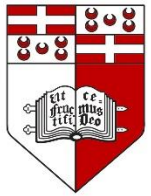
Harnessing Sophisticated Assertion Checking Through Runtime Testing

Dr Christian Colombo

Intelligent Testing™

17th March 2014



University of Malta
L-Università ta' Malta

Bank System Example

Bank System Example

DEPOSIT

WITHDRAW

Bank System Example

DEPOSIT

WITHDRAW

Bank System Example

DEPOSIT

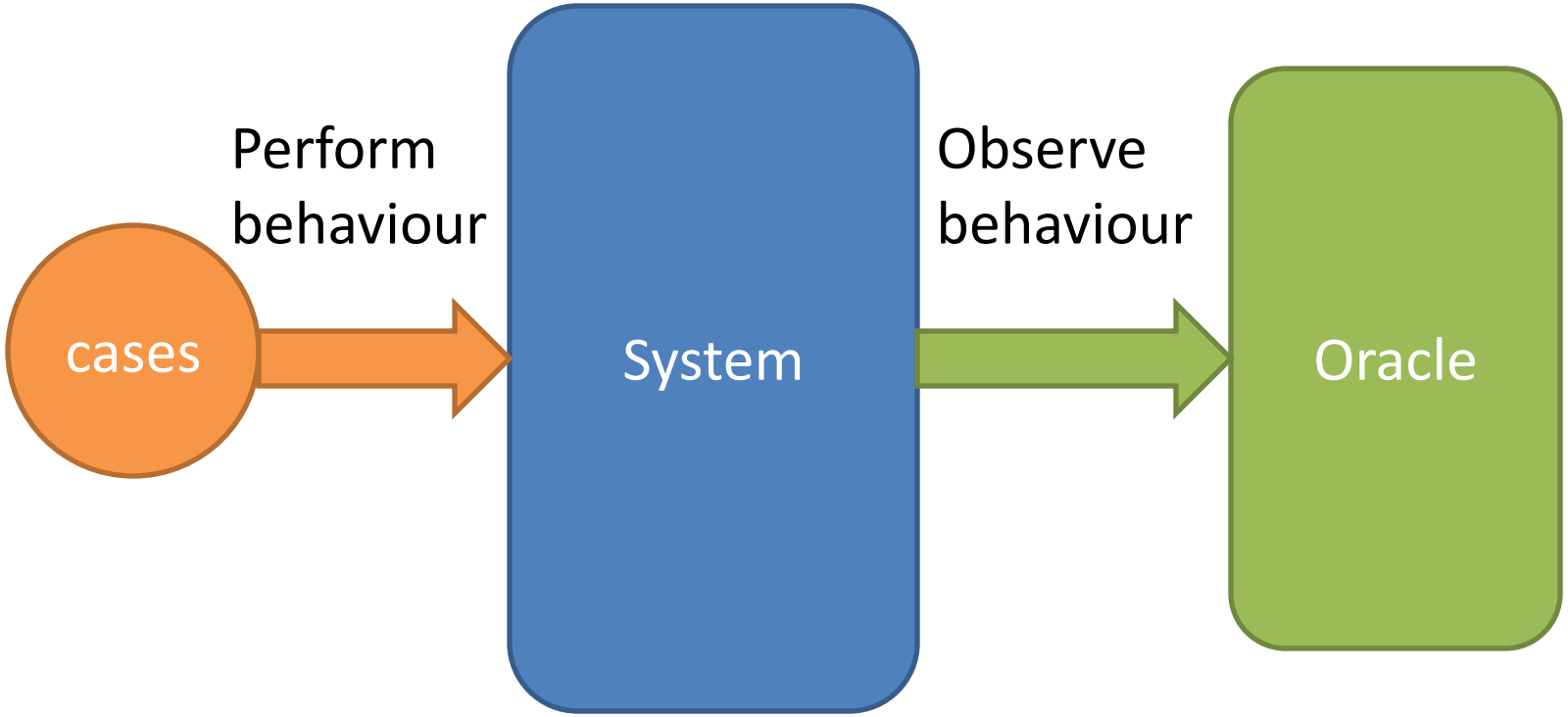
How do we make sure the balance gets updated correctly

And it never goes below zero?

BALANCE?

WITHDRAW

Testing



Unit Test Example

Deposit £100

Withdraw £70

Assert Balance == £30

Unit Test Examples

Deposit £100

Withdraw £70

Assert Balance == £30

Deposit £100

Withdraw £101

Assert Exception Throw

Any problems
with the testing approach?

How big is the testing space?

All possible behaviours of the bank system (good and bad)

Combinations of Actions

- Deposit ; Withdraw ; Deposit
- Withdraw ; Withdraw ; Deposit

Combinations of Actions

- Deposit ; Withdraw ; Deposit
- Withdraw ; Withdraw ; Deposit
- Deposit in Malta ; Withdraw in England
- Deposit in € ; Withdraw in £
- Deposit in € ; Exchange Rate falls ; Withdraw in £

Combinations of Actions

- Deposit ; Withdraw ; Deposit
- Withdraw ; Withdraw ; Deposit
- Deposit in Malta ; Withdraw in England
- Deposit in € ; Withdraw in £
- Deposit in € ; Exchange Rate falls ; Withdraw in £
- Varying amounts, day of week, month, year

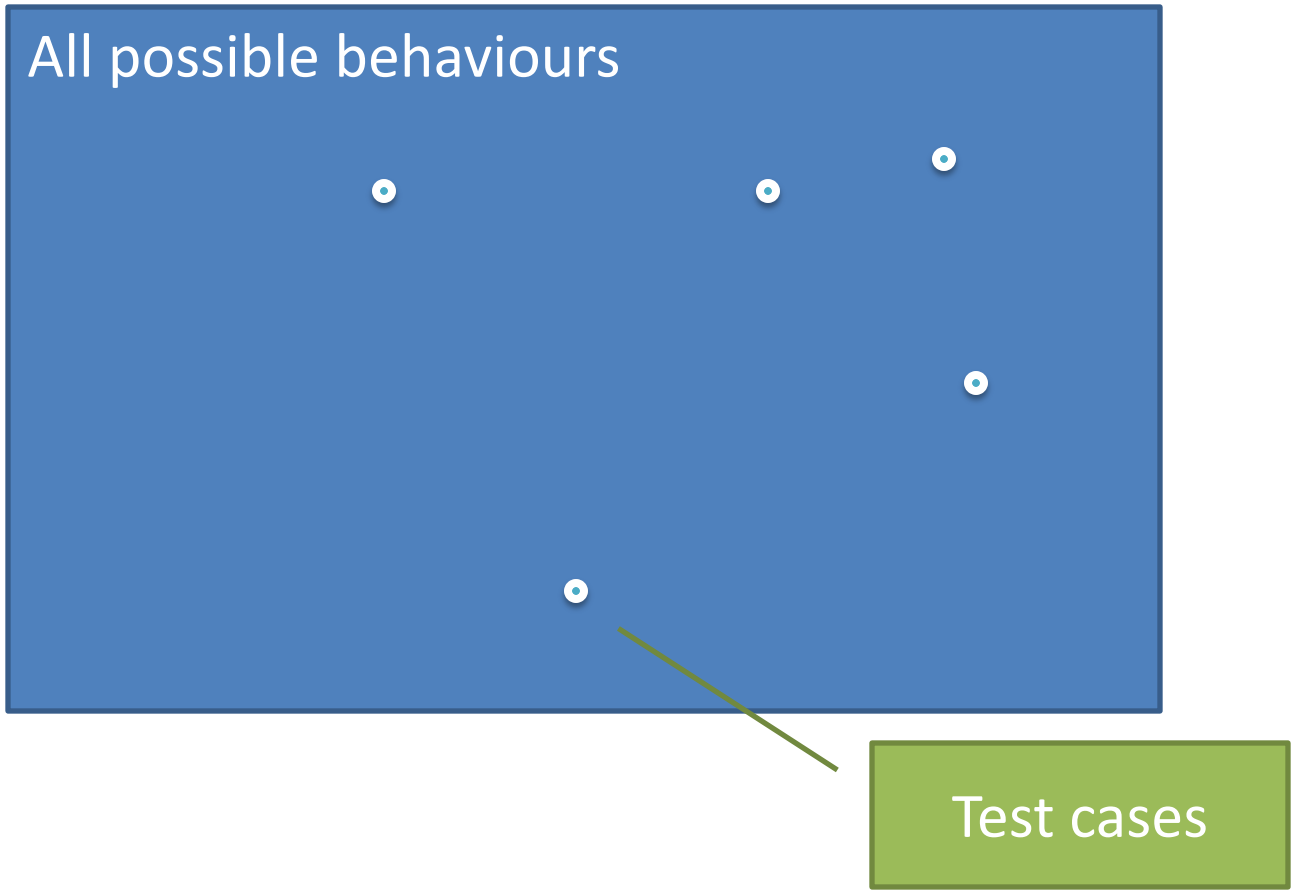
Combinations of Actions

- Deposit ; Withdraw ; Deposit
- Withdraw ; Withdraw ; Deposit
- Deposit in Malta ; Withdraw in England
- Deposit in € ; Withdraw in £
- Deposit in € ; Exchange Rate falls ; Withdraw in £
- Varying amounts, day of week, month, year
- Concurrency issues

Combinations of Actions

- Deposit ; Withdraw ; Deposit
- Withdraw ; Withdraw ; Deposit
- Deposit in Malta ; Withdraw in England
- Deposit in € ; Withdraw in £
- Deposit in € ; Exchange Rate falls ; Withdraw in £
- Varying amounts, day of week, month, year
- Concurrency issues
- Security issues

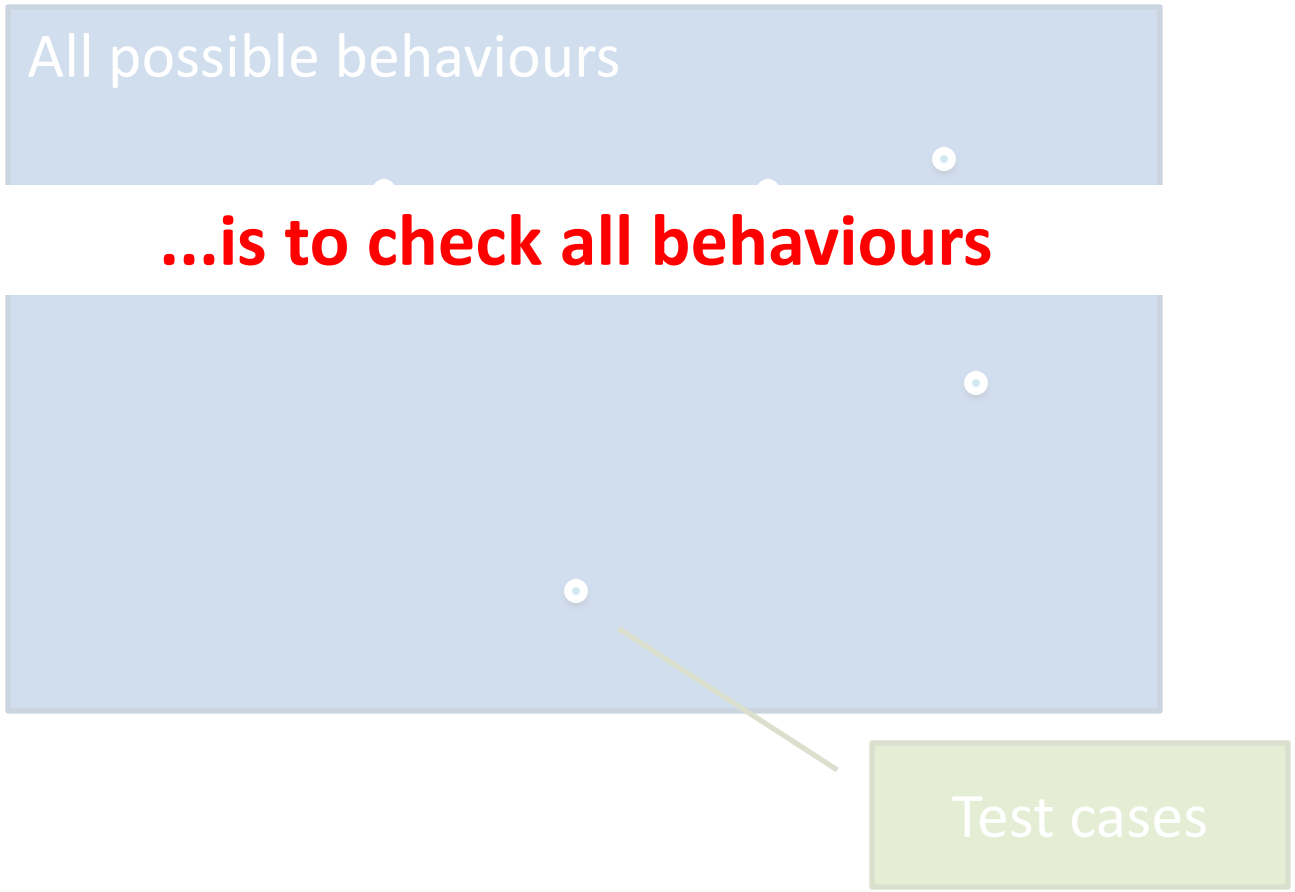
Putting everything into perspective



Limitation of Testing

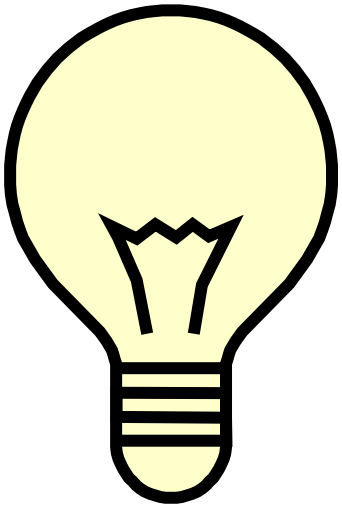
The focus is on a few cases that we can foresee/have time to test!

What We Really Want...

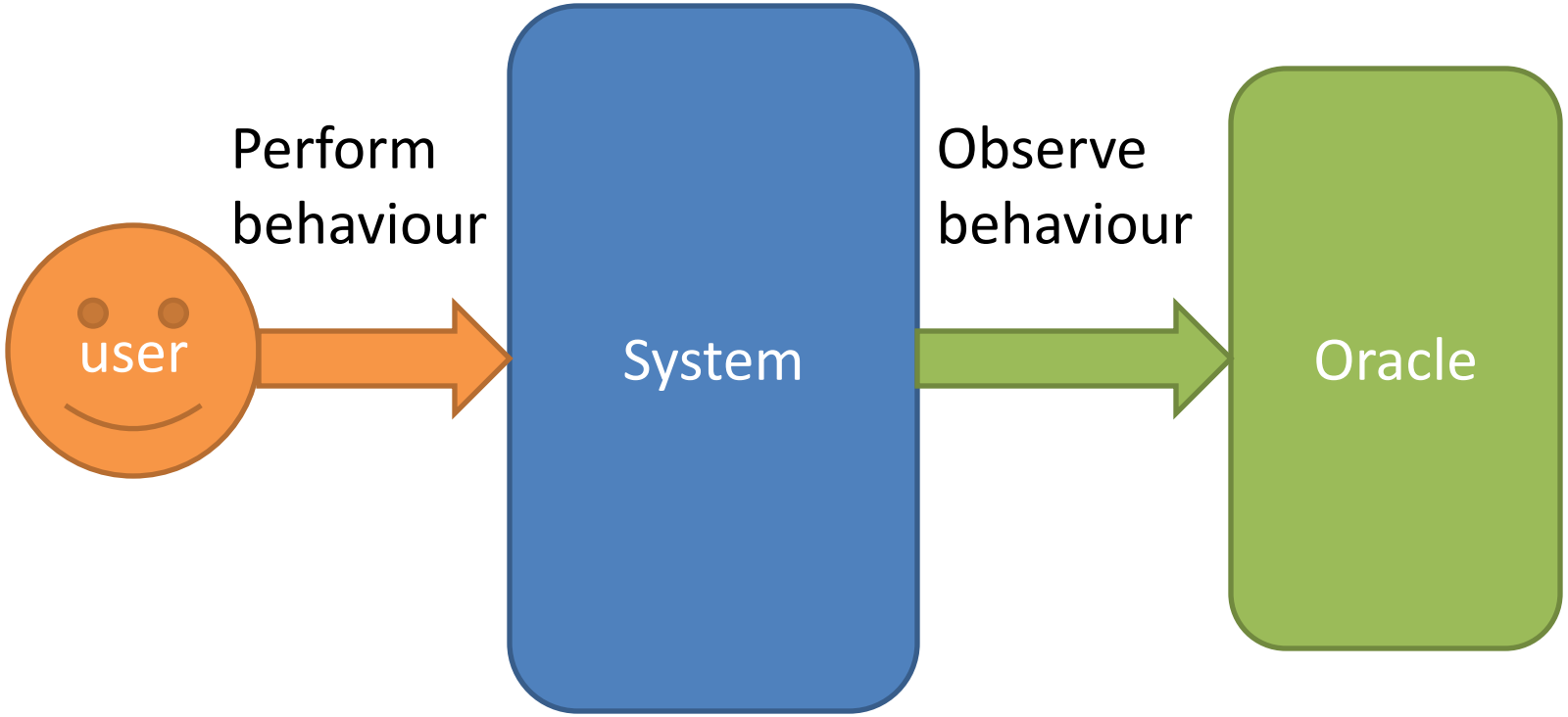


What We Really Want...

What if we check all behaviours by using assertions at runtime?



Runtime Testing



Can we use the testing assertions?

Assertion Examples

Deposit £100

Withdraw £70

Assert Balance == £30

Deposit £100

Withdraw £101

Assert Exception Throw

Assertion Examples

Too specific !

System is now less predictable

Runtime Assertions

Deposit £d

Withdraw £w

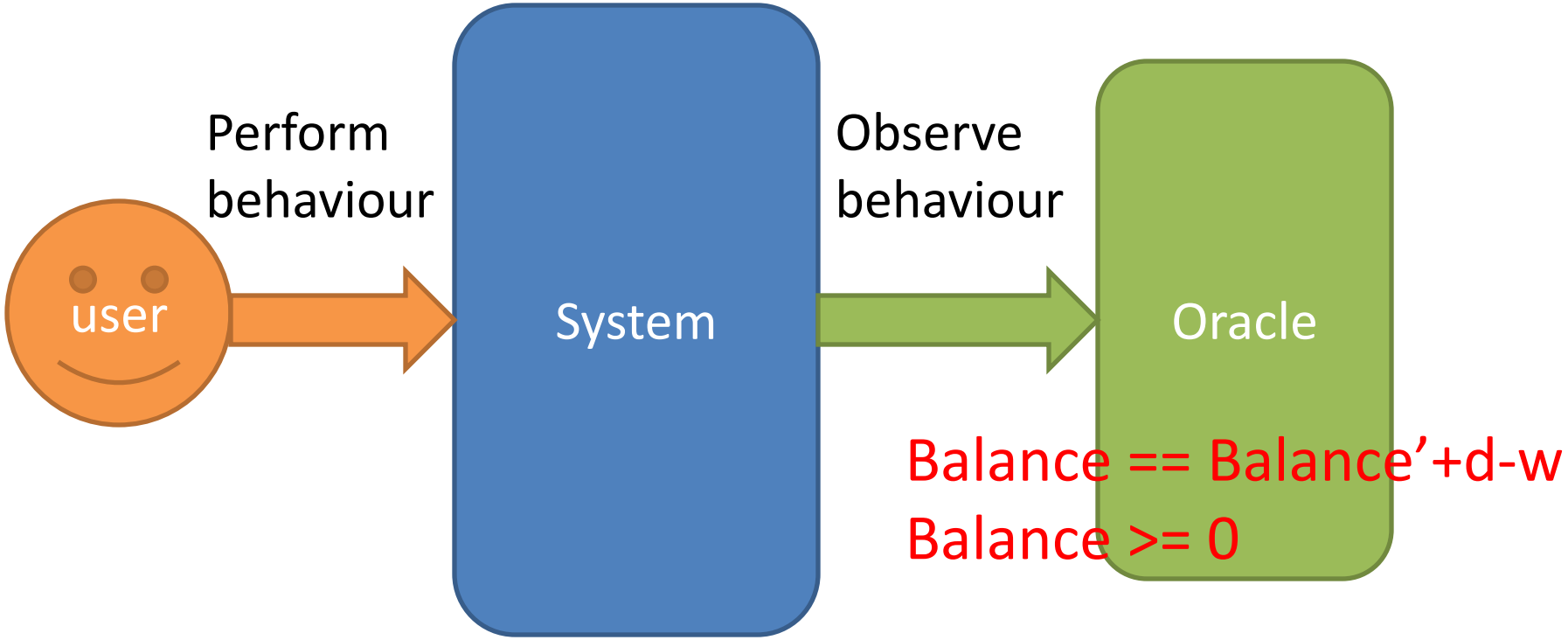
Assert Balance == Balance’ + d - w

Runtime Assertions

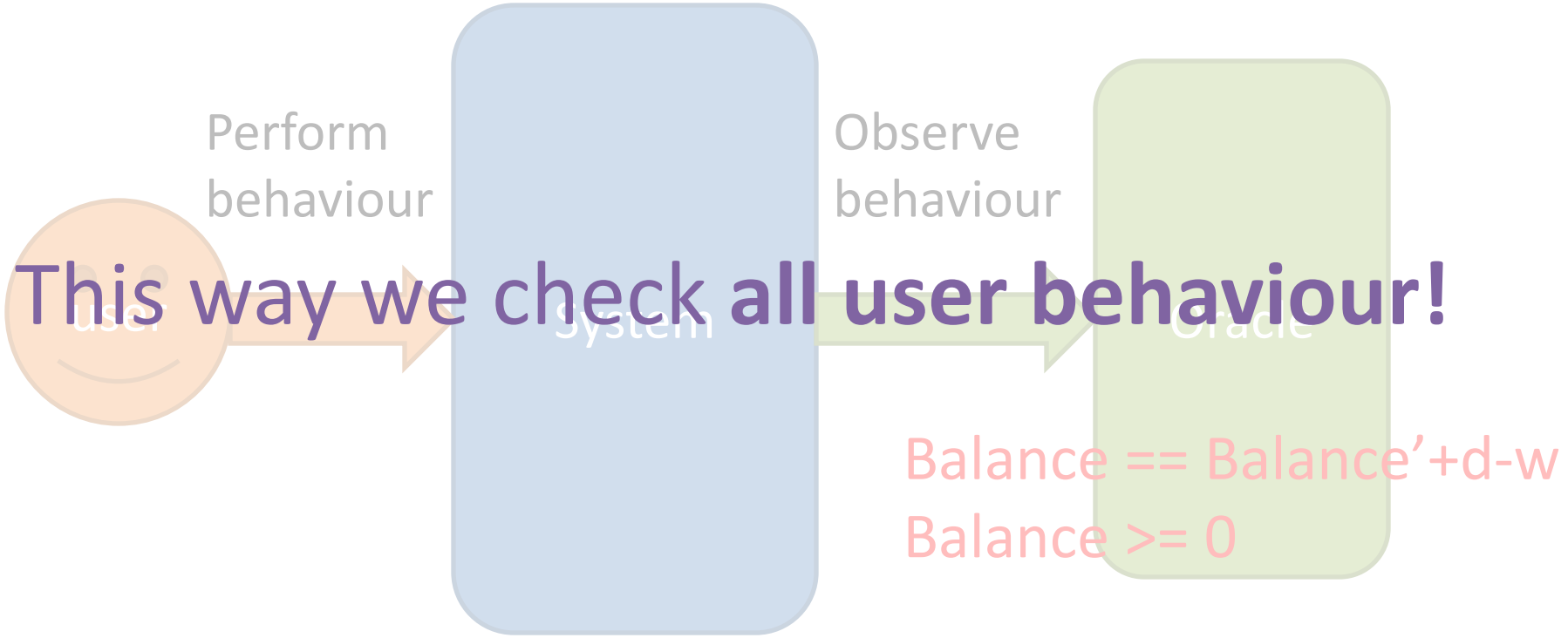
Deposit £d
Withdraw £w
Assert Balance == Balance’ + d - w

Deposit £ ?
Withdraw £ ?
Assert Balance > 0

Runtime Testing



Runtime Testing



This way we check **all user behaviour!**

Runtime Testing

Can we do more with assertions?

Complex Runtime Assertions

- Check sequences of method calls:

No withdrawal can take place before an initial deposit

Complex Runtime Assertions

- Check sequences of method calls:

No withdrawal can take place before an initial deposit

How do you check this using assertions?

No withdraw before deposit

```
boolean firstDeposit = false;

public void deposit() {
        firstDeposit = true; … }
```

No withdraw before deposit

```
boolean firstDeposit = false;

public void deposit() {
        firstDeposit = true; … }

public void withdraw() {
        assert(firstDeposit); … }
```

No withdraw before deposit

```
boolean firstDeposit = false;

public void deposit() {
        firstDeposit = true; … }

public void withdraw() {
        assert(firstDeposit ); … }
```

Cluttered code!

No withdraw before deposit

```
boolean firstDeposit = false;

public void deposit() {
        firstDeposit = true; … }

public void withdraw() {
        assert(firstDeposit ); … }
```

More clutter for more complex checks!

Advanced Assertions

- No withdrawal unless previously logged in and not logged out since

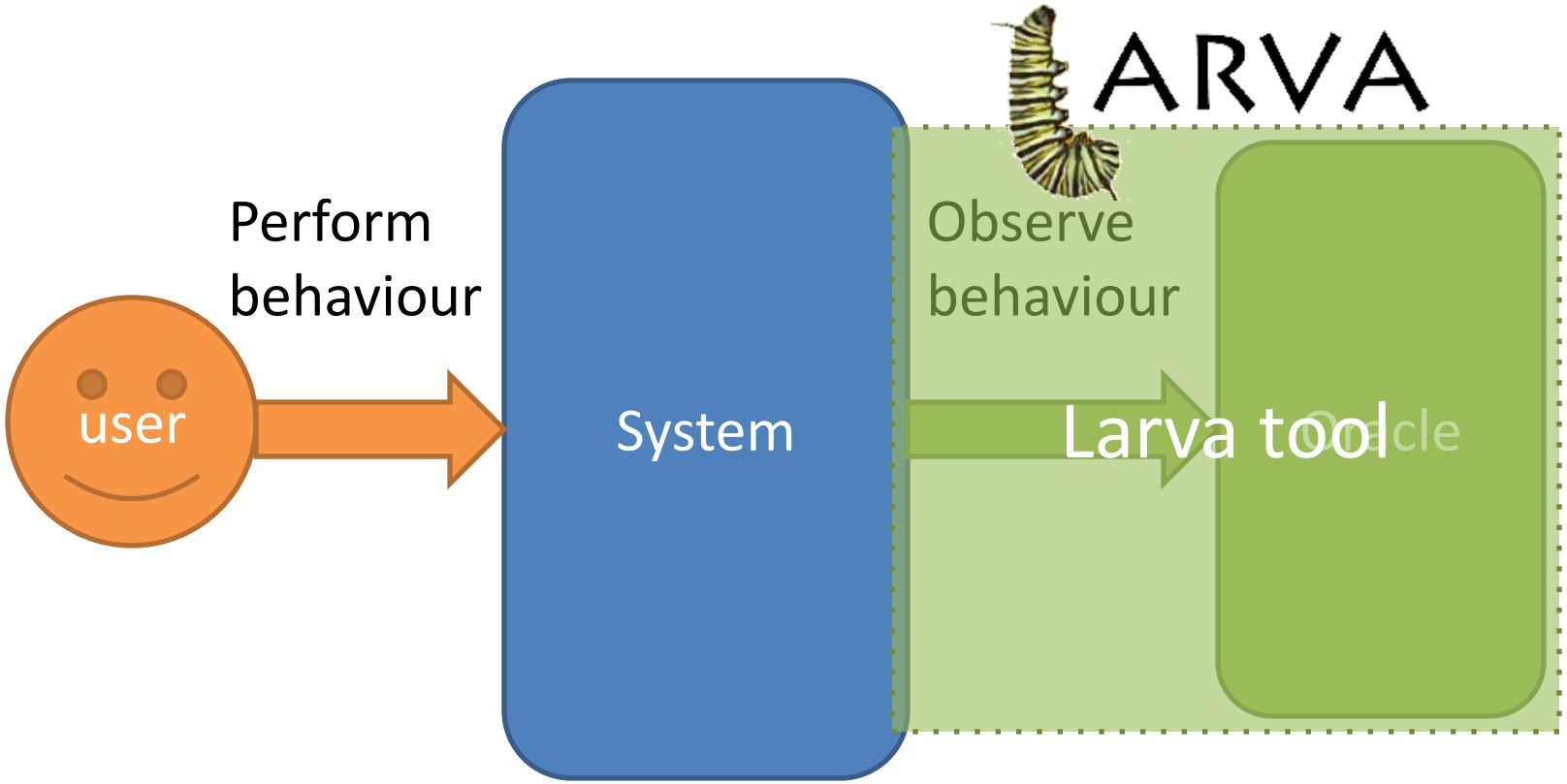
Advanced Assertions

- No withdrawal unless previously logged in and not logged out since
- No withdrawal before 1 day of account activation

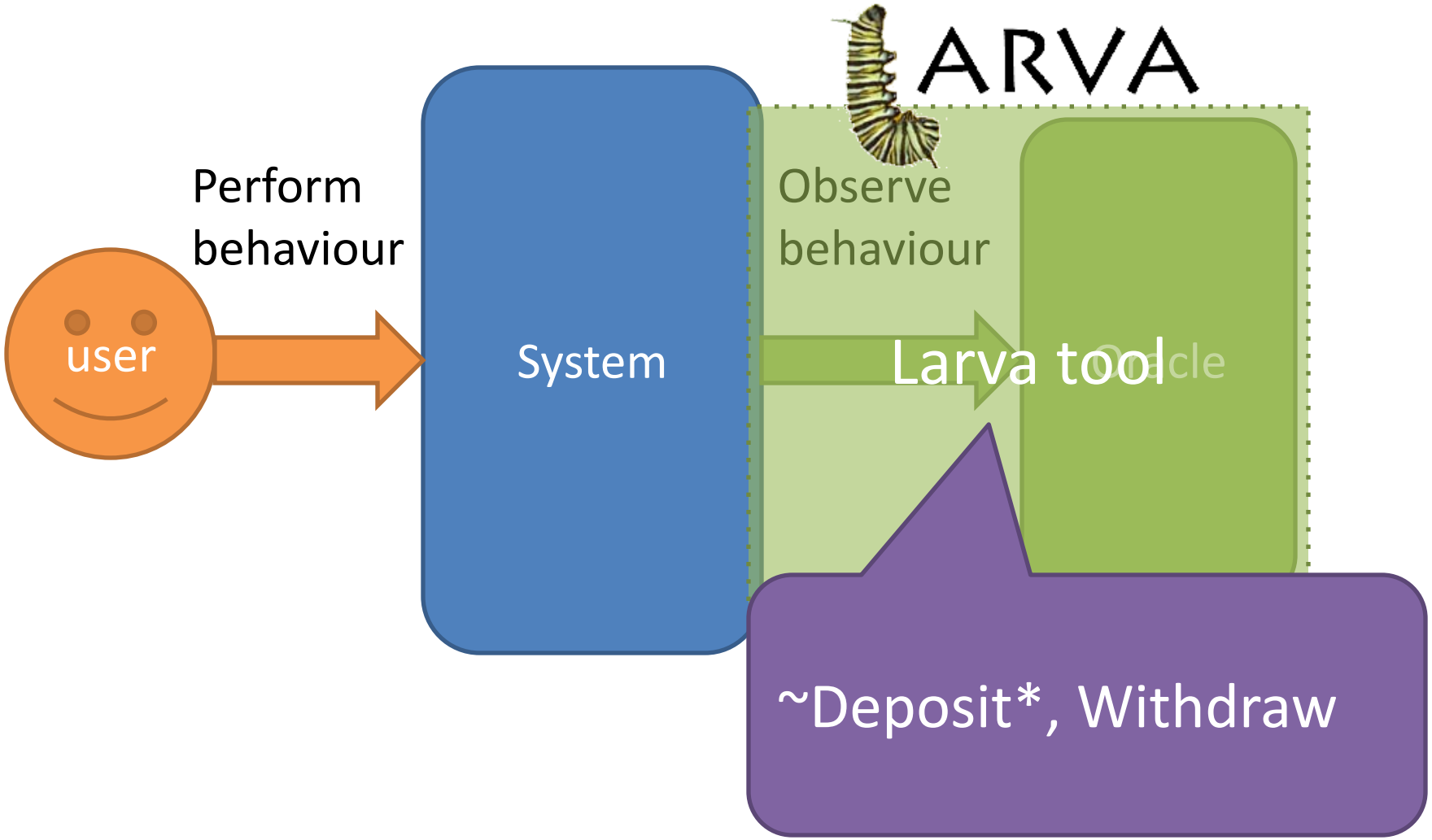
Advanced Assertions

- No withdrawal unless previously logged in and not logged out since
- No withdrawal before 1 day of account activation
- Account should be suspended upon 1 year of account inactivity

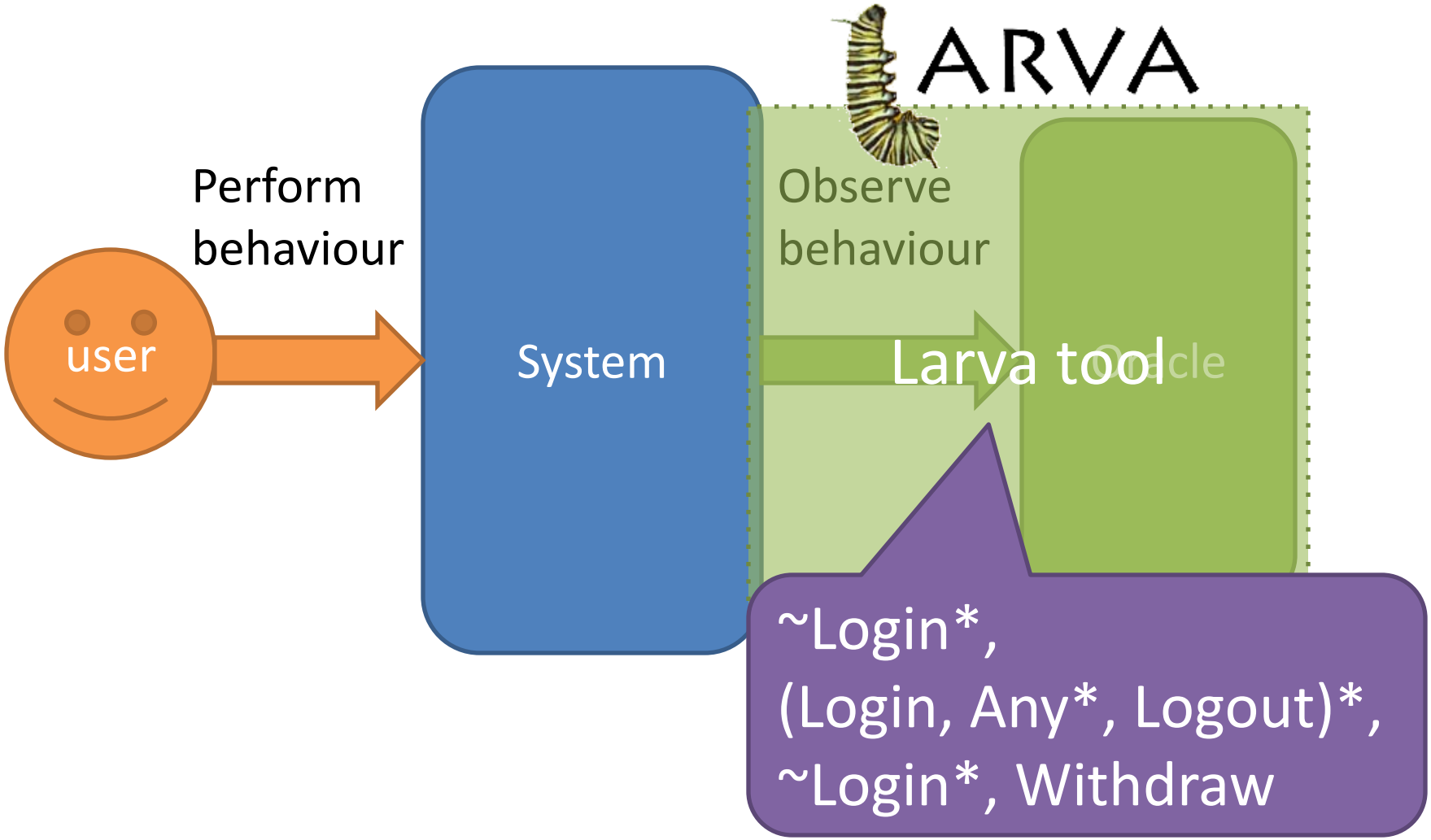
Tool Support



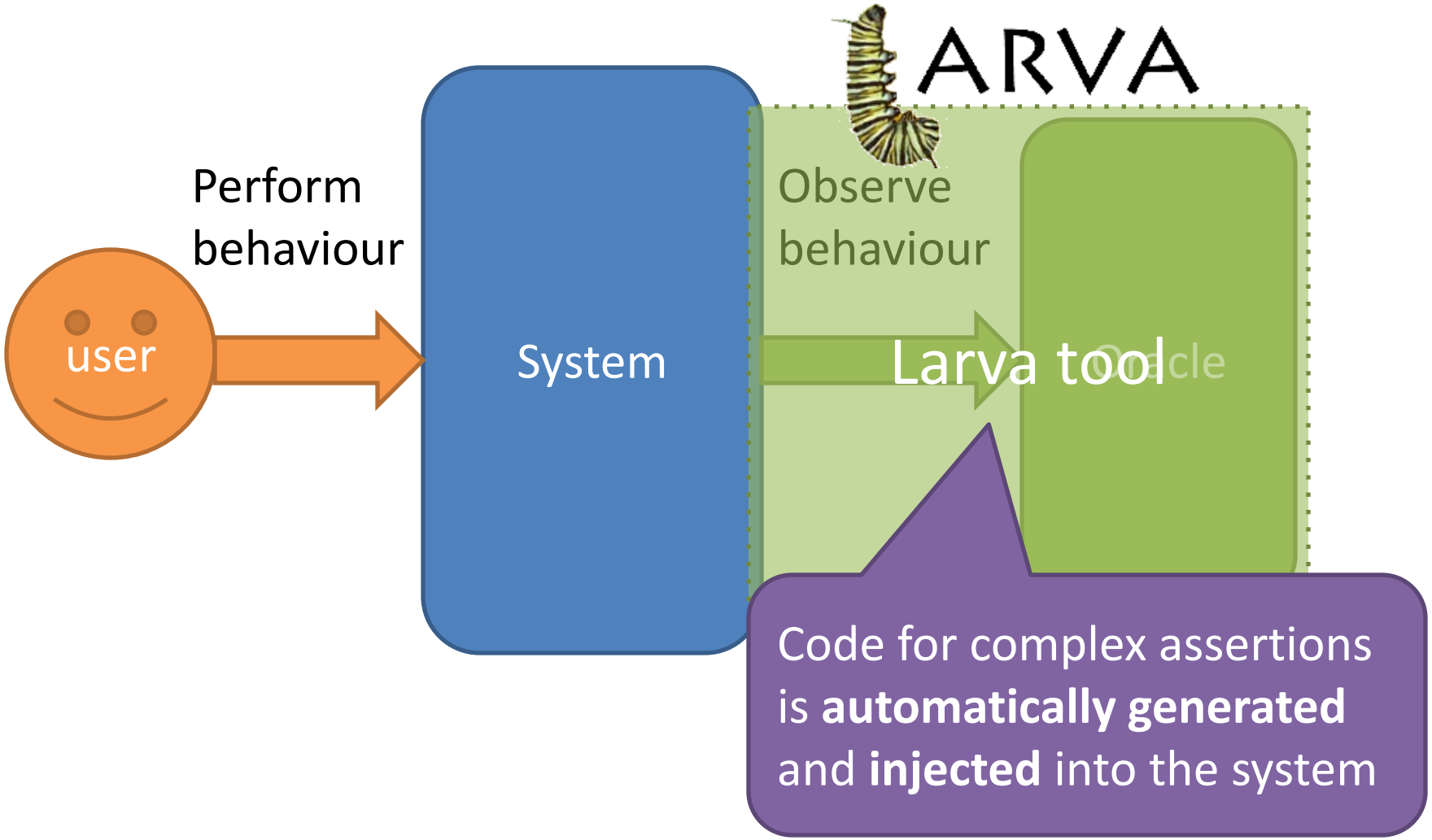
Tool Support



Tool Support



Tool Support

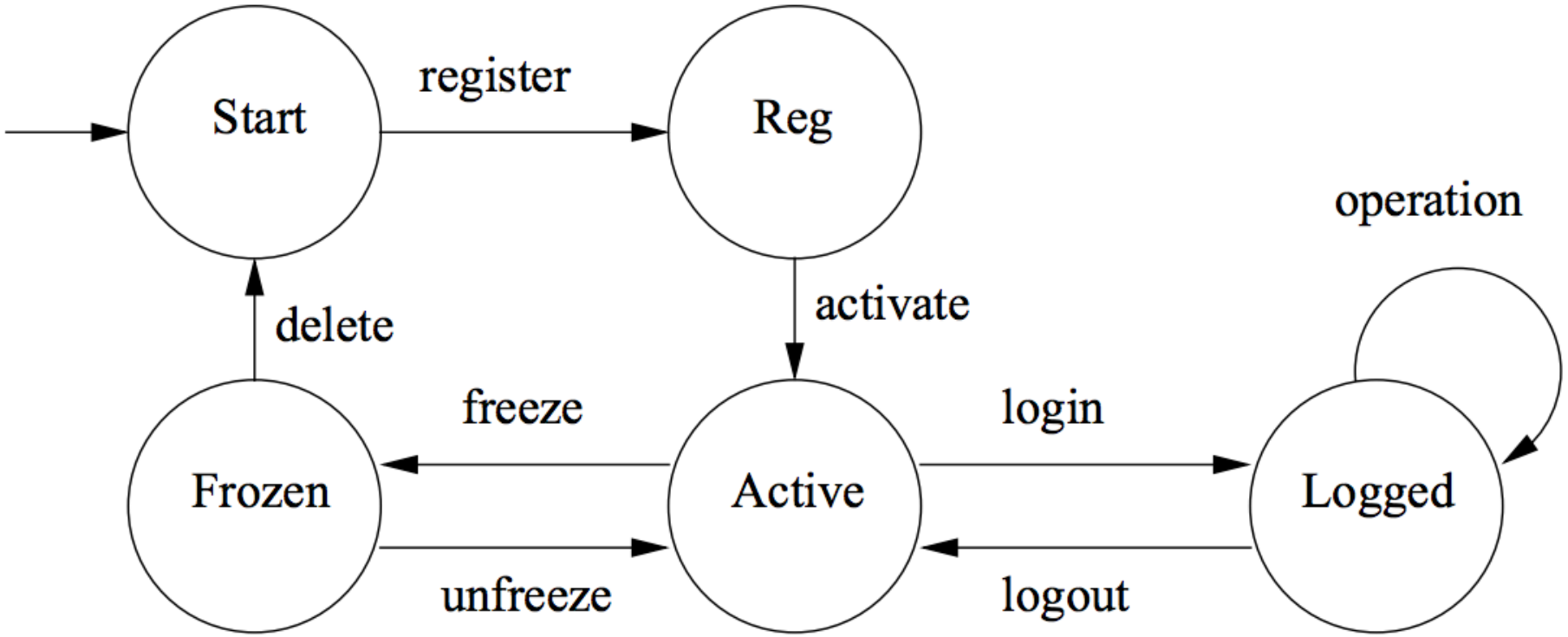


Success Stories

IXARIS

Assertions Checked

- Every user goes through the expected life



Assertions Checked

- Every user goes through the expected life
- No user rights are forfeited

RIGHT	login	deposit	withdraw	spend	...
user1	✓	✓	✓		
user2					
user3	✓	✓			
user4	✓				
user5	✓	✓	✓	✓	
...					

Assertions Checked

- Every user goes through the expected life
- No user rights are forfeited
- No limits are ever exceeded

Spending limit per day
for Gold user

Spending limit per day
for normal user

Conclusions

- Some things are really hard to test

Conclusions

- Some things are really hard to test
- Might be a good idea to keep assertions at runtime

Conclusions

- Some things are really hard to test
- Might be a good idea to keep assertions at runtime
- Some assertions are hard to express (correctly)

Conclusions

- Some things are really hard to test
- Might be a good idea to keep assertions at runtime
- Some assertions are hard to express (correctly)
- Tool support exists and we are on the lookout for more case studies

- Department Homepage
- Ongoing Projects
- Open Proposals
- Past Projects
- Publications
- Talks
- Tools
- Undergraduate
- Collaboration Models

News on Campus

Campus Map

Process Engineering Security & Testing Research Lab

The Process Engineering and Software Testing Research Lab was formed in 2012 and complements the Department of Computer Science's focus on the area of Dependable Systems. As the name implies, the group approaches the problem of building dependable systems from two directions. The first thrust is the application of computer science concepts and techniques in software testing and static analysis. However, experience shows that a key component of the success of such techniques depends on how they are applied and integrated within the development process. Hence the group has a another focus on development process engineering.

Although the group is still in its infancy, it already offers relevant academic courses and dissertations at both undergraduate and graduate levels. A number of research funding proposals have also been submitted.

Notices

MUTATinc Tool Launched
Visit the MUTATinc tool page for more information

Talks updated!
Have a look at the Talks page

Ongoing Projects

Open Proposals

Past Projects

Collaboration Models

Publications

Talks

Tools

Undergraduate