PIC2LNT: Model Transformation for Model Checking an Applied Pi-Calculus

Radu Mateescu and Gwen Salaün

Inria Grenoble – Rhône-Alpes and LIG / Convecs

http://convecs.inria.fr

Motivation

- Pi-calculus [Milner-Parrow-Walker-92]
  - Formalism for describing concurrent mobile processes
  - Various extensions proposed in two decades
- Difficult to provide and maintain analysis tools
  - Mobility Workbench (MWB) [Victor-Moller-94]
- Alternative solution
  - Translation to a value-passing process algebra (LNT)
  - Use of analysis tools for concurrent systems (CADP)

➔ *easy extension to an applied pi-calculus by adding LNT data types and functions*

PIC: an applied pi-calculus
(syntax of behaviour terms)

B ::= **nil**	*empty*
| $P\ (E_1, \ldots, E_n)$	*agent call*
| **tau** . B	*silent prefix*
| $'C <E_1, \ldots, E_n> . B$	*emission*
| $C\ (X_1{:}T_1, \ldots, X_n{:}T_n) . B$	*reception*
| $[\ E\ ]\ B$	*guard*
| $!\ k\ B$	*bounded replication*
| (**new** $C_1, \ldots, C_n$) B	*channel creation*
| **var** $X_1{:}T_1{:=}E_1, \ldots, X_n{:}T_n{:=}E_n$ **in** B **end var**	*variable definition*
| $B_1 + B_2$	*choice*
| $B_1 \mid B_2$	*parallel composition*

Restrictions
(for finite-state verification)

- Finite control fragment [Dam-94]
  - No recursion through parallel composition

    $P = Q \mid P$ 
- Bounded replication
  - No "bang"

    $!P$ 
- Recursion through channel creation allowed
  - But beware of infinite state space

    $P = (\textbf{new}\ c)\ 'a\ \langle c \rangle\ .\ P$

Pi-calculus vs. LNT

[Mateescu-Salaun-10]

Differences

Binary rendez-vous

Unidirectional communication

Mobile channels

Dynamic creation of processes

Names only

Action prefix

Multi-way rendez-vous

Bidirectional communication

Static channels

Static network of processes

Constructed datatypes

Symmetric sequential compo.

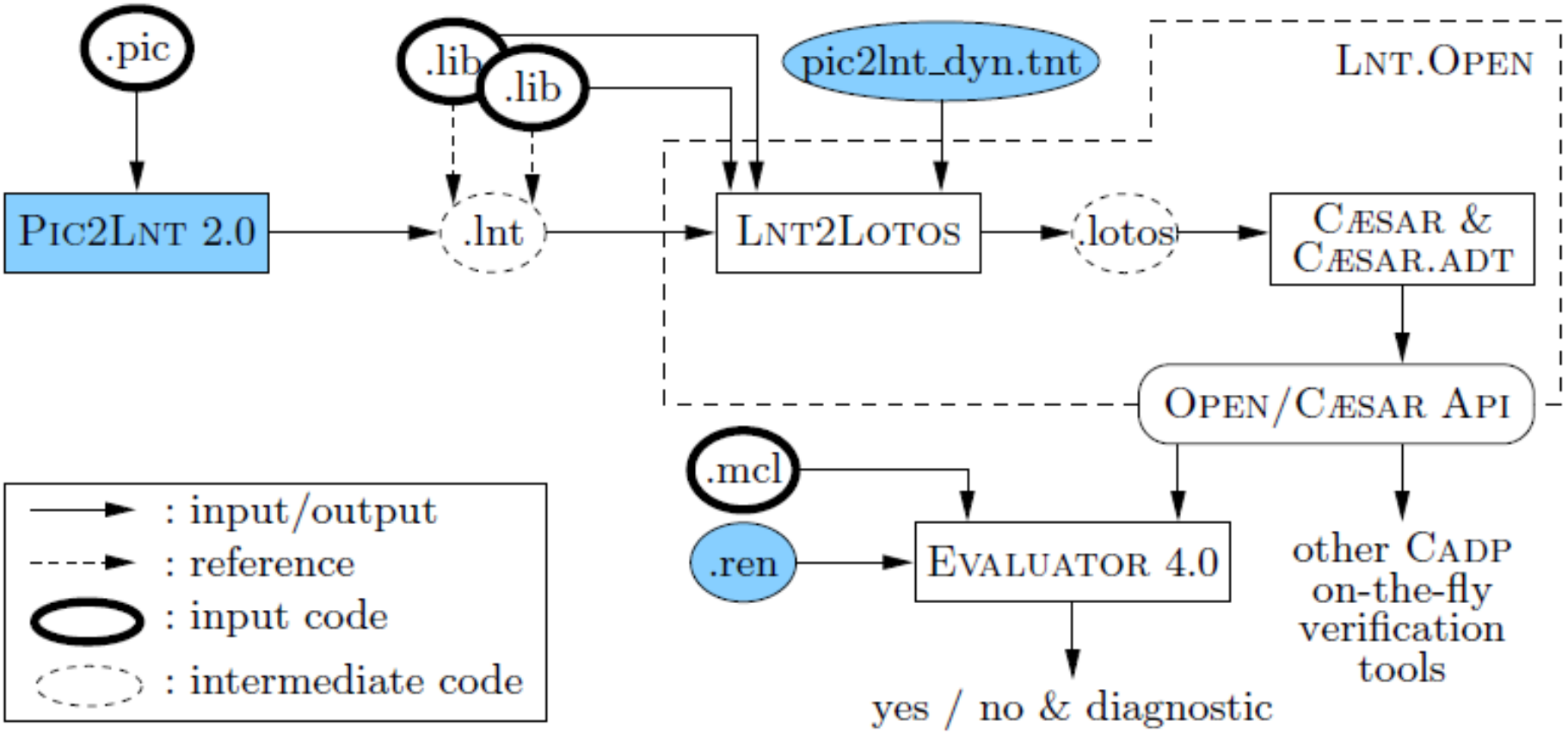
Similarities

Choice, recursion

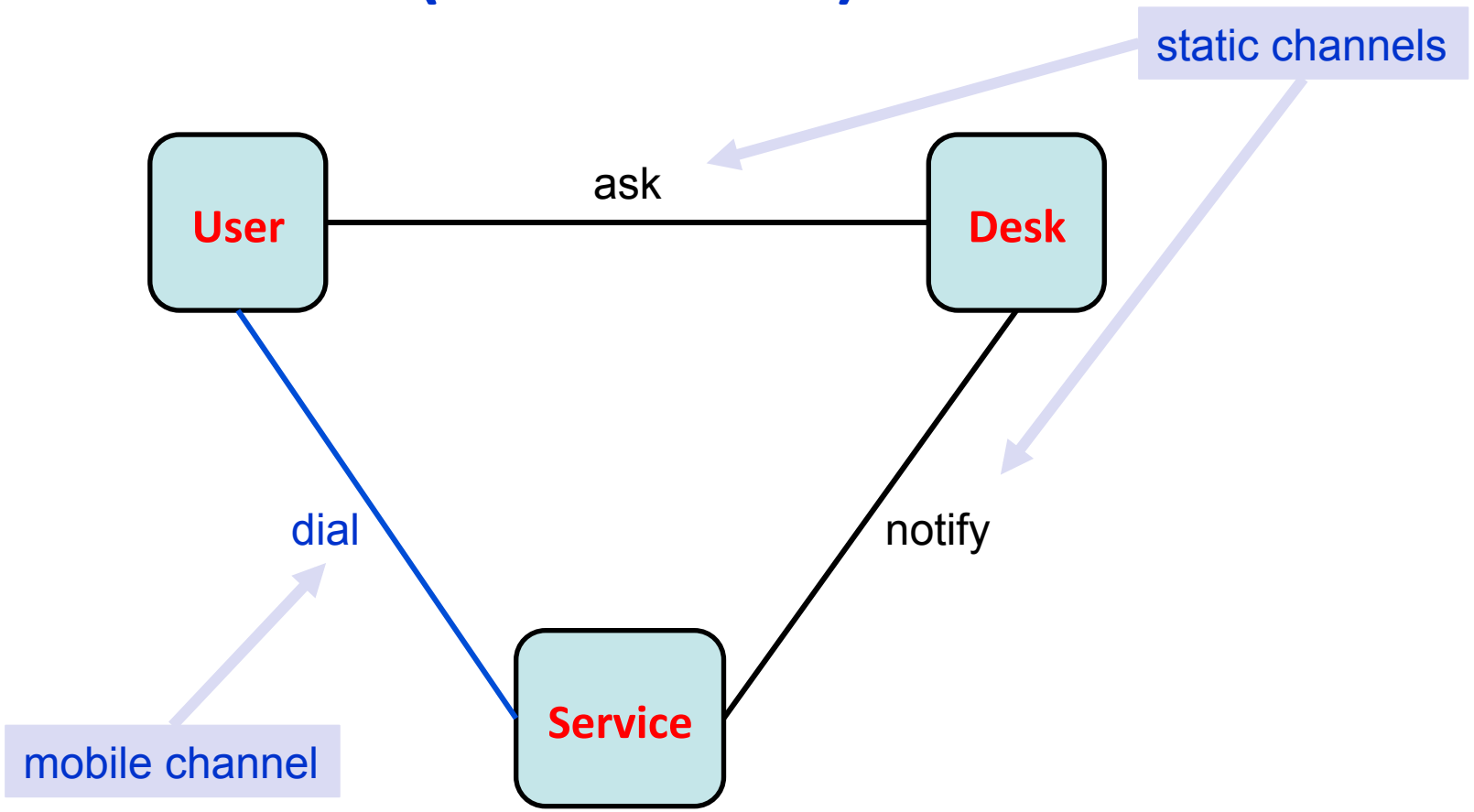
Binary parallel composition

Translation from PIC to LNT

PIC2LNT 2.0



Running example
(architecture)



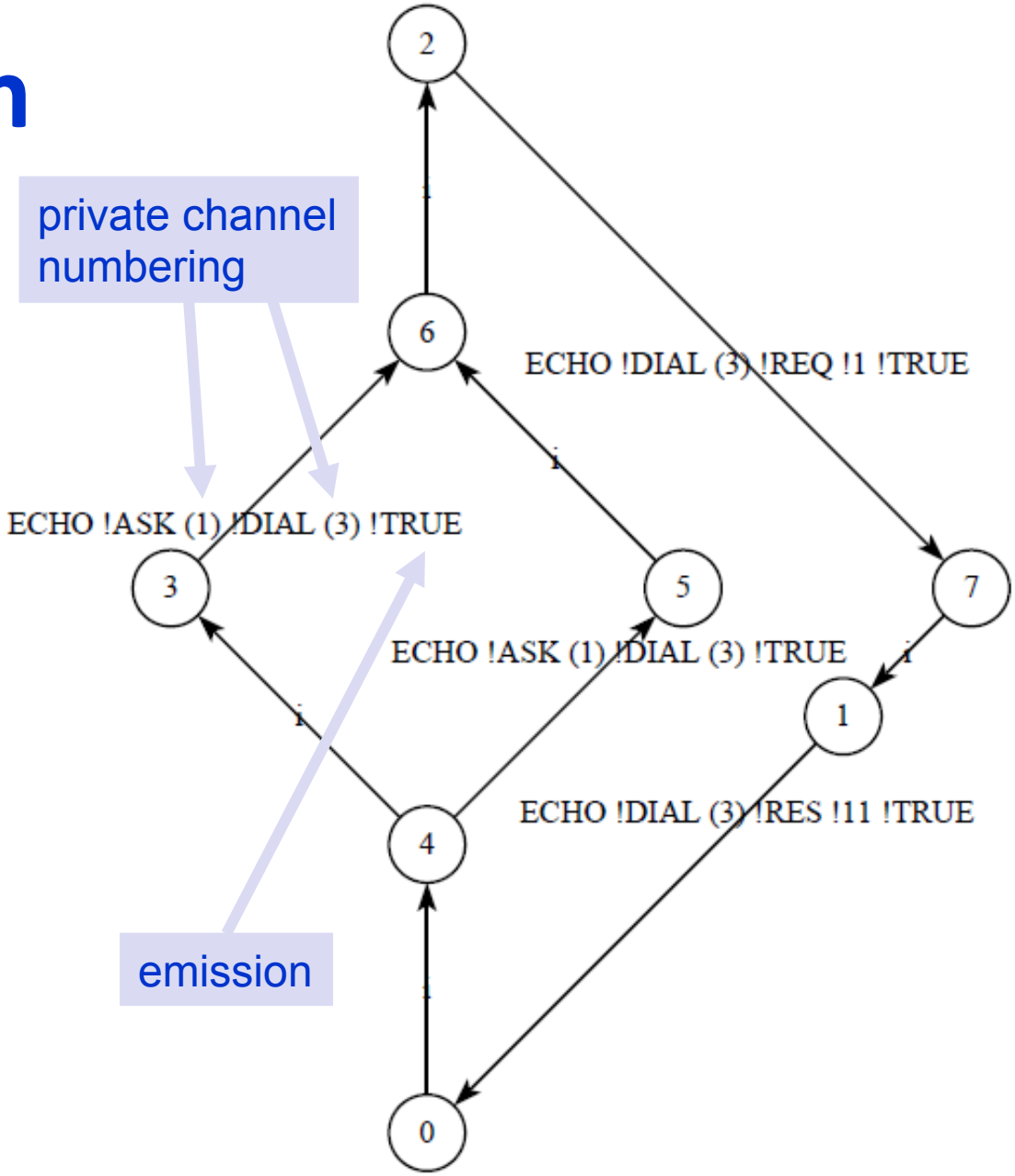
Running example
(PIC code)

```
Main =
    (new ask, notify) (User (ask) | Desk (ask, notify) | Service (notify))
User (ask) =
    ask (c) . 'echo <ask, c> . 'c <1 of Nat> . 'echo <c, req, 1 of Nat> .
    c (r:Nat) . 'echo <c, res, r> . User (ask)
Service (notify) =
    notify (c) . c (n:Nat) . 'c <n + 10> . Service (notify)
Desk (ask, notify) =
    (new dial) Desk_2 (ask, notify, dial)
Desk_2 (ask, notify, c) =
    'ask <c> . 'notify <c> . Desk_2 (ask, notify, c)
```

LTS construction

PIC2BCG:

- Generate the LTS (in BCG format) of a PIC specification
- Rename the labels in pi-calculus style



Verification

- **Evaluator 4.0**:
  - On-the-fly model checking
  - MCL properties (data-based modal mu-calculus)
  - Label hiding and renaming
- Safety property:

  [(not {ASK !"DIAL"})*. {DIAL !"REQ" ?n:Nat}] false
- Liveness property:

  [true*. {DIAL !"REQ" ?n:Nat}] < {DIAL !"RES" !n+10} > true

Conclusion

- PIC: an applied pi-calculus (+ LNT data types)
  - Syntax and semantics
  - PIC2LNT 2.0 translator ➔ connection to CADP
  - Used for teaching concurrency (Saarland University)
- Future work:
  - Application domains
    - Systems biology
    - Cryptographic protocols
    - Cloud computing