Modeling Families of Public Licensing Services: A Case Study

Guillermina Cledou, Luis Barbosa

HASLab INESCTEC and Universidade do Minho

FormaliSE2017
Outline

1 Part I
- Context and motivation
- Modeling families of services with FTA
- Verifying properties of families of services with Uppaal

2 Part II
- Analyzing FTA
- Enriching FTA
Service Delivery in E-Government

Challenges

- Rapid development
- Service integration
- Cost reduction
- Conformance with laws and regulations

In Practice

- Ad-hoc ICT solutions disregarding common functionality and shared processes
Service Delivery in E-Government

Challenges
- Rapid development
- Service integration
- Cost reduction
- Conformance with laws and regulations

In Practice
- Ad-hoc ICT solutions disregarding common functionality and shared processes
Public Licensing Services

Queue

Applicant

Authority

Pre-assess

Assess

Credit Card Paypal

submits

gets

incomplete/reject/accept

appeals*

pays*

conf/error*

Payment

Credit Card*

Paypal*

May 27, 2017 4 / 17
How can we address existing challenges?

Challenges
- Rapid development
- Service integration
- Cost reduction
- Conformance with laws and regulations

Software Product Lines
Formal Methods
A set of software systems that share a high number of features while differing on others, where concrete configurations are derived from a core of common assets in a prescribed way.

**Feature**
- A characteristic or behavior of a system that is visible to the user.
- e.g., pay, cc, pp, ...

**Feature model**
- Expresses valid feature combinations, i.e., the set of systems that can be derived from the SPL
- e.g., 

\[
\{ \{ \text{pay, cc, pp} \}, \{ \text{pay, cc} \}, \{ \text{pay, pp} \}, \{} \}
\]
A set of software systems that share a high number of features while differing on others, where concrete configurations are derived from a core of common assets in a prescribed way.

**Feature**
- A characteristic or behavior of a system that is visible to the user.
- e.g., `pay`, `cc`, `pp`, ...

**Feature model**
- Expresses valid feature combinations, i.e., the set of systems that can be derived from the SPL.
- e.g., `{{pay, cc, pp}, {pay, cc}, {pay, pp}, {}}`
A set of software systems that share a high number of features while differing on others, where concrete configurations are derived from a core of common assets in a prescribed way.

**Feature**
- A characteristic or behavior of a system that is visible to the user.
- e.g., pay, cc, pp, ...

**Feature model**
- Expresses valid feature combinations, i.e., the set of systems that can be derived from the SPL
- e.g., \{\{pay, cc, pp\}, \{pay, cc\}, \{pay, pp\},\}
A Modeling formalism for SPL

Feature Timed Automata (FTA)
- Extends Timed Automata with variability
- Enables the verification of the entire SPL by capturing its behavior in a single model

Diagram:
- Applicant
- Queue
- Authority
- Payment
- Credit Card
- Paypal
- submits
- gets
- incomplete/reject/accept
- appeals
-

Mathematical expressions:

\[
\ell_0 \quad \ell_1
\]

- \( tpp \leq 1 \)
- \( pp \)
- \( paypp, tpp := 0 \)
- \( paidpp, pp \)
- \( fm = \top \)
- Real-time model checker
- Used by academics and industry

![Uppaal Diagram]

**Fig. 2:** UPPAAL automata with FTA variability modeling domain functionality.

**Table I:** Examples of verifiable temporal properties in Uppaal (using Uppaal’s syntax).

<table>
<thead>
<tr>
<th>Property</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liveness</td>
<td>ap0.apply --&gt; (ap0.accepted</td>
</tr>
<tr>
<td>Reachability</td>
<td>!cc --&gt; !(exists(i:app_id) (CreditCard(i).L1</td>
</tr>
<tr>
<td>Safety</td>
<td>A[[]] ap0.submitted imply ap0.tproc &lt;= 90+31</td>
</tr>
<tr>
<td></td>
<td>A[[]] forall(i:app_id) !(auth0.inOpenApps(i) &amp;&amp; auth1.inOpenApps(i))</td>
</tr>
</tbody>
</table>

In alignment with the only-once principle mentioned in Section II, a typical scenario is the ability to consult relevant agencies whether a given applicant possesses a criminal record or has all tax duties in order. We model such external sources as FTA in Figure 4 (left). The model on the top left represents an external database that receives a request to check a given tax number (checkTax) and provides a response certifying whether tax duties are in order (resTax), while the model on the bottom left represents an external database that receives a request to check a person ID (checkCR) and provides a response certifying whether the person has criminal records (resCR). Their presence depends on features tx and cr, respectively.

In this scenario, the authority must consult the required external sources, if available, and wait for their responses before deciding to grant the license. Figure 4 (right) shows such an FTA, which is a simplification of automata Processing from Figure 2. When an application is ready to be assessed (assess), the authority consults external sources (checkES) if supported by the service, waits until all responses are ready (results), and makes a decision. In case the service does not support consultation of external sources the authority can directly make a decision.

The complication arise when modeling the interaction between the new FTA Assess and the external databases. First,
Modeling variability in Uppaal
**Example properties**

- An application eventually results in accepted, rejected, incomplete, or canceled
- An application is processed within 121 days
- An application can not be opened by more than one authority
- ...

<table>
<thead>
<tr>
<th>Property</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liveness</td>
<td>ap0.apply --&gt; (ap0.accepted</td>
</tr>
<tr>
<td>Reachability</td>
<td>!cc --&gt; !(exists(i:app_id) (CreditCard(i).L1</td>
</tr>
</tbody>
</table>
| Safety | A[ ] ap0.submitted imply ap0.tproc <= 90+31  
A[ ] ap3.appealed imply ap3.tapl <=60  
A[ ] forall(i:app_id) !(auth0.inOpenApps(i) && auth1.inOpenApps(i)) |
Part I
- Context and motivation
- Modeling families of services with FTA
- Verifying properties of families of services with Uppaal

Part II
- Analyzing FTA
- Enriching FTA
Analyzing FTA

Feature Timed Automata (FTA)

- Disregards modular and compositional aspects of SPL development
- Implicit communication points
- Lack of variability composition
- Lack of reusable common orchestration mechanisms
Extending FTA

Interface Featured Timed Automata (IFTA)

- Extends FTA with interfaces that restrict the way automata are composed
- Multi-action transitions to simplify design

\[
\ell_0 \quad \ell_1
\]

\[
f_1 \land f_{o_1}, f_2 \land f_{o_2}
\]

\[
f_i \land (f_{o_1} \lor f_{o_2})
\]

\[
f_i \land f_{o_1}, f_i \land f_{o_2}
\]

\[
f_i \land (f_{o_1} \lor f_{o_2}) \leftrightarrow f_i
\]

\[
?!, ! \text{ denote inputs and outputs interfaces, respectively.}
\]

\[
each \text{interface has associated an inferred feature expression.}
\]
Extending FTA

Interface Featured Timed Automata (IFTA)
- Explicit communication points + composition of variability

\[ \text{paypp} \] 
\[ \text{cancelpp} \] 
\[ \text{okpp} \] 

\[ \text{fm} = (f_{o_1} \lor f_{o_2}) \leftrightarrow f_i \] 

\[ \text{Router} \Join \text{PayPal} \]

\[ \text{fm} = f_i \leftrightarrow (f_{o_1} \lor f_{o_2}) \land (pp \leftrightarrow (f_i \land f_{o_1})) \]
Scala DSL: https://github.com/haslab/ifta

- Specification of IFTA
- Uppaal
- Interactive representation
- Dot
### Conclusions

#### E-Government
- Unexplored domain with respect to SPL + Formal methods

#### FTA
- Allows to simplify the modeling and verification of families of timed automata
- Can be enriched to reason about variability during composition

#### IFTA
- *Multi-action transitions* simplify design
- *Interfaces* enables reasoning about variability + visual feedback
- *Composition* takes into account the feature models
- Limitations in the implementation
  - Uppaal doesn’t work very well with sequence of committed states
  - Size of IFTA composition can growth quickly

---

**Questions?**
Conclusions

E-Government
- Unexplored domain with respect to SPL + Formal methods

FTA
- Allows to simplify the modeling and verification of families of timed automata
- Can be enriched to reason about variability during composition

IFTA
- Multi-action transitions simplify design
- Interfaces enables reasoning about variability + visual feedback
- Composition takes into account the feature models
- Limitations in the implementation
  - Uppaal doesn’t work very well with sequence of committed states
  - Size of IFTA composition can growth quickly

Questions?