QuviQ

Connecting QuickCheck and RoseRT to test Radio Base Stations

Hans Svensson

Erlang Factory - London 2009

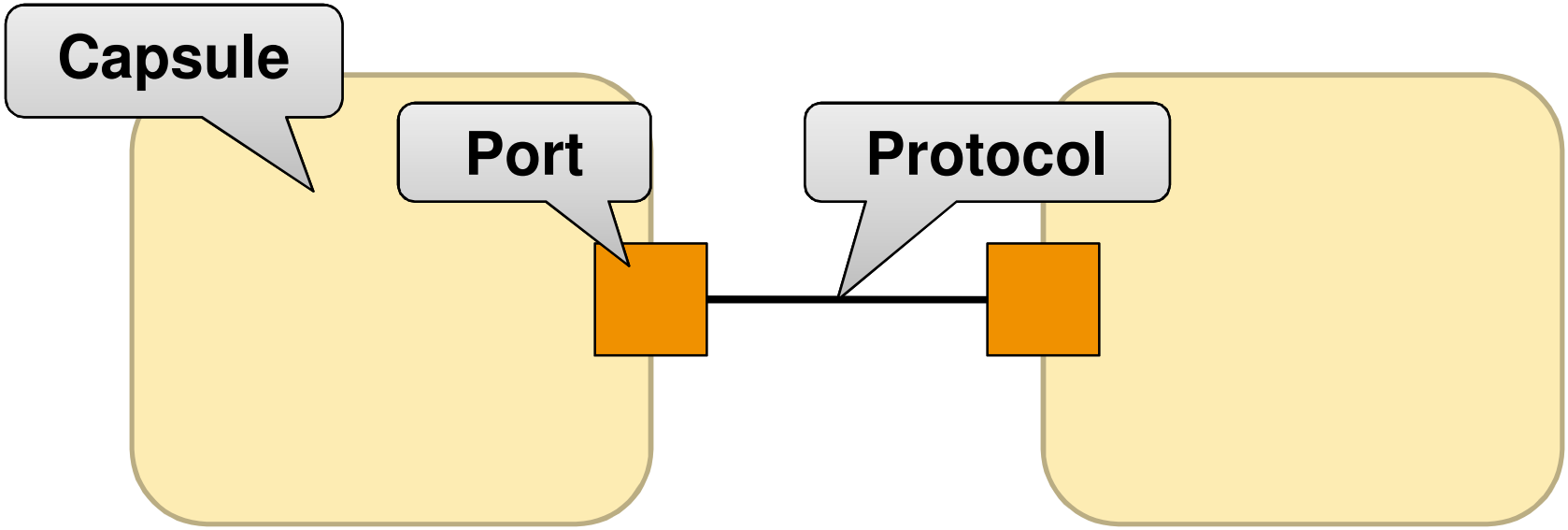
June 25 2009

Introduction

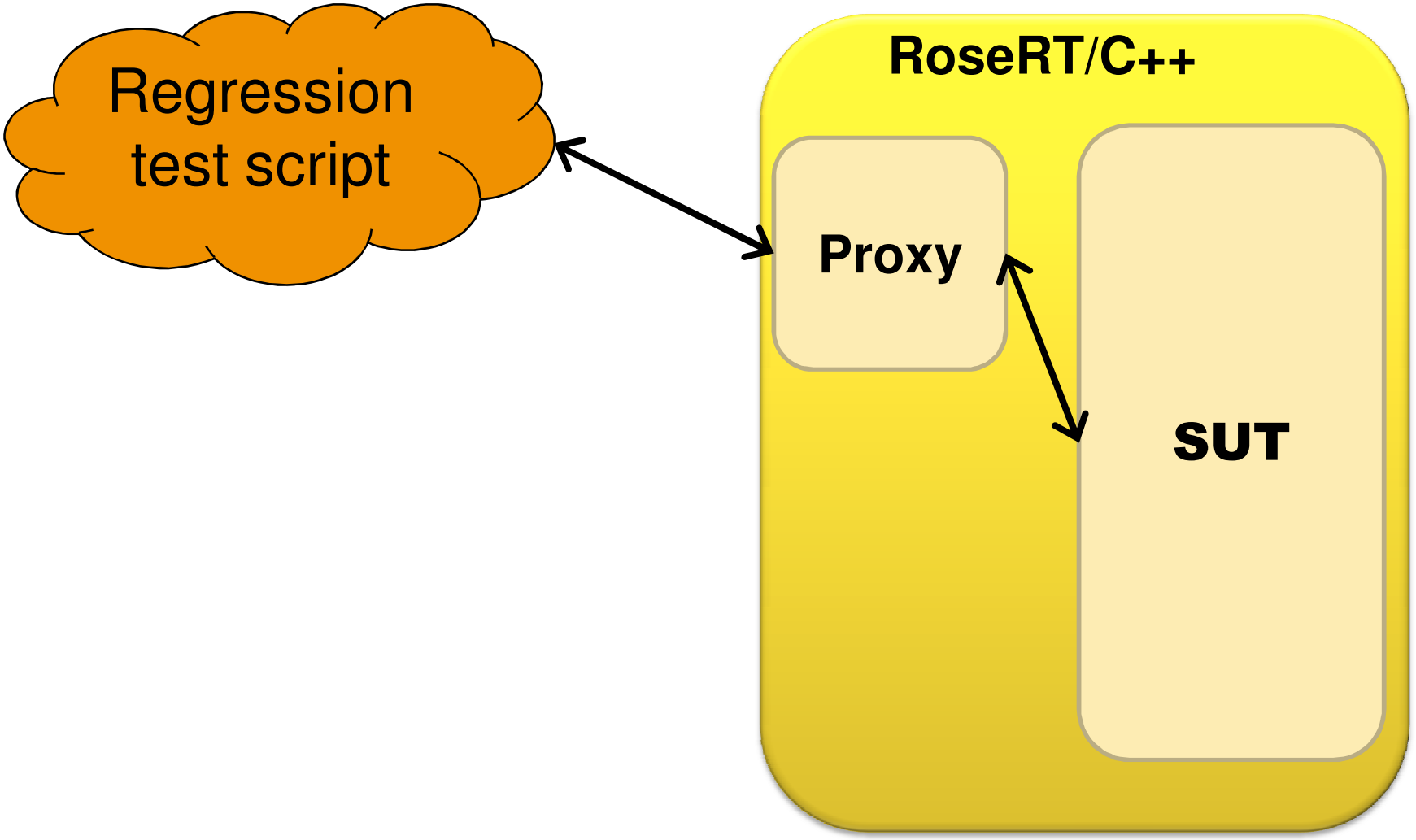
- Strangely not all systems are developed using Erlang
- But we can still have fun and use Erlang
- QuickCheck has already been tested in many different settings (C, Java, Protocol-testing, etc) what about a system in RoseRT?

Rational Rose Real-time -- RoseRT

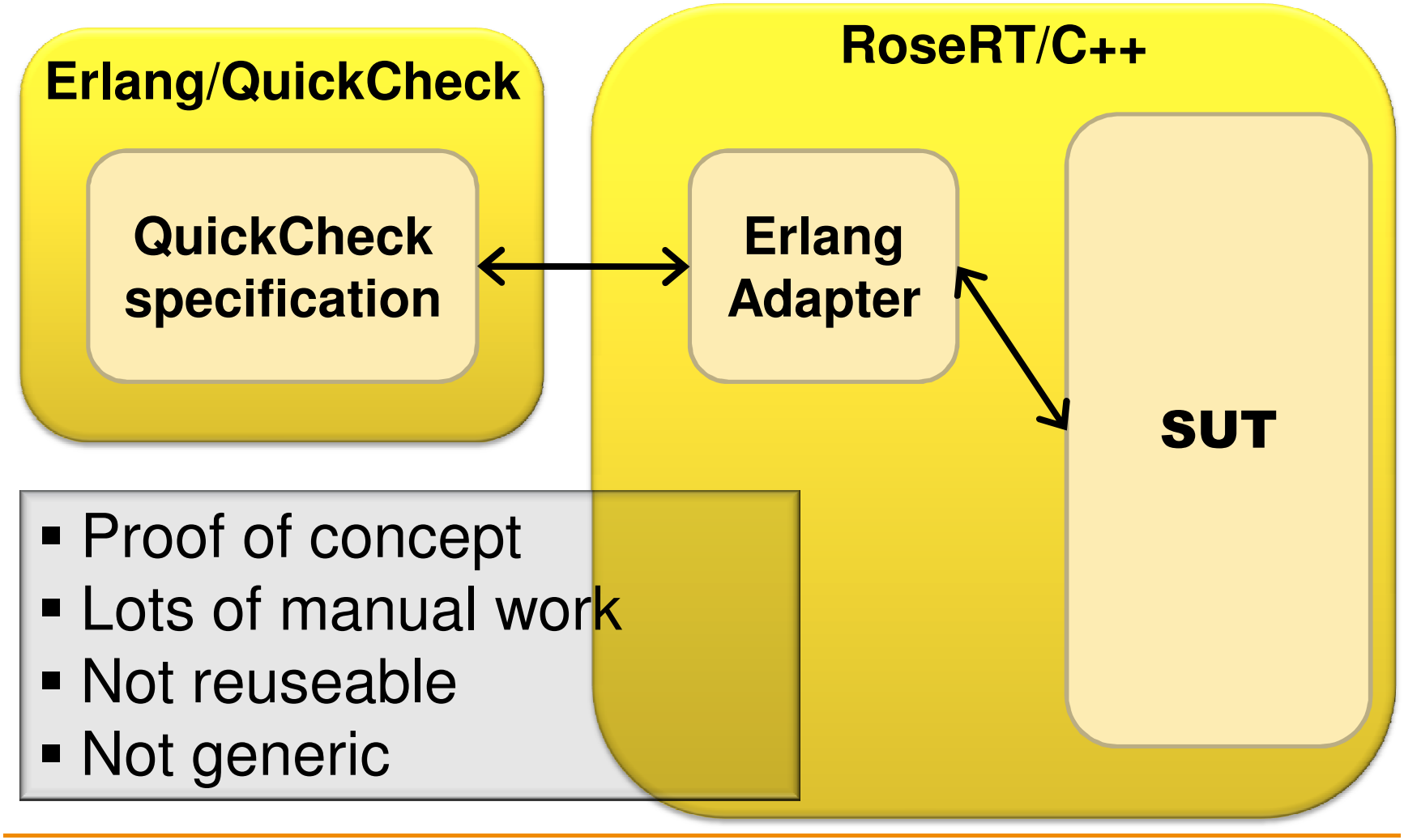
- Based on UML with real-time notation
- Hierarchical and message based
- Uses *actor model concurrency*
- Generates C++ code



Existing Test Configuration



New Test Configuration



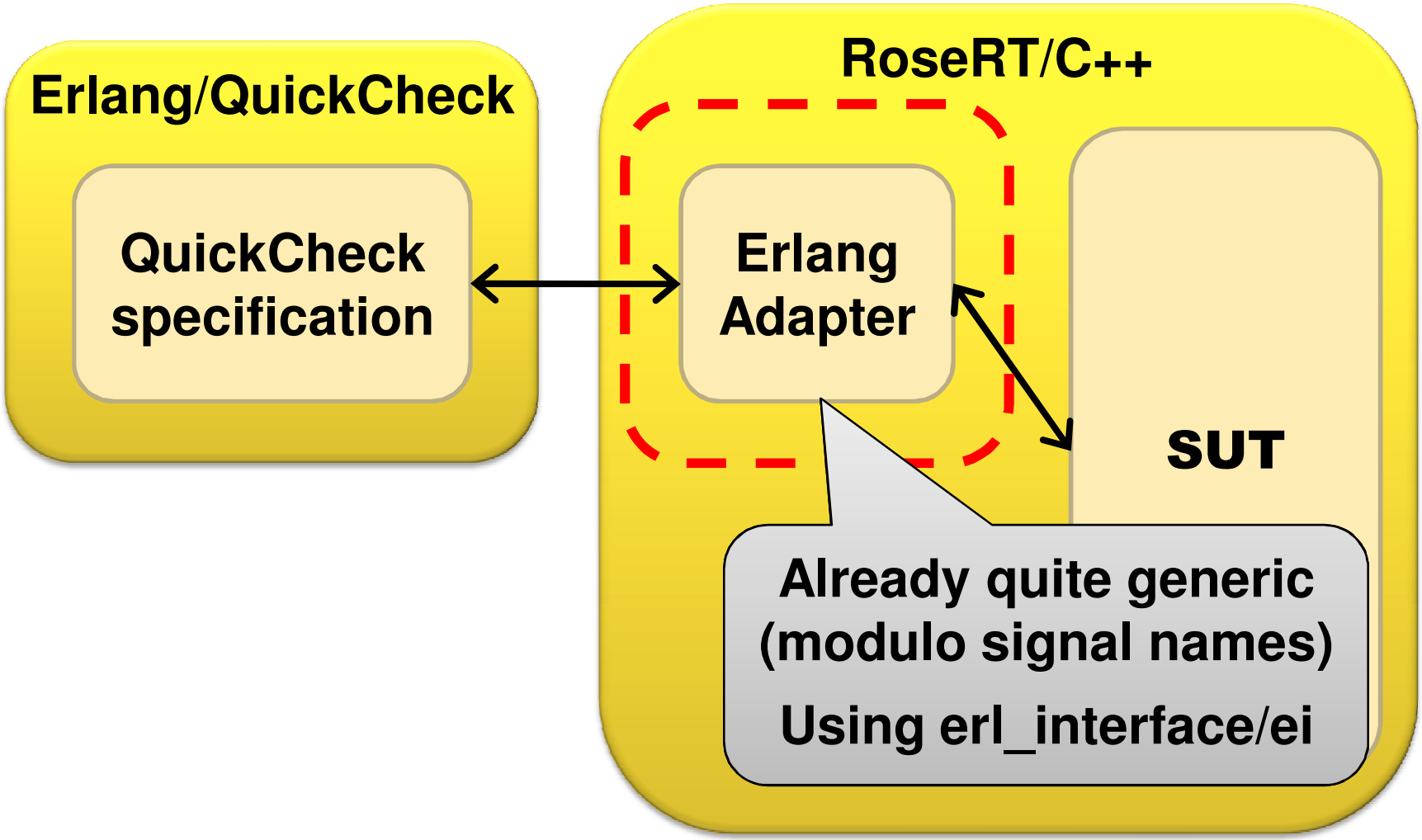
- Proof of concept
- Lots of manual work
- Not reuseable
- Not generic

Generalization -- Main Challenges

- Connecting RoseRT to Erlang/QuickCheck
- Defining a ‘self contained’ part of the Model
- Marshalling of data between Erlang and C++
- Signal generators

+ The usual QuickCheck challenges!

Generalization



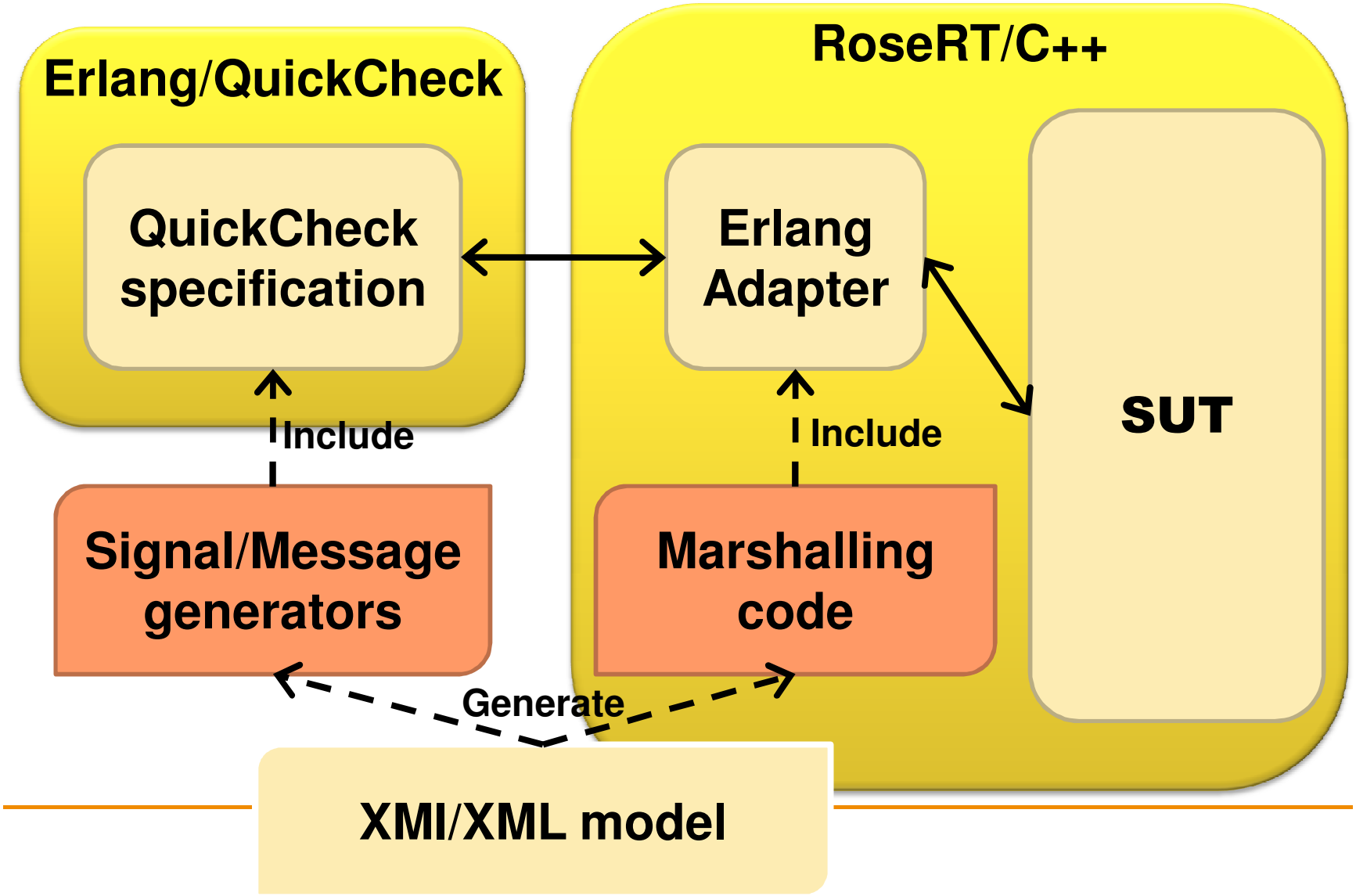
Generalization -- Main Challenges

- Connecting RoseRT to Erlang/QuickCheck
- **Defining a 'self contained' part of the Model**
- Marsh… between Erla…
- Sign…

- XMI/XML representation of model
- In-house developed for **other** purposes
- Some work to find signal definitions

Domain knowledge useful!

Generalization



Signal/Message Generator

- They are just compound data types
- QuickCheck can do `int,boolean,char,etc` – simply put the pieces together
- Sometimes not clever enough!

```
signal open_account:
 int account_nr
 int pin_code
 float balance
```

```
open_account()->
 {signal,open_account,
  int(),int(),float()}.
```

- Not likely to match account with PIN code
- Almost impossible to auto generate

Marshalling example

Marshalling a pointer? An integer? No!

Erlang

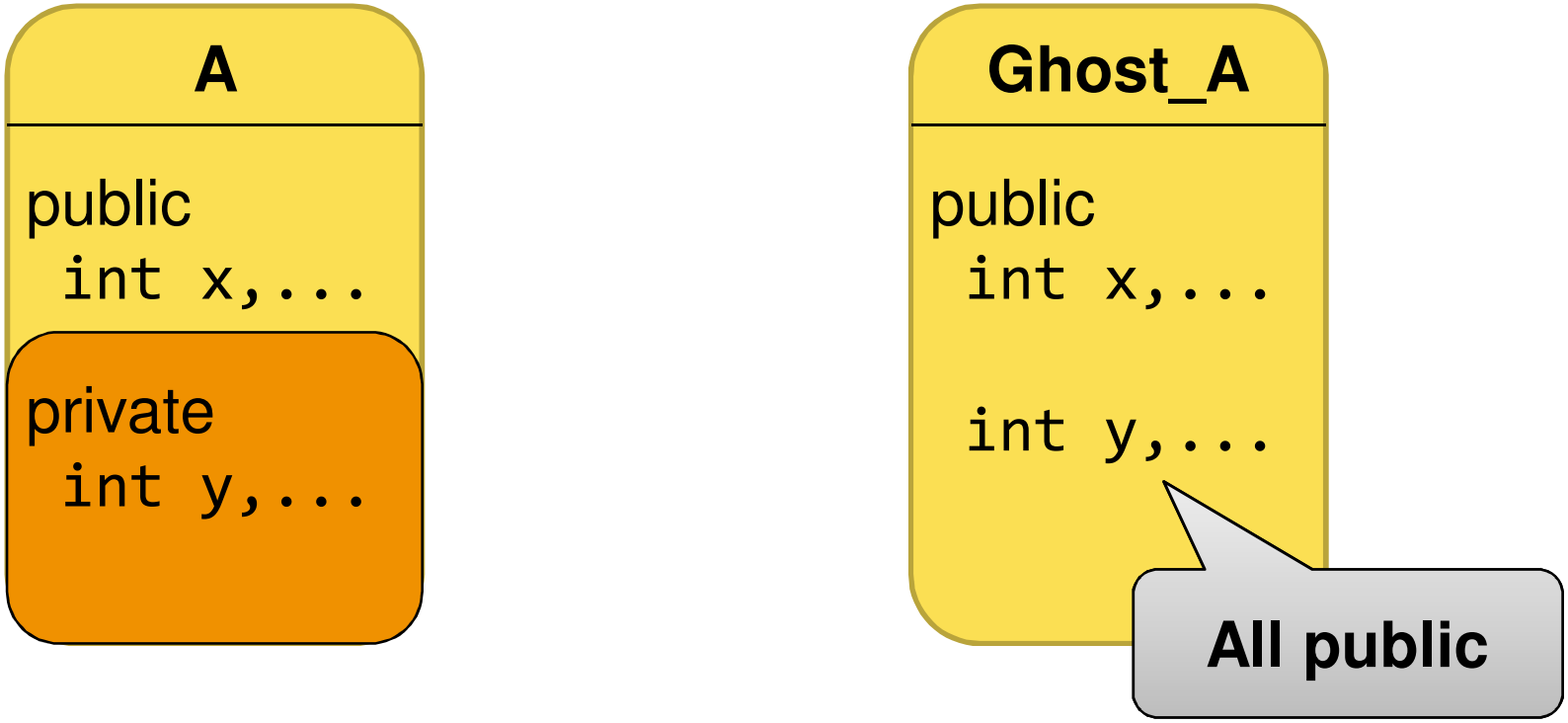
```
{pointer, ptr, class_A, {classA, ...}}
```

C++

```
ClassA* ptr = &marshall_ClassA(...);
```

Marshalling problem – Private attributes

Marshalling functions can't access private fields!



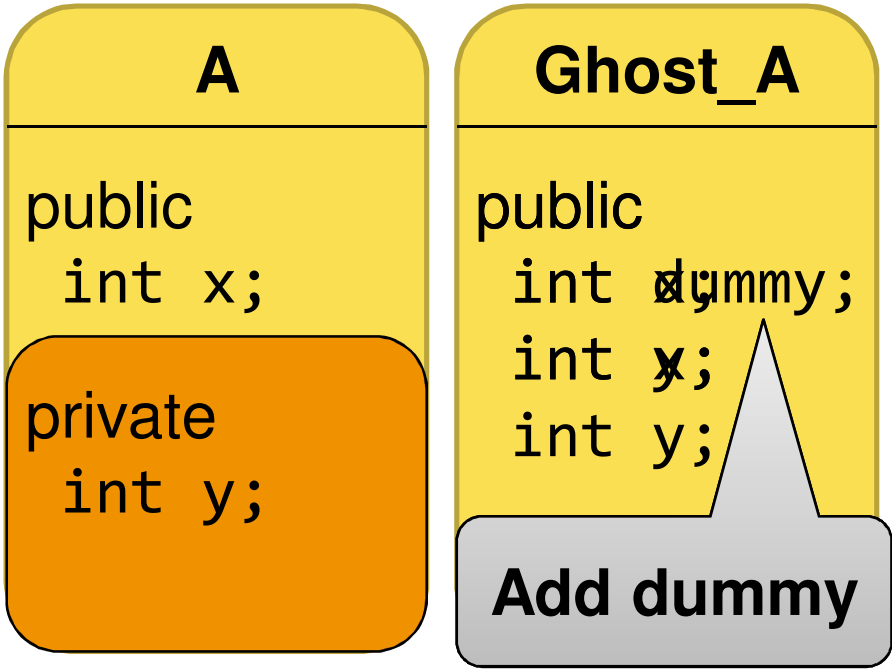
OK, except for a technical detail...

Marshalling problem – Private attributes

OK, except for a technical detail...

Memory alignment!

```
A* ptrA; Ghost_A* ptrGA;
ptrA = new A(1,2);
ptrGA = (Ghost_A *)ptrA;
```



Now we expect:

```
ptrGA->x <==> ptrA->x
```

But instead:

```
ptrGA->x <==> ptrA->y
```

Summary

- With a bit of work we could benefit from QuickCheck while testing a RoseRT system
- The solution is general, but not really general enough – more work is needed
- Generators with implicit meanings are hard to generalize in a clever way

We had fun!

Acknowledgements

- Jia Wang and Shyun Shyun Yeoh, Chalmers University of Technology
- John Hughes, Quviq
- Andreas Granberg, Ericsson AB