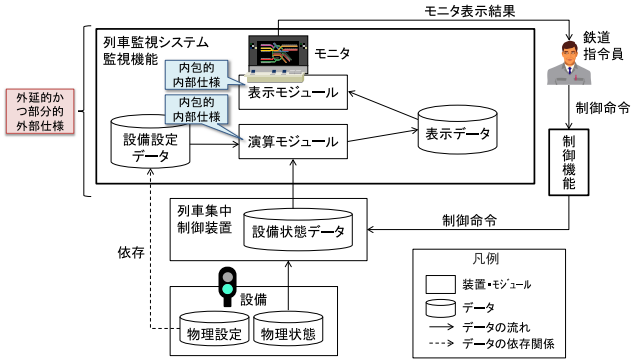

図 1 列車監視システム監視機能とその外部環境の構成

Fig. 1 Structure of train monitoring control system and its environment.

対象システムとその外部環境の構成を**図 1** にまとめる．

3.3 要件

列車監視システムの仕様設計担当者にヒアリングを行った結果，対象システムの目的は，「入力データに応じて漏れなく，誤りなくモニタ表示を行い，列車運行状況を鉄道指令員に伝達すること」であると分かった．そこで本研究では，上記目的を 2 つの要件に分割して定義した．以下では，要件の直観的な意味を自然言語で説明するとともに，LTL（Linear Temporal Logic）の時相演算子を用いて定義する．

Requirement 1：設備状態データの送信回数が正整数 K 回の場合，いずれ必ずモニタ表示回数も K 回になる．
【LTL 論理式】$\Box(\mathrm{ct} = \mathrm{K} \Rightarrow (\Diamond \mathrm{mt} = \mathrm{K}))$

Requirement 2：入力データが外延的かつ部分的な外部仕様の入力として定義されている場合，そのモニタ表示結果は，外部仕様の定義する結果と一致する．
【LTL 論理式】$\Box(\forall \mathrm{n} \cdot \mathrm{n} \in 1..\mathrm{mt} \wedge (\mathrm{ctdSts}(n) \mapsto \mathrm{phyCnf}) \in dom(\mathrm{EX}) \Rightarrow \mathrm{monSts(n)} = \mathrm{EX}(\mathrm{ctdSts(n)} \mapsto \mathrm{phyCnf}))$

時相演算子 “□” を使った論理式 $\Box\phi$ は，今後つねに ϕ は真であることを意味する．同様に $\Diamond\phi$ は，将来いずれかの時点で ϕ は真になることを意味する．また上記論理式に現れる ct は，列車集中制御装置から監視機能への設備状態データの送信回数を表す変数である．同様に mt はモニタ表示回数を表す変数である．K は任意の正整数を表す定数である．また，ctdSts は設備状態データを表す変数で，n 回目に送信した設備状態データを ctdSts(n) と表記する．同様に monSts は，内部仕様に従って出力されるモニタ表示結果を表す変数で，n 回目に行ったモニタ表示結果を monSts(n) と表記する．phyCnf は物理設定を表す定数である．EX は，外延的かつ部分的に与えられる外部仕

様を表す関数で，入力データを引数にとり，モニタ表示結果を戻り値として返す．*dom* は関数定義域を返す関数であり，$\mapsto$ はペアを表す記号である．

Requirement 1 ではモニタ表示結果の正しさについては言及せず，入力データの回数と同じ回数のモニタ表示が漏れなく行われることのみを要求する．そして **Requirement 2** において，それらのモニタ表示結果が，外部仕様と一致する正しい結果であることを要求する．これらの要件を検証することで，対象システムが上記目的を達成するかを確認できる．

3.4 対象設備

列車監視システムの監視機能は通常，信号機や軌道回路をはじめとする数十種類の設備を監視する．本研究ではこのうち，信号機の監視機能のみを対象に Event-B による検証を行った．信号機は，設備の中でも最も複雑な中間データおよび表示データの演算を行う設備の 1 つであるため，本研究で扱う代表設備として選出した．

4. システム活性の検証

4.1 課題

Requirement 1 は，システムの活性に関する性質である．Event-B や B メソッドのような離散的な状態遷移記述において活性を検証する方法は，文献 [12], [13], [14], [15] などで議論されてきた．本研究では，Rodin platform のツール支援を受けて Event-B の活性を検証できる文献 [15] の方法を採用した．

文献 [15] では，活性に関する要件として一般的な “Existence 要件（$\Box\Diamond$P）”，“Progress 要件（$\Box$(P1 $\Rightarrow$ $\Diamond$P2)）”，“Persistence 要件（$\Diamond\Box$P）” に対し，それぞれの証明方法を示している[*1]．これらの方法には，上記のとおり Rodin platform のツール支援を活用できるという利点がある一方で，証明の過程で使用する特定の論理式を検証者自らが発見しなければならないという課題がある．上記論理式は，Event-B モデルの記述内容に依存するため，検証者はモデルごとに上記論理式を発見する必要がある．そして，著者らの知る限り，上記論理式を作成する機械的な方法は提案されていない．

本研究で扱う **Requirement 1** は上記 Progress 要件に属するため，文献 [15] の検証方法を適用できる．その際，上述のとおり，証明の過程で使用する特定の論理式を発見することが課題となる．

以下，課題をより具体化するため，P1，P2 をそれぞれ ct = K, mt = K とし，文献 [15] に基づく **Requirement 1** の検証方法を示す．前提として，Event-B モデルに含まれるイベントを，システムの動作を表すシステムイベントと，外部環境の変化を表す環境イベントの 2 種類に分類する．また以下に表れる ϕ *Unitl* ψ は，ϕ は現在または将来の時点で真であり，かつ ψ はその時点まで真であることを意味する．

【文献 [15] に基づく **Requirement 1** の検証方法】

(1) システムイベントは，そのガード条件が無限にしばしば成立すれば，無限にしばしば実行すること（Strong Fairness Assumption）を仮定する．
(2) P3 $\wedge$ $\neg$mt = K が成立する場合，システムイベントは収束する，つまりあるバリアント V を減少させる Convergent イベントであることを証明する（P3 はシステムの状態を表すメタ変数）．
(3) P3 $\wedge$ $\neg$mt = K が成立する場合，環境イベントはシステムイベントの収束を妨げない，つまり上記バリアント V を増加しない Anticipated イベントであることを証明する．
(4) P3 $\wedge$ $\neg$mt = K が成立する場合，いずれかのシステムイベントは発火可能であること（デッドロックに陥らないこと）を証明する．
(5) 上記 (1)(2)(3)(4) より，$\Box\Diamond(\neg$P3 $\vee$ mt = K) を証明する．
(6) 全イベントについて，イベント実行前に P3$\wedge\neg$mt = K が成立するならば，イベント実行後には P3$\vee$mt = K が成立することを証明する．
(7) 上記 (5)(6) より，$\Box$(P3 $\Rightarrow$ (P3 *Until* mt = K)) を証明する．
(8) $\Box$(ct = K $\wedge$ $\neg$mt = K $\Rightarrow$ P3) を証明する．
(9) 上記 (7)(8) より，$\Box$(ct = K $\Rightarrow$ $\Diamond$mt = K) を証明する．

文献 [15] では，メタ変数 P1，P2，P3 を使って (5)(7)(9) の証明を与えているため，本稿で改めて証明する必要はない．また，(1) は対象システムに対する仮定である．対象システムにおいては，環境イベントの実行とシステムイベントの実行は独立のため，(1) の仮定は妥当と判断できる．以上より，本研究では (2)(3)(4)(6)(8) を証明する．(2)(3) の証明のため，システムイベントを Convergent イベントとして宣言し，環境イベントを Anticipated イベントとして宣言する．(4)(6)(8) の論理式は，インバリアントとして記述し証明する．

ここで，残りの (2)(3)(4)(6)(8) の証明のため，P3 やバリアント V に相当する適切な論理式を作成することが課題となる．以降，P3 に相当する論理式を補助式と呼ぶ．

4.2 解決方法

まず補助式 P3 を発見する方法を示す．P3 に関して証明すべき性質は，$\Box$(P3 $\Rightarrow$ (P3 *Until* mt = K))，および $\Box$(ct = K $\wedge$ $\neg$mt = K $\Rightarrow$ P3) である．つまり，これらの性質を満たす P3 を発見すればよい．これらの性質は，設備

[*1] P，P1，P2 はシステムの状態を表すメタ変数．

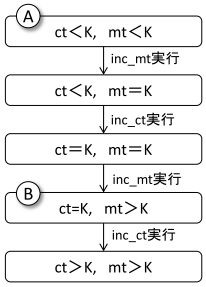

図 2 K を境界とした ct と mt の値遷移
Fig. 2 Transition of ct and mt on the boundary K.

状態データ送信回数 ct およびモニタ表示回数 mt と，正整数 K で構成されている．よって P3 も，ct および mt と，K に関係すると推測できる．そこで，K を境界に ct と mt の値の変化を分析することで，P3 を発見する．

本方法を説明するための簡単な例として，mt と ct の値を更新するイベントとして inc_mt と inc_ct が存在し，それぞれのイベントが交互に実行される場合を考える．mt と ct の初期値が等しいとすると，mt と ct の値は**図 2** の状態遷移図のとおり遷移する．

この状態遷移図上で，$\Box(\mathrm{P3} \Rightarrow (\mathrm{P3}\ \mathit{Until}\ \mathrm{mt} = \mathrm{K}))$，および $\Box(\mathrm{ct} = \mathrm{K} \wedge \neg\mathrm{mt} = \mathrm{K} \Rightarrow \mathrm{P3})$ が真となるために，P3 が真となるべき状態を見つける．たとえば，$\Box(\mathrm{P3} \Rightarrow (\mathrm{P3}\ \mathit{Until}\ \mathrm{mt} = \mathrm{K}))$ が真になるためには，mt が K になるまで P3 が真であればよい．よって P3 は，mt が K になる直前の状態 A において真である必要がある．同様に，$\Box(\mathrm{ct} = \mathrm{K} \wedge \neg\mathrm{mt} = \mathrm{K} \Rightarrow \mathrm{P3})$ が真となるためには，状態 B において P3 が真であればよい．そこで状態 A で成立する論理式 $\mathrm{ct} < \mathrm{K} \wedge \mathrm{mt} < \mathrm{K}$ と，状態 B で成立する論理式 $\mathrm{ct} = \mathrm{K} \wedge \mathrm{mt} > \mathrm{K}$ の論理和をとり，P3 として $(\mathrm{ct} < \mathrm{K} \wedge \mathrm{mt} < \mathrm{K}) \vee (\mathrm{ct} = \mathrm{K} \wedge \mathrm{mt} > \mathrm{K})$ を得る．

次に，バリアント V を作成する方法を示す．まずシステムイベントごとに，「当該システムイベントによって減少し，任意の環境イベントによって増加しない自然数」を表す式を作成する．これを仮バリアントと呼ぶことにする．そしてシステムイベントごとに作成した仮バリアントを足し合わせることで，すべてのシステムイベントによって減少するようなバリアント V を作成する．ところが，仮バリアントの作成ではある 1 つのシステムイベントのみに注目するため，あるシステムイベントの実行によって，他のシステムイベントの仮バリアントが増加する可能性がある．たとえば，システムイベント X，Y と，その仮バリアント vx，vy が与えられたとする．vy の作成時には X を考慮しないため，X は vx を 1 減少させる一方で，vy を 1 増加させる場合がある．この場合，vx + vy はバリアント V として適切ではない．

上記のような場合には，仮バリアントに適切な係数を設定する．たとえば上記例では，vx に対して係数 2 を設定し 2vx + vy とすることで，適切なバリアント V を得られる．

これらの解決方法は，列車監視システムをはじめとするコンソールシステムに限らず，任意の Progress 要件を Event-B で検証する際に有効と考えられる．本提案方法の有効範囲については，8 章で詳しく考察する．

5. 外延的かつ部分的外部仕様と内包的内部仕様の無矛盾性の検証

5.1 課題

Requirement 2 の検証では，システムの外延的かつ部分的な外部仕様 EX の定義するモニタ表示結果と，内包的内部仕様に従って出力されるモニタ表示結果 monSts との間に矛盾がないことを検証する．

Event-B では，外部仕様に従ってモニタ表示結果を出力するマシンを，内部仕様に従ってモニタ表示結果を出力するマシンにリファイメントすると，「内部仕様は外部仕様を模倣する」という意味の証明責務 SIM が課される．この証明責務 SIM を証明することで，外部仕様と内部仕様との間に矛盾がないことを示せる．

しかし本研究のように外部仕様を外延的かつ部分的に与えた場合は，上記方法は適用できない．外延的かつ部分的な外部仕様は，一部の入力に対してのみ出力を定義するが，一方で内包的に与えられた内部仕様は，あらゆる入力に対して（全域的に）出力を定義する．つまり，外部仕様よりも内部仕様の定義域が広いため，内部仕様は外部仕様を模倣しない．よって，外部仕様と内部仕様が矛盾しない場合でも，上記証明責務 SIM は証明できない．

以上より，従来のリファイメントによる方法では検証不可能な，外延的かつ部分的な外部仕様と内包的な内部仕様の無矛盾性検証が課題となる．

5.2 解決方法

本研究では上記課題解決のため，内部仕様と一致する全域的な仮想外部仕様 EX_total を導入する．この EX_total は，内部仕様と同じモニタ表示結果を与える理想的な外部仕様である．EX_total を導入することで，**Requirement 2** は以下の **Requirement 2'** に書き換えられ，従来のリファイメントによる方法の拡張で検証を行えるようになる．**Requirement 2** と **Requirement 2'** の同値性の証明は付録に記載する．

Requirement 2'：外延的かつ部分的な外部仕様は，全域的な仮想外部仕様の部分集合である．ただし，入力データが外延的かつ部分的な外部仕様の入力として定義されている場合，そのモニタ表示結果は，全域的な仮想外部仕様の定義する結果と一致する．
【LTL 論理式】$\Box(\mathrm{EX} \subseteq \mathrm{EX\_total})$
ただし EX_total は，以下の論理式を満たす全域関数で

ある．

$$\square(\forall n \cdot n \in 1..mt \land (ctdSts(n) \mapsto phyCnf) \in dom(EX) \Rightarrow monSts(n) = EX\_total(ctdSts(n) \mapsto phyCnf))$$

ここで，内部仕様である表示データ演算処理およびモニタ表示処理の入出力関係を表す関数として，CTD_2_SYS および SYS_2_MON を導入する．CTD_2_SYS は入力データを引数にとり，表示データを戻り値として返す関数である．同様に，SYS_2_MON は表示データを引数にとり，モニタ表示結果を戻り値として返す関数である．

本研究では **Requirement 2'** の条件を満たす仮想外部仕様として，上記 CTD_2_SYS および SYS_2_MON を用い，EX_total = CTD_2_SYS;SYS_2_MON となる EX_total を与える．

本提案方法は列車監視システムをはじめとするコンソールシステムに限らず，外延的かつ部分的な外部仕様と，内包的な内部仕様が与えられる任意のシステムを，Event-B で検証する際に有効と考えられる．詳しくは，8 章で考察する．

6. Event-B モデル

本研究で対象とした信号機の監視機能では，表示データ演算処理において多くの中間データを作成する．そのため，作成した Event-B モデルも多くの変数やイベントを含む結果となった．本章では説明の簡単化のため，作成したモデルから中間データに関係する部分を省略して示す．

また本研究では Generic Instantiation [17] を適用して Event-B モデルを作成した．Generic Instantiation とは，モデルのテンプレート（テンプレートモデル）を活用して目的のモデルを得る手法である．Generic Instantiation に基づく方法では，テンプレートモデルが含む抽象的なキャリア集合や定数を，より具体的なデータ構造を持つキャリア集合や定数に置き換えることで目的のモデル（インスタンスモデル）を得る．これにより，インスタンスモデルを容易に作成できるという利点がある．さらに，テンプレートモデルで行ったマシンに関する証明は，インスタンスモデルにおいても有効であるため，インスタンスモデル上での証明を省略できるという利点もある．

本研究で対象とした列車監視システムの監視機能は，運用先の路線ごとに基本的な機能や構成が変わることはない．一方で，入力データの種類やその演算の方法は，路線ごとに変更される．そこで本研究では，入力データを表すキャリア集合やその演算を表す定数を抽象化してテンプレートモデルを作成し，それらを具体的なキャリア集合や定数に置き換えることで，路線ごとのインスタンスモデルを作成した．さらに，**Requirement 1** の証明をテンプレートモデル上で行い，インスタンスモデルにおいてその証明結果を再利用した．これにより，今後路線ごとのインスタンスモデルを容易に作成できるとともに，作成したインスタンスモデルにおける **Requirement 1** の検証を省略できるようになる．

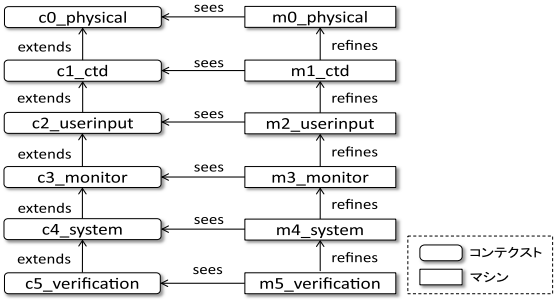

図 3 テンプレートモデルの構成
Fig. 3 Composition of Template Model.

一方で **Requirement 2** は，入力データの種類やその演算方法に依存するため，インスタンスモデル上で検証した．

6.1 テンプレートモデルの作成

作成したテンプレートモデルの構成を図 **3** に示す．

後の 6.3 節で作成するインスタンスモデルでは，仮想外部仕様 EX_total が **Requirement 2'** の前提条件を満たすことを，リファイメントを利用して段階的に証明する．インスタンスモデルの構成は，そのもとになるテンプレートモデルに依存するため，上記リファイメントによる証明を見越してテンプレートモデルを構成する必要がある．そのため，以下のとおりテンプレートモデルを構成した．

仮想外部仕様 EX_total が満たすべき **Requirement 2'** の前提条件とは，「EX_total の定義するモニタ表示結果が，内部仕様に従って実際に出力されるモニタ表示結果 monSts と一致すること」である．そこでまず，仮想外部仕様 EX_total を定義するうえで必要となる外部環境をモデル化する．モデルを見通し良く記述するため，設備や列車集中制御装置などの外部環境の要素は，リファイメントを利用して段階的に追加する．m0_physical では，**Assumption 1**，**2**，**3**，**7** に基づき，路線に設置される物理的な信号機をモデル化する．m1_ctd では，**Assumption 4**，**5** に基づき，物理状態値から設備状態データを作成して列車監視システムへ送信する，列車集中制御装置をモデル化する．m2_userinput では，**Assumption 6** に基づき，鉄道指令員による制御命令の更新と，その制御命令の列車集中制御装置への送信をモデル化する．

次に，対象システムの記述をリファイメントによって追加していく．m3_monitor では，仮想外部仕様 EX_total に従ってモニタ表示結果 monSts を出力するように対象システムを記述する．この m3_monitor において，仮想外部仕様 EX_total の定義するモニタ表示結果が，実際のモニタ表示結果 monSts と一致することを証明する．続いて m4_system では，内部仕様 CTD_2_SYS に従う表示データ演算処理と，

同じく内部仕様 SYS_2_MON に従うモニタ表示処理を導入し，内部仕様に従ってモニタ表示結果 monSts を出力するように詳細化する．m3_monitor から m4_system へのリファイメントの際に課される証明責務 SIM を証明することで，「仮想外部仕様 EX_total の定義するモニタ表示結果が，内部仕様に従って実際に出力されるモニタ表示結果 monSts と一致する」という **Requirement 2'** の前提条件を証明できる．

最後に m5_verification において，**Requirement 1** の検証に必要なインバリアントやバリアントを記述し証明する．

6.2 Requirement 1 の検証

4.1 節に示した検証方法に従い，**Requirement 1** の検証に必要なインバリアントやバリアントを m5_verification に記述した．本節では，その際に課題となった補助式 P3 およびバリアント V の作成方法を示す．

まずはじめに，補助式 P3 の作成方法を示す．4.2 節に示した方法に従い，K を境界に ct と mt の値変化を分析する．ct および mt は，列車集中制御装置による設備状態データの作成を表す環境イベント ctdUpdateStatus と，モニタ表示処理に対応するシステムイベント monUpdateStatus によって増加する．また st は，表示データの演算回数を表す変数で，表示データ演算処理に対応するシステムイベント sysUpdateStatus によって増加する．これらのイベント記述を以下に示す．

```
ctdUpdateStatus ≙
    BEGIN
      atc1：ct := ct + 1
      act2：ctdSts(ct + 1)
            := PHY_2_CTD(phySts ↦ ctrlSts)
    END
sysUpdateStatus ≙
    WHEN
      grd1：st ≠ ct
      grd2：st = mt
    THEN
      act1：st := st + 1
      act2：sysSts(st + 1)
            := CTD_2_SYS(ctdSts(st + 1) ↦ phyCnf)
    END
monUpdateStatus ≙
    WHEN
      grd1：mt ≠ st
    THEN
      act1：mt := mt + 1
      act2：monSts(mt + 1)
            := SYS_2_MON(sysSts(mt + 1))
    END
```

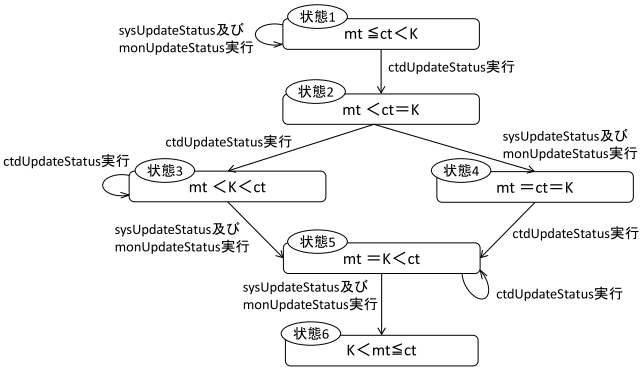

図 4 K を境界とした ct と mt の値遷移
Fig. 4 Transition of ct and mt on the boundary K.

```
    END
```

grd や act は，それぞれガード条件とアクションのラベルである．sysSts(n) は，n 回目に作成した表示データを表す変数である．phySts と ctrlSts はそれぞれ，物理状態と制御命令を表す変数である．PHY_2_CTD は，物理状態と制御命令から設備状態データを作成する列車集中制御装置の仕様を表す関数である．

上記イベント記述より，ct，st，mt に対して以下のインバリアントが成立する．inv はインバリアントのラベルである．

inv1：$st \leqq ct$
inv2：$mt \leqq st$
inv3：$st \leqq mt + 1$

これらのイベントおよびインバリアントより，ct および mt は**図 4** のとおり遷移することが分かる．

この状態遷移図上で，$\Box(P3 \Rightarrow (P3\ \mathit{Until}\ mt = K))$，および $\Box(ct = K \wedge \neg mt = K \Rightarrow P3)$ が真となるために，P3 が真となるべき状態を見つける．$\Box(P3 \Rightarrow (P3\ \mathit{Until}\ mt = K))$ が真となるためには，mt が K になるまで P3 が真であればよい．mt は状態 4 および状態 5 で K になるため，P3 はそれらの直前の状態 2 および状態 3 で真であればよい．同様に，$\Box(ct = K \wedge \neg mt = K \Rightarrow P3)$ が真となるためには，$ct = K \wedge \neg mt = K$ のときに，P3 が真であればよい．$ct = K \wedge \neg mt = K$ は，状態 2 で成立するため，P3 は状態 2 において真であればよい．以上より，状態 2 および状態 3 において真となるように，P3 を構成すればよいと分かる．そこで，$mt < ct = K$ と $mt < K < ct$ の論理和をとり，補助式 P3 として $mt < K \wedge K \leqq ct$ を得る．

次に，P3 は $mt < K \wedge K \leqq ct$ であることを前提に V を作成する．まず，システムイベント sysUpdateStatus および monUpdateStatus に対して仮バリアントを作成する．インバリアント inv1 より，sysUpdateStatus は $ct - st$ を減少させる．また同様に，インバリアント inv3 より，sysUpdateStatus は $mt + 1 - st$ を減少させる．しかし，$ct - st$ は，環境イベント ctdUpdateStatus により増加す

るため，仮バリアントの要件を満たさない．以上より，sysUpdateStatus の仮バリアントとして mt + 1 − st を選択する．

monUpdateStatus についても同様で，インバリアント inv2 に着目することにより仮バリアント st − mt を導ける．

ここで，ある適切な係数 a と b を用いると，バリアント V は a(mt + 1 − st) + b(st − mt) と表せる．sysUpdateStatus は monUpdateStatus の仮バリアント st − mt を増加させるため，a > b となるような係数を選択すればよい．一方で，monUpdateStatus は sysUpdateStatus の仮バリアント mt + 1 − st を増加させるため，b > a となるような係数を選択する必要があり，これらは矛盾する．つまり，これらの仮バリアントの組合せでは，バリアント V を作成できない．

そこで，システムイベントによって操作されない正整数 K の利用を考える．上記 P3 より mt < K が成立するため，monUpdateStatus の仮バリアントとして K − mt を導ける．この場合，バリアント V は a(mt + 1 − st) + b(K − mt) と表せ，b > a となるような係数を適当に選択すればよい．たとえば，b = 2，a = 1 が考えられる．以上より，バリアント V として，(mt + 1 − st) + 2(K − mt) が得られる．

m5_verification では，上記のとおり作成した P3 および V を使ってインバリアントやバリアントを記述し，それらの証明責務を証明することで **Requirement 1** を検証した．

6.3 インスタンスモデルの作成

テンプレートモデルでは，入力データを表すキャリア集合と，その演算方法を表す定数を抽象的に記述した．これらの抽象的なキャリア集合や定数を，与えられた仕様に従って具体化したキャリア集合および定数に置き換えることで，インスタンスモデルを作成した．このインスタンスモデル上で，**Requirement 2** を検証する．

6.4 Requirement 2 の検証

5.2 節の方法に従い，**Requirement 2'** を検証する．まず m3_monitor において，監視機能全体を表すイベント monitoringSystem を，仮想外部仕様 EX_total を使って定義する．EX_total は，c3_monitor で定数として宣言する．

```
monitoringSystem ≙
    WHEN
      grd1 : mt ≠ ct
    THEN
      act1 : mt := mt + 1
      act2 : monSts(mt + 1)
             := EX_total(ctdSts(mt + 1) ↦ phyCnf)
    END
```

同時に c3_monitor で，外延的かつ部分的な外部仕様 EX を宣言する．このとき，以下のインバリアントが成立する．

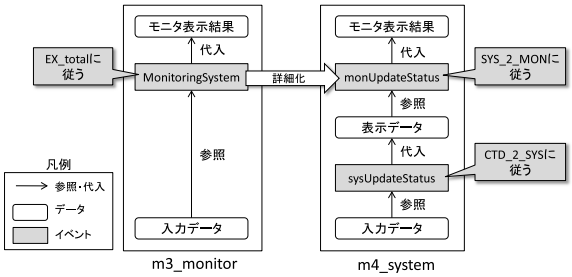

図 5 m3_monitor と m4_system のイベントの関係
Fig. 5 Relation of events in m3_monitor and m4_system.

```
inv4 : ∀n · n ∈ 1..mt ∧ (ctdSts(n) ↦ phyCnf) ∈ dom(EX)
        ⇒ monSts(n) = EX_total(ctdSts(n) ↦ phyCnf)
```

上記 inv4 は，**Requirement 2'** における EX_total の前提条件と一致する．

次に m4_system において，monitoringSystem を，モニタ表示処理に対応するシステムイベント monUpdateStatus に詳細化する．同時に，表示データ演算処理に対応するシステムイベント sysUpdateStatus を追加する，これらのイベント記述は，6.2 節に示したとおりである．これらのイベントの関係を**図 5** に示す．

このとき，以下のインバリアントが成立する．

```
inv5 : ∀n · n ∈ 1..mt ⇒ sysSts(n) = CTD_2_SYS(ctdSts(n)
↦ phyCnf)
```

また c4_system において，以下のとおり EX_total の定義を与える．axm は公理のラベルである．

```
axm1 : EX_total = CTD_2_SYS;SYS_2_MON
```

さらに c4_system では，sysUpdateStatus および monUpdateStatus が使用する関数 CTD_2_SYS および SYS_2_MON を，与えられた内部仕様に従って具体化する．ここでは例として CTD_2_SYS の具体化方法を示す．たとえば，IF-THEN-ELSE 制御構造に基づく「IF ctdSts == 0 THEN sysSts = 2 ELSEIF phyCnf == 0 THEN sysSts = 1 ELSE sysSts = 0」という内部仕様が与えられた場合，CTD_2_SYS は，上記手続きの実行後に成立するデータ制約式によって，以下のとおり具体化できる．

```
axm2 : CTD_2_INT_SGN = {ctdSts, phyCnf, sysSts ·
    (ctdSts = 0 ⇒ sysSts = 2) ∧ (¬(ctdSts = 0) ∧
    phyCnf = 0 ⇒ sysSts = 1) ∧ (¬(ctdSts = 0) ∧
    ¬(¬(ctdSts = 0) ∧ phyCnf = 0)) ⇒ sysSts = 0)
                        | ctdSts ↦ phyCnf ↦ sysSts}
```

上記 axm1 および inv5 に基づき，monitoringSystem から monUpdateStatus への詳細化の際に課せられる以下の証明責務 SIM を証明できる．

```
SIM : ∀n · n ∈ 1..mt ⇒ EX_total(ctdSts(n) ↦ phyCnf)
= SYS_2_MON(sysSts(n))
```

この証明責務 SIM を証明することで，内部仕様導入後の

表 1 証明責務の証明結果
Table 1 Statistics of the proof obligations.

モデル名称	証明責務総数	証明不要	自動証明	対話証明	対話証明率
テンプレートモデル	321		256	65	約 20.2%
インスタンスモデル	385	320	41	24	約 6.5%
合計	**706**	**320**	**297**	**89**	**約 12.6%**

m4_system においても inv4 が成立することを示せる．

続いて c5_verification において，与えられた外延的かつ部分的な外部仕様に従って EX を具体化する．たとえば，外延的かつ部分的な外部仕様が「設備状態データが 0 で物理設定が 1 の場合，モニタ表示結果は 5 である」という仕様の場合，EX は以下のとおり具体化できる．

axm3：$\mathrm{EX} = \{(0 \mapsto 1 \mapsto 5)\}$

本研究では，上記のような仕様を例として 4 つ用意し，EX を具体化した．

最後に m5_verification において，**Requirement 2'** の論理式に相当する以下の定理を導入する．thm は定理のラベルである．

thm1：$\mathrm{EX} \subseteq \mathrm{EX\_total}$

上記インバリアント inv4 と定理 thm1 を証明することで，**Requirement 2'** を検証した．

7. 証明結果

6.2 節および 6.4 節に示した検証方法に基づき，Rodin platform 上で証明責務の生成と証明を行った．その結果，対象の監視機能が **Requirement 1** および **Requirement 2** を満たすことを示せた．

本研究で実際に開発したテンプレートモデルおよびインスタンスモデルのそれぞれに対して，Rodin platform が生成した証明責務の数と，その証明結果を示す．6 章では説明の都合上，Event-B モデルを省略して説明したが，本章に示す証明結果は，実際に作成した Event-B モデルの証明結果である．「自動証明」「手動証明」の項目はそれぞれ，Rodin platform の自動証明機能によって自動証明できた証明責務の数と，Rodin platform の提供する対話証明機能を使って手動で証明した証明責務の数を表す．「証明不要」の項目は，インスタンスモデルの証明責務のうち，テンプレートモデルの証明結果を再利用できるため証明不要と判断した証明責務の数を示す．「対話証明率」の項目は，証明責務全体に対する対話証明の割合を**表 1** に示す．

上表のとおり，12.6%の証明責務を対話証明によって行った．この対話証明には約 7 人日の工数がかかった．また 6.4 節で述べたとおり，本研究では例として 4 つの仕様からなる外延的かつ部分的外部仕様を扱ったが，実際にはより多くの仕様が存在する．仕様を 1 つ追加するごとに証明責務が 1 つ追加生成されるため，仕様の数に応じて，証明責務の数は線形的に増加する．

8. 評価

本研究で検証した列車監視システムはコンソールシステムの一種であり，電力や水道など多くの分野で類似のシステムが運用されている．他分野のコンソールシステムでは，監視対象の設備が異なるため，入力データの種類やその演算方法も異なる．一方で，「外部から受信した入力データを演算し，その演算結果をモニタに表示する」という基本的な機能および構成や，「入力データに応じて漏れなく，誤りなくモニタ表示を行う」という要件は共通である．そのため本ケーススタディは，他分野のコンソールシステムを検証する際の先行事例として有用である．特に本研究では，コンソールシステムの可変点となる入力データの種類やその演算方法を抽象化してテンプレートモデルを作成した．よって，テンプレートモデルが含む抽象的なキャリア集合や定数を，コンソールシステムごとの具体的なキャリア集合および定数に置き換えるだけで，その検証のためのインスタンスモデルを得られる可能性がある．その場合，テンプレートモデル上で行った **Requirement 1** の検証結果を再利用できる．

また，4.2 節および 5.2 節に示した提案方法は，コンソールシステムに限らず一般的に有効と考えられる．4.2 節に示した補助式 P3 の発見方法では，P3 が関係する 2 つの論理式 $\Box(\mathrm{P3} \Rightarrow (\mathrm{P3}\ \mathit{Until}\ \mathrm{P2}))$，および $\Box(\mathrm{P1} \wedge \neg\mathrm{P2} \Rightarrow \mathrm{P3})$ が，対象システムの性質として成立するように P3 を構成する．これらの論理式が対象システムにおいて成立するかを判断するためには，P1 や P2 に含まれる変数や定数に関して，対象システムの振舞いを把握できればよい．そのため，対象システムにおける上記変数の値遷移を分析し，得られた遷移上において，上記論理式が成立するように P3 を構成する．変数の値遷移は，当該変数に代入を行うイベントを参照することで把握できる．上記手順により，本提案方法は $\Box(\mathrm{P1} \Rightarrow \Diamond\mathrm{P2})$ の形式の Progress 要件を持つ，任意のシステムの検証に有効と考えられる．同様にバリアント V の作成方法も，Progress 要件を持つ任意のシステムの検証に有効である．また，文献 [15] では Existence 要件（$\Box\Diamond\mathrm{P}$），および Persistence 要件（$\Diamond\Box\mathrm{P}$）の検証方法を示

しているが，これらの検証でもバリアント V が必要になる．よって本稿で提案したバリアント作成方法は，これらの要件を検証する際にも有効である．一方で本方法の適用の際には，検証者が仮バリアントを発見できることが条件となる．6.2 節に示した本事例のように，バリアント V の作成段階になって仮バリアントが不適切であると判明し，他の仮バリアントを再作成しなければならない場合もある．ただし，仮バリアントの作成は 1 つのシステムイベントに注目して行うため，すべてのシステムイベントを考慮する必要のあるバリアント V の作成よりは容易である．

5.2 節に示した仕様の無矛盾検証方法は，外部環境からの入力を受け付けて出力を行うシステムのうち，外延的かつ部分的な外部仕様と内包的な内部仕様の双方の仕様によって定義されるシステムに有効である．特に，コンソールシステムの開発では，入力データの組合せや条件分岐の多さから全域的に外部仕様を定義できず，外延的かつ部分的に外部仕様を定義することが多い．そのため，本方法が有効となる．本方法ではまず，内部仕様と一致する理想的な仮想外部仕様 EX_total を導入する．次に，外延的かつ部分的な外部仕様 EX の定義域内において，内部仕様に従って出力されたモニタ表示と上記仮想外部仕様 EX_total が一致することを証明する．証明の範囲を EX の定義域内に限定する理由は，EX は部分関数であり，EX の定義域外は仕様未定義により検証範囲外となるためである．そして最後に，EX が EX_total に含まれることを証明する．この方法により，EX の定義域内においては，EX は EX_total と無矛盾であり，同定義域内において EX_total 一致する内部仕様とも無矛盾であることを示せる．

また本研究では，仮想外部仕様として EX_total = CTD_2_SYS;SYS_2_MON となる EX_total を採用したが，**Requirement 2'** の前提条件を満たすなら，どのような仮想外部仕様でもかまわない．たとえば EX_total' ≠ CTD_2_SYS;SYS_2_MON となる EX_total' を仮想外部仕様として採用した場合でも，提案方法による検証が可能である．EX_total' は **Requirement 2'** の前提条件を満たすため，EX の定義域内においては，内部仕様に従って出力されたモニタ表示結果と一致する．このとき，EX が EX_total' に含まれるならば，少なくとも EX の定義域内においては，EX は EX_total' と無矛盾であり，同定義域内において一致する内部仕様とも無矛盾である．本研究では EX の定義域外でも内部仕様と一致するような仮想外部仕様 EX_total を採用したが，EX の定義域外は検証範囲外であるため，必ずしも仮想外部仕様と内部仕様が一致する必要はない．ただし，上記 EX_total' を仮想外部仕様として採用した場合は，6.4 節の証明責務 SIM の証明に axm1 を使用できないため，EX_total' に関する別の公理から証明責務 SIM を証明しなければならない．

また本研究で作成した Event-B モデルでは，システム・装置間の通信の遅れを無視した．通信の遅れによって入力データの誤りや送受信周期の不一致が生じた場合は，本稿で検証した要件を満たさない可能性が高い．今後通信の遅れを考慮する場合は，その場合に求められる要件について検討する必要がある．

9. 関連研究

1 章で述べたとおり，Event-B による検証では，先行事例の模倣が有効である．Event-B のケーススタディとして，文献 [7], [8], [9], [10] などが報告されているが，本稿のようにコンソールシステムに属するシステムを対象としたケーススタディは見当たらない．そのため本ケーススタディは，Event-B によるコンソールシステムの検証方法を示唆する先行事例として位置づけられる．

また 5.1 節でも触れたとおり，従来から B メソッド [11] は，全域的な外部仕様（本稿における仮想外部仕様）とその実現のための手続き仕様（本稿における内部仕様）との無矛盾性を検証する手法を提供しており，近年では Event-B による方法 [10], [18], [19] も提案されている．これに対し，本稿では，従来の手法では検証できない外延的かつ部分的な外部仕様を対象とする検証方法を提案した．本提案方法では，従来手法の対象とする全域的な外部仕様を仮想外部仕様として導入し，従来の段階的詳細化を用いて仮想外部仕様と内部仕様との無矛盾性を検証する．その後に，外延的かつ部分的な外部仕様が，仮想外部仕様に含まれることを証明する．つまり本提案方法を適用する際には，従来の段階的詳細化に基づくモデル化のノウハウや，既存の事例などを活用できる．段階的詳細化を用いることで，モデルを見通し良く記述し，証明すべき性質をより自動証明しやすい性質に分割できるという利点もある．

本稿で示した検証方法は，Rodin platform の生成する証明責務を証明することで，Rodin platform 上で統一的に要件を検証するという方針に基づく．この方針にこだわらない場合，活性の検証にはモデル検査ツール [20] を使用するという選択肢も考えられる．モデル検査では，仕様規模の増加にともなって検査対象の状態数が指数関数的に増加（状態爆発）するため，実用的な時間内に検査が終了しない可能性がある．あるいは，計算機のメモリ不足によって検査が強制終了する場合もある．これに対して文献 [21] では，Event-B で記述した仕様にモデル検査を適用する際に有効となる，仕様抽象化方法を提案している．このような抽象化方法を用いることで，モデル検査で検証可能な Event-B 仕様の規模が広がる．また，外延的な外部仕様と内部仕様との無矛盾性検証手段として，アニメータ [22] を活用する方法も考えられる．アニメータを使ってモデルを実行し，その実行結果が，外延的に定義された外部仕様と一致するかを確認する．ただしアニメータを使用する場合，外延的に与えられる外部仕様の数と同じ回数だけモデルを実行す

ることになるため，モデル実行を自動化できることが望ましい．

また，多くのコンソールシステムでは，入力データの受信からモニタ表示までの実時間制約が存在する．たとえば本稿で対象とした列車監視システムの場合，設備状態データを受信してから 1 秒から 2 秒程度の時間内にモニタ表示を完了することが求められる．しかし上記のような実時間制約は Event-B では直接検証できない．一方でモデル検査を用いる場合は，実時間に関する制約を検証するための手段として，時間オートマトン [23] に基づく手法を活用できる可能性がある．

10. まとめ

列車監視システムの監視機能を，Event-B で検証した．本稿では，まず対象の監視機能に求められる要件を定義し，その検証にあたっての課題を示した．そして，課題解決のための以下の方法を示した．まず活性の検証のため，変数の状態変化を分析することで補助式 P3 を作成する方法を示した．同様に，イベントごとに仮バリアントを作成し，適切な係数を設定して足し合わせることでバリアント V を作成する方法を示した．また外延的かつ部分的な外部仕様の検証のため，全域的な外部仕様を仮想的に導入し，リファイメントに基づいて検証する方法を示した．そして上記の方法に基づいて対象の監視機能を Event-B で形式モデル化し，対象がその要件を満たすことを検証した．

最後に今後の課題を述べる．8 章で述べたとおり，本研究では通信の遅れを無視したため，通信の遅れを考慮した場合の要件の設定と，モデル作成および検証が今後の課題となる．また本稿で提案した方法を，他システムの検証において適用，評価することも今後の課題である．

謝辞 本研究を行う機会を与えていただいた株式会社日立製作所インフラシステム社武澤隆之氏，中野利彦氏，大久保訓氏，および本研究を進めるにあたりご指導，ご協力いただいた株式会社日立製作所横浜研究所宮崎邦彦氏，三部良太氏，三村昌弘氏，スイス連邦工科大学チューリッヒ校 Andreas Fürst 氏をはじめとする関係者の皆様に深く感謝の意を表する．

参考文献

[1] Behm, P., Benoit, P., Faivre, A. and Meynadier, J.-M.: M'et'eor: A Successful Application of B in a Large Project, *FM'99*, Wing, J., Woodcock, J. and Davies, J. (Eds.), Vol.1, LNCS 1708, pp.369–387, Springer-Verlag (1999).

[2] Badeau, F. and Amelot, A.: Using B as a High Level Programming Language in an Industrial Project: Roissy VAL, *ZB 2005*, Treharne, H. et al. (Eds.), LNCS 3455, pp.334–354, Springer-Verlag (2005)

[3] Lecomte, T., Servat, T. and Pouzancre, G.: Formal Methods in Safety-Critical Railway Systems, Formal Methods in Safety-Critical Railway Systems, *10th Brasilian Symposium on Formal Methods* (2007).

[4] 寺田夏樹，高橋一裕，荻野隆彦：段階的詳細化に基づくシステムの高信頼化技法，信学技報，Vol.97, No.98, DC2005-65 (2005).

[5] 寺田夏樹：B メソッドによる単線自動閉そく装置の検証，電子情報通信学会技術研究報告，DC，ディペンダブルコンピューティング，Vol.109, No.334, pp.31–36 (2009).

[6] Zafar, N.A.: Formal Model for Moving Block Railway Interlocking System Based on Un-Directed Topology, *IEEE-ICET 2006 2nd International Conference on Emerging Technologies Peshawar* (2006).

[7] DEPLOY Deliverable D16, D2.1 Pilot Deployment in Transportation (WP2), Public Document (2009).

[8] Su, W., Abrial, J.-R., Huang, R. and Zhu, H.: From Requirements to Development: Methodology and Example, *13th International Conference on Formal Engineering Methods* (2011).

[9] Metayer, C. and Clabaut, M.: DIR 41 Case Study How Event-B Can Improve an Industrial System Specification, *Abstract State Machines B and Z, 1st International Conference*, LNCS 5238 (2008).

[10] Abrial, J.-R.: *Modeling in Event-B: System and Software Engineering*, Cambridge University Press (2010).

[11] Abrial, J.-R.: *The B-Book: Assigning Programs to Meanings*, Cambridge University Press (1996).

[12] Barradas, H.R. and Bert, D.: Specification and Proof of Liveness Properties under Fairness Assumptions in B Event Systems, *Proc. 3rd International Conference on Integrated Formal Methods* (2002).

[13] Mosbahi, O. and Jaray, J.: Specification and Proof of Liveness Properties in B Event Systems, *2nd International Conference on Software and Data Technologies 2007* (2007).

[14] Hoang, T.S., Kuruma, H., Basin, D. and Abrial, J.-R.: Developing Topology Discovery in Event-B, *Science of Computer Programming*, Vol.74 (2009).

[15] Hoang, T.S. and Abrial, J.-R.: Reasoning about Liveness Properties in Event-B, *Proc. International Conference on Formal Engineering Methods 2011* (2011).

[16] Event-B.org: Event-B and the Rodin Platform, available from ⟨http://www.event-b.org/⟩ (accessed 2011-11-10).

[17] Silva, R. and Butler, M.: Supporting Reuse of Event-B Developments through Generic Instantiation, *11th International Conference on Formal Engineering Methods: Formal Methods and Software Engineering* (2009).

[18] Rezazadeh, A., Evans, N. and Butler, M.: Redevelopment of an Industrial Case Study Using Event-B and Rodin, *British Computer Society Formal Aspects of Computing Science Specialist Group Christmas 2007 Meeting, Formal Methods In Industry* (2007).

[19] Hallerstede, S. and Leuschel, M.: Experiments in program verification using Event-B, *Formal Aspects of Computing*, pp.1–29 (2011).

[20] Leuschel, M. and Butler, M.: ProB: An Automated Analysis Toolset for the B Method, *J. Software Tools for Technology Transfer* (2003).

[21] 中島 震，來間啓伸：Event-B デザインのモデル検査における抽象化，情報処理学会研究報告，EMB, 組込みシステム，Vol.2008, No.55, pp.81–88 (2008).

[22] Métayer, C.: available from ⟨http://www.animb.org/index.xml⟩, AnimB Homepage (2012).

[23] Bengtsson, J. and Yi, W.: Timed Automata: Semantics, Algorithms and Tools, *Lecture Notes on Concurrency and Petri Nets*, LNCS 3098, Springer-Verlag (2004).

付　　録

A.1 Requirement 2 と Requirement 2' の同値性の証明

まず **Requirement 2'** が成立するとき, **Requirement 2** が成立することを証明する．**Requirement 2'** より, EX の定義域に含まれる mt 回以内の入力データに対して EX_total を適用した結果は，モニタ表示結果と一致する．また，関数 EX は全域関数 EX_total の部分集合であるため，EX の定義域に含まれる任意の引数に対して EX を適用した結果は，EX_total を適用した結果と一致する．以上より，EX の定義域に含まれる mt 回以内の入力データに対して EX を適用した結果は，モニタ表示結果と一致する．

同様に，**Requirement 2** が成立するとき，**Requirement 2'** が成立することを証明する．**Requirement 2** より，EX の定義域に含まれる mt 回以内の入力データに対して EX を適用した結果は，モニタ表示結果と一致する．ここで，ある任意の全域関数 EX_total を与え，EX の定義域に含まれる mt 回以内の入力データに対して EX_total を適用した結果は，モニタ表示結果と一致するとする．このとき，EX の定義域に含まれる mt 回以内の入力データに対して EX を適用した結果は，EX_total を適用した結果と一致する．EX_total は全域関数であるため，EX は EX_total の部分集合といえる．□

佐藤 直人　（正会員）

2003 年東京工業大学情報工学科卒業．2005 年同大学大学院情報理工学研究科計算工学専攻修士課程修了．同年，株式会社日立製作所入社．モデルベース開発技術，形式手法等の研究開発に従事．

タイ ソン ホアン

スイス連邦工科大学チューリッヒ校情報科学科情報セキュリティ専攻所属．シニアリサーチャー．2006 年豪ニューサウスウェールズ大学にて博士号取得．形式モデリングおよび形式検証技術の研究に従事．Event-B 向けツール RODIN Platform の開発メンバとしても活動し，同ツールのプラグインを複数開発．

デビッド ベイジン

2003 年より，スイス連邦工科大学チューリッヒ校情報科学科情報セキュリティ専攻主任，教授．チューリッヒ情報セキュリティセンタ（ZISC）ディレクター．1989 年に米コーネル大学にて博士号取得．1996 年には，独ザールブリュッケン大学に所属．情報セキュリティ技術，特に，高信頼・高安全システムのモデリングおよび検証技術の研究に従事．

來間 啓伸　（正会員）

1958 年生．1981 年広島大学理学部物理学科卒業．1984 年同大学大学院理学研究科物理学専攻博士課程後期中退，同年（株）日立製作所入社．現在，同社横浜研究所に所属．主として，ソフトウェア工学，形式手法の研究に従事．学術博士（総合研究大学院大学）．日本ソフトウェア科学会，ACM，IEEE 各会員．