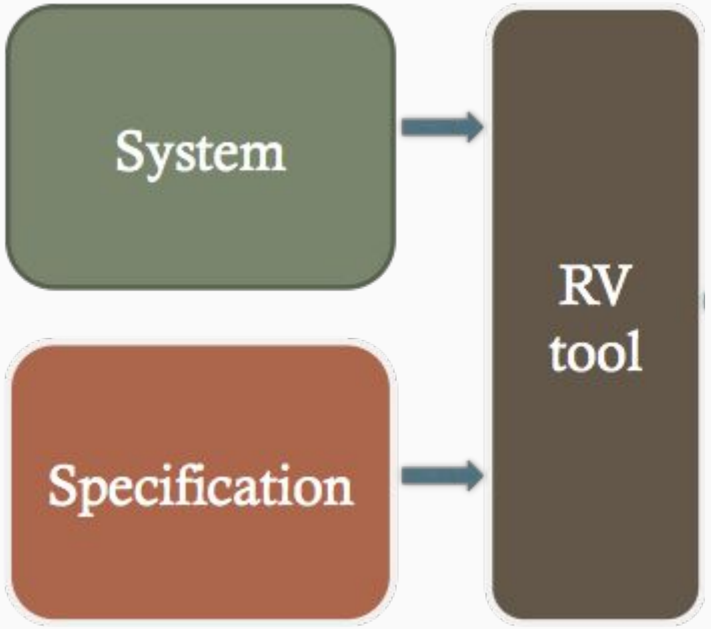
10 years of work in Runtime Verification

Christian Colombo
Athens 2017

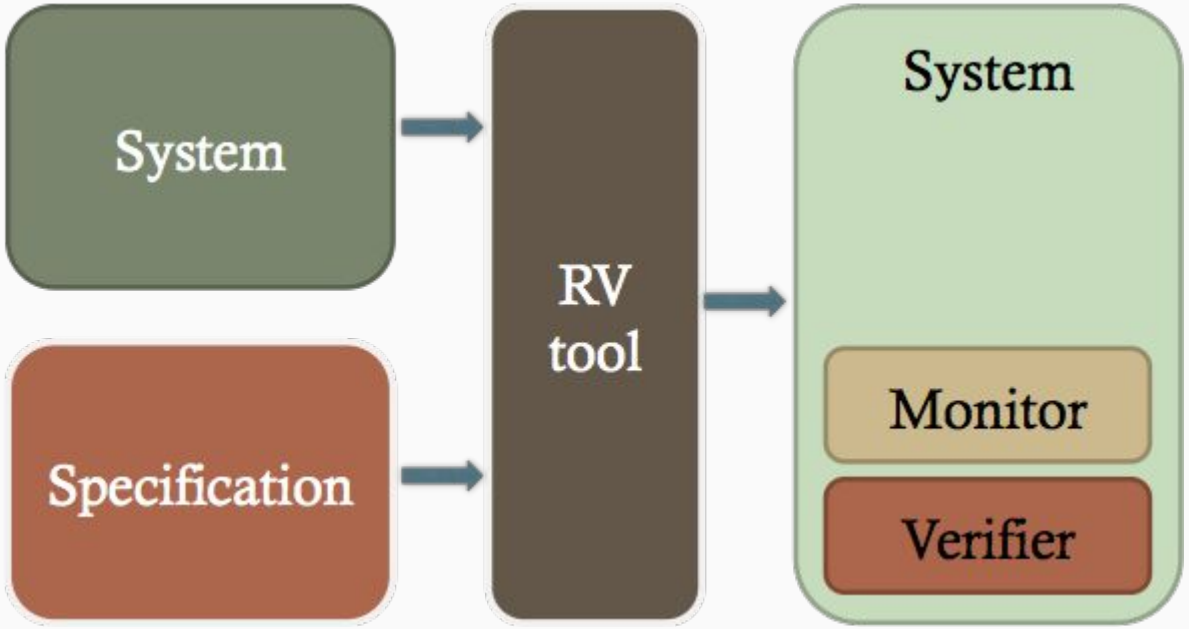
Runtime Verification

System
Specification

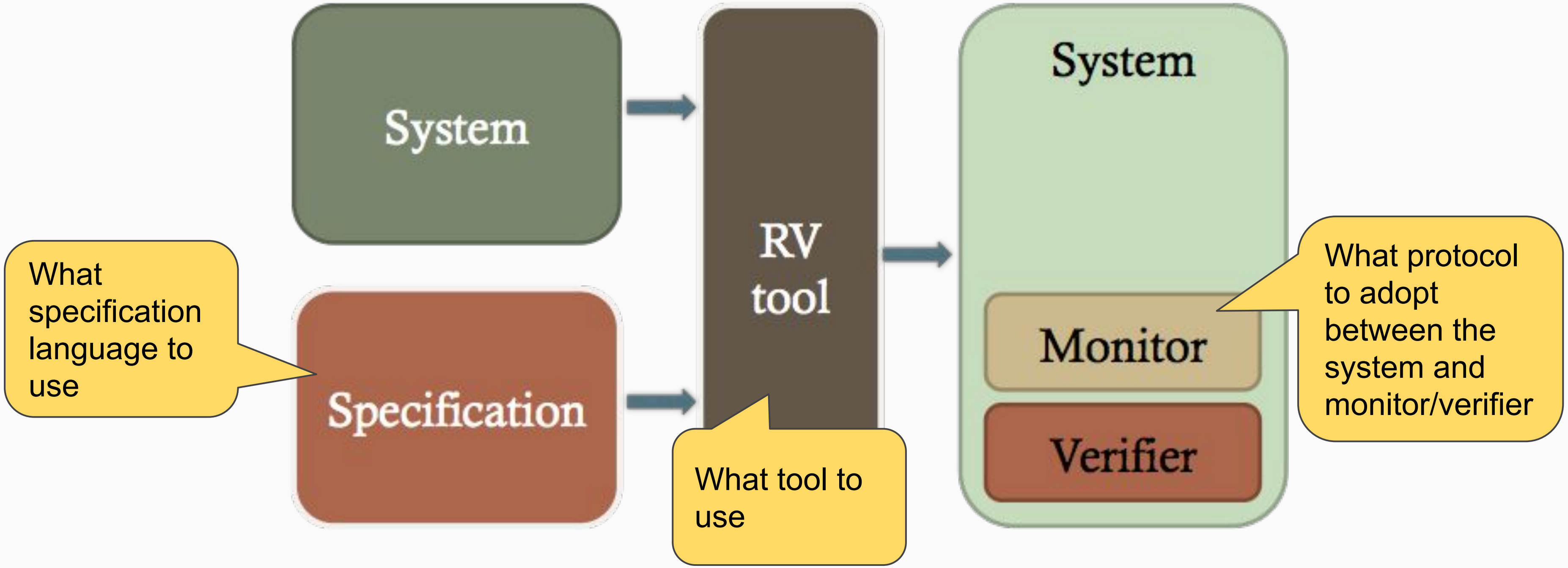
Runtime Verification



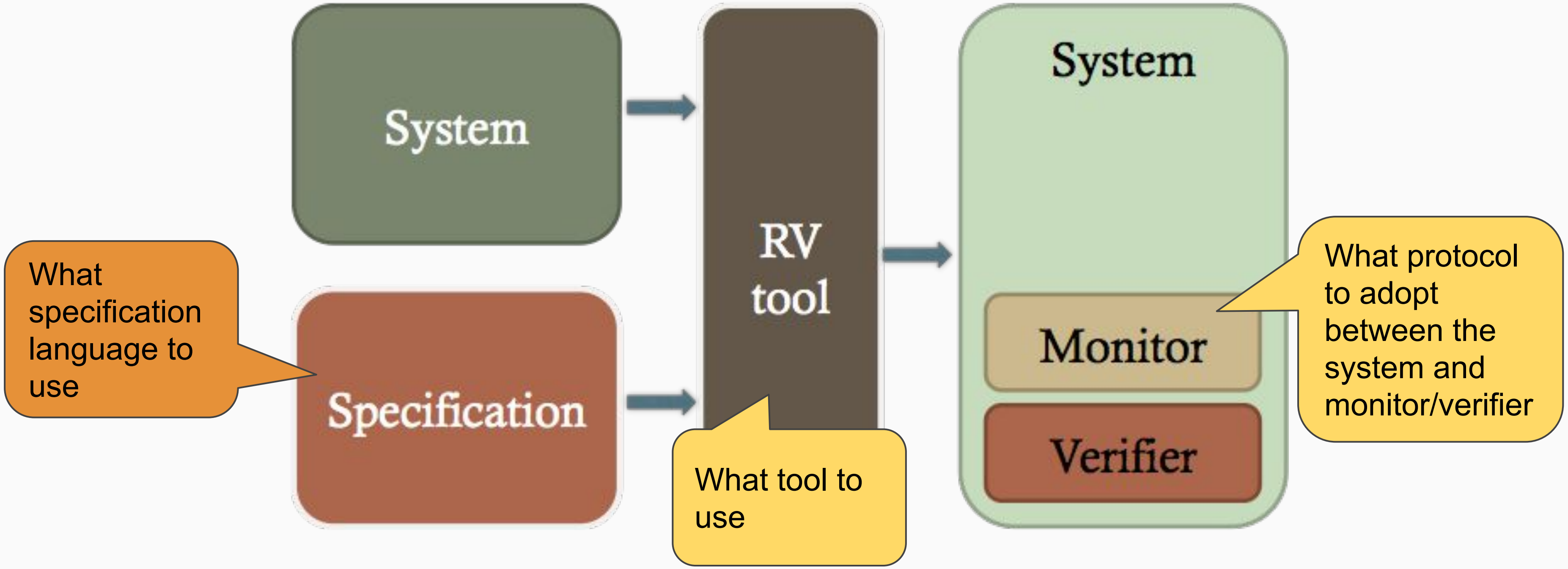
Runtime Verification



Runtime Verification


System
Specification
RV tool
System
Monitor
Verifier
What specification language to use
What tool to use
What protocol to adopt between the system and monitor/verifier

Runtime Verification


System
Specification
RV tool
System
Monitor
Verifier
What specification language to use
What tool to use
What protocol to adopt between the system and monitor/verifier

Specification Languages - Expressivity

Support for:

Sequencing of events **“No write before a login”**

Real-time **“Never more than 5 bad logins in 1 minute”**

Per-object **“For each user, total spending cannot exceed €100 per day”**

Specification Languages - Understandability

Formats:

- Logics **(!login)* write**
- Automata (finite state machines)
- Domain-specific languages (sometimes as controlled natural languages)

Domain-specific languages

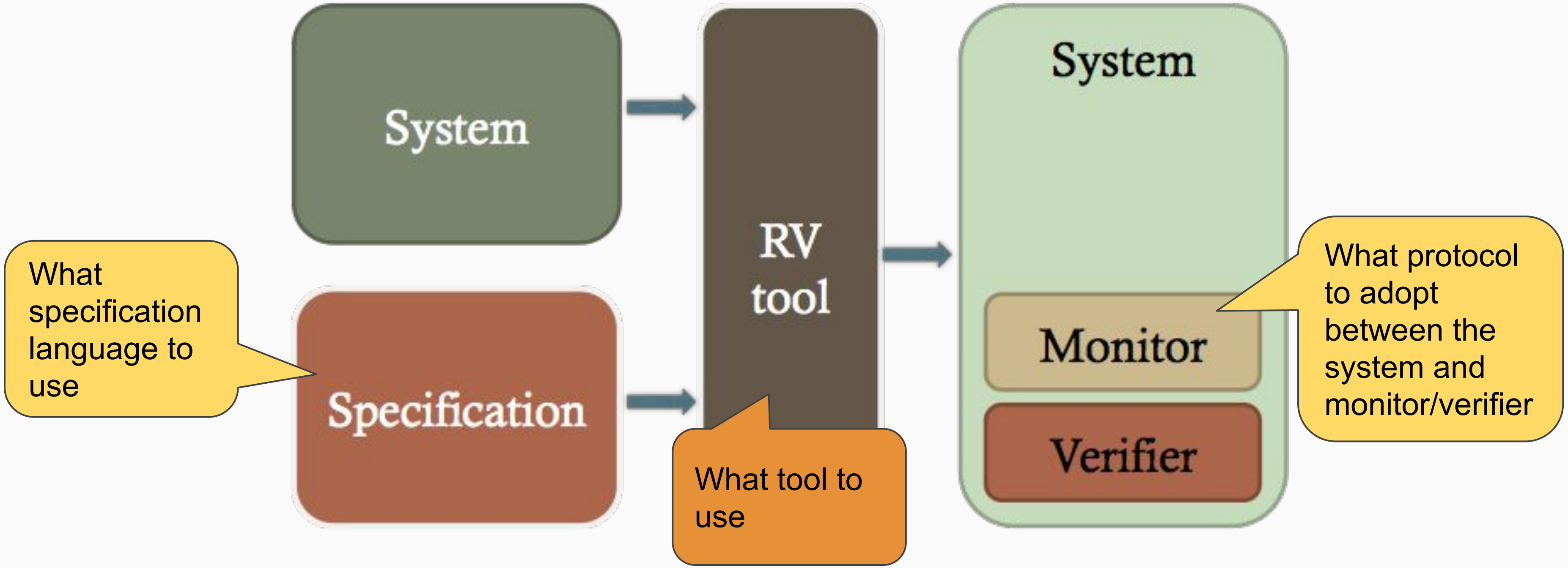
VISA regulations **“A non-verified cardholder can only spend €X per month”**

Tax fraud **“Tax payers declaring an income less than X% of the last Y-year average”**

Business intelligence **“Alert me whenever a post gets more than X negative comments in Y minutes”**

Fraud risk **“Increase risk score by X% for each transfer from country Y with amount greater than Z”**

Runtime Verification


System
Specification
RV tool
System
Monitor
Verifier
What specification language to use
What tool to use
What protocol to adopt between the system and monitor/verifier

Technologies

Java + AspectJ

Java + Kafka/RabbitMQ/etc

Erlang

C

A mixture of technologies

Architectures

Monolith systems

Distributed systems with a global clock

Distributed systems

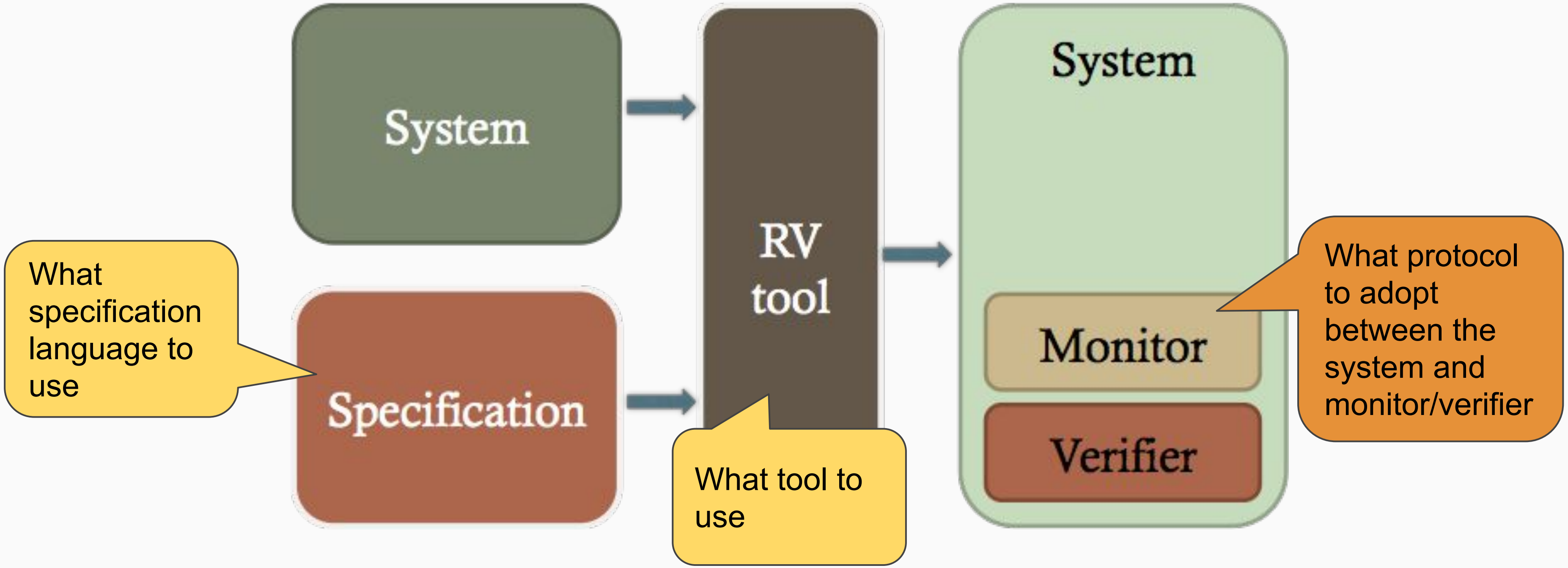
Ways of obtaining events

By modifying the code

By intercepting communication

From a data source (eg: database)

Runtime Verification


System
Specification
RV tool
System
Monitor
Verifier
What specification language to use
What tool to use
What protocol to adopt between the system and monitor/verifier

Monitor and System work in parallel?

System and monitor wait for each other

System runs independently of the monitor

What happens when a problem is detected?

The monitor simply raises an alert

The monitor can “fix” the situation