Supporting Composition of Distributed Business Processes

astro

Verification and Validation of Web Service Compositions

Raman Kazhamiakin
raman@dit.unitn.it
DIT, University of Trento, Italy

Joint work with Marco Pistore, Paritosh Pandya, and Luca Santuari

Context

ASTRO project

Any method that prevents the programmer writing code, is a good method

T. Reenskaug

- Goal
  - Development of formal techniques and (semi) automated tools that **support the design, evolution, and execution of Web service compositions**
- Tasks
  - Service composition **verification and analysis**
  - Service **monitoring**
  - Service **synthesis**
- Means
  - **BPEL** as a Web service description language: business processes and business protocols
  - **Formal techniques** substantially extended in order to address WS-specific problems (model checking, AI planning and synthesis, etc.)
  - Integrated **environment**

In the beginning…

- Problem domain: **Web service compositions**
  - distributed business processes
  - stateful, long-running component services (e.g., **WS-BPEL** services)
- Goal: analysis of **correctness of the composition behavior**
  - deadlock, livelock freeness
  - behavioral requirements (Message Sequence Charts, LTL properties)

> *If the customer makes request, eventually he will receive an offer*
>
> *If the customer cancels the transaction, all the other participants should also cancel*
>
> *Successful termination of the process is possible*

In the beginning…

- Problem domain: **Web service compositions**
  - distributed business processes
  - stateful, long-running component services (e.g., **WS-BPEL** services)
- Goal: analysis of **correctness of the composition behavior**
  - deadlock, livelock freeness
  - behavioral requirements (Message Sequence Charts, LTL properties)
- Initial approach: **model checking**
  - components as State Transition Systems
  - composition as **synchronous** product
  - variables of **finite** ranges, no functions
  - behavior is **timeless**

Outline

- Analysis of communication models
- Data-flow analysis
- Analysis of time-related properties
- Ongoing work and future directions

Outline

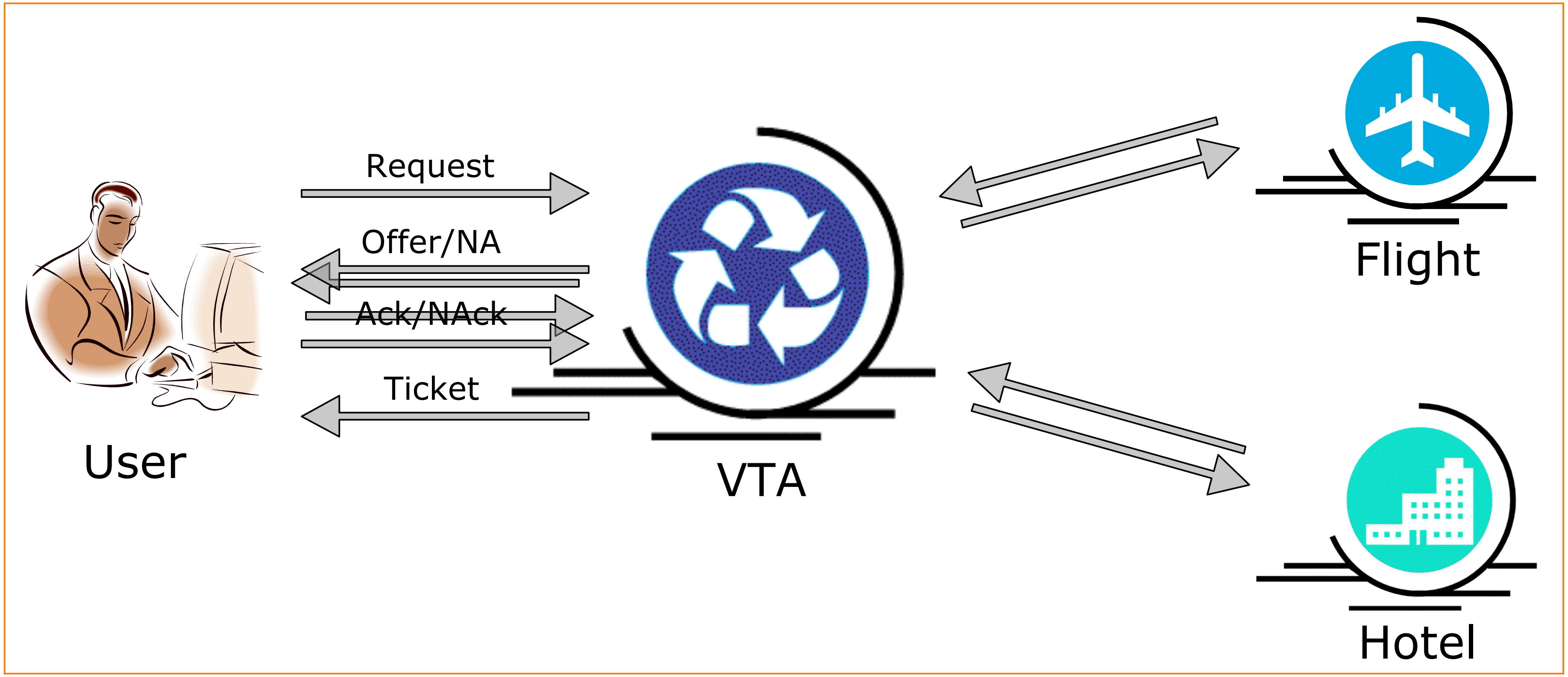
- **Analysis of communication models**
- Data-flow analysis
- Analysis of time-related properties
- Ongoing work and future directions

Problem #1: Interactions

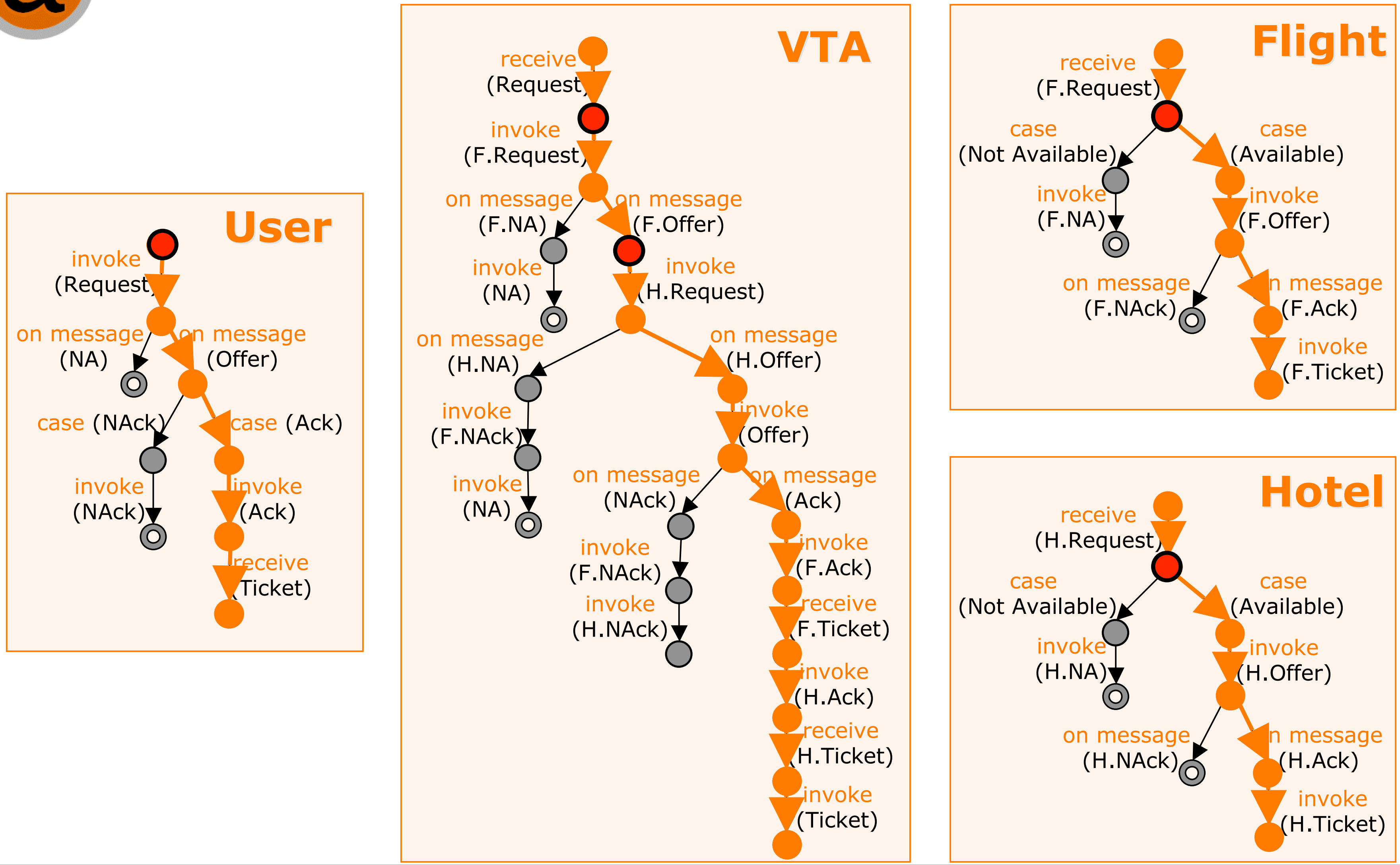
Synchronous model of communications is not adequate!

Virtual Travel Agency

- Provide combined flight and hotel booking service
- Integrate separate **Hotel** and **Flight** booking services
- Participants are represented with their **BPEL** specifications



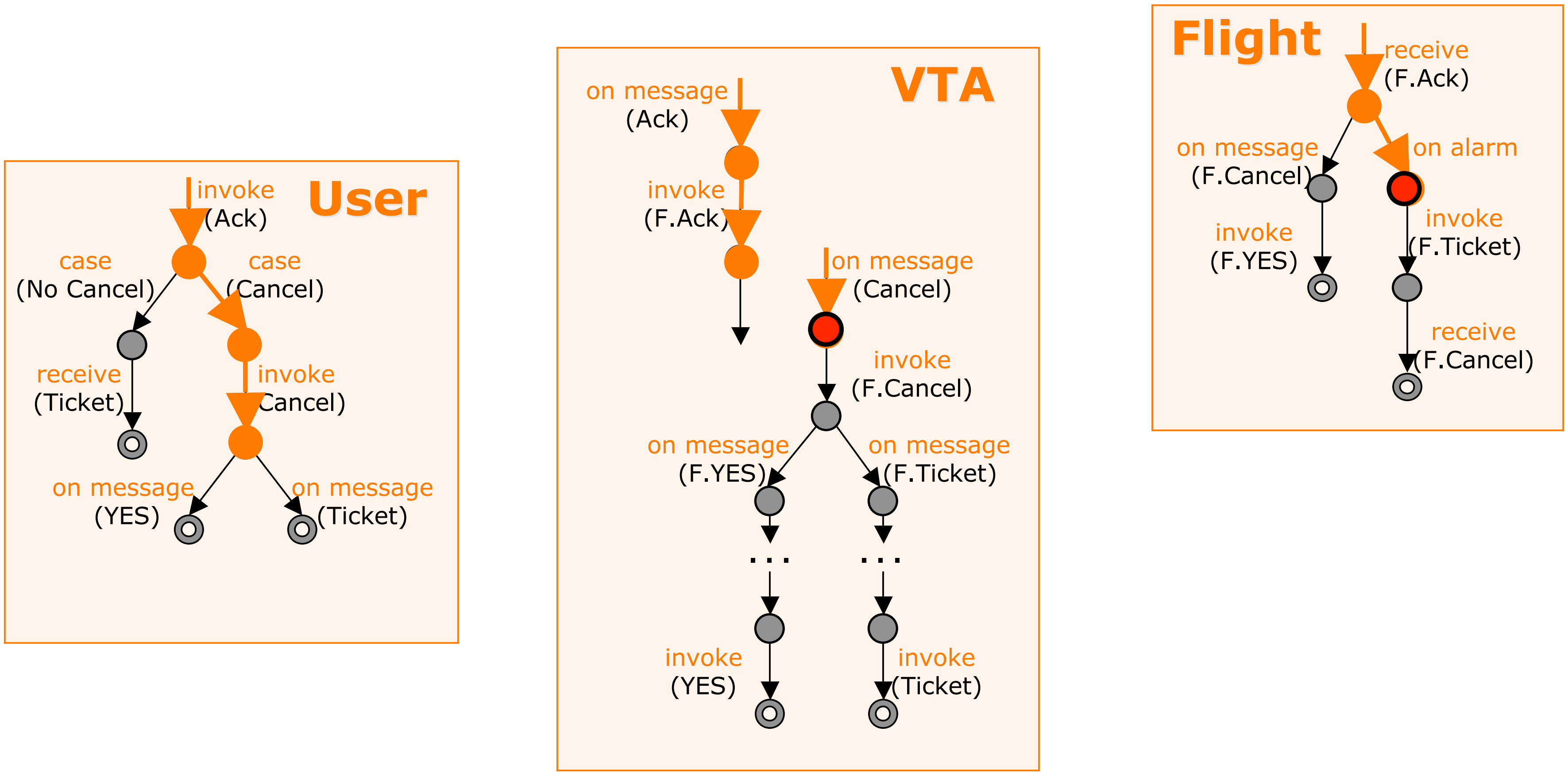
VTA processes



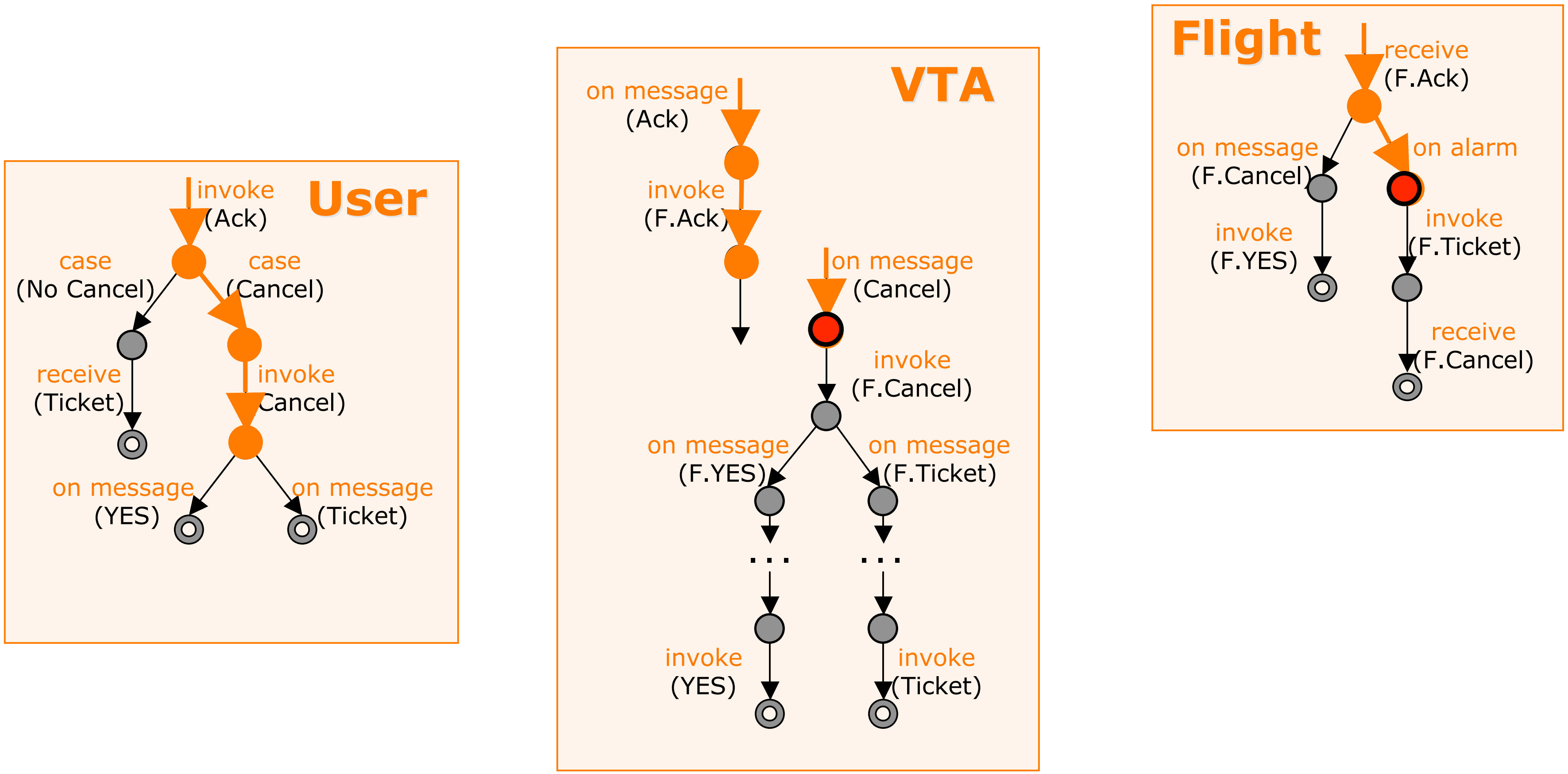
Composition properties

- The composition is **synchronizable**
  - At any moment of time only one component emits a message
  - The receiver is immediately ready to consume the message
- **Synchronous communication model**
  - Components synchronize on shared actions
  - Efficient reasoning techniques
  - Universally used in verification tools for web service compositions
- The synchronous communication model is **adequate** for synchronizable compositions
  - The presence of queues in the implementation does not add new behaviors

VTA Processes – cancellation

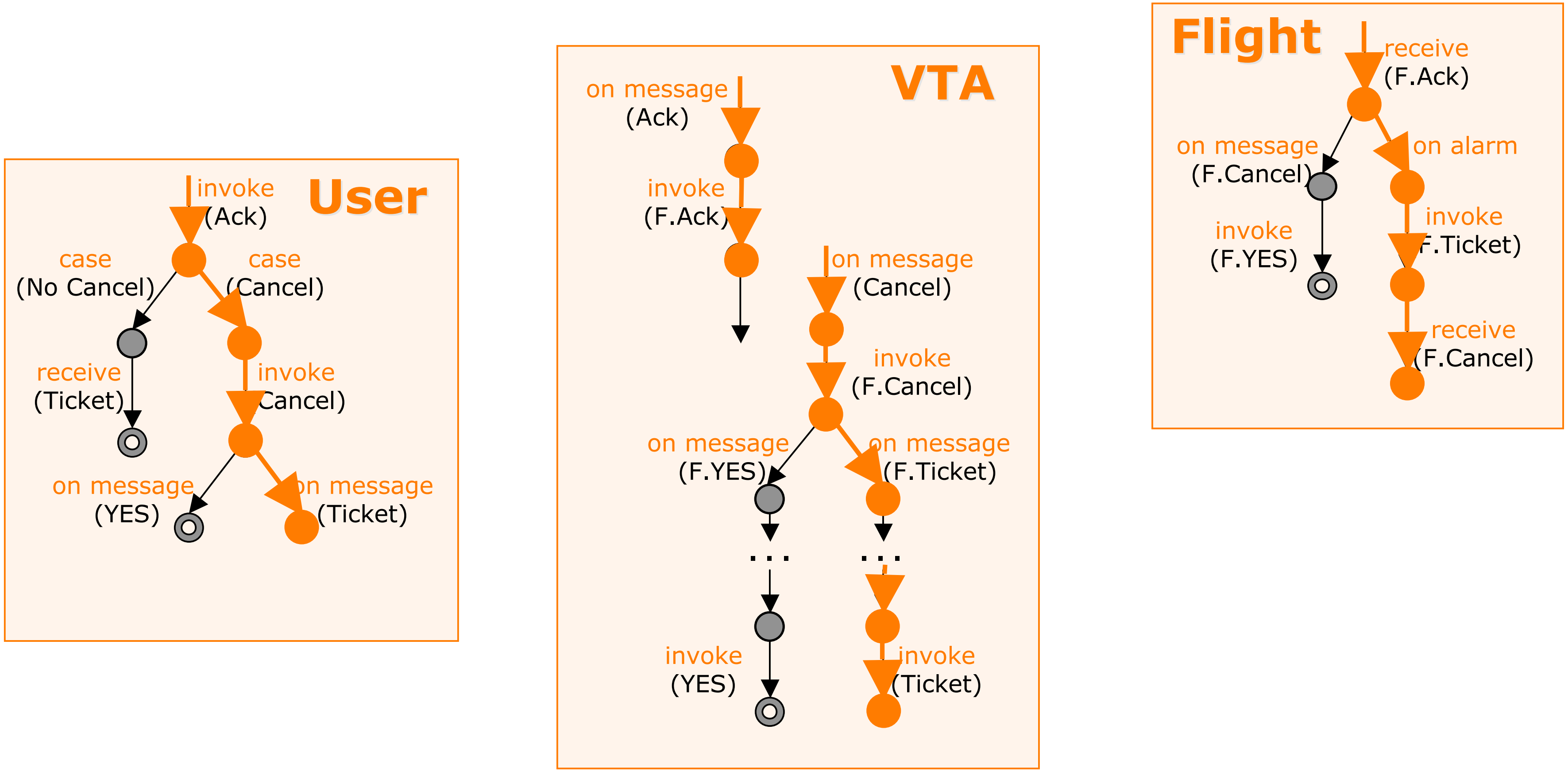


VTA Processes – cancellation



The synchronous communication model is violated

VTA Processes – cancellation



The real execution is correct

Problem #1: Interactions

Synchronous model of communications is not adequate!

- Wide range of advanced scenarios
  - concurrent emissions, message losses, message reordering
- Complex queue and message management mechanisms

We cannot apply one communication model for all compositions!

- Diversity of middleware and protocol implementations
- Tradeoff between expressiveness and analysis complexity

Our approach [Kazhamiakin,Pistore, Santuari, WWW'06]

- Define a **set of communication models**
  - Different levels of complexity
  - Different interaction mechanisms
  - Common framework

- Given a certain composition scenario determine an **adequate** communication model
  - Represents all real executions of the composition
  - Preserves behavioral properties

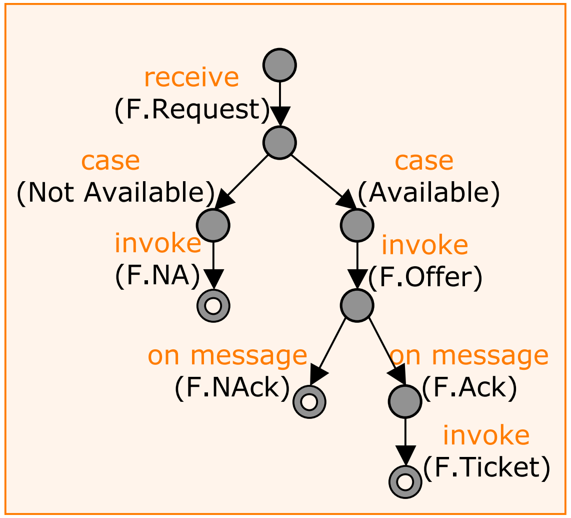
- **Incremental** analysis process
  - From simple to complex communication models
  - Check if the communication model is adequate w.r.t. the scenario
  - If yes, perform the formal verification against this model

Our approach: formal definitions

- Three main ingredients:
  - Component services are formally modeled as **State Transition Systems**
  - The modalities of the communications are formalized as a **Communication Model**
  - The composite behavior of the component services according to a specific communication model is formally described as a **Global State Transition System**

From BPEL to STS

- **State Transition System** $\Sigma = \langle S, S_0, I, O, R \rangle$ where
  - S – finite set of states
  - S_0 – set of initial states
  - I – set of input actions
  - O – set of output actions
  - $R \subseteq S \times (I \cup O \cup \{\tau\}) \times S$ – transition relation




```
PROCESS Flight
STATES {Start, switch_IsAvailable, OUT_FNA, SUCCESSS,…}
INPUT FRequest, FNAck, FAck
OUTPUT FNA, FOffer, FTicket
INIT
      state = Start
TRANS
      Start – [IN FRequest] -> switch_IsAvailable
      switch_IsAvailable – [TAU] -> OUT_FNA
      switch_IsAvailable – [TAU] -> OUT_FOffer
      …
```

Communication model

- A communication model **Δ** is defined by a set of queues

$$\langle Q_1, Q_2, \ldots, Q_n \rangle$$

where each queue Q_i has associated:

  - A set of messages M_i
  - A (finite or infinite) bound B_i on the messages it can contain
  - An ordering constraint: ordered or unordered

- Allows for the definition of various interaction mechanisms
  - Synchronous (1 ordered queue with bound 1)
  - Ordered asynchronous (1 ordered unbounded queue for each actor)
  - Unordered asynchronous (1 unordered unbounded queue)
  - Mixed synchronous/asynchronous, mixed bounded unbounded, …

Global State Transition System

- A Global State Transition System (GSTS):
  - defines the composite behavior of the system.
  - is parametric w.r.t. a communication model **Δ**

- A GSTS is a tuple $G = \langle GS, GS_0, A, T \rangle$, where:
  - **GS** are the global states; each state has the form $gs = (<s_1, s_2, \ldots, s_n>, <q_1, q_2, \ldots, q_m>)$, where:
    - s_i is the state of the i-th component STS
    - q_j describes the content of the j-th queue
  - GS_0 are the initial global states
  - **A** are the input-output actions
  - $T \subseteq GS \times A \times GS$ is the transition relation

GSTS: transitions

- The transition relation $T \subseteq GS \times A \times GS$ is defined as follows:
  - If the i-th STS performs an output:
    - update the status of the STS
    - add the message to the associated queue (if the bound allows)
  - If the i-th STS performs an input:
    - consume a message from the associated queue (the queue has to be non-empty!)
    - update the status of the STS
  - If the i-th STS performs a TAU action:
    - update the status of the STS

Hierarchy of communication models

- Relation between models
  - **$\Delta_1 < \Delta_2$**

    Model 2 simulates model 1 if for any composition scenario **$G_{\Delta 1} < G_{\Delta 2}$**
  - Defined by the structure of the communication model (bounds, ordering and alphabets)
  - There exists the **most general** model that simulates any other model

- Hierarchy of communication models
  - Partially ordered set of models with the MG model as top element and the synchronous model as the bottom element

$$\Delta_{sync} < \ \dots < \Delta_{MG}$$

Finding an appropriate model

Given a set of STS, and a communication model **Δ**, build a reachability graph of the GSTS (DFS algorithm)

- On every state of the search compare the set of enabled transitions with the one under the MG model
- If the sets are different, the model is not adequate

- Efficient analysis algorithm
- The resulting graph is used for further analysis
- On-the-fly boundedness analysis
- Allows for partial order reduction techniques
- Implemented as a part of the **Astro** verification toolkit

Outline

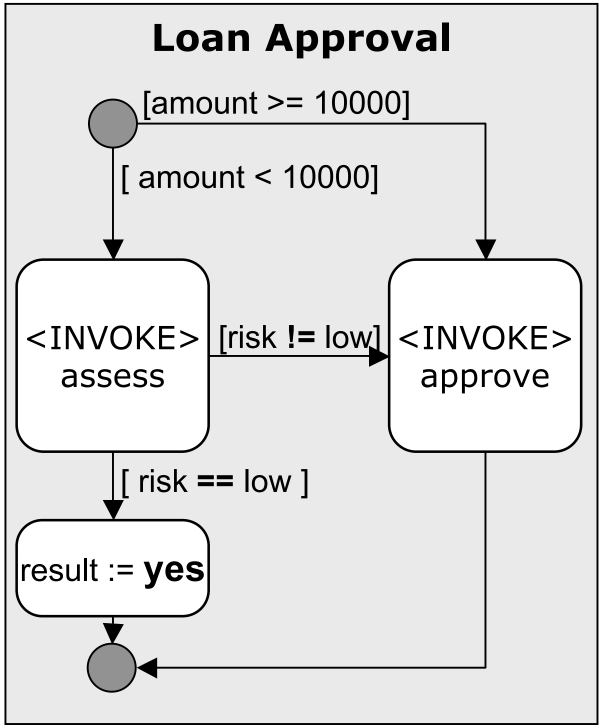
- Analysis of communication models
- **Data-flow analysis**
- Analysis of time-related properties
- Ongoing work and future directions

Problem #2: Data flow

Data flow should be properly modeled and analyzed!

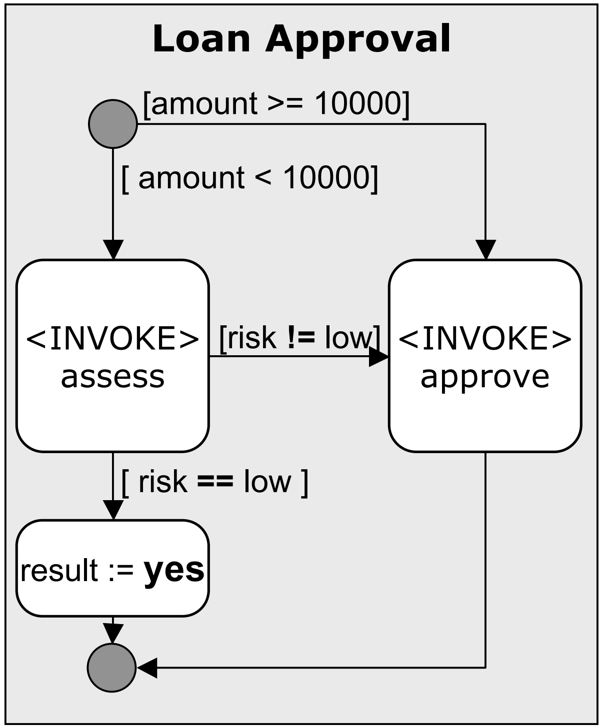
Loan Approval

Ignoring data affects control flow:



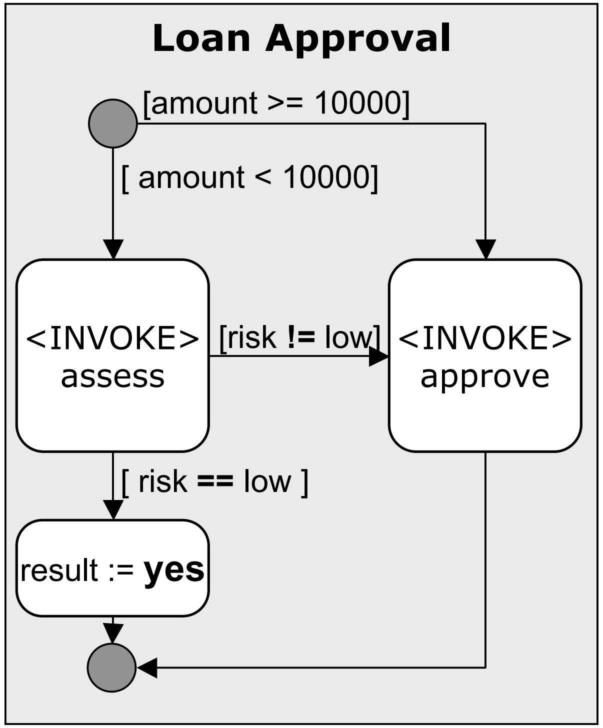
Loan Approval

Ignoring data affects control flow:



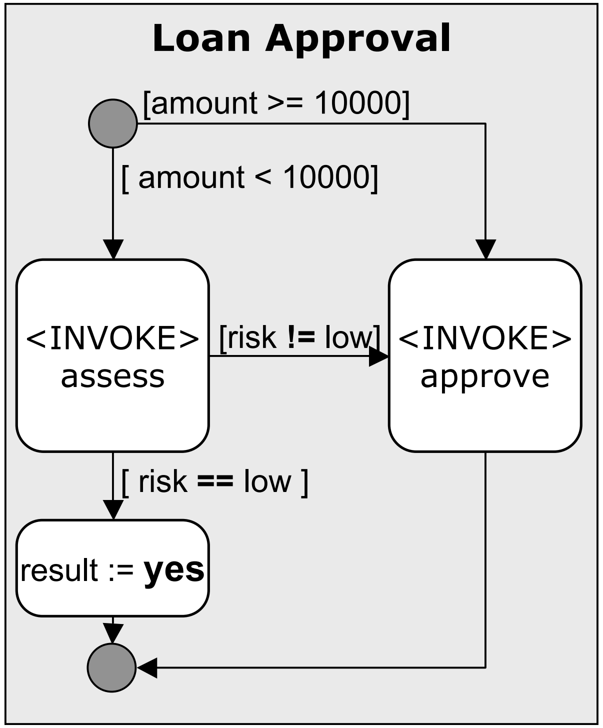
Loan Approval

Ignoring data affects control flow:



Loan Approval

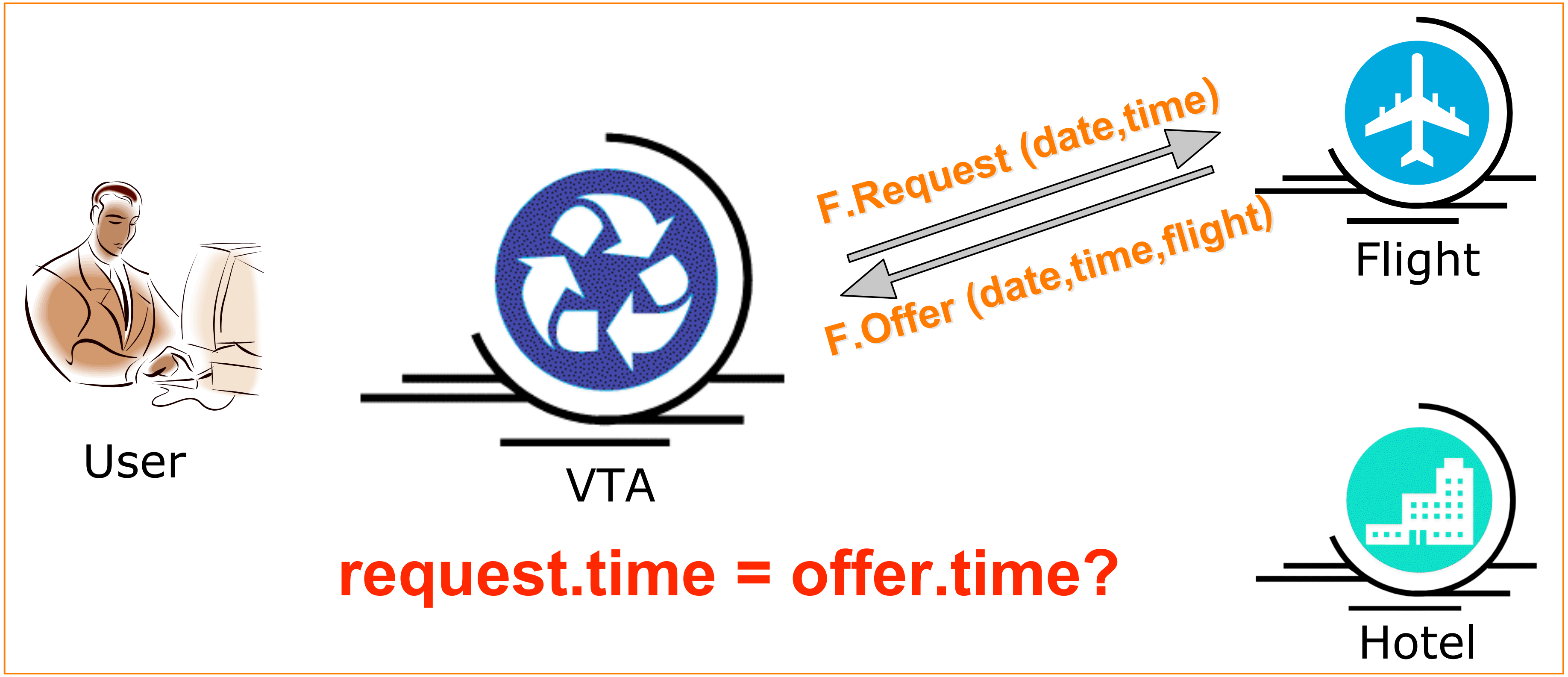
Ignoring data affects control flow:



Bad scenario: amount> 10000, approve.result = ‘no’, assess.risk = ‘low’, result = ‘yes’

Virtual Travel Agency

Incomplete information on service implementations and functions



Additional assumptions on the internal of the service implementations are needed

Problem #2: Data flow

Data flow should be properly modeled and analyzed!

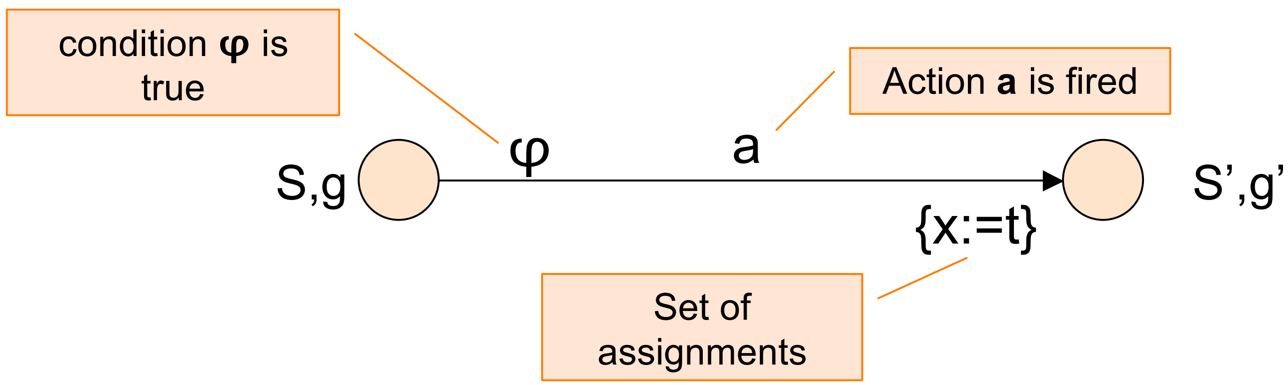
- **Complex data model** of WS compositions
  - Infinite ranges, custom types, custom functions
- Necessity to manage **information incompleteness**
  - Put additional assumptions on unknown functions/operations
- Often ignored in analysis frameworks
  - Only control-flow analysis
  - Finite data ranges

Our approach [Kazhamiakin,Pistore, ICWS'06]

- Extend a composition model with **data flow**
  - Variables, functions, expressions
  - Extended behavioral semantics
- Provide a set of **analysis techniques**
  - **Abstraction-based approach**
  - Support for **universal** (hold for all system executions) and **existential** (hold for some system executions) properties
- **Iterative** analysis process
  - Allow to put additional constraints on unknown functions
  - Combine the verification and the elicitation of requirements

Our approach: formal definitions

- **Data context**: $<V,T,F>$ - variables, types, functions
  - **Expressions**: $E{:=}\ (t_1{=}t_2)\ |\ !E\ |\ E_1\ or\ E_2$, where $t{:=}\ x\ |\ f(t_1,\ldots,t_n)$
  - **Ground state**: $g\ = \{<x,v>\}$ – set of valuations of the variables
- **Extended Transition System** $\Sigma = \langle \boldsymbol{V},\boldsymbol{S},S_0,I,O,\boldsymbol{R}\rangle$ where
  - V – finite set of variables
  - S – set of pairs <s,g>
- transition relation



The model is infinite…

- How can we define abstractions?
  - Define a **set of propositions** representing certain facts
  - **Valuation of propositions** instead of valuation of variables – finite model

- **Conservative** (branching) **abstraction**
  - Concrete system ***C*** is **over-approximated**: when the fact can not be determined, both states are allowed
  - Allow for more behaviors than the real system

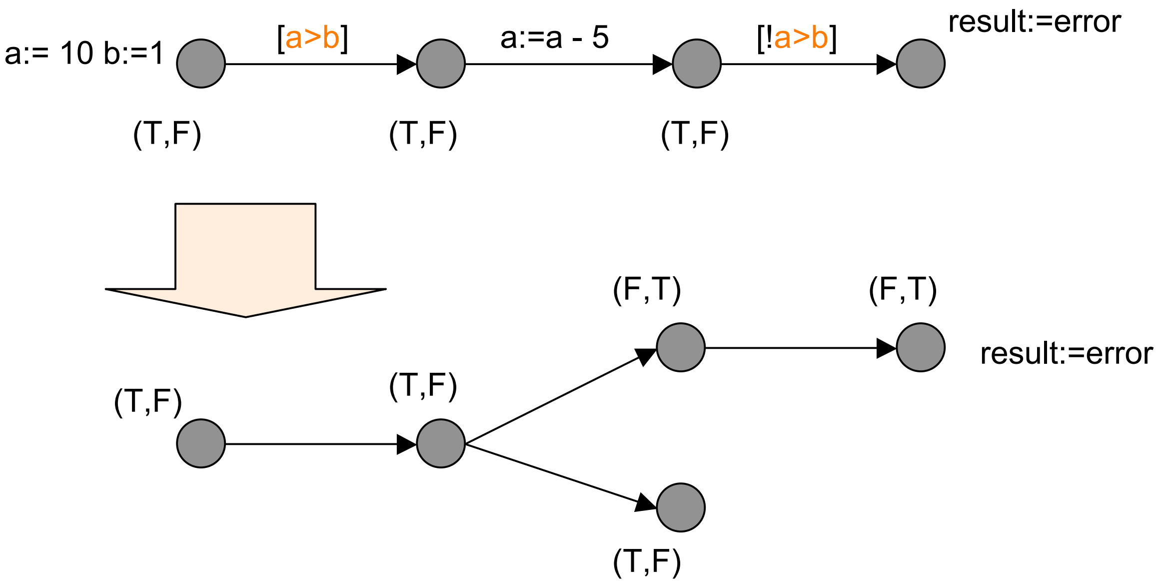
$$L(A) \geq L(C)$$

  - Applicable for universal properties but **not for existential** properties

The model is infinite…

- How can we define abstractions?
  - Define a **set of propositions** representing certain facts
  - **Valuation of propositions** instead of valuation of variables – finite model

- **Conservative** (branching) **abstraction**



P = {(a>b), !(a>b)}

The model is infinite…

- How can we define abstractions?
  - Define a **set of propositions** representing certain facts
  - **Valuation of propositions** instead of valuation of variables – finite model

- **Conservative** (branching) **abstraction**
  - Interpretation of the valuation: *set of states, where the true propositions evaluate to true, false - to false*
  - Applicability of the transition: *the transition is applicable, if its condition evaluates to true in* ***some state*** *of the interpretation*
  - The effect of the transition: *the resulting valuation is a valuation, obtained by* ***modifying some state*** *of the initial valuation*

The model is infinite…

- How can we define abstractions?
  - Define a **set of propositions** representing certain facts
  - **Valuation of propositions** instead of valuation of variables – finite model

- **Knowledge level abstraction**
  - Concrete system ***C*** is **under-approximated**
  - The fact may be either known to be **true**, or **unknown**
  - The safest information on the propositions is used
  - Allow for less behaviors than the real system

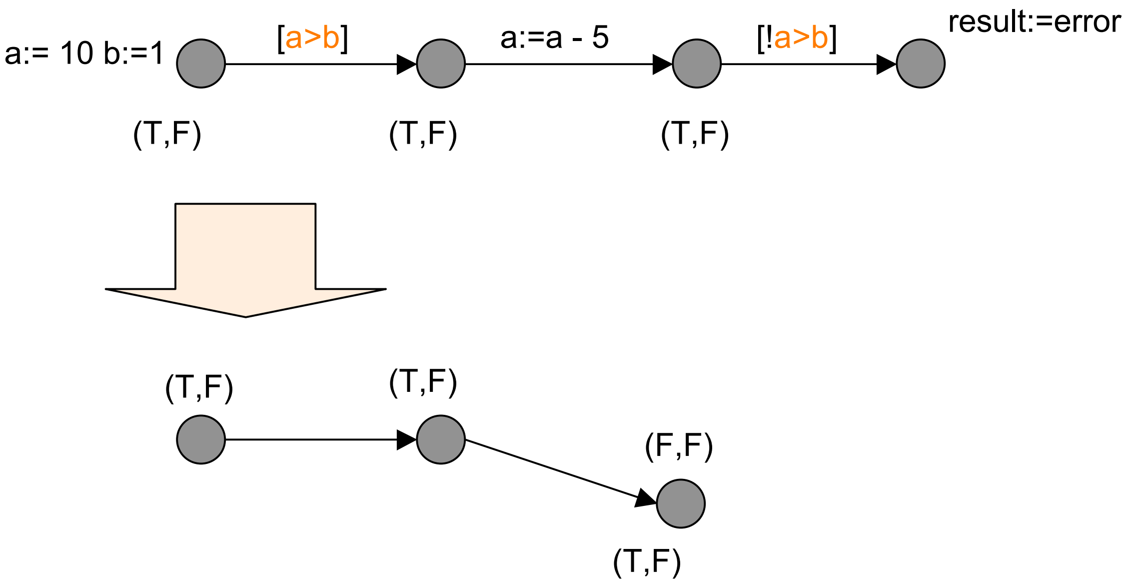
$$L(C) \geq L(A)$$

  - Applicable for **existential properties**

The model is infinite…

- How can we define abstractions?
  - Define a **set of propositions** representing certain facts
  - **Valuation of propositions** instead of valuation of variables – finite model

- **Knowledge level abstraction**



$P = \{(a>b), !(a>b)\}$

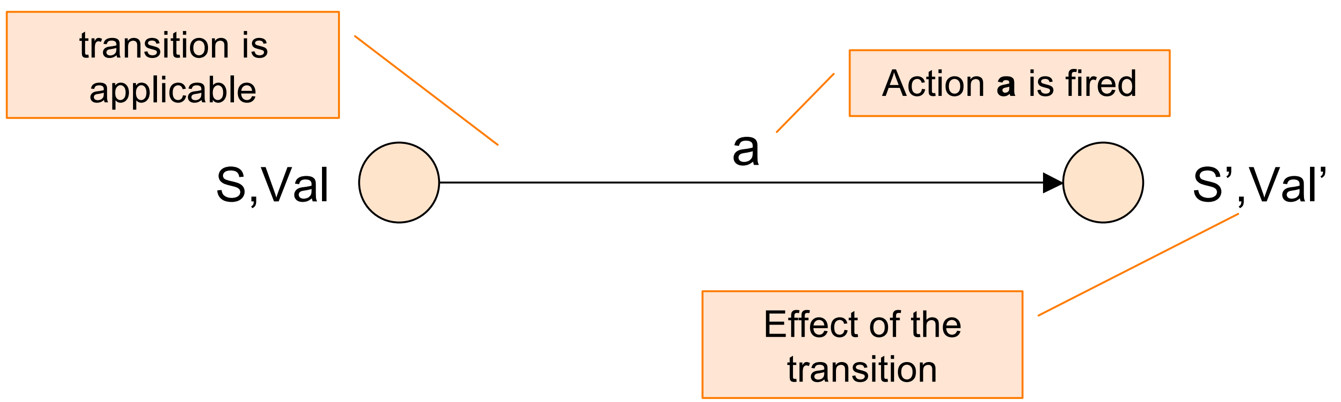
The model is infinite…

- How can we define abstractions?
  - Define a **set of propositions** representing certain facts
  - **Valuation of propositions** instead of valuation of variables – finite model

- **Knowledge level abstraction**
  - Interpretation of the valuation: *set of states, where the "known" facts are true, "unknown" facts may be true or false*
  - Applicability of the transition: *the transition is applicable, if its condition evaluates to true in* ***all the states*** *of the interpretation*
  - The effect of the transition: *the resulting valuation is* ***the most conservative valuation*** *with respect to the set of facts that can be deduced*

Abstract model

- **Abstract Transition System** $\Sigma = \langle B, S, S_0, I, O, R \rangle$ where
  - *B* – finite set of propositions
  - *S* – set of pairs *<s,Val>*, where *Val* is a valuation of propositions
- transition relation



Analysis approach

- **Hybrid approach** both models are used
  - B-model: over-approximation, for universal properties
  - K-model: under-approximation, for existential properties

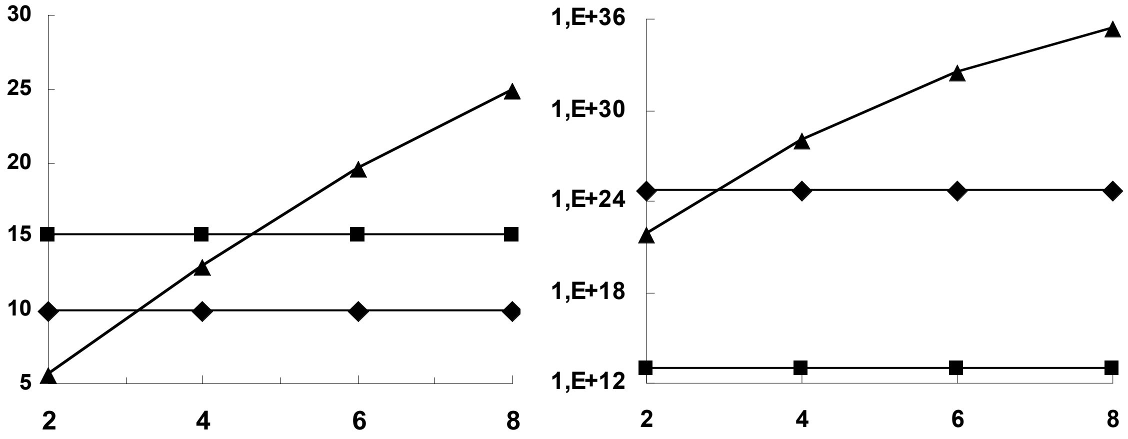
- For the **universal** property (assertion):
  - Prove that B-model satisfies assertion (return TRUE)
  - If not, prove that K-model violates assertion (return FALSE)
  - If not, refine (return UNKNOWN)

- For the **existential** property (possibility):
  - Prove that K-model satisfies possibility (return TRUE)
  - If not, prove that B-model violates possibility (return FALSE)
  - If not, refine (return UNKNOWN)

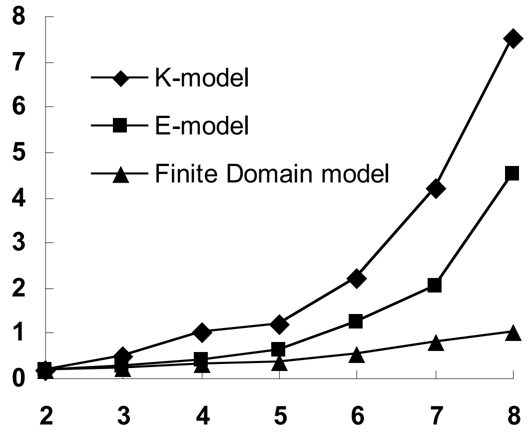
Implementation

- Preliminary implementation
  - Support for both models
  - Automated abstraction generation
  - Allows for the definition of assumptions on uninterpreted functions
  - NuSMV model checker verification

- Experiments



Finite ranges



Scalability

Outline

- Analysis of communication models
- Data-flow analysis
- **Analysis of time-related properties**
- Ongoing work and future directions

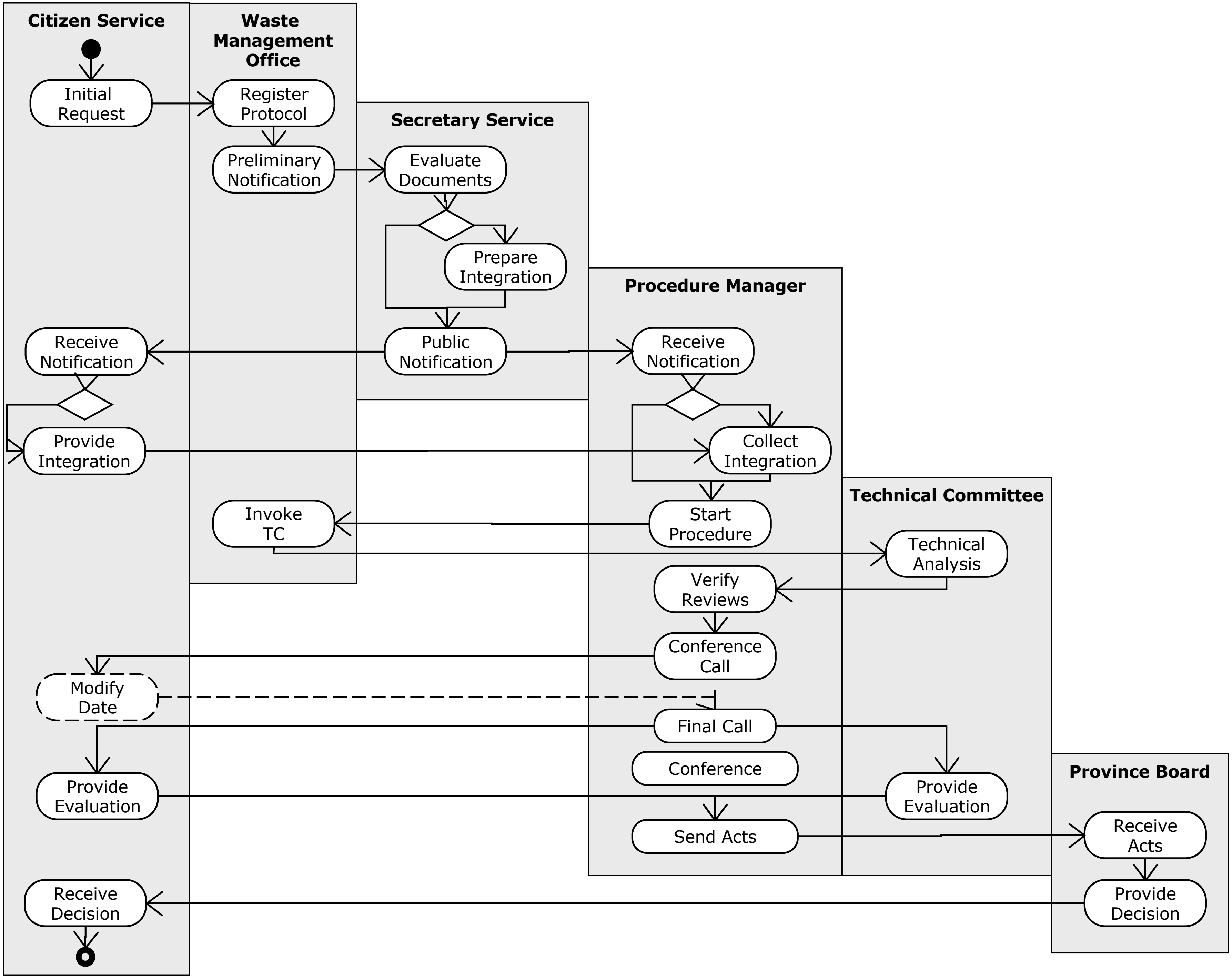
Problem #3: Time-related properties

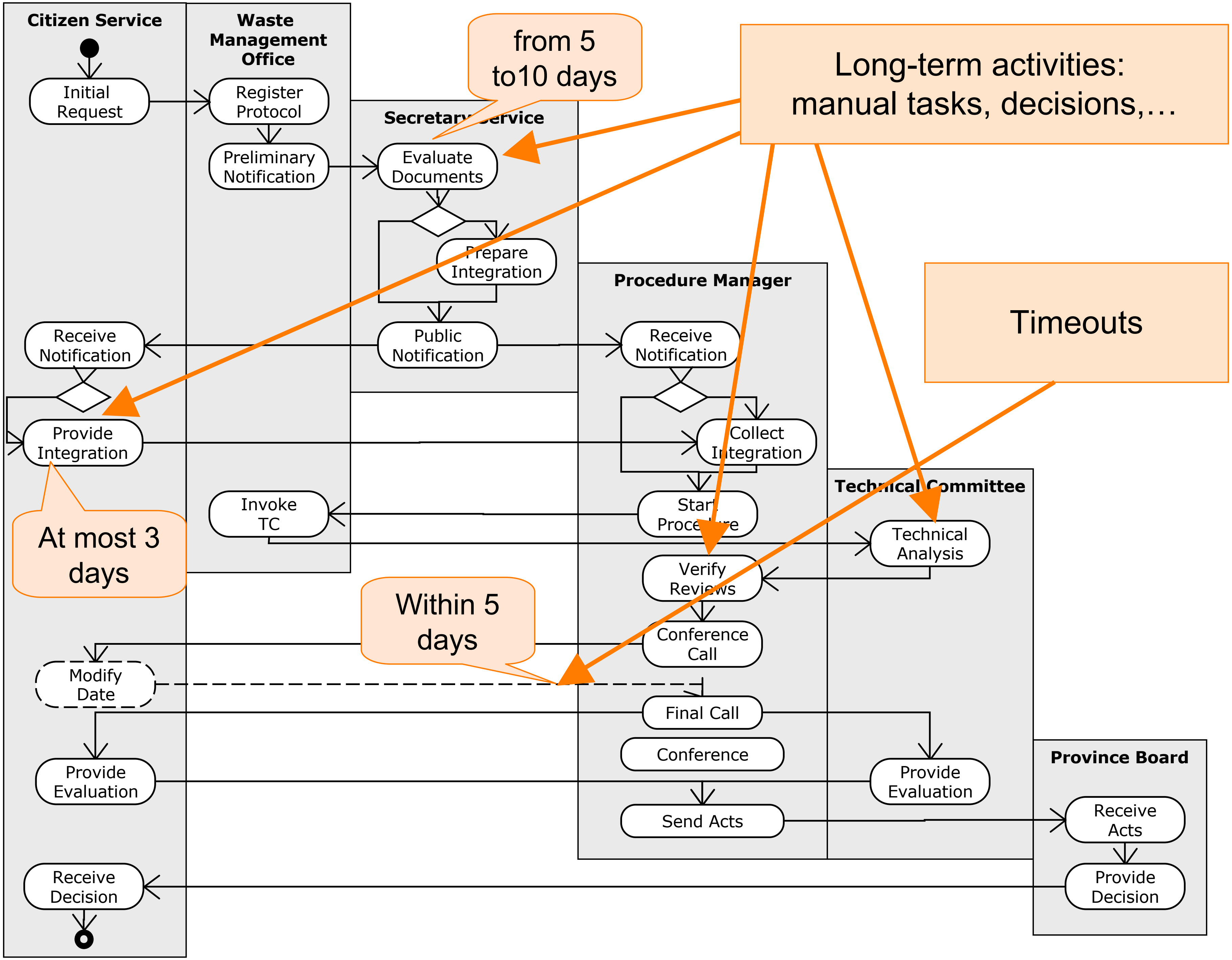
Timed behavior should be properly modeled and analyzed!

e-Government Application

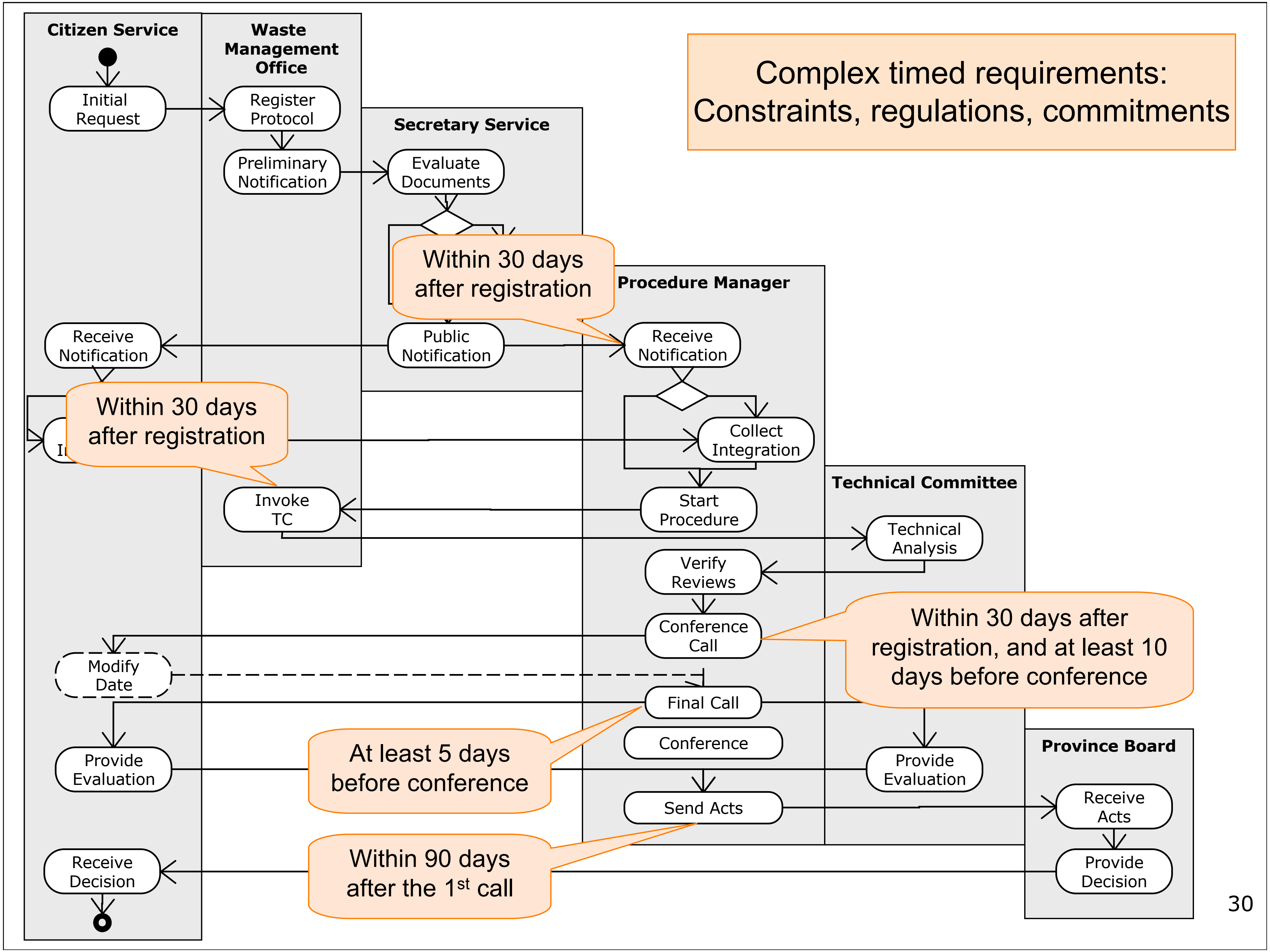
authorization for the establishment and operation of a waste disposal or recycling plant

- Settings
  - **complex distributed process** involving various actors
- Scenario
  - **long-term** process (~3 monthes) with time-consuming activities
- Requirements
  - Local (internal constraints) + global (state regulations, normative acts)
  - Functional + **timed**


Citizen Service
Waste Management Office
Secretary Service
Procedure Manager
Technical Committee
Province Board
Initial Request
Register Protocol
Preliminary Notification
Evaluate Documents
Prepare Integration
Public Notification
Receive Notification
Receive Notification
Provide Integration
Collect Integration
Start Procedure
Invoke TC
Technical Analysis
Verify Reviews
Conference Call
Modify Date
Final Call
Conference
Provide Evaluation
Provide Evaluation
Send Acts
Receive Acts
Provide Decision
Receive Decision



from 5 to10 days
Long-term activities: manual tasks, decisions,…
Timeouts
At most 3 days
Within 5 days
Citizen Service
Waste Management Office
Secretary Service
Procedure Manager
Technical Committee
Province Board
Initial Request
Register Protocol
Preliminary Notification
Evaluate Documents
Prepare Integration
Public Notification
Receive Notification
Provide Integration
Receive Notification
Collect Integration
Start Procedure
Invoke TC
Technical Analysis
Verify Reviews
Conference Call
Modify Date
Final Call
Conference
Provide Evaluation
Provide Evaluation
Send Acts
Receive Acts
Provide Decision
Receive Decision



Complex timed requirements:
Constraints, regulations, commitments
Citizen Service
Waste Management Office
Secretary Service
Procedure Manager
Technical Committee
Province Board
Initial Request
Register Protocol
Preliminary Notification
Evaluate Documents
Within 30 days after registration
Receive Notification
Public Notification
Receive Notification
Within 30 days after registration
Collect Integration
Start Procedure
Invoke TC
Technical Analysis
Verify Reviews
Conference Call
Within 30 days after registration, and at least 10 days before conference
Modify Date
Final Call
Conference
At least 5 days before conference
Provide Evaluation
Provide Evaluation
Send Acts
Receive Acts
Within 90 days after the 1st call
Receive Decision
Provide Decision

Problem #3: Time-related properties

Timed properties should be properly modeled and analyzed!

- Timed constructs of WS-* languages
  - BPEL ***onAlarm***, ***wait*** activities
- Time-specific requirements, constraints commitments
  - simple properties: activity **durations**
  - complex properties: constraints on intervals between events, activities, states

Our approach [Kazhamiakin,Pandya,Pistore ARES'06,ICWS'06]

- Extend the **formal model** of the composition with **time**
  - **TTS** model (close to timed automata network with urgency)
  - formalize BPEL timed constructs and activity durations annotations
  - formalize complex requirements: (subset of) **Duration Calculus**

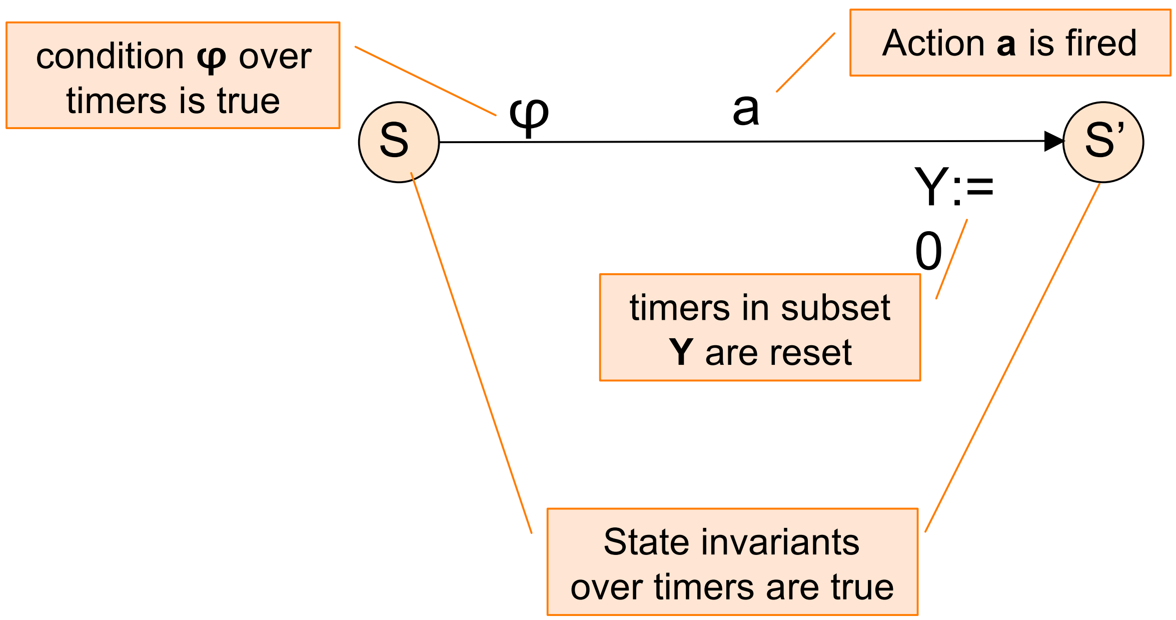
- Provide a set of **analysis techniques**
  - Verification of timeless properties on timed model
  - Verification of timed properties
  - Computation of timed properties

- Provide a formal **analysis framework**
  - Discrete model of time (finite timers, QDDC [Pandya, RTTOOLS'01])
  - NuSMV symbolic model checking

Our approach: formal definitions

- **Timed Transition System** $\Sigma = \langle \mathbf{X}, S, S_0, I, O, R, \mathbf{Inv} \rangle$ where
  - *X* – finite set of **timers**
  - *Inv* – function that associates **invariants** to states

- Transition relation

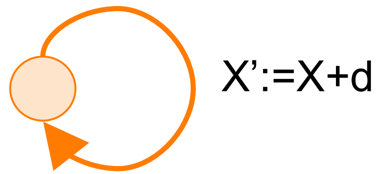


Our approach: formal definitions

- **Timed Transition System** $\Sigma = \langle \boldsymbol{X}, S, S_0, I, O, R, \boldsymbol{Inv} \rangle$ where
  - *X* – finite set of **timers**
  - *Inv* – function that associates **invariants** to states

- Semantics: composition behavior

  Time elapsing transition:
  - global state is not changed
  - all timers synchronously increment



  Action of some TTS:
  - transition of some TTS is executed
  - timers are not changes

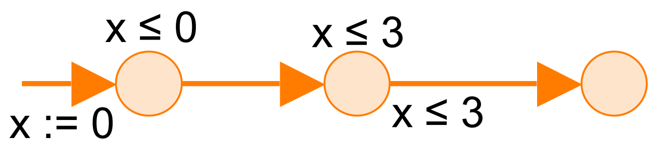
From BPEL to TTS

- Instant activities

<**invoke** operation=“**op**"/>

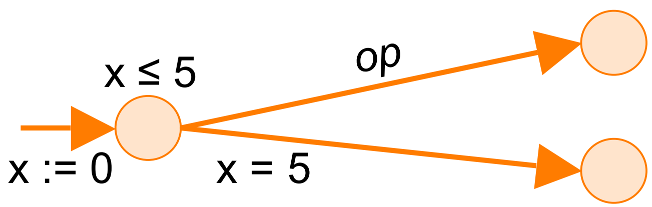
- Duration annotations

<**activity duration**=“**lessEqual(3D)**”/>



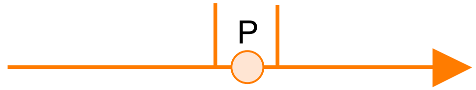
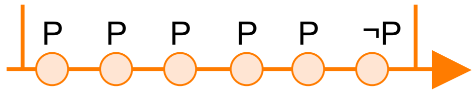
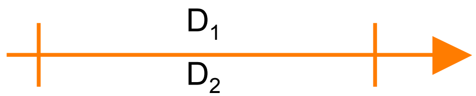
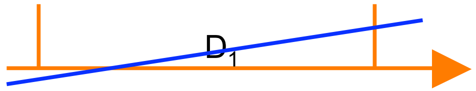
- BPEL timeout

<**pick**>
  <**onMessage** operation=“**op**”>…</>
  <**onAlarm for**=“PT**5D**”>…</>



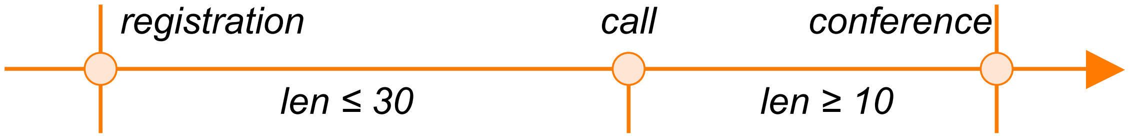
Interval Specifications

- Duration Calculus
  - Properties over intervals
  - Allows to express complex timed requirements of behavioral specifications

$[P]^0$	Single state satisfying propositional formula P	
$[[P]$	P is satisfied in all states of the behavior	
$D_1 \wedge D_2$	D_1 is satisfied in the 1st subinterval of behavior and D_2 in the 2nd	
D_1 AND D_2	Interval satisfies both formulae	
$\neg D$	Interval does not satisfy the formula	
len ~ c	The duration of interval is ~ c	

Interval Formula Example

The conference call should happen within 30 days after the registration and at least 10 days before the conference.



$$[]\ ($$

$$([\textit{registration}]^0 \wedge \textit{true} \wedge [\textit{conference}]^0) \rightarrow$$

$$(\ (\text{len} \leq 30) \wedge [\textit{call}]^0 \wedge (\text{len} \geq 10)\ )$$

$$)$$

Quantitative analysis

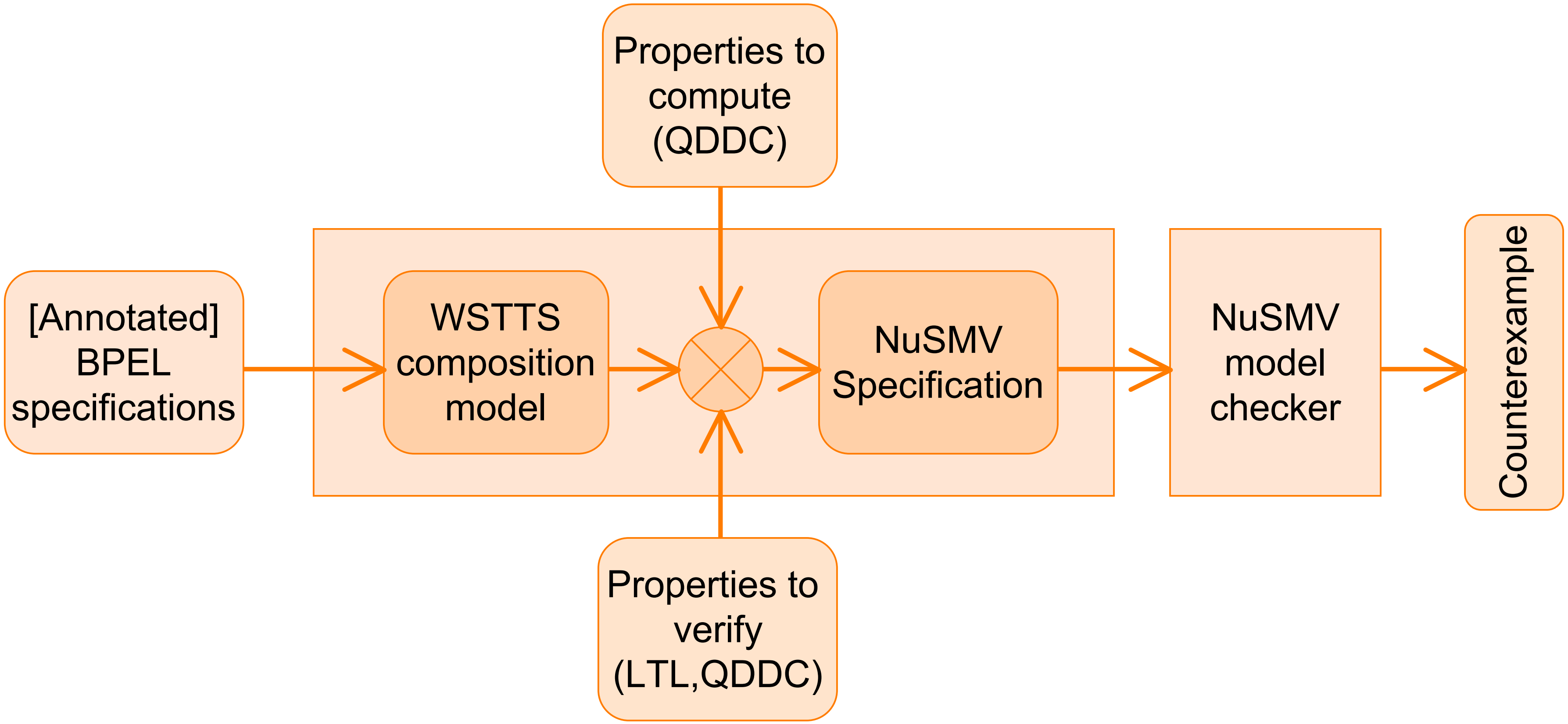
- **Extremal bounds** algorithm
  - Compute minimal/maximal bounds of the intervals, where the property holds
  - Asynchronous versions of the algorithms presented in [Pandya05,Campos et al.96]
  - Symbolic prototype implementation

- Often more effective than verification-based search

Property	**Time**	**States**
Assertion	5.56sec	2919
Max	0.28sec	1005
Possibility	2.28sec	2919
Min	0.32sec	1005

  - *Assertion*: Procedure always terminates within given period
  - *Possibility*: It is possible to receive a conference call within given period

Implementation



Outline

- Analysis of communication models
- Data-flow analysis
- Analysis of time-related properties
- **Ongoing work and future directions**

Ongoing and future work on…

Analysis of communication models

- Better **integration** with the **data flow** analysis
  - Currently: conservative analysis results on skeletons, additional verification on complete model
- The role of communication models in the **conformance testing**
  - Validation of BPEL compositions against WS-CDL specifications [WS-FM'06]
  - Realizability of choreography specifications [FORTE'06]

Ongoing and future work on…

Data-flow analysis

- Better **abstraction-based reasoning** techniques
  - Performance of the generation of K-model
  - Alternative encodings and verifiers
- **Counterexample analysis**
  - How can we extract the missing assumptions and constraints?
- Application to **run-time monitoring**

Ongoing and future work on…

Timed analysis

- Translation **optimizations** and better analysis techniques
  - State space clustering
  - Alternative encodings (mat-sat,…)
- **Alternative encoding**
  - E.g., UPPAAL model checking
- Again, application to **run-time monitoring**

Any ? questions