Using Model Learning for security analysis and simulation

Sébastien Salva

LIMOS – IUT CA - UCA

Who am I?

```
Public void setUp(){
Identity id=new Identity(‘’salva’’);}

Public void testid (){
assertEquals(id.surname, ’’sébastien’’);
assertEquals(id.name, ’’salva’’);
assertEquals(id.labo, ’’LIMOS’’);
assertEquals(id.univ ’’IUT, University Clermont Auvergne, France’’);

assertArrayEquals(i.recherche, new String[] {‘’Soft. Eng.’’, ‘’Testing’’, ‘’Security’’, ’’Data quality’’, ‘’services‘’});
}
```

Who am I?

(Full)Professor

Co-manager of the DSI (Data Service Intelligence) team, LIMOS

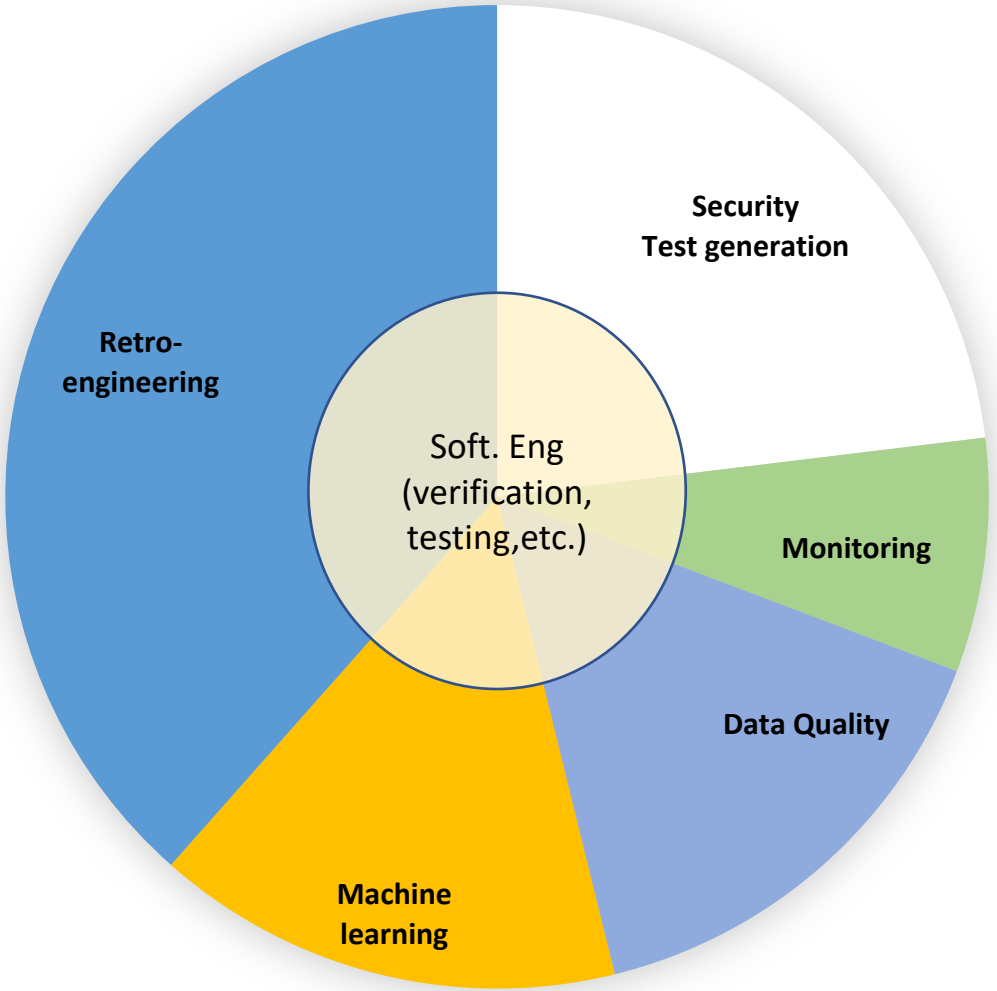
DSI

big data, machine learning, decision making, AI,

onthology

Web service, Cloud computing, model gen., verification

Who am I?



Plusieurs collaborations avec industriels sur syst. Industriels, services, IoT, etc.

New Topics ?

Qualité de données et génie logiciel

Qualité de données et détection anomalies, securité, sur de l'embarqué ((edge computing) ?

Analyse de logs, (retro-engineering) -> green computing

Lien Apprentissage de l'orthographe/grammaire et qualité de développeur info

Avoir une bonne orthographe permet-elle d'être un bon développeur ? Vice versa (apparemment non)

Paper presentation

- **Using Model Learning for security analysis and simulation**
  (in the context of IoT systems)

  1. Model learning from logs of IOT systems
  2. Security analysis
  3. Simulation with mocks

Context

- Projet VASOC (Security Audit of IOT systems)
  - UCA, University Jean Monnet, Saint Etienne, 6 Industrial partners

- Make IoT audit easier:

  From logs:
  - Retro-engineering (Model learning) of devices
  - Retro-engineering of IoT systems
  - Vérification of security properties
  - Generation of mock services to simulate devices

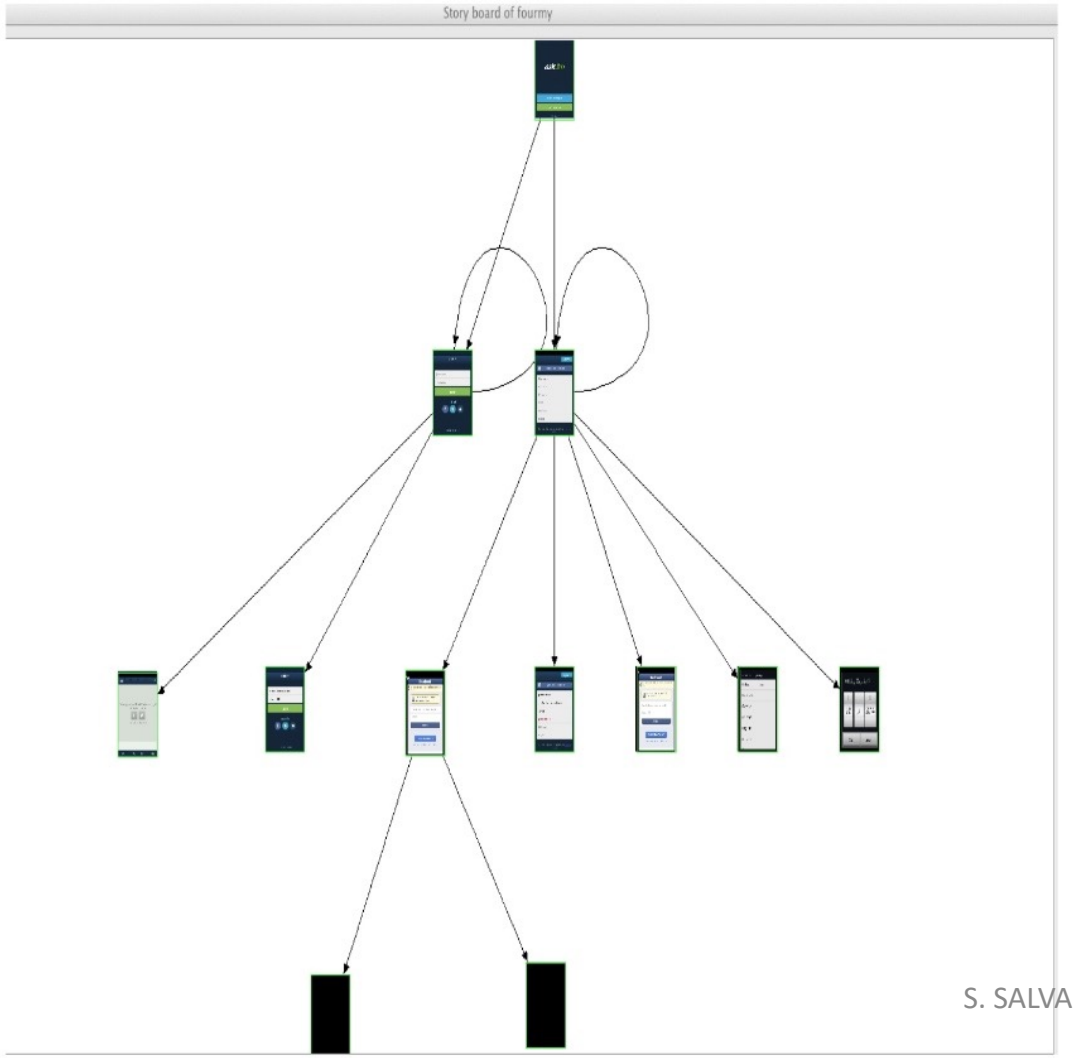
- 3 main tools

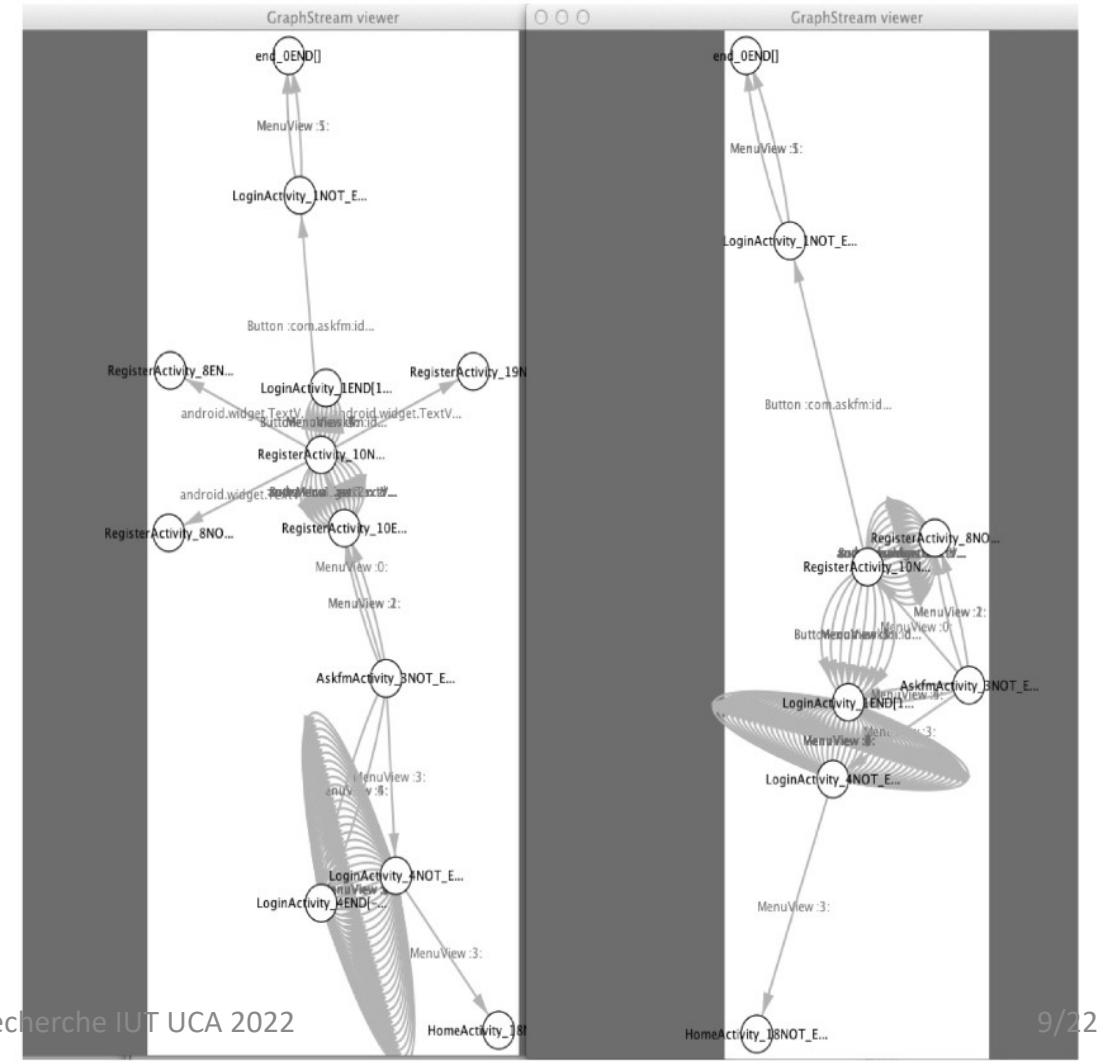
1. Model learning from logs

- Generation of models (behavioural IOLTS , UML activity diag. Dependency) from logs

- To retrieve documentation
- To analyse the system (errors, security, recommendations, etc.)

1. Model learning from logs



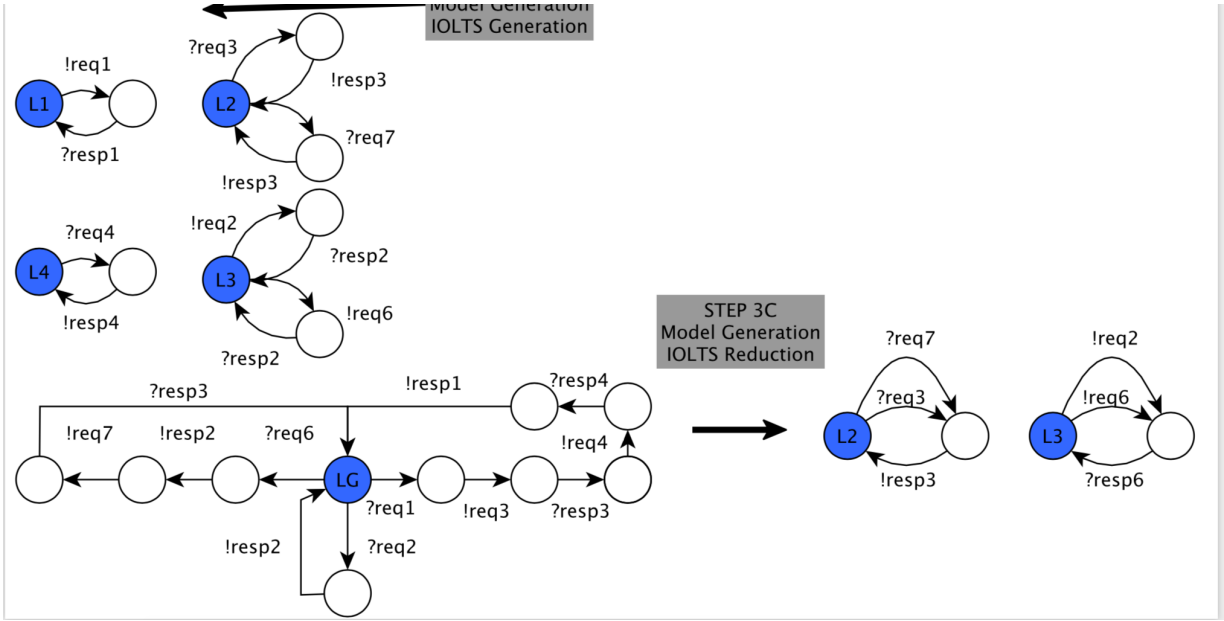


1. Model learning from logs

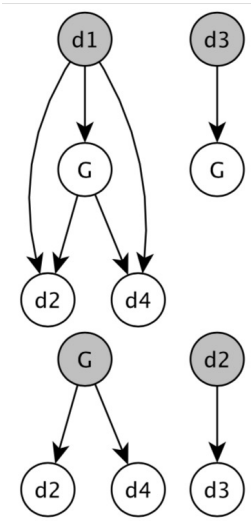
Tool example : CkTail

```
09:39:53.416 req1(from:=d1,to:=G,param:=udevice,svalue:=open)
09:39:53.848 req2(from:=d3,to:=G,param:=udevice,svalue:=66)
09:39:55.416 req3(from:=G,to:=d2,param:=heating,cmd:=On)
09:39:55.429 resp3(from:=d2,to:=G,switchcmd:=done)
09:39:55.430 req4(from:=G,to:=d4,param:=heating,cmd:=On)
09:39:55.433 resp4(from:=d4,to:=G,switchcmd:=done)
09:39:55.567 resp1(from:=G,to:=d1,content:=req sent)
09:39:55.629 resp2(from:=G,to:=d3,content:=ok)
09:44:19.714 req6(from:=d3,to:=G,param:=udevice,svalue:=68)
09:44:19.727 resp2(from:=G,to:=d3,content:=ok)
09:44:19.727 req7(from:=G,to:=d2,param:=heating,cmd:=Off,svalue:=68)
09:44:19.866 resp3(from:=d2,to:=G,switchcmd:=done)
```

Logs



Behavioral and general models
1 model / component



Dependency graphs

2. Security Analysis

- Checking whether security recommendations are satisfied in models
  - Recommendations for IoT from ENISA

- Recommendations modelled with property types

- G((*loginAttempt*(*c*)∧ *credential*(*x*)) → *encrypted*(*x*)) derived from the ENISA measure GP-TM-24.

3. Simulation with mocks

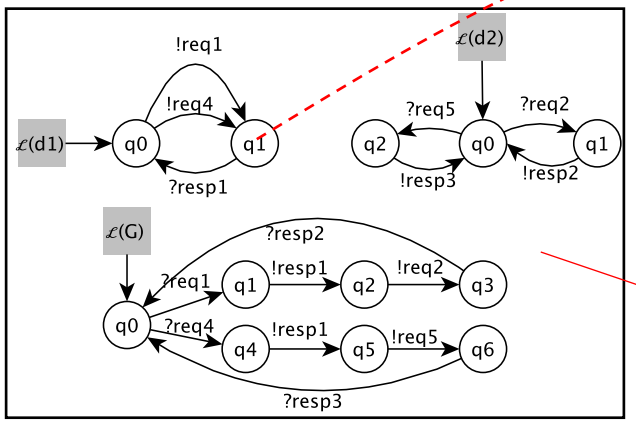
- Use of models to generate Mock components
- Completion with new functionalities for testing
  - Security testing
  - Robustness
  - Interoperability

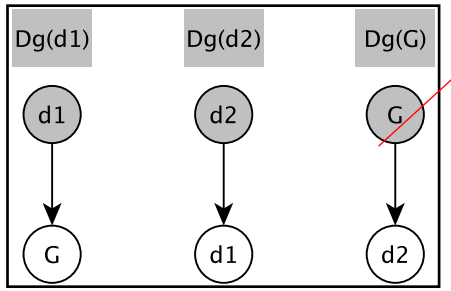
3. Simulation with mocks

Event Log → Models Learning → Choice of the devices to simulate → Mock model generation → Testing with mocks

Event LOG

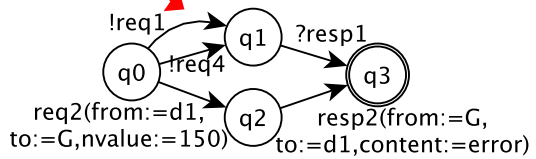
09:39:53.416
req1(from:=d1,to:=G,param:=udevice,nvalue:=64,svalue=TEMP)
09:39:53.848
resp1(from:=G,to:=d1,content:=ok)
09:39:54.216
req2(from:=G,to:=d2,type:=command,nvalue:=64,svalue=TEMP)
09:39:55.429
resp2(from:=d2,to:=G,content:=received 64)
…





Quality metric evaluation (testability, dependability, etc.)

Choose



Mock Runner

3. Simulation with mocks

Choice of the components to mock

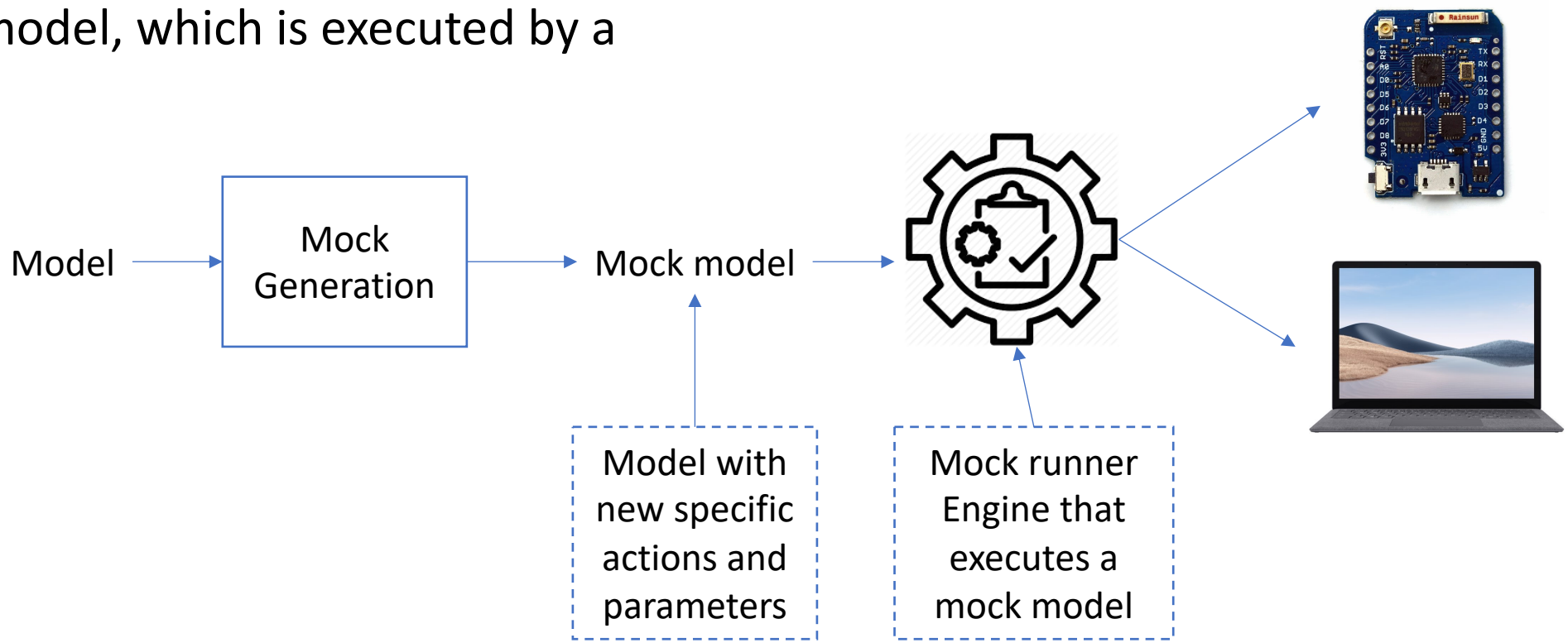
- Could be manually performed but … seems difficult

Evaluation of models with 6 quality metrics

- Understandability
- Accessibility
- Testability (obs, cont)
- Dependability (in-deps, out-deps)

3. Simulation with mocks

Mock = Mock model, which is executed by a Mock Runner

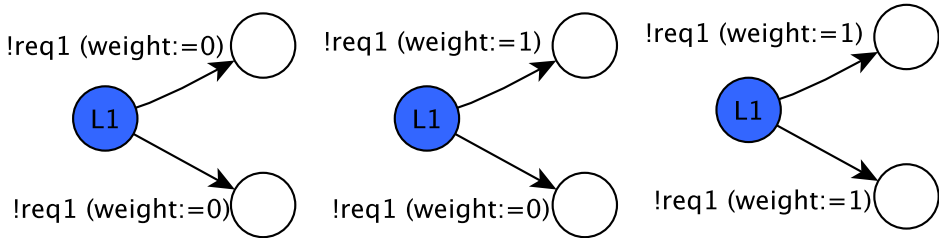


3. Simulation with mocks

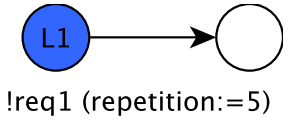
Mock model

IOLTS with some specific parameters:

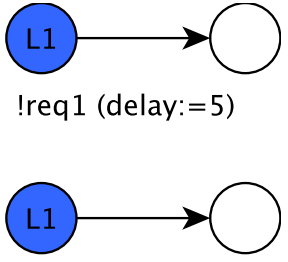
weight:



repetition:

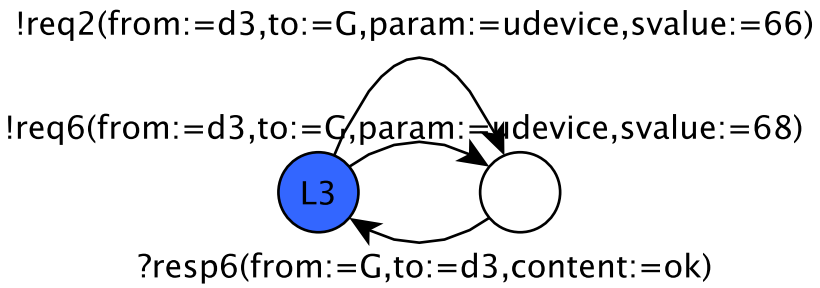


delay:

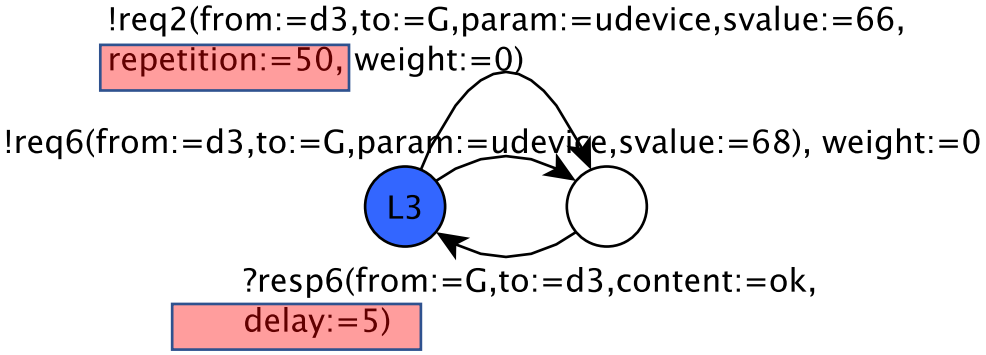


3. Simulation with mocks

Mock model examples:



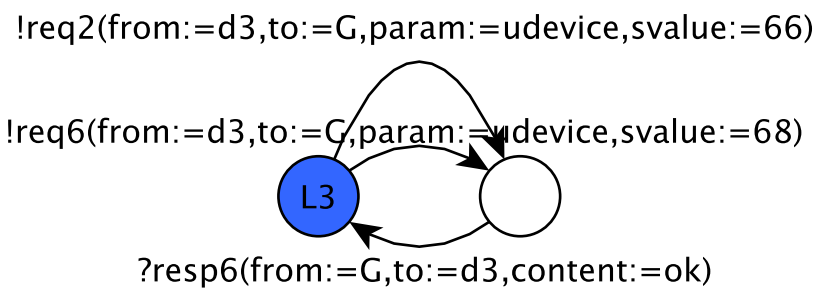
Original IOLTS



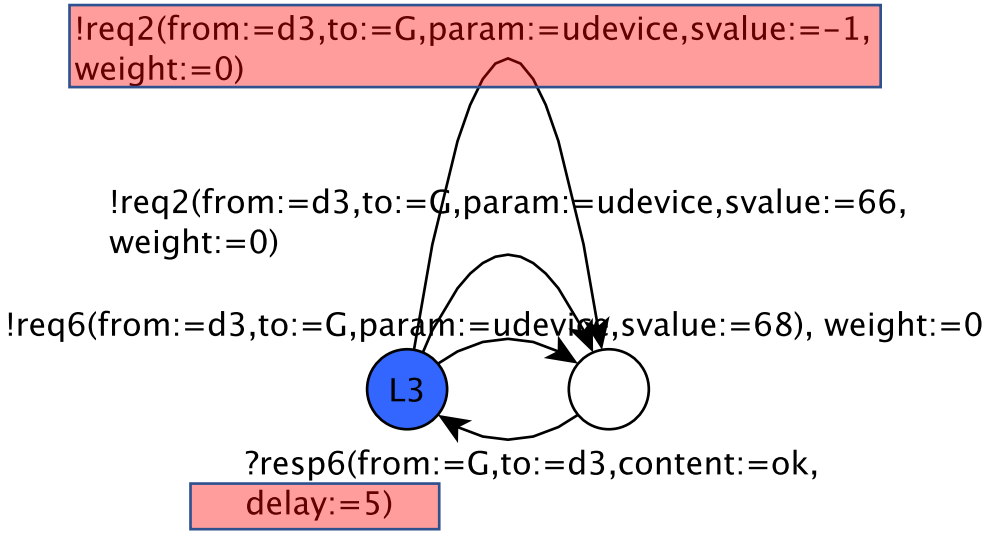
Mock model
(repetition of req2, delay for ?resp6)

3. Simulation with mocks

Mock model examples:



Original IOLTS



Mock model
(new action, delay for ?resp6)

3. Simulation with mocks

(preliminary) Evaluation :

Perfomed from a real IoT system (20 devices, 2 gateways)

Conducted with 24 students (Lpro)

- Measured the times required to code mocks from logs from scratch and times for generating mocks
  - Approach provides greater efficiency, cuts the time by 75%

- mocks can replace reals devices if the inferred IOLTS are precise enough precise (no over- or under-approximated)

Limitations

- Mock runner can be called from test cases but in a limited way at the moment
  - (start mock runner, give mock model, get runs and errors)
  - -> could be extended (get number of inputs received?, outputs sent ?)

- Current Mock runner impl. requires some resources (mem, and cpu),
  - ok if deployed on Web servers

- Choice of mockable devices made from the interpretation of metrics.
  - depends on the systems ? Or the dev. Tools etc.

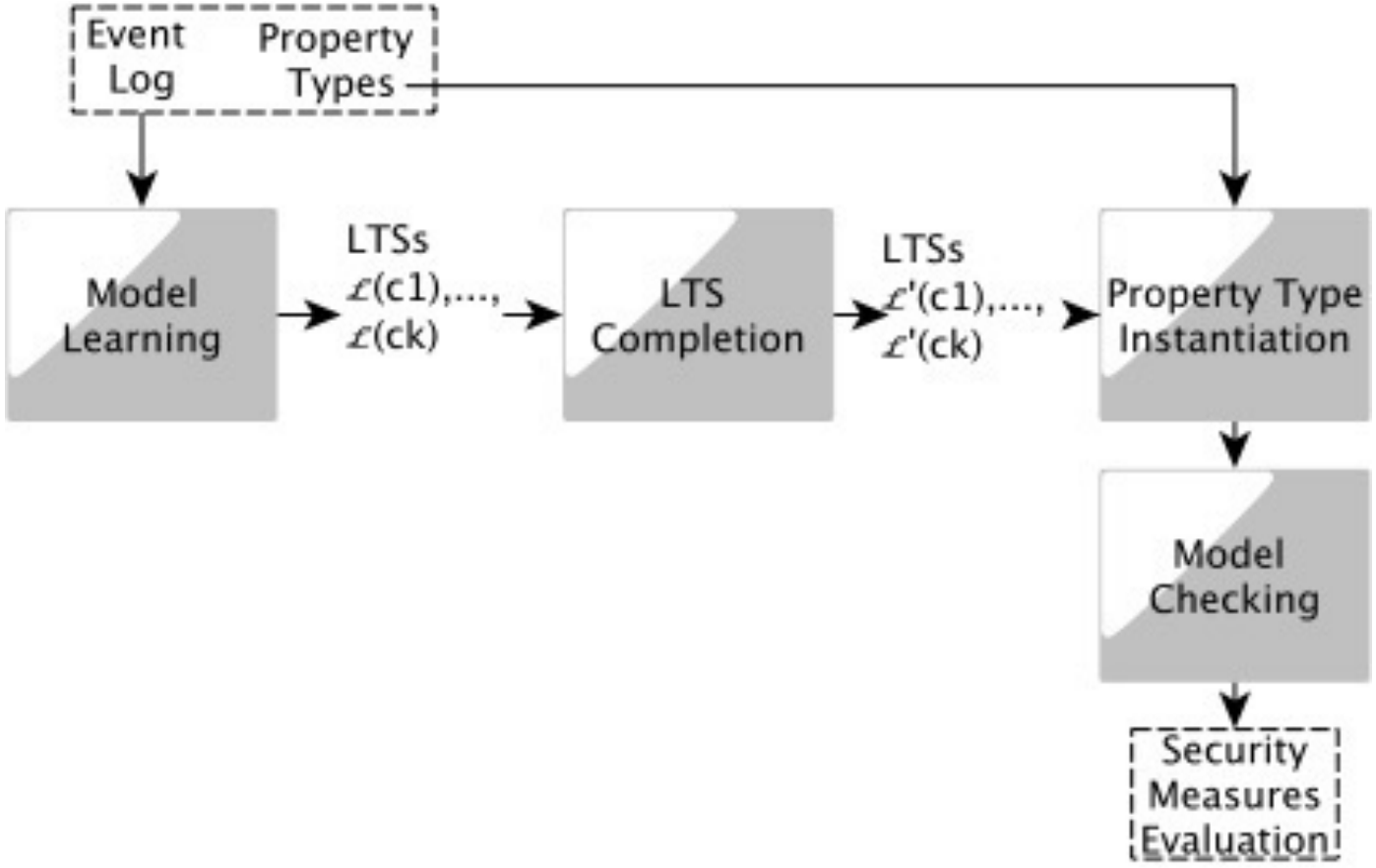
Perspectives

- Hardware (sensor, memory), network not taken into account
  - How to link software, hardware with models,
  - How to get data related to hardware
- Automatic generation of mocks with different strategies
  - Strategy for robustness testing (!nject unexpected events, etc.)
  - Strategy for security testing (inject malicisous behaviours, etc.)
- Model learning for green computing?
  - To help in sustainable development
  - Model analysis to help reduce energy costs, etc.

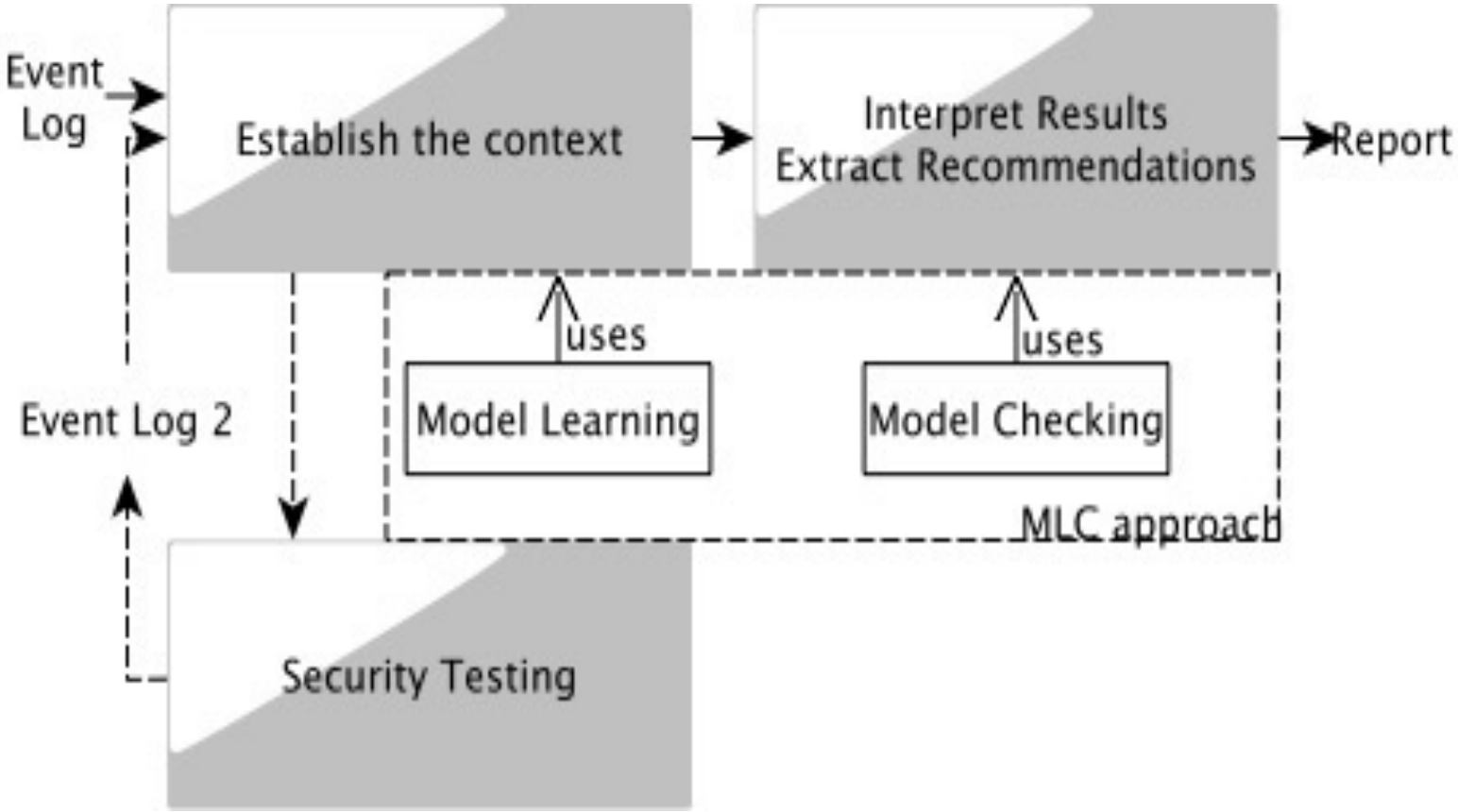
Thanks

- Questions ?

Model learning and model checking overview



Integration to security audit



Mock generation/execution

Mock Runner ~ Engine that executes a mock model

- Implemented as a Rest service

- Performs concrete executions by following the paths of a mock model
  - Starts from q0 and either waits for an input or executes an ouput
  - Builds and stores runs (alternate sequence of states and actions)
  - If Mock runner receives an unexpected action, it returns an error in its log (can be used by testers)
  - Stops a current execution in an terminal or deadlock state