Vol.2012-SLDM-155 No.12
Vol.2012-EMB-24 No.12
2012/3/2

SysML を用いたシステム開発における制約の充足可能性検証

福田哲志[†1]　久住憲嗣[†2]　福田晃[†3]

SysML は，ハードウェア，ソフトウェア，人などを含めたシステムの包括的なモデリングに利用でき，システムの設計，解析，検証が可能になると期待されている．しかしながら，SysML の要求図やパラメトリック図などを用いて設計上の各種制約を記述することはできるが，それらの制約をどのように設計に織り込み，設計結果がそれらの制約を満たすかどうかを検証する方法については明らかにされていない．そこで，本稿では FMEA を用いて要求図で表現できる抽象的な制約を具体的にし，設計上の制約を洗い出し，さらに，SMT ソルバである Yices を用いて，それらの制約を設計が充足しているかどうかを検証する手法を提案する．

Satisfiability Verification of Constraints in System Development Using SysML

Tetsushi Fukuda,[†1] Kenji Hisazumi[†2] and Akira Fukuda[†3]

SysML enables us to model hardware, software and humans related to systems comprehensively and SysML is expected to enable system design, analysis and verification. However, SysML enables to describe constraints using requirement diagram and parametric diagram, but method to use the constrains in design and verify satisfiability of the constraints is not obvious. In this paper, we specify abstract constraints of requirement diagram using FMEA and identify constraints in design. Moreover, we propose a method to verify a satisfiability of the constraints using Yices, a SMT solver.

[†1]九州大学大学院システム情報科学府
Graduate School of Information Science and Electrical Engineering, Kyushu University
[†2]九州大学システム LSI 研究センター
System LSI Research Center, Kyushu University
[†3]九州大学大学院システム情報科学研究院
Faculty of Information Science and Electrical Engineering, Kyushu University

1. はじめに

組込みシステムが大規模化，複雑化する一方で，より短期間での開発が求められている．その上で開発者はシステムの高い品質と信頼性を保証しなければならない．また組込みシステム開発では様々な分野の開発者がコミュニケーションを取り開発を行う必要がある．さらに，組込みシステム内には物理条件など多くの制約が存在する．これらの制約を分析，検証することでシステムの信頼性を高めることができる．

SysML（Systems Modeling Language）[3] は UML（Unified Modeling Language）[4] を拡張したモデリング言語であり，ハードウェア，ソフトウェア，人などを含めたシステムを包括的に記述できる．SysML を用いることでシステムの設計，解析，検証が可能である．SysML ダイアグラムの 1 つであるパラメトリック図は，システム内の制約を可視化し，検証に用いることができる．

本研究では，まずシステムの信頼性分析の手法である FMEA（Failure Mode and Effects Analysis）を用いてシステムの制約を分析し，SysML モデルを記述する．さらに SMT ソルバである Yices を用いて，制約の充足可能性を検証する．最後に飛行船システムに対して本手法を適用し，ケーススタディを行う．

本稿の構成は以下のとおりである．

2. ESS ロボットチャレンジ

ESS ロボットチャレンジ[5] の前身である MDD ロボットチャレンジは，情報処理学会組込みシステムシンポジウムの特別企画として開催されてきた．このコンテストではモデル駆動開発に主眼が置かれていたが，2011 年度より組込みシステムに関わる人々が分野を問わず集える ESS ロボットチャレンジへと変更された．

本プロジェクトでは ESS ロボットチャレンジ 2011 で用いる飛行船自動航行システムを開発するが，プロジェクト開始時点では競技内容が未定であったため，MDD ロボットチャレンジ 2010 の仕様[6] を想定して開発を行った．

MDD ロボットチャレンジ 2010 では小型飛行船を自動航行させるシステムを開発し，その飛行動作を競う．飛行船自動航行システムの構成図を図 1 に示す．

システムの構成要素は以下の通りである．

- 飛行船に指令を出すグランドコントロール
- 飛行船搭載の上下，右，左の 3 つのプロペラによる推進

© 2012 Information Processing Society of Japan

Vol.2012-SLDM-155 No.12
Vol.2012-EMB-24 No.12
2012/3/2

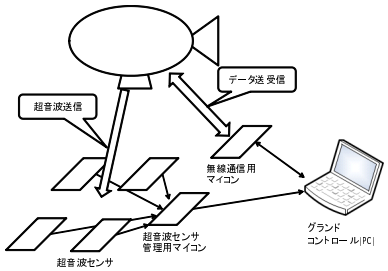

図 1 システム構成図

- 飛行船搭載ソナーによる高度測位
- 飛行船搭載角速度センサによる方位測位
- 地上超音波マトリクスによる経緯測位

グランドコントロールは飛行船と通信を行い，高度，方位の情報を受信する．さらに，地上超音波マトリクスにより xy 座標を測定する．グランドコントロールはこれらの情報をもとにプロペラの推進力を決定し，飛行船に送信する．競技ではグランドコントロールが飛行指示を行い，離陸，ホバリング，指定された地点への移動，着陸などの決められたシナリオでの飛行動作を競う．

3. SysML を用いた飛行船自動航行システムの開発

SysML を用いて飛行船自動航行システムを開発する．フェーズを 3 つのイテレーションに分割し，各イテレーションの終わりに振り返りを実施することでプロセスの改善を図る．その後，モデル駆動開発を導入する．

3.1 飛行船自動航行システムの開発

飛行船に指令を与えるグランドコントロールを SysML を用いて開発する．

はじめに要件定義を行いシステムを分析する．要件定義では，要求を整理するための要求図を作成し，機能要求についてユースケース図とユースケース記述を作成する．作成した要求図を図 2 に示す．その後，ブロック定義図と内部ブロック図を作成しシステムの構造を分析する．ブロック定義図では，ハードウェア，ソフトウェアおよび競技者をモデリングした．さらに，ユースケース記述をもとにシーケンス図を作成しシステムの相互作用を分析する．要件定義では，各ダイヤグラム間のトレーサビリティを意識し，モデリングを行った．

次に設計を行う．設計では，要件分析のドキュメントをもとに，ブロック定義図，内部ブロック図，シーケンス図，ステートマシン図を作成することでシステムの構造と振る舞いを詳細化した．

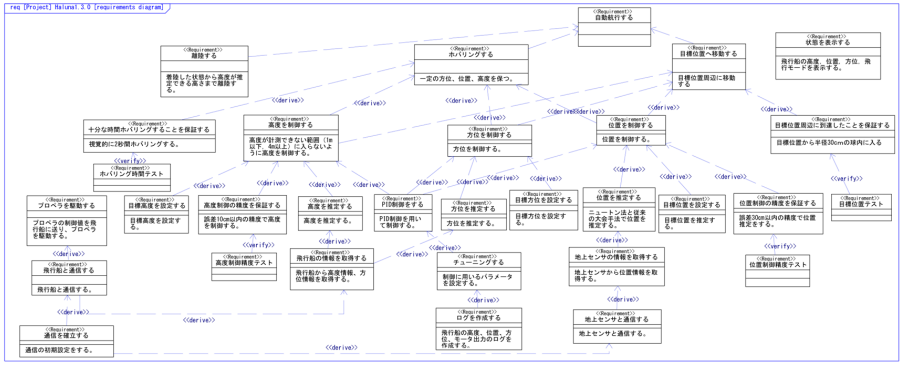

図 2 要求図

その後，実装とテストを行う．以上の流れのイテレーションを 3 回繰り返した．

例外をイベントとして定義することで，例外が発生した際の振る舞いを容易に追加，変更することが可能となる．また，モデル上でのシミュレーションでは開発者が任意のタイミングでイベントを送信することが可能であるため，これらのイベントを定義することで検証が容易になる．その結果，MDD における設計とシミュレーションによる検証のトライアンドエラーが容易になる．イベントのリストを表 1 に示す．

4. FMEA の適用

FMEA(Failure Mode and Effects Analysis) は，システムの信頼性分析のための手法である[10)11)12)]．FMEA では部分と全体の機能的関連をもとに，システム全体にどのような潜在的欠陥があるか，どの部分が致命的な損害につながるか分析する．本プロジェクトでは飛行

表 1 イベントリスト

イベント名	説明
evStartFlight	自動航行の開始
evStartHovering	ホバリングの開始
evHoveringFinished	ホバリングの終了
evTooLow	高度が下がりすぎた

© 2012 Information Processing Society of Japan

船自動航行システムのハードウェアとソフトウェアに対して FMEA を適用し，大会で起こりうる飛行船自動航行システムの故障を予測し，考慮すべき例外を洗い出す．FMEA を実施する際，２～４人でブレーンストーミングを行うことで故障の抜け漏れを防ぐ．

まず飛行船システムのハードウェアに対して FMEA を適用する．飛行船の部品ごとに故障モードとその原因，故障が及ぼす影響を分析し，FMEA ワークシートを作成した．

次に飛行船システムのソフトウェアに対して FMEA を適用する．図**??**に示した各ブロックをソフトウェアの部品として捉え，分析した．分析では，第 1 フェーズまでに得られた知見や第 1 フェーズで発生した障害を参考にした．ソフトウェアの FMEA では故障モード，原因，影響に加え，故障の検出法と対処法を決定した．さらに，故障の頻度，故障の重大度，シナリオ順の観点から，各故障モードに対して優先度を付けた．飛行船競技では実行する飛行シナリオの順序に制約があり，前のシナリオが完了しなければ次のシナリオに移れない．シナリオ順の項目では飛行競技の順序を考慮し，序盤に起こりうる故障に対して，より高い優先度を設定した．開発フェーズでは，優先度が高い故障から順に対処を実装する．

5. Yices を用いた制約の充足可能性検証

高品質なシステムを実現するためには，システム内に存在する制約を明確に定義し，最終的な成果物において制約が充足される必要がある．本節では，FMEA を用いて制約を抽出するプロセスと，SMT ソルバである Yices を用いてシステムに存在する制約の充足可能性を検証する手法について述べる．

5.1 関 連 研 究

文献 2) は，SysML モデルから抽出した制約を SMT ソルバである Yices が解析できる形式へと変換し，制約の充足可能性を検査することを目的としている．この文献で提案された手法では，SysML のブロック定義図，内部ブロック図，パラメトリック図から Yices 記述への変換方法を定義している．さらにケーススタディとして二輪型倒立振子ロボットに対して提案手法を適用し，有用性について検討している．

5.2 提 案 手 法

本節では，制約の充足可能性を検証する手順について述べる．検証は制約の定義，SysML モデルの作成，Yices を用いた検証の順で行う．

5.2.1 制約の定義

FMEA ではシステムの部品や機能に対して，故障モード，故障原因，故障検出法などを分析する．これらの分析結果はシステム内の制約を定義するための有用な情報となる．本手法では，まず FMEA 分析結果の故障モードを検証すべき制約と考える．次に，故障原因や故障検出法の項目をもとに，制約を Yices で記述できる数式の形へ変換し，FMEA シートに追加する．

5.2.2 SysML モデルの作成

要求図はシステムの機能要求，非機能要求と各要求間の関係を記述するために用いられる．FMEA シートの機能を機能要求として，機能に対する制約を機能要求から派生した非機能要求として要求図に記述する．この際，制約にはステレオタイプ ⟨⟨constraint⟩⟩ を付加することで，制約であることを明示する．

ブロック定義図はシステムの構造を記述するために用いられる．また，システム内の制約を制約ブロックとして記述できる．要求図で記述した検証すべき制約を制約ブロックへと変換し，ブロック定義図に記述する．

内部ブロック図では，ブロック定義図で定義されたシステムの構成要素であるブロックの関連を記述する．

パラメトリック図では，制約ブロックとして定義された制約のパラメータやブロック定義図で定義されたバリューを配置し，バインディングすることでパラメータ間の関係を記述する．

5.2.3 Yices を用いた検証

Yices を用いて検証を行うために，SysML モデルを Yices 記述へと変換する．変換方法は文献 2) の手順に従う．変換後の Yices 記述を実行し，検証を行う．

5.3 ケーススタディ

ケーススタディとして飛行船自動航行システムに対して提案した手法を適用する．

飛行船は取得した方位情報をもとに，現在方位と目標方位の偏差を用いて PID 制御をする．制御値は左右のプロペラの推力として，モータへ出力される．この出力値が小さすぎた場合，飛行船に十分な推力が与えられず，飛行船が動かないという現象が起こる．そこで，直進時に進まないという故障モードを定義し FMEA を実施する．さらに，この故障モードの件地方をもとに，Yices で検証できる数式へと変換する．飛行船の直進における故障を分析した FMEA シートを図 3 に示す．

次に，前節で述べた手順に従い，SysML モデルを記述する．

FMEA シートの故障モードを制約として抽出する．直進の故障モードである「進まない」に対する制約を抽出し,「プロペラ出力の下限」という制約を定義する．機能要求である「直進」に関係する制約として，図 2 で示した要求図に追加する．要求図に追加した部分を図 4

© 2012 Information Processing Society of Japan

Vol.2012-SLDM-155 No.12
Vol.2012-EMB-24 No.12
2012/3/2

機能	故障モード	推定原因	故障検知法	Yices検証式
直進	進まない	PIDパラメータの設定ミス	推力の下限を設定し，検出する	-20＜方位の偏差＜20 and 推力＞20
	蛇行する	PIDパラメータの設定ミス	左右の推力の差を制限し，検出する	-20＜(右の推力－左の推力)＜20
	速すぎる	PIDパラメータの設定ミス	推力の上限を設定し，検出する	

図 3 FMEA シート

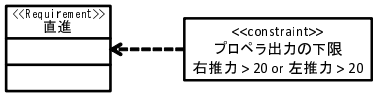

図 4 要求図（追加部分）

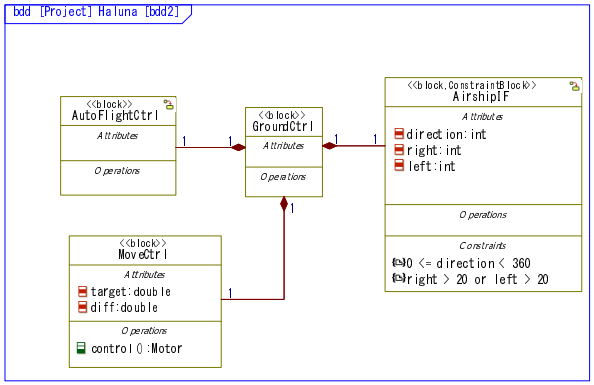

図 5 ブロック定義図

に示す．

システムの構成要素であるブロックをブロック定義図に記述する．ブロック定義図を図 5 に示す． ブロック定義図ではブロックの持つ属性を定義し，これらの属性がとるべき値の範囲も記述する．要求図で定義された制約は AirshipIF ブロックの属性である right と left の制約として記述する．

制約のブロック定義図を記述する．制約ブロックはシステム内に現れる値の関係を数式で記述する．本ケーススタディでは PID 制御の計算式などを制約ブロックとして記述する．制約のブロック定義図を図 6 に示す．

ブロック定義図で定義されたブロックのインスタンスの間の関連を内部ブロック図に記述

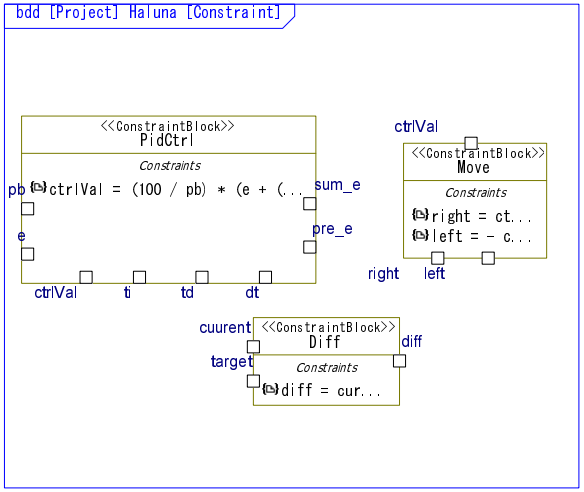

図 6 制約ブロック定義図

する．内部ブロック図を図 7 に示す．

制約ブロックとして定義された制約のパラメータや，ブロック定義図で定義されたブロックのバリューの間の関係をパラメトリック図で記述する．パラメトリック図を図 8 に示す．

文献 2) の手順に従って，以上の SysML モデルを Yices 記述に変換する．

変換後の Yices 記述を図に示す．

6. 考　　察

Yices 記述を実行した結果を図 10 に示す．

その結果，方位の現在値と目標値の偏差が小さいとき，プロペラの推力が定めた下限値以上という制約を充足しない．このような条件のときに，飛行船は推力を得ることができず，故障に陥ると考えられる．

今回のケーススタディでは，制御におけるフィードバックや連続的なデータに対する検証ができていない．今後の課題として，MATLAB/simulink などのシミュレーションのデータを用いて検証することが挙げられる．

© 2012 Information Processing Society of Japan

Vol.2012-SLDM-155 No.12
Vol.2012-EMB-24 No.12
2012/3/2

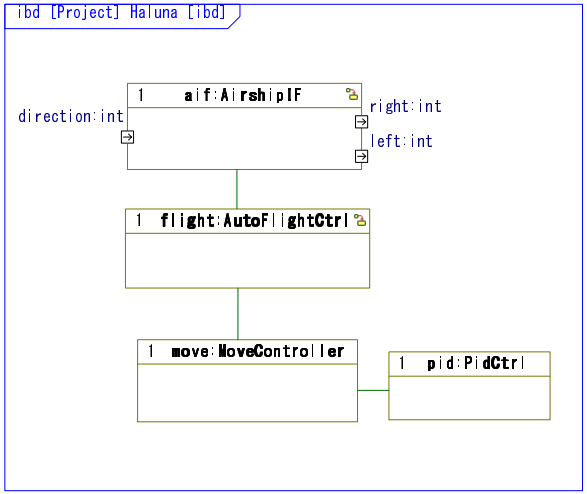

図 7 内部ブロック図

7. おわりに

本研究では，信頼性分析手法である FMEA を用いてシステムの制約を定義し，SMT ソルバである Yices を用いて制約を設計が充足しているか検証する手法を提案した．さらに，本手法を飛行船システムに適用し，ケーススタディを行った．

参 考 文 献

1) Sanford Friedenthal, Alan Moore, and Rick Steiner, *A Practical Guide to SysML*, Morgan Kaufmann Publishers (2008).
2) 加藤 秀明，上田 賀一，中島 震：SysML モデルの制約妥当性検証に関する考察，組込みシステムシンポジウム 2010 論文集，pp.5–12 (2010).
3) Object Management Group: OMG Systems Modeling Language, http://www.omgsysml.org/.
4) Object Management Group: OMG Unified Modeling Language, http://www.uml.org/.
5) ESS ロボットチャレンジ 2011，http://www.sigemb.jp/rc2011/.
6) MDD ロボットチャレンジ 2010，http://sdlab.sys.wakayama-u.ac.jp/mdd2010/.
7) 福田哲志，末安史親，庭木勝也，森田健治，久住憲嗣，峯恒憲，鵜林尚靖，平山雅之，濱田直樹，二上貴夫：飛行船自動航行システム開発における SysML を用いたプロセス改善事例，電子情報通信学会技術研究報告，110(458)，pp.151–156 （2011）．
8) 福田哲志，末安史親，庭木勝也，森田健治，久住憲嗣，峯恒憲，鵜林尚靖，平山雅之，濱田直樹，二上貴夫：飛行船自動航行システム開発における SysML を用いたモデル駆動開発事例，組込みシステムシンポジウム 2011 論文集，pp.14-1–14-8 （2011）．
9) IBM: Java Tutorial for Rational Rhapsody, http://publib.boulder.ibm.com/infocenter/rsdp/v1r0m0/topic/com.ibm.help.download.rhapsody.doc/pdf75/tutorialj.pdf.
10) 鈴木順二郎，牧野鉄治，石坂茂樹：FMEA・FTA 実施法，日科技連 （1982）．
11) 塩見弘，島岡淳，石山敬幸：FMEA，FTA の活用，日科技連 （1983）．
12) 小野寺勝重：グローバルスタンダード時代における実践 FMEA 手法，日科技連（1998）．
13) 梅村晃広：SAT ソルバ・SMT ソルバの技術と応用，コンピュータソフトウェア，Vol.27, No.3, pp.24–35 (2010).

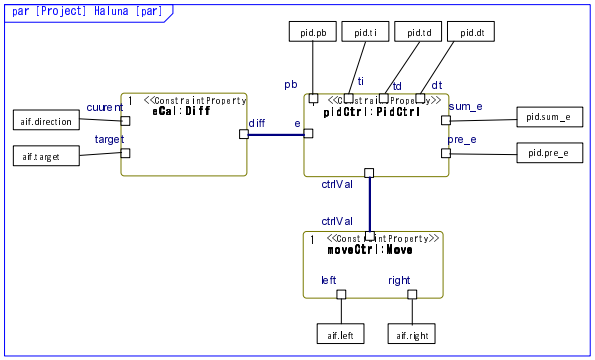

図 8 パラメトリック図

© 2012 Information Processing Society of Japan

Vol.2012-SLDM-155 No.12
Vol.2012-EMB-24 No.12
2012/3/2

```
(define aif.direction::real)
(define aif.right::real)
(define aif.left::real)
(define pid.pb::real 167)
(define pid.ti::real 1000000000)
(define pid.td::real 0)
(define pid.dt::real 100)
(define pid.e::real)
(define pid.ctrlVal::real)
…
(define move.target::real 0)
(define move.diff::real)
…
(assert (and (⟨= 0 aif.direction) (⟨ aif.direction 360)))
(assert (= moveCtrl.right (- 3 moveCtrl.ctrlVal)))
(assert (= moveCtrl.left (+ 3 moveCtrl.ctrlVal)))
…
(assert (not (or (⟩ moveCtrl.right 10)
 (⟩ moveCtrl.left 10))))
```

図 9 Yices 記述（一部抜粋）

```
(= eCal.diff 4509/400)
(= pidCtrl.ctrlVal 27/4)
(= pidCtrl.e 4509/400)
(= moveCtrl.right -15/4)
(= moveCtrl.ctrlVal 27/4)
(= moveCtrl.left 39/4)
(= aif.right -15/4)
(= aif.left 39/4)
```

図 10 Yices 実行結果（一部抜粋）

© 2012 Information Processing Society of Japan