Specifying, Testing and Verifying a Networked Server

From C to Interaction Trees

Nicolas Koh, Yao Li, Yishuai Li, **Li-yao Xia**
Lennart Beringer, Wolf Honoré, William Mansky
Benjamin C. Pierce, Steve Zdancewic

January 14, 2019 (CPP)

Verification from RFCs to transistors

deep spec

Verified Software Toolchain

CERTIKOS

Kami

Application

OS

Hardware

01011…

One theorem to verify…

Quick Chick

… and test!

More projects at deepspec.org…

Towards a verified web server

deep spec server

Verified Software Toolchain

HTTP Server

CERTIKOS

OS

Kami

Hardware

QuickChick

01011...

Towards a verified web server

deep spec server

Verified Software Toolchain

CERTIKOS

Kami

Swap Server

OS

Hardware

Today: a simplified server

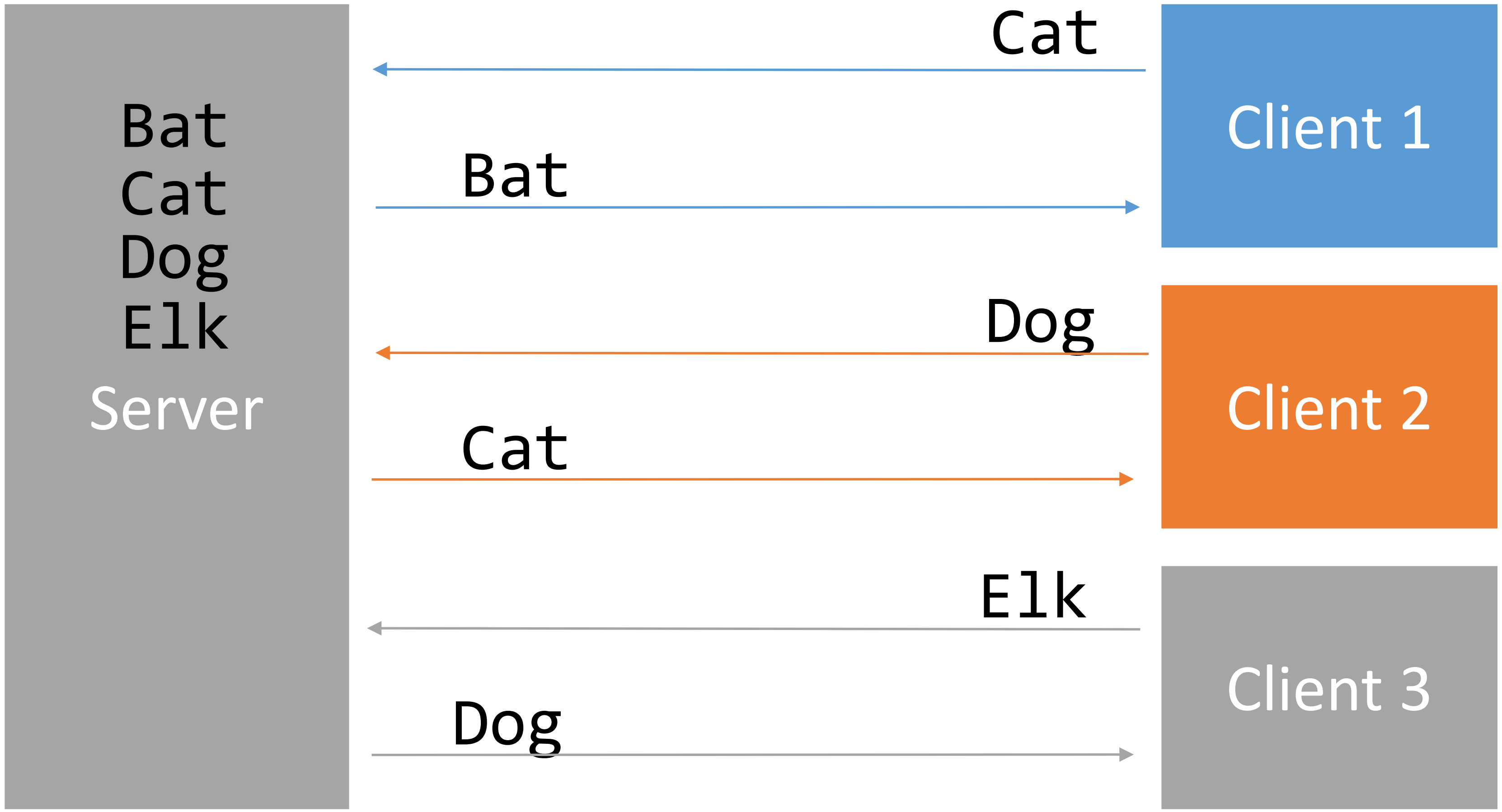
QuickChick

01011...

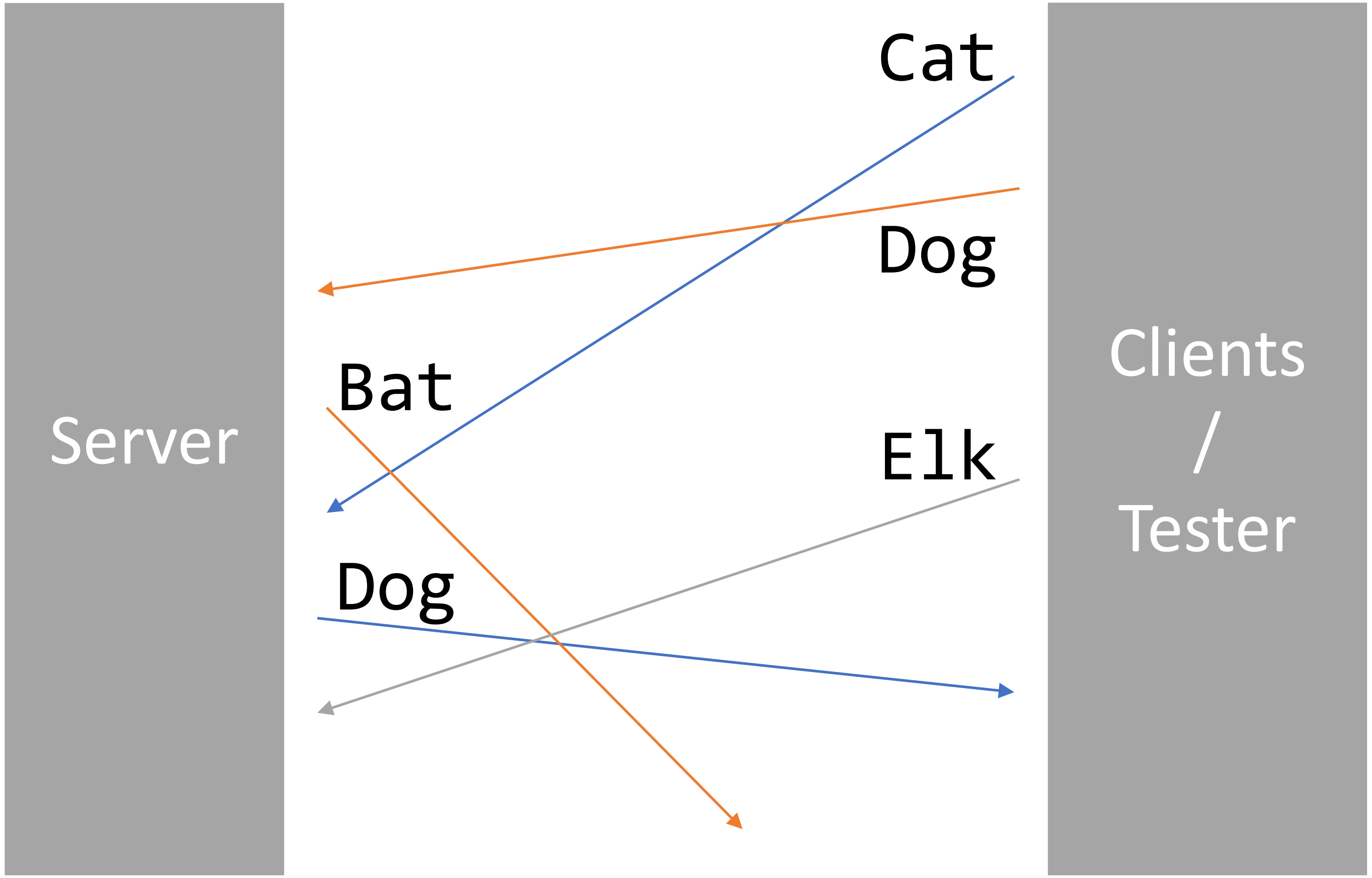
Main contributions

- Verifying a networked C program using VST, which can run in CertiKOS
- Specification describes what a client can observe *over the network*
- Testable specification, using QuickChick

Swap server specification

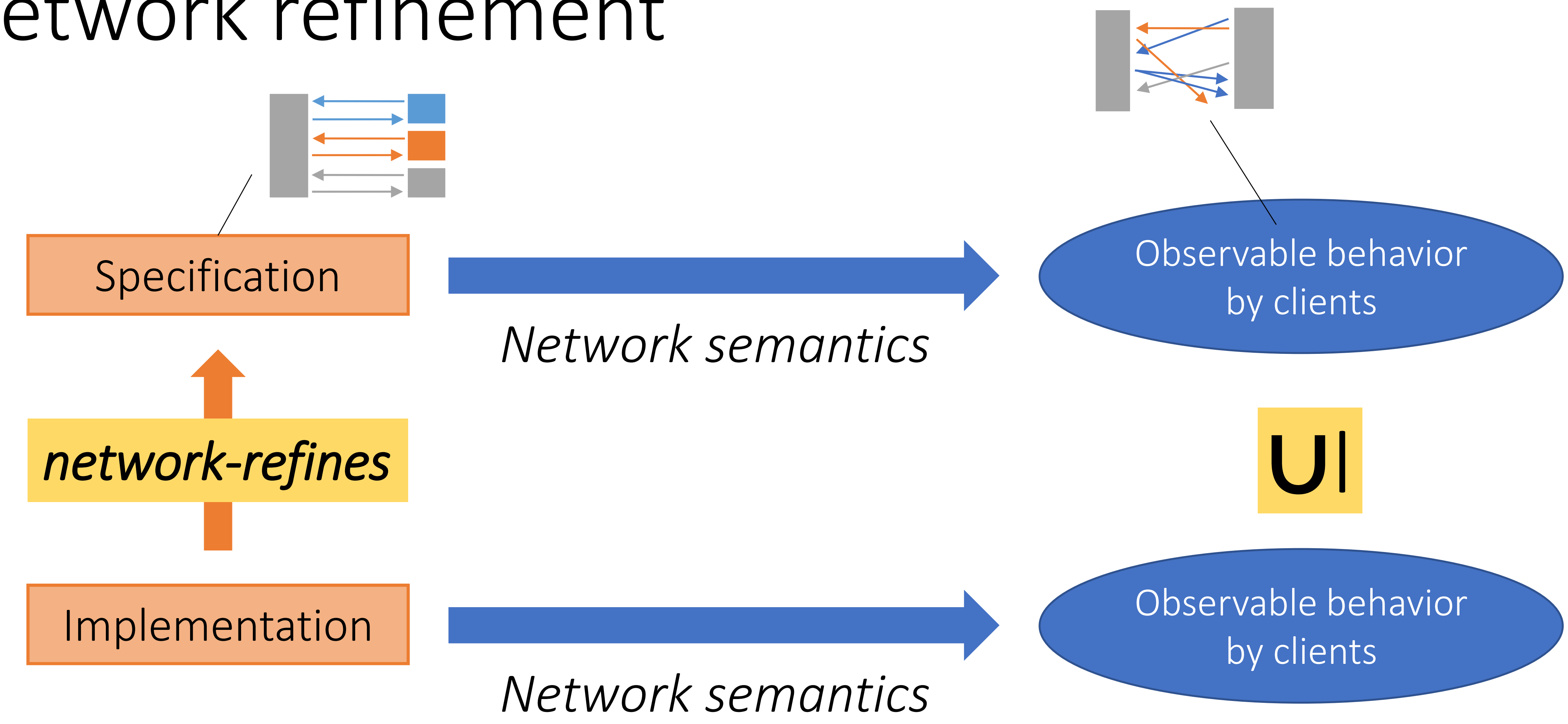


Swap server: in the real world



- Messages on different connections can be reordered
- Messages can be delayed indefinitely

Network refinement

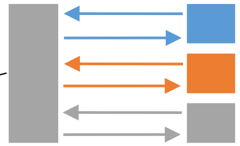


Adaptation of Observational refinement/Linearizability

*: concepts defined in the paper

Overview: proof architecture

*Specification**



Written in Coq

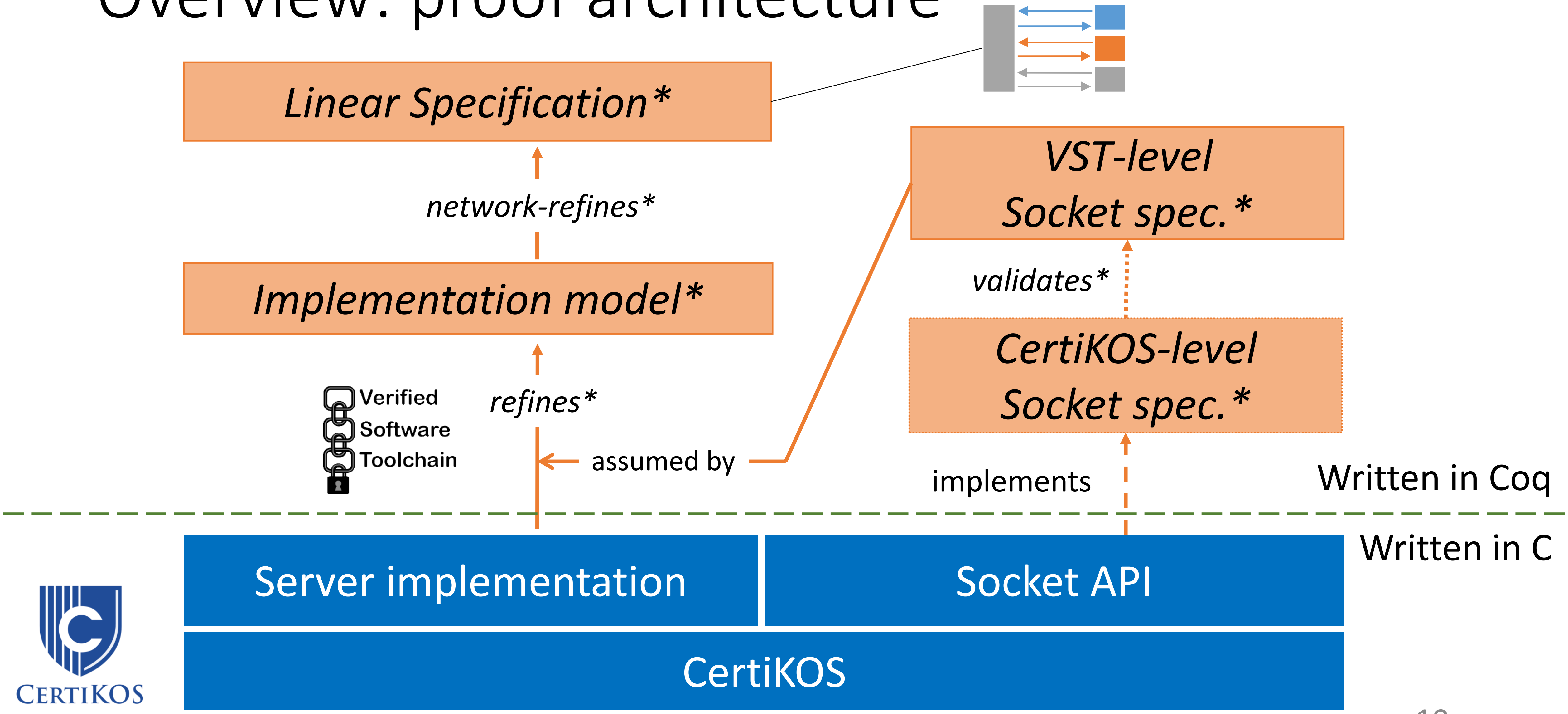
Written in C

Server implementation	Socket API
CertiKOS	

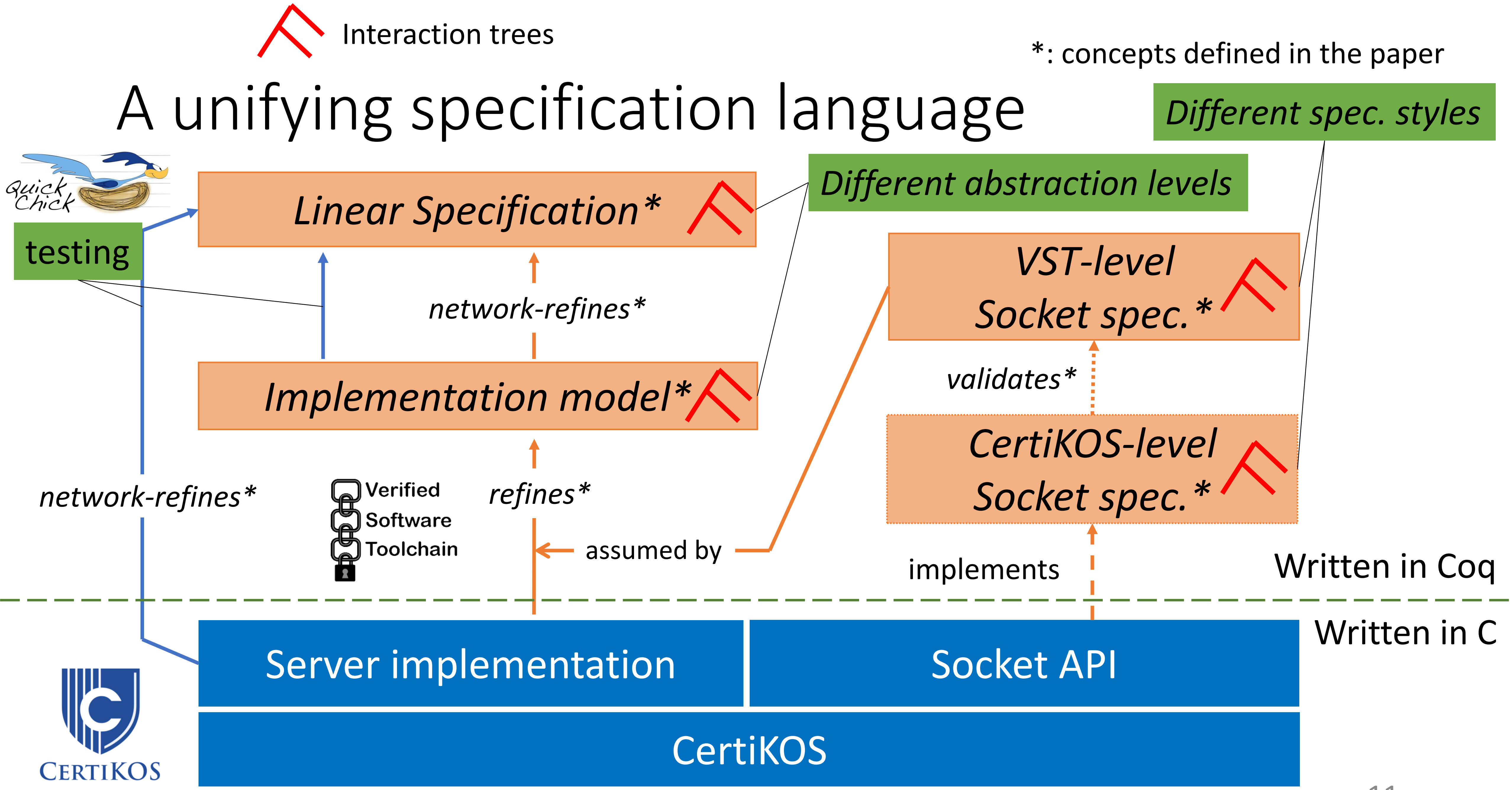
CertiKOS

*: concepts defined in the paper

Overview: proof architecture

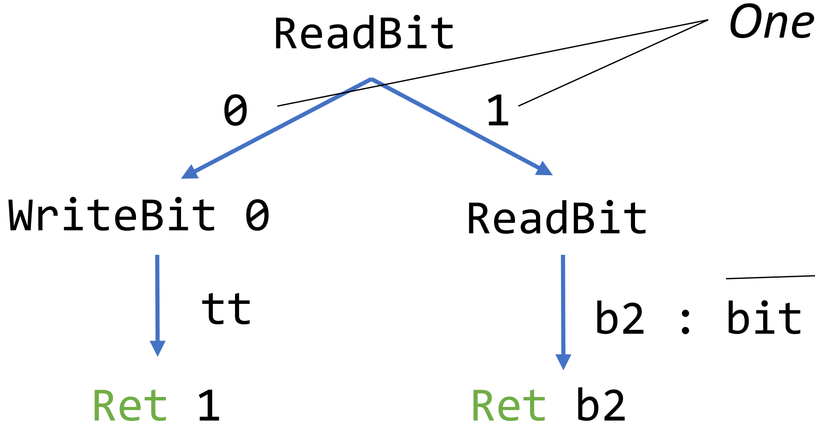


A unifying specification language



Interaction trees: example

(aka. Free monads)



One branch for each possible result

Shorthand notation for two or more branches

Type of effects ioE:

```
Inductive ioE : Type -> Type :=
| ReadBit  : ioE bit
| WriteBit : bit -> ioE unit
.
```

Result type

Interaction trees: definition

(aka. Free monads)

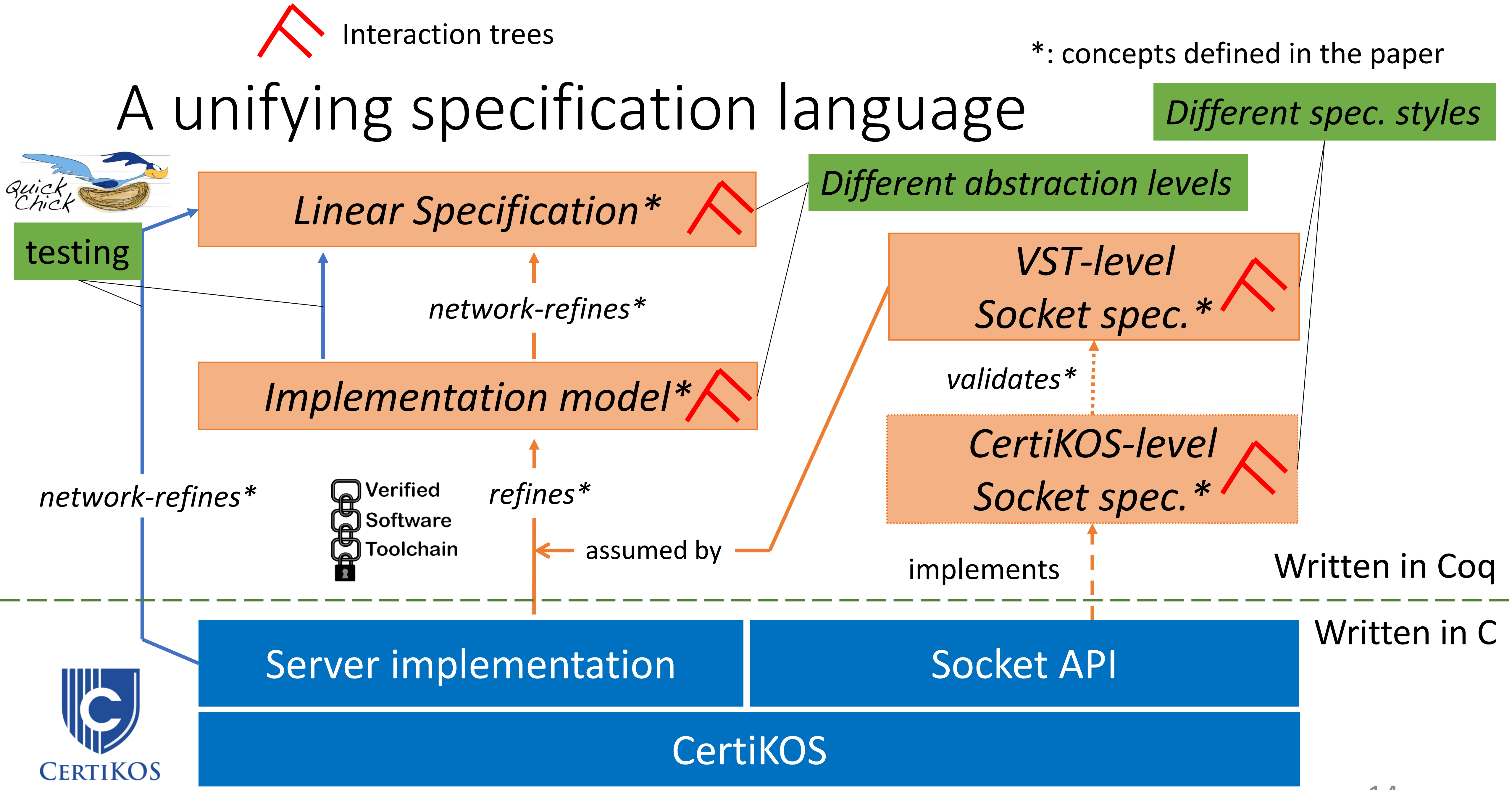
Type of effects (e.g., ioE)

Type of results

```
CoInductive itree (E : Type -> Type) (R : Type) :
Type :=
| Vis : ∀ Y, E Y -> (Y -> itree E R) -> itree E R
| Ret : R -> itree E R
| Tau : itree E R -> itree E R
.
```

Effect

Continuation


Interaction trees
*: concepts defined in the paper
A unifying specification language
Different spec. styles
Different abstraction levels
QuickChick
testing
Linear Specification*
network-refines*
Implementation model*
VST-level
Socket spec.*
validates*
CertiKOS-level
Socket spec.*
network-refines*
Verified
Software
Toolchain
refines*
assumed by
implements
Written in Coq
Written in C
Server implementation
Socket API
CertiKOS
CERTIKOS

The Swap server "linear specification"

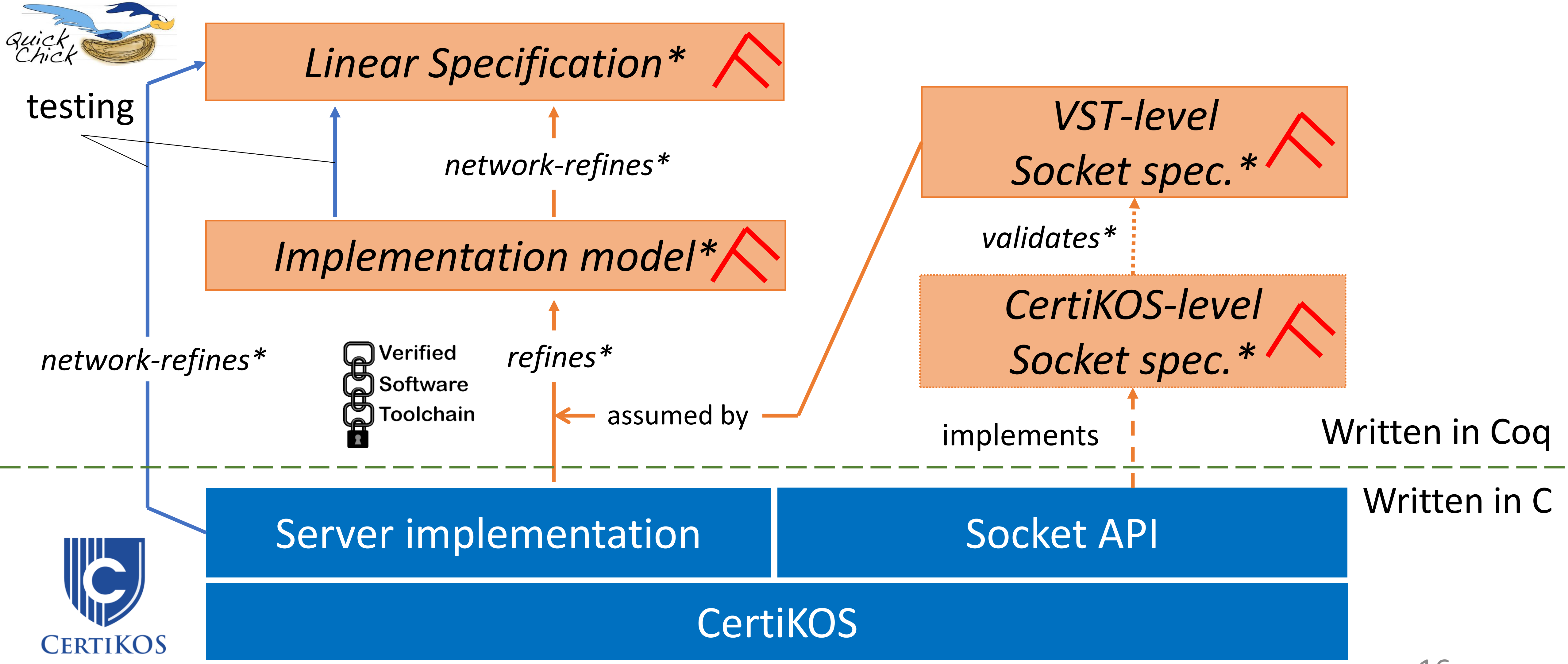
```
CoFixpoint
  loop (open_conns : list conns)
       (last_msg : bytes) : itree serverE unit
  :=
    c <- choose open_conns ;;
    new_msg <- recv_msg c ;;
    send_msg c last_msg ;;
    loop open_conns new_msg.
```

Simplified version (see paper)

Interaction trees

*: concepts defined in the paper

Overview: proof architecture



Refinement: from C to ITrees

Hoare triple: `{ pre1 * … * preN } C_program { post1 * … * postN }`

*Interactions **allowed** by the environment* — *Assertions on C memory* — *Separating conjunction*

```
{ ITree(impl_model) * … } C_program { … }
```

Example of a networked C program with its implementation model:

```
{ ITree(msg <- Recv c ;;
        Send c msg ;;
        t) * … }
```

Implementation model (itree)

```
  recv(c, buf, len);
  send(c, buf, len);
```

C implementation

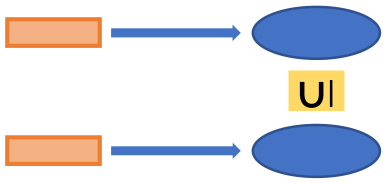
```
{ ITree(t) * … }
```

Simplified triples (see paper)

The Swap server correctness theorem

```
{ ITree(impl_model) } C_prog { … }

Theorem correct_server :
  exists impl_model,
          refines C_prog impl_model /\
    network_refines impl_model linear_spec.
```



Summary and next steps

Complete connection

- Verifying a networked C program using VST, which can run in CertiKOS
- The specification describes a client can observe *over the network*
- The specification is testable, using QuickChick *and Interaction trees*

Scale up:
Swap server ->
HTTP Server

Add more interfaces:
filesystem, encryption…

Improve proof
and testing
techniques

New library

https://github.com/DeepSpec/InteractionTrees