Checking the Realizability of BPMN 2.0 Choreographies

Gwen Salaün
Grenoble INP, INRIA, France

joint work with
Pascal Poizat
LRI, University of Evry, France
Realizability of Choreographies

- Interactions among a set of services involved in a new system can be described from a global point of view using choreography specification languages.
- Given a choreography specification, local implementations, namely peers, can be automatically generated via projection.
- However, peers do not always implement the choreography: this problem is known as realizability.
Contributions

- We propose an encoding of BPMN 2.0 choreographies into the LNT specification language.

- We chose LNT because:
  - It provides a good level of expressiveness for describing BPMN constructs.
  - It is equipped with CADP which offers state-of-the-art tools for state space exploration and verification.

- This encoding allows us to:
  - Automate service peer generation.
  - Verify choreography specifications using CADP.
  - Check the realizability for both synchronous and asynchronous communication.
Outline

1. Preliminaries: BPMN 2.0, LNT, and CADP
2. Encoding into LNT
3. Verification and Realizability
4. Tool Support
5. Concluding Remarks
BPMN 2.0 Choreographies

- Choreography tasks and loop types

- Control flows and gateways

- Sequence flow
- Start state
- End state
- Exclusive gateway
- Inclusive gateway
- Parallel gateway
- Event-based gateway

- Diverging pattern (diverging parallel gateway)
- Converging pattern (converging parallel gateway)
Running Example

An e-booking system involving four peers: a booking system (bs), a database (db), an online bank service (bk), and a client (cl)

Peers are described using Labelled Transition Systems (LTSs)
LNT

- LOTOS NT (LNT) is a value-passing process algebra with user-friendly syntax and operational semantics

- LNT is an imperative-like language where you can specify data types, functions (pattern matching and recursion), and processes

- Excerpt of the LNT process grammar:

```plaintext
B ::= stop | G(!E, ?X) where E' | if E then B1 else B2 end if 
    | x:=E | hide G in B end hide | P [G1,...,Gm] (E1,...,En) 
    | select B1 [] ... [] Bn end select | B1 ; B2 
    | par G in B1 || ... || Bn end par
```

- Verification using CADP through an automated translation to LOTOS
Construction and Analysis of Distributed Processes (CADP)

- Design of asynchronous systems
  - Concurrent processes
  - Message-passing communication
  - Nondeterministic behaviours

- Formal approach rooted in concurrency theory: process calculi, Labeled Transition Systems, bisimulations, branching temporal logics

- Many verification techniques: simulation, model and equivalence checking, compositional verification, test case generation, performance evaluation, etc.

- Numerous real-world applications: avionics, embedded systems, hardware design, middleware and software architectures, etc.
Outline

1. Preliminaries: BPMN 2.0, LNT, and CADP
2. Encoding into LNT
3. Verification and Realizability
4. Tool Support
5. Concluding Remarks
Encoding BPMN into LNT (1/3)

- Translation of BPMN via state machines

  - Sequence flow

  - Message sending

  - Message receiving
    - Synchronous communication
    - Asynchronous communication via FIFO message buffers
**Encoding BPMN into LNT (2/3)**

- Exclusive gateway

```
select
  option_A[...]
[] option_B[...]
[] option_C[...]
end select
```

- Parallel gateway

```
par
  option_A[...]
|| option_B[...]
|| option_C[...]
end par
```

- Parallel merge (multiple merges)

```
hide sync1, sync2 in
par
  sync1, sync2 -> option_A[...],sync1,sync2]
|| sync1, sync2 -> option_B[...],sync1,sync2]
|| sync2 -> option_C[...],sync2]
end par
```
Encoding BPMN into LNT (3/3)

- Inclusive gateway

Inclusive Merging:
  - Analogous to parallel merge
  - Default case needs no synchronization

```plaintext
select
    option_A[...] || ((option_B[...][] null) || (option_C[...][] null))
[[]]option_B[...] || ((option_A[...][] null) || (option_C[...][] null))
[[]]option_C[...] || ((option_A[...][] null) || (option_B[...][] null))
[[]]default[...]
end select
```
Outline

1. Preliminaries: BPMN 2.0, LNT, and CADP
2. Encoding into LNT
3. Verification and Realizability
4. Tool Support
5. Concluding Remarks
Compilation and Verification

- LTS models can be generated using CADP exploration tools, and verified using the Evaluator model-checker.
- E-booking system: LTS obtained by hiding “sync_” messages, and minimizing the resulting LTS.

We can check that a client can make a booking or abort only if a request has been issued (safety property):

\[
[ (\text{not 'CL\_BS\_REQUEST'})^* . \left( \text{'CL\_BS\_BOOK'} \text{ or 'CL\_BS\_ABORT'} \right) ] \text{ false}
\]
Realizability Checking

- Realizability is computed by comparing the BPMN LTS with the system composed of interacting peers using behavioural equivalences.
- If these two systems are equivalent, the choreography is realizable.
- In case of asynchronous communication, we generate LNT code to implement bounded FIFO buffers, and associate a buffer to each peer.

For asynchronous communication, undecidability is avoided by imposing buffer bounds or by using recent synchronizability results [BasuBultan-WWW11].
Our running example is not realizable for both communication models (synchronous and asynchronous).

The trace consisting of messages `cl_bs_connect`, `cl_bs_request`, `bs_cl_reply`, `cl_bs_book` appears in both systems, but `bs_db_store` is then in the distributed system, and not in the choreography LTS.
E-booking System, Revisited

We use a diverging parallel gateway instead and realizability checks return positive results for both communication models.
Outline

1. Preliminaries: BPMN 2.0, LNT, and CADP
2. Encoding into LNT
3. Verification and Realizability
4. Tool Support
5. Concluding Remarks
Tool Support

Eclipse IDE (Helios + BPMN2 Editor)

BPMN2 Choreography

Encoding
- BPMN2Py
- Py2LNT

LOTOS NT Processes

Model Retrieval
- Caesar
- Reductor

Peer Models (LTS)

Intended Choreography Model (LTS)

Verification & Realizability Results

Verification & Realizability
- Evaluator
- Bisimulator
Outline

1. Preliminaries: BPMN 2.0, LNT, and CADP
2. Encoding into LNT
3. Verification and Realizability
4. Tool Support
5. Concluding Remarks
Concluding Remarks

- We have presented an encoding of BPMN 2.0 choreographies into LNT, which makes the formal analysis of BPMN possible using CADP verification tools.

- As far as perspectives are concerned, we would like to:
  - Extend the subset of BPMN choreographies accepted by our approach with hierarchical structuring aspects (sub-choreography).
  - Integrate looser realizability notions to our framework (pre-order, partial order, etc.).
  - Use recent compositional aggregation techniques [CrouzenLang-FASE11] to reduce intermediate state spaces size and computation times.
  - Enforce realizability proposing smart projection techniques.
  - Apply our approach to a real-size case study in the e-governance domain.