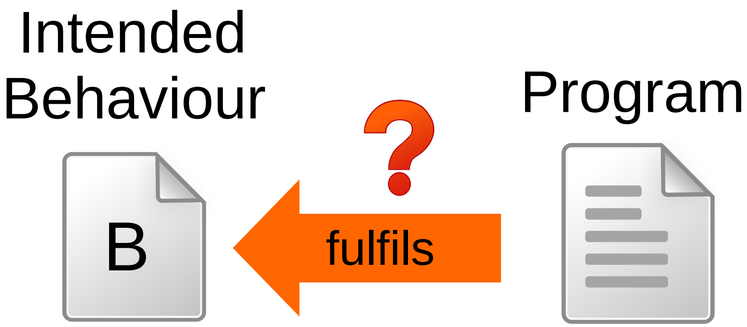
Unified Static and Runtime Verification of Object-Oriented Software

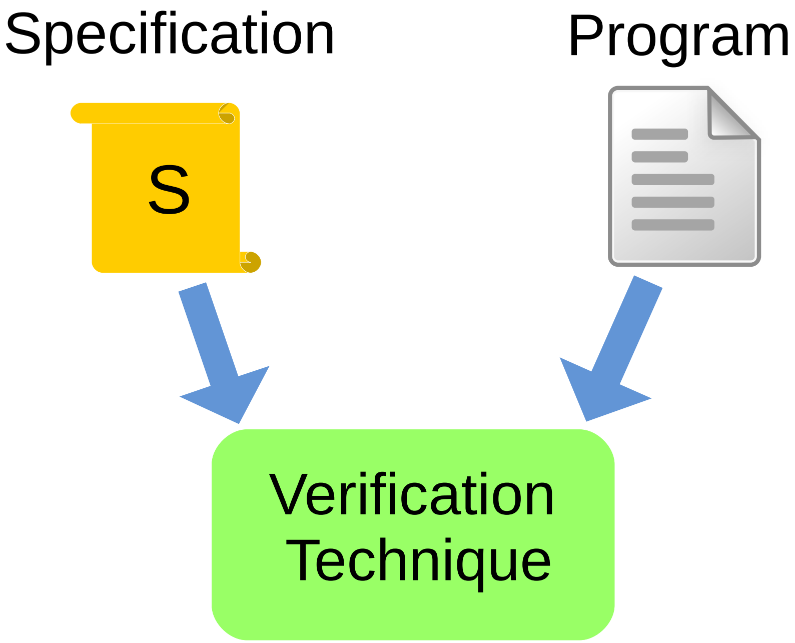
Mauricio Chimento

13 November 2017

Program Verification



Program Verification



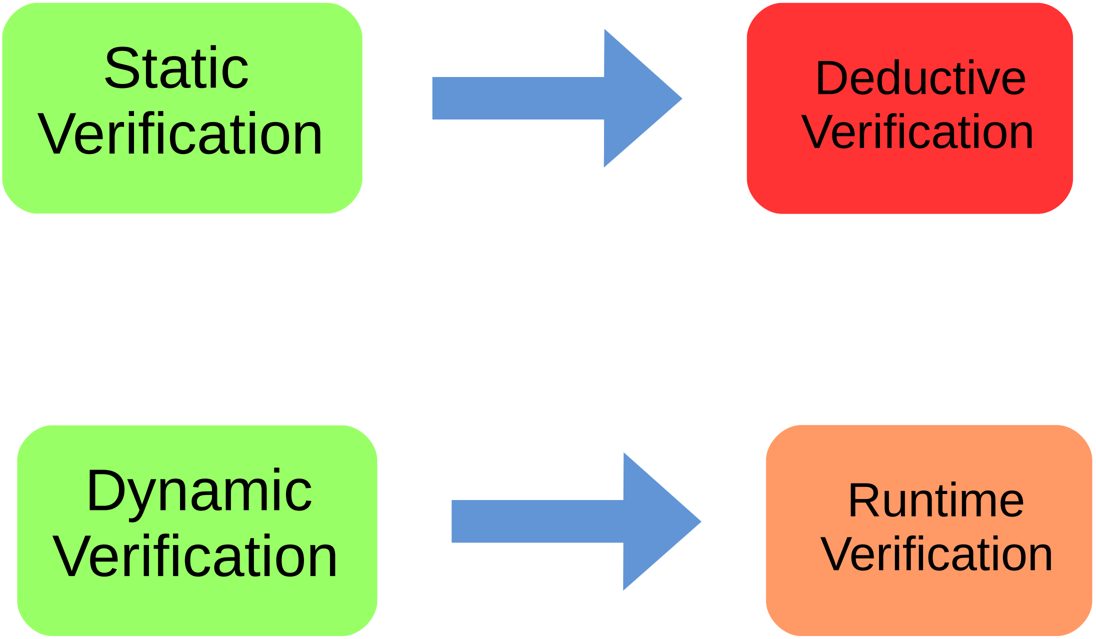
Specification

S

Program

Verification Technique

Verification Techniques



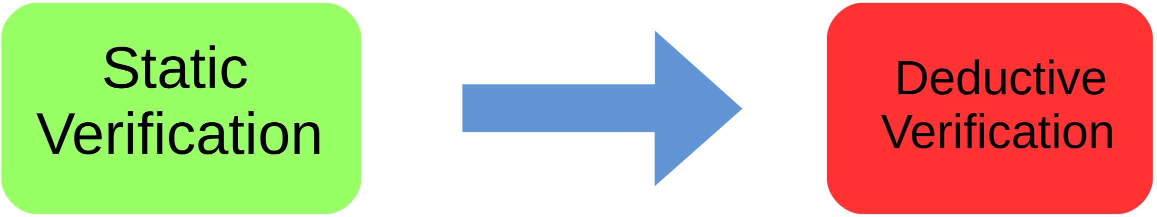
Static Verification

Deductive Verification

Dynamic Verification

Runtime Verification

Deductive Verification



Static Verification

Deductive Verification

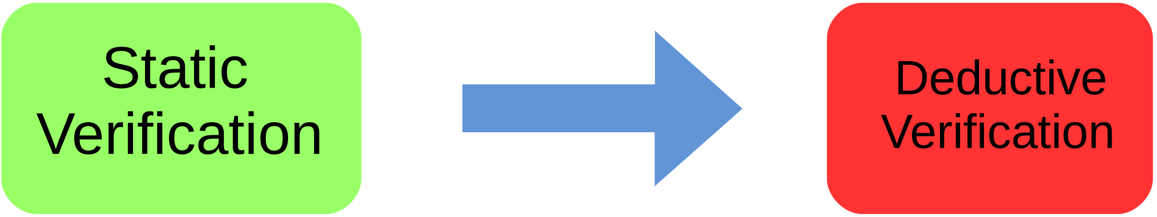
- Properties written as logical formulae

{ P } foo() { Q }

- Formulae are verified by deduction in a calculus

$$\frac{\Gamma, \quad b \ \vdash \quad < s_1 \ \omega > \phi \qquad \Gamma, \quad \neg b \ \vdash \quad < s_2 \ \omega > \phi}{\Gamma \vdash \quad < \texttt{if} \ b \ s_1 \ \texttt{else} \ s_2 \ \omega > \phi}$$

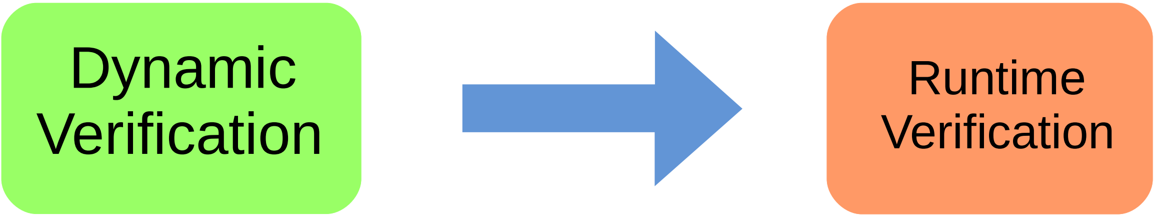
Deductive Verification



- Analysis over all possible executions of the program ✓
- Absence of source code ✗

  (e.g. library methods)

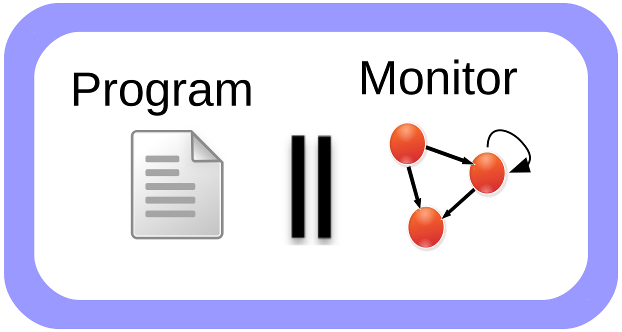
Runtime Verification



Dynamic Verification

Runtime Verification

- Monitoring of program executions

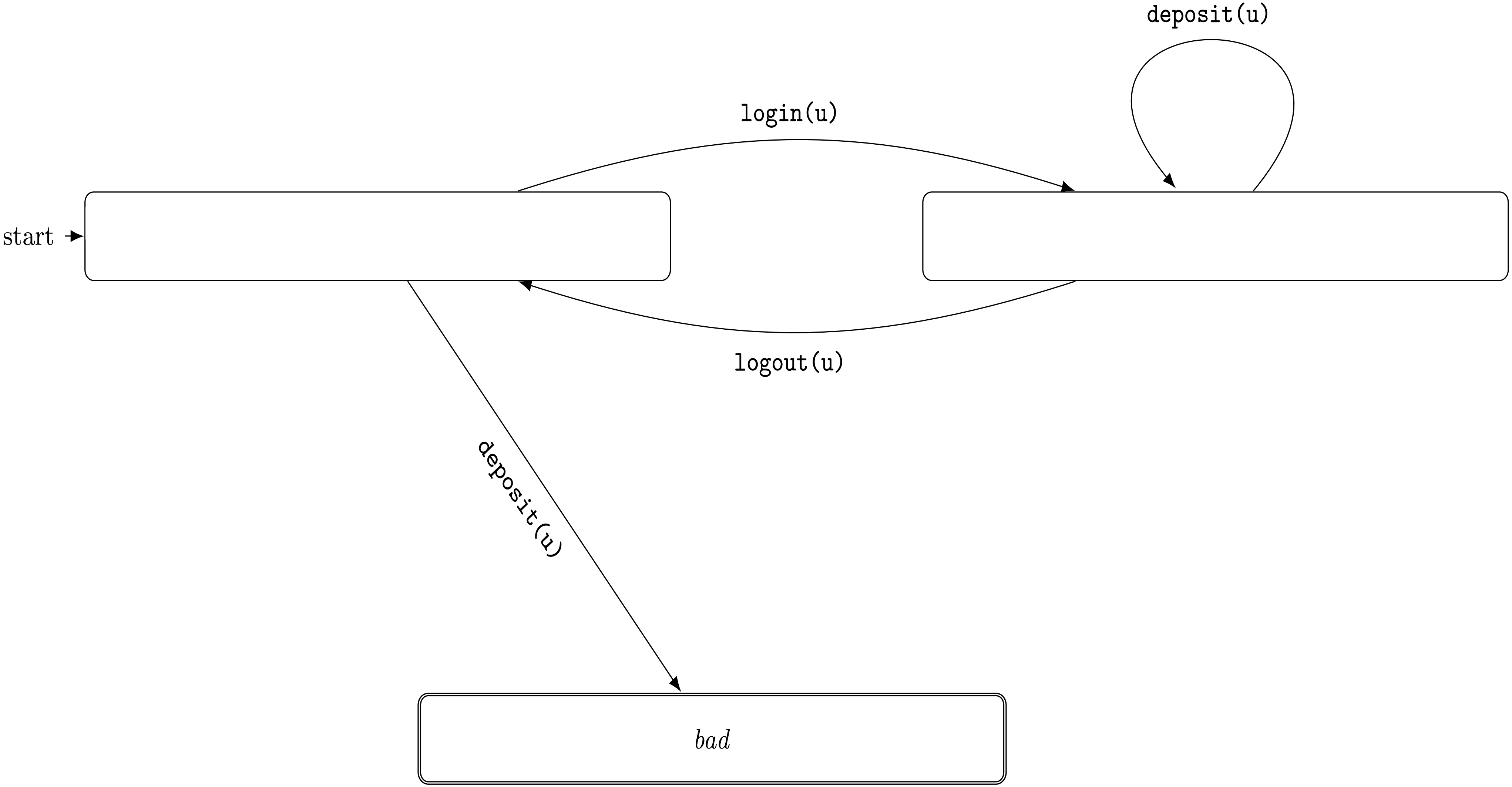


Program

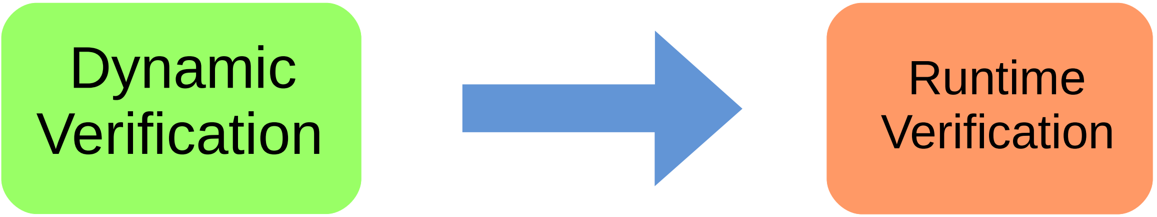
Monitor

- Offline – Online verification

Runtime Verification



Runtime Verification

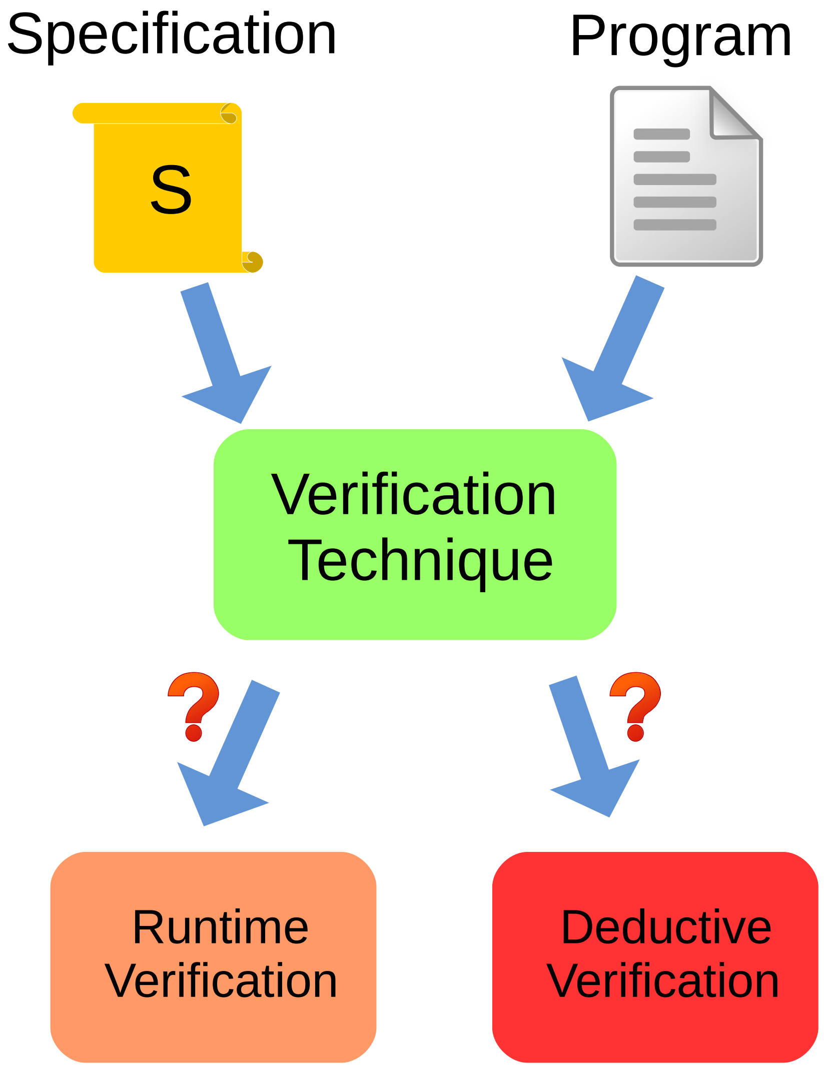


Dynamic Verification

Runtime Verification

- All data available at runtime ✓
- Only current execution ✗
- Execution Overhead ✗

Using the Techniques



Specification

S

Program

Verification Technique

?

?

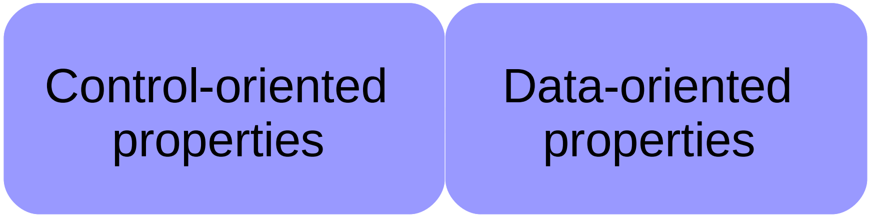
Runtime Verification

Deductive Verification

Using the Techniques

Properties

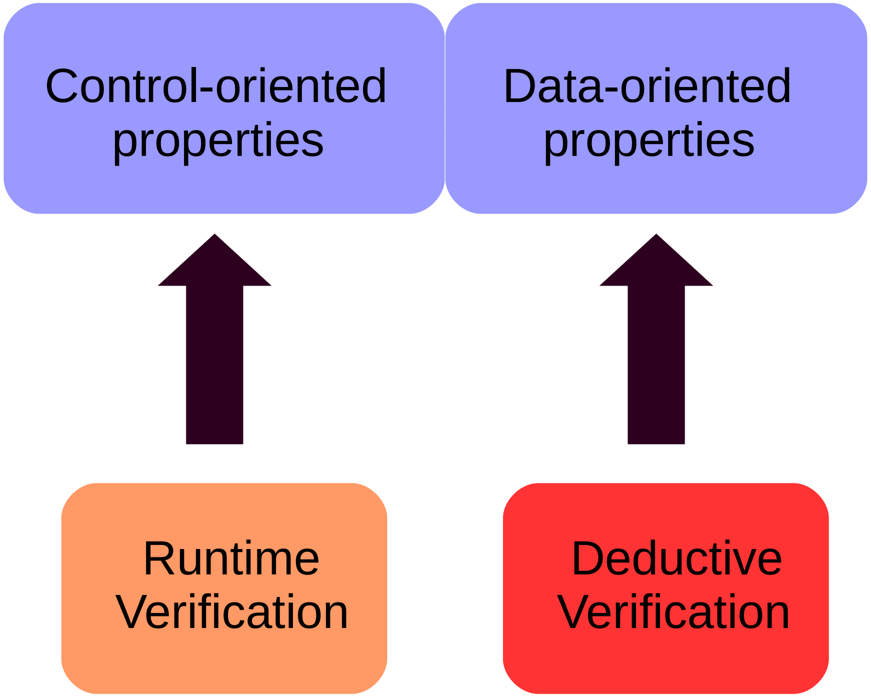
Using the Techniques



Control-oriented properties

Data-oriented properties

Using the Techniques



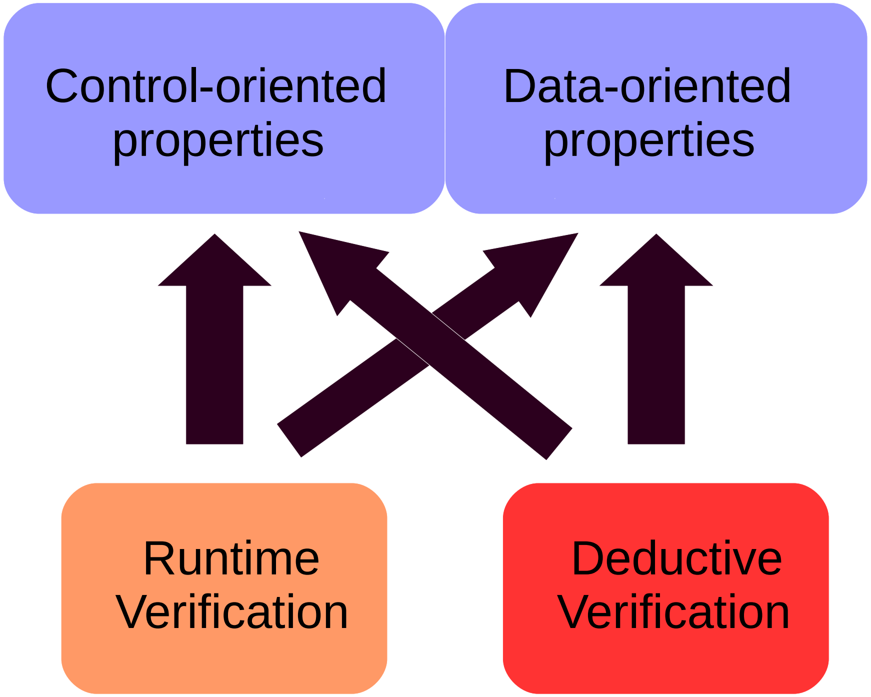
Control-oriented properties

Data-oriented properties

Runtime Verification

Deductive Verification

Using the Techniques



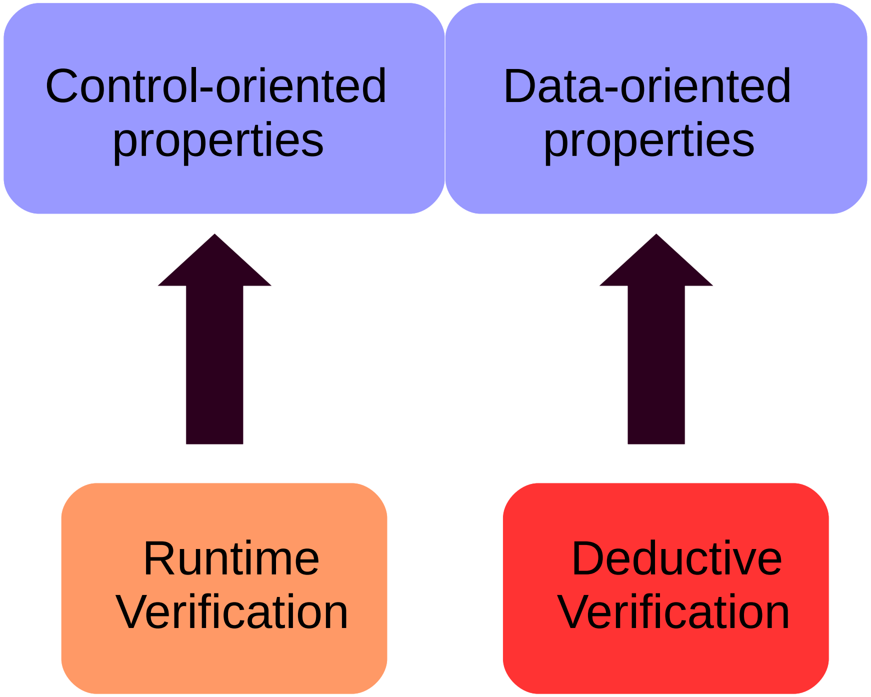
Control-oriented properties

Data-oriented properties

Runtime Verification

Deductive Verification

Using the Techniques



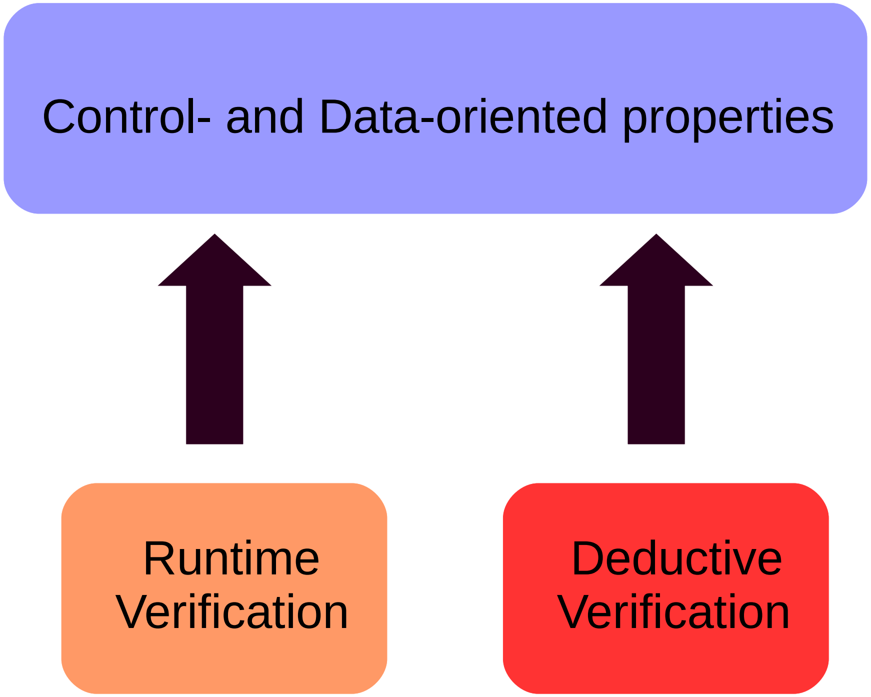
Control-oriented properties

Data-oriented properties

Runtime Verification

Deductive Verification

Using the Techniques

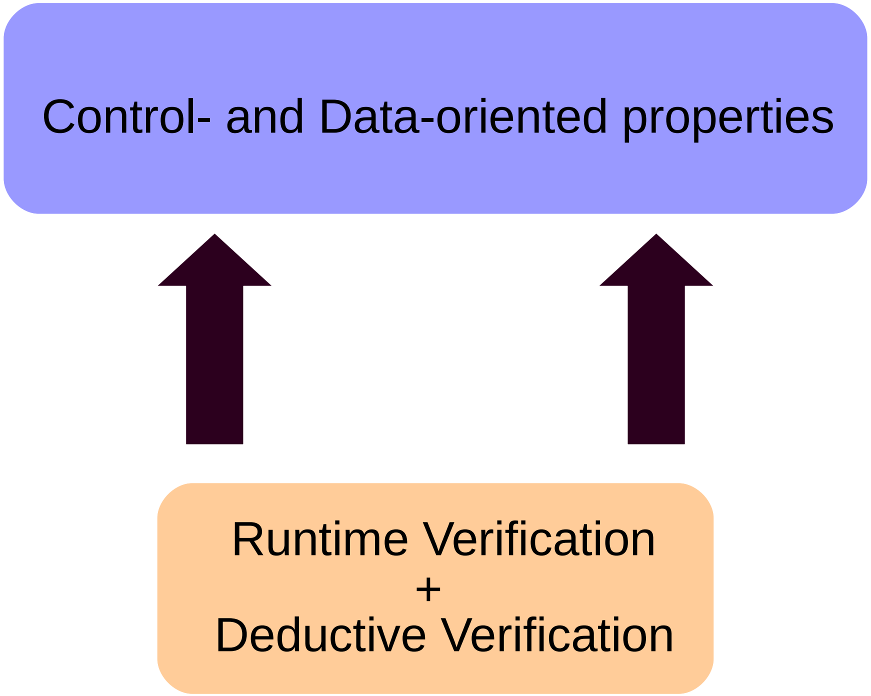


Control- and Data-oriented properties

Runtime Verification

Deductive Verification

Using the Techniques



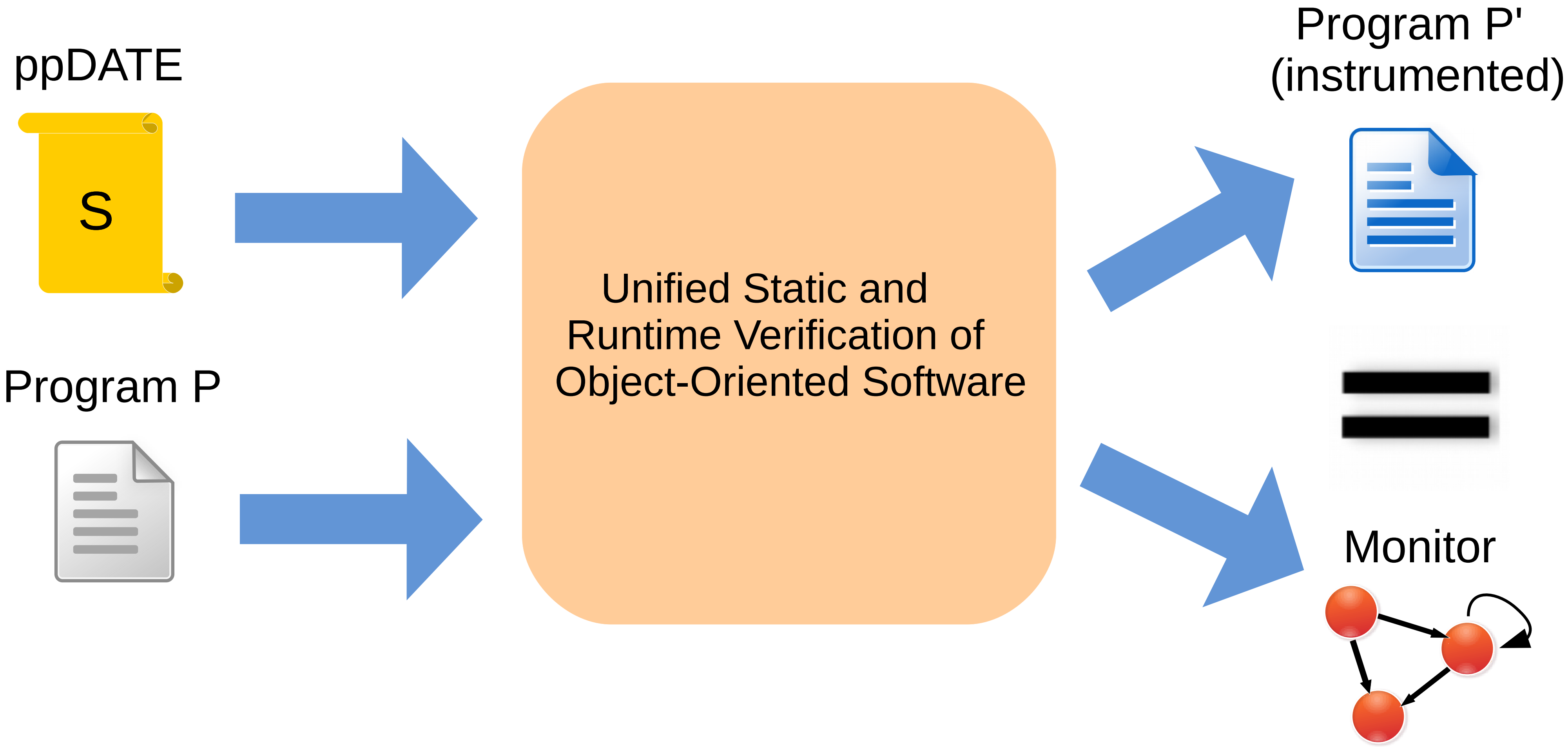
Control- and Data-oriented properties

Runtime Verification
+
Deductive Verification

Combination of Techniques

- Instead of adding abstractions for DV, check library method results at runtime
- Avoid verifying at runtime properties which are statically verified
- How to combine the techniques?

Verification Framework



ppDATE

S

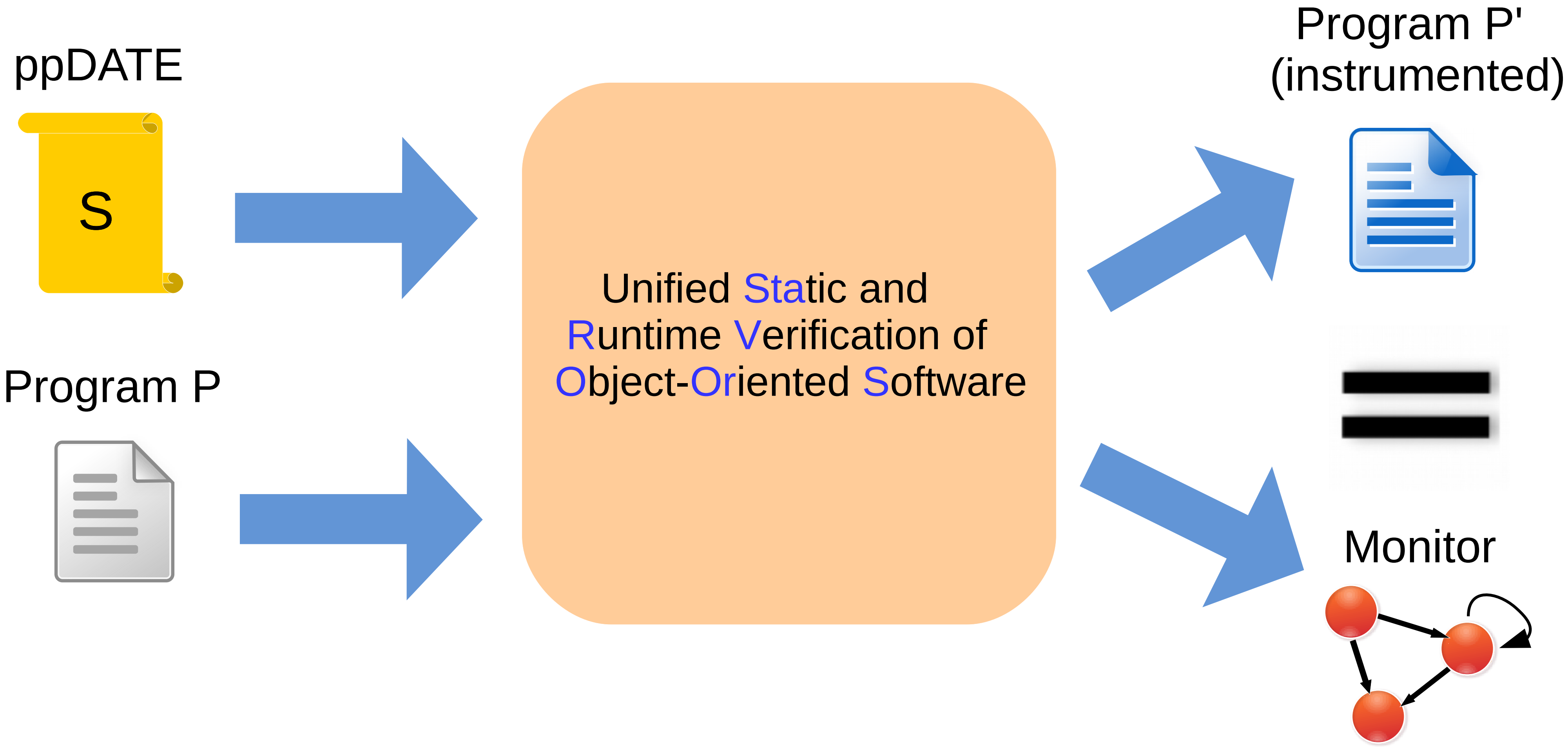
Program P

Unified Static and Runtime Verification of Object-Oriented Software

Program P' (instrumented)

Monitor

Verification Framework



ppDATE

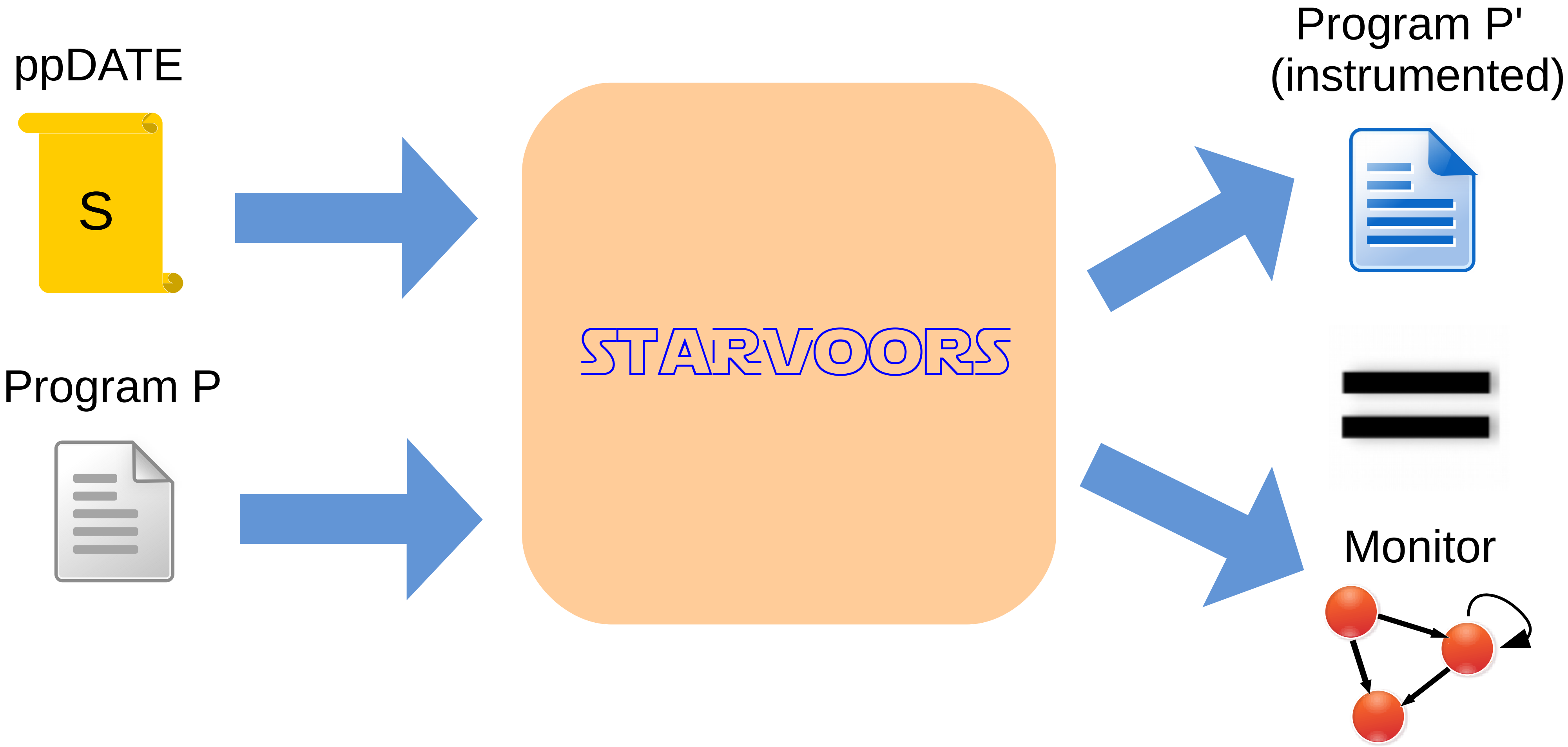
Program P

Unified Static and Runtime Verification of Object-Oriented Software

Program P' (instrumented)

Monitor

Verification Framework



ppDATE

Program P

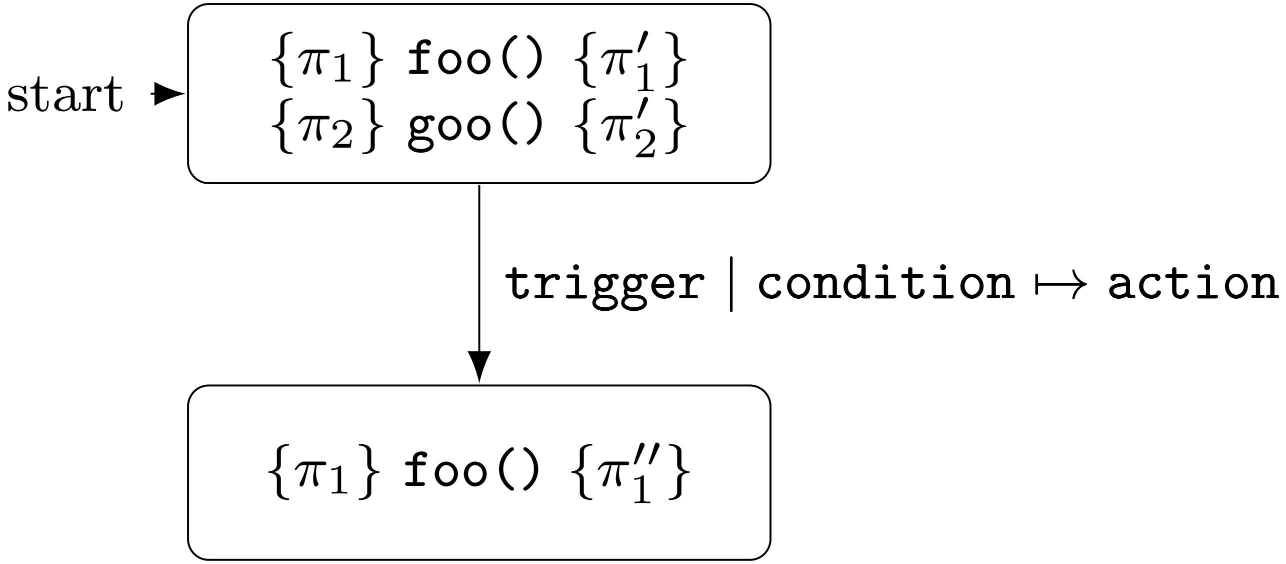
STARVOORS

Program P' (instrumented)

=

Monitor

Specification language: ppDATE



start

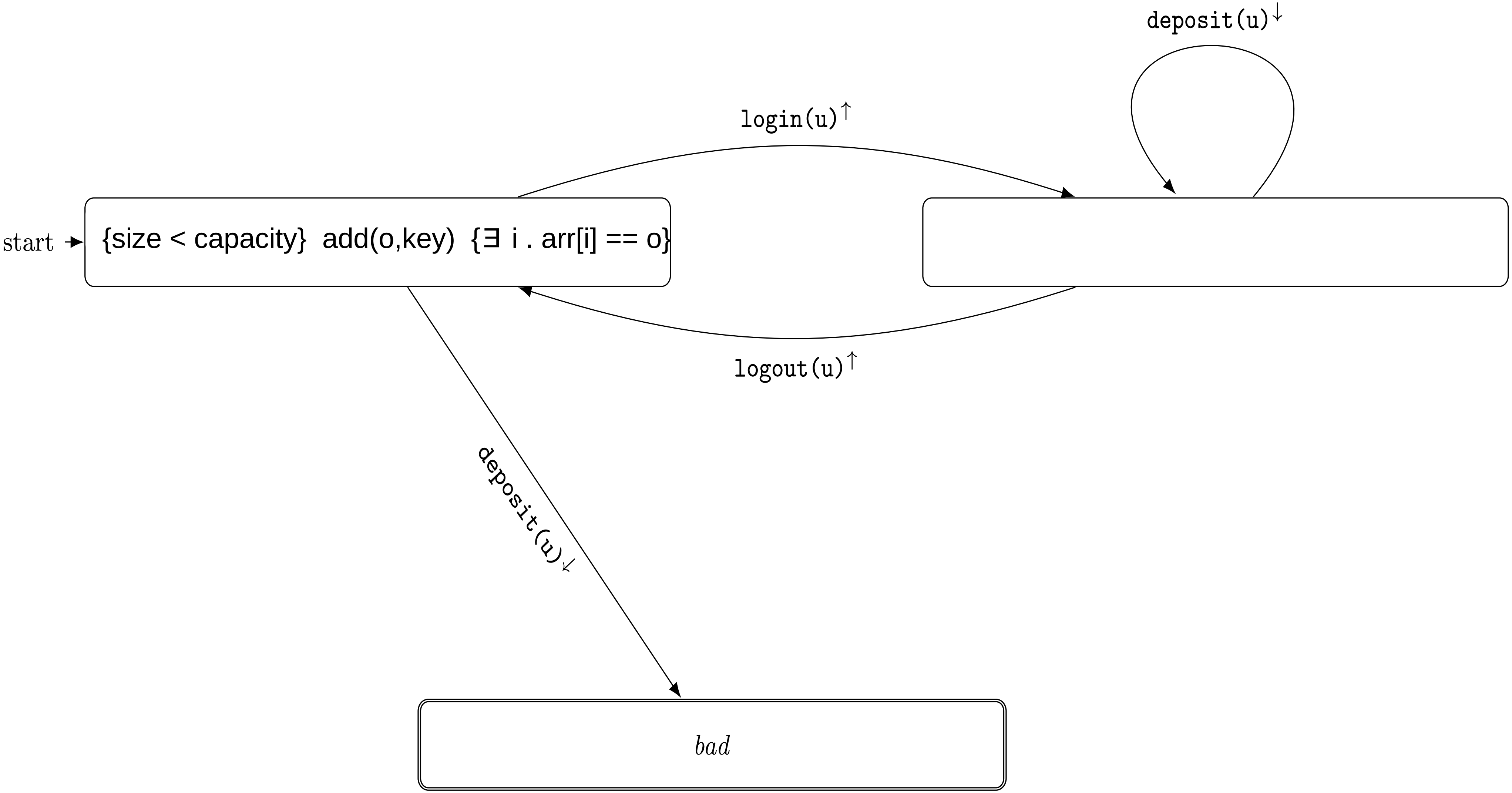
$\{\pi_1\}$ `foo()` $\{\pi'_1\}$
$\{\pi_2\}$ `goo()` $\{\pi'_2\}$

`trigger` $\mid$ `condition` $\mapsto$ `action`

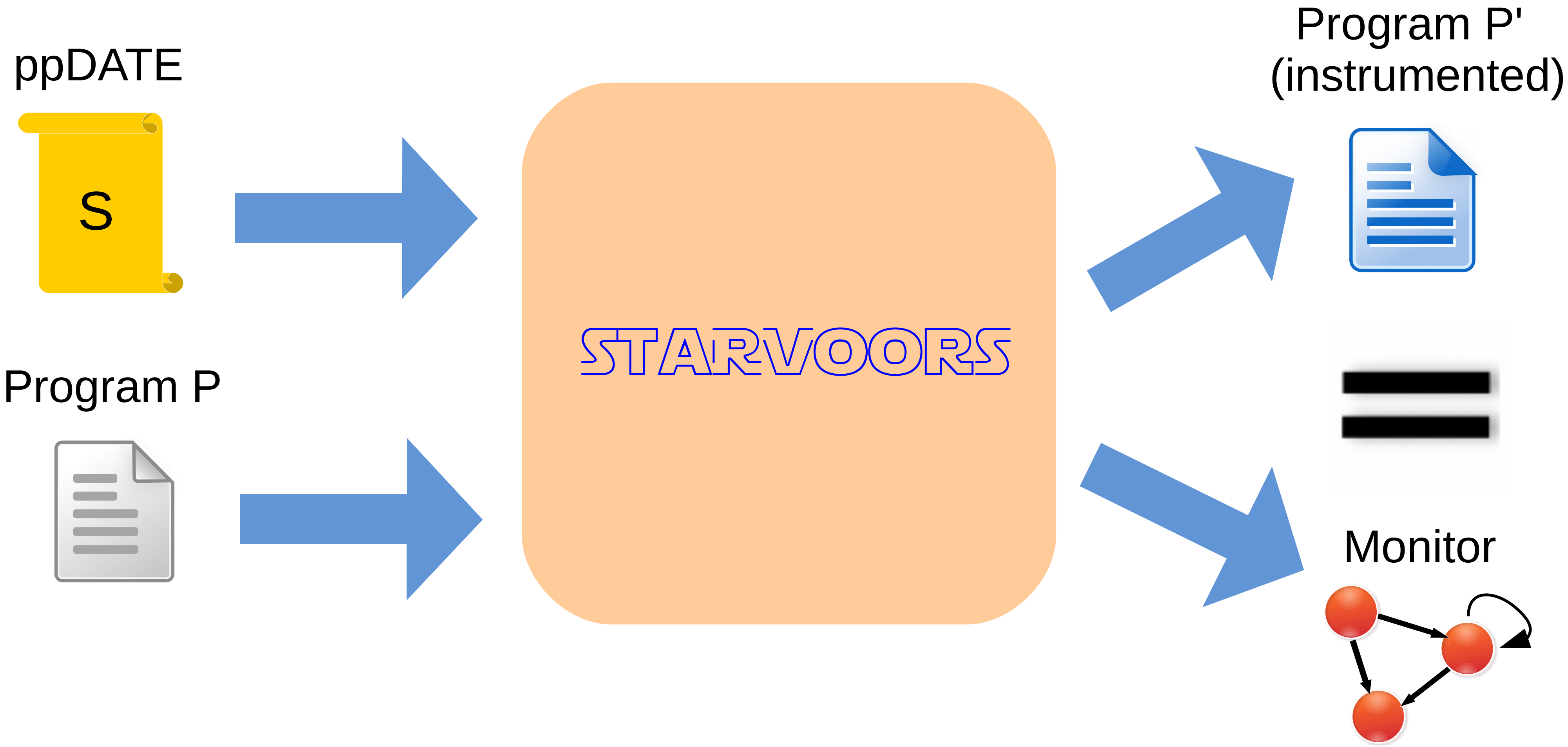
$\{\pi_1\}$ `foo()` $\{\pi''_1\}$

- In general:
  - event-triggered transitions
  - Zero or more Hoare triples in each state of the automata
  - Normal, acceptance and bad states for describing automata
  - Parallel automata, communication
  - Templates, ppDATEs creation

Example


deposit(u)↓
login(u)↑
start
{size < capacity} add(o,key) {∃ i . arr[i] == o}
logout(u)↑
deposit(u)↓
bad

Verification Framework



ppDATE

S

Program P

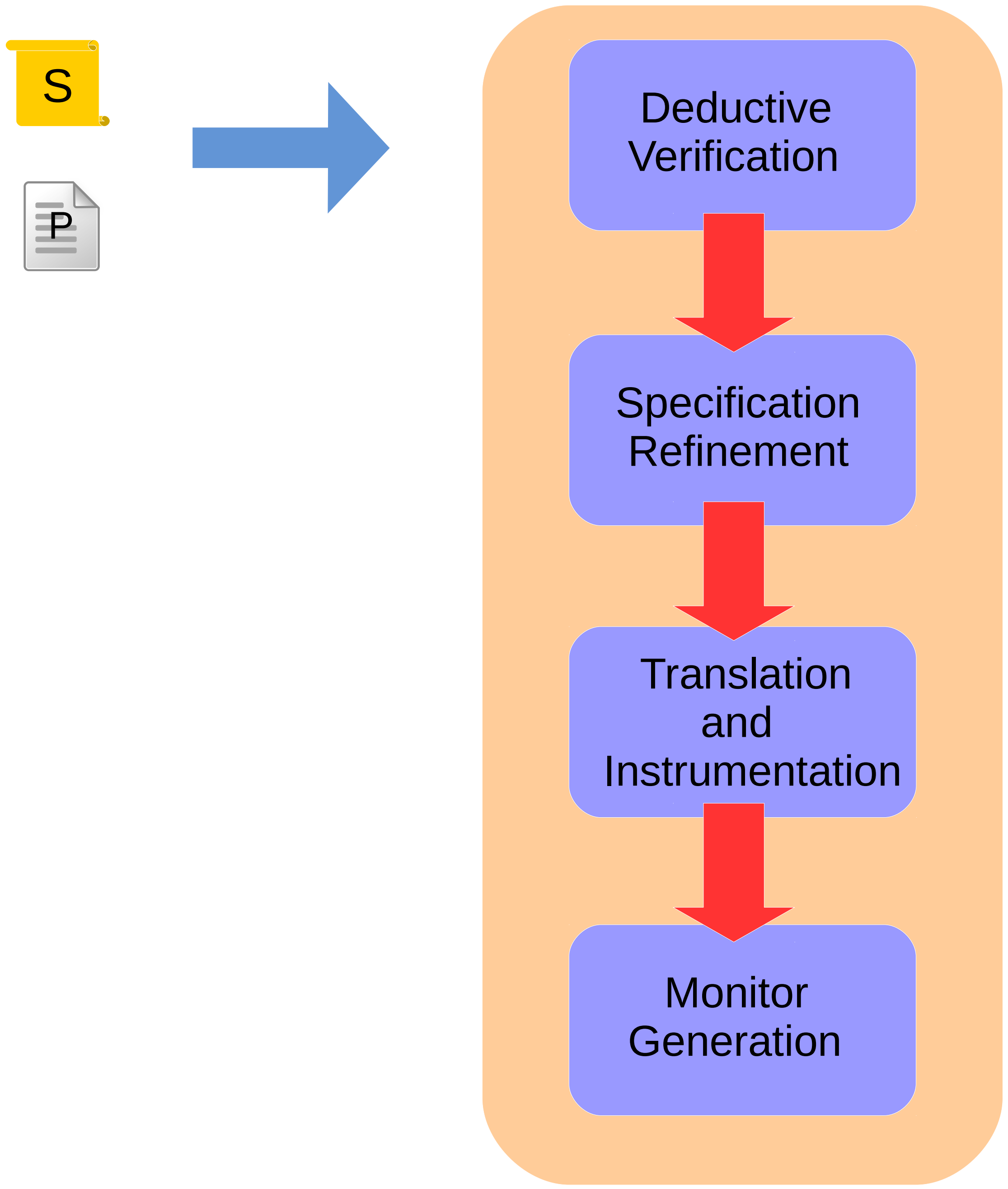
STARVOORS

Program P'
(instrumented)

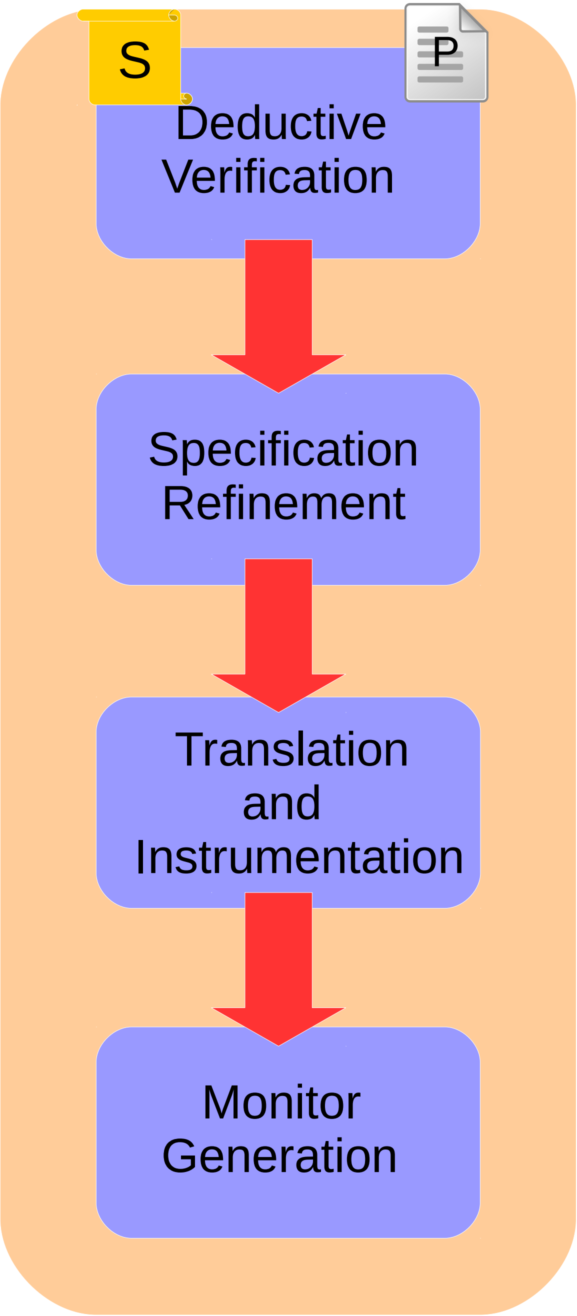
=

Monitor

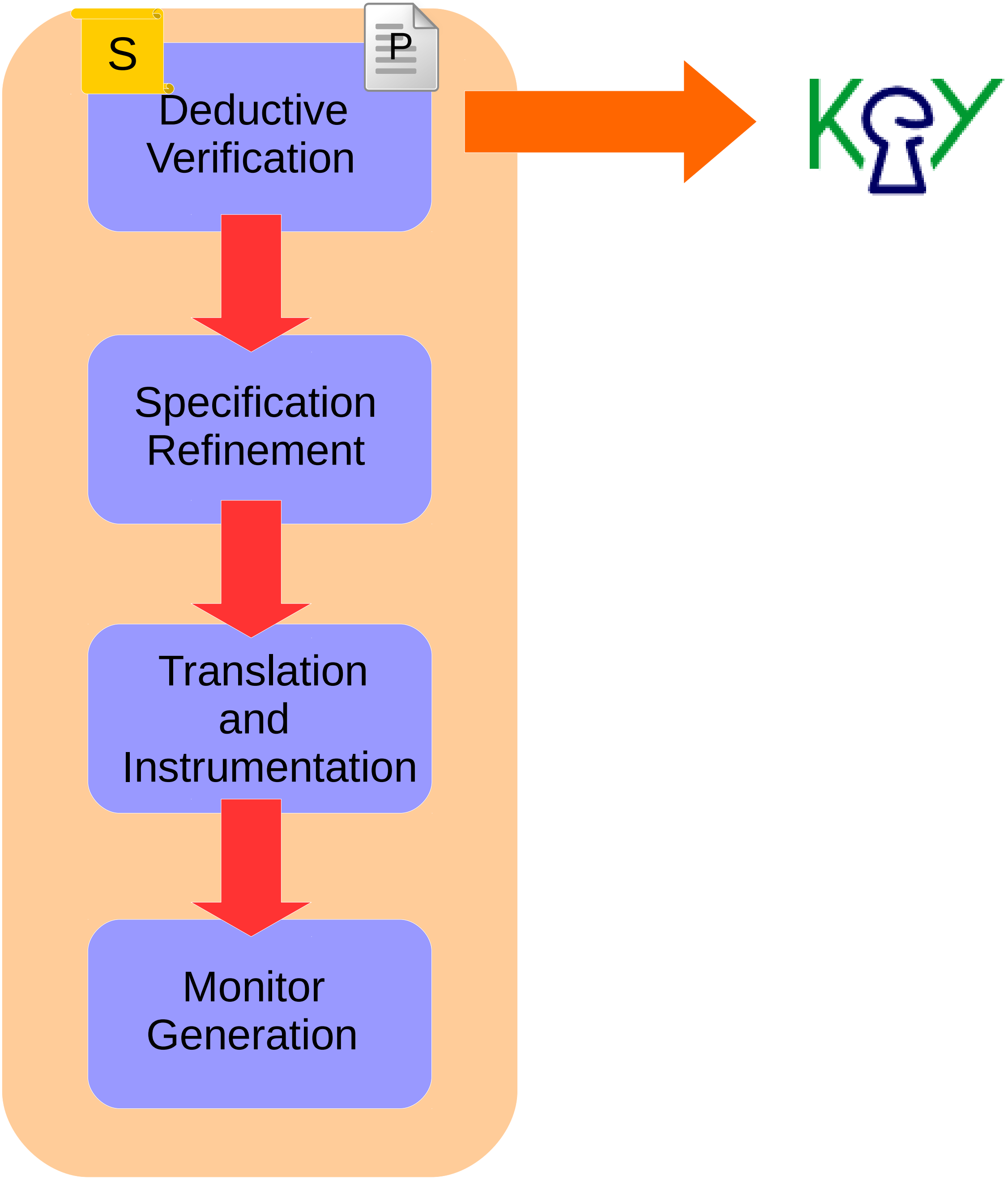
High-level description of StaRVOOrS



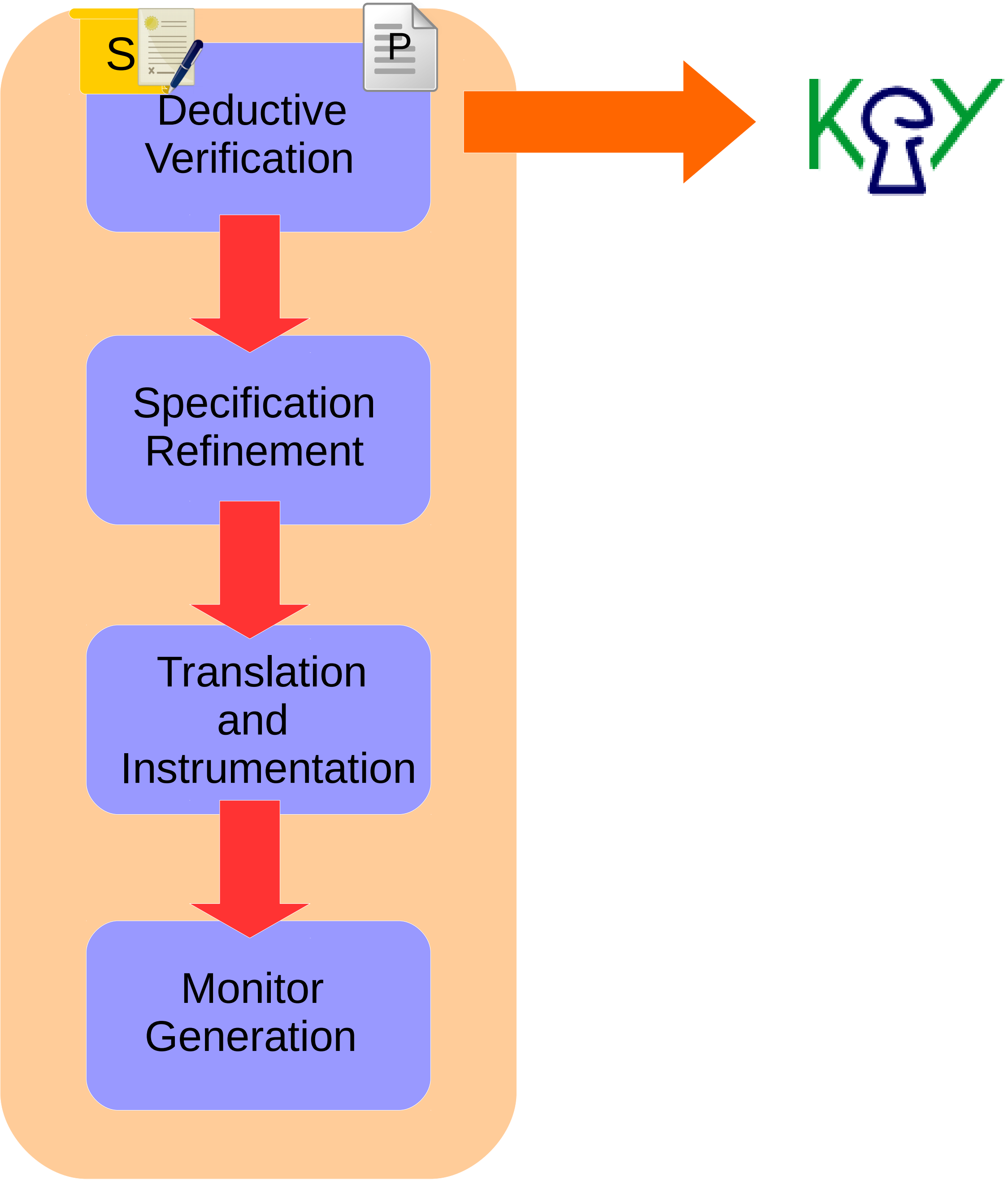
High-level description of StaRVOOrS



High-level description of StaRVOOrS



High-level description of StaRVOOrS



S

P

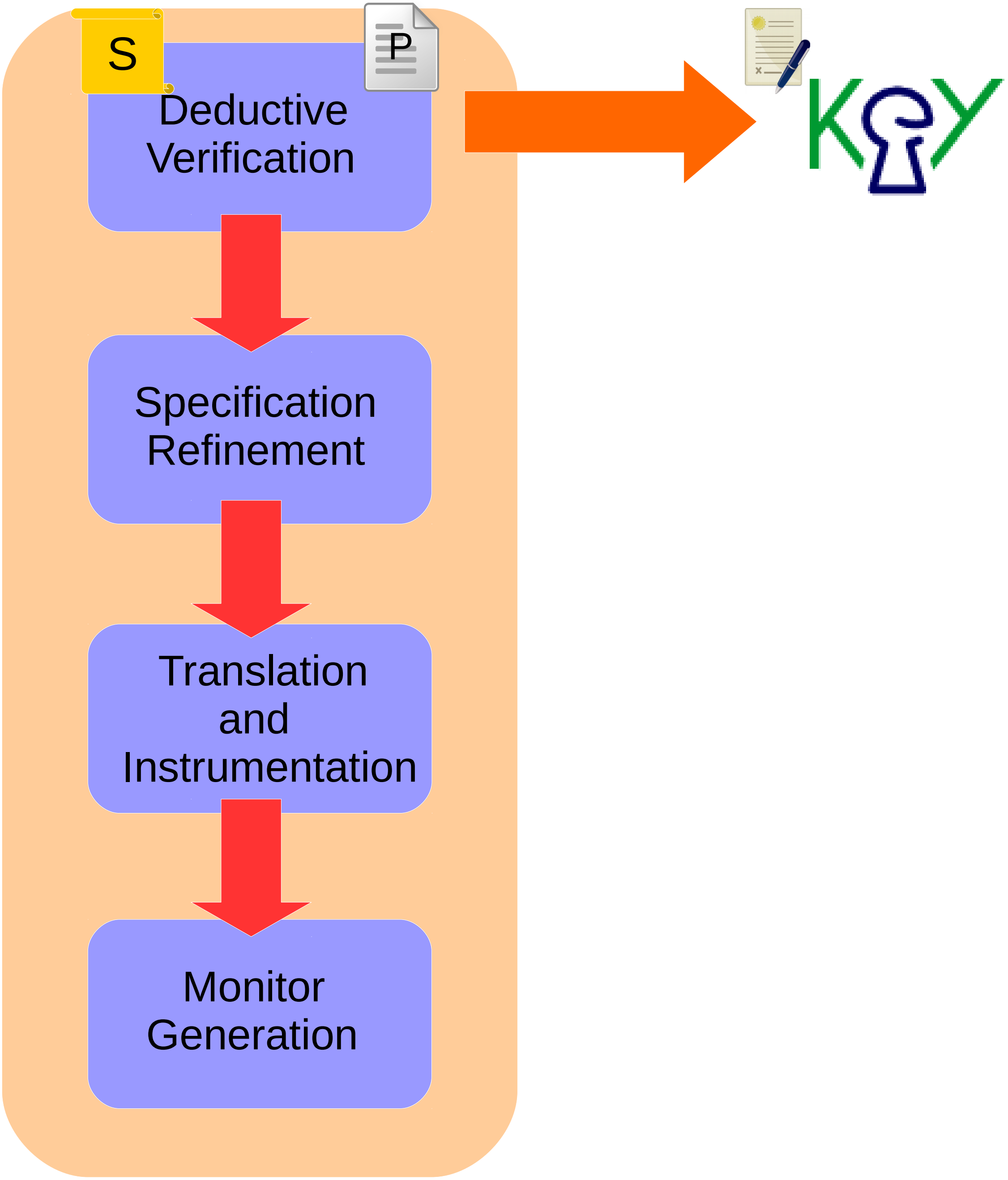
Deductive Verification

Specification Refinement

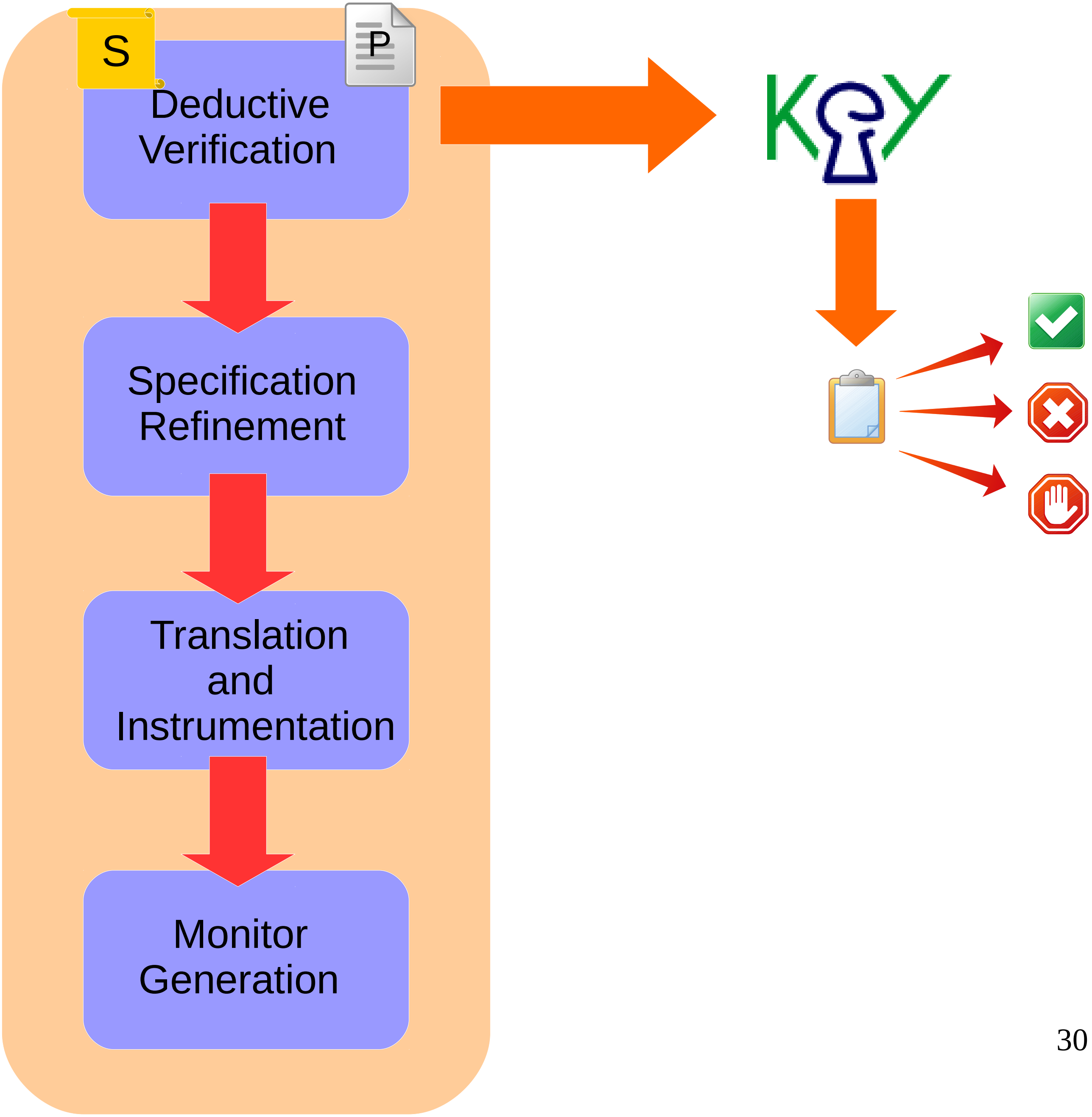
Translation and Instrumentation

Monitor Generation

High-level description of StaRVOOrS



High-level description of StaRVOOrS

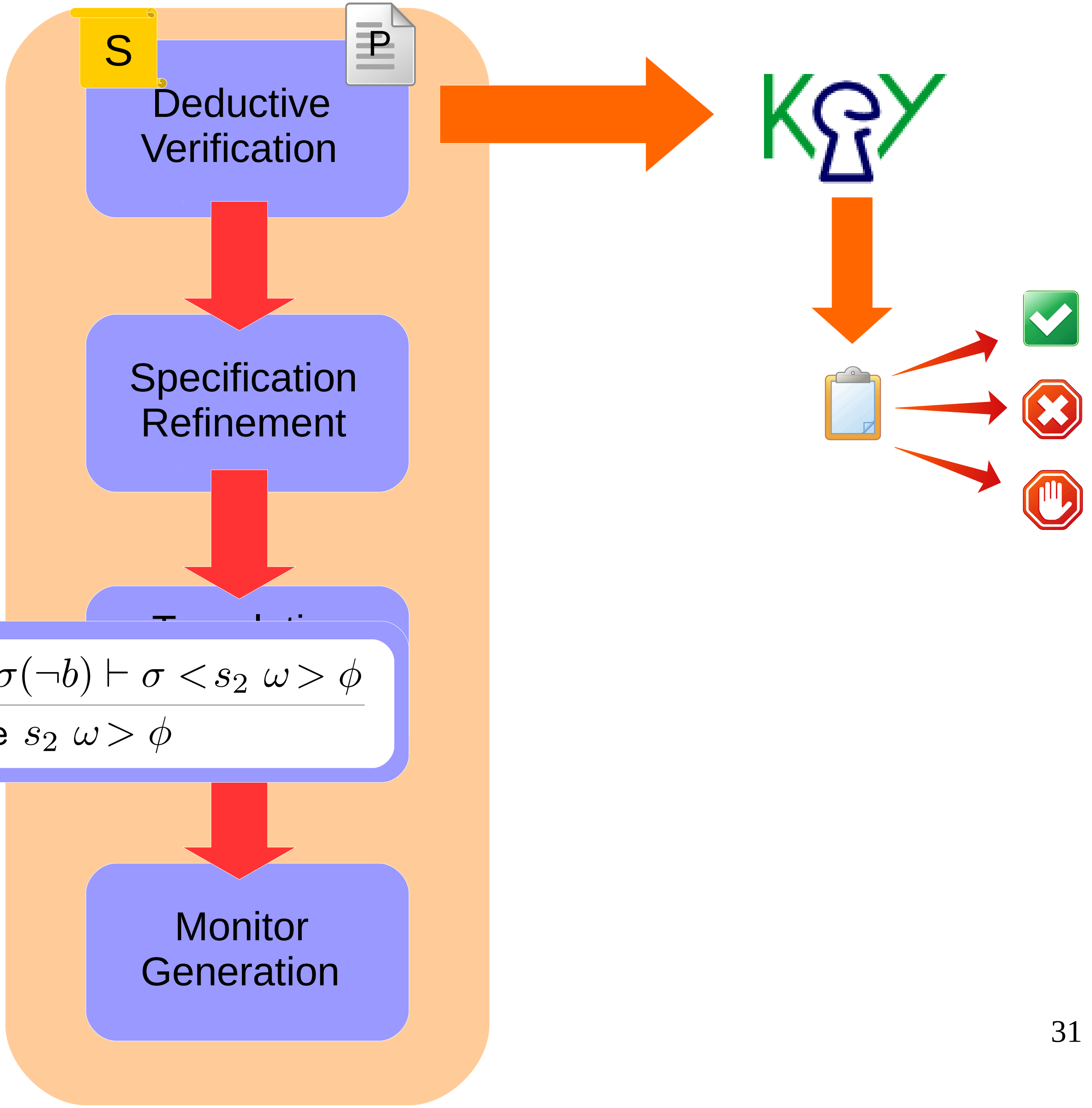


High-level description of StaRVOOrS

```
foo() {
        if (cond) {
            E1 ;
        } else {
            E2 ;
        }
}
```

{ P } foo() { Q }

$$\frac{\Gamma, \sigma(b) \vdash \sigma < s_1\ \omega > \phi \qquad \Gamma, \sigma(\neg b) \vdash \sigma < s_2\ \omega > \phi}{\Gamma \vdash \sigma < \texttt{if}\ b\ s_1\ \texttt{else}\ s_2\ \omega > \phi}$$



Deductive Verification

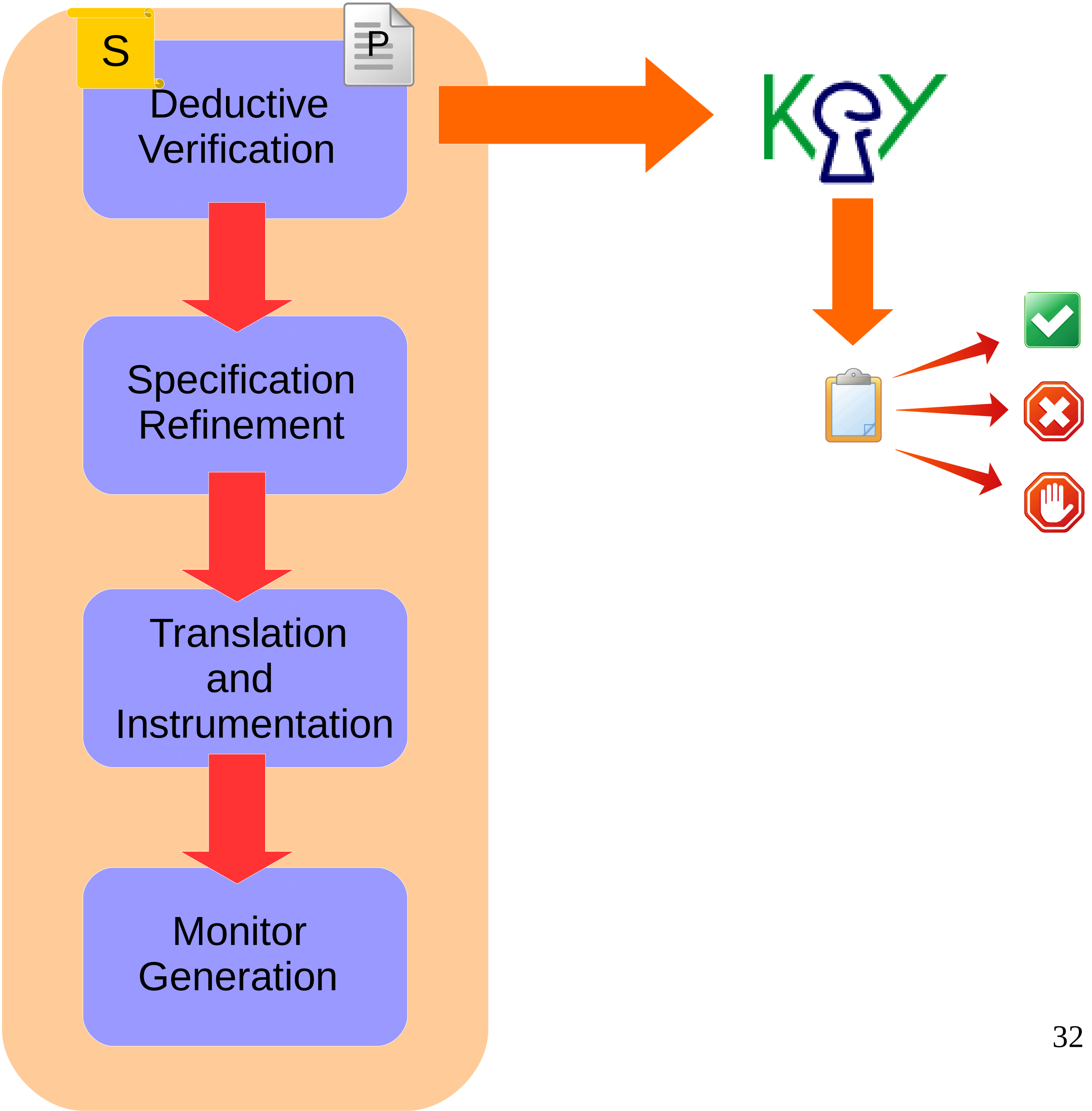
Specification Refinement

Monitor Generation

High-level description of StaRVOOrS

```
foo() {
      if (cond) {
         E1 ;   <E1>Q
      } else {
         E2 ;   <E2>Q
      }
}
```

{ P } foo() { Q }



S

P

Deductive Verification

Specification Refinement

Translation and Instrumentation

Monitor Generation

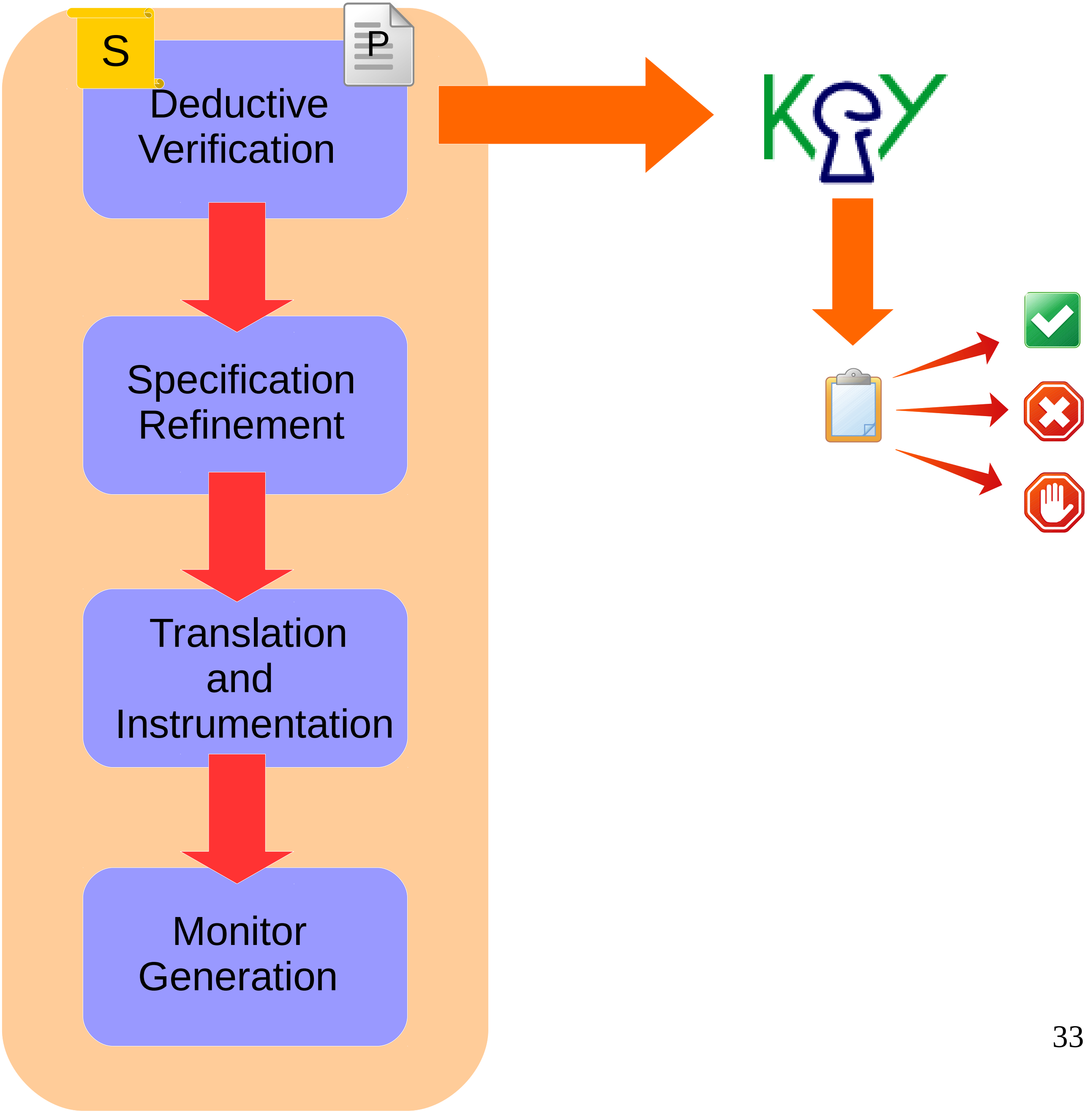
KeY

High-level description of StaRVOOrS

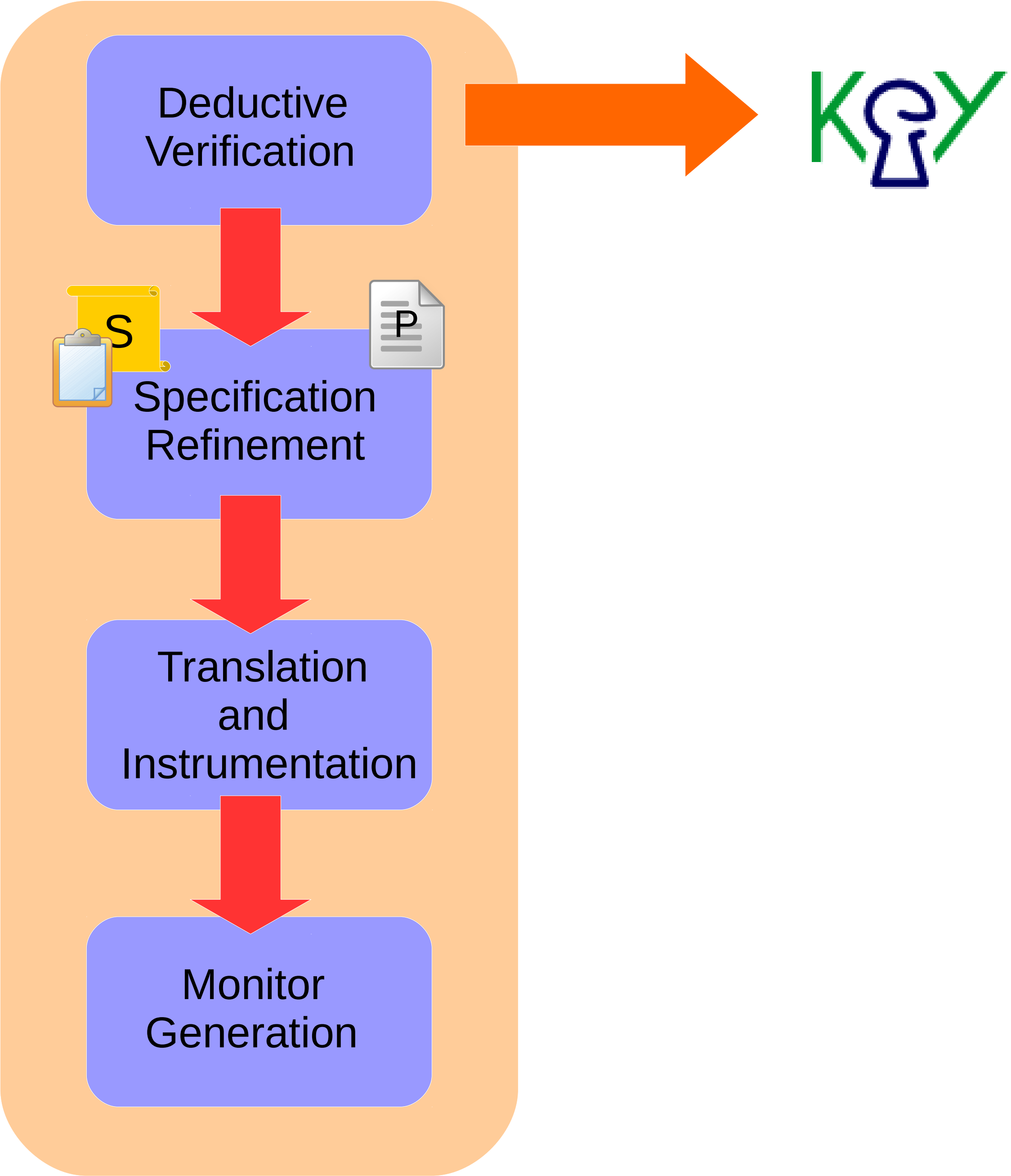
```
foo() {
        if (cond) {
            E1 ;  <E1>Q
        } else {
            E2 ;  <E2>Q
        }
}
```

{ P } foo() { Q }

Q(E1)	Q(E2)	KeY
✓	✓	✅
✓	✗	✋
✗	✓	✋
✗	✗	❌



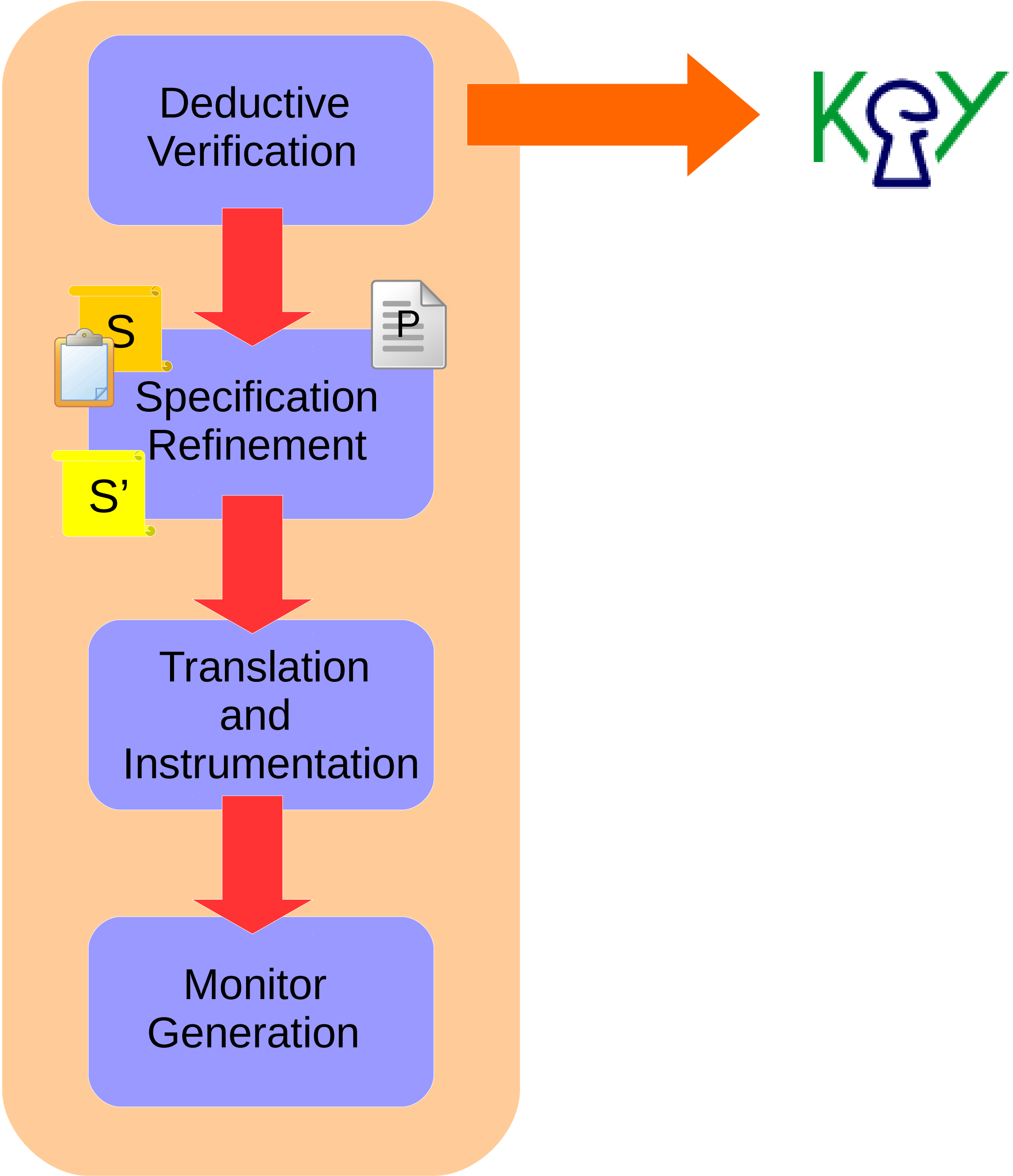
High-level description of StaRVOOrS



{ P } foo() { Q }

Q(E1)	Q(E2)	KeY
✓	✓	☑
✓	✗	✋
✗	✓	✋
✗	✗	⊗

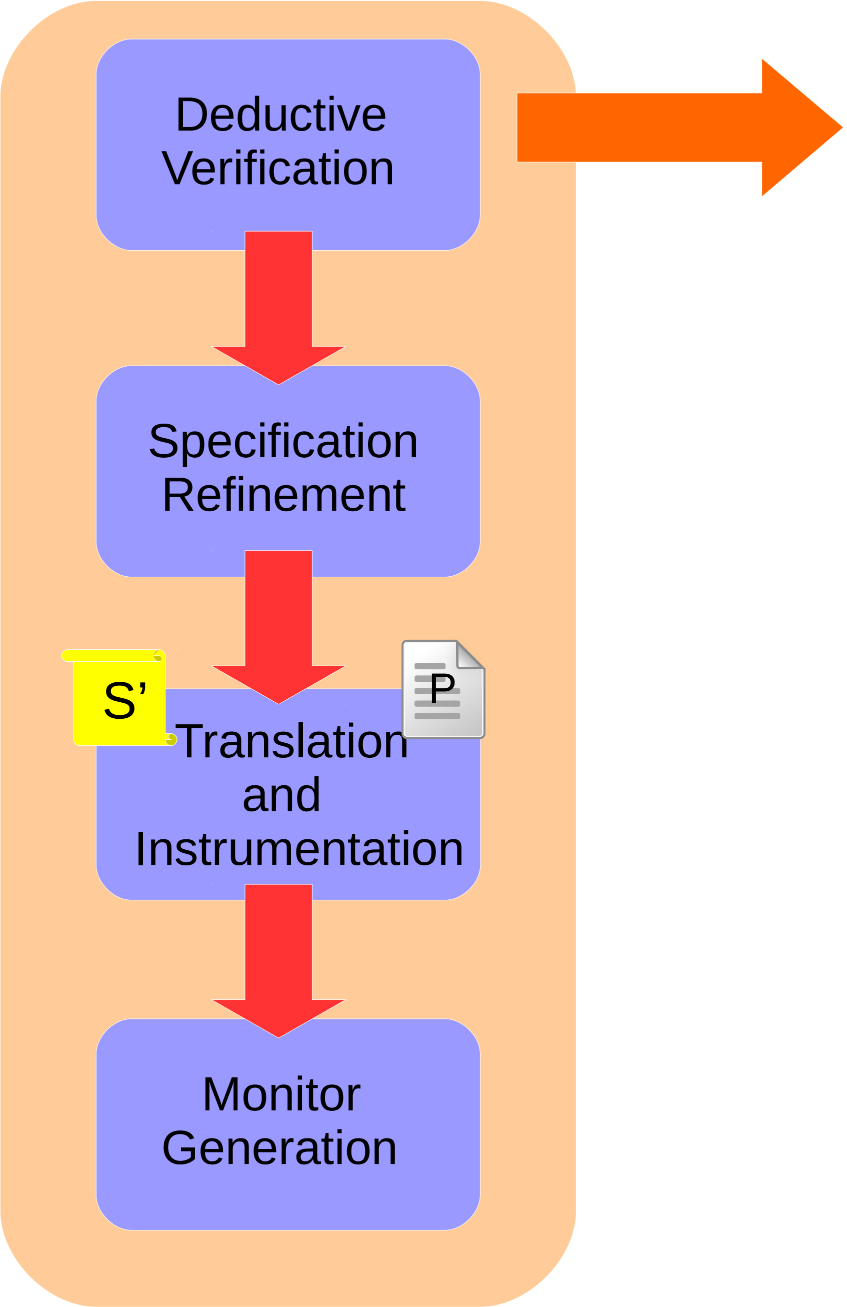
High-level description of StaRVOOrS



{ P /\ !cond } foo() { Q }

	Q(E1)	Q(E2)	KeY
	✓	✓	✅
!cond	✓	✗	✋
	✗	✓	✋
	✗	✗	⊗

High-level description of StaRVOOrS

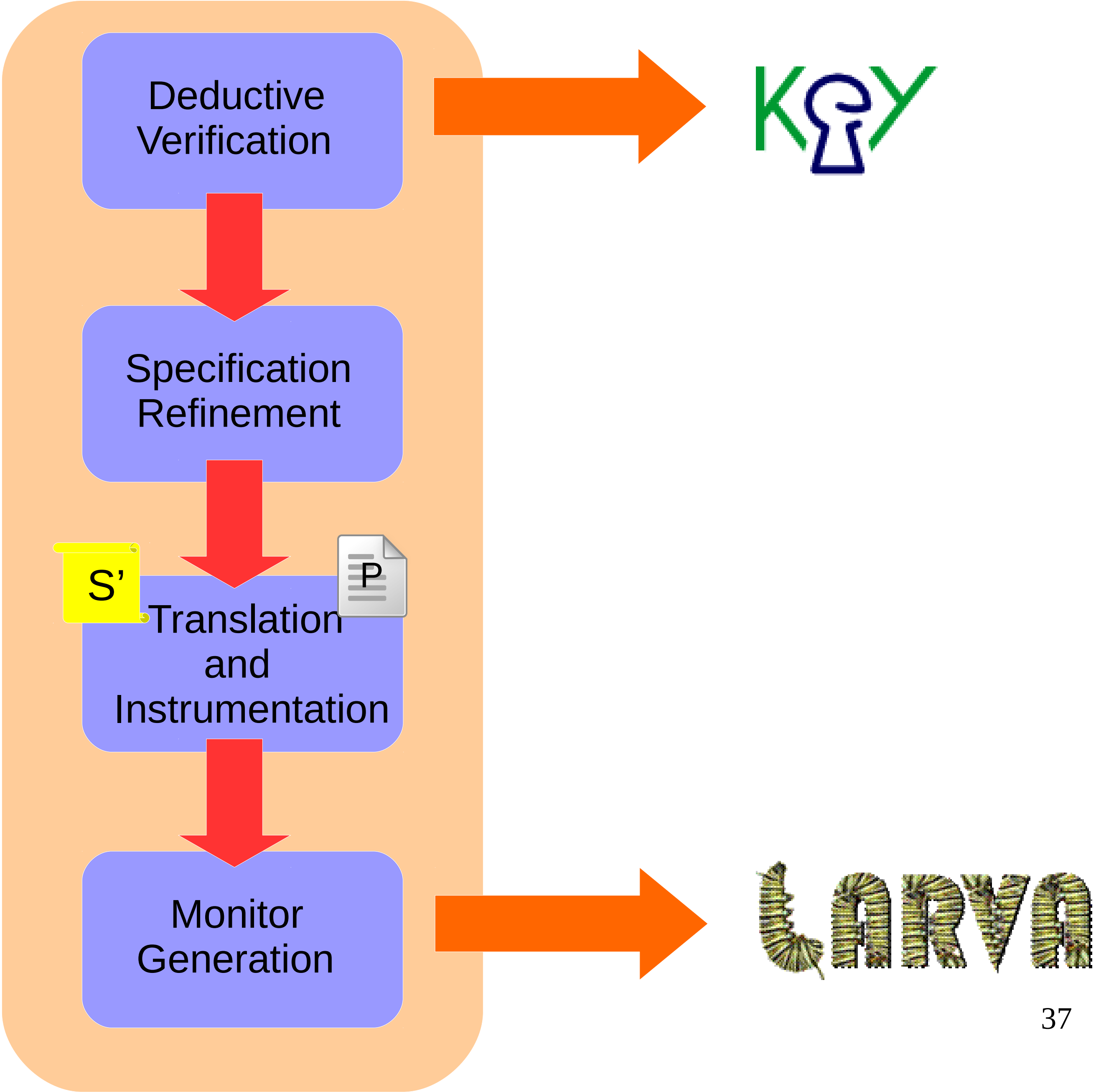

Deductive Verification

Specification Refinement

Translation and Instrumentation

Monitor Generation

High-level description of StaRVOOrS



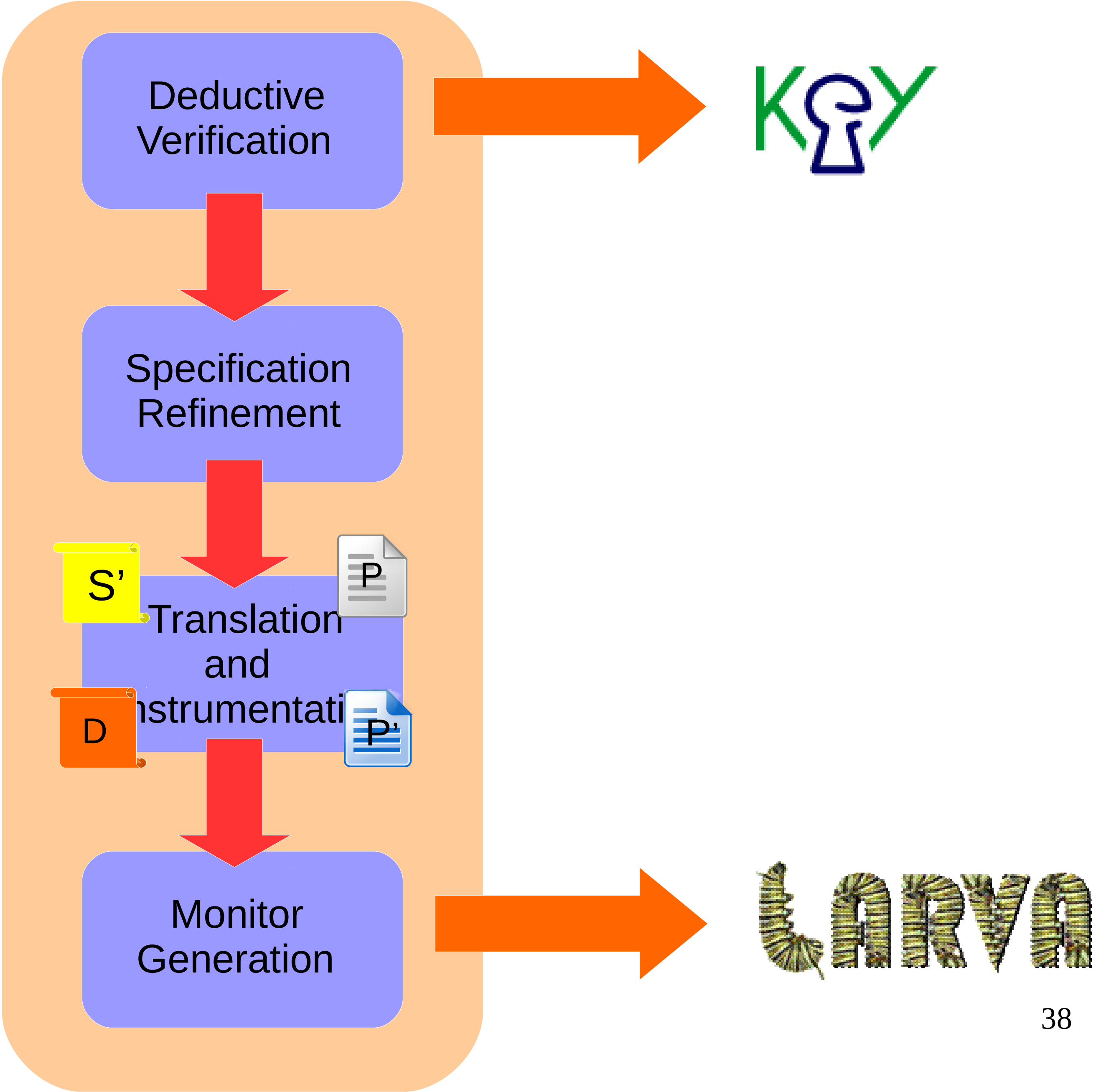
Deductive Verification

Specification Refinement

Translation and Instrumentation

Monitor Generation

High-level description of StaRVOOrS



Deductive Verification

Specification Refinement

S'

P

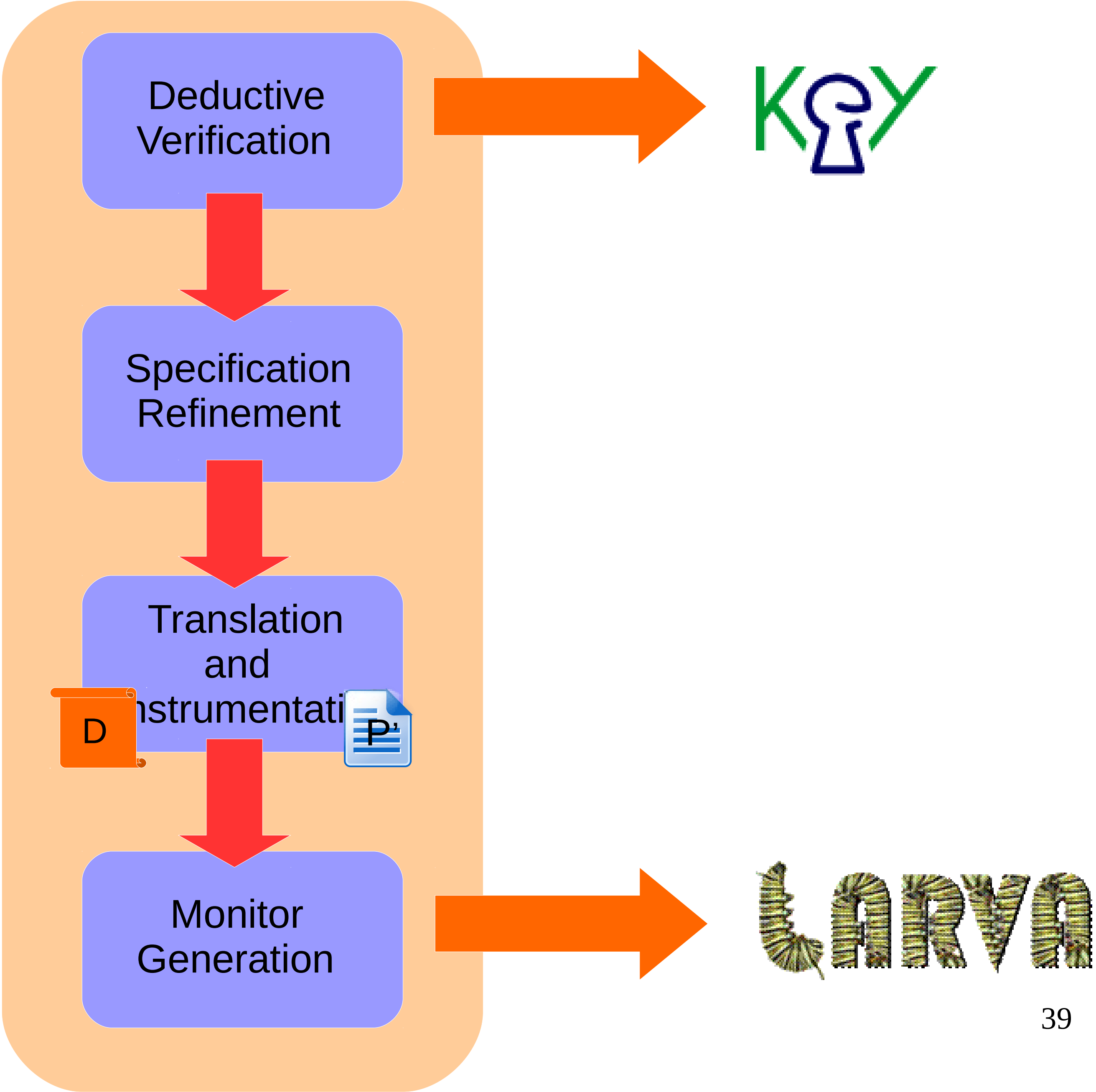
Translation and nstrumentati

D

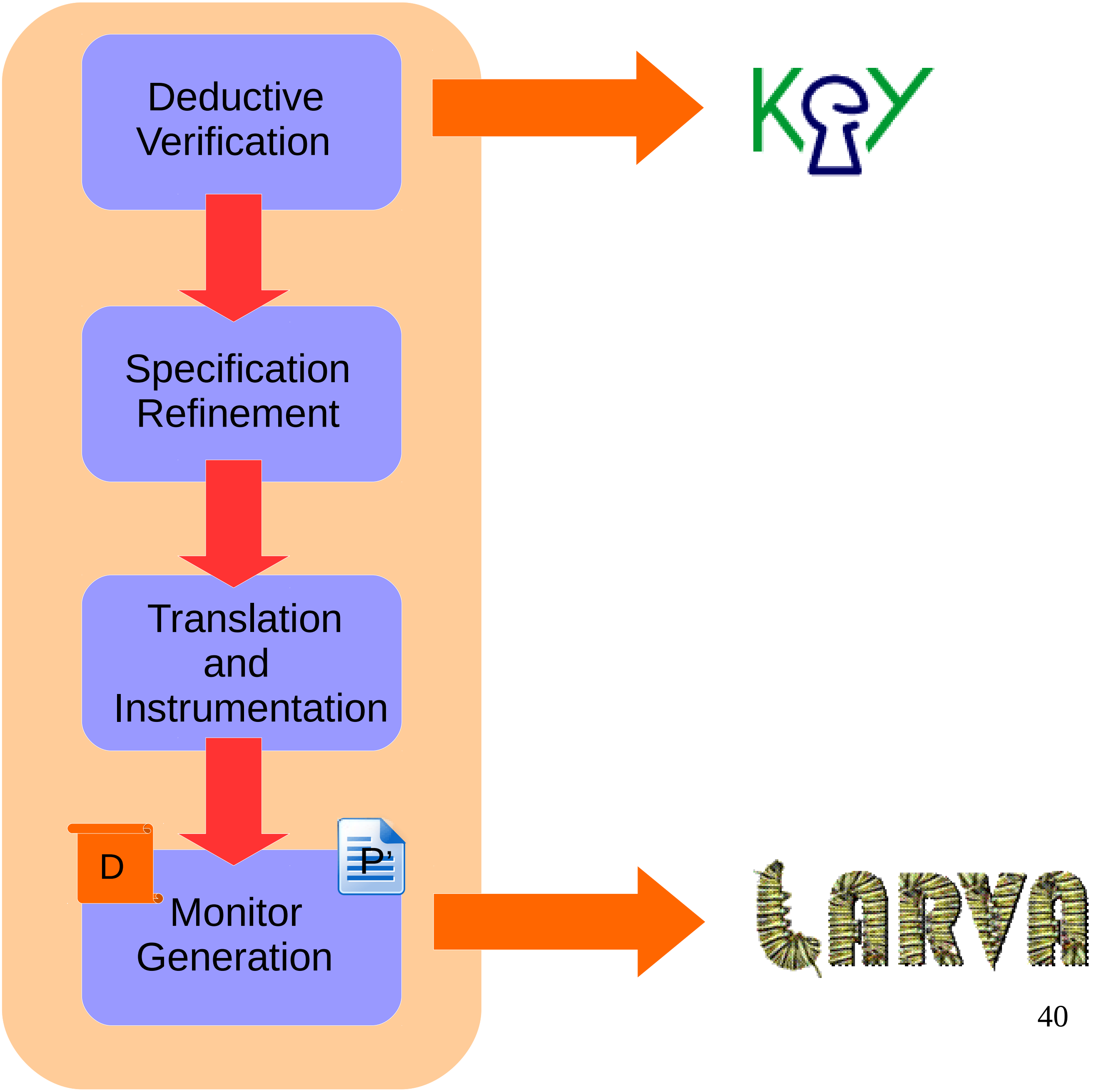
P'

Monitor Generation

High-level description of StaRVOOrS



High-level description of StaRVOOrS



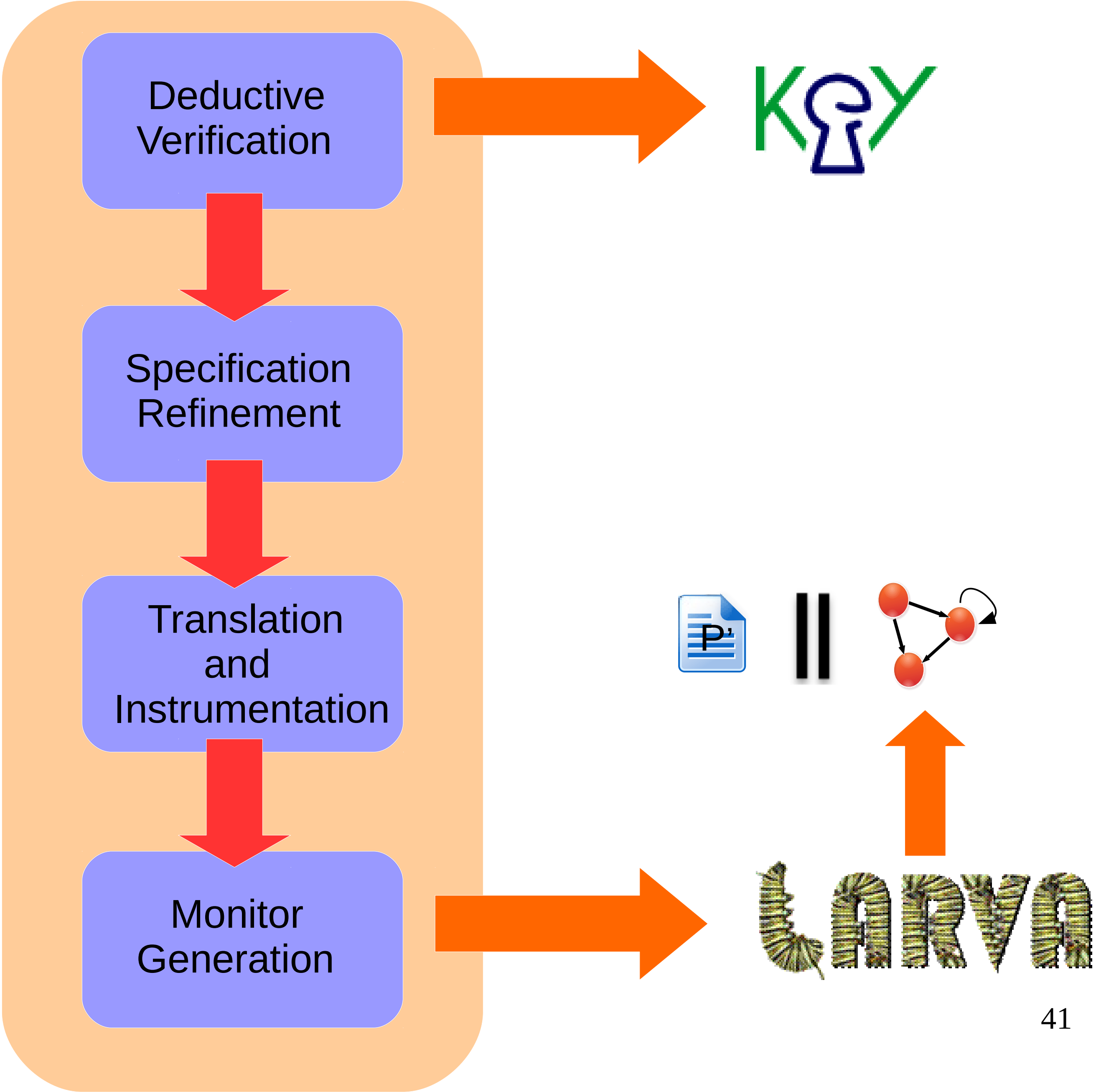
Deductive Verification

Specification Refinement

Translation and Instrumentation

Monitor Generation

High-level description of StaRVOOrS



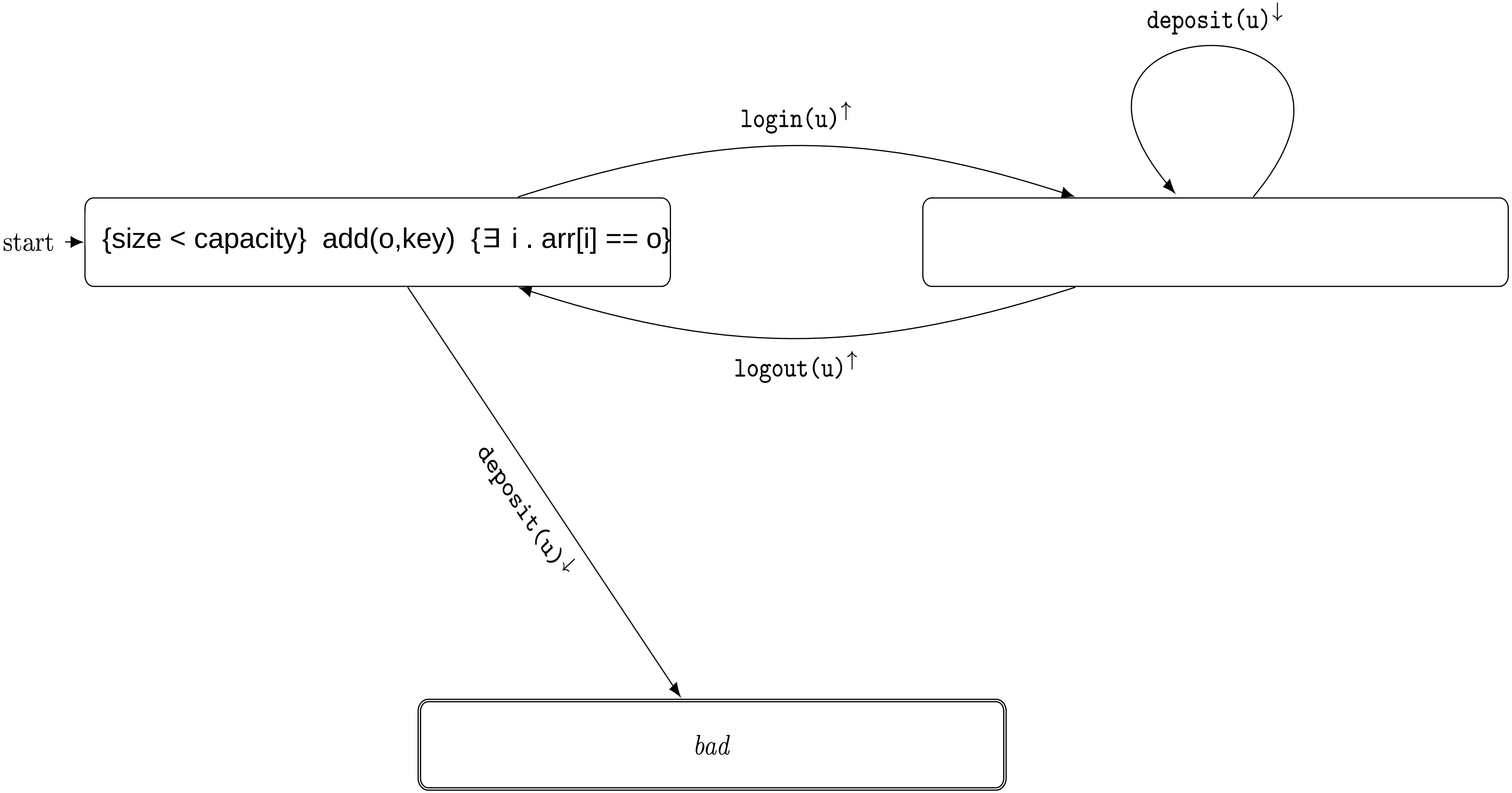
Deductive Verification

Specification Refinement

Translation and Instrumentation

Monitor Generation

Demo


deposit(u)↓
login(u)↑
start
{size < capacity} add(o,key) {∃ i . arr[i] == o}
logout(u)↑
deposit(u)↓
bad

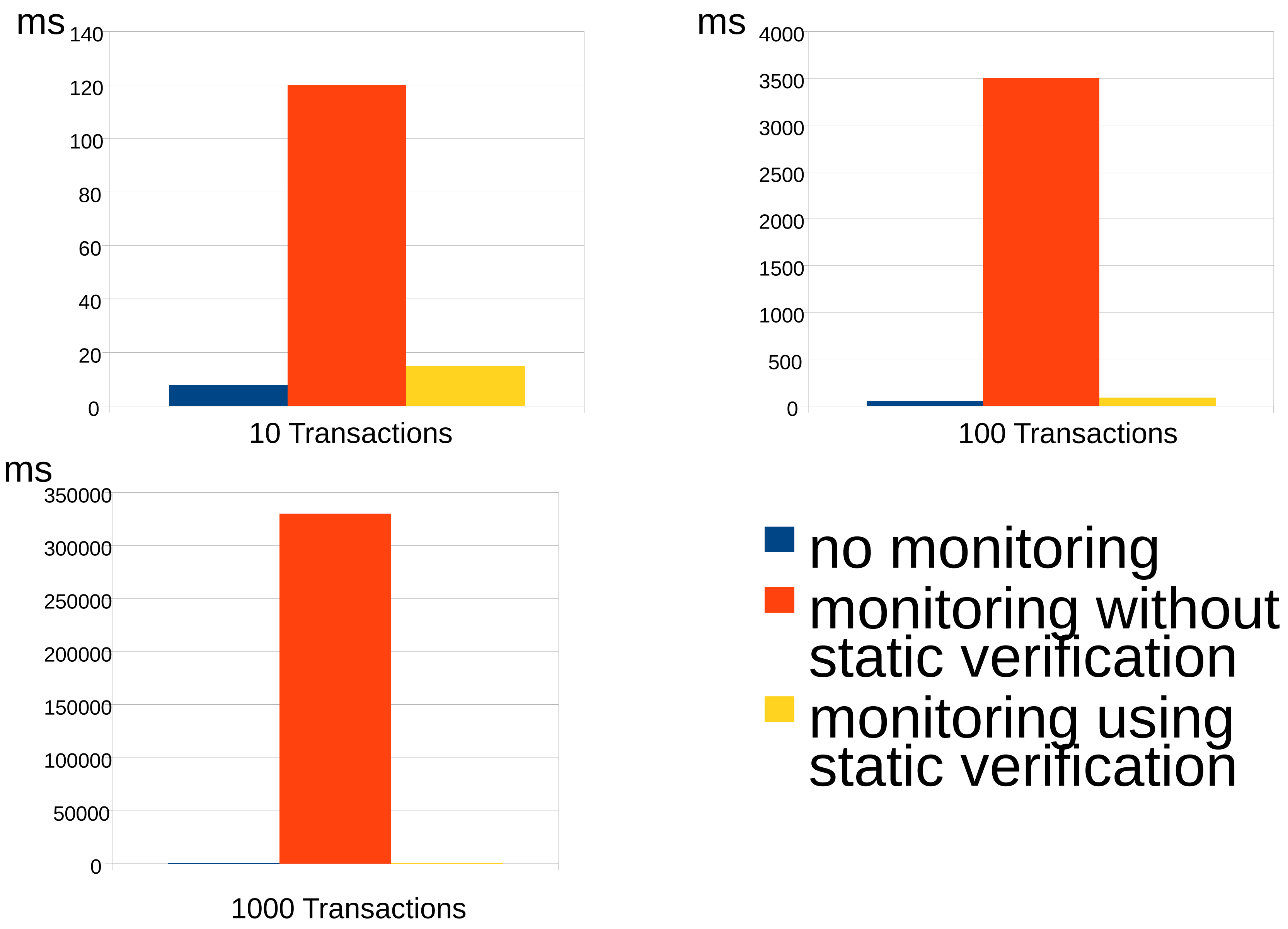
StaRVOOrS Implementation

- https://github.com/starvoors/StaRVOOrS-tool
- http://cse-212294.cse.chalmers.se/starvoors
- Fully automatic

Mondex Case Study

- Standard formal methods benchmark
- Electronic purse application
- Financial transaction move funds between accounts
- Multi-step message exchange protocol

Experimentation

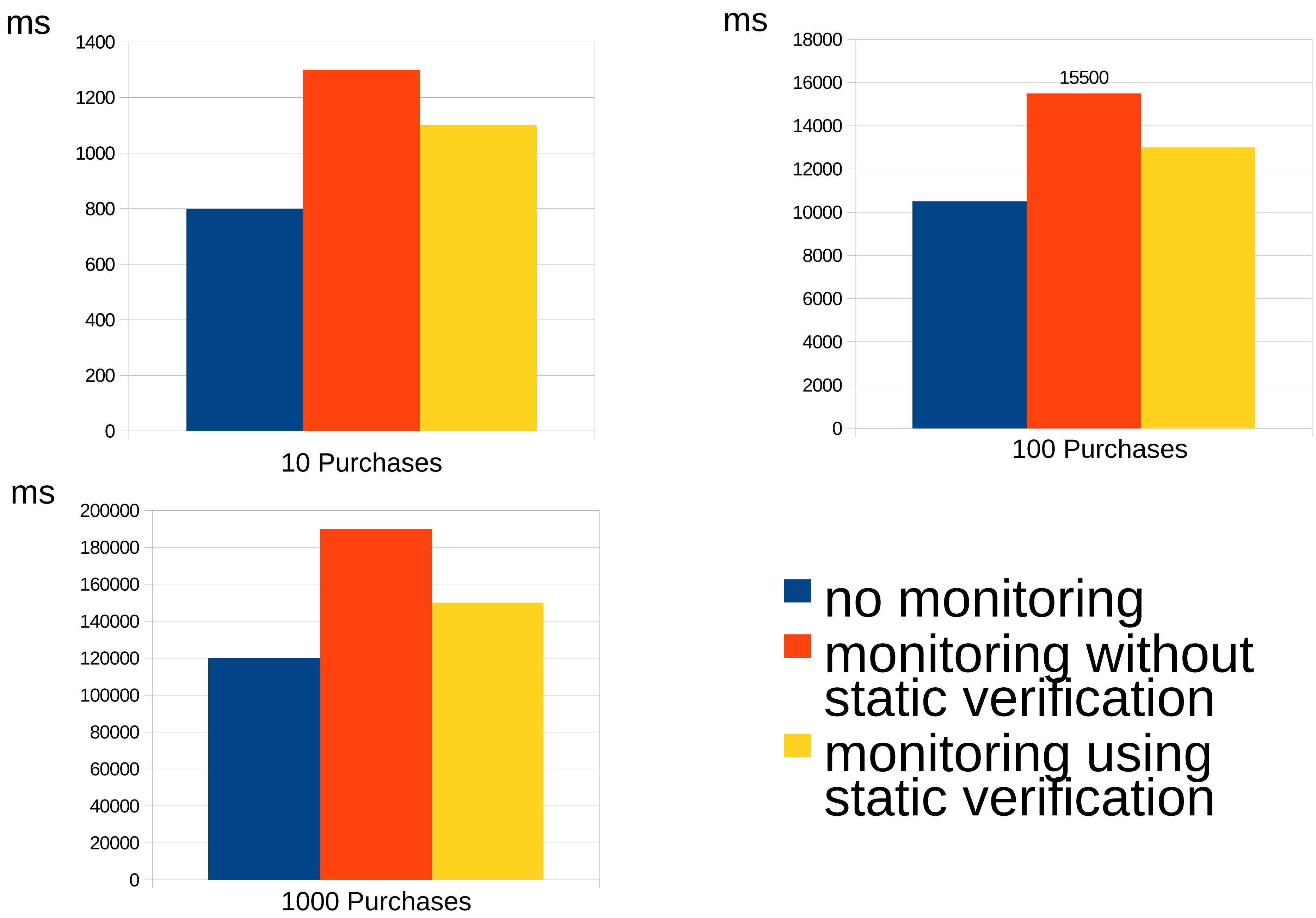


SoftSlate Case Study

- Online shopping cart application
- 4-steps purchases checkout

  (address,shipping,credit card,confirmation)
- User has to be logged for checkout

Experimentation



Future of StaRVOOrS

- Further static optimisations to the runtime checking
- Analysis of state invariants
- Expanding the framework towards testing
- Adding timers to ppDATE

Kiitos
Ευχαριστω
Dziękuję
Hvala
有
Obrigado
谢谢
有り難う
Danke
Terim
Tack
תודה
Merci
ข
Thank you
Gracias
謝
Grazie
شكرا
謝謝
감사합니다
ありがとう
Спасибі
Спасибо
Mulţumesc
Asante
Dě
e
Благодаря