Vol.2013-SE-179 No.34
2013/3/12

リスク分析手法とモデル検査を組合せた高信頼設計プロセスの提案

若林昇[†1]　吉岡信和[†2]

近年，様々な機器がネットワーク連携してシステムを形成している．特にシステムの構成要素ならびに構成要素間の関係が運用環境によって異なるようなオープンシステムが普及しようとしている．このようなシステムでは，運用時の振舞いを完全に予測することは困難であり，想定できていなかった重大な障害を引き起こし重大事故に陥る可能性がある．

本論文では，重大事故につながるシステムの脆弱性を発見する手法として，障害を事前に想定し信頼性を確保するリスク分析手法，特に FMEA と，システムの状態を網羅的に検証するモデル検査を組合せ，重大事故となる結果から脆弱性の原因を自動的に追求するアルゴリズムを使った高信頼設計プロセスを提案する．

Suggestion of the high reliable design process that combined model checking with risk analysis technique

Recently various information systems are connected through a network and become the huge information system group that provides a service. Especially open systems are going to spread, in which components of the system and the relationship with the components change by operation environments. In such a system group, it is difficult to completely predict the behavior of the system, there is possibility that an unpredicted serious failure causes a serious accident.

This paper proposes a high reliable design process that combines model checking that can verify cyclopedically with a risk analysis technique by which we can assume faults in advance and provide reliability.

NOBORU WAKABAYASHI[†1]　NOBUKAZU YOSHIOKA[†2]

1. はじめに

近年，ネットワークを介し，様々な情報システム同士が直接あるいは間接的に接続され巨大な情報システム群となってサービス提供されている．このようなサービスでは，運用開始後にシステムの仕様が変わったり，OSS（Open Source Software）のようなブラックボックス部品を用いたりすることが多い．

このようなシステムでは，下記に示す「不完全さ」「不確実さ」[1] により，出荷時や運用時のシステムの振舞いを完全に予測することは困難であり，当初想定できていなかった重大な障害が起こる可能性がある．

・不完全さ

要求に対して仕様が完全ではなく，また，仕様に対して実装が完全ではなく，出荷時や運用時のシステムの振舞いを完全に把握することが困難であること

・不確実さ

事業者や利用者の要求やシステム環境がライフサイクルを通して変化し，設計時や運用時に機能や挙動を完全に予測できないこと

このようなシステムにおいても，想定できていなかった重大な障害に至る故障を検出し対策することで，信頼性を高める高信頼設計プロセスを研究することを目的とする．

重大な障害に至る故障を引き起こすあらゆる原因を検出することは一般的に難しい．障害を事前に想定し信頼性を確保するリスク分析手法として FMEA（Failure Mode and Effect Analysis）があるが，FMEA をソフトウェアに適用する際には，「故障モードの妥当性を検証することの困難さ」「原因を解析することの困難さ」「対策案が効果的であるのかを評価することの困難さ」があり，これらの困難さを解決する必要がある．

本報告では，これらの課題を解決するために，網羅的に検証するモデル検査を FMEA と組合せて，事前に想定できていなかった重大な障害に至る故障を検出し対策することで信頼性を高める高信頼設計プロセスを提案する．

2 章で従来手法とその課題について述べ，3 章で従来手法の課題を解決する提案手法について述べ，4 章で提案手法を HEMS（Home Energy Management System）に適用した結果について述べる．5 章で関連研究について述べ，最後に 6 章でまとめを述べる．

2. 従来手法と課題

2.1 オープンシステムにおける信頼性上の課題

システムの「不完全さ」「不確実さ」により，出荷時や運用時のシステムの振舞いを完全に予測することは困難であり，事前に重大な障害に至る故障を引き起こすあらゆる原因を検出することは難しい．例えば，このようなシステムではソフトウェアの規模が大きく，多くのソフトウェアモジュールの複雑な組合せから作られており，モジュールごとに開発される．このような開発では，他のモジュールの仕様を完全に把握することが難しい．また運用中に新た

†1 (株)日立製作所　横浜研究所
Hitachi Ltd. Yokohama Laboratory
†2 国立情報学研究所　GRACE センター
National Institute of Informatics GRACE Center

なモジュールがシステムに追加されることもある．このような場合，新たなモジュールが当初想定していなかった動作を行う，或いは，何らかの原因でモジュールが故障し，当初想定していなかった動作を行い，システム全体がダウンしてしまうことがある．このような重大な障害に至る故障を引き起こすあらゆる原因を，事前に検出することは難しいと言える．

オープンシステムの例の一つとして HEMS が挙げられる．HEMS は家庭内の機器をネットワークでつなぎ，電力管理，自動制御するシステムである[2]．図 1 に示す通り，HEMS においても上記と同様の課題が当てはまる．例えば，運用中の個々の機器の故障（ハードウェアの故障，或いは，複雑なソフトウェアから起因する故障），新たな機器の追加による当初想定していなかった挙動，他社製品との連携不完全などにより，間違ったデータがコントローラに送信され，電力管理が不適切になり，停電を引き起こしてしまうといったことが起こり得る．

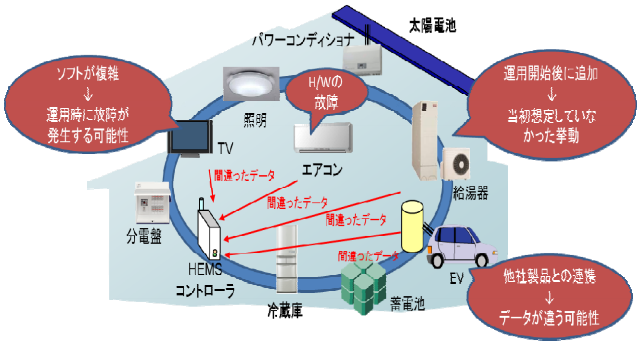

図 1　HEMS 構成と課題

2.2 従来のリスク分析手法

事前に障害を分析する手法としてリスク分析手法がある．リスク分析手法の代表的なものに FTA（Fault Tree Analysis）及び FMEA（Failure Mode and Effect Analysis）がある．FTA は障害など望ましくない事象（トップ事象）を設定し，その要因を探るトップダウンによる分析であるが，分析は人手によりなされるため，列挙された故障要因は気が付いた要因だけになりやすく，気が付かない要因は漏れてしまう恐れがある．例えば，トップ事象に関係しない下位部品が，複雑に影響し障害を引き起こすことも考えられ，この場合，FTA では検出できない．すなわち，重大な障害に至る故障を引き起こすあらゆる原因を検出することは難しい．

一方，FMEA はシステムを構成する部品を抽出し，各部品で起こり得る障害の要因（故障モード）を列挙し，上位機能に及ぼす影響を検討するボトムアップによる分析であるため，検討漏れが起こりにくい．そのため，重大な障害に至る故障を引き起こすあらゆる原因を検出しやすい手法だと言える．そこで本研究では，設計工程でリスク分析手法である FMEA を用いることが有効であると判断し，FMEA を用いることにする．

2.3 FMEA のソフトウェア適用における課題

2.2 節で述べた通り，重大な障害に至る故障を引き起こすあらゆる原因を検出しやすい手法である FMEA を用いることは，オープンシステムの信頼性を向上する上で有効である．

しかし，FMEA はハードウェアに対する分析手法として考案されたものであり，大規模なソフトウェアに適用する場合は，大規模ソフトウェアが持つ複雑性などにより故障モードを抽出しにくいといった課題がある．以下，FMEA の一般的な手順を示し，ソフトウェア適用における課題を挙げる．

図 2 に FMEA の一般的な手順を示す[3]．

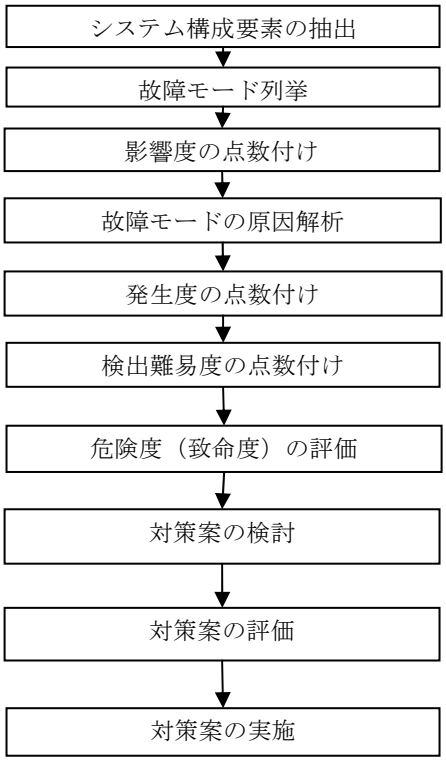

図 2　FMEA の手順

まずシステムを構成する要素（部品）を抽出し，各要素に対して故障モードを列挙していく．この際，前述したように大規模ソフトウェアが持つ複雑性などにより故障モードを列挙することが難しく，列挙できたとしても，図 3 に示す通り列挙した故障モードが重大な影響を及ぼす障害に至らない場合もあるが，故障から障害に至るパスが複雑で長いことがあり，その故障モードが重大な影響を及ぼす障害に至るか判断することが難しい．

故障モードが列挙できると，その故障モードの影響解析及び原因解析を行うが，ソフトウェアの場合，その原因は複雑かつ多様であり，図 4 に示す通り，故障から障害に至るパスが長いことがあり，原因を解析することは難しいという課題がある．

その後，その故障モードがどれくらいの発生頻度である

のかを解析する．

その後，故障モードの検出難易度を解析し，影響度，発生度，検出難易度から危険度を評価し，対策案の検討を行う．なお，一般的には危険度の高い故障モードから対策していく．

対策案が決まれば，対策案を評価し，効果的な対策だと評価されれば，対策案を実施する．なお，ソフトウェア適用においては，対策案が効果的であるのか評価することは難しい．なぜなら，新たな故障を引き起こす可能性もあるからである．

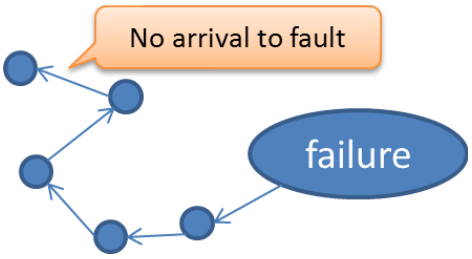

図 3　列挙した故障モードが障害に至らない例

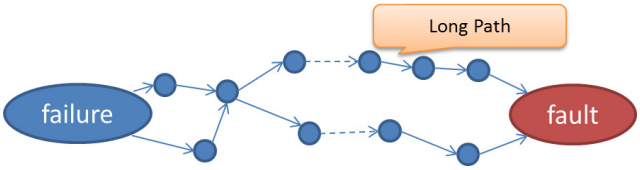

図 4 列挙した故障モードから障害へ至るパスが長い例

上記の課題をまとめると下記になる．

（課題 1）故障モードが重大な影響を及ぼす障害に至るか判断することが難しい

（課題 2）故障モードから障害に至るパスが長いことがあり，原因解析が難しい

（課題 3）対策案が効果的であるのかの評価が難しい

3. FMEA とモデル検査を組み合わせた提案手法

3.1 概要

FMEA をソフトウェアに適用する際の課題を解決するために，網羅的に検証し，反例による解析が可能なモデル検査を組み合わせた設計プロセスを提案する．

2.2 節で述べた通り，FMEA は重大な障害に至る故障を引き起こすあらゆる原因を検出しやすい手法であるが，大規模なソフトウェアに適用するのは，大規模ソフトウェアの複雑性などにより故障モードを抽出しにくいなどといった課題がある．

一方，システムをモデル化し，そのシステムが持つべき性質（＝仕様）を満たしているかを，網羅的に検査するモデル検査がある．モデル検査は，システムをモデル化し，要求が満たされることを取り得る全状態について網羅的に検査する手法である．通常，モデル化する際，システムを適切な粒度に分割しモデル化するため，そのモデルを FMEA における部品ととらえることができ，モデル検査との親和性が高いといえる．また，障害の状態をモデルに追加し，その状態に至るかを LTL（Linear Temporal Logic：線形時相論理）式や assertion 等で形式的に定義することができ，モデル検査の網羅性を利用してあらゆる反例を検出することができる．この反例の有無から，故障モードが重大な障害の状態に至るのかが判別できるので，故障モードが重大な影響を及ぼす障害に至るか判断することが難しいといった課題（課題 1）を解決できることが期待できる．

また，モデル検査による反例を故障としてとらえることで，原因の分析に利用できる．モデル検査では，故障モードから障害に至るパスを反例として定義できるため，原因を解析することが難しいという課題（課題 2）を解決できることが期待できる．

更に対策案の評価では，対策案をモデルに反映させ，再度モデル検査で検証することにより，仮に新たな故障を引き起こしたとしても，新たな故障も検出することができる．すなわち，対策案が効果的であるのか評価することが難しいという課題（課題 3）を解決できることが期待できる．

3.2 詳細プロセス

以下，提案する FMEA とモデル検査を組み合わせた設計プロセスについて説明する．

図 5 に提案する設計プロセスを示す．図 5 で示す提案プロセスは，図 2 で示した FMEA の手順にモデル検査のプロセスを加え，順番も少し変更したものである．

まずシステム構成要素を抽出しモデル化する．なお，ソフトウェアの設計が既に UML（Unified Model Language）などを用いたモデル設計がされている場合は，この手順を省くことができる．

次に，重大な影響を及ぼす障害を列挙する．例えば，あるネットワークシステムにおいて，全ての機器の電源が OFF になってしまうなどの障害を列挙していく．なお，この部分は新規となる．

次に，列挙した障害に対して，影響度の点数付けを行う．列挙した障害は影響度が大きいという観点で抽出されたものであるため，どれも同程度の影響度となるが，列挙した障害間での相対的なレベル付けという意味で影響度を点数付けする．なお，FMEA では故障モードを列挙した後に故障モードの影響解析を行うが，本提案プロセスでは，故障モードの列挙前に点数付け可能であるので，この手順で影響度の点数を付ける．

次に，前述の手順で抽出したモデル及び障害を基に，モデル検査を行う．モデル検査では，「検証モデルの設計」「検証モデルの実装」「検証式の作成」「検証」といった基本的なモデル検査の手順に従って検証していく．

「検証モデルの設計」では，下記を行う．

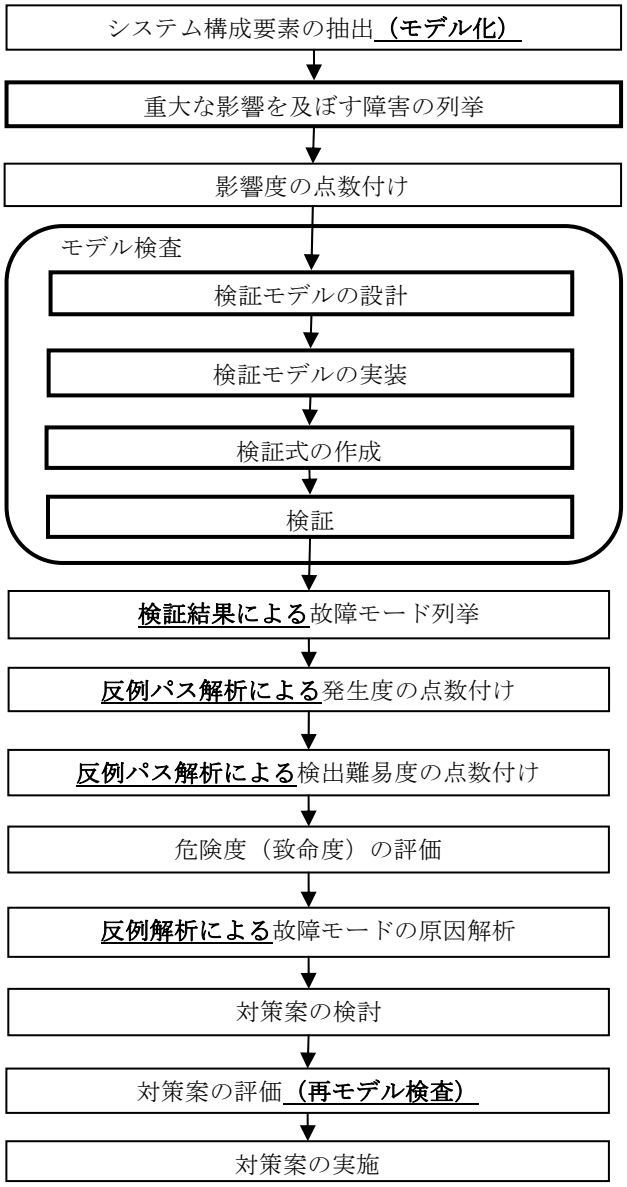

図 5　提案プロセス

・障害の状態を作成し，モデルに追加する

・「タイミング」「データ」などのキーワードを基にシステム外部の振舞いをモデルに反映する

なお，障害の状態をモデルに追加する際は，オブザーバで実現しても良いし，assertion を追加して実現しても良い．

「検証式の作成」では，上記の障害の状態に至らないことを LTL 式で記述する．

「検証」では，イベント／アクションを故意に変えることで，

・障害の状態に達成するか？

・想定していない遷移が発生するか？

を検証する．

検証の結果，反例が検出される場合，その反例を故障モードとして登録する．イベント／アクションを他のものに変えて繰り返し検証していき，反例が検出される度に故障モードとして登録する．なお，モデル検査ツールが持つ複数の反例を検出する機能などを用いて一度に検出しても良い．

次に，モデル検査で検出された反例に至るパスを解析し，そのパスが普段ありえるパスなのかを解析し，発生度の点数付けを行う．同時に，解析したパスから，通常検出しにくいパスなのかの検出難易度を点数付けする．

これらの影響度，発生度，検出難易度より危険度の計算を行う．これにより複数ある故障モードより，危険度の高いものから対策を考えていくことができる．

まず，危険度の高い故障モードの原因解析を行う．原因の解析ではモデル検査の結果である反例を用いて解析を行う．これにより，原因の解析がやりやすくなる．解析した原因の対策を検討し，対策案の評価を行う．この際，対策案をモデルに反映させ，再度モデル検査で検証することにより，仮に新たな故障を引き起こしたとしても，新たな故障も検出できるようになる．

モデル検査による再検証の結果，対策案が問題なければ，すなわち反例が検出されなければ，対策案は有効であると判断し，対策案を設計に反映する．

4. 提案手法の適用

提案手法を検証するために HEMS をモデル化し，提案手法を適用した．提供した対象 HEMS システムと結果について説明する．

4.1 対象 HEMS システム

HEMS で提供するサービスとして，コントローラからの家電製品制御が挙げられる．今回は，指定した時刻になるとコントローラからエアコンが起動される機能を例にして適用検討した．以下に今回対象とする HEMS システムの概要を説明する．

・HEMS 内機器（図 6 ）：

HEMS コントローラ，エアコン，テレビ，ブレーカ

・サービス仕様：

- 各機器（エアコン，TV）は一定周期で自身の消費電流をコントローラに通知

- ある時刻になると，コントローラは宅内の総使用電流を勘案して，エアコン起動

- ブレーカは宅内の総配線電流が宅内許容電流を超えた場合に OFF の状態になる

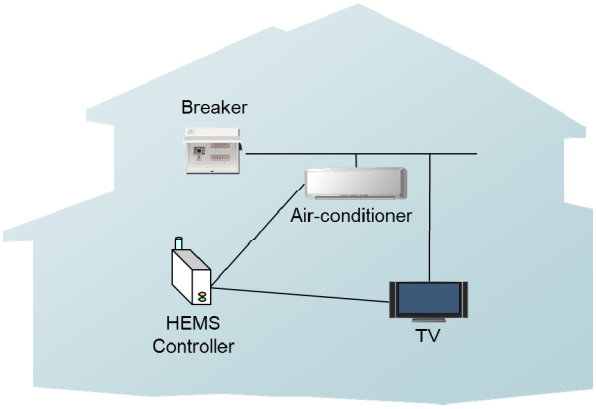

図 6　HEMS 構成概要

4.2 適用結果

3 章で示した提案プロセスを 4.1 節の HEMS システムへ適用した．以下，図 5 に示したプロセスに従って適用した結果を説明する．

「システム構成の要素の抽出（モデル化）」では HEMS システムを抽象化して状態遷移図にてモデル化を行った．各機器についてのモデルを図 7 から図 10 に示す．

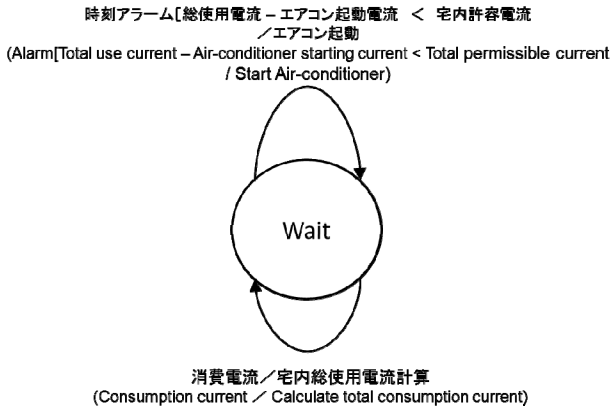

図 7　HEMS コントローラモデル

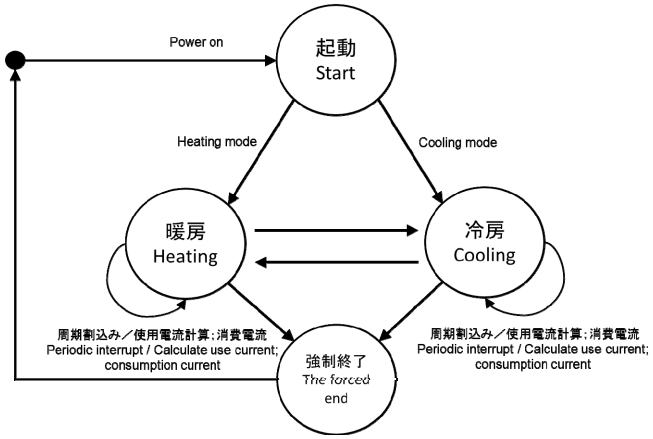

図 8　エアコンモデル

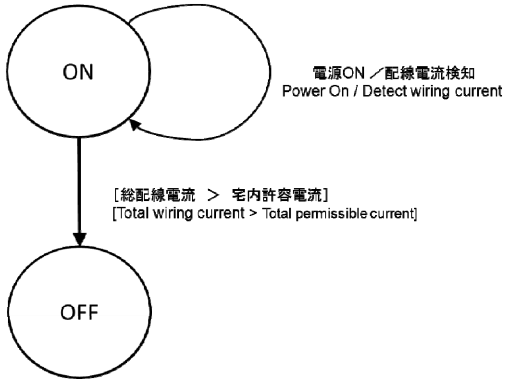

図 9　ブレーカモデル

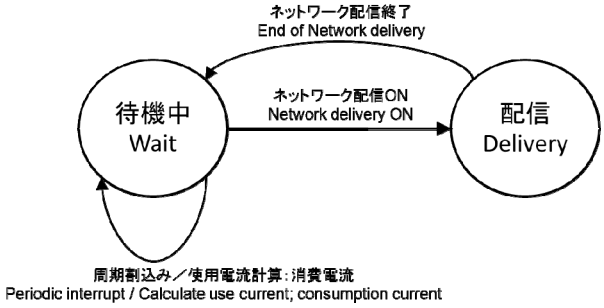

図 10　TV モデル

「重大な影響を及ぼす障害の列挙」としては，本来複数の障害を列挙すべきであるが，今回は，「ブレーカが OFF になり，全機器の電源が落ちてしまう」という障害を挙げた．「影響度の点数付け」では，単一の障害を対象としたので，今回は最高点である「5」とした．

「検証モデルの設計」においては，上述のモデル化において機器間のやりとりをモデル化したものを検証モデルとした．図 11 に検証モデルを示す．なお，障害の状態としてはブレーカが OFF になる状態をモデルに記述しており，ブレーカが OFF の状態にならないことを assertion で記述して検証した．また，状態遷移時のイベント・アクションについては，「データ」を意識し，コントローラに通知する消費電流をモデルに反映した．なお，今回の検証では，故障モードを列挙する検証は行わず，事前に下記の故障モードを想定して検証することにした．

・TV 側の消費電流通知に誤りがある
・エアコン側の消費電流通知に誤りがある
・コントローラからのエアコン起動コマンドに誤りがある

上記のうち今回は「TV 側の消費電流通知に誤りがある」場合，「ブレーカが OFF になり，全機器の電源が落ちてしまう」という重大な影響を及ぼす障害に至るかを検証することにした．

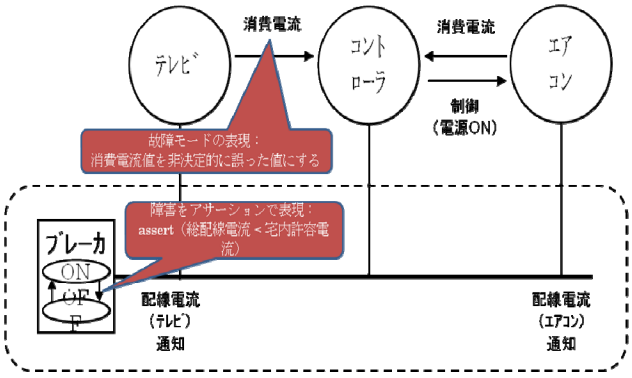

図 11　検証モデル

「検証モデルの実装」では，設計した検証モデルに従って Promela を用いて作成した．

「検証式の作成」では，前述した通り，ブレーカが OFF の状態にならないことを assertion で記述した．

「検証」では，作成した Promela で assertion が出ないかを SPIN で検査した．なお，その際，「データ」を意識して，コントローラに通知する消費電流を故意に変えるような Promela 記述にした．

検査の結果を，assertion が成立する反例が見つかった．なお，反例のステップ数は 66 ステップであった．これにより「TV 側の消費電流通知に誤りがある」という故障モードは「ブレーカが OFF になり，全機器の電源が落ちてしまう」という重大な影響を及ぼす障害に至ることが確認できた．

「検証の結果による故障モード列挙」では，反例が出力されたので，assertion に至るパスを解析し，「TV 側の消費電力通知に誤りがあるが，コントローラは誤りを検出できずに，エアコンを起動し，宅内許容電流を超えてしまい，ブレーカ OFF の状態になる」という解析結果を故障モードとして登録した．

「反例パス解析による発生度の点数付け」では，反例のパスにより，発生しやすさを検討した．今回の例では起動時のみに起こり得るので，あまり発生しないとし，5 段階中 2 とした．

「反例パス解析による検出難易度の点数付け」では，同様に反例のパスを解析することで，検出しにくさを検討した．今回の例では，5 段階評価とし，中程度の検出しにくさと設定し，検出難易度を 2 とした．

「危険度（致命度）の評価」では，発生度と検出難易度を掛け算し，10 点と点数付けした．

「反例解析による故障モードの原因解析」では，反例を解析することで，原因を解析した．今回の例では，反例を解析することで，TV からコントローラに通知する消費電流値とブレーカが参照する配線電流値が異なっており，コントローラに通知する消費電流値が配線電流値より小さい値だったことが原因だと容易にわかった．すなわち，コントローラが総消費電流値でしかエアコン起動の判断をしておらず，データの誤りを検出していないことが原因であるといえる．

上記の手順を繰り返し，他の故障モードについても同様の手順を繰り返し，危険度の高い故障モードから対策していくが，今回は他の故障モードについては検証を省略した．

「対策案の検討」では，反例解析の結果，コントローラが総消費電流値でしかエアコン起動の判断をしておらず，データの誤りを検出していないことが原因であるとわかったので，コントローラでも配線電流値を把握し，誤りがない場合にのみエアコンを起動するという対策をすることにした．

「対策案の評価」では，上記対策案を検証モデルに追加し，再度モデル検査にて検証した．モデル検査の結果，assertion エラーは出ず，すなわち「TV 側の消費電流通知に誤りがある」という故障モードが発生しても，「ブレーカが OFF になり，全機器の電源が落ちてしまう」という重大な影響を及ぼす障害にはならないことが確認でき，対策案の有効性を確認することができた．

「対策案の実施」では，対策案を設計に反映して機器を実装するが，提案プロセスの効果に対しては無関係であるので，今回の報告では割愛した．

4.3 結果の検討

4.2 節で述べた提案手法の適用結果から，2.3 節で述べた下記 FMEA のソフトウェア適用におけるそれぞれの課題に対して提案手法の有効性を検討する．

（課題 1）故障モードが重大な影響を及ぼす障害に至るか判断することが難しい

（課題 2）故障モードから障害に至るパスが長いことがあり，原因解析が難しい

（課題 3）対策案が効果的であるのかの評価が難しい

・課題 1 に対する有効性：

想定した故障モードが重大な影響を及ぼす障害の状態に至ることが，網羅的に状態検査するモデル検査により確認できた．モデル検査から得られた反例のステップ数は 66 ステップであり，人手で検討した場合，容易に検証できるステップ数ではないと判断できる．なお，モデル検査にかかる時間は数秒程度であったので，このことから提案手法の有効性が確認できる．

・課題 2 に対する有効性：

故障モードから障害に至るパスを反例として定義できるため，反例解析を行うことで原因を解析することが容易であった．例えば、今回得られた反例の各ステップにおける各変数値の変移をみることで、ブレーカが OFF になるという障害に至る原因が、コントローラでは総消費電流値でしかエアコン起動の判断をしておらず、データの誤りを検出していないことであったいうことが、容易に見つけることができた。

ただし，今回の例では反例を容易に解析できたが，パスが更に長い場合など，反例解析自身が難しい場合は原因の解析も困難になると想定できる．しかし，モデル検査の反例解析については様々な支援ツールの研究開発がなされており，それらを用いることで問題解決できると考える．

・課題 3 に対する有効性：

対策案をモデルに反映させ，再度モデル検査で検証することにより，assertion エラーが出ず，対策案の有効性を確認することができた．

以上より，適用事例に対して，本提案手法の有効性が概ね確認できた．

5. 関連研究

オープンシステムのような新産業領域はまだ実績がなく，エネルギー分野（スマートグリッド/EMS）やクラウドの分野でオープンシステムとしての問題意識が出てきた段階である[4]. 今回例で挙げた HEMS に関しても各社製品システムは販売しているが，家庭内電力の見える化がメインであり，電力管理して自動制御するようなものまでは製品化されていない．この事からも，オープンシステムにおける信頼性向上に関する研究開発はまだ発展途上であるといえる．

FMEA とモデル検査を組み合わせる手法についてはいくつか提案されている．例えば、故障発生から危害達成までのシナリオを PetriNet を用いて可視化し曖昧性を低減する手法[5]や FMEA に確率的モデル検査を加えた手法[6]がある．これらの研究では，本研究における課題 1 に対する解決策としてモデル検査を用いているが，検証モデルの設計方針などは示されておらず，またその他の課題について言及されていない．

6. おわりに

オープンシステムにおける信頼性向上を目的として，リスク分析手法である FMEA と網羅的に状態検査するモデル検査を組み合わせた高信頼設計プロセスを提案し，FMEA をソフトウェア適用する際の課題を洗い出し，全ての課題解決について，提案手法は有効であることを確認した．

今後は，FMEA 及びモデル検査単独検証との比較や，他手法との比較を行っていく．

謝辞 本研究はトップエスイーの修了制作をもとに発展させました．

参考文献

1) 所 眞理雄他，科学技術振興機構，DEOS プロジェクト White Paper Version3.0，2011
2) 安東 宣善，松本 誠一郎，今井 光洋，小村 勝人，片岸 誠，日立評論 2010. 10 月号（p88-p95），スマートハウス実証事例と基盤技術，2010
3) FMEA/FMECA 初級編 第 1 章 FMEA の全体像，リスクマネジメントタイムズ，http://www.safety-book.net/fmeca.html
4) ディペンダビリティを必要とする新産業領域，ディペンダブル組込み OS 研究開発センター，http://www.dependable-os.net/osddeos/top_contents/shinsangyou.html
5) 豊嶋 伊知郎，西川武一郎，株式会社東芝，PetriNet によるシステム非正常系可視化の検討, 電子情報通信学会 信学技報, 2008
6) Lars Grunske, Robert Colvin, Kirsten Winter, University of Queensland, Probabilistic Model-Checking Support for FMEA, IEEE, 2007