15-B-2

通信制御ソフトウェア開発における状態遷移設計の品質向上への取組み [1]

～状態遷移表へのモデル検査の適用～

1. 背景と取り組みの概要

ネットワーク技術の進展や通信ネットワークの利用の拡大に伴い、通信ネットワークを構築する通信制御ソフトウェアには顧客から多機能、高品質、短納期といった厳しい条件が要求され続けている。通信制御ソフトウェアのこれまでの開発では実績のある大量の既存資産を流用（派生開発）して、その資産に機能追加や変更を行うことで対応してきた（図 15-B-2-1）。

通信制御ソフトウェアでは、通信制御に係る「状態遷移設計」が重要であるが、複数のノード、通信プロトコル、機能ブロック間の連携増加などにより、その設計がより複雑化し、また、検証（レビュー／試験）においても、複雑な遷移ルートに伴う問題をすべて検出することが困難なことから、問題が下流工程に流出し、品質安定までに多くの時間を要するなどの課題が明らかになった。

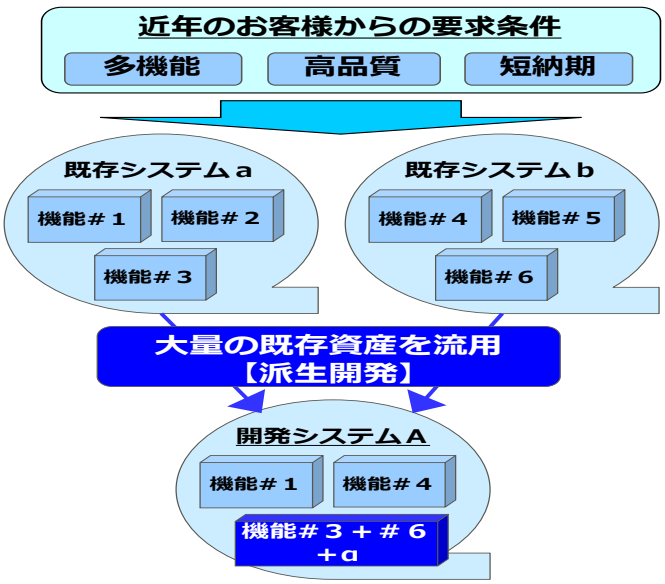


図 15-B-2-1　既存資産流用開発

[1] 事例提供: 富士通株式会社 共通開発本部　安岡 大知 氏、
ネットワークソリューション事業部　三橋 崇 氏、松下 亮太郎 氏

これらの課題を解決し、より短期間で高品質なソフトウェアの提供を可能とするため、設計工程で作成した状態遷移表に形式手法モデル検査を適用し､網羅的に検査することとした。具体的には、品質確保のための開発プロセスに「形式手法モデル検査ツール SPIN：Simple Promela Interpreter（以下、SPIN)」を導入し、形式手法モデル検査を実行した。

本事例では上記の通信制御ソフトウェアの複雑化した状態遷移表の検証のために導入した SPIN ツールの適用手法について紹介する。導入に際して発生した課題ならびにその対策を表 15-B-2-1 に示し詳細は「3.適用した技術や手法をスムーズに導入し活動を定着させるための、事前準備や工夫等」に記す。

導入体制はプロジェクトの 90%程度は実作業に従事し、10%程度の要員で SPIN 検査支援ツールや SPIN の対応を実施した。SPIN 検査支援ツールの開発は、専任で 2 人、2 ヶ月程度で従事した。

表 15-B-2-1　SPIN 導入時の課題と対策

項番	課題	対策
1	Promela 言語／LTL 式／反例解析といった専門スキルの修得や SPIN の環境構築/操作方法の修得が必要	Promela 言語の自動生成、LTL 式の自動生成、反例解析支援､モデル検査 Web システムを実現する SPIN 検査支援ツールを作成し、特別な知識・スキルなしに SPIN の適用を可能とした
2	膨大な遷移ルート検査において検査不能となる状態爆発が発生する	①モデル検査ターゲット（検査の着眼点）の決定、②状態遷移表の分割、③データ種別による分割等を実施した
3	SPIN 適用における費用対効果が不明確	状態遷移表の各種パラメータと、過去の SPIN 検査実績を元に、検査対象の状態遷移表の複雑度を点数化するツールを作成し、適用効果や適用難易度(例えば､状態爆発の対策が必要かどうか)を事前に把握可能とした

形式手法モデル検査ツール SPIN 選定理由について

モデル検査はシステムから導出されたモデルが仕様を満足するかどうかをアルゴリズム的に検証する手法である。モデル検査にも多くのツールが存在するが、通信制御ソフトウェアでは大量のイベント処理や状態制御を行うことからその主要な開発成果物は状態遷移表であるという特徴がある。

「SPIN」は、状態遷移表と検査観点を入力すると全遷移ルートが検証観点に合致しているかを検査器にて自動検証が可能なツールであり､状態遷移表との親和性やツールの安定度、速度、完成度の高さから選定した。

2. 手法の概要と導入に至った理由や経緯

ここでは、通信制御ソフトウェアの開発手法の概要を示すとともに、形式手法モデル検査ツール SPIN の開発導入に至った状態遷移設計における課題について示す。

2.1. 通信制御ソフトウェアの開発手法

通信制御ソフトウェアでは、複数の通信プロトコルの信号受信や各種タイマ満了（イベント）を契機に通信制御ソフトウェアの状態を変化させるため、この振る舞いを決定するため状態毎に処理を決定する「状態遷移設計」が重要となる。

「状態制御」は、多くの通信プロトコルを扱うことや、１つの通信プロトコルを他の複数のプロトコルに変換するなどにより、「大量」かつ、「複雑」な構成となる。状態制御により実現される通信プロトコル制御の概要を図 15-B-2-2 に示す。

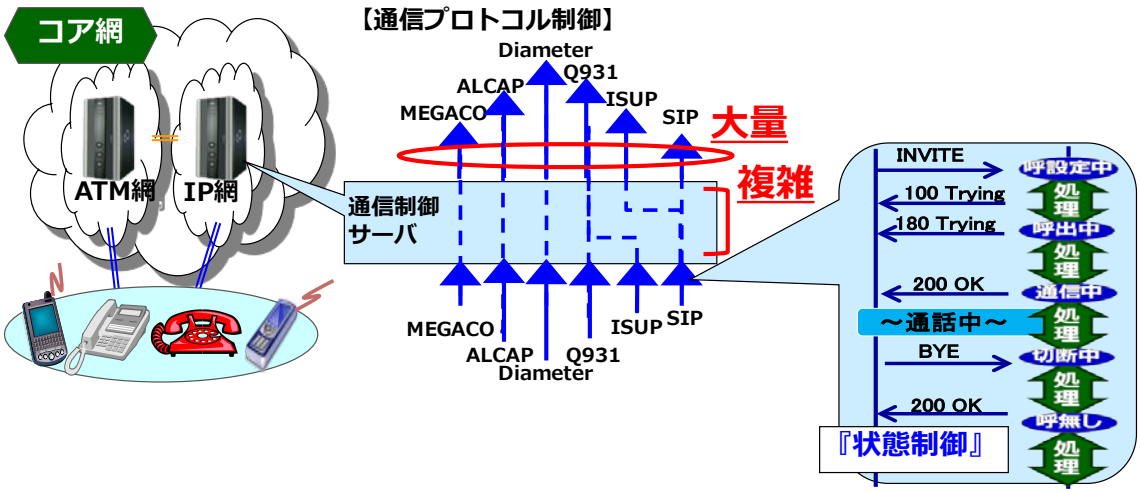


図 15-B-2-2　状態制御により実現される通信プロトコル制御

このような特徴を有する通信制御ソフトウェアにおける状態遷移設計では、状態制御の網羅性を担保するため状態遷移表を用いている。

2.2. 状態遷移設計における課題と解決策

状態遷移表は状態制御の網羅性を担保しやすい反面、規模が大きくなりやすい。実際に作成してきた状態遷移表は大きいものになると、状態とイベントの組み合わせが数万以上となる。このように大きな状態遷移表を設計工程で多くの時間をかけて新規作成、あるいは、派生開発による追加・変更をしているが、状態遷移表の検証（レビュー／試験）では、複雑な遷移ルートに伴う全ての問題混入に対し検出が困難な傾向にあり、問題が下流工程に流出し、品質安定までに多くの時間を要していた。

このため、SPIN を導入し、状態遷移表の効率的／網羅的検証を実施することで、上記の課題を解決することとした。

3. 適用技術・手法の導入と活動定着のための事前準備や工夫

3.1. SPIN プロジェクト適用の課題

SPIN による状態遷移表の検査作業と、各作業における課題を図 15-B-2-3 に示す。SPIN 検査では、大きく以下の作業が必要となる。

a) 仕様から状態遷移表を作成し、その状態遷移表が実現する動作モデルを SPIN が解釈可能な Promela 言語へ手動で変換する。

b) 仕様から状態遷移表の検査観点を抽出し、検査観点から LTL 式を作成する。
SPIN では、状態遷移表について線形時相論理式（以下、LTL 式）による検査が可能である。LTL 式による検査とは、与えた条件が常に、あるいはいつか必ず真となるといった時間軸上で命題の真偽を記述できる論理を使用し、矛盾がないかどうかを確認することで状態遷移表の不備を検出することである。

c) 作成した Promela 言語と LTL 式を SPIN に入力し、SPIN を実行する。

d) 実行の結果、動作モデルが検査観点を満たさない場合、反例が出力され、反例を元に状態遷移ルートを解析し、不具合があれば状態遷移表にフィードバックする。

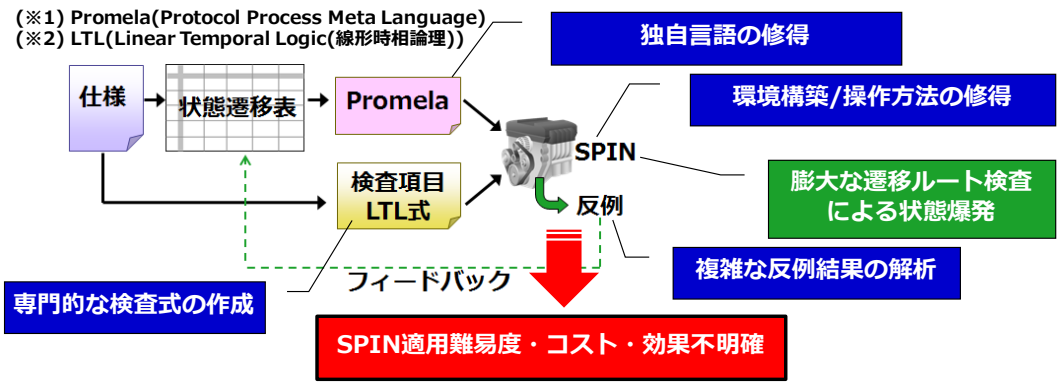


図 15-B-2-3　SPIN 作業上の課題

これら作業をプロジェクトで実行するうえで、以下の課題に直面した。

① Promela 言語／LTL 式／反例解析といった専門スキルの修得や SPIN の環境構築／操作方法の修得
⇒専門スキルの修得などが必要である場合、手法の普及に阻害となる

② 膨大な遷移ルート検査において検査不能となる状態爆発（状態爆発防止）
⇒探索すべき状態数が膨大な場合、SPIN ツールの動作が完了しない

③ SPIN 適用における費用対効果が不明確（費用対効果の事前把握）
⇒SPIN による費用対効果が不明である場合、手法の適用阻害となる。

これら課題について、解決方法を以下 3.2 に記述する。

3.2. 課題解決方法

3.2.1. 専門スキルの修得や SPIN の環境構築／操作方法の修得

SPIN のプロジェクト適用において、専門スキルやノウハウの修得不要とする検査手順を確立することを目的として、Promela 言語の自動生成、LTL 式の自動生成、反例解析支援、モデル検査 Web システムを実現する SPIN 検査支援ツール(図 15-B-2-4 を作成した。以下に検査支援ツールの概要を示す。

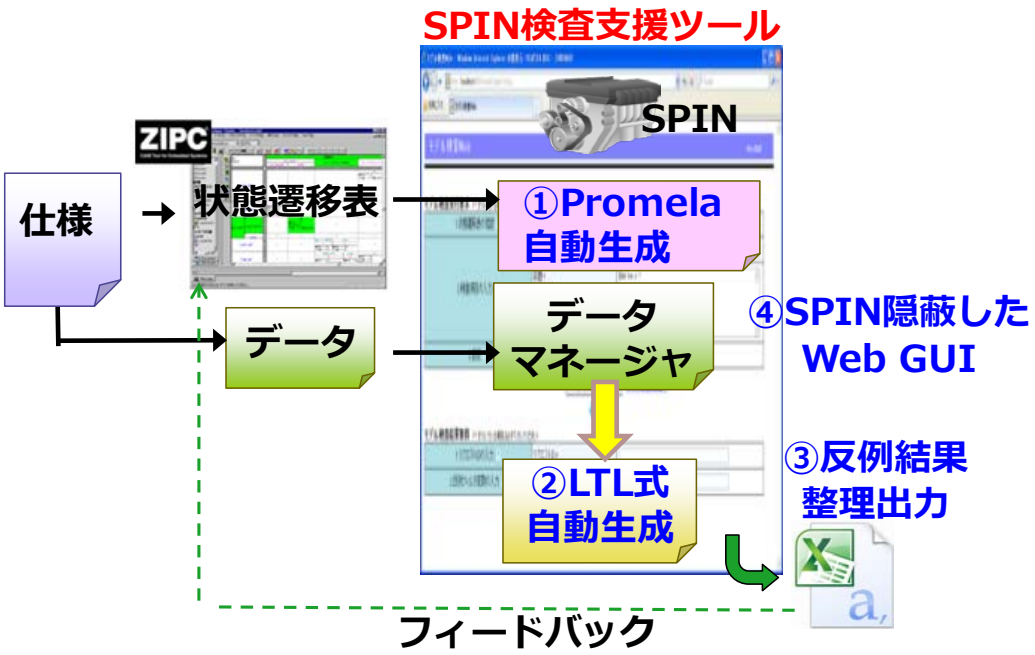


図 15-B-2-4　SPIN 検査支援ツール

(1)　Promela 言語の自動生成

各プロジェクトの状態遷移表は、共通の記述ルールがないため、記述の仕方がそれぞれ異なり、Promela 言語を自動生成する阻害要因となっていた。そこで、自動生成のためのツール処理の簡素化として、市販の状態遷移設計支援ツールを導入し、状態遷移表の記述レベル統一（定型化）を行った。これにより、定型化した状態遷移表からツールによって Promel 言語の自動生成は現実的な工数で対応可能となり、Promela 言語を記述するためのスキル修得を不要とした。

(2)　LTL 式の自動生成

各種状態遷移処理や、変数の操作を検査するための LTL 式の雛形を作成した。具体的な状態定義や変数等のデータはプロジェクト毎に異なるため、検査時にプロジェクト個々のデータ定義情報を入力することで、LTL 式の雛形にデータ定義が補完され、LTL 式の自動生成を実現し、LTL 式を記述するためのスキル修得を不要とした。

（3）　反例解析支援

SPIN が出力する反例ログについて状態／イベント等の情報をインデックスにしてログを整理し、検査実行の結果、動作モデルが検査観点を満たさないポイントに至るまでの状態遷移ルートを明確にして CSV ファイルに出力することで、可読性が向上し反例結果の効率的な解析ができるようになった。

（4）　モデル検査 Web システム

SPIN によるモデル検査を開発プロセスに導入する仕組みとしてモデル検査 Web システムを構築した。簡単な GUI 操作によるモデル検査 Web システムによって、SPIN の環境構築や SPIN 操作が隠ぺいできるようになった。このシステムに Web アクセスし、状態遷移表とデータ定義を入力すると Promela 言語生成、LTL 式の自動生成、SPIN 検査実行、反例結果出力まで自動で行われ、誰でも・いつでも簡単に検査が可能となった。

3.2.2. 状態爆発防止

状態爆発とは、例えば状態遷移表の検査において、探索すべき状態数が膨大な場合、モデル検査を実行するコンピュータの記憶容量の限界によって検査不能となることである。

細かい粒度の状態遷移表をモデル検査すると状態爆発により検査不能となるため、状態・イベント・データの粒度を粗くしたり、不要なイベント・状態・データを削除するといった状態遷移表の抽象化を図ることで状態爆発の防止を図った。

しかし、抽象化によって状態爆発は回避されるが、肝心の状態遷移表の問題（反例）が見つからない傾向にあった（図 15-B-2-5）。

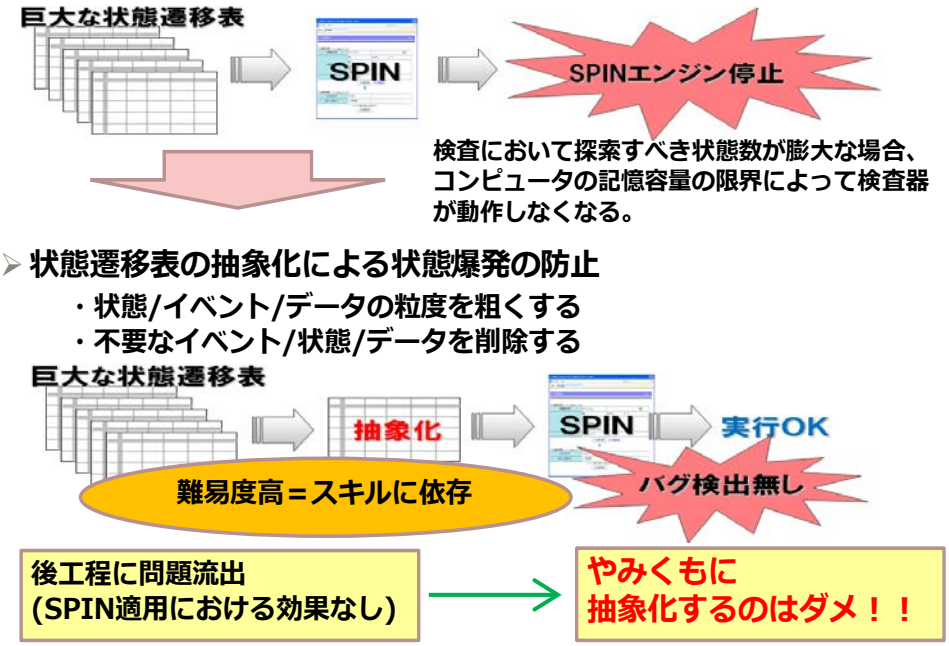


図 15-B-2-5　状態遷移表の爆発

そこで、SPIN をプロジェクト適用する上で、状態爆発の課題の解決のため、①モデル検査ターゲットの決定、②状態遷移表の分割、③データ種別による分割、に取り組んだ。

① モデル検査ターゲットの決定

モデル検査ターゲットとは、検査を実施する際の着眼点（データ操作等）のことである。モデル検査ターゲット決定にあたり、各種通信制御ソフトウェアの状態遷移設計に関する流出問題分析を行った結果、状態切り替え時のデータ操作誤り、特にデータの二重設定や二重解放、未設定データの参照といった問題が問題原因の大半であることが分かった。

このことから、状態遷移表の記載すべてを検査するのではなく、モデル検査ターゲットとして、特にデータ操作にターゲットを絞ることにした。ここで言うデータ操作とは、ソフトウェアで扱う受信イベント情報、リソース、タイマ等を指している。これにより、状態遷移表の記載内容・粒度が定まり、状態爆発の防止と状態遷移表の問題検出の両立が可能となった（図 15-B-2-6）。

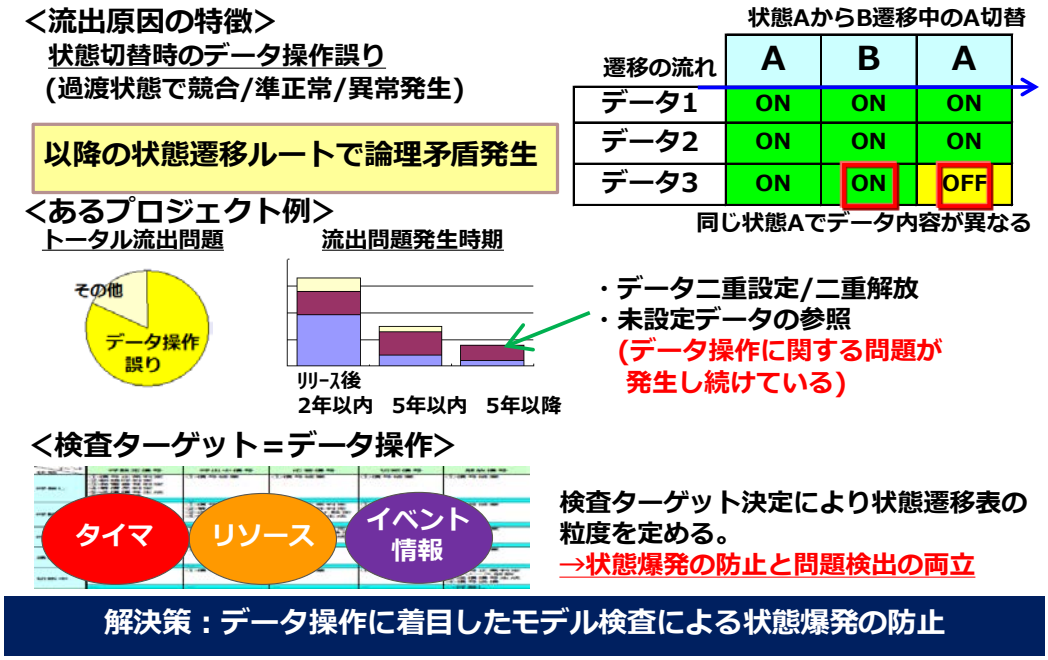


図 15-B-2-6 データ操作に着目した状態爆発の防止

ただし、状態爆発はコンピュータの記憶容量に依存するため、モデル検査ターゲットの決定が完全な解決策とはならない。そこで、モデル検査ターゲット決定に加えて実施することが望ましい対策を以下に記載する。

②　状態遷移表の分割

データ操作を検査するにあたって、１つのシステムの全てのデータ操作を一度に検査するのではなく、状態遷移表をノードや機能ブロック等に分割してそれぞれを検査することで、状態爆発が回避できる場合がある(図 15-B-2-7)。

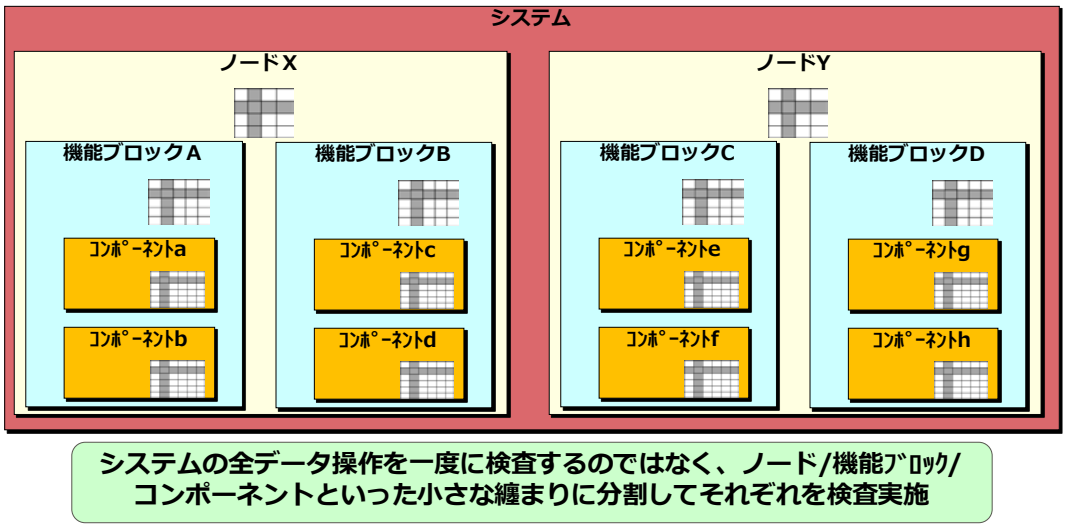


図 15-B-2-7　状態遷移表の分割

③　データ種別による分割

状態遷移表を分割しても状態爆発が回避できない場合、分割した状態遷移表をデータ種別毎に分割してそれぞれを検査することで、状態爆発が回避できる場合がある。

例えば、図 15-B-2-8 に示すように、１つの状態遷移表のデータ操作を「タイマ」、「リソース」、「受信イベント情報」に分類し、分類したデータ毎に状態遷移表を記述し、検査することで、状態爆発が回避できる場合がある。

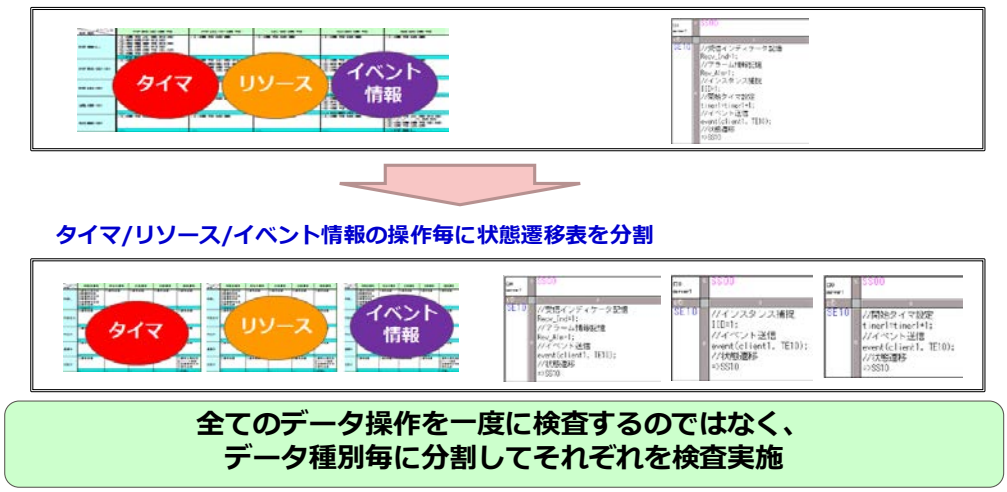


図 15-B-2-8　データ操作の分割イメージ

3.2.3. 費用対効果の事前把握

SPIN は、ある程度複雑な状態遷移設計に適用した際に、その効果（トータル工数削減）を発揮する。このため、SPIN を初めて適用するにあたっては、適用により発生する工数を還元できる効果が得られるか事前に確認していく必要がある。そのため、状態遷移表の各種パラメータと、過去の SPIN 検査実績を元に、検査対象の状態遷移表の複雑度を点数化するツールを作成した。

過去の SPIN 検査実績からおおよその SPIN が効果を発揮するあるいは状態爆発が発生する状態遷移表サイズや複雑度を情報として蓄積している。この２つの要素を決定するのは、イベント数、状態数、階層数、状態遷移表間の遷移数、アクションセル内分岐数、アクションセル内データ数といったパラメータが関係する。この各種パラメータを数値入力することで状態遷移表サイズや複雑度が蓄積した情報にどれだけ近いか、あるいは、どれだけ遠いかを状態遷移表のスコアという形で出力するツールを作成した。

このツールを SPIN 適用前に使用することで、適用効果や適用難易度（例えば、状態爆発の対策が必要かどうか）を事前に把握可能とした。

4. 適用した技術や手法の効果測定の方法とその結果

SPIN 適用事例として、各種状態遷移ルートで多種のタイマを設定・停止する仕様を持つソフトウェアについて、設定・停止の妥当性検証に人手でのレビューに加えて SPIN を活用した検証を行った。この適用結果を図 15-B-2-9 に示す。SPIN を適用しなかった場合、設計工程で問題が潜在し、下流工程以降に流出していたが、これを未然に取り除くことができた。

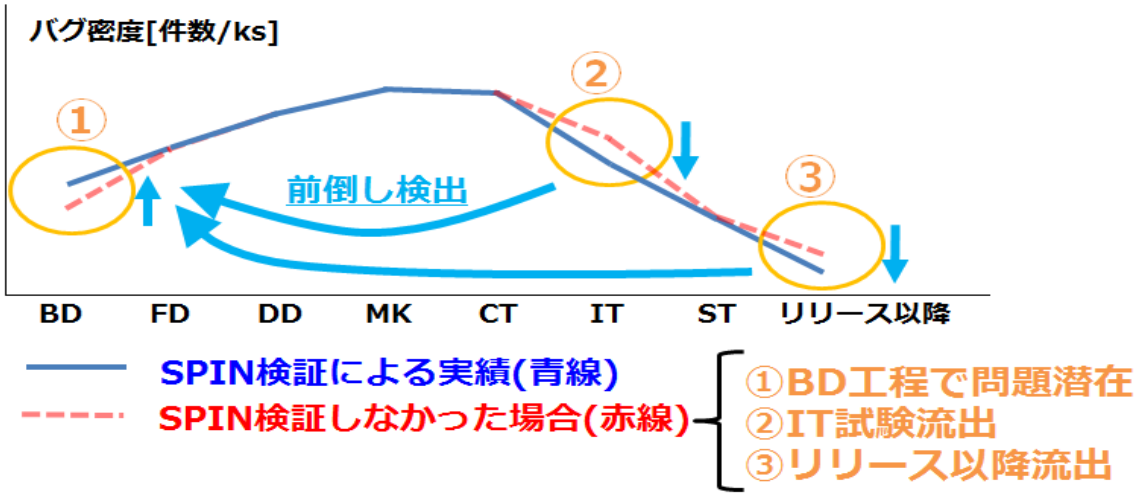
これにより、設計工程での SPIN 検査適用の有効性を確認することができ、上流工程において品質の作り込みができたと考える。特に ST（System Test）までで検出できなかったと想定される問題の設計工程検出は大きな成果と考える。

この適用事例では上流工程の BD（Basic Design)工程では 10%程度検出バグ件数が増加するが、下流工程の IT（Integration Test）工程においては 10～20%程度バグ件数が減少し、リリース以降については 50%程度バグが減少した。

人手による検証(レビュー)では検出することが難しい、複雑な状態遷移ルートに伴う混入問題の検出に効果があると考える。

検証段階で問題を検出するとその解決コストが大幅に増加する。これもモデル検査の手法を適用することで、検証段階での問題は減少し、工数も減少する。設計工程での SPIN 検証に伴い追加コストが発生するが、トータルでの期間短縮に寄与することが可能になる。

SPIN を活用することで品質、生産性ともに効果があることを確認できた。



BD(Basic Design),FD(Function Design),DD(Detail Design),MK(MaKe),CT(Code Test)、IT(Integration Test),ST(System Test)

図 15-B-2-9　SPIN 検証の適用有無によるバク密度の比較

5. 現場への普及状況と適用に関する注意点

現場への普及については、社内で SPIN の適用を発表して普及活動を進めている。また、プロジェクトのメンバーへ個別に SPIN の説明を十分に実施している。

SPIN 適用にあたっては、品質問題に直面している、あるいは、直面するリスクのあるプロジェクトに有効であると考えるが、適用にあたって検査ターゲットを明確にしないと効果を得られにくいと考える。

6. 適用後の課題と水平展開状況

今後の活動としては、SPIN や SPIN 検査支援ツールの改善よりは、SPIN を適用するプロジェクトを増やすための普及活動を進めている。

今後の課題として、以下のような問題が顕在化する可能性があると考えている。

1 点目は、例えば検査対象の状態遷移表のイベント定義を書き漏らした場合、漏れたイベント自体を SPIN では検出するのは不向きである。この対策としては、SPIN を普及するにあたっては状態遷移設計技術の普及も併せて実施する必要がある。この面の技術資料や教育資料を作成することで問題の顕在化を低減できると考える。

2 点目は、SPIN を隠ぺいした SPIN 検査支援ツールの使用は、初級開発者にとって SPIN 技術の標準化によって、検証技術力の底上げ効果となると考える。しかしながら、SPIN を隠ぺいした SPIN 検査支援ツールは検証方法を型決めしているため、SPIN を熟知している上級開発者にとっては使い勝手が悪くなる可能性がある。現時点ではモデル検査適用プロジェクトの拡大やモデル検査スキル修得が主となるため、大きく表面化していないが今後問題として顕在化する可能性がある。

次に、水平展開状況だが、現在、富士通の通信制御ソフトウェアに対して SPIN を適用す

るプロジェクトは年々拡大してきている。しかし、まだ拡大の余地は残っている。これからも改善を実施しながら適用拡大していく。

7. まとめ

通信制御ソフトウェアでの状態遷移表の複雑化や検証の困難さから、検証過程において形式手法モデル検査ツール SPIN を導入した。

SPIN の導入では、開発プロジェクトにその適用を進めるにあたって、Promela 言語などの SPIN 特有な専門スキルの習得や、大きな状態遷移表の場合 SPIN ツールによる検査が完了しない場合があること、SPIN ツールによる検証作業の費用対効果の事前把握が不明確など運用上の課題が明らかになった。

これらの運用上の課題を解決するため、SPIN 特有の専門知識がなくても SPIN による検証が可能にするための SPIN 検証支援ツールを開発し、また、状態遷移表が膨大にならないようデータ操作に着目した検査モデルを構築し、さらに、検査対象の状態遷移表の複雑度を点数化することにより事前に費用対効果を明らかにする工夫を実施した。

これらの工夫により、通信制御ソフトウェアの SPIN ツールによる検証を容易に導入することができ、以前に比べ、上流工程でより多くのバクを検出することができ、下流工程でのバグ対応を減少することにより、品質安定までの時間を短縮することができた。

本事例は、形式手法モデルによる検証ツールを効果的に開発プロジェクトに導入するための工夫とその効果について紹介したが、これにより、開発プロジェクトにおいて形式手法モデルによる検証作業の有効性が発揮できると期待される。

参考文献

[1] 富士通株式会社 安岡大知 : 通信制御ソフトウェア開発における状態遷移設計の品質向上への取り組み～ZIPC 状態遷移表のモデル検査適用への取り組み～

[2] 富士通株式会社 松下亮太郎 : ZIPCWATCHERS Vol.15 通信制御ソフトウェア開発における状態遷移表の実装効率化への取組み

[3] 富士通株式会社 三橋崇 : ZIPC WATCHERS Vol.15 通信制御ソフトウェア開発における状態遷移表の実装効率化への取組み ～ZIPC によるコード生成自動化への取組み～

[4] 産業技術総合研究所 システム検証研究センター著 : モデル検査[初級編]—基礎から実践まで 4 日で学べる

[5] 産業技術総合研究所 システム検証研究センター著 : モデル検査[上級編]—実践のための三つの技法

[6] Mordechai Ben-Ari 著 中島震 監訳 谷津弘一・野中哲・足立太郎 共訳 : SPIN モデル検査入門

独立行政法人情報処理推進機構 技術本部 ソフトウェア高信頼化センター（IPA/SEC）