Model Checkingを適用した実践的非同期制御検証

Go with the Early Bird

富士ゼロックス株式会社
オフィスプロダクト事業本部
コントローラソフトウェア開発部

村石 理恵/服部　彰宏/野村 秀樹/山本 訓稔

2007 年 1月30日

1. こんな経験ありませんか？

不具合を見つけた。
2度と再現できない。

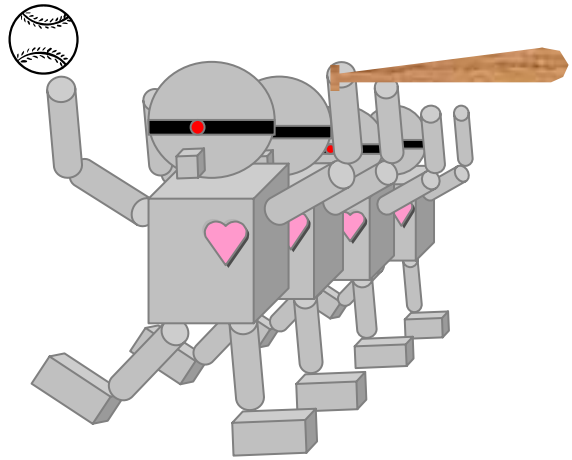
テスターさんに
「再現するまで続けてくれ！」
と協力を(泣いて)頼んだ。

市場でトラブルが起きる。特定のお客様である。
再現させようにも情報が少ない。
情報も取らせてくれない。
限られた情報から原因を推理してゆくしかない。

2. 非同期系の問題ってどんなもの？

THE DOCUMENT COMPANY FUJI XEROX

自律型　人型ロボットの野球を考える

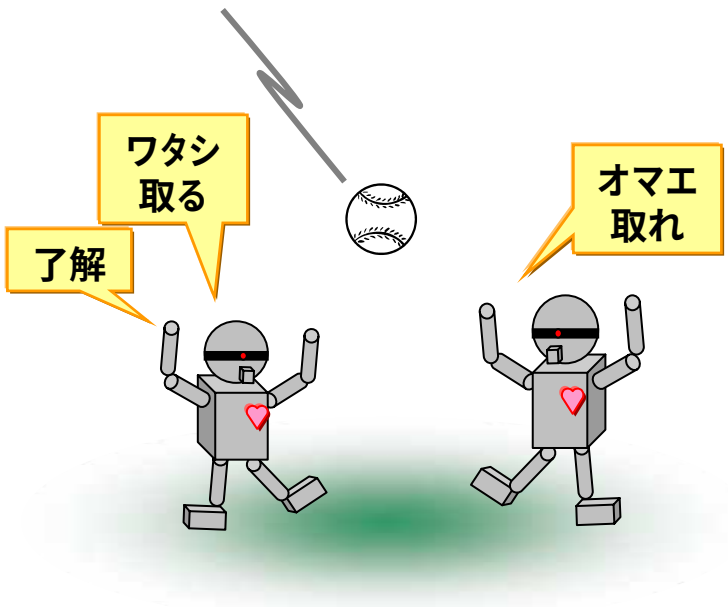


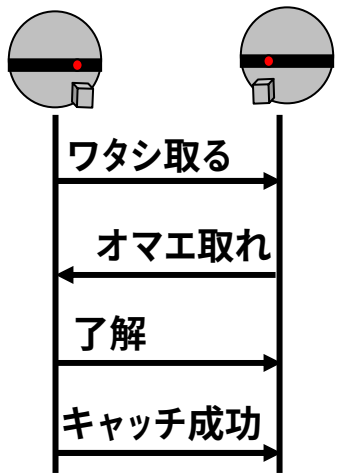
要求

ロボットプレイヤーは、**自律的**、且、
他のプレイヤーと**協調**しながらプレイすること
Mono Eye　がいいな

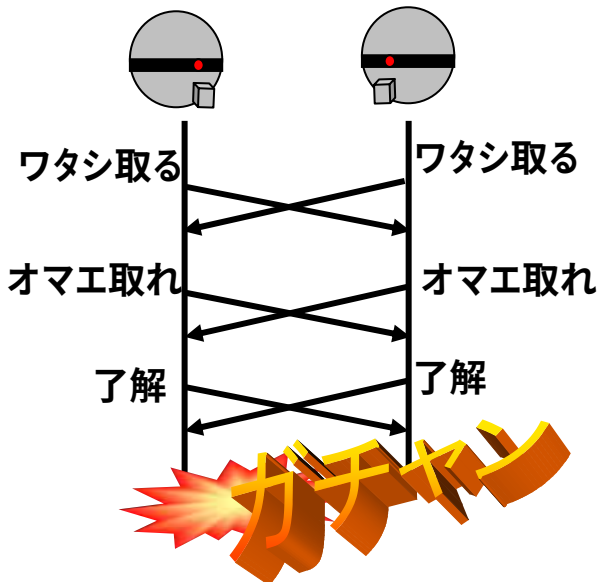
仕様

野球ルールを遵守（投げる/打つ/守備妨害/・・・）
ルールブックには書かれていない案件
　・守備の時に中間地点に落ちたら、
　　声をかけ合って先に声をかけたほうがとる、、





メッセージ　クロス
という代表的設計ミス



3. プリンタ複合機開発の現状

THE DOCUMENT COMPANY
FUJI XEROX

機能追加、開発規模の増大、構成の複雑化、開発期間の短縮……

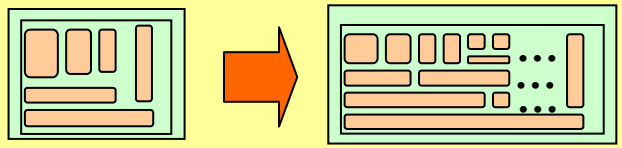
⇒ それでも、市場導入後に不具合を出してはいけない

(不具合を出すなということに**無理がある**気がするのだが…)

規模は増大

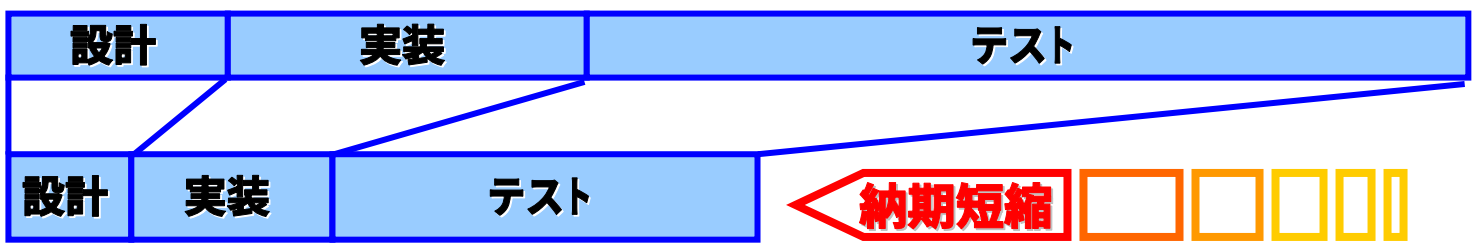
300万行 ⇒ 600万行 ⇒ 1000万行

構成は複雑化



タスク数は増加…

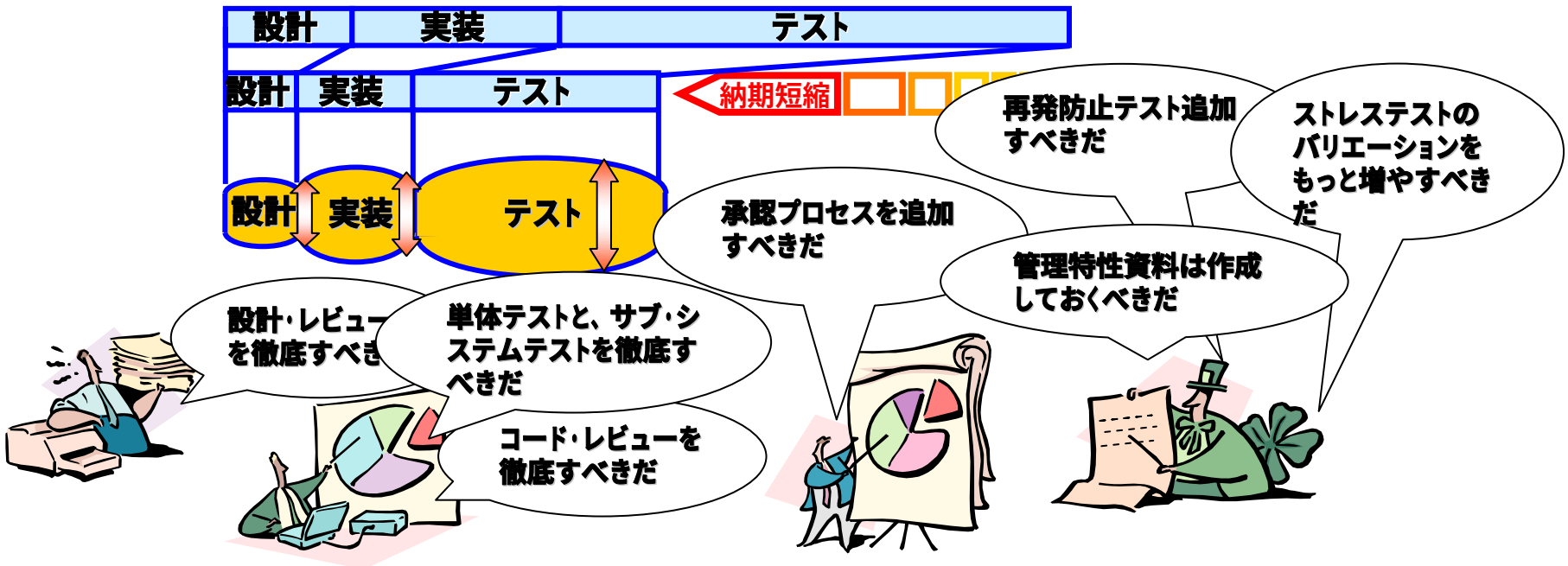
プリント、FAX、スキャナ、UI、後処理装置、セキュリティー ・・・



4. 何か良い方法はないの？

THE DOCUMENT COMPANY
FUJI XEROX

- 不具合が発見される度に「×××の追加」という施策は、効果はあるが．．．
  既に、これまでに工数は膨らみすぎている。



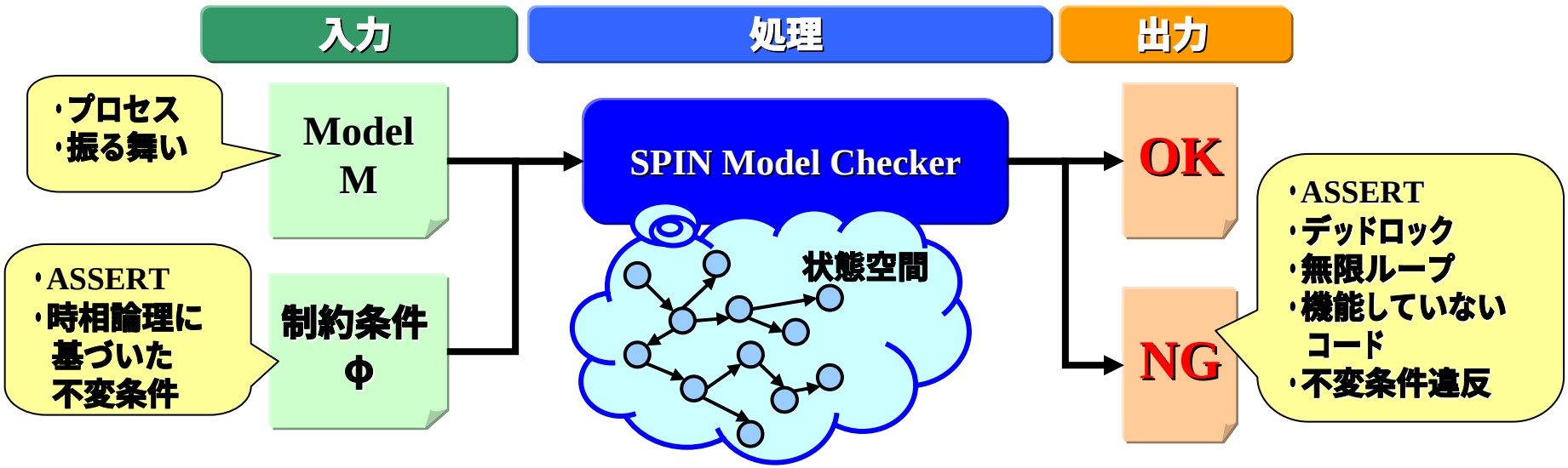
- デッドロック、通信交錯、制御設計ミス、などの検出は現状の施策では限界

非同期制御の振る舞いの全てを、“きっちり”検出し、且つ、工数をこれ以上増大させない手法はないのか？

5. モデルチェッキングという技術

THE DOCUMENT COMPANY FUJI XEROX

システムをモデル化し、その状態を全てシミュレーションし、仕様の正しさを抜け漏れなく検証する技術



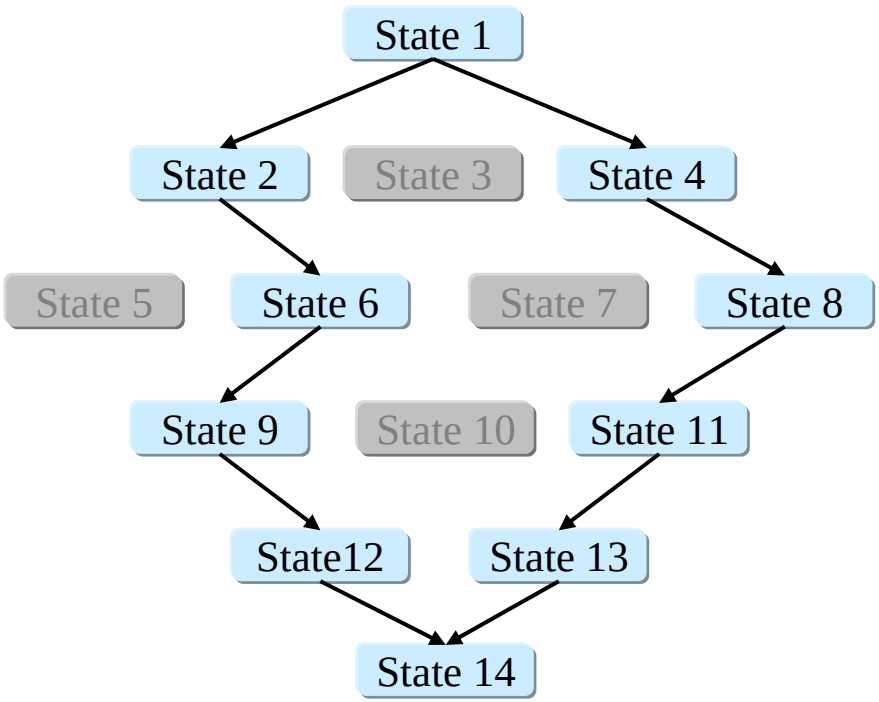
- SPIN (=Simple Promela Interpreter)　開発者:G.J.Holzmann
  - Promelaで書かれたモデルの検証を行うツール
  - 検証対象のシステムの状態空間を制約条件に基づいて徹底的に検証する
  - NGの箇所で停止し、NGまでの到達経路をLogとして保存する
- Promela(=Protocol / Process Meta Language)
  - –モデル記述言語
  - –非同期通信に関する検証が可能
  - –ネットワーク・プロトコル、電話交換機システムなどに適用

6. モデルチェッキングの役割

THE DOCUMENT COMPANY
FUJI XEROX

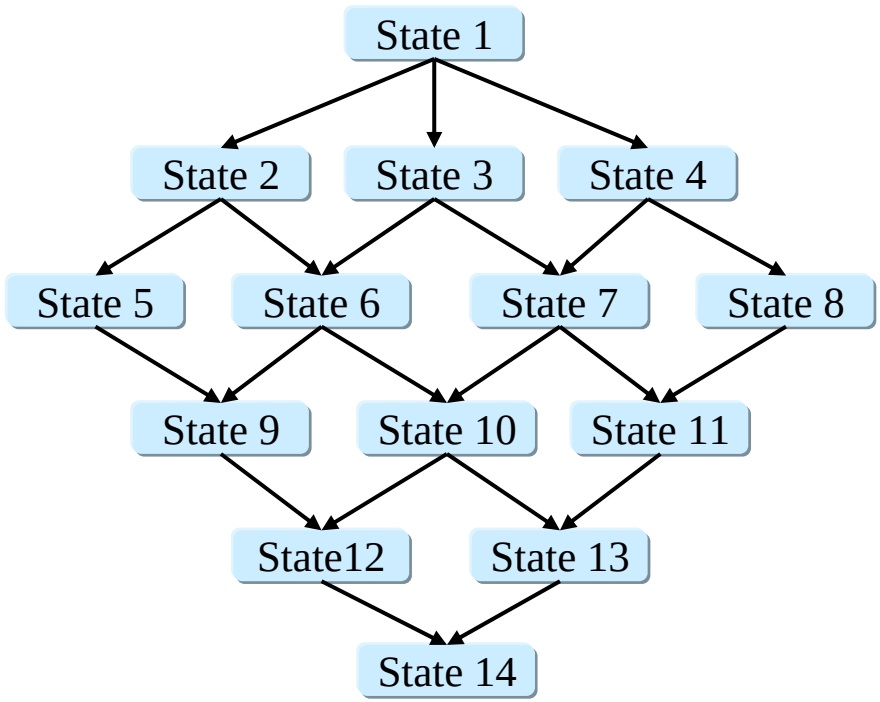
従来の検証方法との比較

従来のテスト
状態の検証に抜けが生じる



テスト項目の抽出がカギ
思いつかないパスはテストできない

モデルチェッキング
網羅的に検証を実施する

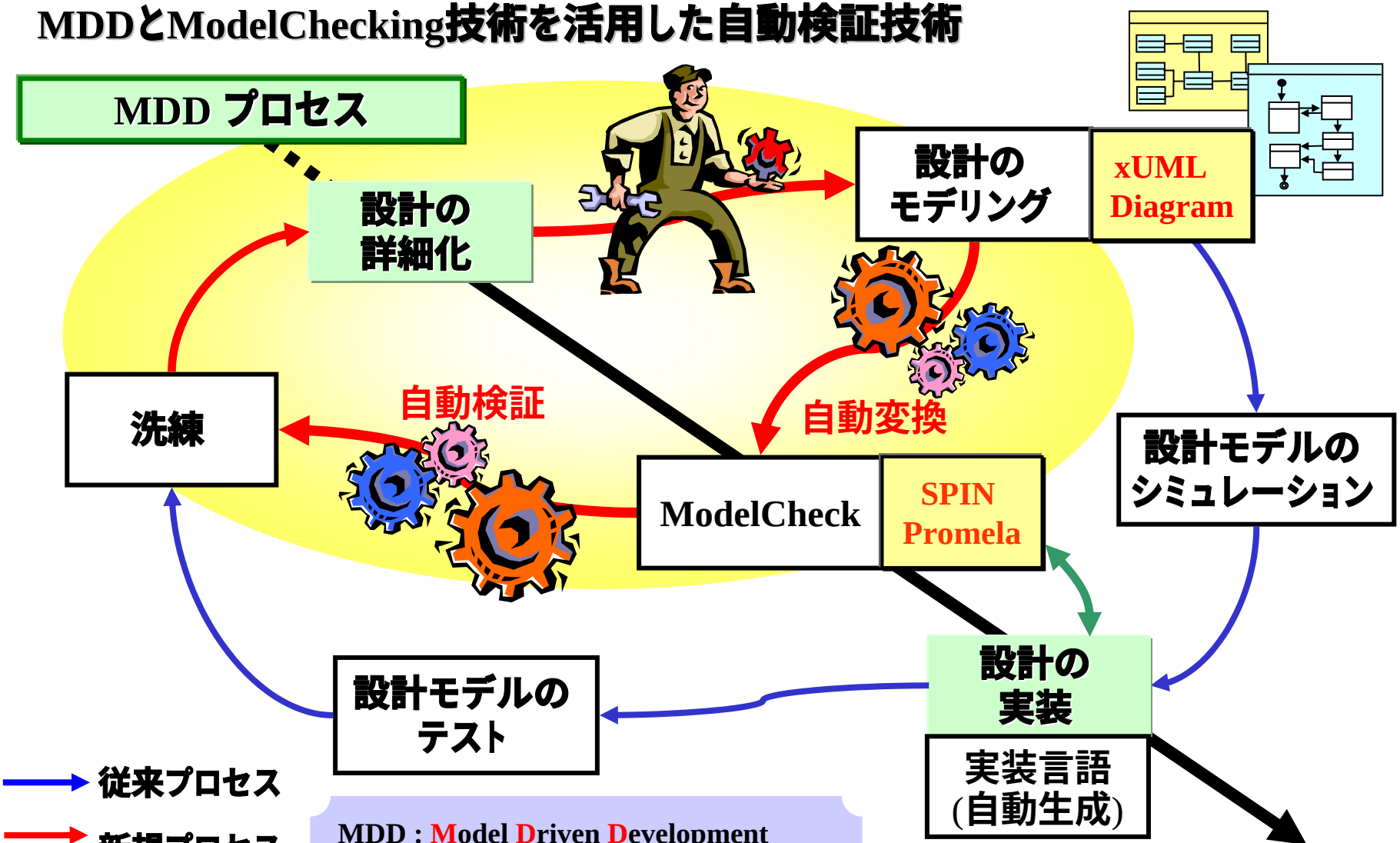


フルパス実行をツールがサポート
自動的に全てのパスを網羅できる

7. 実現したプロセス　Early Bird Process

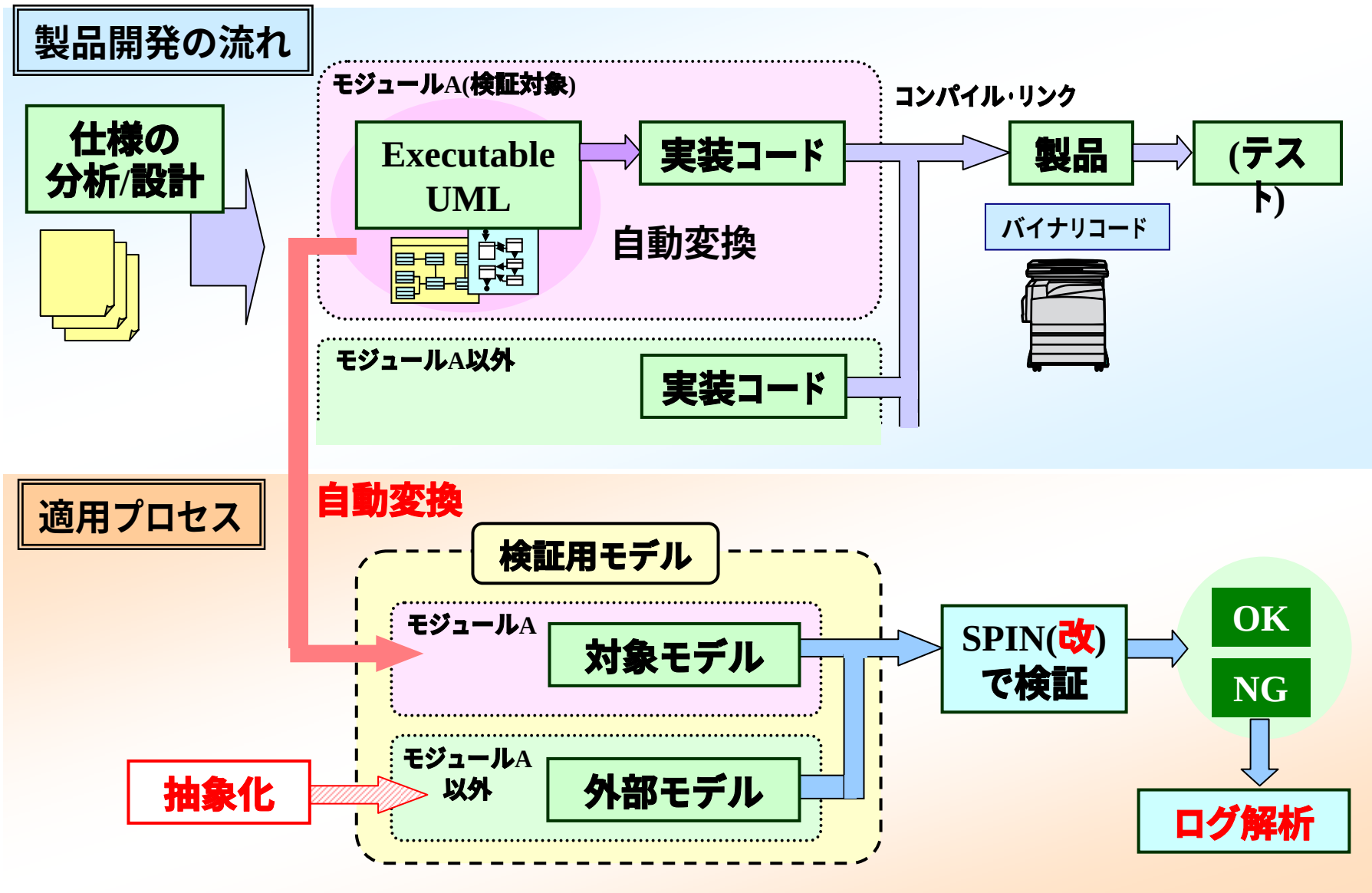
THE DOCUMENT COMPANY
FUJI XEROX

MDDとModelChecking技術を活用した自動検証技術



8. 適用プロセスの概要

THE DOCUMENT COMPANY
FUJI XEROX



9. どんな不具合がどう抽出できるようになったのか

THE DOCUMENT COMPANY FUJI XEROX

どんな不具合

品質

◆ 当初の狙い

下記不具合の検出
- デッドロック
- Assert
- メッセージクロス
- インスタンス寿命の検証

◆ 結果

高ストレス下で発生する可能性がある不具合を2件発見。

検出した不具合の例	
原因	予期しないタイミングのイベント受信
症状	ハングアップ
発生頻度	低 高ストレス下で発生する可能性がある
難易度	実機デバッグ、設計レビューでは発見が困難

どう抽出

工数

◆ 高価な設備は不要

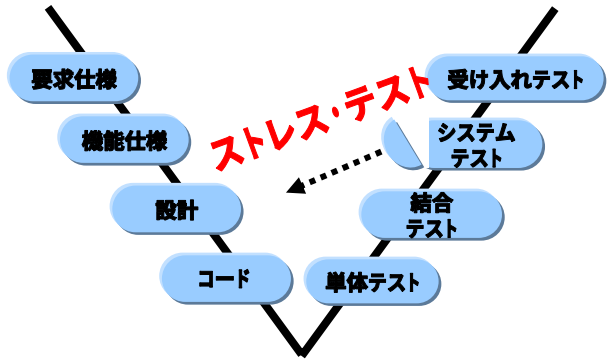
SPINの環境	
PC	PentiumM 1.5GHz MEM: 500MB
OS	Windows XP
Cygwin	SPIN動作環境
SPIN	モデルチェッカー

◆ 高い検証効率

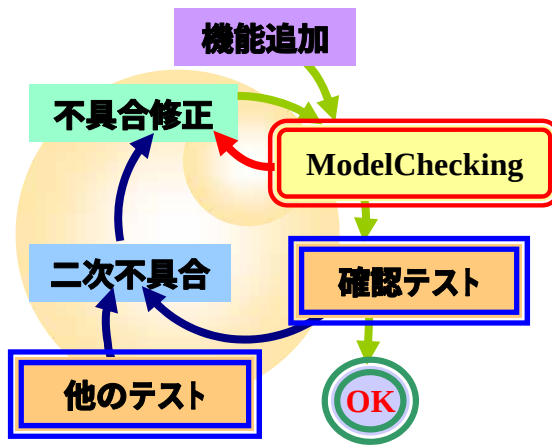
モデル検証時間	
検証モデル生成時間	数分
検証時間	1分程度
検証した状態数	数万個
デバッグ時間	1人日/バグ

プロセス上の意味

◆ ストレステストの前倒し



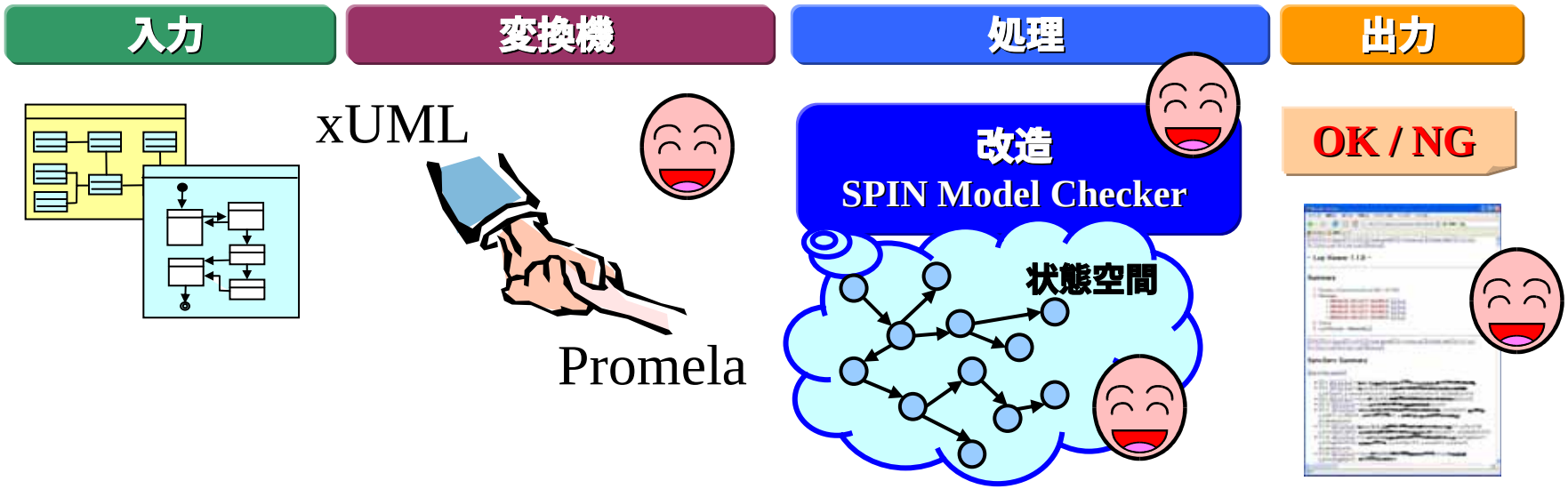
◆ 完全回帰テストの実現



10. 運用上の課題

THE DOCUMENT COMPANY FUJI XEROX

道具はできた！



運用するには、まだ課題が、、、

- 検証モデルの大規模化
- 複数個の同値不具合が検出される
- 実機テストとモデルチェックとの住み分け

どんな手法やツールも万能ではない!!

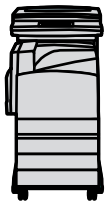
道具は使い方を考えなければ、逆に怪我をすることも、、、

11. 検証モデルの拡大に伴う問題と施策 - 1

THE DOCUMENT COMPANY FUJI XEROX

外部デバイス連携機能への適用

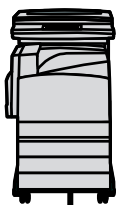
正常系の検証モデルの構築プロセス



設計 → 実装 → 仮検証 (2分) → 検証完了

修正×n 回

外部サービスAと必要な異常系を追加して検証モデルを構築



外部サービスA

設計 → 実装 → 仮検証 (60分) → 検証完了

修正×n 回

必須プロセス

外部モデルは、何度も仮検証を実施して、信頼性を向上させなければならない のじゃ

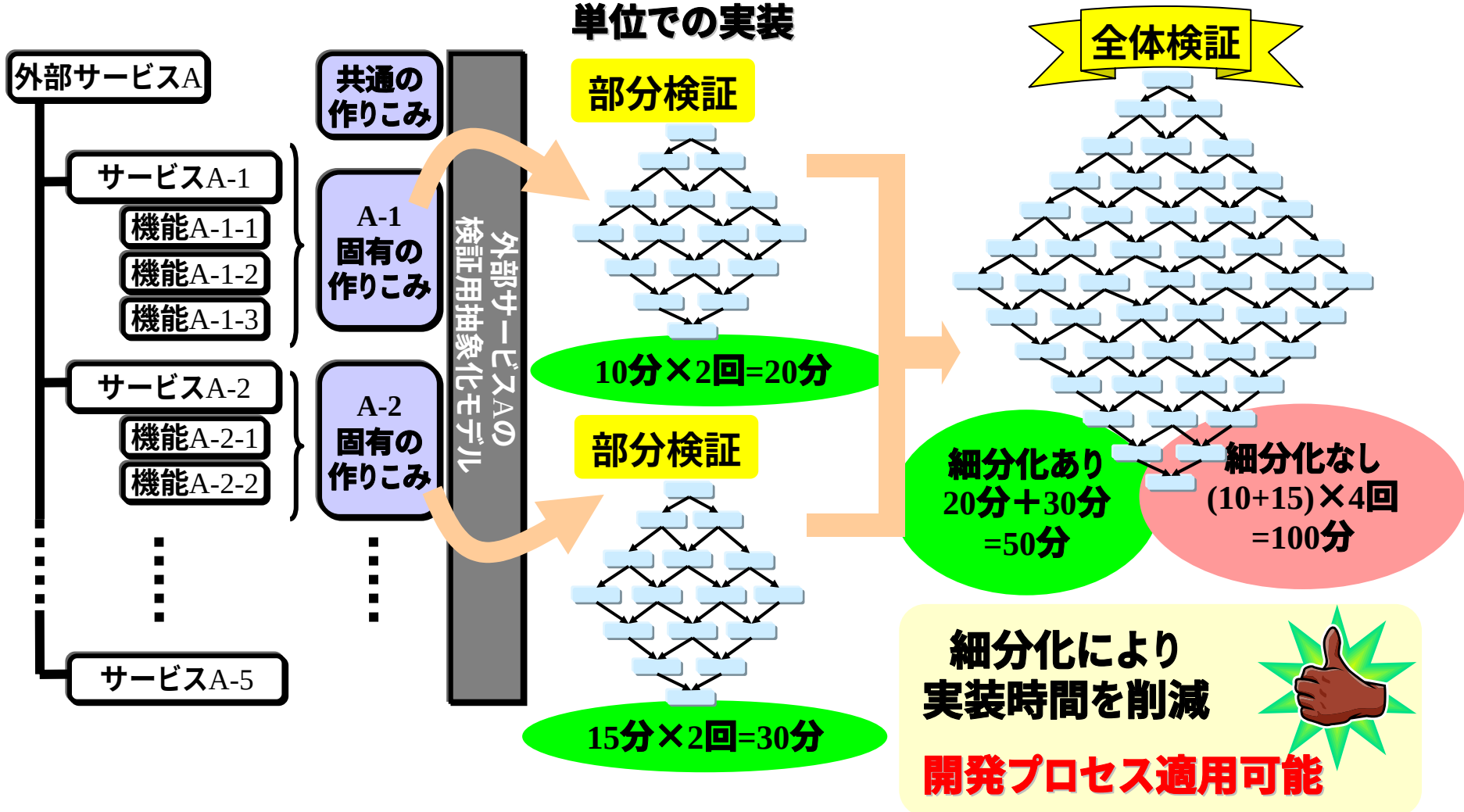
（設計 + 実装 + 仮検証 [60分] ）× n回 ⇒ 検証用モデル完成

そもそも、最初は何時間後に仮検証が終わるのか不明 ！
これでは、やってられない！！

12. 検証モデルの拡大に伴う問題と施策 - 2

THE DOCUMENT COMPANY FUJI XEROX

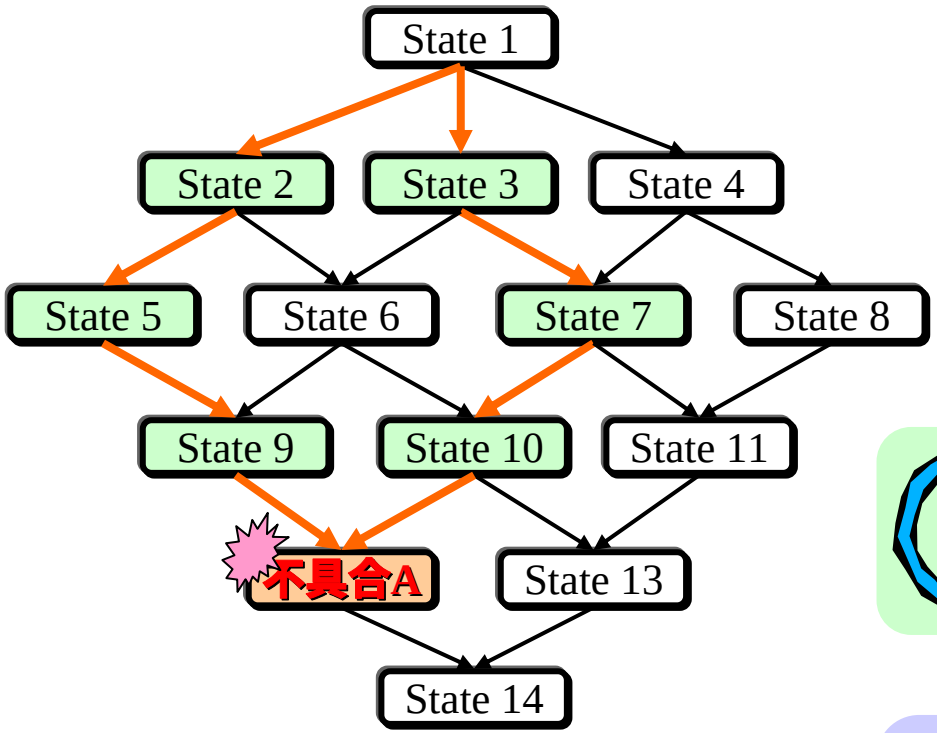
細分化⇒統合による検証用モデルの作りこみプロセス



13. 同値不具合の自動選別

THE DOCUMENT COMPANY FUJI XEROX

同値不具合はまとめないと、使い物にならない



不具合Aに到達する経路

1: State1⇒State2⇒Sate5⇒State9 ⇒不具合A
2: State1⇒State2⇒Sate6⇒State9 ⇒不具合A
3: State1⇒State2⇒Sate6⇒State10⇒不具合A
4: State1⇒State3⇒Sate6⇒State9 ⇒不具合A
5: State1⇒State3⇒Sate6⇒State10⇒不具合A
6: State1⇒State3⇒Sate7⇒State10⇒不具合A
7: State1⇒State4⇒Sate7⇒State10⇒不具合A

報告される不具合は同値を含む
本当の不具合はいったいどれだけ??

同値不具合を自動選別し、
報告しないようなフィルタを作成

不具合規模の推定が可能

同値不具合:
到達経路は異なるが本質的に同じ不具合

SPINは、オプションの設定によって、
一回の検証で複数個レポートしてくれる

14. 実機テストとモデルチェッキングの住み分け - 1

THE DOCUMENT COMPANY FUJI XEROX

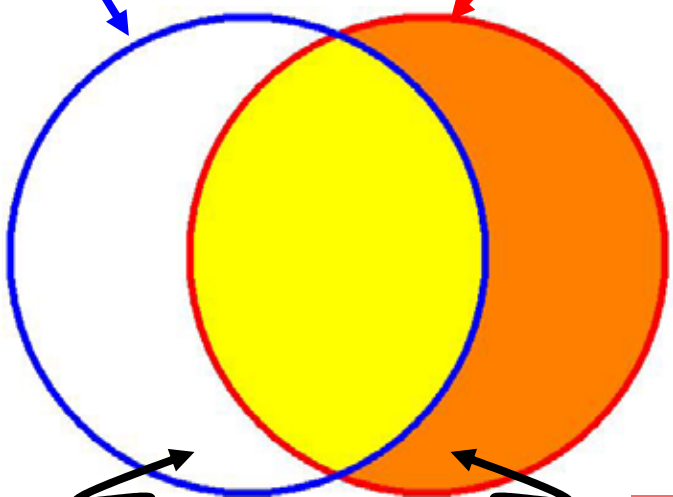
何ができる？
何ができない？

実機テストで発見できる不具合

- 人間が現実的にオペレーションできるもの
- 人間が精神的に耐えられるもの

モデルチェッキングで発見できる不具合

- デッドロック
- インスタンス残り
- アサートで検出できるもの
- タイミングに関連するもの



実機テスト向き

MCの不得意分野
- 機能組合せテスト
- 統合テスト

モデルチェッキング向き

MCの得意分野
- タイミング徹底テスト
- 単体テスト

モデルチェッキングでは論理上無理 / 逆に不向き

- 外部モデルの作成が大変
- 不変条件の記述が大変
- パラメータに関連するもの

実機テストでは事実上無理

- 数十マイクロ秒の通信の隙間を狙うテスト
- 工数的/コスト的な限界
- テスト環境が作れないもの
- テスターが耐えられないもの

15. 実機テストとモデルチェッキングの住み分け -2

THE DOCUMENT COMPANY FUJI XEROX

実機テストの何と置き換えられるの？

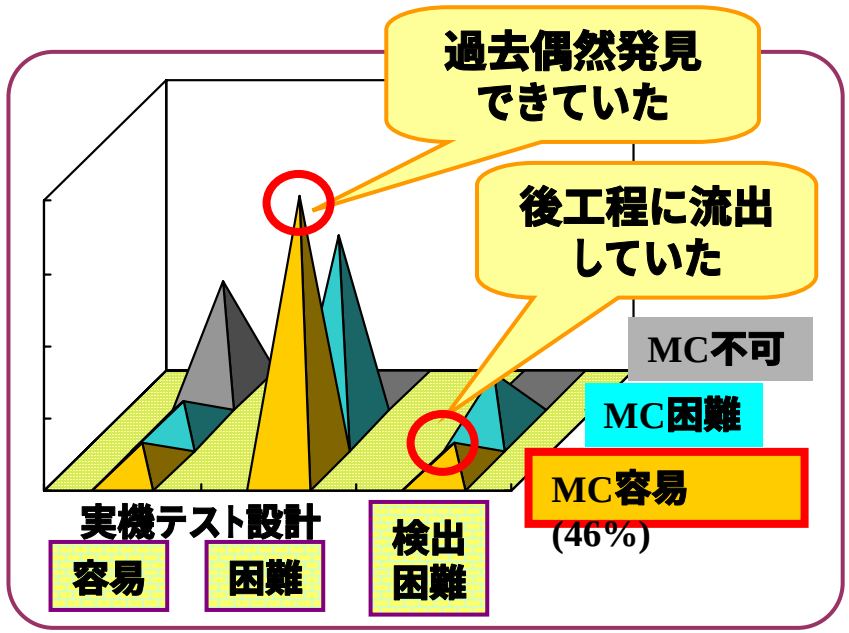
・モジュールテストの段階から使用可能
・タイミング・I/F検証が得意(高負荷環境を含む)
・現実的な検証対象の規模には限界がある

不具合種類とテスト・フェーズ

	不具合の種類		UT	モジュールテスト	Model Checking	結合テスト	システムテスト
1	UI不具合	機能性	×	×	×	●	●
2		コミュニケーション	×	×	×	●	●
3		機能体系	×	×	×	×	●
4		機能不足	●	●	×	●	●
5		パフォーマンス	×	×	×	●	●
6		アウトプット	×	×	×	●	●
7	エラー処理のバグ		●	●	▲	●	●
8	境界値関連のエラー		●	●	▲	●	●
9	初期状態・起動後の状態		×	●	×	●	●
10	制御フローエラー		×	▲	●	●	●
11	データ処理・解釈のエラー		×	×	×	●	●
12	競合状態		×	×	●	▲	●
13	負荷状態		×	×	●	×	●
14	ハードウェア		×	×	×	×	●
15	計算エラー		●	●	●	●	●

期待できること

外部サービスAの過去不具合調査の結果から



モデルチェッキングが向いている領域において

“過去偶然発見できていた” ⇒ 完全に検出?

“後工程に流出していた” ⇒ 0件?

Total 50%程度の不具合が検出可能見込み
(現在 モジュールテスト適用に向けて活動中)

16. まとめ

THE DOCUMENT COMPANY
FUJI XEROX

開発プロセス適用の準備段階として以下を実施

◆ 細分化⇒統合による外部モデル作りこみ

開発プロセスへの適用を可能とした

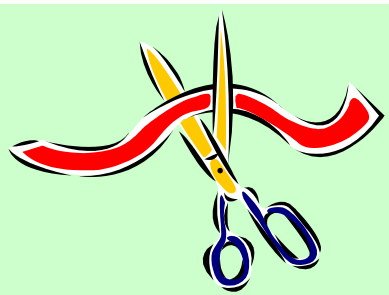
◆ 同値不具合の自動選別

SPIN検証結果からの不具合対応工数を予測可能とした

◆ モデルチェッキングの適用領域/効果予測調査

モジュールテスト時に使用することによる2つの効果
①市場不具合未然防止
②早期品質確保･･･解析困難な状況に持ち込まない

開発プロセス適用の準備ができた
→いよいよ適用へ

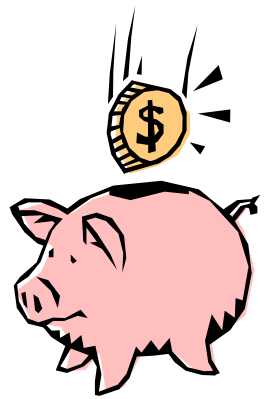


17. 今後の課題

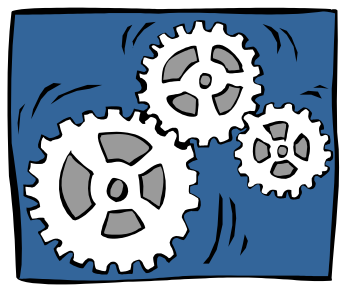
THE DOCUMENT COMPANY
FUJI XEROX

モデルチェッキングを、皆が簡単に、安心して使えるように・・・

- **検証モデルの資産化(プロセス適用実践)**
  機能ごとに分割した検証モデルの作りこみ実施
  →資産として溜め込み、適用範囲を広げつつ、
  各プロダクト開発にプロセスとして適用していく



- **外部モデルの効率的な実装方法**
  現在はPromelaコードを直接手書き
  →Promelaコードを知らなくても
  外部モデルが作れる環境を作る



- **プロダクト全体のテスト最適化**
  プロセス適用実践による効果測定
  →実機テスト削減項目の策定