Specifying and Verifying the SYNERGY Reconfiguration Protocol with LOTOS-NT and CADP

Gwen Salaün

Grenoble INP, INRIA, France

joint work with

Fabienne Boyer and Olivier Gruber

UJF-Grenoble 1, INRIA, France

Introduction

- SYNERGY is an ongoing research program on reconfigurable and robust component-based virtual machines

- We focus here on a part of this system, the reconfiguration protocol, which aims at reconfiguring an architecture while supporting failures (possibly several ones)

- Reconfiguration capabilities guarantee the semantic and architectural consistency of the reconfigured software

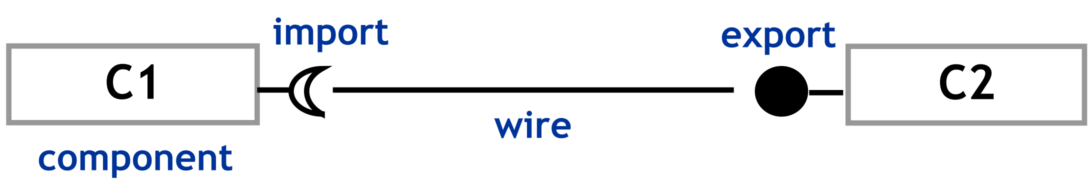
- We specify this protocol in LOTOS-NT, and verify it using CADP verification tools

Outline

- The reconfiguration protocol
- LOTOS-NT and CADP
- Specification in LOTOS-NT
- Verification using CADP
- Concluding remarks

Architecture of Components

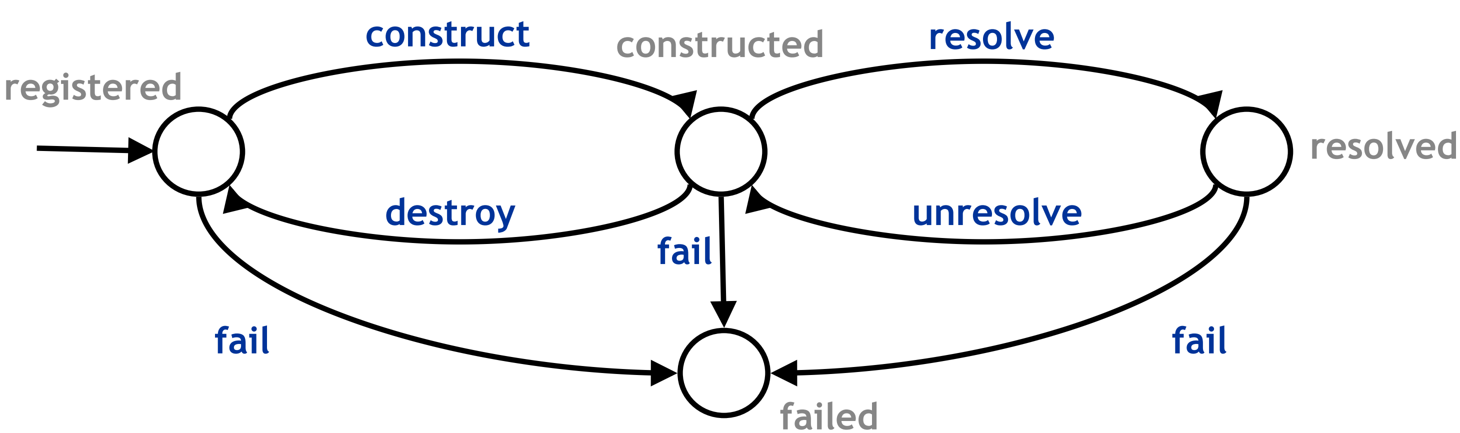
- An architecture consists of a set of components and a set of wires connecting these components together
- A component is composed of input and output ports, namely imports and exports
- A wire connects an import of one component to an export of another component



- An import exhibits several characteristics: tightly-coupled, mandatory, and vital

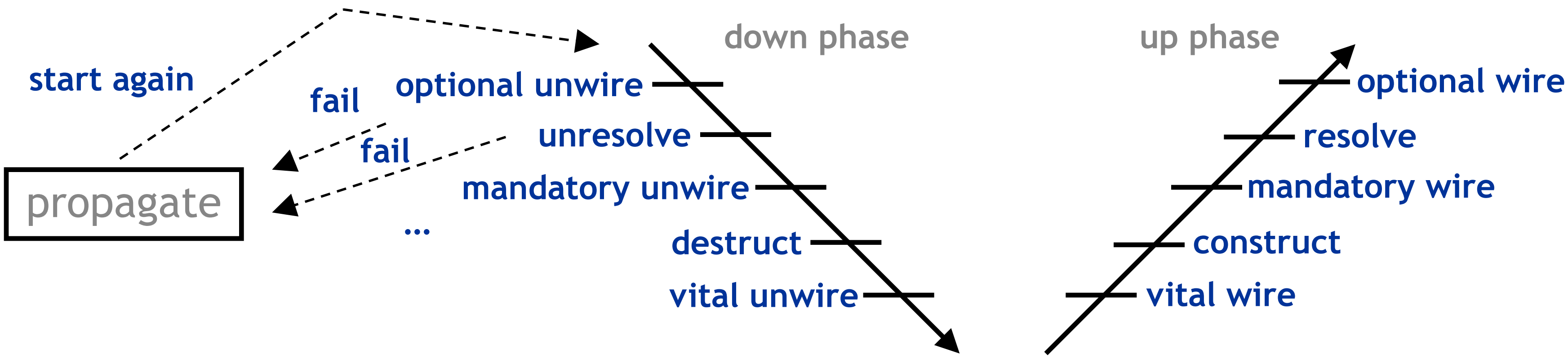
Component Life Cycle

- During its lifetime, a component can go through different states:
  - Resolved: all vital and mandatory imports wired to resolved compo.
  - Constructed: all vital imports wired to resolved components
  - Registered: missing wires on vital imports
- At any moment a component can fail, and in such a case it moves to a failed state



The Reconfiguration Protocol

- This protocol applies a set of reconfigurations (10 phases) on a source architecture to reach a target architecture



- If a component fails, the 10 phases are interrupted and both architectures modified to take this failure into account (propagate)
- Once the propagate is done, the 10 phases start again
- The process stops when the source and target architectures are identical

Outline

- The reconfiguration protocol
- LOTOS-NT and CADP
- Specification in LOTOS-NT
- Verification using CADP
- Concluding remarks

LOTOS-NT

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

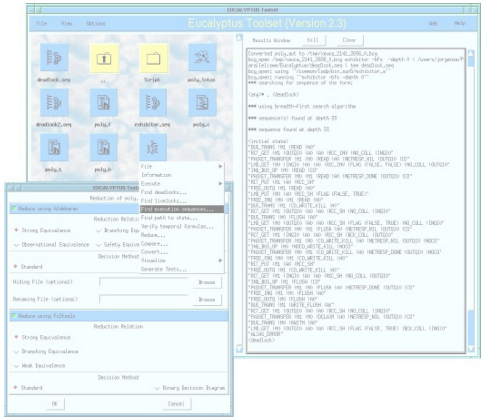
- The specification language consists of two parts:
  - A functional language to describe data types
  - An imperative-like formalism to specify processes

- Grammar of the behavioural LOTOS-NT fragment we use:

B	::=	**stop** | G(!E, ?X) **where** E'	| **if** E **then** B **end if**
	|	**var** x:T **in** x:=E ; B **end var**	| **hide** G **in** B **end hide**
	|	P [G1,..,Gm] (E1,..,En)	| **select** B1 [] … [] Bn **end select**
	|	**par** G **in** B1 || … || Bn **end par**	

- Verification using CADP through a translation to LOTOS

Construction and Analysis of Distributed Processes (CADP)



- Design of asynchronous systems
  - Concurrent processes
  - Message-passing communication
  - Nondeterminism
- Formal approach rooted in concurrency theory: process calculi, Labeled Transition Systems, temporal logics
- Many verification techniques: simulation, model and equivalence-checking, compositional verification, test case generation, performance evaluation, etc
- Numerous practical applications, *e.g.*, telecommunications, middleware and software architectures, hardware

Outline

- The reconfiguration protocol
- LOTOS-NT and CADP
- Specification in LOTOS-NT
- Verification using CADP
- Concluding remarks

Specification in LOTOS-NT (1/2)

- The specification consists of three parts: data types (300 lines), functions (2500 lines), processes (900 lines)
- Data types describe the architecture (components and wires), and functions handle these data types
- Example: removing some wires given a component identifier

```
function disconnect_wires (cid: TID, wires: TWires): TWires is
      case wires in
      var w: TWire, l: TWires in
        nil          -> return nil
      | cons(w,l)    -> if (w.cimport==cid) or (w.cexport==cid) then
                            return disconnect_wires(cid,l)
                        else
                            return cons(w,disconnect_wires(cid,l)
                        end if
      end case
end function
```

Specification in LOTOS-NT (2/2)

- Processes specify the behaviour of the whole specification, *i.e.*, 10 phases, failure propagation, etc.

- Each phase is specified as a process, and another process simulates failures

- Labels in transition systems generated from this specification correspond to component operations (*e.g.*, wire, resolve, etc.), failures, and some information we need for verification purposes

- This specification was revised several times due to several issues found in the protocol

Main Process of the LOTOS-NT Specification

```
process MAIN [UNRESOLVE:any, UNWIRE:any, REMOVEIMPORT:any,
  REMOVEEXPORT:any, …, FAILURE:any, START:any, PROPAGATE:any,
  CHECKINVARIANTS:any, REVOKEWIRE:any, VERIFWIRE:any,…]
     var source,target: TArchitecture in
          source := archi_source();
          target := archi_target();

          par FAILURE in
               p10 [UNRESOLVE,UNWIRE,…] (source,target)
               ||
               pfailure [FAILURE]
          end par

     end var
end process
```

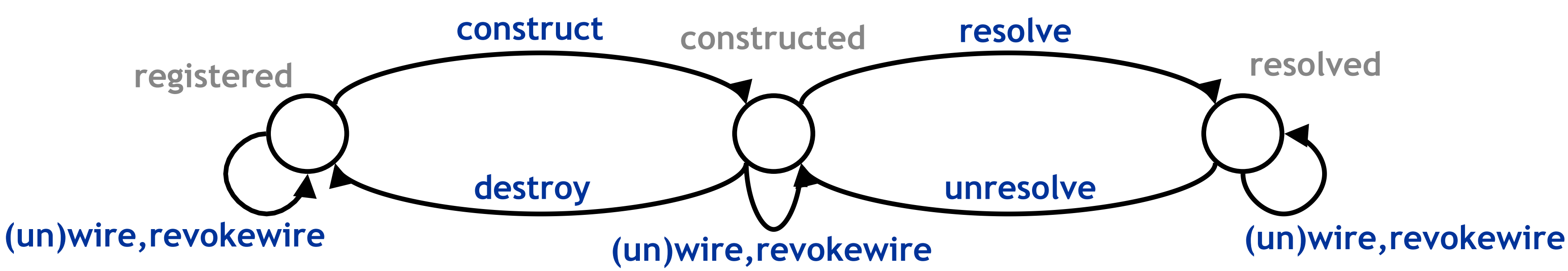
Outline

- The reconfiguration protocol
- LOTOS-NT and CADP
- Specification in LOTOS-NT
- Verification using CADP
- Concluding remarks

What Needs to be Checked?

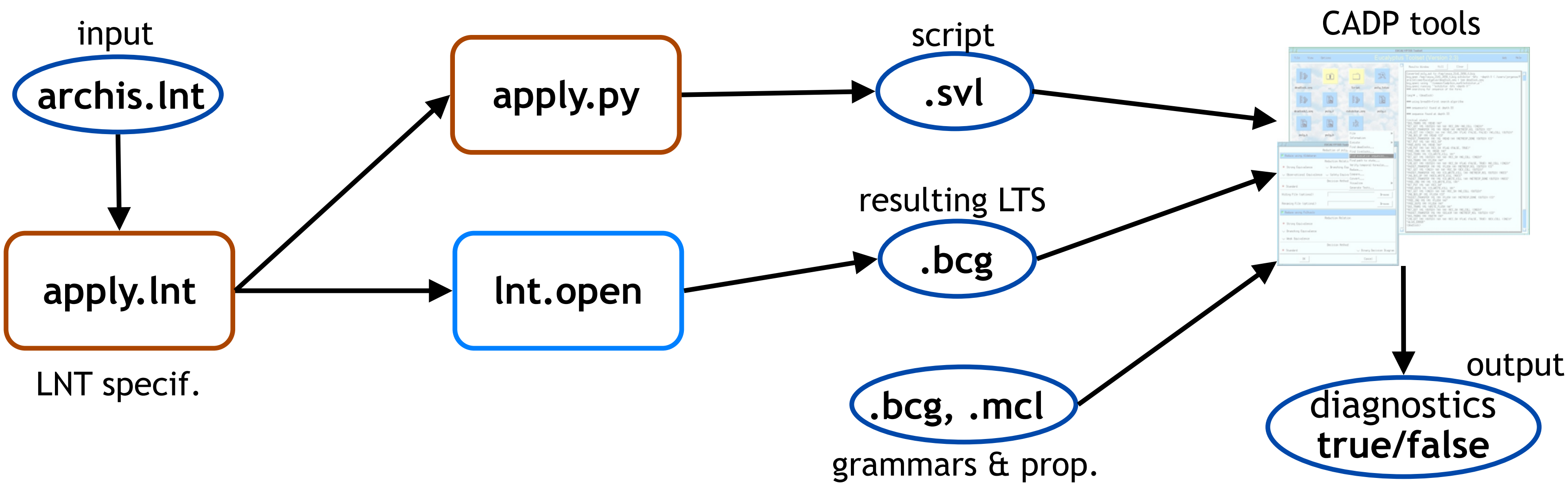
- We identified 8 invariants that both architectures must respect, *e.g.*, for an import, if its component is resolved, the wired export must belong to a resolved component

- Each component must respect a pre-defined grammar of operations



- The whole protocol must verify some temporal properties (15), *e.g.*, if a component is resolved, it is illegal to wire vital or mandatory imports

Tool Support



- Ex.: an architecture with 5 components, requiring 13 changes to reach the target, a failure can occur at any change
- The resulting LTS contains 38830 st./39042 tr. (497 st./580 tr. after strong reduction), and the whole process takes 1:25
- Experiments were conducted on more than 200 examples

Outline

- SYNERGY
- The reconfiguration protocol
- LOTOS-NT and CADP
- Specification in LOTOS-NT
- Verification using CADP
- Concluding remarks

Concluding Remarks (1/2)

- We have presented the specification and verification of the reconfiguration protocol implemented in SYNERGY

- The experience was successful because we have detected several issues which allowed to revise and correct several parts of the protocol

- Two major modifications were made on the protocol:
  - Introduction of two additional (un)wire phases (a single wire/unwire was originally present in the V-shaped protocol)
  - Several corrections of the failure propagation algorithm

Concluding Remarks (2/2)

- As far as CADP languages and tools are concerned:
  - One of the first nontrivial applications of the LOTOS-NT specification language
  - About 30 possible improvements have been identified either in the LOTOS-NT language or in the translation to LOTOS
  - A promising experience in an innovative area (dynamic systems) where CADP tools have not been often used

- Perspectives:
  - Implementing a test case generator: a first version already exists (500 lines of Python)
  - Co-simulating the Java code and LOTOS-NT specification
  - Proposing a parallel version of the reconfiguration protocol