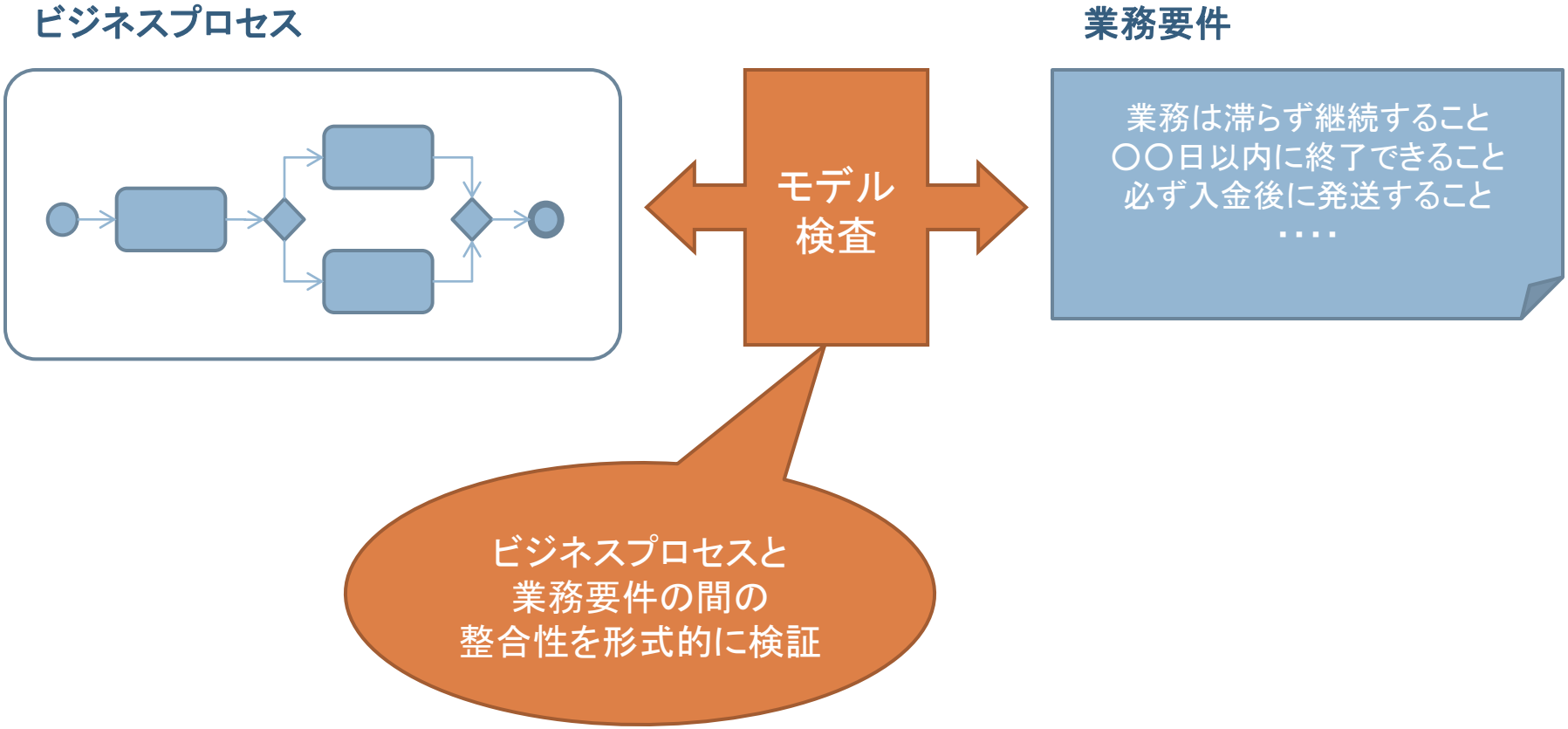
トップエスイー修了制作

モデル検査を用いたビジネスプロセスの検証

綿引　健二

テーマ

- モデル検査技法を用いたビジネスプロセスの検証



アジェンダ

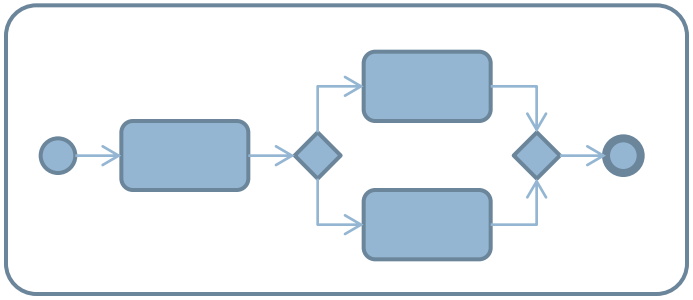
- 背景と目的
- 手法
- 実験
- まとめ
- 今後の課題

背景と目的

- 背景
  - ビジネスプロセス検証の必要性
    - 上流工程における重要な成果物
    - SOAとBPM
  - レビューによる検証の限界
    - すべてのプロセス状態を把握することは困難
  - モデル検査技法
    - システムの状態空間を網羅的に探索して性質を検証
- ビジネスプロセスと業務要件の整合性をモデル検査技法を用いて形式的に検証する手法を提案

アプローチ

ビジネスプロセス



整合？

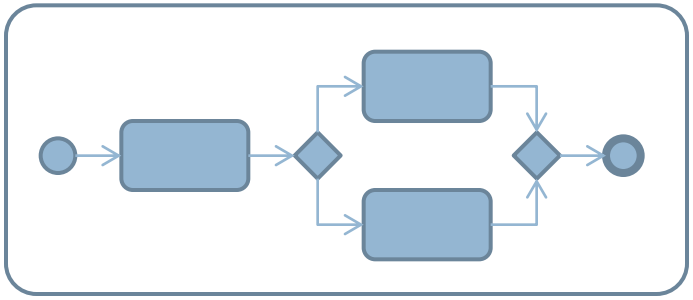
業務要件

業務は滞らず継続すること
〇〇日以内に終了できること
必ず入金後に発送すること
・・・・

- 基本方針
  - ビジネスプロセスはBPMNで記述
  - モデル検査ツールとして“UPPAAL”を利用
    - 時間制約の扱いに優れるため

手法

ビジネスプロセス（BPMN)



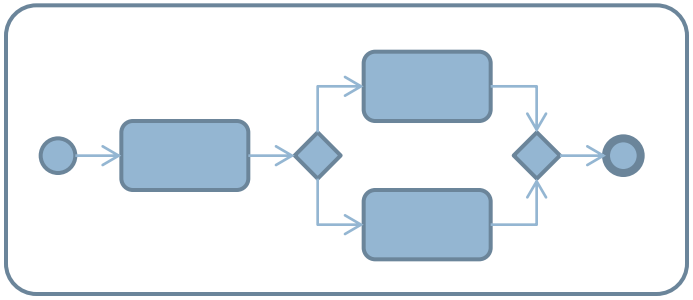
整合？

業務要件

業務は滞らず継続すること
〇〇日以内に終了できること
必ず入金後に発送すること
・・・・

手法

ビジネスプロセス（BPMN)

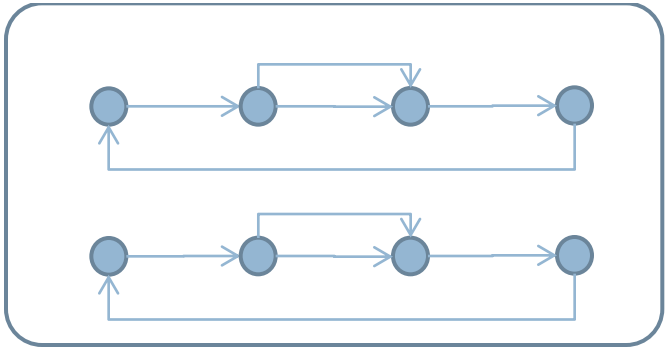


整合？

業務要件

業務は滞らず継続すること
〇〇日以内に終了できること
必ず入金後に発送すること
・・・・

UPPAALモデル（時間オートマトン）

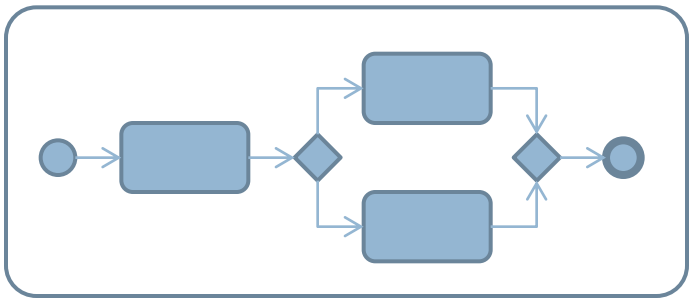


検証式(TCTL)

A□ not deadlock
A□ A1.finish imply ・・・

手法

ビジネスプロセス（BPMN)

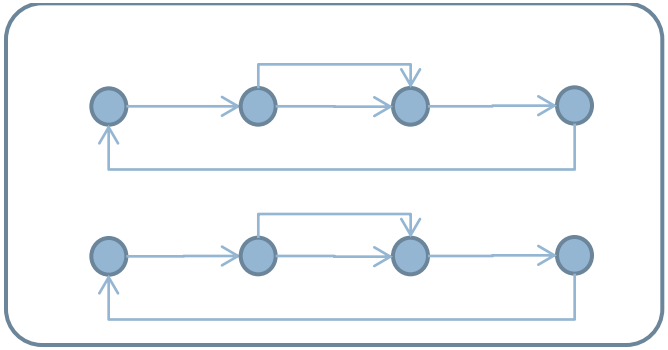


整合？

業務要件

業務は滞らず継続すること
〇〇日以内に終了できること
必ず入金後に発送すること
・・・・

UPPAALモデル（時間オートマトン）



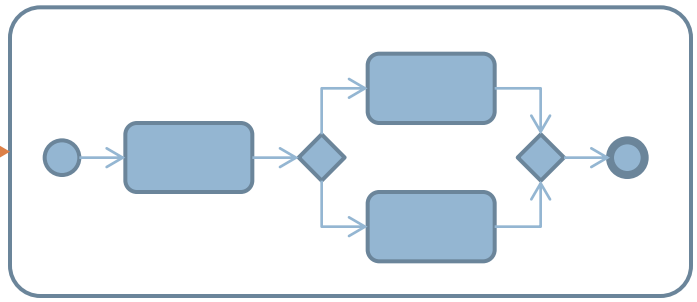
UPPAAL
による
検証

検証式(TCTL)

A□ not deadlock
A□ A1.finish imply ・・・

手法

ビジネスプロセス（BPMN)

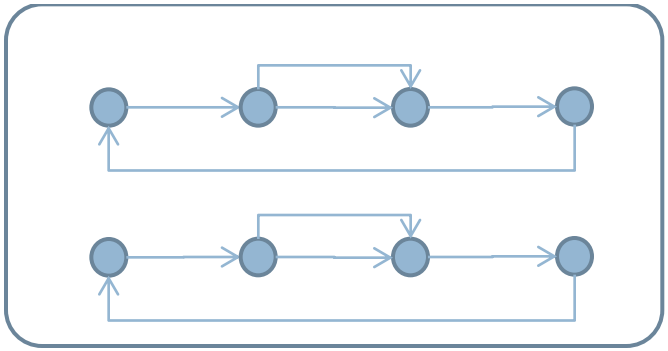


整合？

業務要件

業務は滞らず継続すること
〇〇日以内に終了できること
必ず入金後に発送すること
・・・・

UPPAALモデル（時間オートマトン）



UPPAAL
による
検証

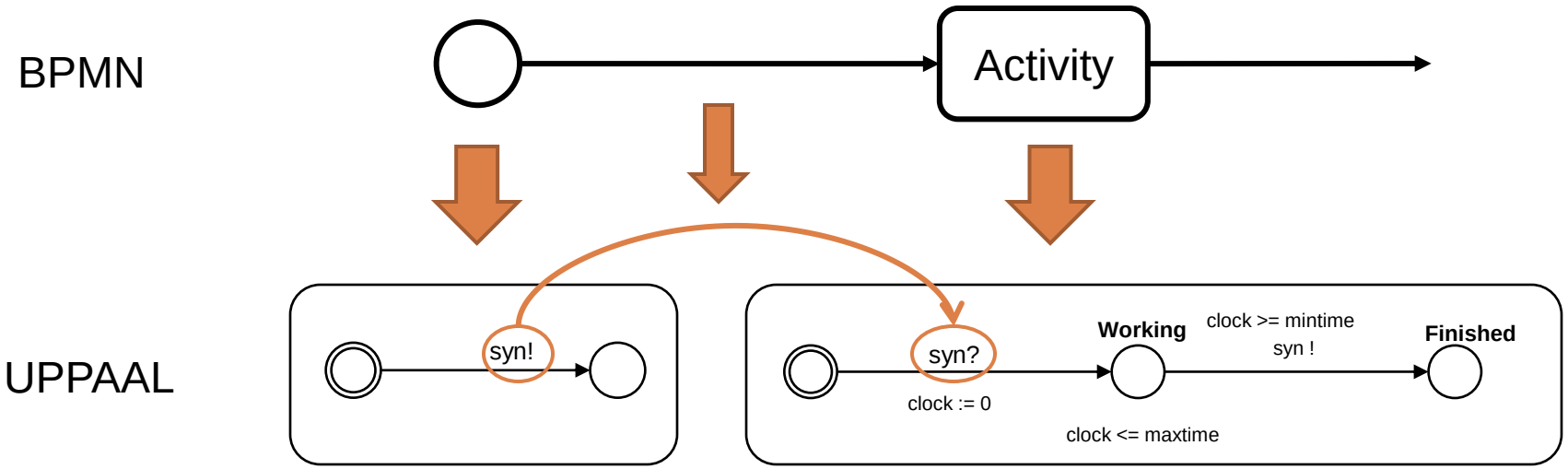
NG

検証式(TCTL)

A□ not deadlock
A□ A1.finish imply ・・・

BPMNからUPPAALへの変換（1）

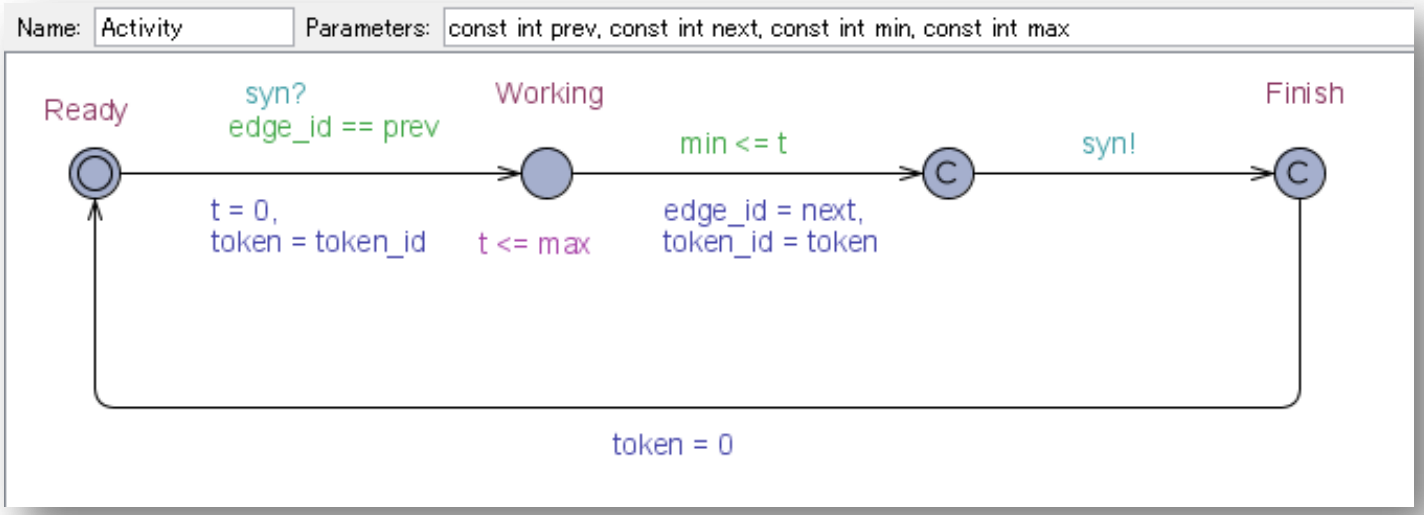
- 変換方針
  - BPMNフローオブジェクト（節）
    - 時間オートマトン（UPPAALでいう1システム）で表現
  - BPMN接続オブジェクト（枝）
    - UPPAALシステム間の同期チャネル通信で表現



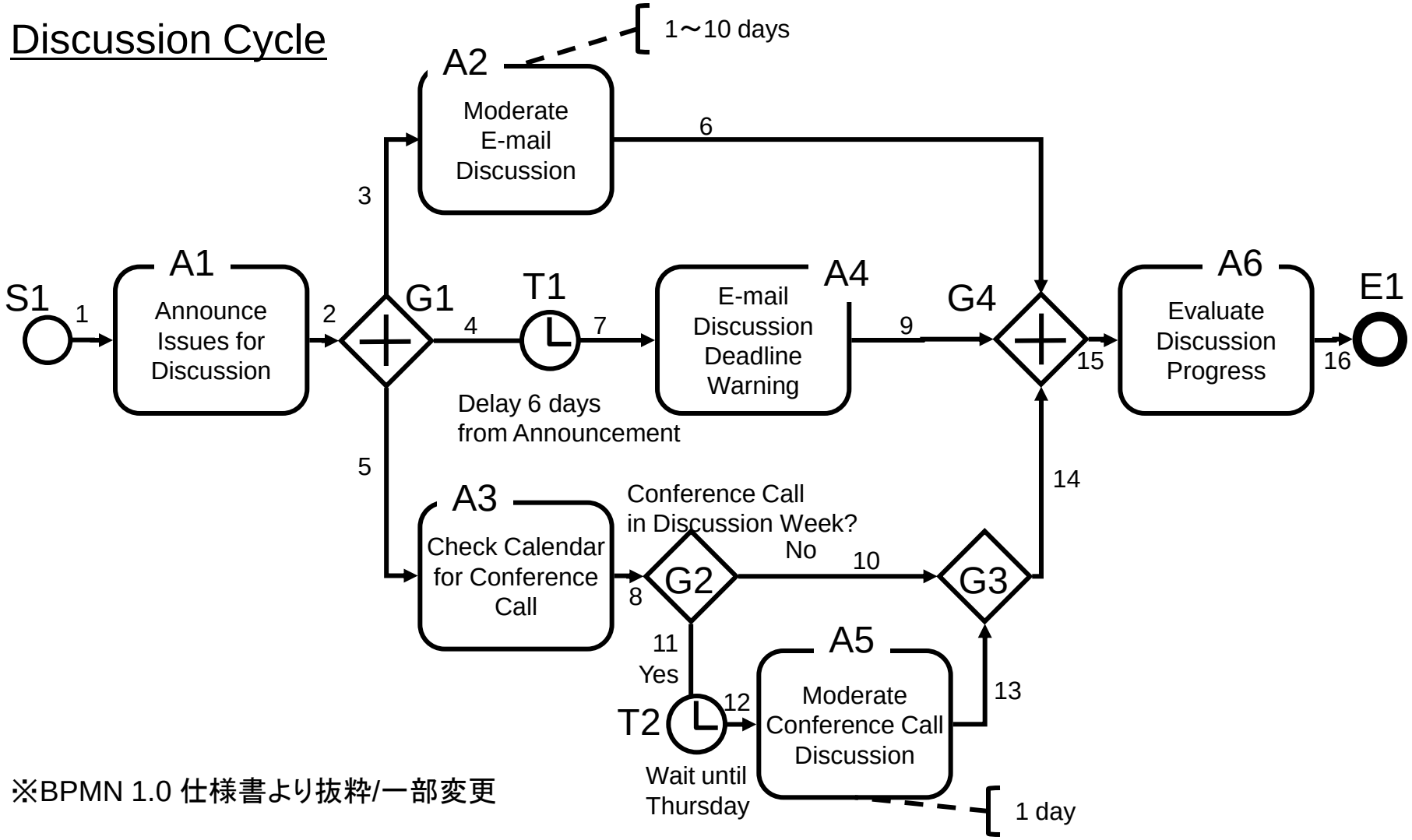
BPMNからUPPAALへの変換(2)

- BPMNの各要素に対応する時間オートマトンは、UPPAALのシステムテンプレートとして蓄積しておく
  - 変換時にはパラメータを与えて利用

- 例:アクティビティの変換テンプレート
  - Activity(前フロー、後フロー、最小実行時間、最大実行時間)



実験



※BPMN 1.0 仕様書より抜粋/一部変更

実験

- 検証したい業務要件
  - 業務が停止しないこと（継続性）
  - 1週間以内で1サイクルが終了すること（リソース）
  - 議論が終了していない場合にのみ、締切の警告をすること（業務ルール）

BPMNからUPPAALへの変換（1）

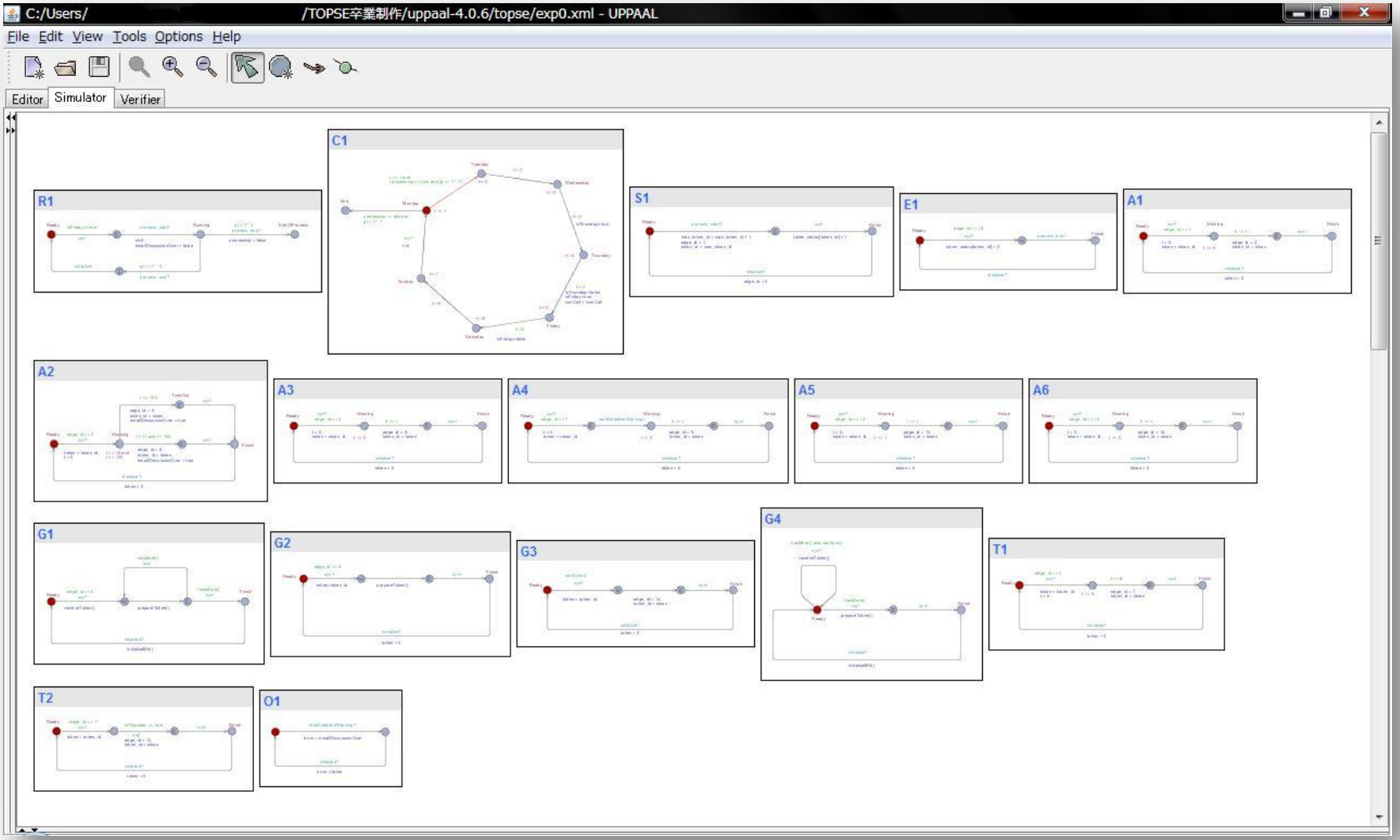
- システム定義（抜粋）

```
S1 = StartEvent(1);
E1 = EndEvent(16);

A1 = Activity(1, 2, 0, 0);
 （中略）
A6 = Activity(15, 16, 0, 0);

G1 = AND(2, g1next, 3);
 （中略）

T1 = TimerDelay(6, 4, 7);
T2 = TimerAt(isThursday, 11, 12);

R1 = Root();
C1 = Calendar();
O1 = DiscussionObserver();

system R1, C1, S1, E1, A1, A2, A3, A4, A5, A6, G1, G2, G3, G4, T1, T2, O1;
```

テンプレートにパラメータを
与えてシステムを定義

追加作成したシステムと
検証のためのオブザーバ

BPMNからUPPAALへの変換(2)



業務要件の検証

- 検証式への変換と検証実行
  - 業務が停止しないこと
    - A[] (not deadlock) or R1.EndOfProcess
  - 1週間以内で1サイクルが終了すること
    - A[] E1.Finish imply ct <=7
  - 議論が終了していない場合にのみ、締切の警告をだすこと
    - A[] not O1.error

業務要件の検証

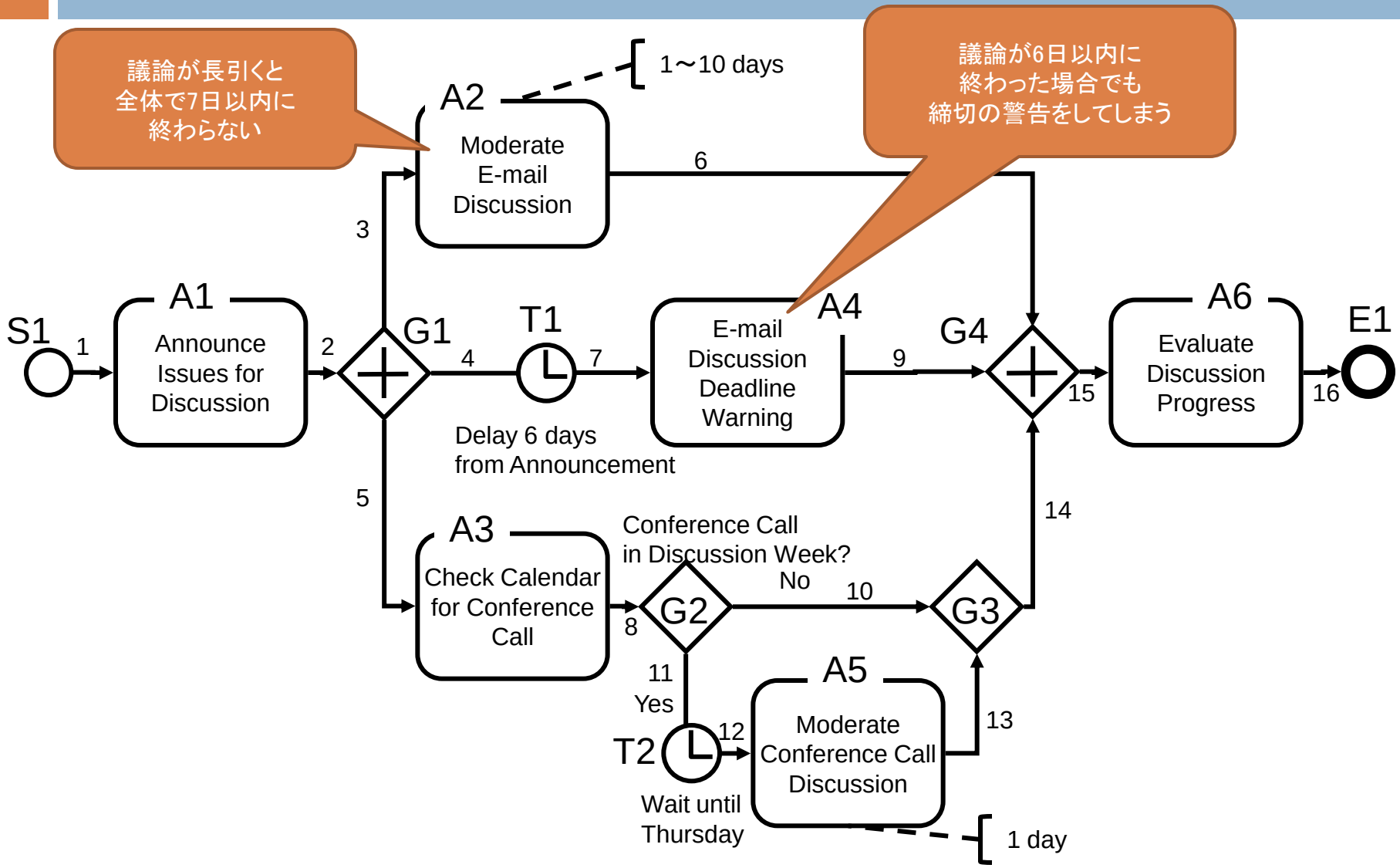
- 検証式への変換と検証実行

OK
- 業務が停止しないこと
  - A[] (not deadlock) or R1.EndOfProcess

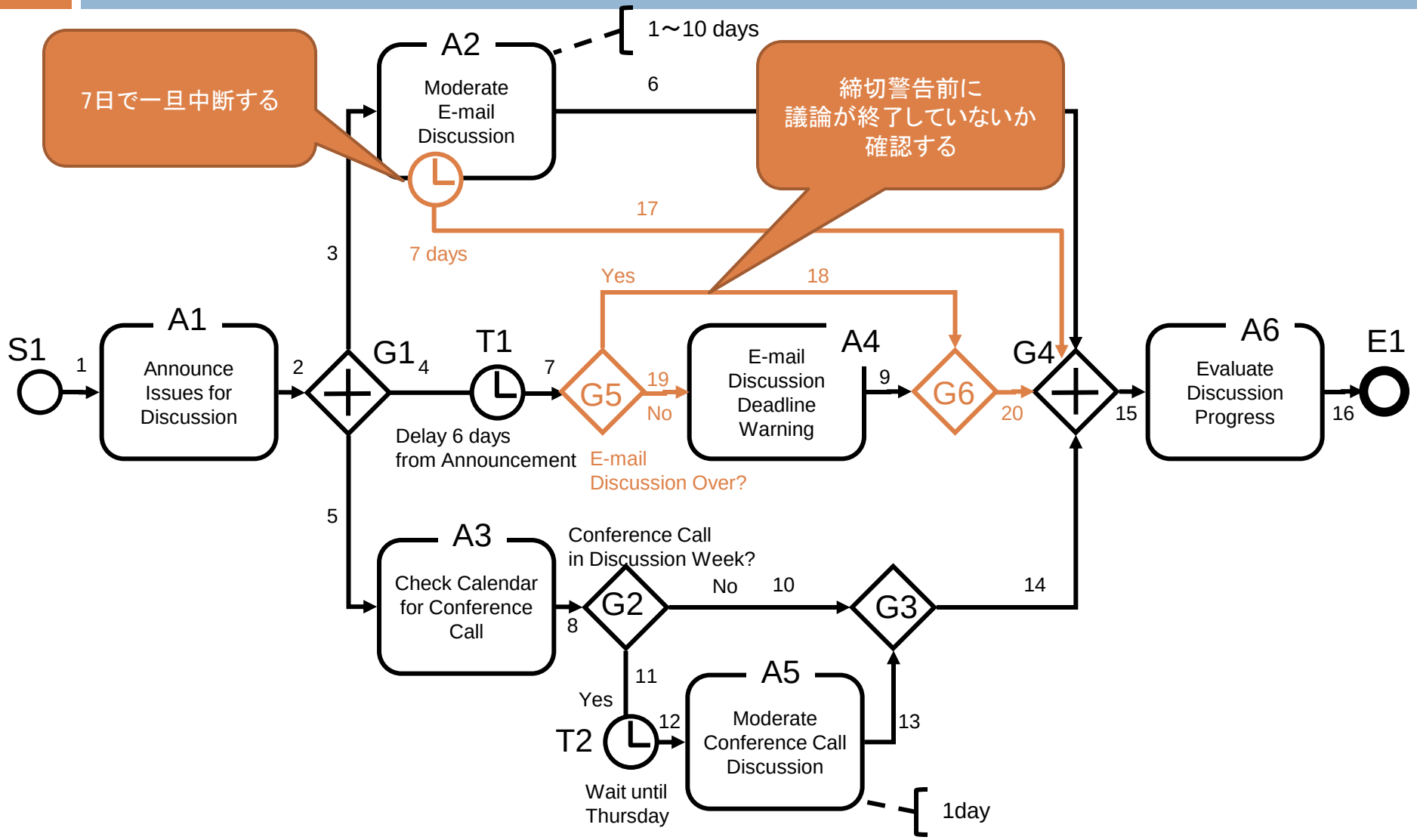
NG
- 1週間以内で1サイクルが終了すること
  - A[] E1.Finish imply ct <=7

NG
- 議論が終了していない場合にのみ、締切の警告をだすこと
  - A[] not O1.error

ビジネスプロセスのデバッグ



ビジネスプロセスの修正



業務要件の検証(再)

- 検証再実行
  - 業務が停止しないこと
    - A[] (not deadlock) or R1.EndOfProcess
  - 1週間以内で1サイクルが終了すること
    - A[] E1.Finish imply ct <=7
  - 議論が終了していない場合にのみ、締切の警告をだすこと
    - A[] not O1.error

業務要件の検証(再)

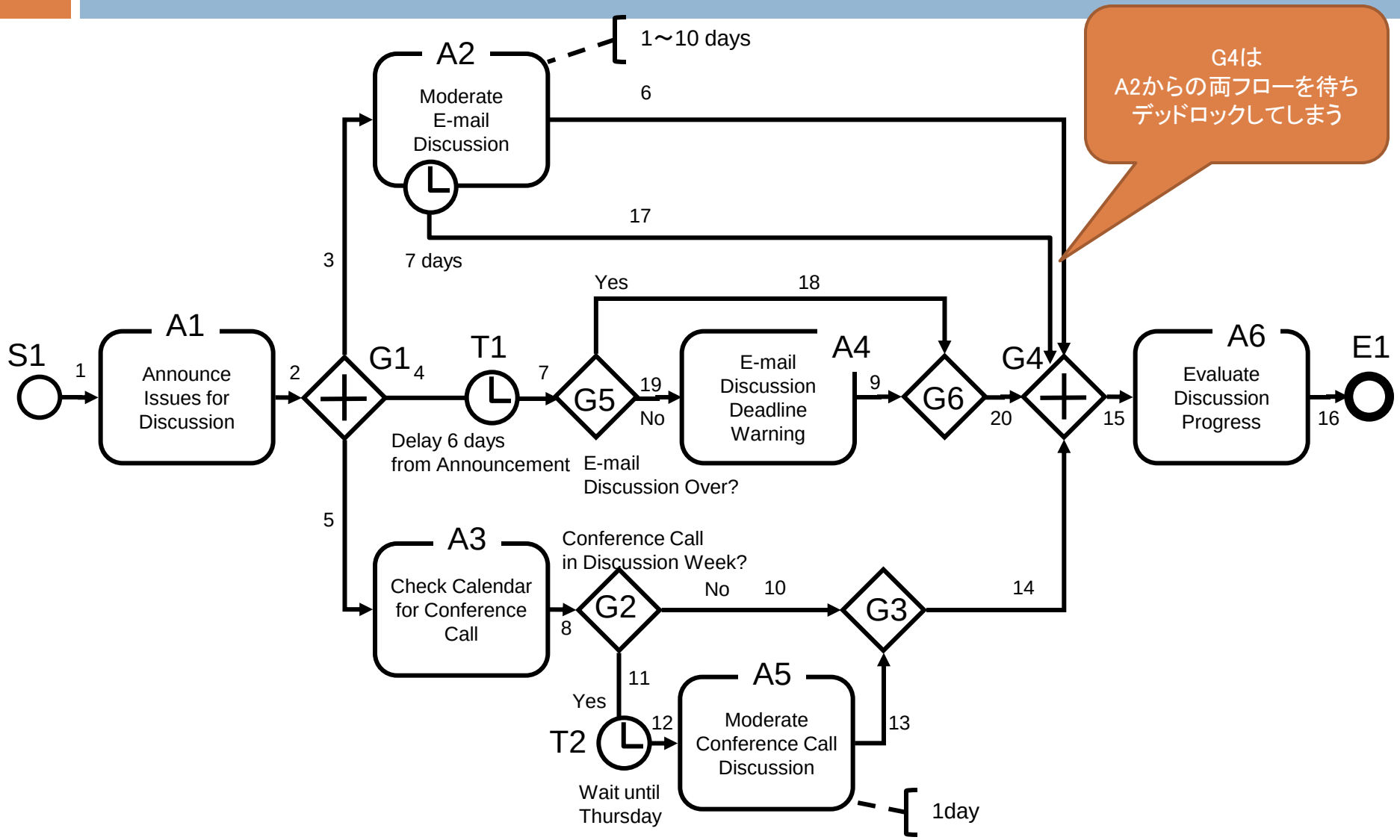
- 検証再実行

NG
- 業務が停止しないこと
  - A[] (not deadlock) or R1.EndOfProcess

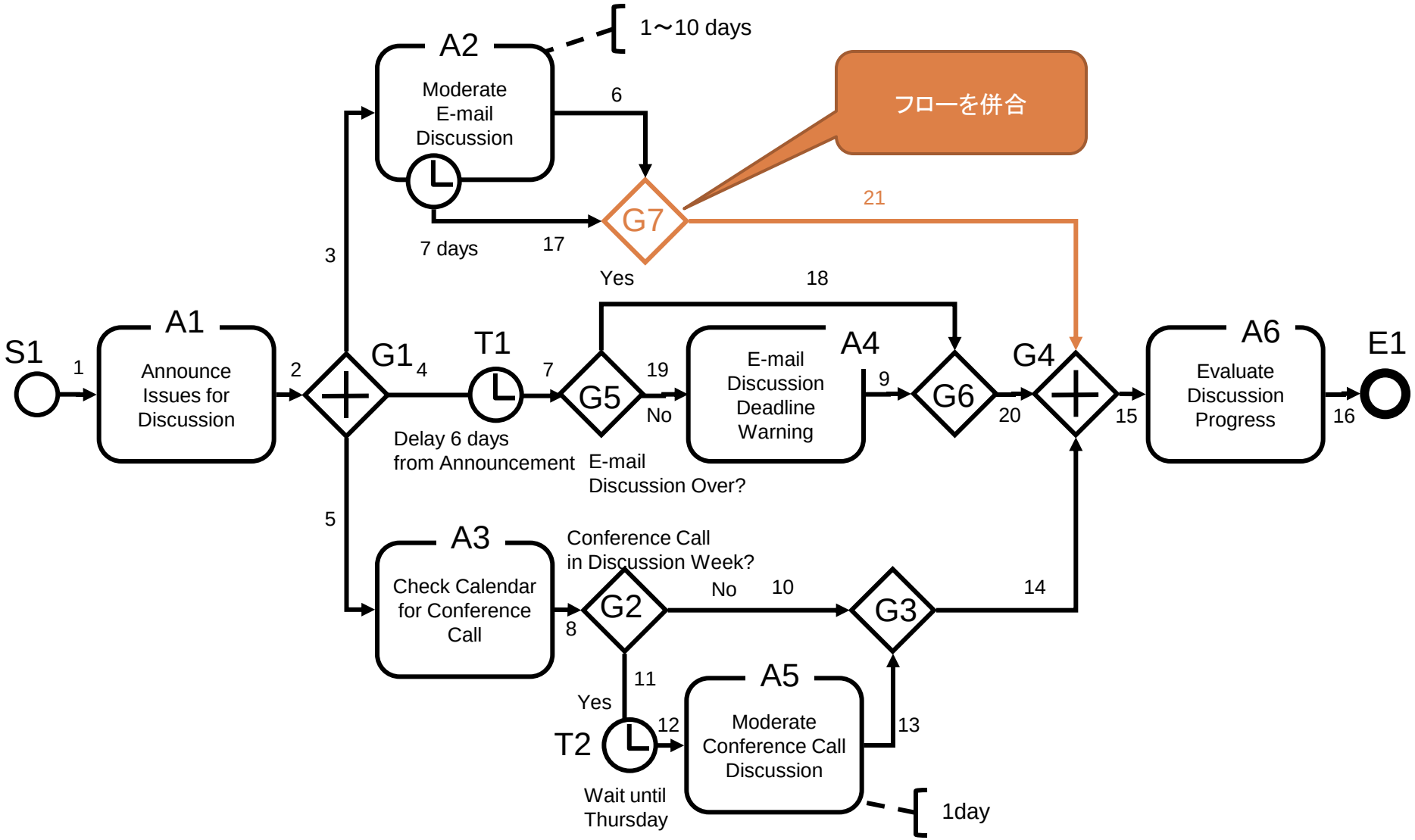
OK
- 1週間以内で1サイクルが終了すること
  - A[] E1.Finish imply ct <=7

OK
- 議論が終了していない場合にのみ、締切の警告をだすこと
  - A[] not O1.error

ビジネスプロセスのデバッグ（再）



ビジネスプロセスの修正（再）



まとめ

- UPPAALによるビジネスプロセス検証手法を提案
  - BPMNをUPPAALモデル（時間オートマトン）に変換
    - 変換はテンプレートにより半自動化可能
  - 業務要件をUPPAAL検証式（TCTL）に変換

- 適用実験を行い手法の有効性を確認
  - 継続性、実行リソース（時間）、業務ルールに関する性質を検証、デバッグ

今後の課題

- 変換テンプレートの拡充
- 業務要件から検証式への変換方法
- UPPAAL上での反例から、BPMN上での不具合箇所の特定方法