

Inspired
logo

Modelling mobility aspects of security policies

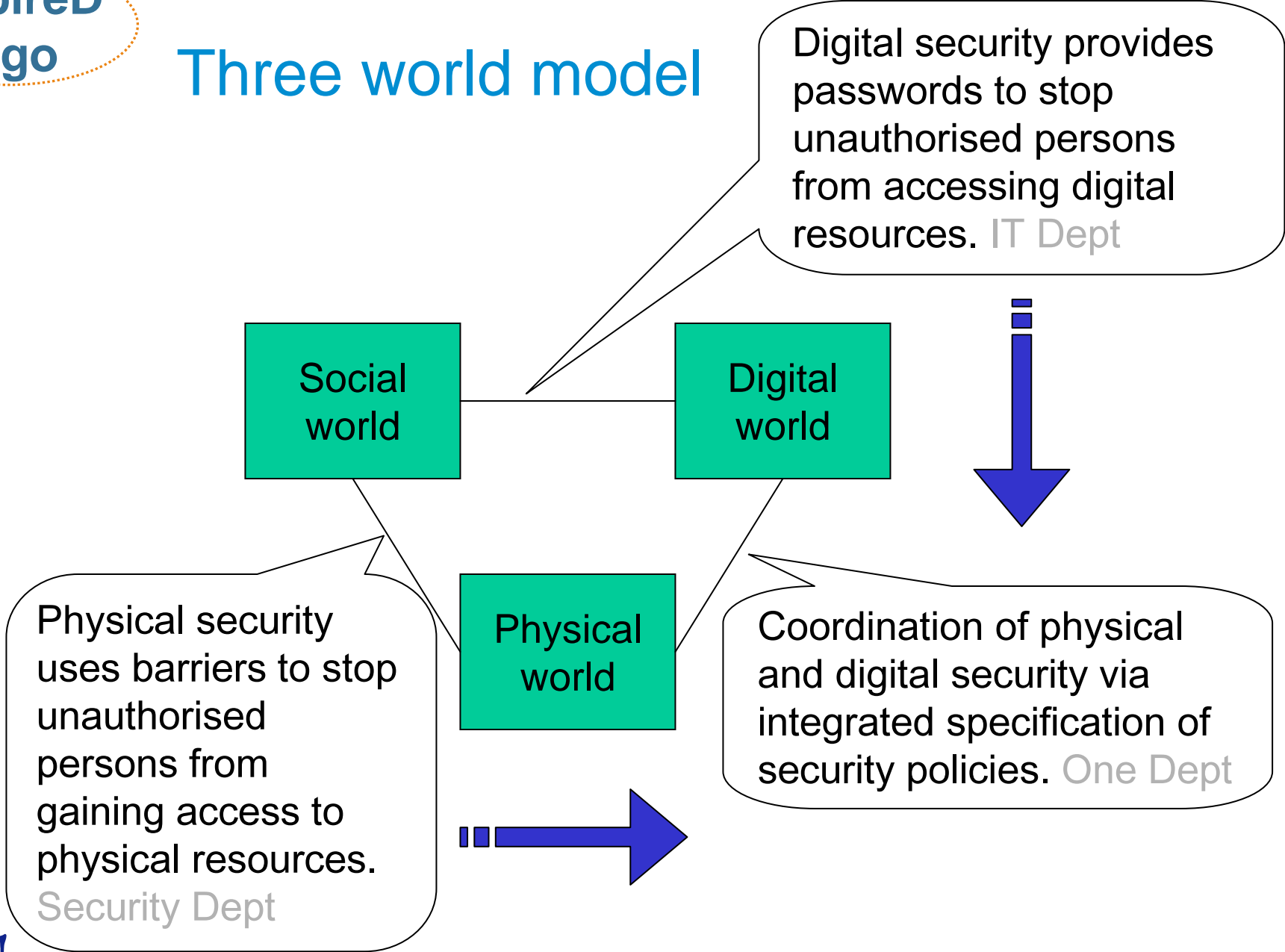
Pieter Hartel, Pascal van Eck
Sandro Etalle, Roel Wieringa
University of Twente

Objectives

- ❖ Explain the problem that mobility may invalidate (or confuse) good security policies
- ❖ Show that the solution requires the integration of physical and logical security policies
- ❖ Illustrate this using case studies
- ❖ Model the essence of the solutions using SPIN



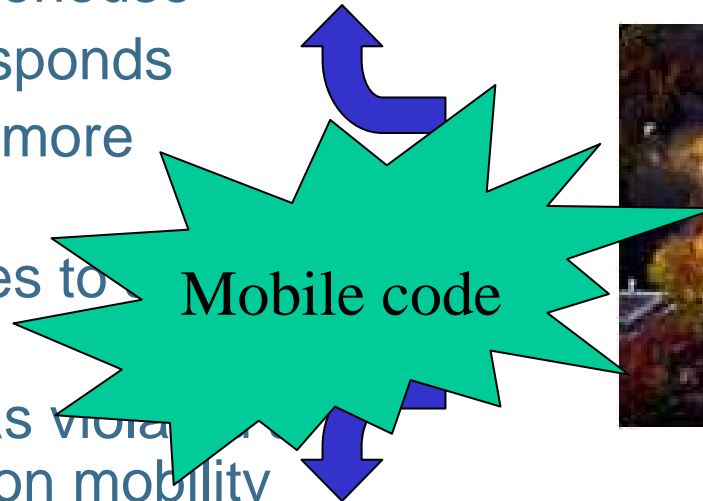
Three world model



Ping

```
$ ping www.whitehouse.gov  
Reply from 194.109.217.178: time=110ms  
Reply from 194.109.217.178: time=1072ms  
Reply from 194.109.217.178: time=90ms
```

- ❖ Bob pings Whitehouse
- ❖ Whitehouse responds
- ❖ Bob moves, or more interestingly...
- ❖ Only ping moves to machine
- ❖ Moving ping has violation security policy on mobility

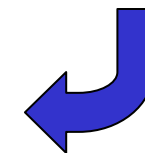
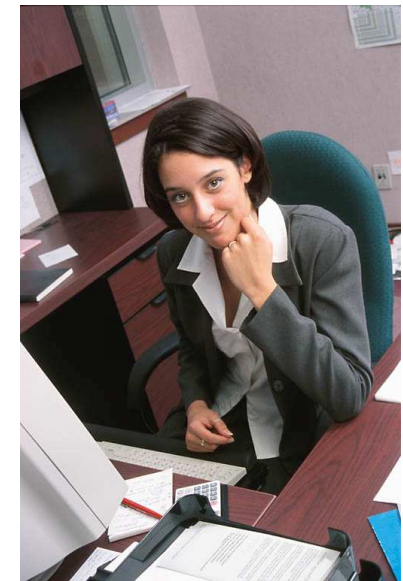
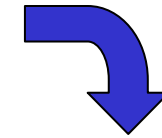


```
$ ping www.whitehouse.gov  
Usage: ping [-t] [-a] [-n count] ...
```

Data base

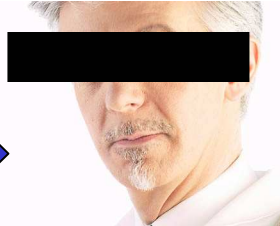
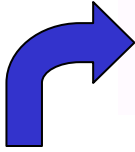
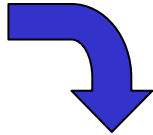
- ❖ Bob submits his tax return anywhere
- ❖ Alice receives Bob's tax return at the office
- ❖ Alice discusses Bob's tax return with her boss in the park
- ❖ Eve spies on Alice
- ❖ Alice has violated the security policy on mobility

Mobile computers



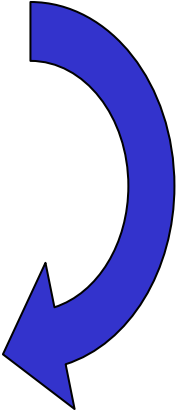
InspiredD
logo

Smart card



Mobile smart card

- ❖ Alice collects a card from the bank
- ❖ Bob offers 'special deal' if she pays with plastic...
- ❖ Can Bob be trusted?
- ❖ Get new security policy at the bank for 'special deals'



Modelling

- ❖ System: objects from the 3 worlds
- ❖ Policy: (un)acceptable behaviour
- ❖ Principle: guidance & formula
- ❖ Verification: $(\text{system} \parallel \text{policy}) \models \text{principle}$
- ❖ : communicating processes with a declaration

Rule out behaviour that has been deemed unacceptable
Examples:

- *Restrict access control*
- *Restrict information flow*
- *Maintain availability*

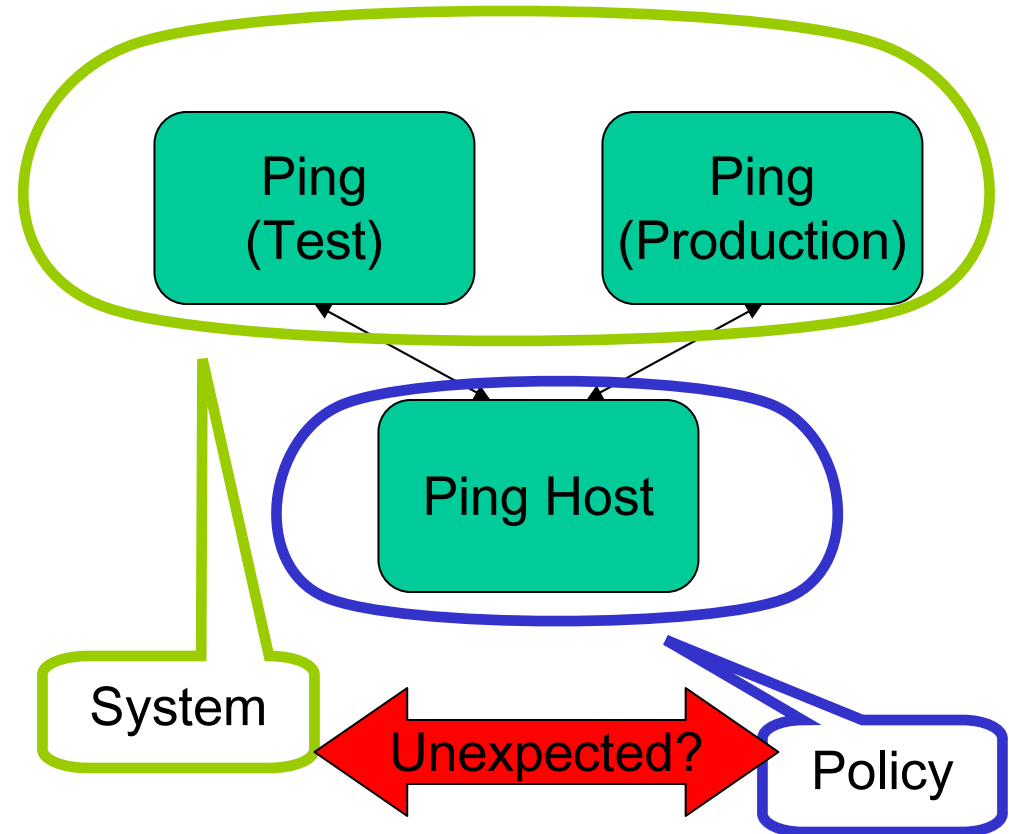
Identify & highlight security objectives. Examples:

- *Follow least privilege*
- *Compartmentalise*
- *Be reluctant to trust*

Host accepts “standard” ping only

❖ Principle of the “least privilege”

- Any system call except Socket(anywhere) *
- Socket(Test)
- Any system call except Socket(anywhere) *





System

Enumeration type

Channel

```

mtype = {
  Test_Env, Production_Env,
  Exit, Printf,
  Recvfrom, Sendto,
  Socket
};

```

```

chan c = [0] of {mtype, mtype} ;

```

Global state

```

active proctype system() {
  do
    :: c!Exit(env)
    :: c!Printf(env)
    :: c!Recvfrom(env)
    :: c!Sendto(env)
    :: c!Socket(env)
  od
}

```

```

byte env = Test_Env ;

```

```

active proctype mobility() {
  do
    :: env = Test_Env
    :: env = Production_Env
  od
}

```

Two concurrent processes

Non deterministic choice

Policy & Principle

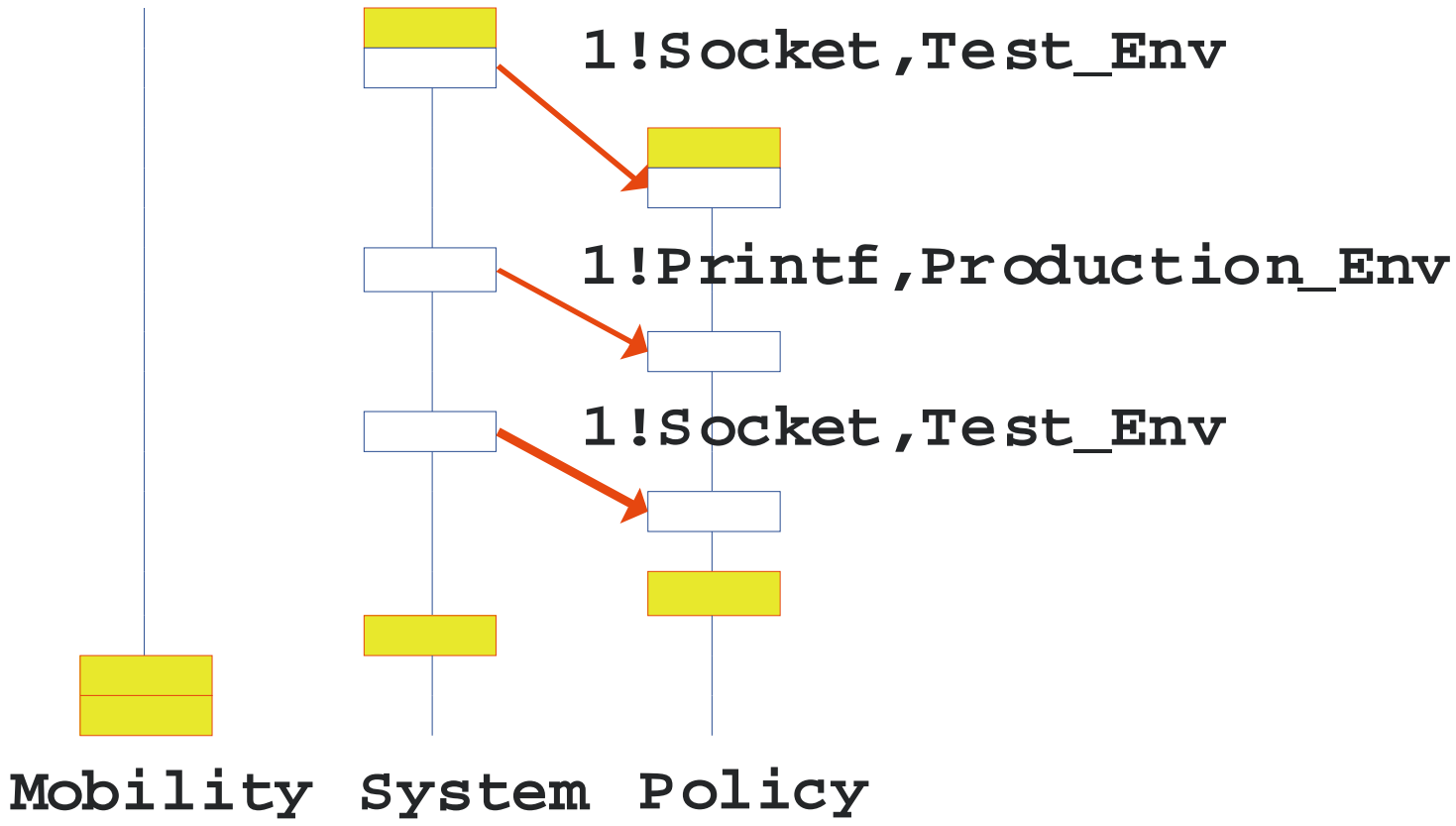
```
active proctype policy() {
  do
    :: c?Printf(_);
    c?Exit(_);
    :: c?Socket(_);
    c?Printf(_);
  do
    :: c?Sendto(_);
    c?Recvfrom(_);
    :: break
  od
od
}
```

```
trace {
  do
    :: c?Exit(_);
    :: c?Printf(_);
    :: c?Recvfrom(_);
    :: c?Sendto(_);
    :: c?Socket(Test_Env) -> break
  od;
  do
    :: c?Exit(_);
    :: c?Printf(_);
    :: c?Recvfrom(_);
    :: c?Sendto(_);
  od
}
```

Based on
work by
Mycroft et al



Counter example

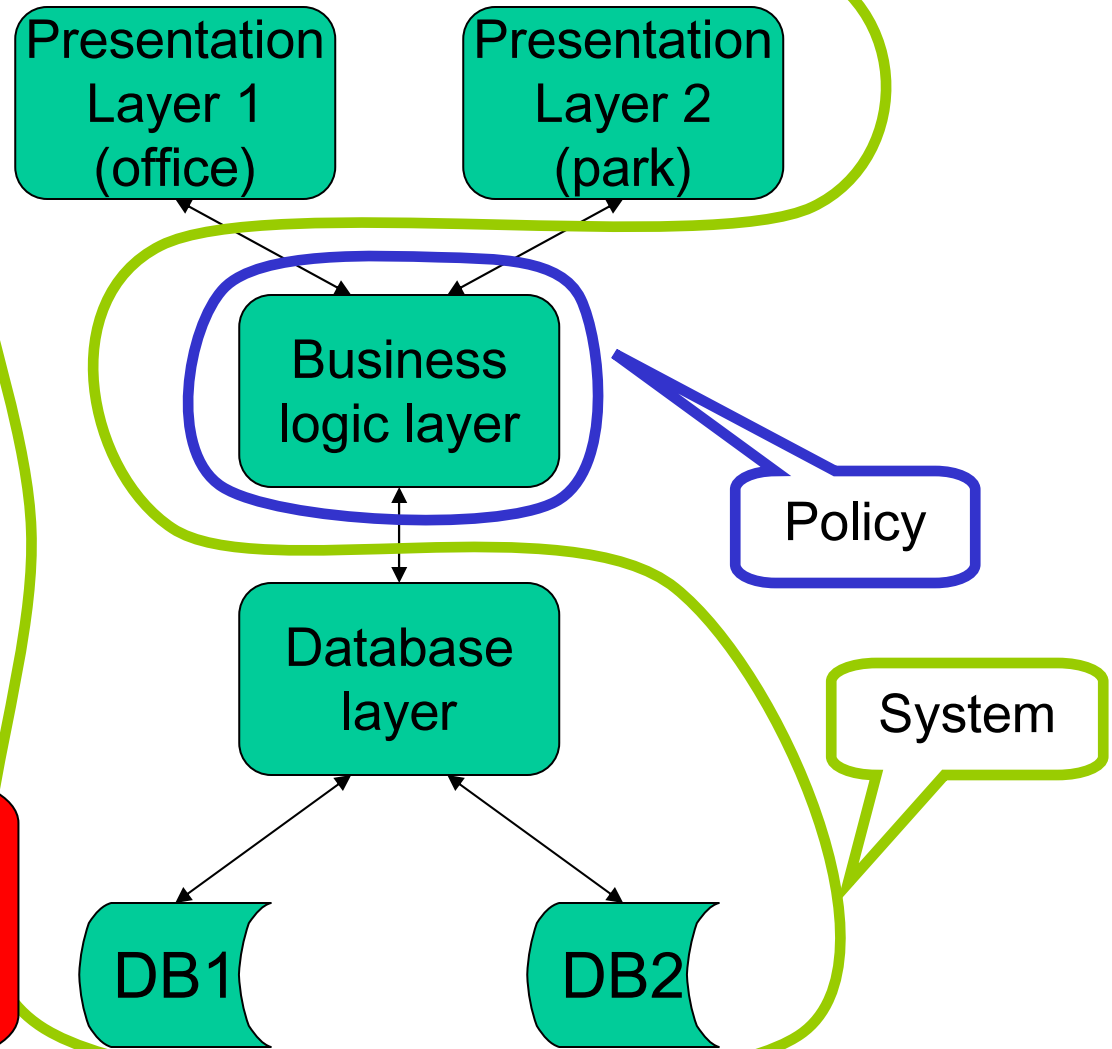


Location determines data base

❖ Principle
“Compartmentalise”

- Connect(here)
- Request(here)
- Reply(here)
- Request(here)
- Reply(here)
- ...
- Disconnect(here)

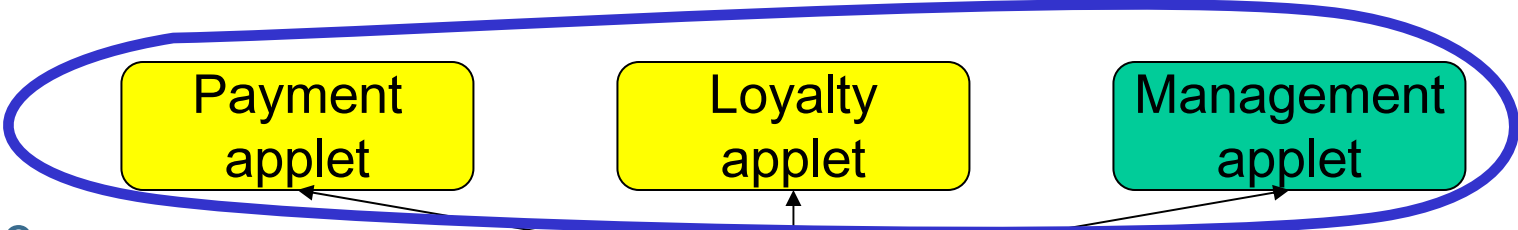
Not checked by
Standard policy





Location determines applet

Policy



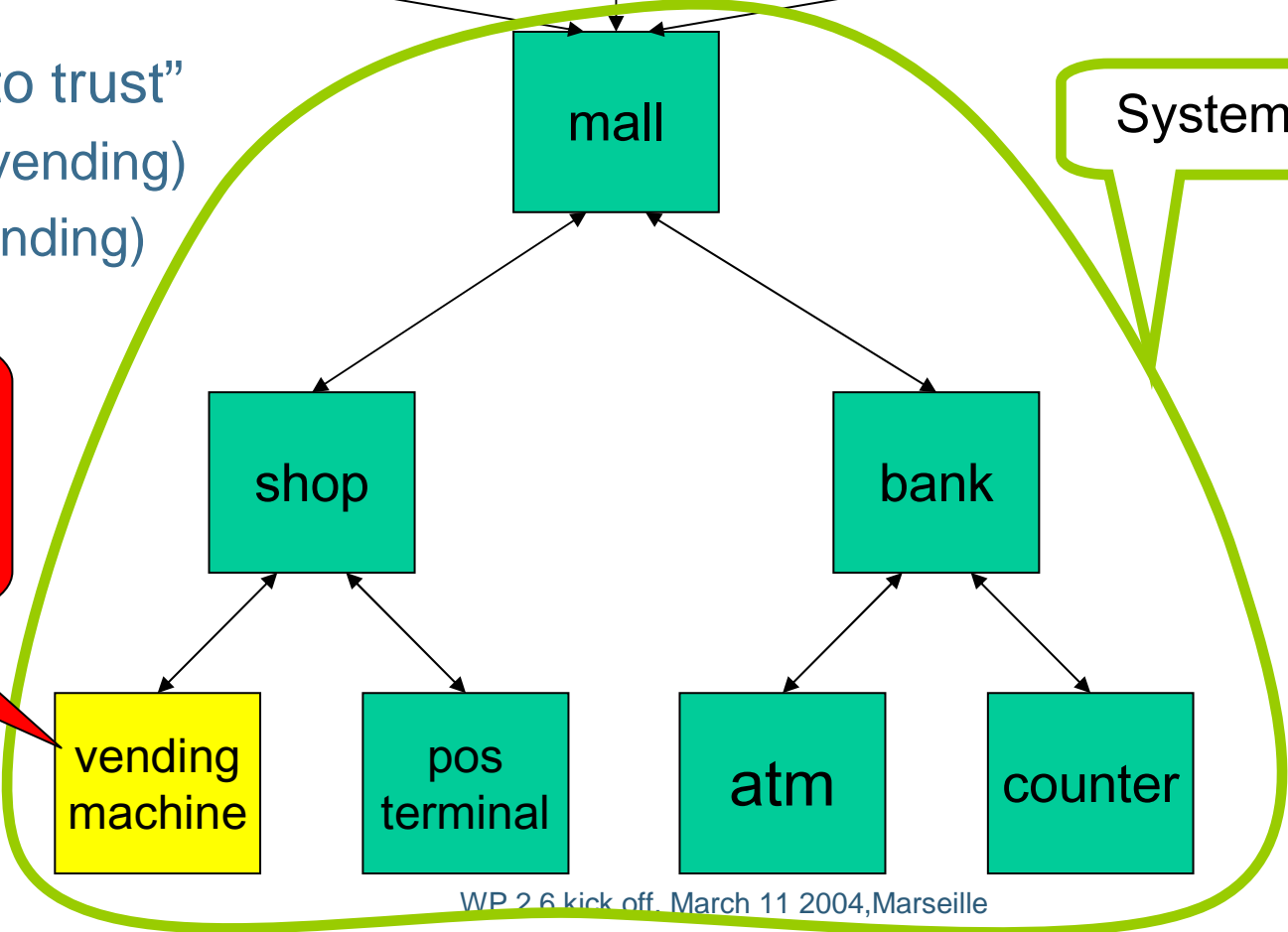
Principle

“Be reluctant to trust”

- Payment(vending)
- Loyalty(vending)

System

Not checked by Standard policy



Conclusions and future work

- ❖ We can analyse policies, showing that none of the models satisfy our principles
- ❖ Trace declarations limited, more tool support needed
- ❖ Models should be more systematic, and hierarchical
- ❖ How can we enforce policies?