http://www.exmotion.co.jp/

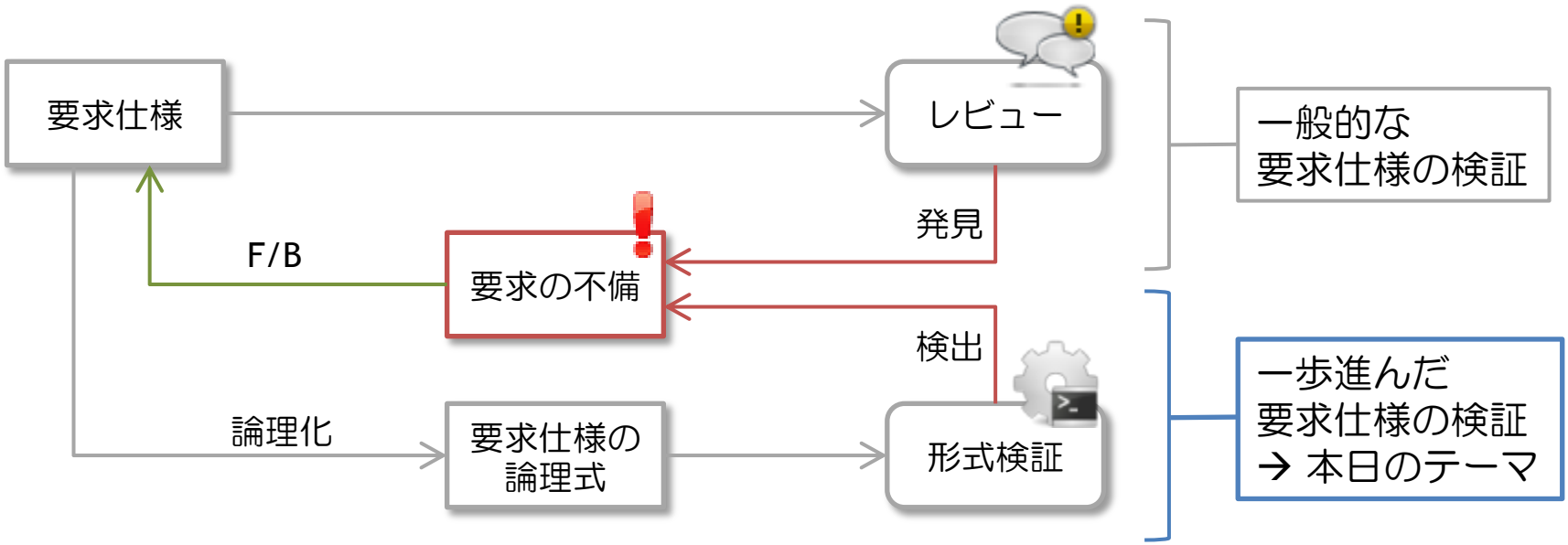
ユースケースとUSDMに
セミフォーマル手法を適用した要求検証

2011.06.17

株式会社エクスモーション
藤倉俊幸

概要

- USDM は要求の仕様化/文書化に役立つ技術である
- 要求の階層的な表現により、レビューで仕様のモレや衝突を発見しやすくなっているが、人為的な検証ではどうしてもムラが生じるものである
- 実は、USDM に“一手間かける”ことで形式検証が可能となる
- ここでは、USDM を使用した形式検証のアプローチと事例を紹介する
- 形式手法は計算機科学の成果を利用して高品質のソフト開発をする技術である



内容

- USDM と形式手法の親和性
- 要求項目の形式化
- 形式検証のアウトライン
- 検証事例：ワイパー制御システム
- まとめ

USDMと形式手法の親和性

USDMのおさらい

- Word文書等の散文的な要求仕様のまとめ方では要求と仕様の区別が付き難い
- そこで階層構造を導入して明確に区別したのがUSDM
  - 要求は本来「～できるか」と言う問い
  - 仕様は「～すればできる」と言う答え

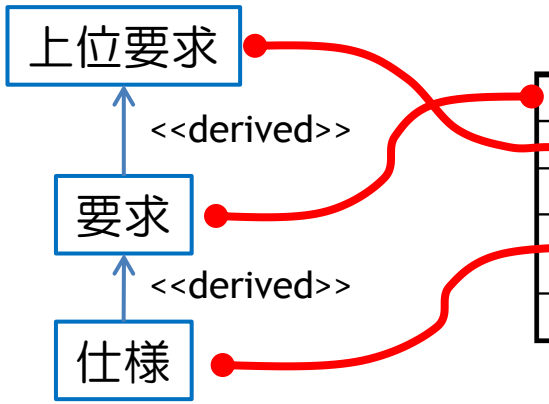
要求	PRC05	撮影範囲とスピードアップで1台の防犯カメラでカバーする範囲を広げる	
	理由	モニターも含めて設置コストを下げること	
	□ □ □	PRC.05.1	防犯カメラの首振り角度を30度から45度に変更する
	□ □ □	PRC.05.2	DCモーターの電流値を○○から□□に変更して、首振り動作の速度を50%スピードアップする
	□ □ □	PRC.05.3	画像のブレを押えるためにスキャン速度を○○から□□に変更する

USDMの深層構造

- 階層化することで実は既に論理構造が導入されている
  - 「仕様」が満たされれば「要求」満たされる
  - 「要求」が満たされれば「上位要求(理由)」満たされる
- この関係は「ならば(⇒)」を使って論理式で表現できる
  - SysML要求図では<<derive>>関係で表現される

深層構造(SysML)

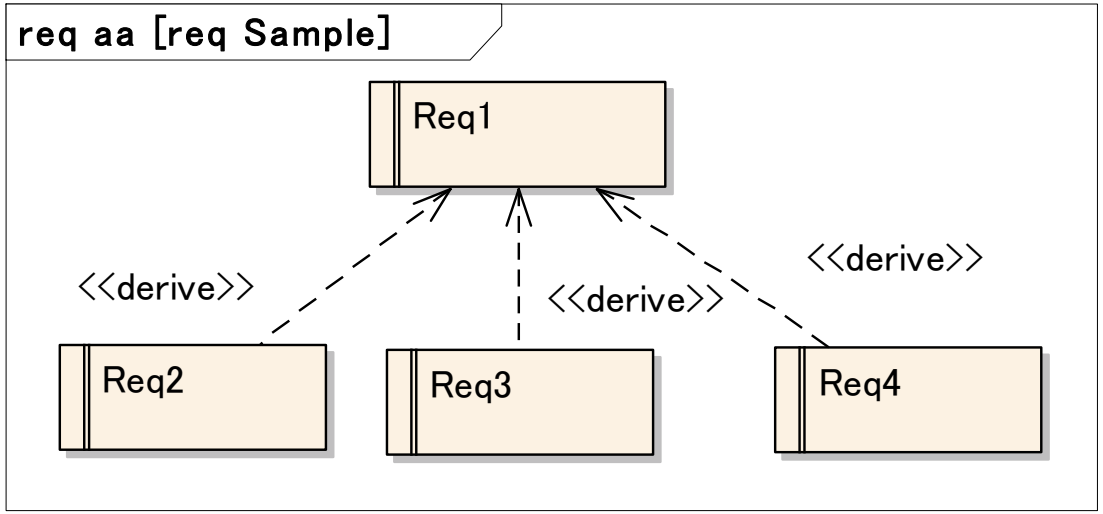
表層構造(USDM)



要求	PRC05	撮影範囲とスピードアップで1台の防犯カメラでカバーする範囲を広げる	
	理由	モニターも含めて設置コストを下げること	
	□ □ □	PRC.05.1	防犯カメラの首振り角度を30度から45度に変更する
	□ □ □	PRC.05.2	DCモーターの電流値を○○から□□に変更して、首振り動作の速度を50%スピードアップする
	□ □ □	PRC.05.3	画像のブレを押えるためにスキャン速度を○○から□□に変更する

深層構造を利用すれば形式化は容易

- たとえば以下の要求図は簡単に論理式化できる
  (Req2∧Req3∧Req4)⇒Req1
- 論理式化することで色々な解析ができる様になる
  - 上位要求を満足するための条件の明確化
  - 障害解析、FTAとの対応付け等



USDMと形式手法の親和性 まとめ

- USDM自身に論理構造を含んでいる
- その構造を利用して検証することができる

要求項目の形式化

要求項目の形式化

- 要求項目間の関係はUSDMで形式化できることを説明した
- 次に各要求項目自身の形式化について説明する

- 要求項目から命題を抽出するステップ
  1. 要求文を命題に分割する
     - これは文章を単文に分解する作業で、USDMでは既に分解されている

> ドライバがドアのロックを要求したことを受け付け(p1)、
> ドアがロックできる条件が成立(p2)していることを判定した場合は、
> ドアをロックする(p3)こと。

  2. さらに命題の組合せで要求文の意味を表現する
     - 振る舞いの順序にかかわる内容では時相論理式という形式を使用する

時相論理式

> [](p01∧p02⇒ <> p03)　：p01とp02が成立したらp03を実行する

要求項目の形式化例

時相論理式

要求記述	素命題	形式化要求記述
雨滴を払拭するために、ワイパーを作動させる	p14 雨滴を払拭する p1 ワイパーが動作する	G(p1 -> F p14) G(!p14 U p1)

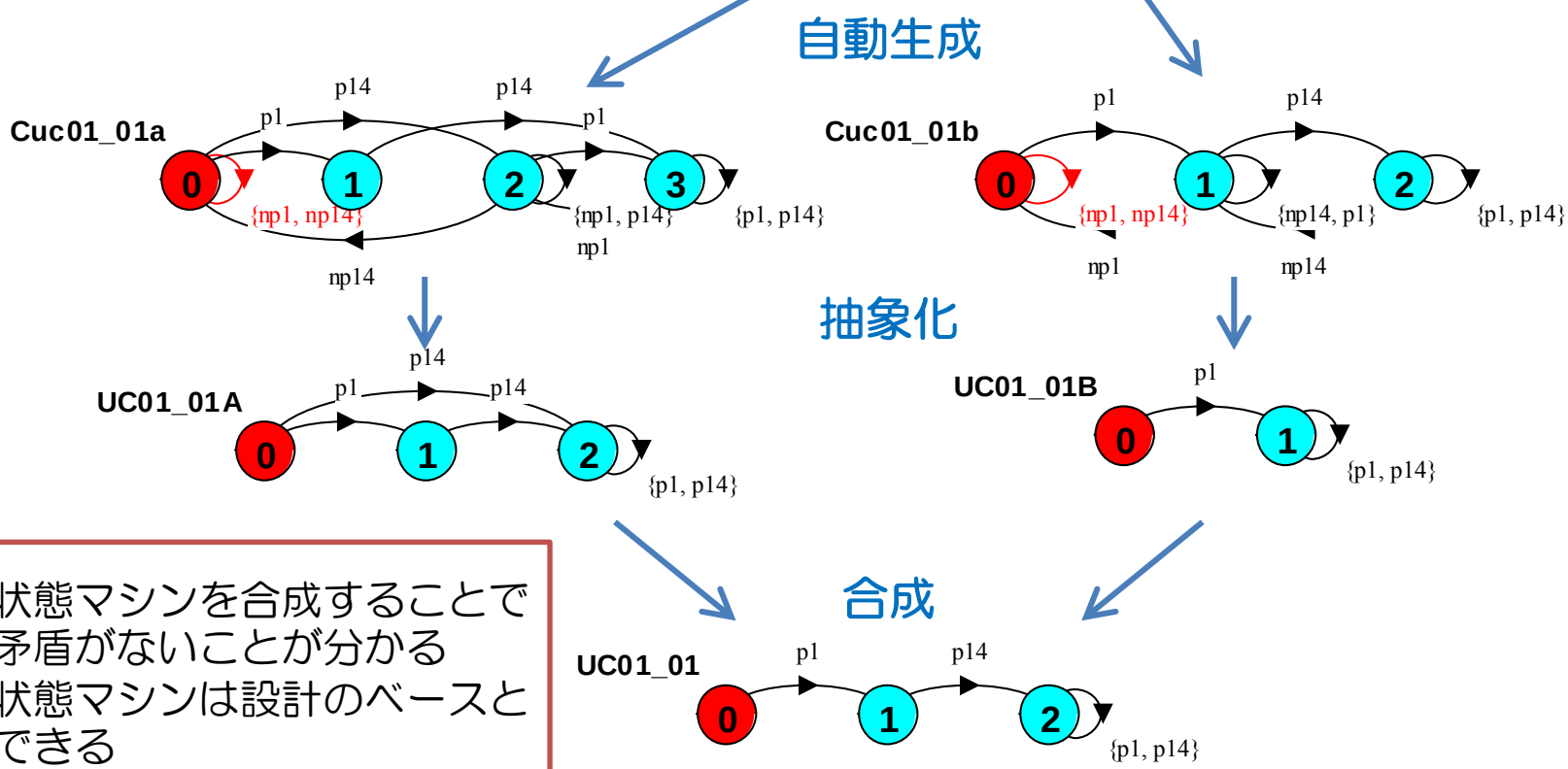
「ワイパーを動作すれば、何時か雨滴を払拭できる」 → G(p1⇒F p14)

「ワイパーを動作させるまで、雨滴を払拭できない」 → G(!p14 U p1)

- 時相論理式は使用するツールによって記号が変わるが同じ意味
  - [] / G　：常に
  - ◇ / F　：何時か
  - U　：まで

状態マシンの自動生成

要求記述	素命題	形式化要求記述
雨滴を払拭するために、ワイパーを作動させる	p14 雨滴を払拭する p1 ワイパーが動作する	G(p1 -> F p14) G(!p14 U p1)



1. 各要求の状態マシンを合成することで要求間に矛盾がないことが分かる
2. 合成した状態マシンは設計のベースとして使用できる

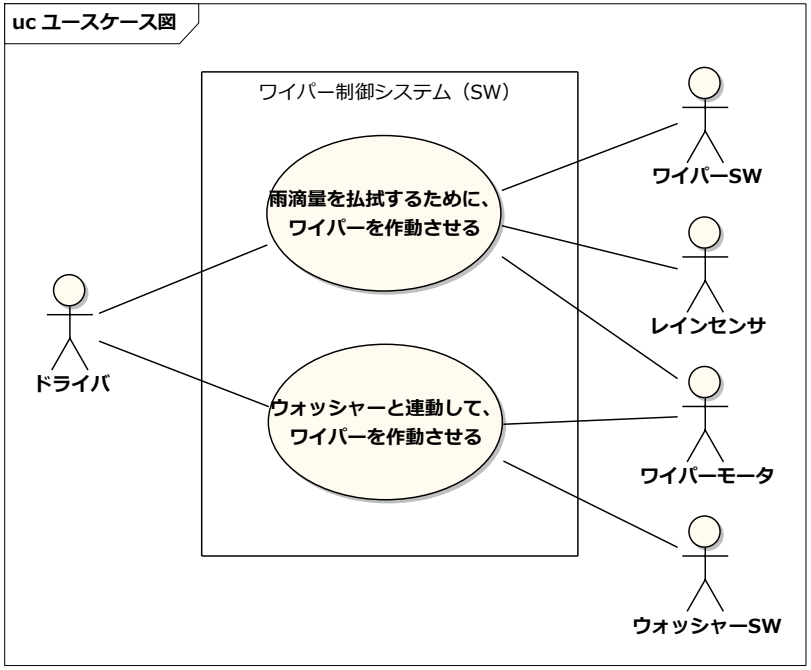
要求項目の形式化 まとめ

- 要求文から命題を抽出して論理式化する
- 振る舞い仕様の形式化には時相論理式を使用する
  - □, G(常に) ◇, F(いつか)を使う
- 時相論理式は状態マシンに変換することができる
  - 意味を状態図で確認することができる
  - 要求仕様から設計仕様を演繹で求めることができる

形式検証のアウトライン

ユースケースの併用

- ユースケースとは、システムがユーザに対してどのような価値(機能)を提供するのかを明確にした記述
- ユースケースには、USDMに含まれない以下の内容を持っている
  - 基本フローや例外フロー、代替フローによって振る舞いを表す
  - 各UCの事前条件・事後条件によってUC間の関係定義できる
- 全体を把握するためにUSDMの前段階としてユースケースを利用



ユースケースとUSDMの対応関係

- ユースケースは機能要求をステップに分解して記述する
  - ユースケース全体を上位要求に展開
  - 各ステップは下位要求あるいはグループに展開
  - 各ステップの実現方法を仕様に展開

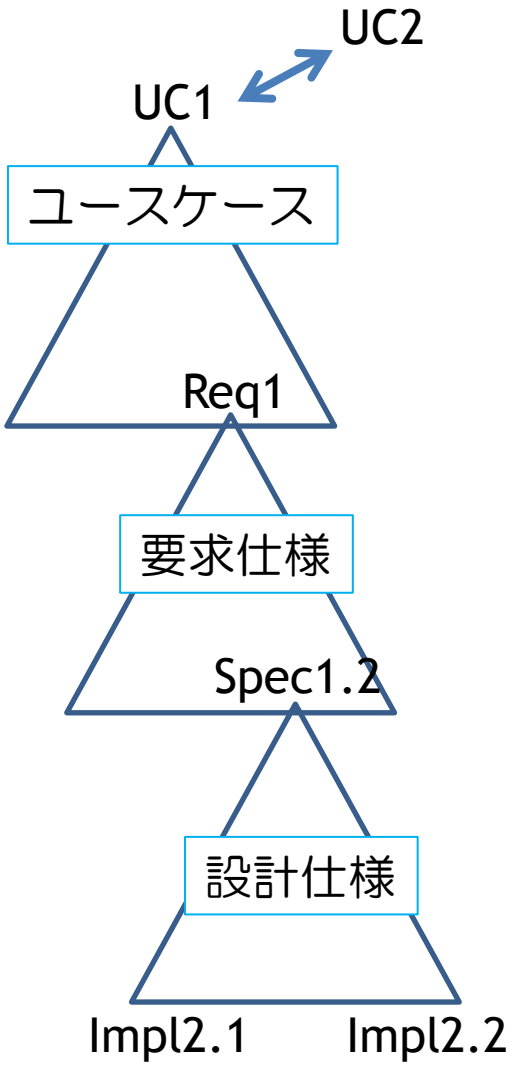
ユースケース記述

ユースケース番号		UC-WPR-02
ユースケース名		ウォッシャーと連動して、ワイパーを作動させる
概要		ドライバがウォッシャーを使ってフロントウィンドウを洗浄している場合、 洗浄液が吐出している間は、システムはワイパーを作動させる。 油膜、泥、虫を洗浄除去する
主アクタ		ドライバ
副アクタ		・ワイパーアクチュエータ ・洗浄液吐出ポンプモータ ・ウォッシャーSW（洗浄液吐出スイッチ、PushSW2） ・洗浄液吐出ポンプモータの動作を模擬するためのLED(LED1)
事前条件		ウォッシャーSW を操作
事後条件		ワイパーの作動が停止している
フロー	**STEP**	**ステップ**
基本フロー(B)	1	ドライバは、ウォッシャーSW を操作して、フロントウィンドウに洗浄液を吐出する。
	2	システムは、洗浄液が吐出している間、ワイパーを作動させる。 <A1><E1><*2>
	3	ドライバは、ウォッシャーSWの操作を止め、洗浄液の吐出を止める。
	4	システムは、洗浄液がきれいに除去されるまでワイパーの作動を継続する。
	5	システムは、ワイパーの作動を停止させる。 停止位置まで動作させ停止、または停止位置まで動作させUC01に復帰 （ユースケースの終了） <A2>

USDM

要求	WPR.02	ドライバーが洗浄液を出している間は、ワイパーを連動して作動する。
	理由	ワイパーのみでは油膜、泥、虫などの汚れを除去することは困難なため、洗浄液吐出とワイパー動作を連動させる。 また、その際のワイパー操作を簡素化したい。
	説明	・ウォッシャー連動は、ワイパーの動作モードよりも優先して行う。 ・フロントガラスを対象とし、リアガラスは対象外とする。
	<ウォッシャー連動判定>	
要求	WPR.02.01	ウォッシャースイッチの状態（ON/OFF）から、ワイパーを連動させるか判断する。
	理由	
	説明	
	< ウォッシャースイッチ押下判定 >	
□□□	WPR.02.01.01	以下の条件が成立した場合、ウォッシャースイッチ状態が ON であると判定する。 ・ ウォッシャースイッチ接点の CLOSE が規定サンプリング回数以上連続して ・ ウォッシャースイッチ接点が OPEN になった
	WPR.02.01.02	以下の条件が成立した場合、ウォッシャースイッチ状態が OFF であると判定する。 ・ ウォッシャースイッチ状態が ONでない
	< ウォッシャー連動判定 >	
□□□	WPR.02.01.11	以下の表に示すとおり、 ウォッシャースイッチが ON の場合に、連動してワイパーを作動させる。 ウォッシャースイッチ状態 / ワイパーウォッシャー連動設定 OFF / INDIVIDUAL（個別） ON / SEQUENTIAL（連動）
<ウォッシャー連動作動>		
要求	WPR.02.02	ウォッシャーと連動させる場合は、一定の速度でワイパーを作動させ、連動作動していることをインパネに表示する。
	理由	
	説明	連動の様子は、ウォッシャー動作指示LEDに表示する。
	<ワイパー間欠時間判定>	
□□□	WPR.02.02.01	ワイパーウォッシャー連動設定が SEQUENTIAL の間は、 「ワイパー間欠時間レベル = 0」とする。
	<ワイパー速度判定>	
□□□	WPR.02.02.11	ワイパーウォッシャー連動設定が SEQUENTIAL の間は、 「ワイパー速度レベル = 1」とする。
□□□	WPR.02.02.12	ワイパーウォッシャー連動設定が SEQUENTIAL から INDIVIDUAL に切り替わった場合、 ワイパーが停止位置に移動するまでは、「ワイパー速度レベル = 1」を継続する。
	< ウォッシャー動作指示表示 >	
□□□	WPR.02.02.21	以下の表に示すとおり、 ウォッシャースイッチが ON の場合に、ウォッシャー動作指示LED を点灯させる。 ウォッシャースイッチ状態 / 点灯指示 OFF / LIGHT_OUT ON / LIGHTING
要求	WPR.02.03	洗浄液がきれいに除去されるまで、ワイパーの作動を継続する
	理由	フロントウィンドウに洗浄液が残っている可能性があるため

検証スコープ



- 抽象度の高い上位要求から順に検証を進める
- システム全体を把握できるレベルで形式化をおこない、そのレベルでの矛盾を排除しベースラインとする
- 以降の詳細化はコンポーネント化して検証をおこなう
  - 全体ベースラインが無い場合効果も局所的
- 検証ツールも使い分ける

形式検証ツール

- 多様なスコープに応じて、UC検証ではLTSA、USDM検証ではNuSMVを使用した。
  - 抽象度やツールの特性によって使い分けた
  - LTSAモデルはNuSMVモデルに変換することができる

要求仕様記述法	検証したい内容
ユースケース	システム全体の振る舞い、上位のユーザ要求間の矛盾
USDM	詳細化した仕様間の矛盾

検証ツール	検証できる内容
LTSA	抽象度の高いアクション間の矛盾、変数を使用しないレベルの仕様検証
NuSMV	変数を使用するレベルの仕様検証、操作的仕様の検証

検証事例: ワイパー制御システム

テーマ「ワイパー制御システム」

＜ 機能 ＞

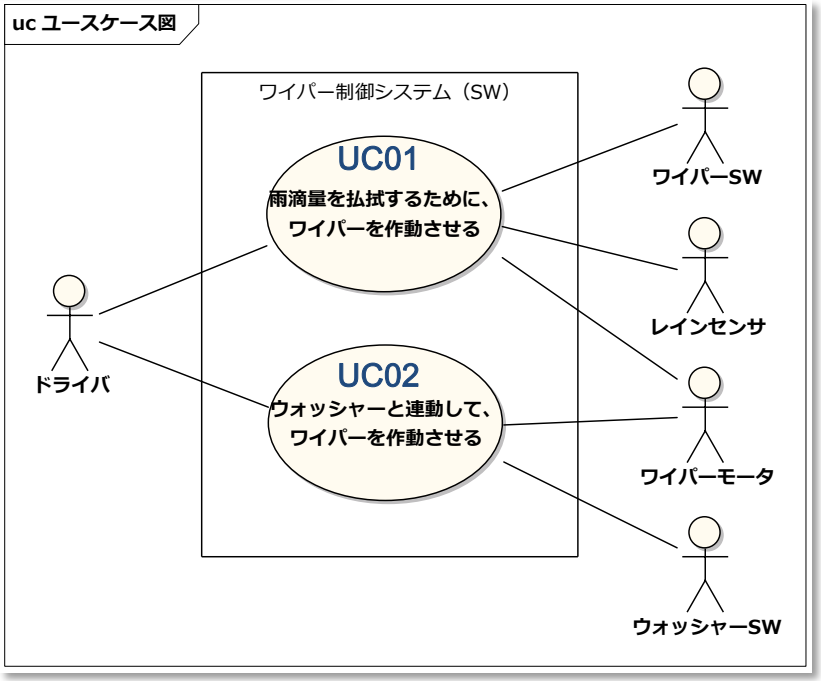
◆ 動作モードに応じたワイパー作動
(自動/低速/高速/間欠)

◆ ウォッシャー連動のワイパー作動

ユースケース

- ユーザに提供する機能とシステム全体での動作を記述した

ユースケースモデル



ユースケース記述

ユースケース番号		UC-WPR-01
ユースケース名		雨滴を払拭するために、ワイパーを作動させる
概要		ドライバがワイパーSW を操作することにより、システムは動作モードに応じた間欠時間と速度でワイパーを作動させる。
主アクタ		ドライバ
副アクタ		・ワイパー ・ワイパーSW ・レインセンサ（評価システムでは 雨滴量設定SW）
事前条件		ACC あるいは IG が ON であること
事後条件		ワイパーの作動が停止している
フロー	STEP	ステップ
基本フロー(B)	1	ドライバは、ワイパーSW を操作して、ワイパーの作動を要求する。 <A5>
	2	ドライバは、ワイパーSWを操作して、ワイパーの動作モードを指定する。 ※ワイパー動作モードには、次の4種類がある。 － AUTO / MANU-INT / MANU-LOW / MANU-HIGH
	3	システムは、指定されたワイパー動作モードをインパネに表示し、ドライバに通知する。
	4	システムは、指定された動作モードに応じて、 動作させるワイパーの間欠時間と作動速度を判定する。 <A1><A2><A3><A4>
	5	システムは、規定の間欠時間と速度でワイパーを作動させる。 <E1><*1>
	6	ドライバは、ワイパーSW を操作して、作動停止を要求する。 <A6>
	7	システムは、ワイパー作動を停止させる。 （ユースケースの終了）
代替フロー		
A1： 雨滴感知式モード(AUTO)が 指定された場合	1	ドライバは、ワイパー動作モードを AUTO にする。
	2	システムは、レインセンサが感知した雨滴量を受け取る。
	3	システムは、雨滴量に応じて、動作させるワイパーの間欠時間と作動速度を判定する。
	4	B-5 に戻る。
A2： 間欠時間調整モード(MANU-INT) が指定された場合	1	ドライバは、ワイパー動作モードを MANU-INT にする。
	2	ドライバは、間欠時間SW を操作して、間欠時間を調整する。
	3	システムは、間欠時間SW の調整値から間欠時間を判定する。
	4	システムは、MANU-INT モードの作動速度を判定する。
	5	B-5 に戻る。
A3： 低速作動(MANU-LOW)が 指定された場合	1	ドライバは、ワイパー動作モードを MANU-LOW にする。
	2	システムは、MANU-LOW モードの間欠時間と作動速度を判定する。
	3	B-5 に戻る。

USDM

- 導出したユースケースの各ステップを詳細化し、さらに品質要求や設計制約を織り込み、設計・実装できる具体的なレベルまで仕様化した

USDM

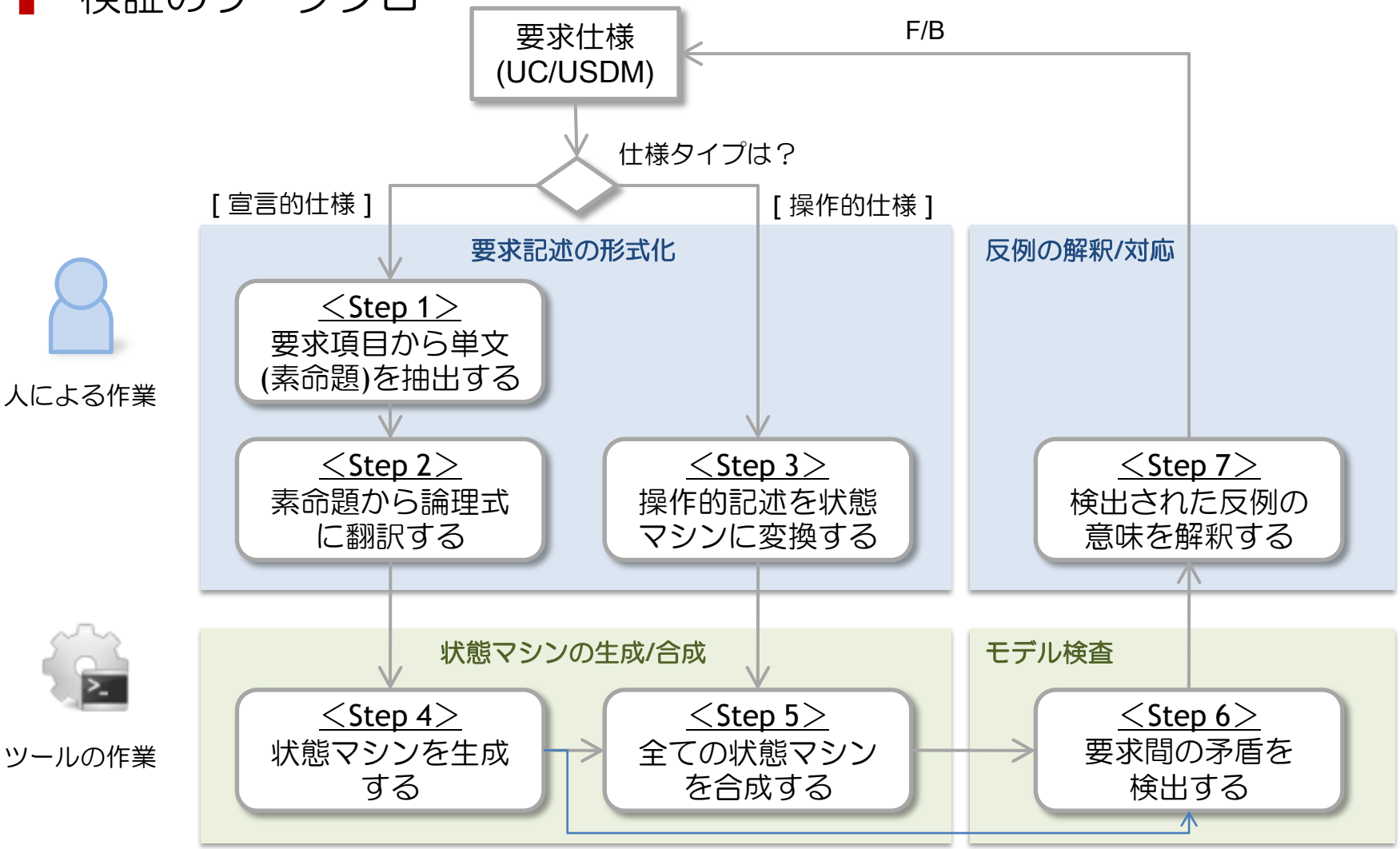
ワイパー制御 要求仕様書

機能要求				
	要求	WPR.01		ドライバーがワイパースイッチを操作することで、所定の速度・間欠でワイパーを作動させる。
		理由		雨、油膜、泥、虫などの運転の視界を邪魔する要因を取り除いて、前方の視認性を確保したい。
		説明		・視界不良の対象はフロントガラスのみとし、それ以外のサイドガラス、リアガラスは対象外とする。 ・視界不良となる要因は雨、油膜、泥、虫を想定し、それ以外の雪や凍結などは想定していない。
	動作モード切替え	＜ワイパー作動要求の判定＞		
		要求	WPR.01.01	ワイパースイッチが押されるたびに「OFF→オート作動→間欠作動→低速作動→高速作動」と動作モードを切替え、 現在の動作モードをインパネに表示する。
		理由		視認不良の程度に応じて、適度な間隔/速度でワイパーを作動させたい
		説明		動作モードは 7セグLED1 に表示する
		＜ワイパースイッチ押下判定＞		
		□□□	WPR.01.01.01	以下の条件が成立した場合、ワイパースイッチ状態が ON であると判定する。 ・ワイパースイッチ接点の CLOSE が 2サンプリング以上連続して ・ワイパースイッチ接点が OPEN になった
		□□□	WPR.01.01.02	以下の条件が成立した場合、ワイパースイッチ状態が OFF であると判定する。 ・ワイパースイッチ状態が ON ではない
		＜動作モード切替え＞		
		□□□	WPR.01.01.11	作動モードの初期状態は OFF とする。
		□□□	WPR.01.01.12	ワイパースイッチが押されるたびに「OFF→オート作動→間欠作動→低速作動→高速作動」と動作モードを切り替える。 また、高速作動に達したら、以降は逆順で動作モードを切り替える。 スイッチの押下回数と動作モードの対応、および押下による動作モードの遷移を以下の表に示す。
		＜動作モードの表示＞		

ワイパースイッチ状態ONの判定回数	ワイパーモード	スイッチ押下によるモード遷移	
0	OFF	↓	(←)
1	AUTO	↓	↑
2	MANU-INTERVAL	↓	↑
3	MANU-LOW	↓	↑
4	MANU-HIGH	(→)	↑

検証方法

- 検証のワークフロー

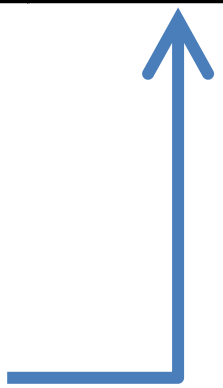


論理式化作業

UCステップレベル仕様	論理式
ドライバは、ウォッシャーSW を操作して、フロントウィンドウに洗浄液を吐出する。	□(p3⇒◇p0)
システムは、洗浄液が吐出している間、ワイパーを作動させる。	□(p2⇒p1)
ドライバは、ウォッシャーSWの操作を止め、洗浄液の吐出を止める。	□(¬p3⇒◇¬p2)
システムは、洗浄液がきれいに除去されるまでワイパーの作動を継続する。	□(p1 U p4)
システムは、ワイパーの作動を停止させる。 停止位置まで動作させ停止、または停止位置まで動作させUC01に復帰	□(p4 ⇒ ◇ ¬ p1)

命題リスト

p0	ウォッシャー動作中
p1	ワイパー動作中
p2	洗浄液吐出
p3	ウォッシャーSWを操作(off以外)
p4	洗浄液がきれいに除去される
p5	ワイパーSW を操作する、OFF以外にする
p6	ACC あるいは IG が ON である
p7	雨滴量を受け取る
p8	動作させるワイパーの間欠時間と作動速度を判定する
p9	間欠箇時間確定
p10	ワイパー動作速度確定
p11	間欠時間SW を操作
p12	モードの作動速度を判定する
p13	動作モード確定
p14	雨滴なし
p15	7セグLED1 に動作モードを表示
p16	雨滴量変化
p17	ワイパーが故障



＜Step 1＞
要求項目から単文
(素命題)を抽出する

＜Step 2＞
素命題から論理式
に翻訳する

状態マシン化作業

- 操作的な仕様記述は直接ハンドライティングする

＜Step 3＞
操作的記述を状態マシンに変換する

USDM操作的仕様記述

雨滴量レベル	ワイパー間欠時間レベル
0	0
1	12
2	10
3	8
4	6
5	4
6	2
7	1
8	0
9	0

Autoモードにおける間欠時間仕様

NuSMV動作モデル

```
next(wpintlevel) := case
  wpmode = auto & rain in {0,8,9} : 0;
  wpmode = auto & rain = 1 : 12;
  wpmode = auto & rain = 2 : 10;
  wpmode = auto & rain = 3 : 8;
  wpmode = auto & rain = 4 : 6;
  wpmode = auto & rain = 5 : 4;
  wpmode = auto & rain = 6 : 2;
  wpmode = auto & rain = 7 : 1;
  wpmode = mint : wpintSWlevel;
  wpmode = mlow : 0;
  wpmode = mhigh : 0;
  wpmode = off : 0;
esac;
```

検証結果

- 今回の検証では抽象度の異なるUCとUSDMを組み合わせることで多様な検証スコープを持つことができた
  - そのため、ユースケースによる外部仕様の定義/検証、USDMによる内部仕様の定義/検証と段階的に要求定義を進めることができる
- 今回紹介する検証例

	検証例	検証スコープ
1	仕様の未定義部分を検出 さらに、場当たり的な仕様変更によって作りこんだ他の問題も検出	UCステップ UC間
2	仕様を詳細化する過程での条件モレと仕様モレの検出	USDM仕様
3	複数仕様間での不整合の検出	USDM仕様間 上位要求
4	上位要求と詳細化した設計レベルの仕様との不整合の検出 （システム全体の設計検証は状態数が増えるため難しいが多様なスコープを持つことで検証可能になる）	USDM仕様 設計仕様 上位要求
5	不適切な仕様化によって上位要求が満たされないことを検出	USDM仕様 ハード仕様

検証例-1 前半

仕様の未定義部分を検出
さらに、場当たり的な仕様変更によって作りこんだ他の問題も検出

- ウォッシャー連動(UC2)の終了条件が未定義であることを検出
  → 未定義条件があると無限ループになるため検出可能

ユースケース記述

STEP	ステップ
1	ドライバは、ウォッシャーSW を操作して、フロントウィンドウに洗浄液を吐出する。
2	システムは、洗浄液が吐出している間、ワイパーを作動させる。
3	ドライバは、ウォッシャーSWの操作を止め、洗浄液の吐出を止める。
4	システムは、洗浄液がきれいに除去されるまでワイパーの作動を継続する。
5	システムは、ワイパーの作動を停止させる。

ユースケースの検証

<ツール出力>

-- specification G (!p3 -> F !p1) is false
…
-- Loop starts here
-> State: 1.4 <-
p4 = FALSE
…

ウォッシャーSWがOffであればいずれ止まる

<検証結果>

「きれいに除去されるまで」の判定方法が未定義のため、ワイパーが停止しない

要求の不備？

[Yes]

終了条件を具体化し、要求仕様を修正

[No]

内部仕様にて具体化することがわかっているなら、ここでは不備として扱わず、次のステップで仕様を定義する。

ユースケース記述(修正版)

STEP	ステップ
1	ドライバは、ウォッシャーSW を操作して、フロントウィンドウに洗浄液を吐出する。
2	システムは、洗浄液が吐出している間、ワイパーを作動させる。
3	ドライバは、ウォッシャーSWの操作を止め、洗浄液の吐出を止める。
4	システムは、ワイパー作動を停止させる。
5	システムは、ワイパーの作動が完了してから 3秒後に、ワイパーを一往復作動させる。

検証例-1 後半

仕様の未定義部分を検出
さらに、場当たり的な仕様変更によって作りこんだ他の問題も検出

- 前ページの修正によって、ワイパーSW要求による動作中(UC1)にウォッシャー(UC2)を使用した場合に3秒間停止する別の障害を作り込んでしまったことを検出した

UC01 ユースケース記述

備考	UC01
1	本ユースケース実行中に、ウォッシャー連動（UC-WPR-02）が要求された場合には、 本ユースケースは一時中断される。

UC02 ユースケース記述

STEP	UC02ステップ
4	システムは、ワイパー作動を停止させる。
5	システムは、ワイパーの作動が完了してから 3秒後に、ワイパーを一往復作動させる。

ユースケースの検証

<ツール出力>

-- specification G ((p1 & p3) -> F ((!p3 & !p21 -> F (!p1 U p21)) -> (p1)))is false

…

UC01中は動作する

<検証結果>

ワイパー作動中にウォッシャーを利用して、ウォッシャーをoffにした後、ワイパーonでも3秒間ワイパーが停止してしまう

UC01作動中の復帰条件をUC02の備考に追加

UC02 ユースケース記述(再修正版)

備考	UC02
1	UC01ワイパー作動中に本ユースケースが呼び出された場合はstep5を実施せずにUC01に復帰する

検証例-2

仕様を詳細化する過程での
条件モレと仕様モレの検出

- オートモードでは雨滴レベルの判定結果に応じてワイパー間欠時間を決めるが、雨滴レベルの判定に失敗した場合の仕様が不在である
  → 検査用モデルのコンパイルに失敗するため検出可能

USDM記述

<ワイパー間欠時間判定>

WPR.01.03.11 以下の表に示すとおり、
雨適量レベルに応じて、ワイパー間欠時間レベルを決定する。

雨滴量レベル	ワイパー間欠時間レベル
0	0
1	12
2	10
3	8
4	6
5	4
6	2
7	1
8	0
9	0

動作モデル

```
wpmode : {off, auto, mint, mlow, mhigh};
p8  -- 動作させるワイパーの間欠時間と作動速度を判定する

next(wpint) := case
  wpmode = auto & p8 : 0..12;
  wpmode = mint : wpintlevel;
  wpmode = mlow : 0;
  wpmode = mhigh : 0;
  wpmode = off : 0;
esac;
```

line 240: case conditions are not exhaustive
このアルゴリズムには条件漏れが存在する

(auto∧p8)∨mint∨mlow∨mhigh∨off
が全ての条件を網羅していない

動作モデル(修正版)

```
next(wpint) := case
  wpmode = auto & p8 : 0..12;
  wpmode = auto & !p8 : XXXX;
  wpmode = mint : wpintlevel;
  wpmode = mlow : 0;
  wpmode = mhigh : 0;
  wpmode = off : 0;
esac;
```

雨滴レベルが不定の場合が未定義、
XXXの部分を決める必要がある

検証例-3

複数仕様間での不整合の検出

- 間欠ワイパーの間欠時間設定方法には複数ある
- “12段階の速度レベル”が要求されているが、どのような設定方法を使っても、間欠時間を11秒に設定することができない

COMN.01	指定された速度でワイパーを一往復作動させ、間欠時間停止した後、再度ワイパーを作動させる。
説明	・ワイパー間欠時間は、動作モードに応じて、12段階の速度レベルを設ける。 ・モータ回転速度は、動作モードに応じて、3段階のレベルを設ける。 ・モータ回転速度(r/min)の有効範囲は −750〜+750 とする

間欠時間判定の仕様（自動モード）

雨滴量レベル	ワイパー間欠時間レベル
0	0
1	12
2	10
3	8
4	6
5	4
6	2
7	1
8	0
9	0

間欠時間判定の仕様（手動モード）

スイッチレベル	ワイパー間欠時間レベル
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9

動作モデル

```
next(wpintlevel) := case
  wpmode = auto & rain in {0,8,9} : 0;
  wpmode = auto & rain = 1 : 12;
  wpmode = auto & rain = 2 : 10;
  wpmode = auto & rain = 3 : 8;
  wpmode = auto & rain = 4 : 6;
  wpmode = auto & rain = 5 : 4;
  wpmode = auto & rain = 6 : 2;
  wpmode = auto & rain = 7 : 1;
  wpmode = mint : wpintSWlevel;
  wpmode = mlow : 0;
  wpmode = mhigh : 0;
  wpmode = off : 0;
esac;

next(wpintSWlevel) := case
  wpmode = mint : 0..9;
  TRUE: 0;
esac;
```

検証例-3 続き

複数仕様間での不整合の検出

ワイパー間欠時間はモードによって別々のテーブルを使用して決定される。
どのテーブルにも定義されない時間帯があった。

検証結果

```
-- specification AG (EF wpint = 0)  is true
-- specification AG (EF wpint = 1)  is true
-- specification AG (EF wpint = 2)  is true
-- specification AG (EF wpint = 3)  is true
-- specification AG (EF wpint = 4)  is true
-- specification AG (EF wpint = 5)  is true
-- specification AG (EF wpint = 6)  is true
-- specification AG (EF wpint = 7)  is true
-- specification AG (EF wpint = 8)  is true
-- specification AG (EF wpint = 9)  is true
-- specification AG (EF wpint = 10)  is true
-- specification AG (EF wpint = 11)  is false
-- specification AG (EF wpint = 12)  is true
```

wpintは11秒にならない。

検証例-4

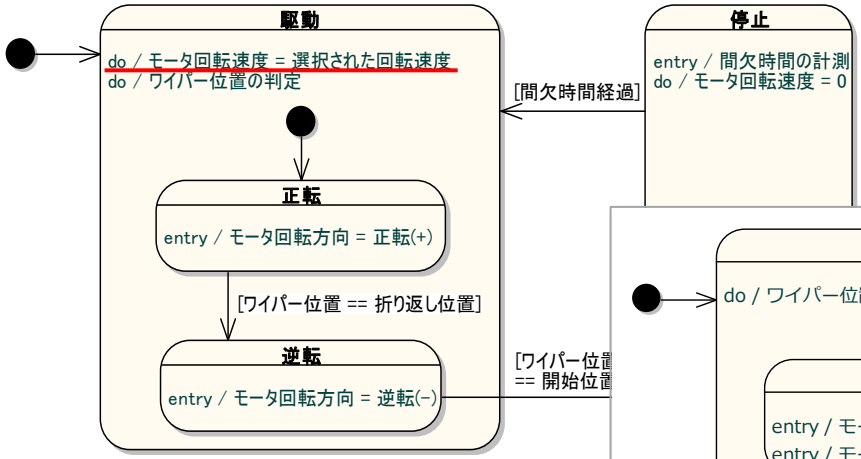
上位要求と詳細化した設計仕様との不整合の検出

モータ制御仕様に関して、機能間での意図しない競合

USDM

<モータの駆動>

COMN. 01.02.02	以下の手順で、モータの駆動と停止を繰り返す。 (1) 指定された回転速度でモータを正転させて、ワイパーを開始位置(=停止位置)から折り返し位置まで移動させる。 (2) ワイパーが折り返し位置に到達したら、開始位置に来るまで、指定された回転速度でモータを逆転させる。 (3) 再びワイパーが開始位置へ到達したら、指定された間欠時間モータの回転を停止させる。 (4) 指定された間欠時間経過したら、(1)へ戻り、ワイパーの作動を繰り返す。



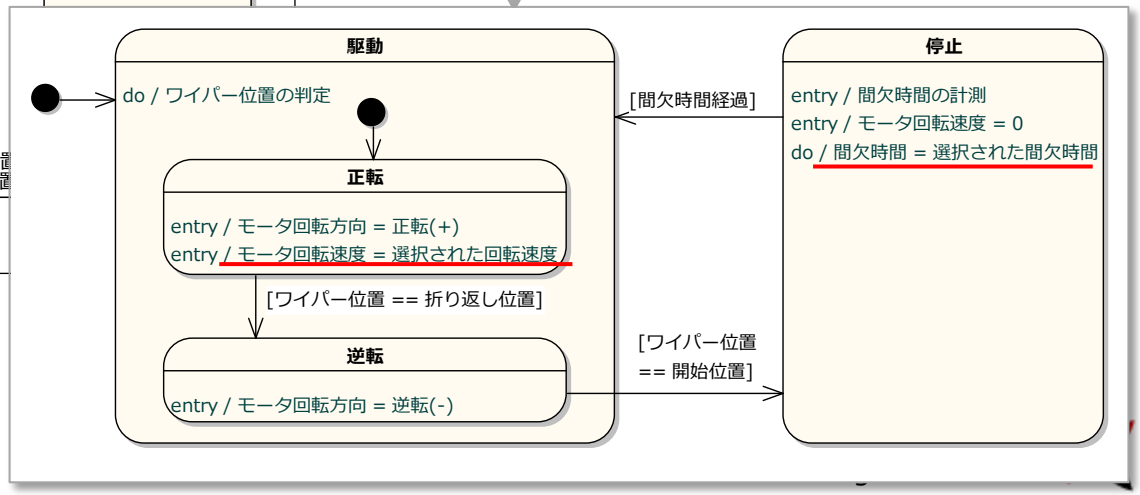
USDM の検証

<検証結果>

この状態マシンでは、二つの上位要求と矛盾する可能性がある

- ✓ 間欠時間指定：間欠時間を変更してもその時の間欠時間が経過するまで反映されない
- ✓ ワイパー速度変更：「速度の変更はホームポジションに戻った時に反映する」と矛盾する

スイッチON判定の仕様を
期待する振舞いに修正



検証例-5

不適切な仕様化によって上位要求が満たされないことを検出

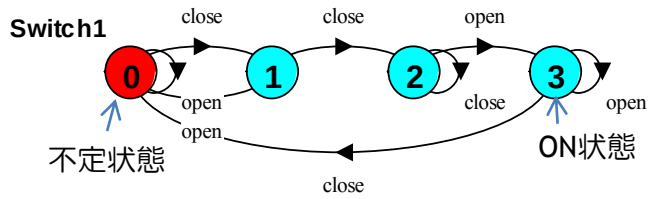
- スイッチの読み取り仕様に問題があるため、ON状態が継続してしまう可能性を検出した

USDM仕様

< ウォッシャースイッチ押下判定 >	
WPR. 02.01.01	以下の条件が成立した場合、ウォッシャースイッチ状態が ON であると判定する。 ・ ウォッシャースイッチ接点の CLOSE が規定サンプリング回数(2)以上連続して ・ ウォッシャースイッチ接点が OPEN になった
WPR. 02.01.02	以下の条件が成立した場合、ウォッシャースイッチ状態が OFF であると判定する。 ・ ウォッシャースイッチ状態が ONでない

動作モデル

<要求仕様の振舞いモデル>



<検証結果>

-- specification G (!p3 -> F !p2) is false
…

OPENになった瞬間にしかON状態にならず、
スイッチ押下の間 吐出するという要求と矛盾する

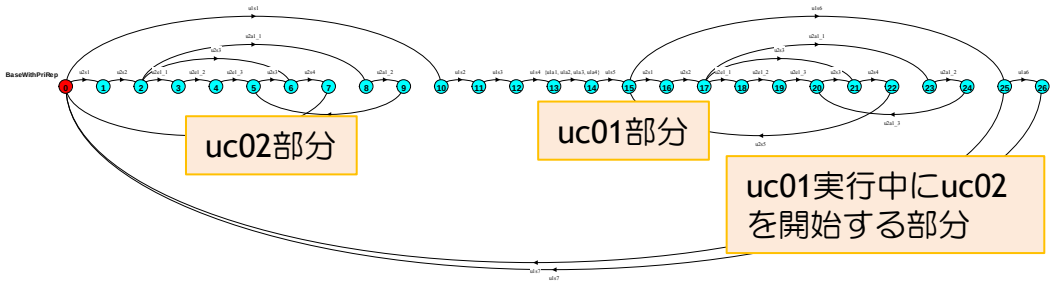
スイッチON判定の仕様を
期待する振舞いに修正

USDM仕様(修正版)

< ウォッシャースイッチ押下判定 >	
WPR. 02.01.01	以下の条件が成立した場合、ウォッシャースイッチ状態が ON であると判定する。 ・ ウォッシャースイッチ接点の CLOSE が規定サンプリング回数(2)以上連続して ・ ウォッシャースイッチ接点が OPEN になり、 OPEN が継続している間
WPR. 02.01.02	以下の条件が成立した場合、ウォッシャースイッチ状態が OFF であると判定する。 ・ ウォッシャースイッチ状態が ONでない

要求検証事例 まとめ

- UC間/UC内/USDMの各レベルで検証を実施可能



UC間

UC内
- step1 → [](p1→<>p2)
- step2 → [](p2→¬p3)
- …

詳細化

USDM
- spec1
- spec2 → next(..) =case…
- …

宣言的仕様

操作的仕様

UC検証
例-1前半
例-2

USDM検証
例-5

UC/USDM検証
例-3
例-4

UC/USDM検証
例-1後半

形式化　形式モデル化　形式検証

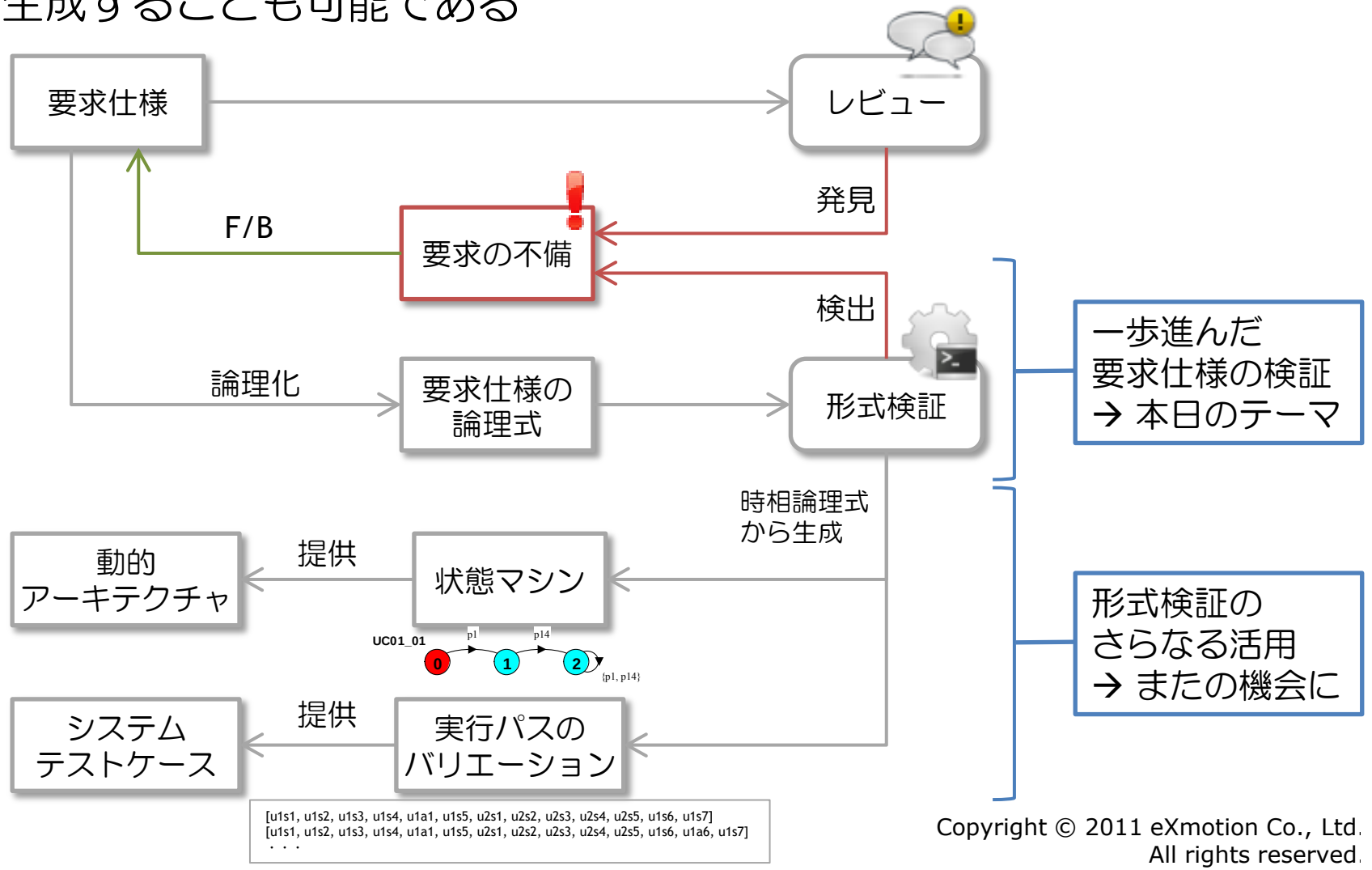
まとめ

全体のまとめ

- 一般に形式検証は難解とされているがUSDMで表現された要求仕様があれば“一手間かける”ことで形式検証が可能となる
  - 設計・実装以前に間違いを検出・修正できる
  - テストでは発見が困難な複合的な間違いを発見できる
- 検証のレベルをリソースに応じて選ぶことができる
  - 例えばユースケースの検証だけでも有効である
  - 機能安全でいう高ASILなど、 重要な要求を選択して検証しても良い

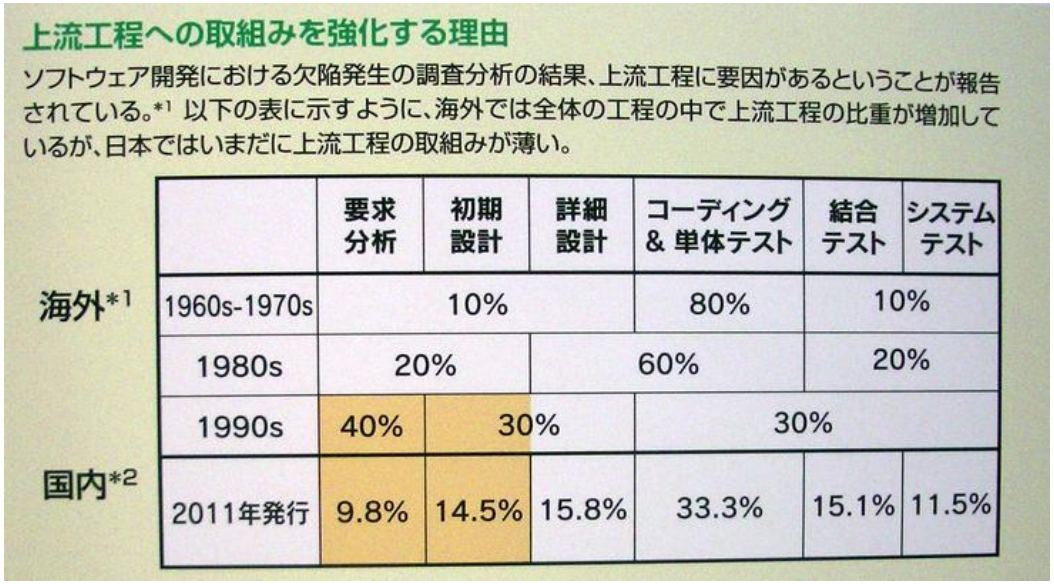
【参考】形式検証のさらなる活用

- 形式化されたモデルを使えば設計状態マシンやシステムテストケースを生成することも可能である



さいごに

- 日本では要求工程に掛ける工数が未だに少なく、欠陥を先送りしてしまっている



上流工程への取組みを強化する理由

ソフトウェア開発における欠陥発生の調査分析の結果、上流工程に要因があるということが報告されている。*1 以下の表に示すように、海外では全体の工程の中で上流工程の比重が増加しているが、日本ではいまだに上流工程の取組みが薄い。

		要求分析	初期設計	詳細設計	コーディング & 単体テスト	結合テスト	システムテスト
海外*1	1960s-1970s	10%			80%	10%	
	1980s	20%		60%		20%	
	1990s	40%	30%		30%		
国内*2	2011年発行	9.8%	14.5%	15.8%	33.3%	15.1%	11.5%

SECソフトウェア開発データ白書2010-2011, p. 204 - 205

- 「このままでは駄目だ！」そこで
  - まず、USDMを導入する
  - それだけではもったいないので形式検証にも挑戦してみたら?

Q&A