状態遷移モデルを活用した
赤外線自動テスト装置の紹介

2010年1月28日

キャッツ株式会社
ソフトウェア事業部
田村　政和

Communication
Art
Technology
Systems

目次

格言

- “If it can happen, it will happen.”
  「起こる可能性のあることは、いつか実際に起こる。」
  （マーフィーの法則より）

- “Anything that can go wrong will go wrong – at the worst possible moment.”
  「うまく行かなくなりうるものは何でも、うまく行かなくなる－しかも最悪のタイミングで」
  （マーフィーの法則より）

- 情報システムの開発では、計画時の2倍の費用と2倍の時間がかかり、1/2の機能しか実現しない
  「222の法則」（作者不明）

引用元：
マーフィーの法則　－Wikipedia－、
http://ja.wikipedia.org/wiki/%E3%83%9E%E3%83%BC%E3%83%95%E3%82%A3%E3%83%BC%E3%81%AE%E6%B3%95%E5%89%87
マーフィーの法則がIT版になって帰ってきた！　－@IT情報マネジメント－
http://www.atmarkit.co.jp/im/cits/serial/murphy/01/01.html

Communication
Art
Technology
Systems

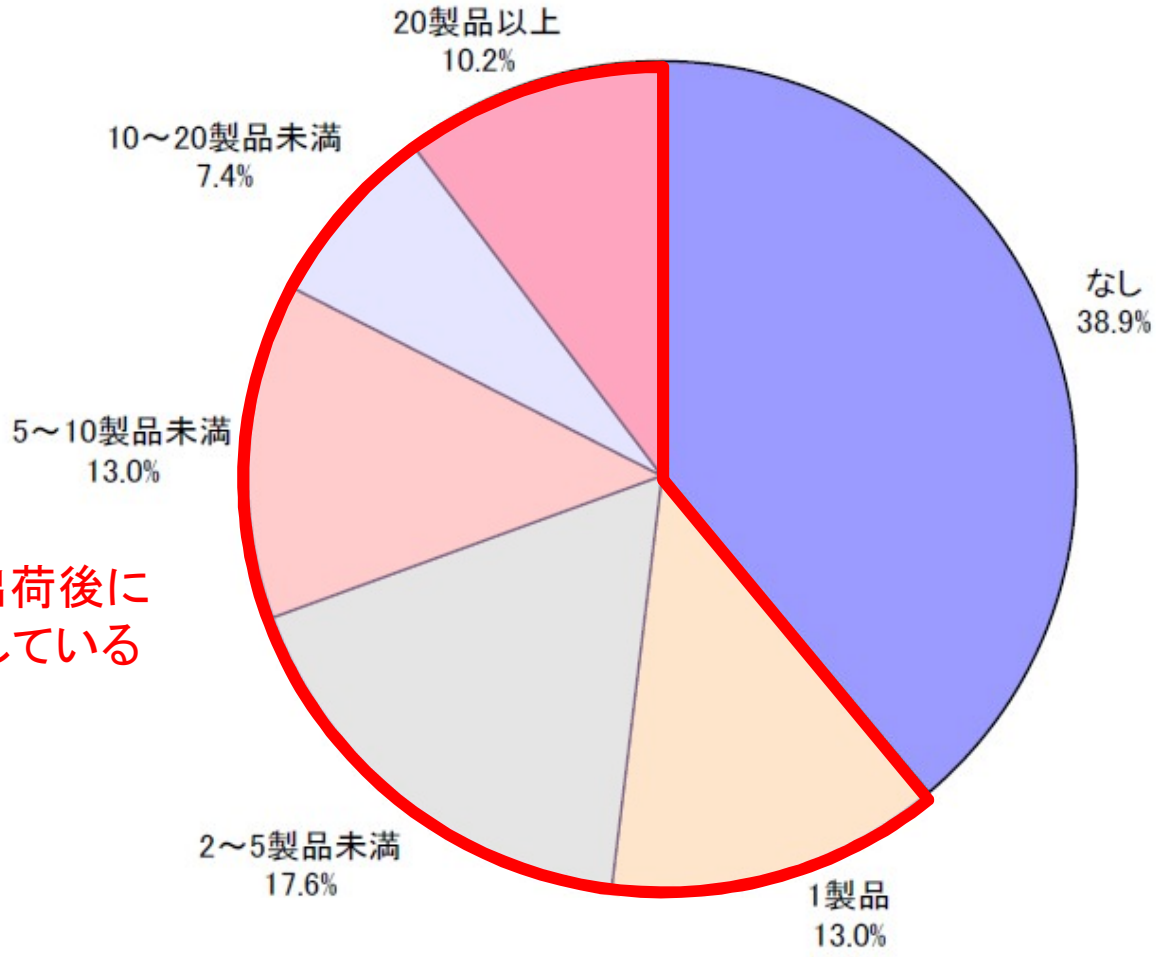
伝えたいこと

◆ソフトウェア開発において、
ツールを活用することで、
開発効率／品質の向上ができる

◆ツールの本質を理解することが
重要になる

1. 組込み開発におけるテストの現状

製品出荷後に不具合を起こした製品数

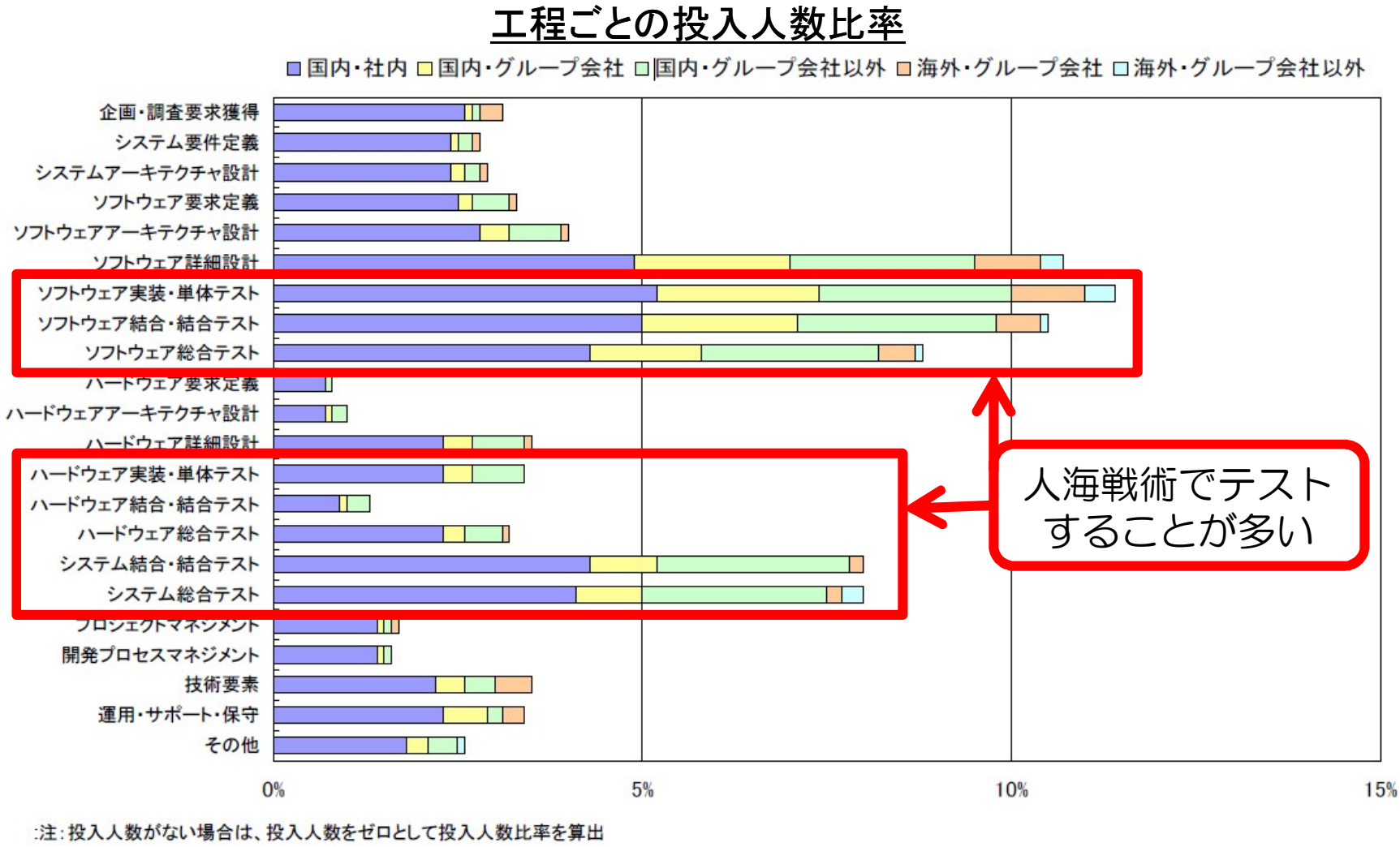


約6割は製品出荷後に
不具合を起こしている

2009年版　組込みソフトウェア産業実態調査報告書：経営者・事業責任者向け調査
https://sec.ipa.go.jp/download/files//report/200706/es07rALL.zip

Communication Art Technology Systems

1. 組込み開発におけるテストの現状

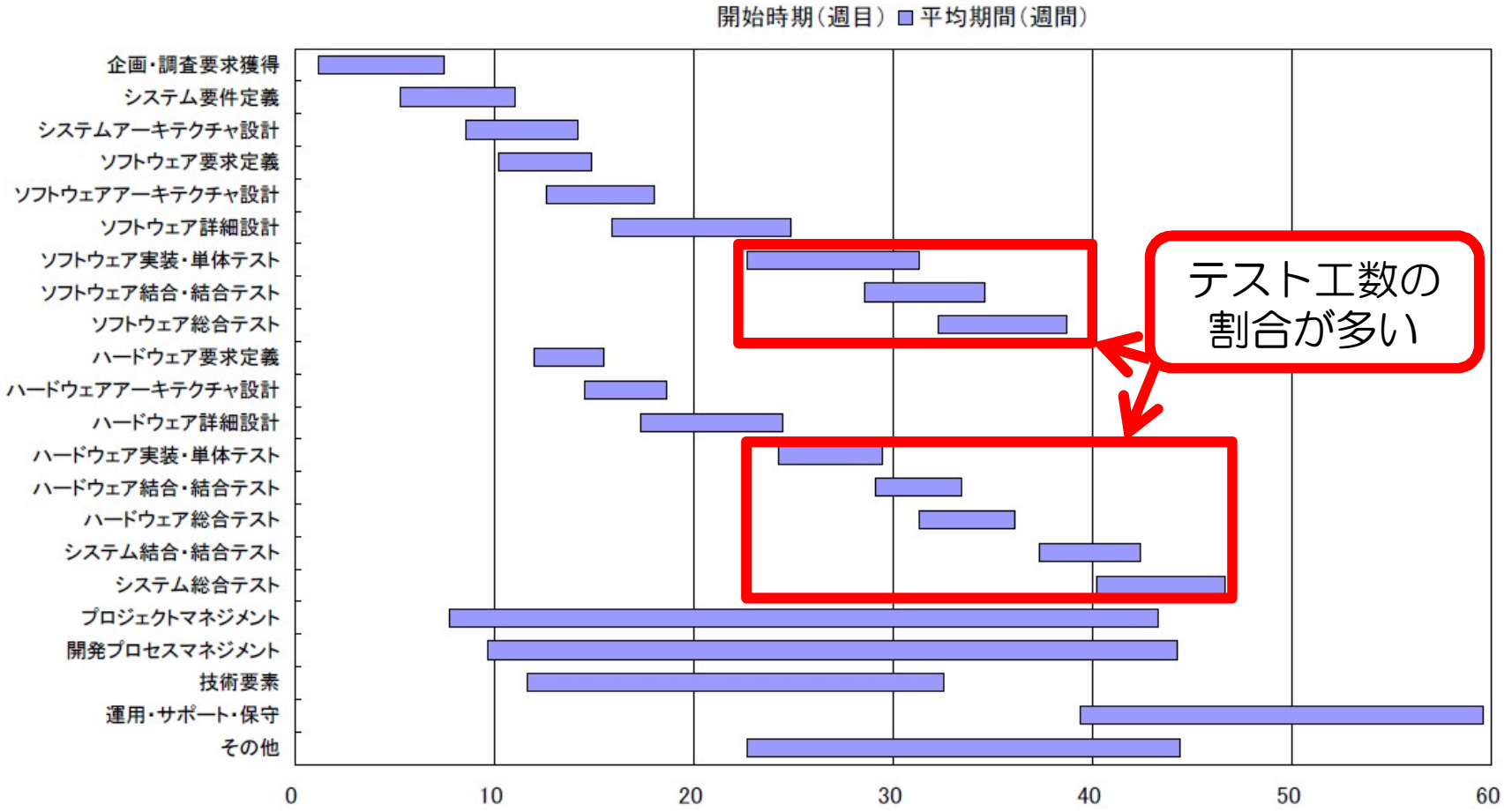


2009年版　組込みソフトウェア産業実態調査報告書：経営者・事業責任者向け調査
https://sec.ipa.go.jp/download/files//report/200706/es07rALL.zip

Communication Art Technology Systems

1. 組込み開発におけるテストの現状

各工程の平均開始時期とその平均期間

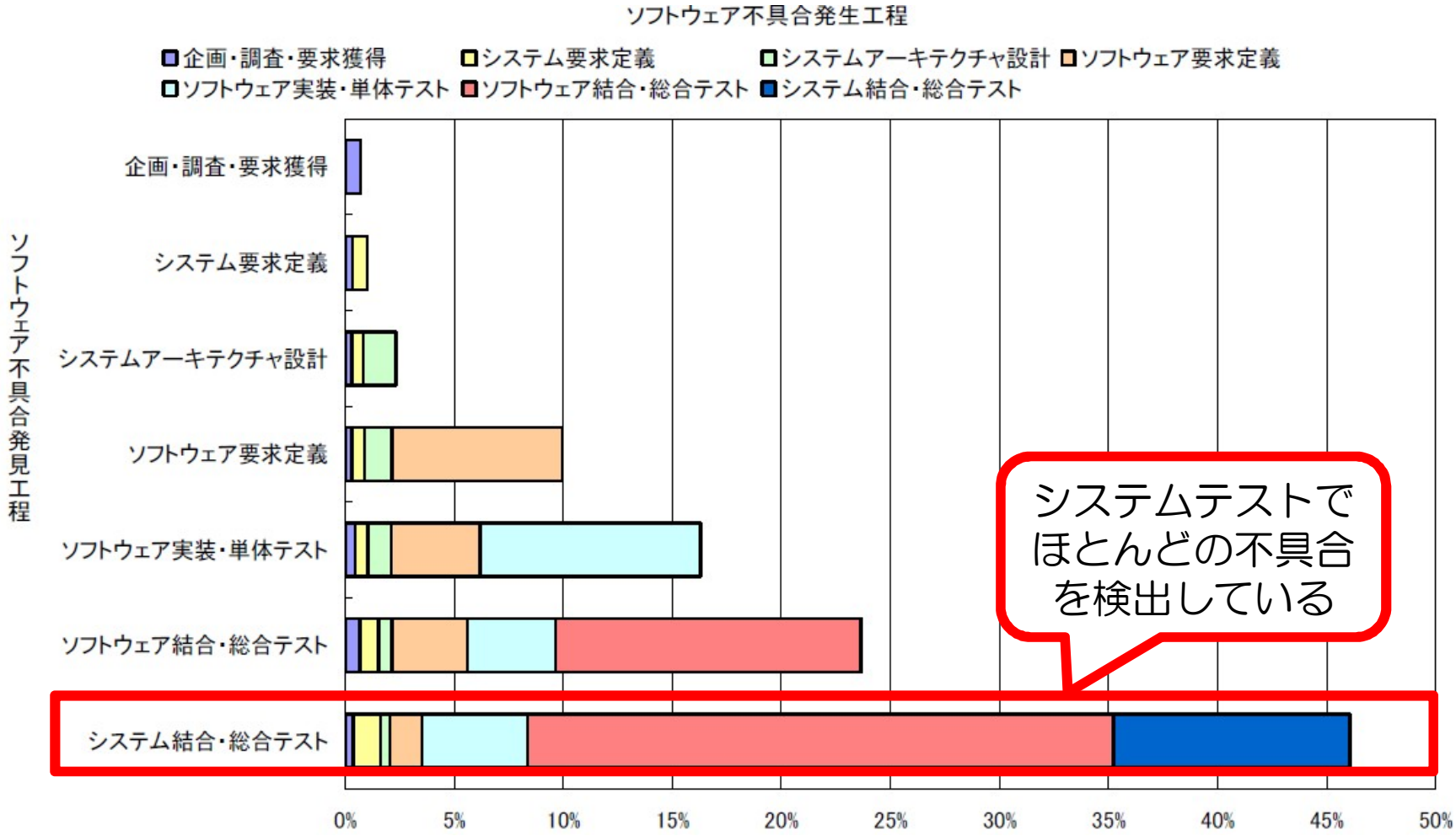


2009年版　組込みソフトウェア産業実態調査報告書：経営者・事業責任者向け調査
https://sec.ipa.go.jp/download/files//report/200706/es07rALL.zip

Communication
Art
Technology
Systems

1. 組込み開発におけるテストの現状

ソフトウェア不具合発見工程別件数比率と発生工程の内訳

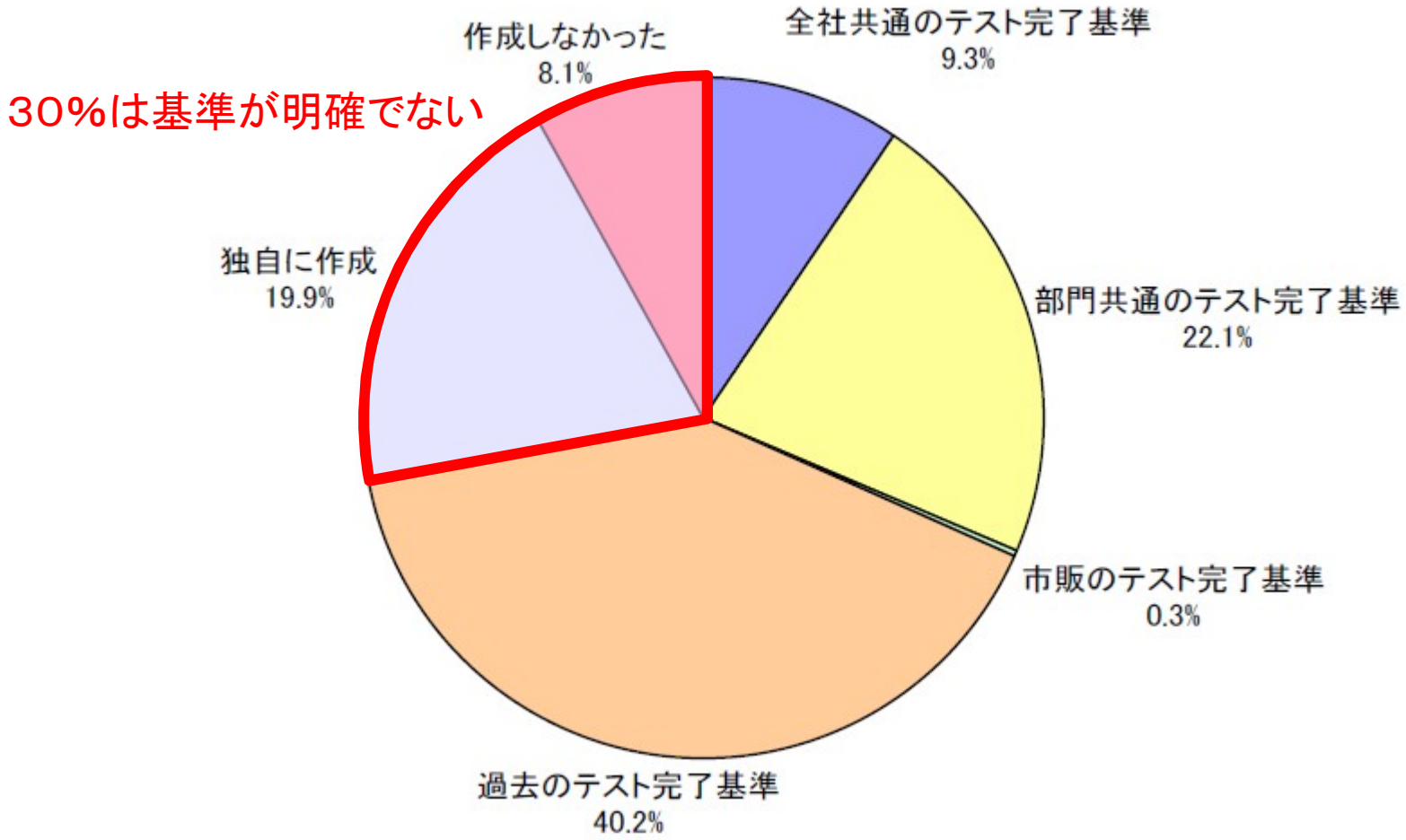


2009年版　組込みソフトウェア産業実態調査報告書：経営者・事業責任者向け調査
https://sec.ipa.go.jp/download/files//report/200706/es07rALL.zip

Communication Art Technology Systems

1. 組込み開発におけるテストの現状

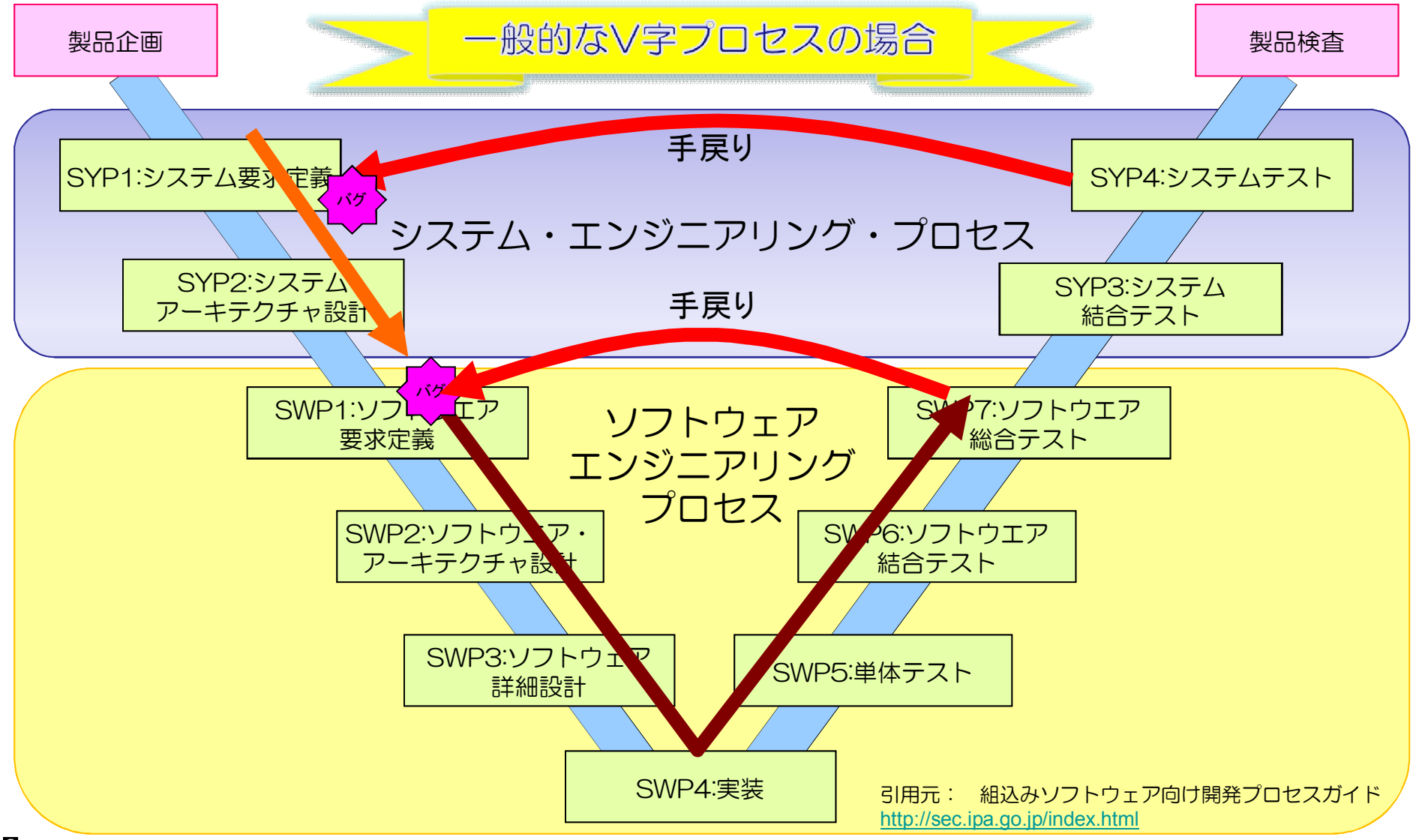
テスト完了基準



2009年版　組込みソフトウェア産業実態調査報告書：経営者・事業責任者向け調査
https://sec.ipa.go.jp/download/files//report/200706/es07rALL.zip

Communication Art Technology Systems

1. 組込み開発におけるテストの現状

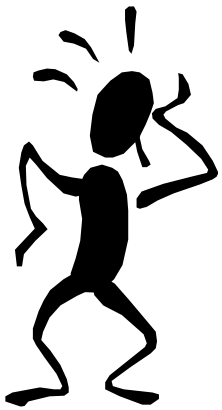


引用元： 組込みソフトウェア向け開発プロセスガイド
http://sec.ipa.go.jp/index.html

Communication Art Technology Systems

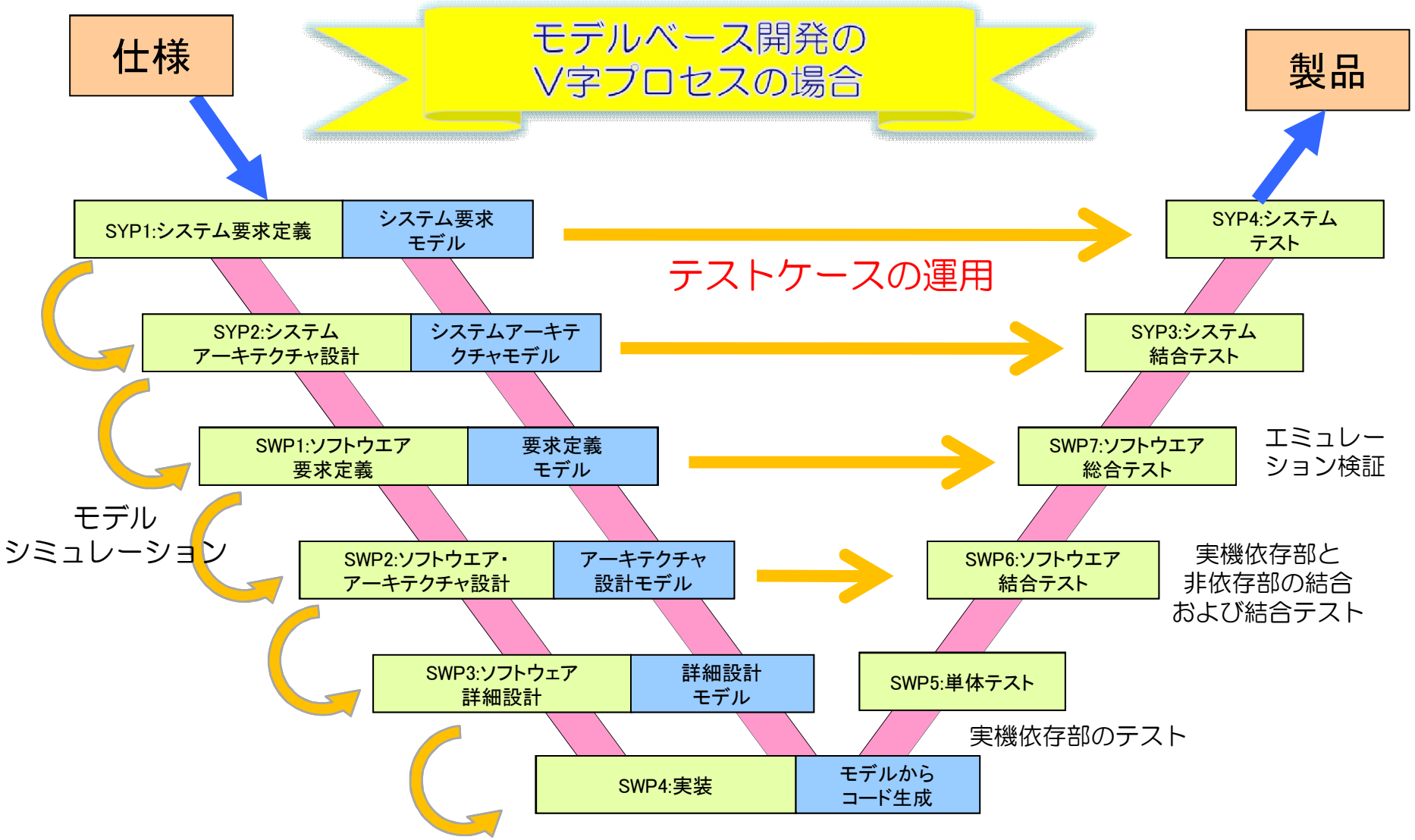
1. 組み込み開発におけるテストの現状

◆現状から見えてくる課題
- システムテストが最後の砦である
- 人海戦術でテストすることが多い
- テストにかかる工数の割合が多い
- テスト完了基準が明確でない場合がある
- 過去のテスト完了基準で実施していることが多い



工数の削減、担当者依存の改善、人為的なミス削減のため、テストケースの自動生成とシステムテスト自動化の方法を紹介する

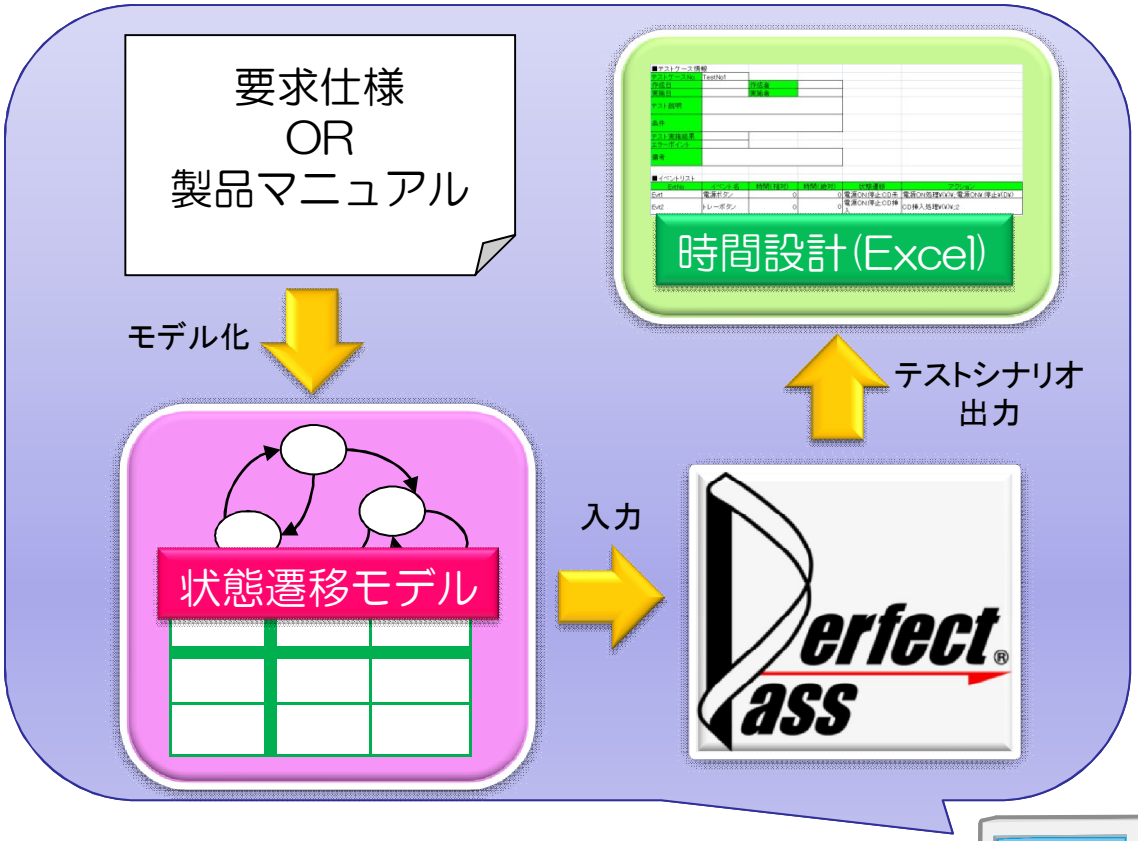
2. システムテストを改善する方法

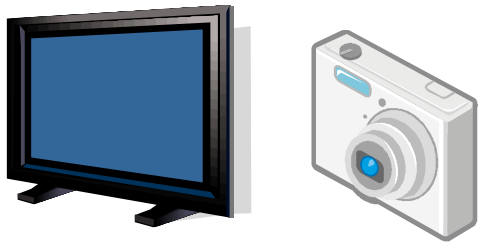


モデル・ベース開発は上流行程の成果物を下流工程へ運用できる

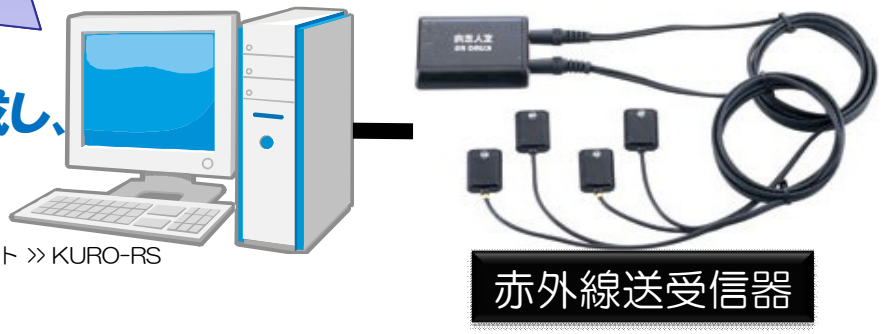
Communication Art Technology Systems

2. システムテストを改善する方法





赤外線イベント



状態遷移モデルからテストシナリオを自動生成し、赤外線通信で自動テストを実施する

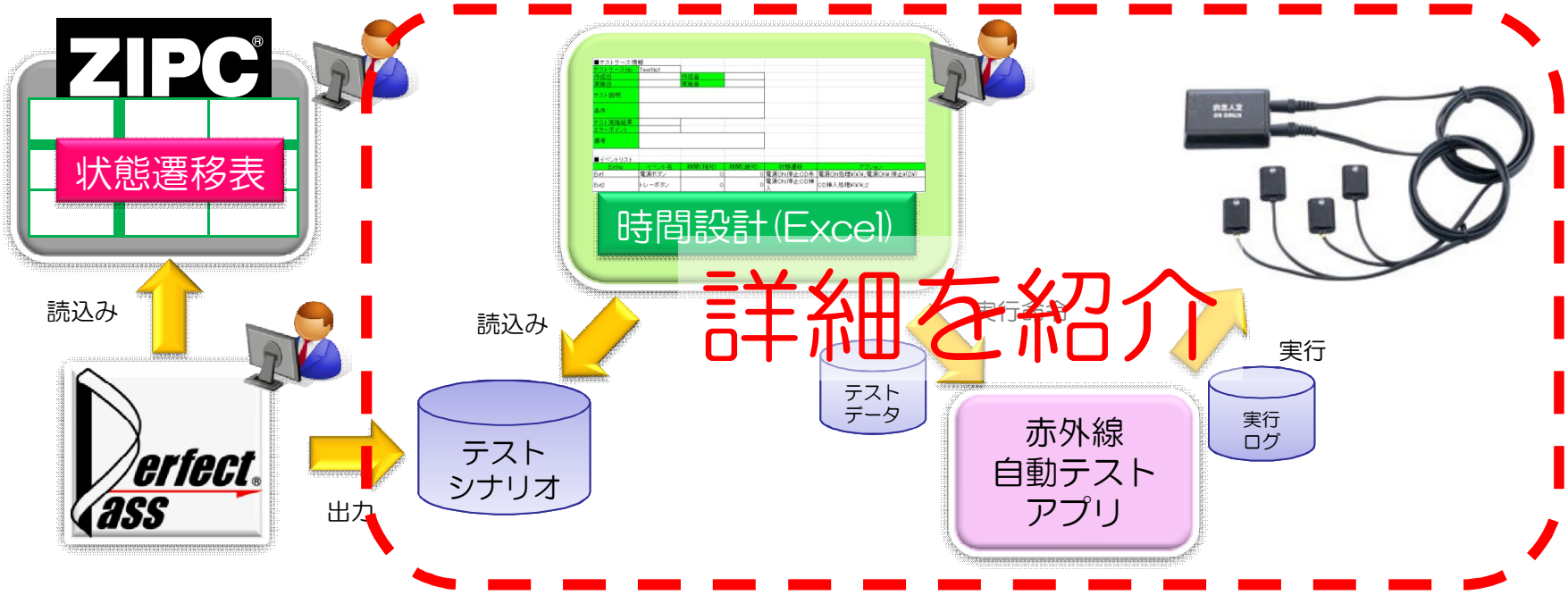
画像引用元：　玄人志向 一商品一覧 ≪ キワモノシリーズ ≫ マルチメディア ≫ 学習リモコンキット ≫ KURO-RS
http://www.kuroutoshikou.com/modules/display/?iid=928

Communication Art Technology Systems

3. 赤外線自動テスト装置の紹介

◆ 赤外線自動テストツールの概要

1）ZIPCで状態遷移表を作成する
2）PerfectPassでテストケース群（テストシナリオ）を自動生成する
3）赤外線自動テストツール（エクセル）で、タイミング設計、
　　イベントの詳細設定をする
4）赤外線リモコンで自動テストする

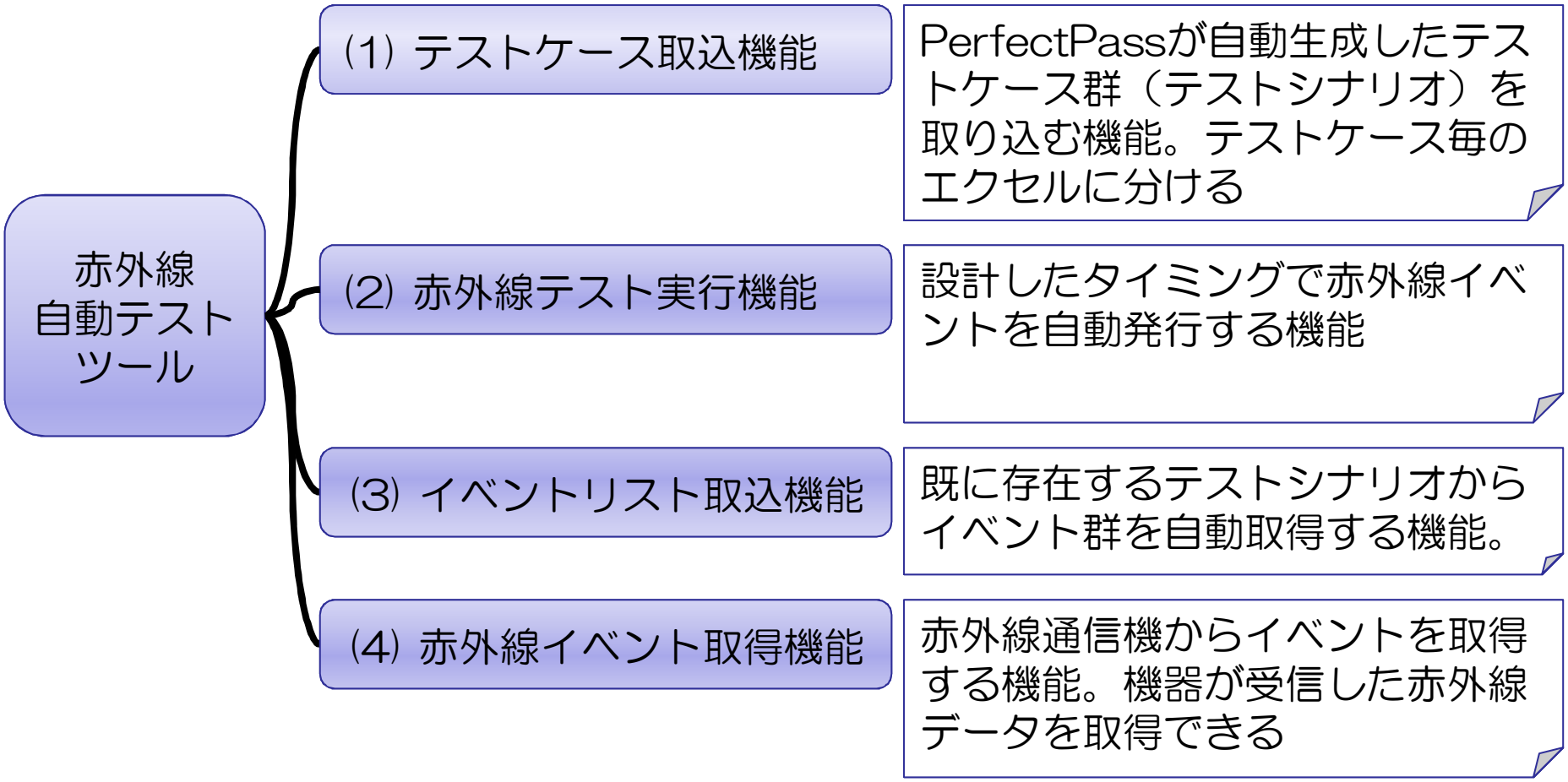


画像引用元：　玄人志向 ー商品一覧 ≪ キワモノシリーズ ≫ マルチメディア ≫ 学習リモコンキット ≫ KURO-RS
http://www.kuroutoshikou.com/modules/display/?iid=928

Communication Art Technology Systems

3. 赤外線自動テスト装置の紹介

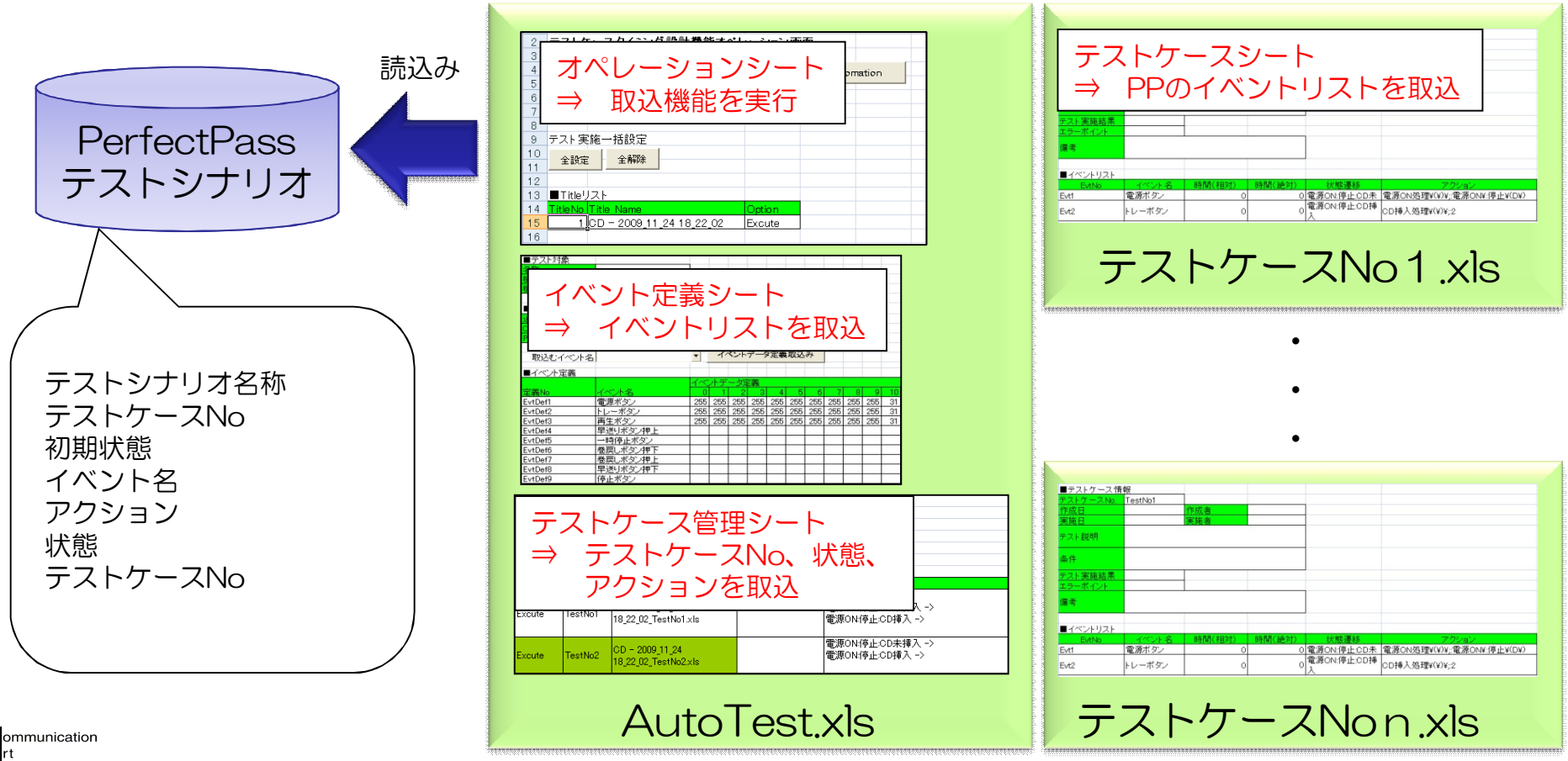
◆ 赤外線自動テストツールの機能構成



3. 赤外線自動テスト装置の紹介

(1) テストケース取込機能

PerfectPassが出力するテストシナリオ（CSV）を読み込み、
AutoTest.xls、テストケース.xlsに取り込む
（エクセルなので、手入力も可能）



3. 赤外線自動テスト装置の紹介

(1) テストケース取込機能（ユーザ設定項目）

取り込んだテストケースにイベント発行タイミングを設計する

■テストケース情報

テストケースNo.	TestNo1		
作成日		作成者	
実施日		実施者	
テスト説明			
条件			
テスト実施結果			
エラーポイント			
備考			

イベントの発行タイミングを設計する

■イベントリスト

EvtNo	イベント名	時間(相対)	時間(絶対)	状態遷移	アクション
Evt1	電源ボタン	1000	1000	電源OFF=> 電源ON:停止:CD未挿入	電源ON処理();
Evt2	トレーボタン	1000	2000	電源ON:停止:CD未挿入=> 電源ON:停止:CD挿入	CD挿入処理();

テストケース.xls

3. 赤外線自動テスト装置の紹介

(1) テストケース取込機能（ユーザ設定項目）

イベントに実際の赤外線データを割り当てる

■テスト対象

名称	
機器バージョン	
機器番号	

■テスト方法

通信モジュール	KURO-RS
COM番号	1
PORT	1

取込むイベント名 [▼] イベント

イベントに赤外線
データを割り当てる

■イベント定義

定義No	イベント名	イベントデータ定義 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
EvtDef1	電源ボタン	63	0	0	0	122	25	0	0	1	12	0	0	0	0	0	224
EvtDef2	トレーボタン	1	0	96	0	0	0	0	0	0	0	0	0	0	0	0	0
EvtDef3	再生ボタン																
EvtDef4	停止ボタン																
EvtDef5	早送りボタン押下																
EvtDef6	早送りボタン押上																
EvtDef7	一時停止ボタン																
EvtDef8	巻戻しボタン押下																
EvtDef9	巻戻しボタン押上																

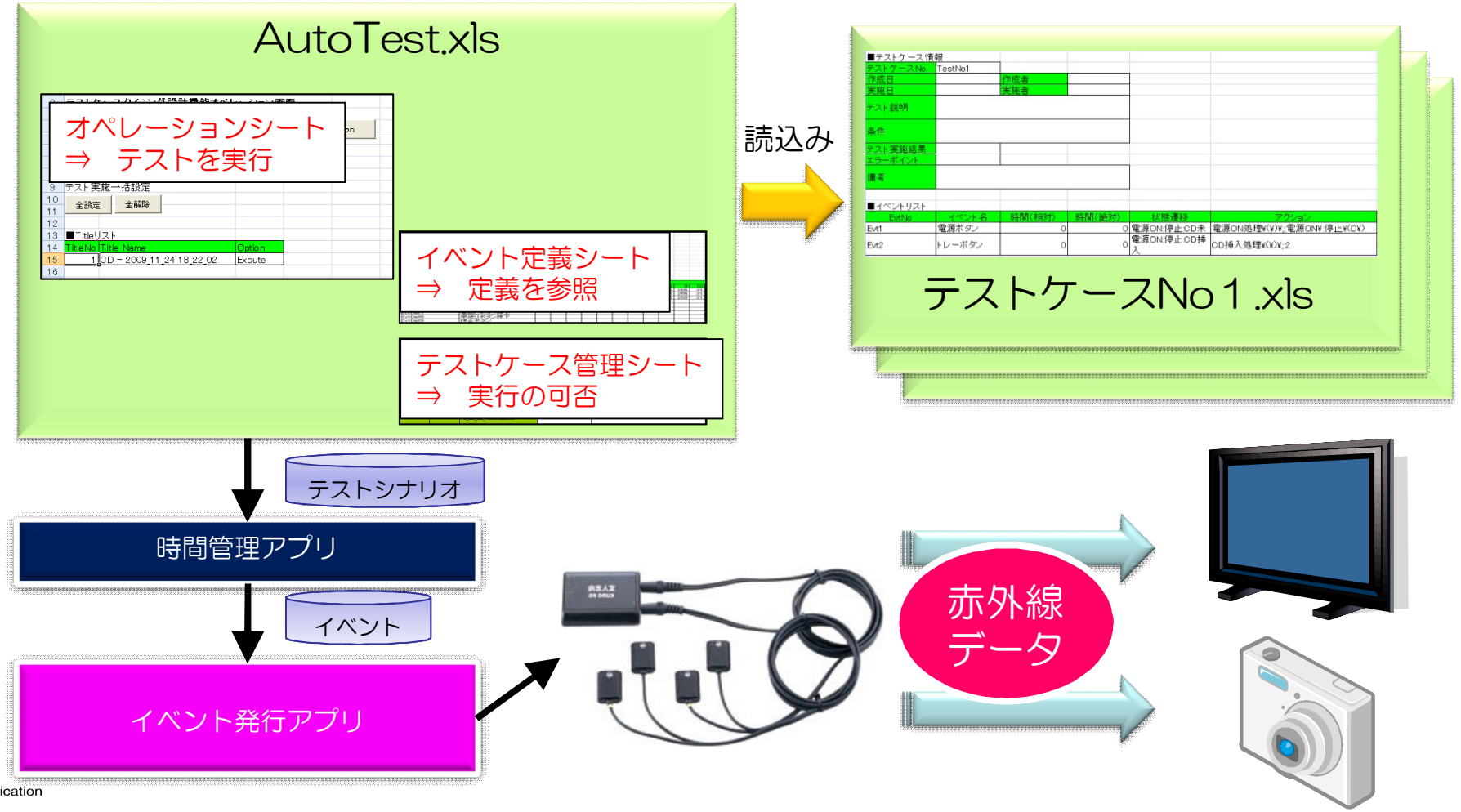
AutoTest.xlsのイベント定義シート

Communication Art Technology Systems

3. 赤外線自動テスト装置の紹介

(2) 赤外線自動テスト機能

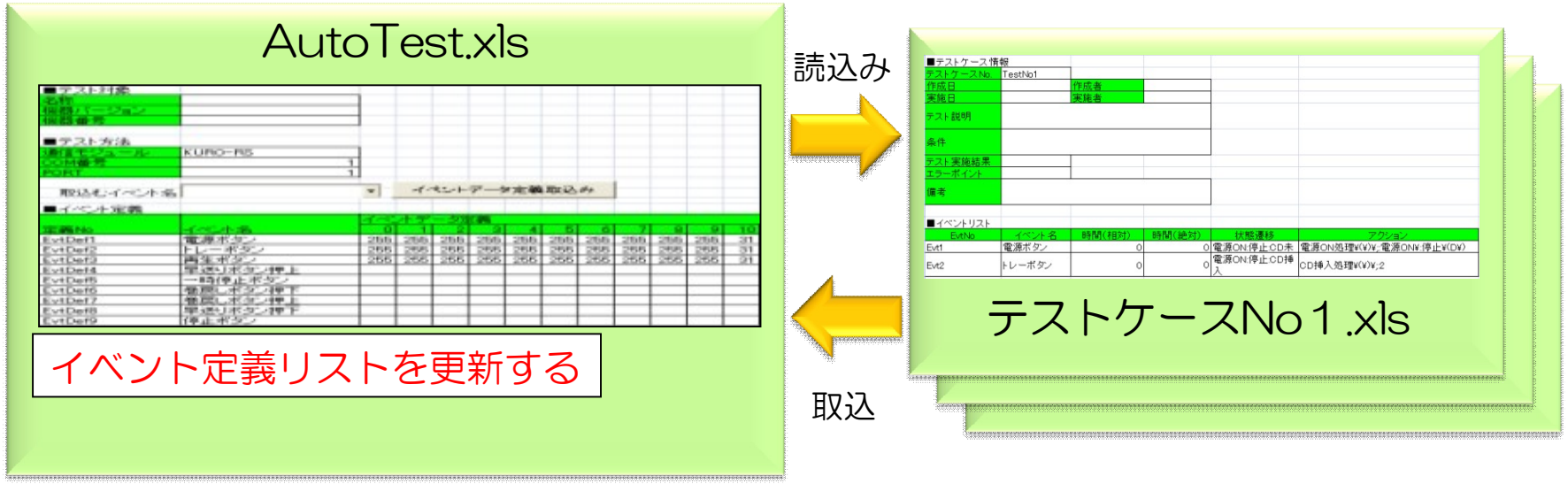
テスト実行ボタンでテストを赤外線テストを自動実行する



3. 赤外線自動テスト装置の紹介

(3) イベントリスト取得機能

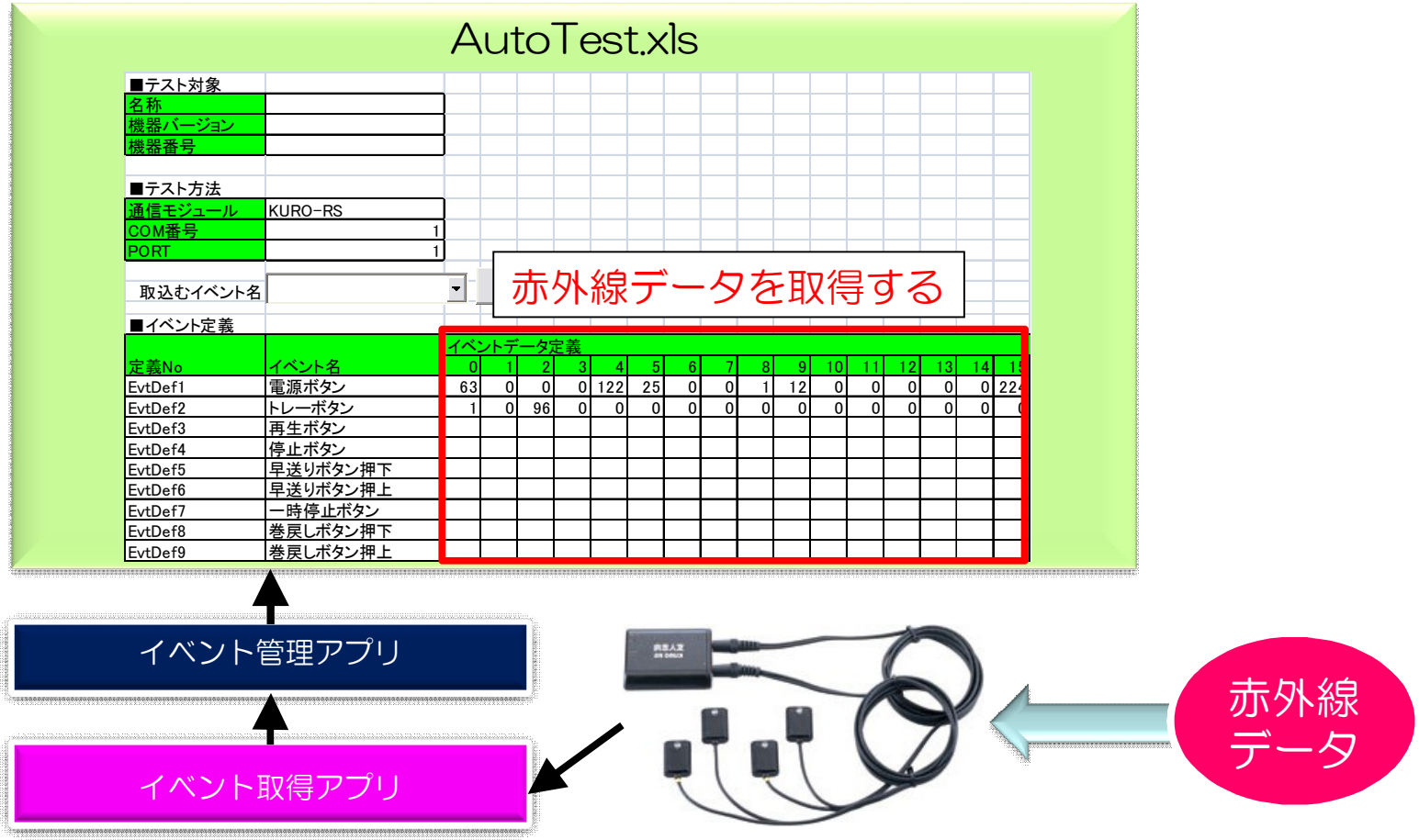
テストケース.xlsに記述されているイベントをイベント定義リストに取り込む
⇒　ユーザが直接打ち込んだテストケースの場合などに便利



3. 赤外線自動テスト装置の紹介

(4) 赤外線データ取得機能

赤外線リモコン（KURO-RS）で受信したデータをイベント定義データに反映させる。イベント定義を作成する際の手作業によるミスを削減できる。また、赤外線送信データに誤りがないことの確認もできる。



3. 赤外線自動テスト装置の紹介(デモ)

◆ 学習用リモコンカーへの適用例
弊社で組込みソフトウェア研修で使用している学習用のリモコンカーを対象にし、赤外線自動テストを適用した。

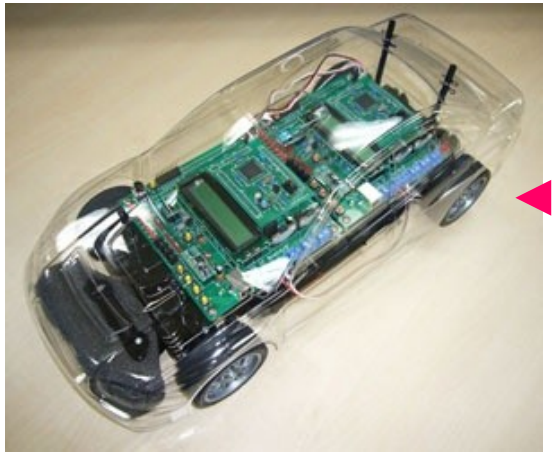
◆ 学習リモコンの概要

■ 高性能32ビットマイコン搭載
富士通 FR60 Lite（MB91F273）

■ マイコンのフラッシュメモリ書き換えはUSB経由で簡単
さらに、ＵＳＢバスパワー動作でＡＣアダプタ不要
モータ駆動時は外部電源使用で5Vレギュレータ搭載

■ 充実した周辺デバイスで様々なアプリケーションに対応
入力：　プッシュスイッチ 8個
赤外線リモコン受信ユニット
3軸加速度センサ（アナログ値）
VR（アナログ値）
出力：　赤色ＬＥＤ 8個、　赤外線LED 1個
液晶表示器（16桁2行）
小型(1.5A)DCモータドライバ 2個
圧電ブザー
I/F：　USB、CAN、シリアルI/F

■ サイズ　奥行120mm、幅125mm、高さ27mm

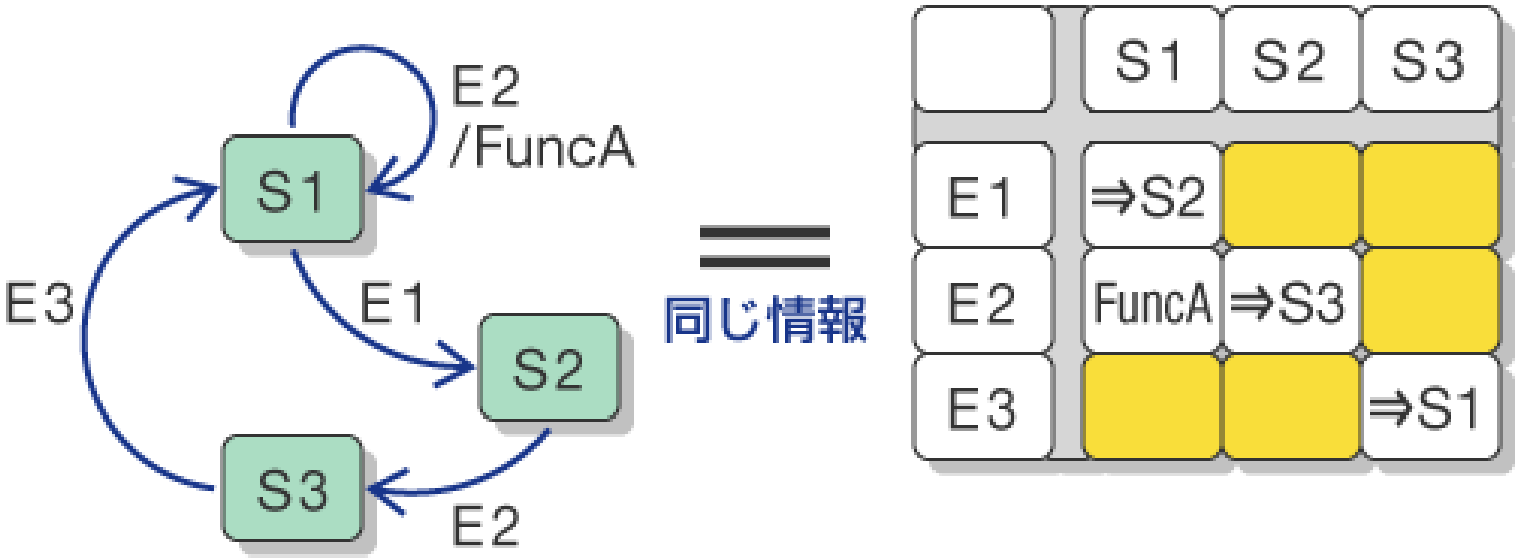


Communication Art Technology Systems

4. 状態遷移モデルとは

状態遷移図（STD/SC）／状態遷移表（STM/ST）

状態遷移図　　状態遷移表



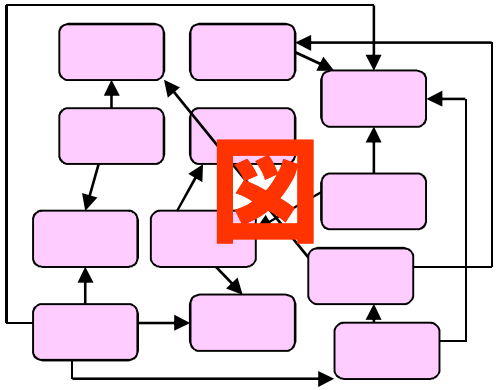
状態遷移図とは？

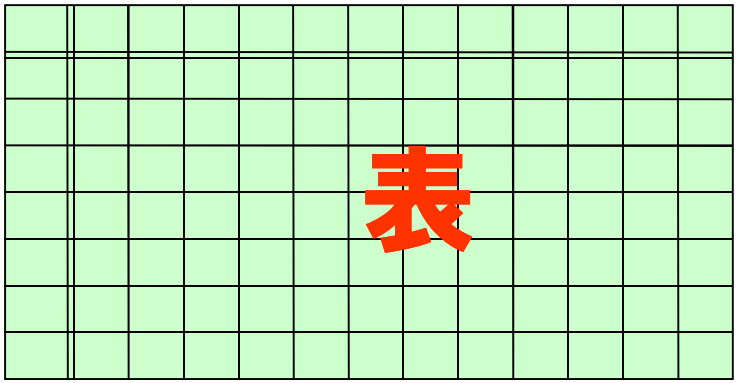
ある事物が、さまざまなイベント（事象）によってステート（状態）が移り変わる様子を表現した図。

状態遷移表とは？

ある事物が、さまざまなイベント（事象）によってステート（状態）が移り変わる様子を表現した表。

4. 状態遷移モデルとは





状態遷移図の特徴	状態遷移表の特徴
概要を把握するのには適している	概要を把握しにくい
異常・例外ケースを入れると図の収拾がつかない	異常・例外ケースも全て網羅している
仕様・設計に基づいて作成するので、仕様・設計の不備を見付けにくい	枠を埋めていくことで、仕様・設計の不備が見付かる
モレ・ヌケし易い（書き忘れたのか　書かなかったのかの区別ができない）	遷移しないことも明記できるのでモレ・ヌケを防止できる
ドキュメントの変更に手間取る（配置空間や結線を意識する）	行・列の変更だけで済む
イベントや処理が分散しているので類似性を見付けにくい	イベントや処理の類似性を見付け易い

CATS Communication Art Technology Systems

4. 状態遷移モデルとは

◆ どちらの状態遷移モデルを使えばよいか？

開発フェーズや実現したいことによって、どちらの状態遷移モデルを使用するか考慮すべき

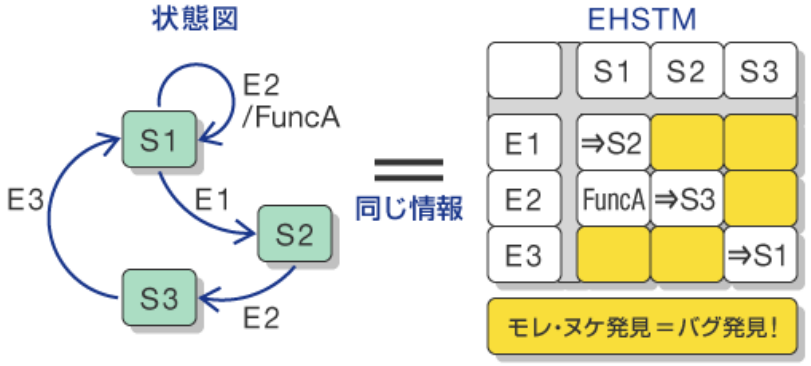
状態遷移図で正常系を設計
⇒　概要が解りやすい
（正常系を表現している）

状態遷移表で異常系を設計
⇒　モレ・ヌケを発見できる
（異常系を表現できる）

◆ 使い分けの例

＜状態遷移図＞
・仕様／設計書などの説明に使用する

＜状態遷移表＞
・実装前の仕様／設計検証に使用する



5. 状態遷移モデルとテスト

◆ そもそもテストとは

テストは、要求／仕様をテスト対象が満たしていることを確認することである。

◆ 状態遷移モデルを活用したテストとは

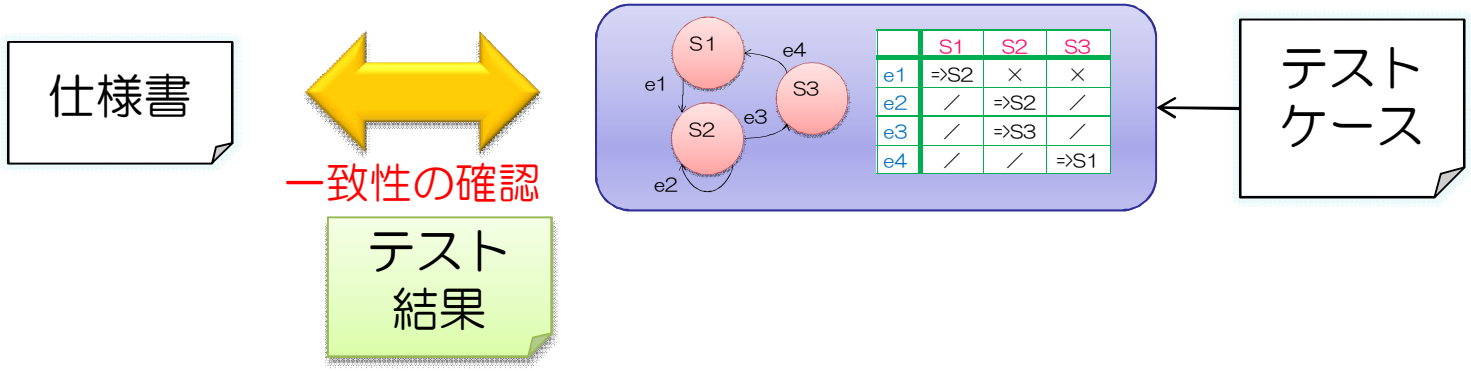
状態遷移モデルを活用したテストは、状態遷移モデルとテスト対象が同様の動作をしていることを確認するテストである
弱双模倣性（バイシミュレーション）が中心となる。

具体的な例を用いて説明する

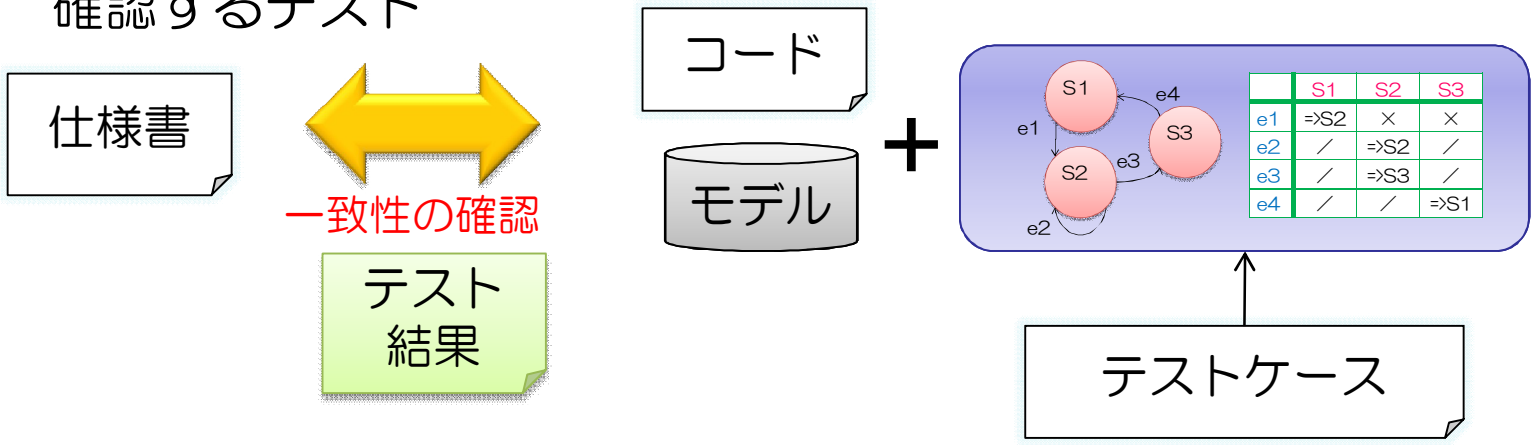
5. 状態遷移モデルとテスト

◆ 状態遷移モデルを活用したテストの例

①状態遷移モデルが仕様通りに設計されていることを確認するテスト



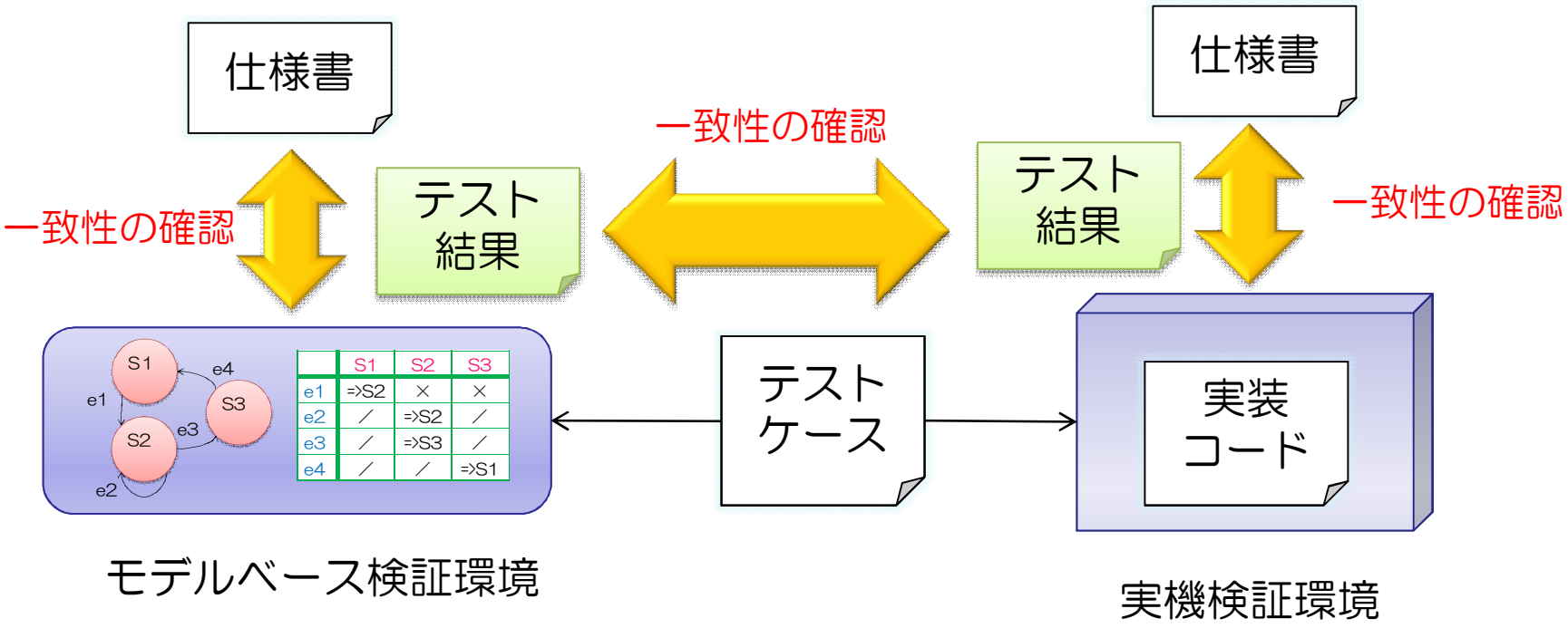
②状態遷移モデルを組合せたとき、他モジュールへ影響無いことを確認するテスト



5. 状態遷移モデルとテスト

◆ 状態遷移モデルを活用したテストの例

③状態遷移モデルテストで活用したテストケースを実装コードに活用し、同様の動作をすることを確認するテスト



	S1	S2	S3
e1	=>S2	×	×
e2	／	=>S2	／
e3	／	=>S3	／
e4	／	／	=>S1

5. 状態遷移モデルとテスト

◆状態遷移モデルを活用したテストのまとめ

状態遷移モデルを活用したテストは、例で示したように
状態遷移モデルの動作を仕様書と比較し、期待する動作をすることを
確認するテストになる（バイシミュレーション）。

＜主なメリット＞
- ・上流フェーズで動的に試験できるため、早期に不具合を発見できる
- ・上流フェーズでテストケースを設計でき、下流フェーズに運用できる
- ・モデル化によって差分がわかりやすい
- ・モデルによって可視化されるので、不具合状況が分析しやすい

状態遷移モデルからイベント列を自動抽出し、自動テストする方法を紹介する。

5. 状態遷移モデルとテスト

◆ 状態遷移図の特徴

状態遷移図は、正常系のシステムの振舞を記述したモデルになる

↓

実現したいシステムの基本的な振舞（動作）を表現するため、概要を捉えやすい

◆ 状態遷移図とテスト

状態遷移図から作成するテストケースは、正常系のテストケースとなる

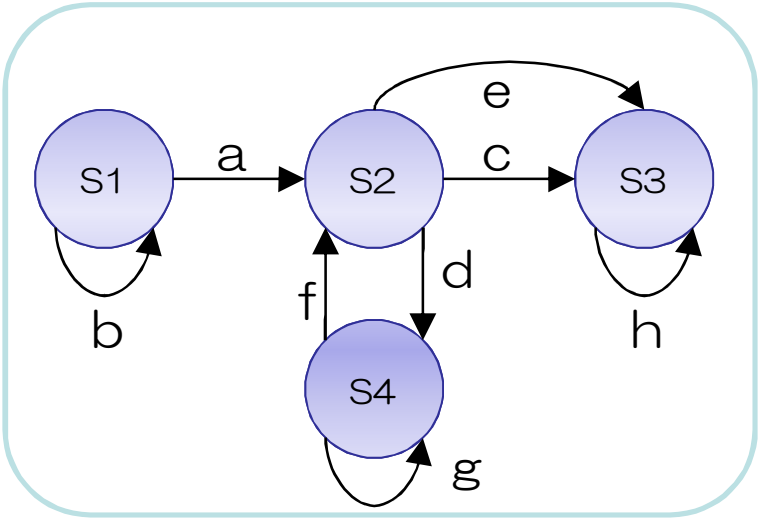
↓

正常系テストを作成しやすいモデル

↕

異常系テストを作成しにくい

状態遷移図の例

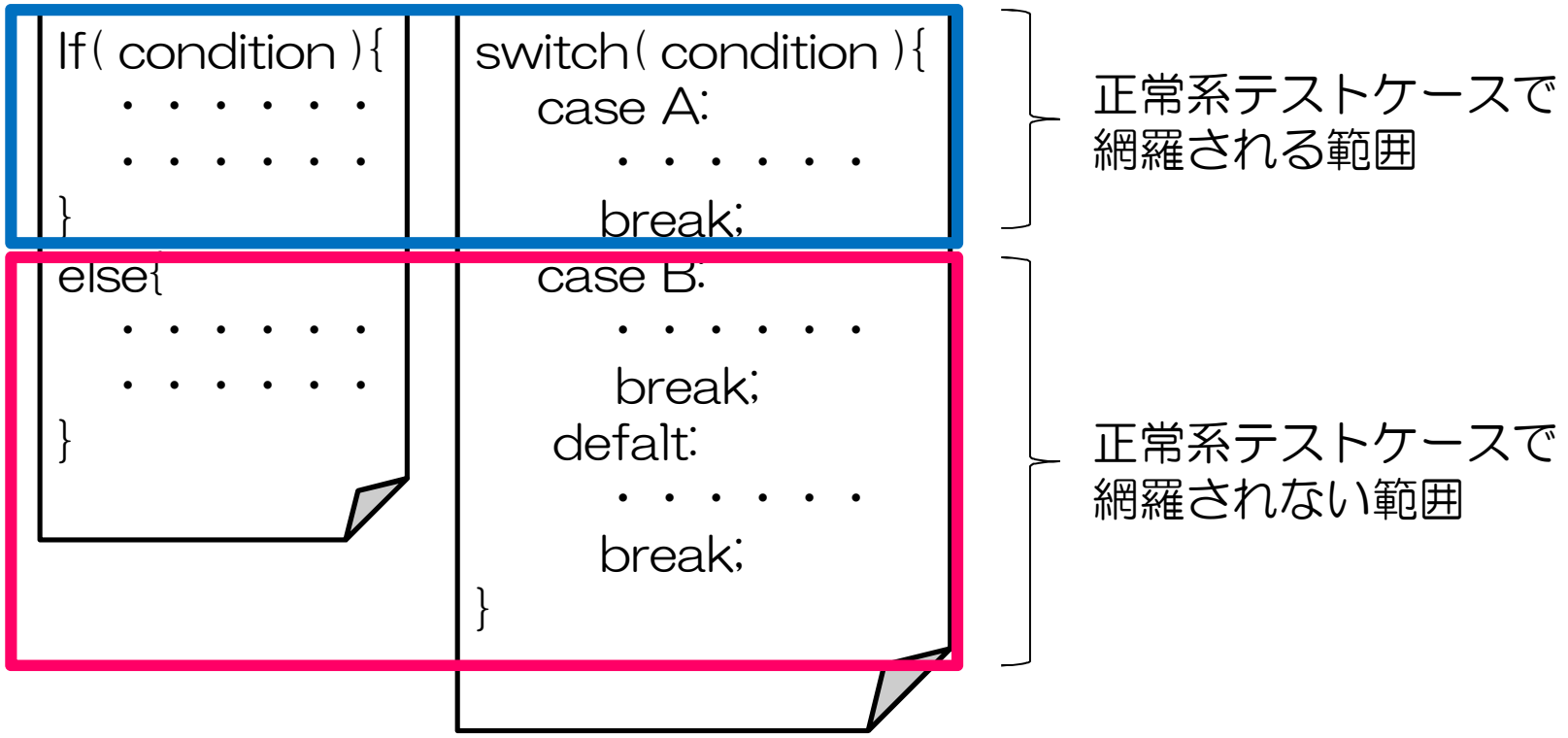


状態遷移図から作成したテストケースの例

事前条件：　S1状態
イベント：　事象aが発生
事後条件：　S2状態であること

5. 状態遷移モデルとテスト

◆ 正常系のテスト網羅度



正常系テストケースは考えやすいが、正常系テストケースだけでは、テスト網羅度が低くなってしまう

5. 状態遷移モデルとテスト

◆ 状態遷移表の特徴

状態遷移表は、システムの振舞を表形式で網羅的に記述したモデルになる

実現したいシステムの事象と状態の組み合せを表現するため、漏れ・抜けがない

◆ 状態遷移表とテスト

状態遷移表から作成するテストケースは、網羅的に表現するため、あり得ない組み合せを異常系のテストを作成することができる。

異常系を考慮したいため、
今回は、状態遷移表を用いる

状態遷移表の例

	S1	S2	S3	S4
a	=> S2	／	／	／
b	=> S1	／	／	／
c	／	=> S3	／	／
d	／	=> S4	／	／
e	／	=> S3	／	／
f	／	／	／	=> S2
g	×	／	／	=> S4
h	×	／	=> S3	／

状態遷移表から作成したテストケースの例

＜正常系＞
事前条件： S1状態
イベント： 事象aが発生
事後条件： S2状態

＜異常系＞
事前条件： S1状態
イベント： 事象gが発生
事後条件： S1状態

5. 状態遷移モデルとテスト

◆ 状態遷移モデルを活用するテストの前提

状態遷移モデルに表現した内容からテストケースを作成するので、状態遷移モデルの誤りはないことを前提とする

➡ 状態遷移モデルの検証がされていること

状態遷移モデルを使用したテストは、弱模倣性（バイシミュレーション）になる（適合テスト）。
そのため、状態遷移モデル自体に誤りがある場合、検出できない。

➡ ハードウェア、通信メッセージの種類が決定していること

状態遷移モデルからテストケースを生成するので、表現されていないことは、考慮されない（異常系のテストが考慮されにくい）。

しかし、プログラムが処理するのは、マイコンのポートに繋がったイベント以上に増えることはないので、モデル作成時に考慮すれば問題ない。

5. 状態遷移モデルとテスト

◆ テストケースの定義

＜正常系テストケース＞
状態遷移モデルに表現した内容は、基本的に正常系テストケースとする。
状態遷移図から作成するテストケース。状態遷移表の無視（何もしない）は、正常系のテストケースとする。

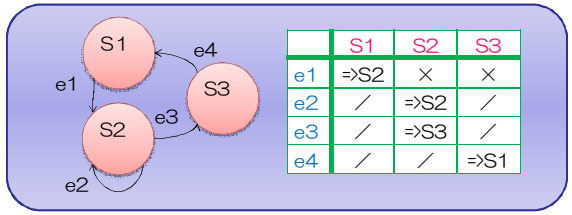
＜異常系テストケース＞
状態遷移モデルに表現した制約を満たさないテストケースとする。
状態遷移表の不可（あり得ない組み合せ）を実行するテストケース、及び優先度を無視する事象を組み合せたテストケース。

6. テストケース生成方法

◆ 状態遷移モデルからテストケースを生成するメリット

① コードからテストケースを生成することに比べてロジックが抽象化されているため、解析しやすく。また、状態爆発しづらい

実装
コード



	S1	S2	S3
e1	=>S2	×	×
e2	／	=>S2	／
e3	／	=>S3	／
e4	／	／	=>S1

「複雑度が高い」かつ「ルールが無い」　「複雑度が高くない」かつ「体系だっている」

② 状態遷移モデルを解析するので、Cコード表現だけでなく、日本語レベルのテストケースを作成できる

	状態1	状態2	状態3
事象１	=>状態2	×	×
事象2	／	=>状態2	／
事象3	／	=>状態3	／
事象4	／	／	=>状態１

	State1	State2	State3
FlgA == ON	=>State2	×	×
Val > 3	／	=>State2	／
Val > 10	／	=>State3	／
FlgB == OFF	／	／	=>State 1

文字列として扱うので複雑さは変わらない

Communication Art Technology Systems

6. テストケース生成方法

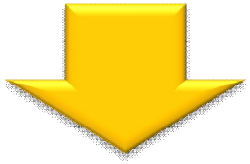
◆ テストケース生成の課題

テストにおいて網羅度は、重要な基準になるが、状態遷移表モデルから
テストケースを作成する場合、開始状態から、終了状態までの全パスを
探索すると膨大なテストケースになってしまう。

例）スイッチの状態遷移表の場合（開始状態：OFF　終了状態：OFF）

	OFF	ON
SW_OFF	／	=>OFF
SW_ON	=> ON	／

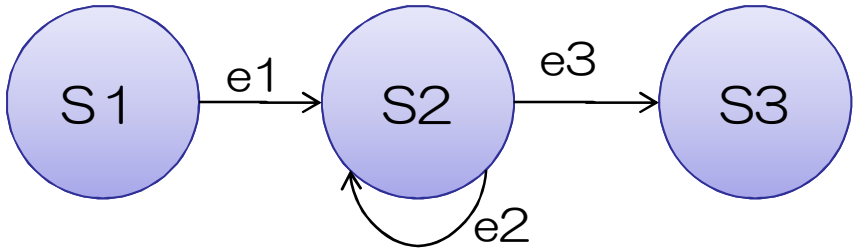
P1： OFF.SW_OFF => OFF
P2： OFF.SW_OFF => OFF.SW_ON => ON.SW_OFF => OFF
P3： OFF.SW_OFF => OFF.SW_ON => ON.SW_ON
=> ON.SW_OFF => OFF
P4： OFF.SW_OFF => OFF.SW_ON => ON.SW_OFF => OFF
P5： OFF.SW_OFF => OFF.SW_ON => ON.SW_ON
=> ON.SW_OFF => OFF



網羅度が高いテストケースは必要だが、全てを実施することは、
困難である。

6. テストケース生成方法

◆ 正常系テストケースだけを考えた場合



状態遷移図の例

	S1	S2	S3
e1	=>S2	×	×
e2	／	=>S2	／
e3	／	=>S3	／

状態遷移表の例

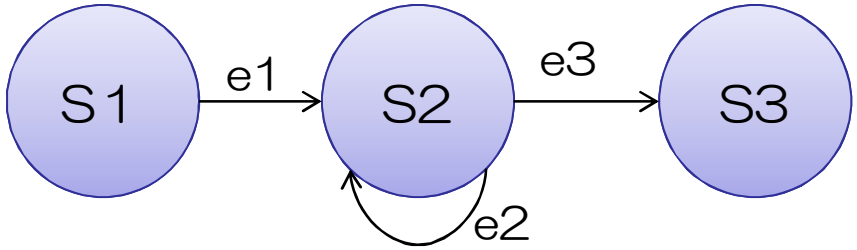
No	説明	テストケース（イベント列）
P1	状態遷移モデルの全パスを 1回網羅するテストケース	e1=>e2=>e3
P2	P1のパスを分解したテストケース1	e1
P3	P1のパスを分解したテストケース2	e1=>e2
P4	P1のパスを分解したテストケース3	e1=>e3
P5	P1に無視イベントを追加したテストケース	e2=>e3=>e1=>e2=>e3=>e2=>e3
P6	P5を分解したテストケース	e2=>e3=>e1
・ ・ ・	・ ・ ・	・ ・ ・

P1だけでは物足りない
（一筆書きアルゴリズム）

ユーザの設定によって、
テストケースを追加できる
仕組みを構築する必要がある

6. テストケース生成方法

◆異常系テストケースを考慮した場合



状態遷移図の例

	S1	S2	S3
e1	=>S2	×	×
e2	／	=>S2	／
e3	／	=>S3	／

状態遷移表の例

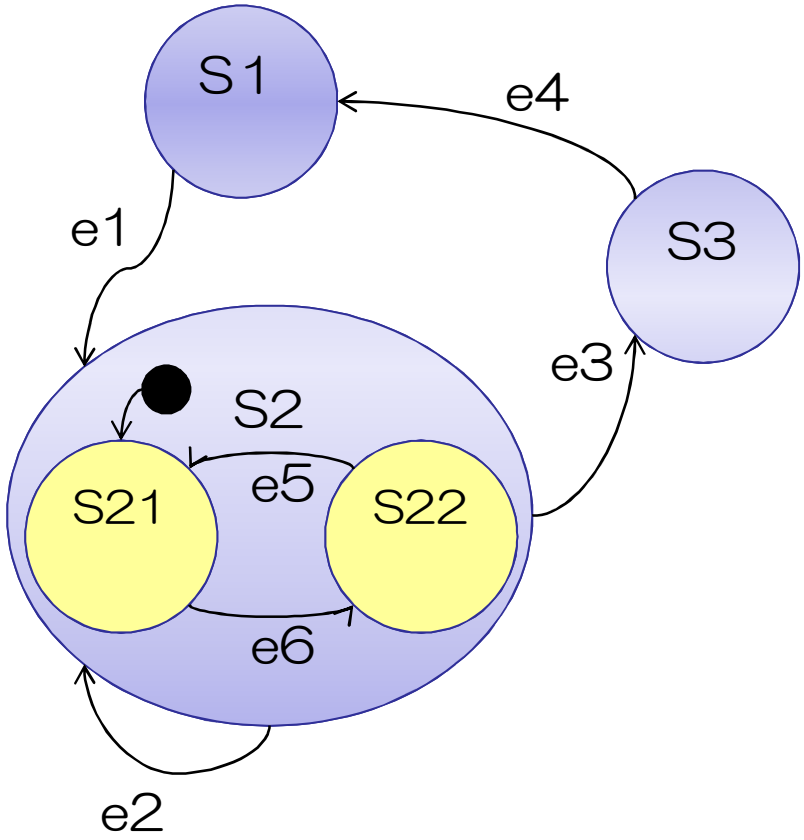
No	説明	テストケース（イベント列）
P1	不可（あり得ない処理）をするテストケース１	e1=>e1=>e2=>e3=>e1
P2	P1を分解したテストケース１	e1=>e1
P3	P1を分解したテストケース２	e1=>e3=>e1
P4	P1の順序を入れ替えたテストケース	e1=>e2=>e1=>e3=>e1
・ ・ ・	・ ・ ・	・ ・ ・

不可（あり得ない）テストケースを考慮でき、
また、イベントには優先度があることが多々あることが、想定できるため、優先度が逆転したテストケースを構築したい

6. テストケース生成方法

◆膨大なテストケースにならないための対策

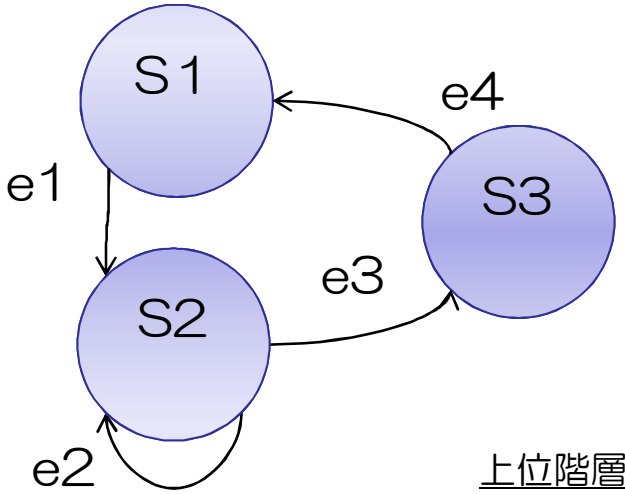
状態遷移モデルが階層構造の場合、全ての範囲を対象とするとテストケースが膨大になる。そのため、対象範囲を選択できるようにする



	S1	S2		S3
		S21	S22	
e1	=>S2	×	×	×
e2	／	=>S21	=>S22	／
e3	／	=>S3	=>S3	／
e4	／	／	／	=>S1
e5	／	／	=>S21	／
e6	／	=>S22	／	／

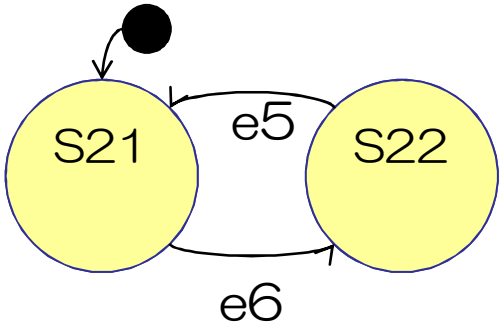
6. テストケース生成方法

◆ 状態階層毎のテストケース生成



	S1	S2	S3
e1	=>S2	×	×
e2	／	=>S2	／
e3	／	=>S3	／
e4	／	／	=>S1

上位階層の状態遷移モデル



	S21	S22
e5	／	=>S21
e6	=>S22	／

下位階層の状態遷移モデル

6. テストケース生成方法

◆ 状態遷移モデルの表現方法

・状態遷移表で表現する
⇒状態遷移図表現は課題
・遷移に遷移条件を設定できるようにする

	S1	S2	S3
e1	=>S2	×	×
e2	／	=>S2	／
e3	／	=>S3:(e2>1)	／
e4	／	／	=>S1

遷移条件

◆ テストケース作成方法

(1) 全網羅パス生成
(2) 一筆書きテスト生成
　(2-1) 正常系列を作成する（基本パターン）
　　e1=>e2=>e3=>e4
　(2-2) 正常系列に無視イベントを追加する（無視追加オプション）
　(2-3) 正常系列の部分列を作成する(部分列作成オプション)
　　e1, e1=>e2, e1=>e2=>e3
　(2-3) 部分列に不可イベントを加える（不可追加オプション）
　(2-4) 部分列に遷移条件を違反するイベントを追加する
　　（条件違反追加オプション）

6. テストケース生成方法

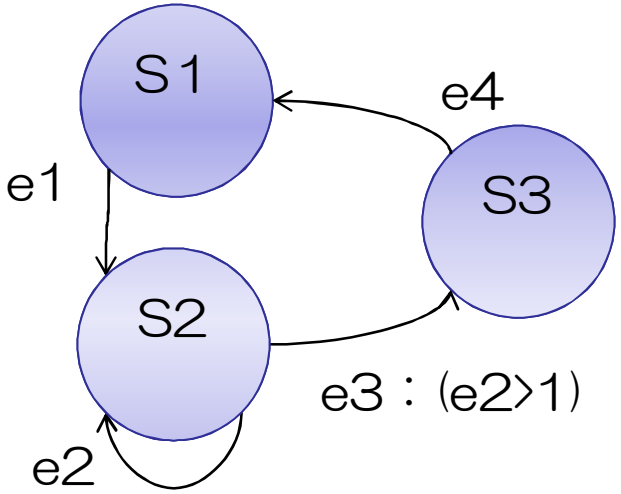
◆テストケース生成方法の方針

状態遷移図から作成できるテストケース（正常系）をベースに、オプションで、テストケースを追加する仕組みにする

テストケースが使う側が納得いくものになる
（自動生成したときのテスト網羅度の基準がわかりやすい）

◆ステップ1（正常系テストケース作成）

状態遷移図を一筆書きで全網羅するようなテストケースを生成する



開始状態：S1、終了状態：S1の場合

P： e1 => e2 => e3 => e4

※このとき、遷移条件を考慮する

6. テストケース生成方法

◆ ステップ２（無視追加オプション）

正常系テストケースに、状態遷移表の無視イベントを追加する

	S1	S2	S3
e1	=>S2	×	×
e2	／	=>S2	／
e3	／	=>S3：(e2>1)	／
e4	／	／	=>S1

正常系P： e1 => e2 => e3 => e4

無視追加P： e2 => e3 => e4=> e1 => e4 => e2
=> e3 => e2 => e3=> e4

◆ ステップ３（部分列分解オプション）

正常系テストケースを、部分列に分解する

正常系P： e1 => e2 => e3 => e4

部分列P1： e1
部分列P2： e1 => e2
部分列P3： e1 => e2 => e3

6. テストケース生成方法

◆ ステップ４（不可追加オプション）

部分列に分解したテストケースに不可イベントを追加する

部分列P1： e1
部分列P2： e1 => e2
部分列P3： e1 => e2 => e3

不可追加P1： e1 => e1
（部分列P1に不可イベントを追加）
不可追加P2： e1 => e2 => e1
（部分列P2に不可イベントを追加）
不可追加P3： e1 => e2=> e3 => e1
（部分列P3に不可イベントを追加）

	S1	S2	S3
e1	=>S2	×	×
e2	／	=>S2	／
e3	／	=>S3：(e2>1)	／
e4	／	／	=>S1

◆ ステップ５（条件違反追加オプション）

部分列に分解したテストケースに条件違反イベントを追加する

条件違反追加P： e1 => e3　（部分列P1に追加）

6. テストケース生成方法

◆ ステッ

部分列に

部分列P1
部分列P2
部分列P3

不可追加
（部分列
不可追加
（部分列
不可追加
（部分列

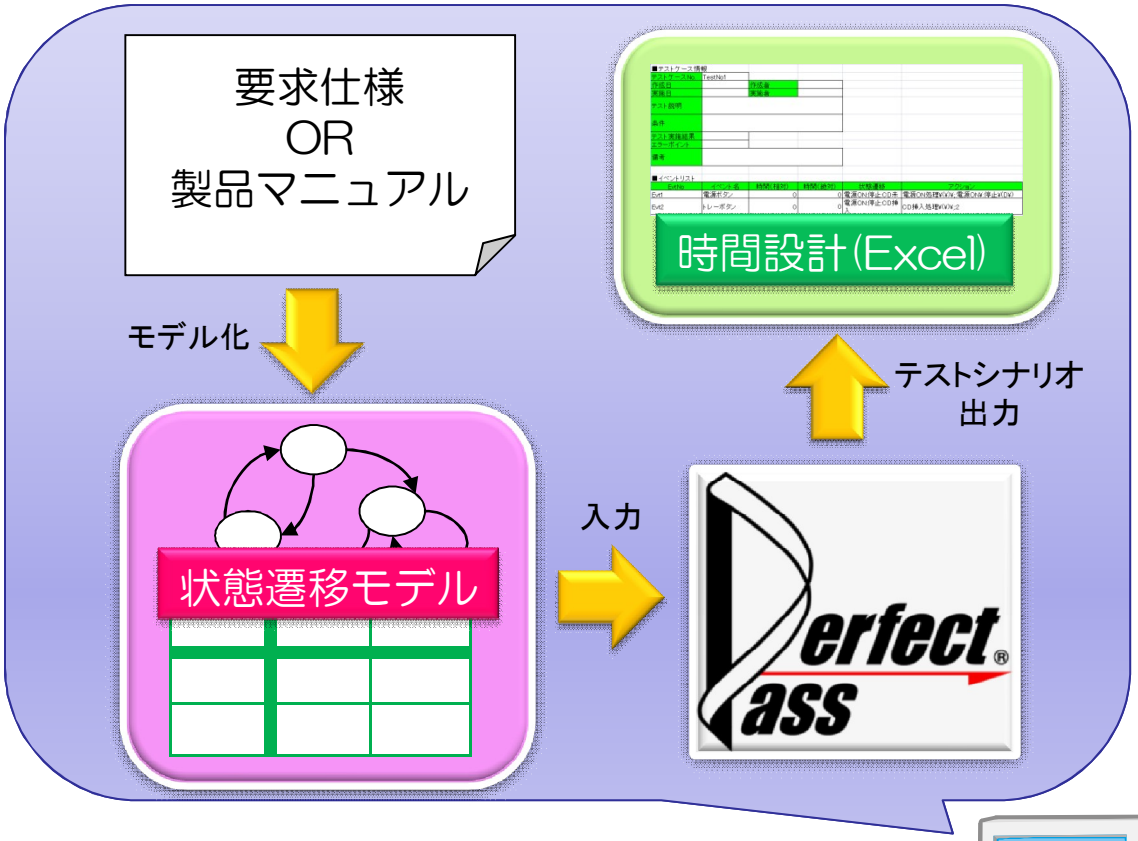
◆ ステッ

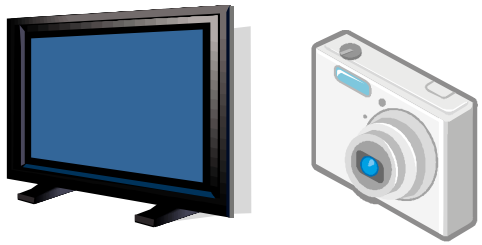
部分列に

S2	S3
×	×
=>S2	／
S3：(e2>1)	／
／	=>S1

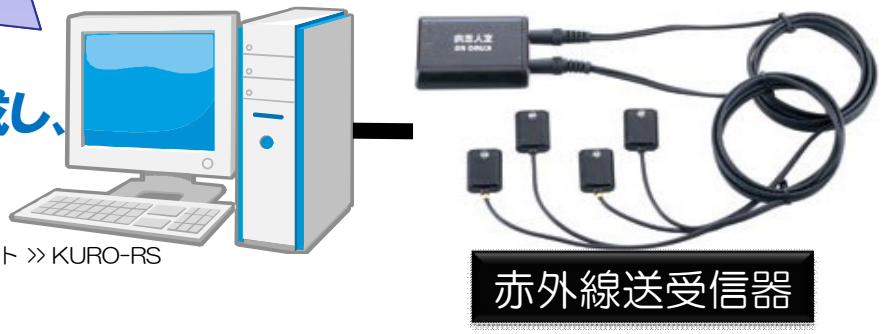
PerfectPassで紹介したテストケース生成の自動化を検討中

7. 適用案の提案





赤外線イベント



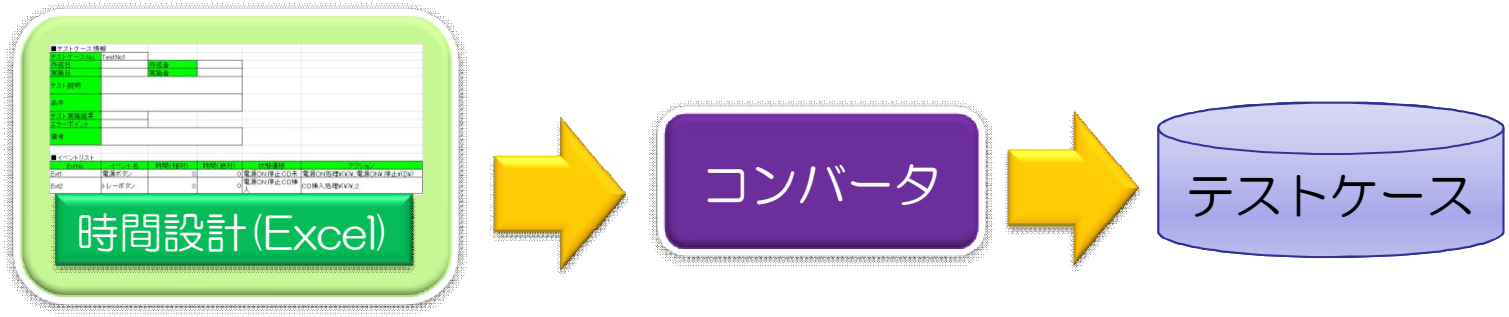
状態遷移モデルからテストシナリオを自動生成し、赤外線通信で自動テストを実施する

画像引用元：　玄人志向 一商品一覧 ≪ キワモノシリーズ ≫ マルチメディア ≫ 学習リモコンキット ≫ KURO-RS
http://www.kuroutoshikou.com/modules/display/?iid=928

Communication Art Technology Systems

7. 適用案の提案

- システムテスト以外への適用方法（今後の拡張案）
  - テストケースをコンバート
    他のテストの入力にコンバートする
    ⇒　単体テスト、結合テストにも対応可能
    ⇒　上流フェーズのテストにも対応可能



  - タイミングチャート作成
    テストケースとタイミング設計より、タイミングチャートを
    自動生成する。

- 管理面の強化案
  - TestLinkとの連携
    テストケースをTestLinkで管理できるようにする

7. 適用案の提案

◆ テストケース自動生成の適用で効果がありそうな製品
状態遷移モデルで表現できる製品すべてに適用できる！

＜具体的な運用方法＞

・要求レベルの状態遷移モデルからテストケースを生成する

例１）

事象名称のイベント列を抽出したテストケース

事象Ａ⇒事象Ｂ．．．⇒事象Ｃ

⇒　紹介した赤外線テスト

例２）

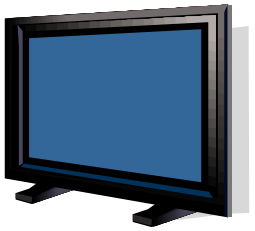
詳細レベルの状態遷移モデルからテストケース生成

u1_value = 1⇒ u1_value = 2

⇒　状態遷移モデルの文字列を詳細化すれば、その文字列を抽出できる

7. 適用案の提案

◆ 赤外線自動テストツールの適用で高い効果が得られそうな製品
前提：　赤外線を利用している製品
（テレビ、エアコンなど）



＜高い効果が得られそうな部分＞
・入力のタイミングによって挙動が異なる部分
例１）
液晶パネルのRGB調整などの状態がタイムアウトしてしまう機能
⇒　時間で状態が異なり、同じイベントでも挙動が異なるため。

・入力を連続するもの
例２）
エアコンの温度調節機能
⇒　温度調節機能は、連続して入力される項目のため。
・赤外線通信の混線テスト
例３）
複数の製品がある場合
⇒　赤外線は、無線通信なので混線が起こるため

Communication
Art
Technology
Systems

8. 期待される効果

＜品質改善＞

- ◆テストケース作成の自動化により、人為的なミスを削減できる
- ◆自動テストにより、人為的なミスを削減できる
- ◆状態遷移モデルの活用により、上流工程の漏れ・抜けを発見できる
- ◆状態遷移モデルのカバレッジにより、テスト基準を明確にできる

＜工数削減＞

- ◆テストケース作成の自動化により、テスト工数が削減する
- ◆自動テストにより、テスト工数が削減する
- ◆状態遷移モデルの活用により、上流工程で漏れ・抜けを発見できるため、工数の手戻りが少ない
- ◆テストケースの再利用により、効率化でき、工数が削減する

9. 今後の課題

＜テストケース生成＞

- ◆正常系テストケース自動生成後の微調整
- ◆テストケース自動生成のバッチ処理
- ◆詳細なテストケース生成
- ◆対象モデルの状態遷移図表現
- ◆タイミングの自動生成

＜自動テストツール＞

- ◆自動テストの結果確認方法までのサポート方法
- ◆テストケースのタイミングチャート表現

Communication
Art
Technology
Systems

10. まとめ

＜テストケース生成＞

- ◆状態遷移モデルからテストケースを生成することは、比較的容易である
- ◆テストの全自動化は困難だが、半自動化するだけで効率が上がる
- ◆自動化の際のテストケース生成方法を、正常系テストケースに条件を付加する方法にすることで、テスト基準が解りやすい

＜自動テストツール＞

- ◆赤外線通信を使用することで、システムテストを容易に実施できる

格言

- “If it can happen, it will happen.”
  「起こる可能性のあることは、いつか実際に起こる。」
  （マーフィーの法則より）

- “Anything that can go wrong will go wrong – at the worst possible moment.”
  「うまく行かなくなりうるものは何でも、うまく行かなくなる－しかも最悪のタイミングで」
  （マーフィーの法則より）

- 情報システムの開発では、計画時の2倍の費用と2倍の時間がかかり、1/2の機能しか実現しない
  「222の法則」（作者不明）

引用元：
マーフィーの法則　－Wikipedia－、
http://ja.wikipedia.org/wiki/%E3%83%9E%E3%83%BC%E3%83%95%E3%82%A3%E3%83%BC%E3%81%AE%E6%B3%95%E5%89%87
マーフィーの法則がIT版になって帰ってきた！　－@IT情報マネジメント－
http://www.atmarkit.co.jp/im/cits/serial/murphy/01/01.html

Communication
Art
Technology
Systems

ご清聴ありがとうございました。

Communication
Art
Technology
Systems

ZIPC®
CASE Tool for Embedded Systems

JaSST' 10 Tokyoの展示スペースで製品／デモを展示しております。
お気軽に足を運んで下さい