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

福田哲志[†1] 末安史親[†1] 庭木勝也[†1]
森田健治[†1] 久住憲嗣[†1] 峯恒憲[†1]
鵜林尚靖[†1] 平山雅之[†2]
濱田直樹[†3] 二上貴夫[†4]

システムモデリング言語である 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，システム設計，解析，検証できるようになることが期待されている．しかしながら，SysML をプロジェクトに導入することは容易ではなく，現状では導入事例の報告も少ない．そこで本稿では飛行船自動航行システム開発に SysML を適用し，さらに自動化されたモデル駆動開発を適用した事例を報告する．また，プロジェクト全体を振り返り，発生した問題点や課題，得られた知見について議論する．

A Model-Driven Development Case of Airship Auto Navigation System Using SysML

Tetsushi Fukuda,[†1] Fumichika Sueyasu,[†1] Katsuya Niwaki,[†1]
Kenji Morita,[†1] Kenji Hisazumi,[†1] Tsunenori Mine,[†1]
Naoyasu Ubayashi,[†1] Masayuki Hirayama,[†2] Naoki Hamada[†3]
and Takao Futagami [†4]

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, it is difficult to introduce SysML into projects and there are not many introduction cases. We applied SysML and model-driven development to an airship auto navigation system development. In this paper, we report the case, and discuss problems and knowledge which we found through this project.

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．

組込みシステムの 1 つである飛行船システムは，制御，位置・高度推定，通信などの技術を複雑に組み合わせて開発する．また，飛行船には様々な例外が発生する可能性があり，例外の洗い出しとその対処が重要になる．さらに，例外への対処を人間の運用に任せることも考えられるため，ハードウェアやソフトウェアに加えてシステムに関わる人を含めたモデリングが必要である．

SysML（Systems Modeling Language）[1] は UML（Unified Modeling Language）[2] 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，検証ができ，異なる分野の開発者同士のコミュニケーションが促進されることが期待される．SysML は組込みシステム開発において有用であると考えられるが，SysML をプロジェクトに導入することは容易ではなく，現状では導入事例の報告も少ない．我々は飛行船自動航行シス

†1 九州大学
Kyushu University
†2 日本大学
Nihon University
†3 情報処理推進機構
Information-technology Promotion Agency, Japan
†4 東洋テクニカ
TOYO Corporation

テム開発に SysML を適用することで，複雑な飛行船システムをモデリングする．さらに，SysML をプロジェクトに導入する際の有用な知見を得る．

モデル駆動開発（MDD : Model Driven Development）では設計したモデルからコードを自動生成する．そのため，モデルを変更することで容易に実装に反映できる．さらに，設計モデルを用いてシミュレーションを行い，早期に検証を行うことが可能となる．飛行船システム開発に MDD を適用することで，例外への対処を含めた振る舞いの変更を容易にする．さらに，シミュレーションをシステムテストのサイクルを短い周期でまわすことで開発を促進する．

FMEA（Failure Mode and Effects Analysis）は FMEA はシステムの信頼性分析の手法である．本手法を適用することで飛行船システムに起こりうる例外を洗い出し，例外への対処を分析する．

本稿の構成は以下のとおりである．第 2 節では，我々が参加する ESS ロボットチャレンジの概要を述べる．第 3 節では，飛行船自動航行システム開発プロジェクトの概要を述べる．第 4 節では，技術調査として行った簡易飛行船システム開発と MDD ロボットチャレンジ 2010 視察について述べる．第 5 節では，SysML を用いた飛行船自動航行システムの開発事例を報告する．第 6 節では，モデル駆動開発の導入について述べる．第 7 節では，飛行船自動航行システムに対して FMEA を適用する．第 8 節では，プロジェクト全体を振り返り，発生した問題点や課題，得られた知見について述べる．最後に 9 節で本稿をまとめる．

2. ESS ロボットチャレンジ

ESS ロボットチャレンジ[3] 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[4] 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

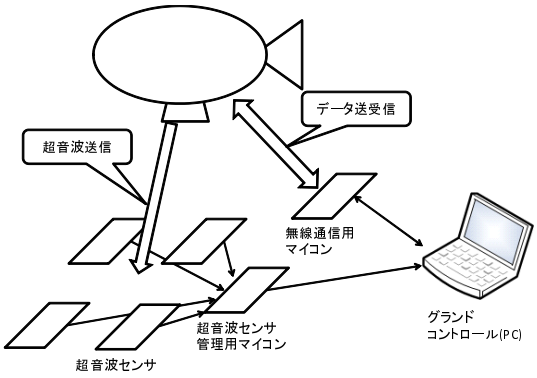

図 1 システム構成図

- 飛行船搭載の上下，右，左の 3 つのプロペラによる推進
- 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. プロジェクト概要

本プロジェクトでは 2011 年に開催される ESS ロボットチャレンジで用いる飛行船自動開発システムを開発する．

本プロジェクトは九州大学大学院システム情報科学府社会情報工学コースで行われる PBL（Project-Based Learning）の一環として行う．チームは情報工学を専攻する修士の学生 4 名で構成される．4 名の学生は C 言語や Java でのプログラミング経験や UML を用いたモデリング知識を有している．しかしながら，プロジェクト開始時点では SysML や MDD に関する知識はない．

本プロジェクトの期間は 2010 年 10 月から 2011 年 7 月の約 10ヶ月である．まず，第 0 フェーズとして技術調査を行う．技術調査では，簡易飛行船システムを開発することで飛行船の基本的操作を習得する．さらに，MDD ロボットチャレンジ 2011 を視察することで開発に有用な知見を獲得する．次に，第 1 フェーズとして，SysML を用いて飛行船自動航行システム

図 2 簡易飛行船システム

を開発する．フェーズ全体を 3 つのイテレーションに分割し，徐々に機能を追加することで開発を行う．その後，第 2 フェーズとして飛行船自動航行システム開発に MDD を適用する．シミュレーションを用いて，設計，検証のトライアンドエラーを短い周期で繰り返し，開発を促進する．

4. 技 術 調 査

本プロジェクトでは，はじめに第 0 フェーズとして技術調査を行う．飛行船の基本的操作，飛行船に起こりうる障害などの知見を予め獲得し，要件に反映することで大幅な手戻りのリスクを軽減する．

4.1 簡易飛行船システム開発

まずはじめに技術調査として，開発対象である飛行船のドメイン知識を獲得するために簡易飛行船システムを開発した．簡易飛行船システムを図 2 に示す． 簡易飛行船システムは実際の飛行船の駆動部のみを釣り竿で釣るし，その釣り竿を支持する棒を中央に配して，円形に飛行できるようにしたシステムである．飛行船の風船部分は空気抵抗が大きく制御が難しいため，初期段階ではこのような簡易飛行船システムを利用してドメイン知識の獲得を行う．

簡易飛行船システム開発はソースコード中心の開発であり，ノウハウの獲得に注力した．我々は簡易飛行船システムの開発を通して，飛行船および地上センサマトリクスとの通信方法，飛行船の位置・高度の推定方法，PID 制御を用いた飛行船の高度の制御方法を習得した．

4.2 MDD ロボットチャレンジ 2010 視察

次に，技術調査として 2010 年の大会を視察した．実際に競技を見ることで，簡易飛行システム開発では発見できなかった検討すべき事柄を発見した．その後，発見した事柄とその対処法を整理し，14 点の知見としてまとめた．作成した知見集を図 3 に示す．この知見集は今後の開発に活用する．

5. SysML を用いた飛行船自動航行システムの開発

次に，第 1 フェーズとして 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．このフェーズでは，SysML の調査とシステム開発を並行して行う．

5.1 SysML 調査

本プロジェクトでは，文献 5) を参考に SysML の調査を行った．調査はメンバー 1 人が文献を読み，他のメンバーに対して発表するという輪講形式で行った．文献を用いて，SysML の各ダイアグラムの記述法および SysML を用いた開発のケーススタディに関して調査した．

5.2 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．グランドコントロールの開発では 3 回のイテレーションに分割し，徐々に機能を追加することで開発する．各イテレーションではまず計画を立て，その計画に基づいて開発を行う．各イテレーションの最後に振り返りを実施し，次イテレーションでの開発プロセスの改善を図る[6]．

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 4 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．

分類	気づき	対策
HW	飛行船の浮力と姿勢の調整	飛行船の高度が徐々に下がるくらいに、姿勢は水平に保つように調整する。 微妙な調整のためテープ、クリップ等を準備する。
	飛行船にひもを付けると、コースアウト時の復帰は早いが、高度変化に伴う重量変化で制御が不安定になりうる	ひもの重量の変化をモデルにする手間を考え、現段階では飛行船にひもを付けないことにする。
戦略	飛行順路	残り時間により飛行モードを遷移できるようにし、なるべく得点が高くなる順路をとる。
本番	時間の有効利用	動作確認手順、整備手順のリストを作成する。本番で障害が発生した場合のチェック表を作成し、慌てず対策出来るようにする。
	本番ならではの状況を想像し、対処できるようにする必要がある	温度等も考慮したモデルを作成する必要があると思われるが、現段階では気づきに留め後々考慮することにする。
実装	ホバリングに関し、プログラムで2秒停止と記述していても審査員にはそうは見えない	プログラム上でホバリングの時間を長く設定し、第三者が見てもホバリングを認識できるようにする。 ホバリングの判定条件を別に設ける。
	全プロペラを止めて着陸する方法を考える必要がある	着陸を目指す際、高度が70cm以下になったら、全モータを停止させ、着陸を図る。
	エラー処理をきっちりする必要がある	センサデータ取得部分にログーを埋め込み、エラーが起きた際に原因箇所を特定できるようにする。
	飛行船の電源を入れた時の方位が0となり、中断時などでリセットがしにくい	GC側で方位の初期値を0にできるようにする。
	通信障害への対策が重要	電波を飛ばすものを切ってもらう。 エラー処理を真面目にやる。
	UIが小さいのはアピール度が小さい	審査員・観客にわかりやすいように大きく表示させる。
	PID制御パラメータ・飛行順路等の設定を簡単に行えたほうがいい	パラメータの変更をしやすいようなUIを作成する。
	位置推定に使用するセンサの数は多いほうがよいはず	反応した超音波センサを可能な限り使用する。 逆に、3つ以下のときの振る舞いも設定する。
	高度が計測できない場合がある （1m以下・4m以上）	ログを参照し、正常な値がとれる高度まで上昇・下降することで復帰できるようにする。

図 3 知見集

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，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．

その後，実装とテストを行い，イテレーションを振り返る．振り返りではプロセスフロー図をもとに分析し，次イテレーションでのプロセスの改善を図った．作成したプロセスフロー図を図 5 に示す．以上の流れのイテレーションを 3 回繰り返した．

第 1 フェーズの最後に，飛行船自動航行システムの運用モデルを作成する．大会では限られた時間内でできるだけ多くの飛行動作を実行する必要がある．そこで，システムの運用をアクティビティ図を用いて明確化した．作成した運用モデルを図 6 に示す．この運用モデルは後に，第 7 節で実施する FMEA の結果を反映し，開発終了時に実装できなかった機能がある場合は変更を加える予定である．

6. モデル駆動開発の適用

次に，プロジェクトに MDD を導入する．

MDD の特徴として以下の点が挙げられる．

- 設計したモデルからソースコードを自動生成できる．
- モデルを用いたシミュレーションにより，早期に検証できる．
- モデルとソースコードの一貫性を保つことができる．

本プロジェクトでは，MDD を導入することで，開発の早い段階でシミュレーションによる検証を行い，手戻りが発生するリスクの軽減を図る．また，MDD を用いてモデルからソースコードを自動生成することで，

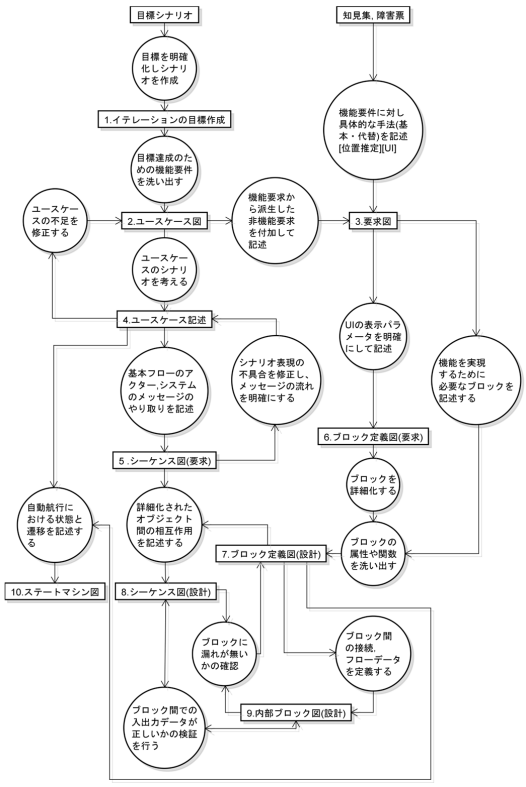

図 5 プロセスフロー

大会当日の柔軟なシナリオの変更を可能にする．

6.1 モデル駆動開発の導入

本プロジェクトでは，MDD ツールとして IBM Rational Rhapsody（以下，Rhapsody）を用いる．MDD 環境の導入は，1 人のメンバーが調査を行い，その後他のメンバーに伝えるという形式で行う．Rhapsody の使用法の習熟には，付属の入門ガイドおよび Java

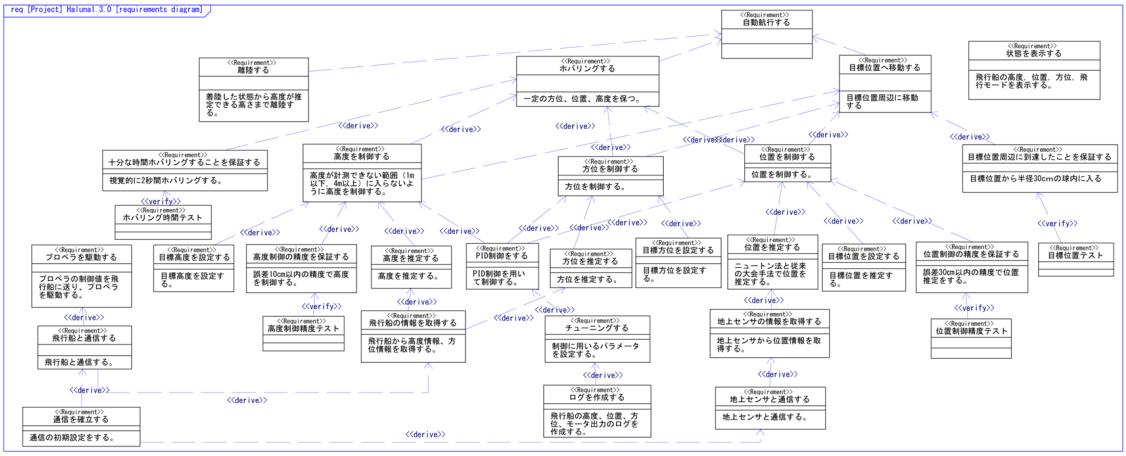

図 4 要求図

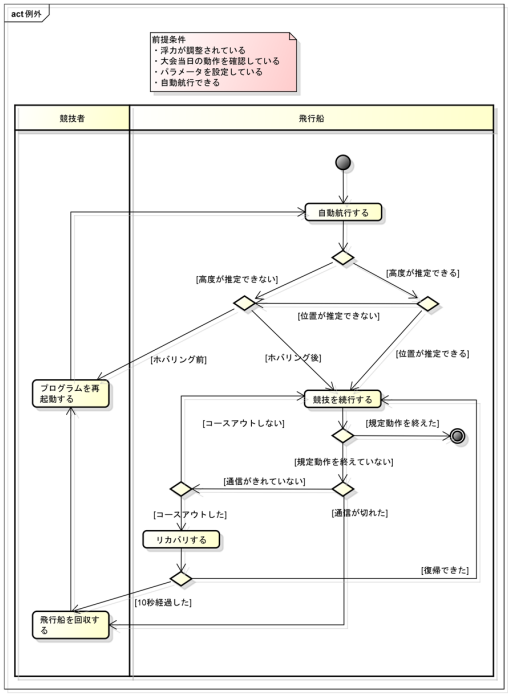

図 6 運用モデル

Tutorial[7] を参考にした．まず Rhapsody の環境を構築し，入門ガイドのチュートリアルを参考にコードの自動生成およびモデルを用いたシミュレーションの方法を習得した．開発では，SysML のブロック定義図，ステートマシン図を設計し，これらのモデルから実行可能なソースコードを自動生成する．その後，作成したモデル上でシミュレーションを行い，動作を検証する．離陸，ホバリングなどのシナリオごとに正常フローを開発し，シミュレーションによる検証とシステムテストを繰り返す．さらに，第 7 節で述べる FMEA の結果をもとに例外を考慮し，例外時の振る舞いを追加していく．

6.2 グランドコントロールアーキテクチャの修正

MDD を導入するに当たり，まずグランドコントロールのアーキテクチャを見直す．MDD ではブロックごとにステートマシン図を作成し，ソフトウェアの振る舞いを実装する．ステートマシンの状態はブロックが特定のイベントを受信することで遷移する．

以前のアーキテクチャでは，シナリオに関する各ブロックが飛行船の状態を判断し，状態を遷移させていた．新しいアーキテクチャでは，現在の飛行船の状態を判断して適切なイベントを送信するためのブロックを追加した．さらに，飛行シナリオの開始と終了，例外の発生などをイベントとして定義した．例外については第 7 節で述べる FMEA の結果を反映する．グランドコントロールのアーキテクチャを記述した SysML ブロック定義図を図 7 に示す．また，各ブロックの責務を表 1 に示す． 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 2 に示す．

7. 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[8)9)10)]．

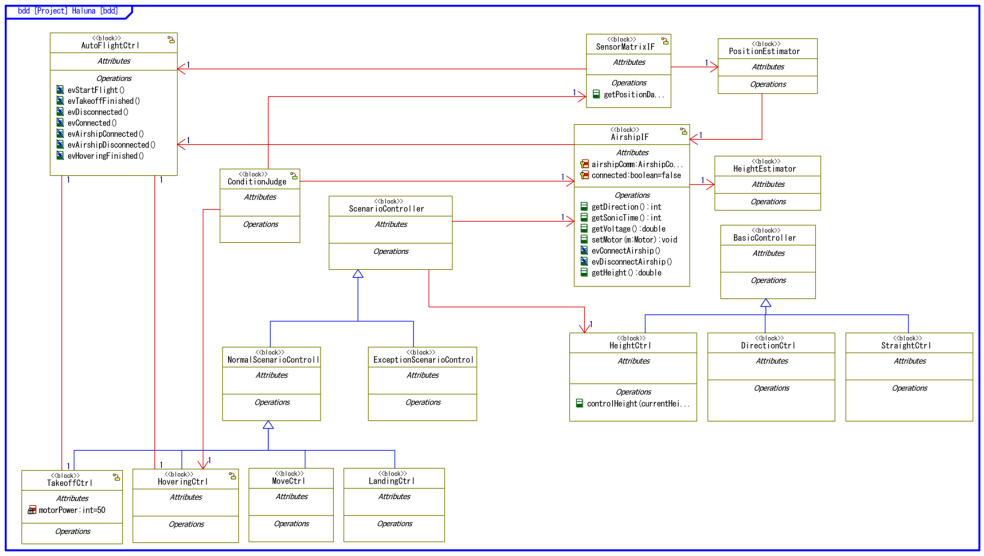

図 7　グランドコントロールアーキテクチャ

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．作成したハードウェアの FMEA ワークシートの抜粋を示す．

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 7 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，図 3 に示した知見や第 1 フェーズで発生した障害を参考にした．作成したソフトウェアの FMEA ワークシートを図 9 に示す．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．

表 1　ブロックの責務

ブロック名	責務
AutoFlightCtrl	飛行船の状態を取得し，飛行船のとるべきシナリオの管理を行う
ConditionJudge	現在の状態を判断し，適切なイベントを送信する
TakeoffCtrl	離陸時の振る舞いを管理する
HoveringCtrl	ホバリング時の振る舞いを管理する
HeightCtrl	高度調整のためのモータ制御量を計算する
DirectionCtrl	方位調整のためのモータ制御量を計算する
StraightCtrl	直進するためのモータ制御量を計算する
HeightEstimator	高度を計算する
PositionEstimator	位置を計算する
AirshipIF	飛行船から送られてきたデータを解析し，方位，高度，電圧を取得する 飛行船にモータの制御量を送信する
SensorMatrixIF	地上センサマトリクス MPU から送られてきたデータを解析し，位置情報を取得する

表 2　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

8. 考　　察

本節では，SysML，MDD，FMEA の適用それぞれ対して振り返りを行う．

8.1 SysML 適用の振り返り

第 1 フェーズでは，SysML でのモデリングに非常

番号	対象品目	機器名	機能	故障モード	推定原因	影響 サブシステム	システム	任務
	飛行船	エンベロープ		空気圧不足	ヘリウムガス不足			
					穴が開く	機能低下	損失有りうる	損失有りうる
				空気圧超過	ヘリウムガス過充填	同上	同上	同上
				空気漏れ	穴が開く	同上	同上	同上
				紛失	置き忘れ			
					盗難	機能せず	機能せず	達成できず
					どこかに飛んでいく			
				マジックテープ部分が外れる	ゴンドラ部分を急に引っ張る	同上	同上	同上
					経年劣化			
		超音波送受信機		超音波送信できない	電源が入っていない	機能低下	損失有りうる	損失有りうる
					ハードウェア故障			
				超音波受信できない	電源が入っていない			
					送信されていない	機能せず	同上	同上
					ハードウェア故障			
				計測誤差が異常に出る	跳ね返り距離が短い	機能低下	同上	同上
					天井からの跳ね返りを受信している			
				電気信号が発生しない	電源が入っていない	機能せず	同上	同上
					ハードウェア故障			
		ジャイロセンサ		方位が計測できない	電源が入っていない	同上	同上	同上
					ハードウェア故障			
				方位の誤差が異常に出る	グランドコントロールの通信不具合	機能低下	同上	同上
					ハードウェア故障			
		モータ		正転しない	電源が入っていない	機能せず	同上	同上
					ハードウェア故障			
				逆転しない	電源が入っていない	同上	同上	同上
					ハードウェア故障			
				出力が小さすぎる	電圧不足	機能低下	同上	同上
					データ受信が途切れた			
					制御量計算が間違えている			
				出力が大きすぎる	データ受信が途切れた	同上	同上	同上
					制御量計算が間違えている			
				プロペラが回らない	出力が小さい	機能せず	同上	同上
					何かに引っかかっている			
					プロペラ故障			

図 8　ハードウェアの FMEA ワークシート（抜粋）

図 9　ソフトウェアの FMEA ワークシート（抜粋）

に時間がかかるという問題があった．その理由として，我々の SysML に関する知識不足が挙げられる．プロジェクト開始時点では，SysML の知識は全くなかった．SysML の調査を開発と並行して行ったが，文献の調査に合計 148 人時もの工数がかかったことが原因である．その上，第 1 イテレーションでは要求図，ユースケース記述，ブロック定義図，シーケンス図などの大量のドキュメントを整合性を保ちつつ書いた．そのためドキュメントの作成，レビュー，修正に多くの工数が必要となってしまった．そこで，我々はイテレーションごとにプロセスをプロセスフロー図として可視化し，ドキュメント間のトレーサビリティの明確化やドキュメントの削減を試みた．その結果，徐々にプロセスが改善され，ドキュメントの作成効率が向上した．要件定義工程と設計工程における作成したドキュメントの量，ドキュメント作成工数およびドキュメント作成効率の推移を表 3，4 に示す．

また，本プロジェクトでは SysML のパラメトリック図を上手く活用することができなかった．今後の課題として，パラメトリック図を用いた制約のモデリングを飛行船システムに適用することが挙げられる．

表 3　ドキュメント作成効率の推移（要件定義工程）

	Itr1	Itr2	Itr3
作成ドキュメント量 [枚]	15	10	11
作成工数 [人時]	62	37	36
作成効率 [枚/人時]	0.24	0.27	0.31

Itr はイテレーションの略である．

表 4　ドキュメント作成効率の推移（設計工程）

	Itr1	Itr2	Itr3
作成ドキュメント量 [枚]	(5)	6	11
作成工数 [人時]	(14)	23	25.5
作成効率 [枚/人時]	(0.36)	0.26	0.43

Itr はイテレーションの略である．
設計工程の Itr1 では UML を用いた．

8.2 モデル駆動開発適用の振り返り

MDD の導入では開発環境の習熟に 23 人時，アーキテクチャの修正に 11 人時の工数を要し，プロジェクトに遅延が生じた．その結果，現状ではチーム全体に MDD を広めることができていない．また，十分に例外時の振る舞いを実装できていない．しかしながら，MDD を適用することでモデル上でのシミュレーションによるテストが容易になった．今後はチーム内に MDD の技術を広め，まだ実装できていない例外時の振る舞いを追加する．開発終了時に実装できなかった機能に関しては，運用モデルに反映させる．

8.3 FMEA 適用の振り返り

FMEA は本来，ハードウェアを対象とした信頼性向上手法であるが，本プロジェクトではソフトウェアであるグランドコントロールに対して FMEA を適用し，グランドコントロールのブロックを部品とみなし FMEA を実施した．ソフトウェアでは，あるブロックの故障が他のブロックに波及し新たな故障が発生する．そのため，類似する故障が複数のブロックに出現し，うまく整理できないという問題が起こった．そこで，ブロックの責務を明確化し，ブロックが対処すべき故障のみを洗い出した．その結果，類似する故障の数が減り，簡潔に整理できた．FMEA を実施するに当たりシステムを実際に動かさなければ発見できない故障も数多くあったが，本プロジェクトでは第 0，第 1 フェーズの知見を参考にすることで比較的容易に故障モードを洗い出せた．

9. ま と め

今回のプロジェクトでは飛行船自動航行システムを開発した．

まず我々は開発に SysML を適用した．SysML に関する調査と並行して，グランドコントロールを SysML を用いて開発した．開発を 3 回のイテレーションに分割し，振り返りを行うことでプロセスの改善を図った．SysML を用いたモデリングでは，ドキュメント間のトレーサビリティを意識した結果多くの工数がかかってしまったが，振り返りでのドキュメント削減の検討やプロセスの見直しにより，プロセスを改善した．

さらに，開発に MDD を適用した．MDD を適用するにあたり，開発環境の調査を行った．そして，グランドコントロールの振る舞いを変更し易くするためにアーキテクチャとイベントを再定義した．ここで，飛行船に起こりうる例外を分析するために FMEA を実施した．MDD を適用することでテストが容易になったが，現状では十分にチーム内に浸透していないという課題が残った．

謝辞

本研究は、九州大学大学院システム情報科学研究院情報知能工学専攻社会情報システムコース（通称，QITO）における教育カリキュラムの一環として行われている PBL（Project-Based Learning）の成果である．

参 考 文 献

1) http://www.omgsysml.org/
2) http://www.uml.org/
3) http://www.sigemb.jp/rc2011/index.html
4) http://sdlab.sys.wakayama-u.ac.jp/mdd2010/
5) Alan Moore，Rick Steiner:A Practical Guide to SysML，Morgan Kaufmann，2008.
6) 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）
7) IBM: Java Tutorial for Rational Rhapsody
8) 鈴木順二郎，牧野鉄治，石坂茂樹: FMEA・FTA 実施法，日科技連（1982）
9) 塩見弘，島岡淳，石山敬幸: FMEA，FTA の活用，日科技連（1983）
10) 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）