Synthesizing Multi-View Models of Software Systems

PhD dissertation

Lambeau Bernard

ICTeam Institute

Université catholique de Louvain

November 2011

Outline

- Introduction
  - A step-by-step explanation of the thesis title
- Two model synthesis techniques
  - Model synthesis for formal process analysis
  - Inductive synthesis of state machines from scenarios
- Conclusion
  - Stuff you should remember

Synthesizing Multi-view Models of Software Systems

- A *system* is a set of active components, called *agents*, that behave and interact so as to fulfill goals
- Agents restrict their behavior to ensure the goals they are responsible for [Fea87, Avl09]

Synthesizing Multi-view Models of Software Systems

- A *system* is a set of active components, called *agents*, that behave and interact so as to fulfill goals
- Agents restrict their behavior to ensure the goals they are responsible for [Fea87, Avl09]
- Some agents are *software* components, *i.e.* automated agents

Examples

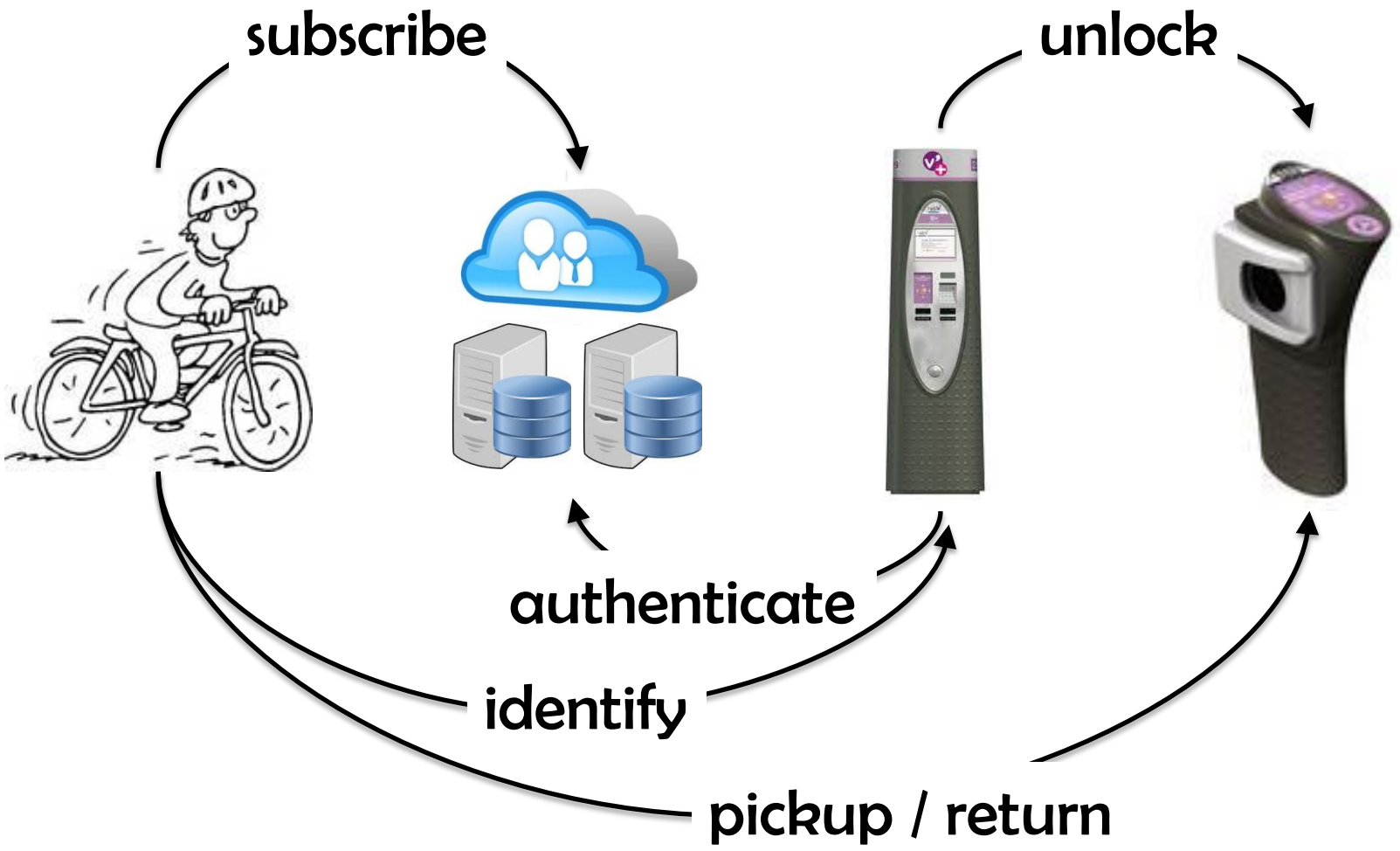
Examples

Examples

with courtesy of quechoisir.fr

Examples

Synthesizing Multi-view Models of Software Systems



Building software systems is hard

The solution is highly technical

website

mails, sms, etc.

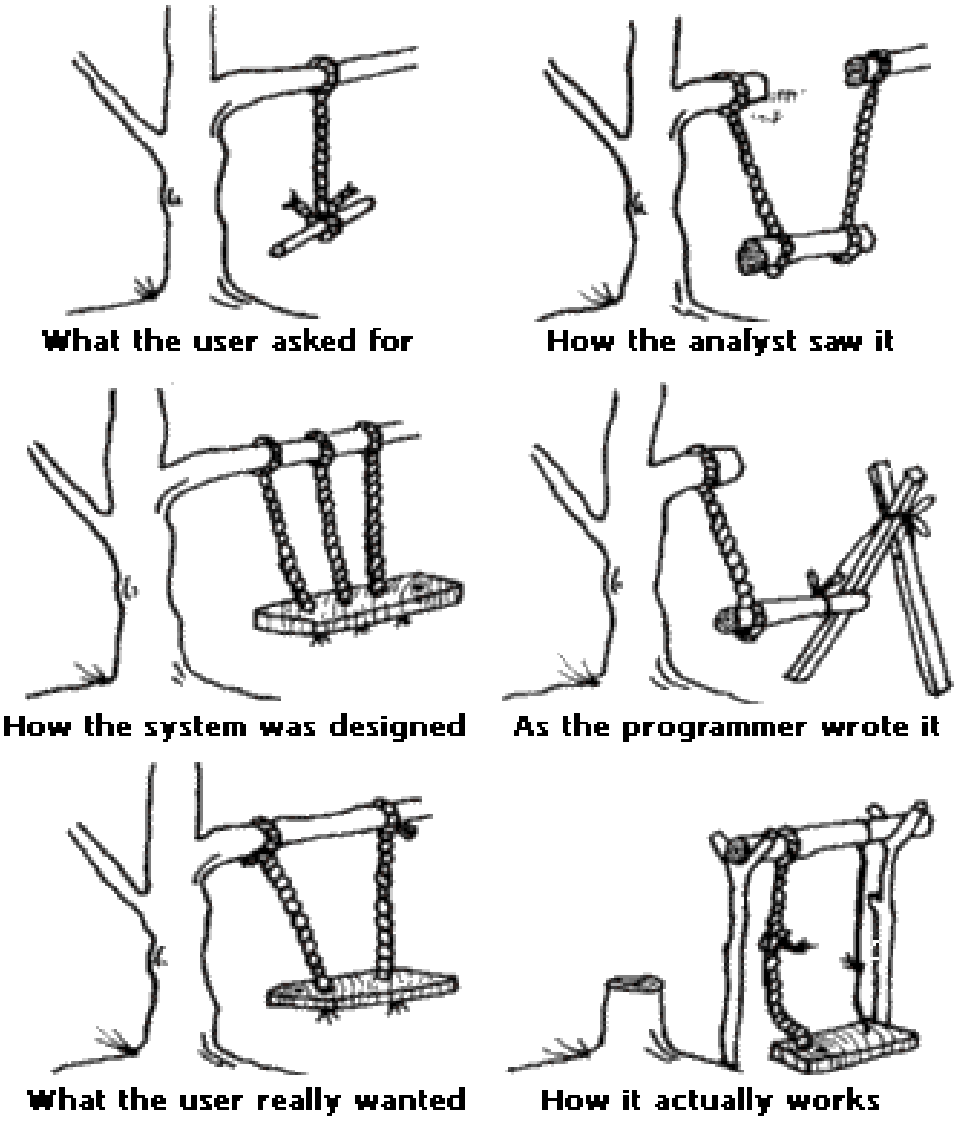
internet

electro-mechanical devices

smart card, keyboard, touch screen

What is the problem?

The hardest part of software development is determining what the system should do and why it should do so [Bro87]



What is the problem?

The hardest part of software development is determining what the system should NOT do and why it should NOT do so

Synthesizing Multi-view Models of Software Systems

- Models help reasoning about the problem
  - Elaborating requirements and exploring designs
  - Abstracting from numerous details

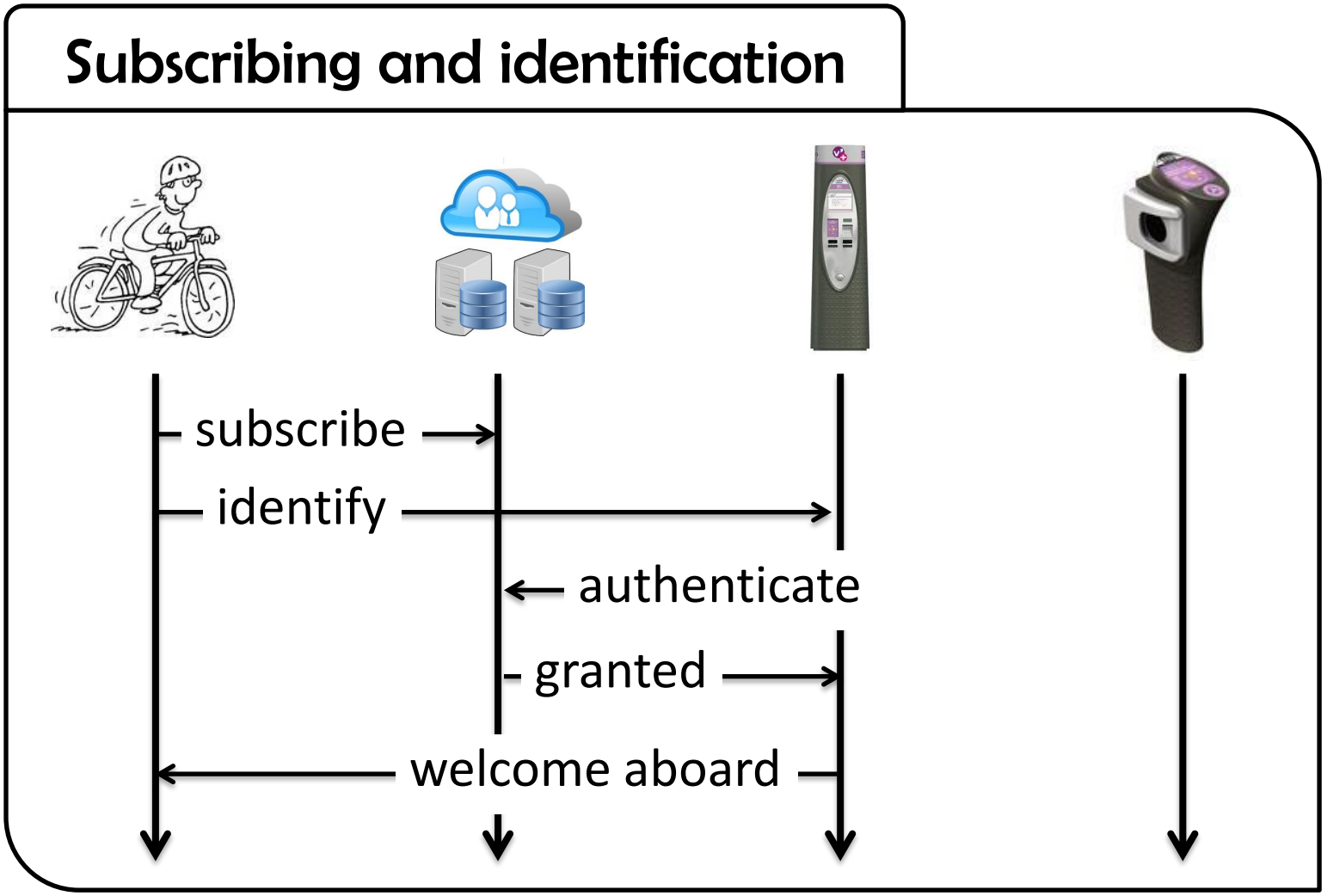



- They also help building the solution
  - Code generation from higher-level abstractions
  - Design documentation

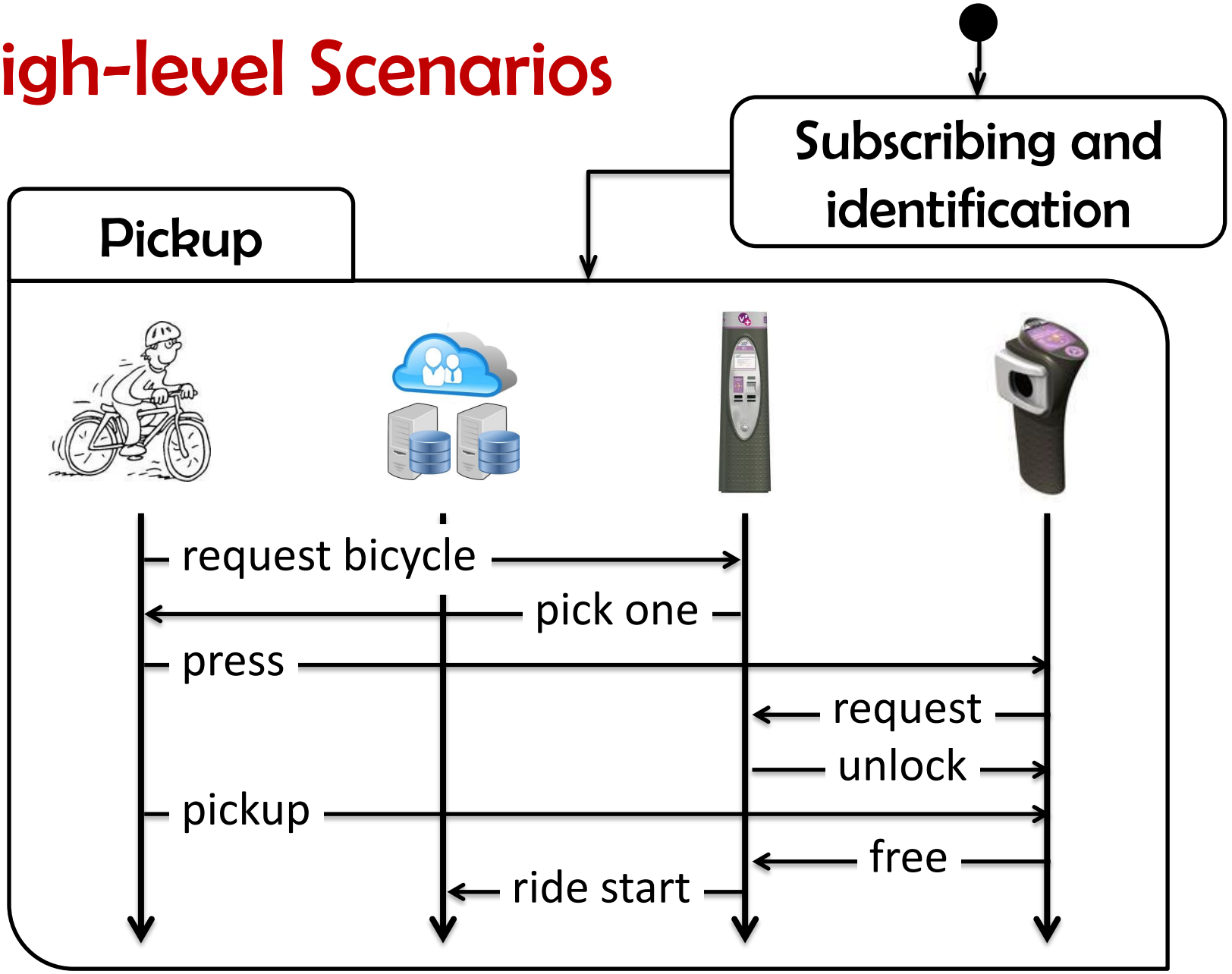
Synthesizing Multi-view Models of Software Systems

- Different models capture different system aspect
  - Structural: agents and their interfaces?
  - Behavioral: how do they behave?
  - Intentional: why do they behave that way?
  - Operational: what tasks, in what order?
- Models also overlap in their description of the target system

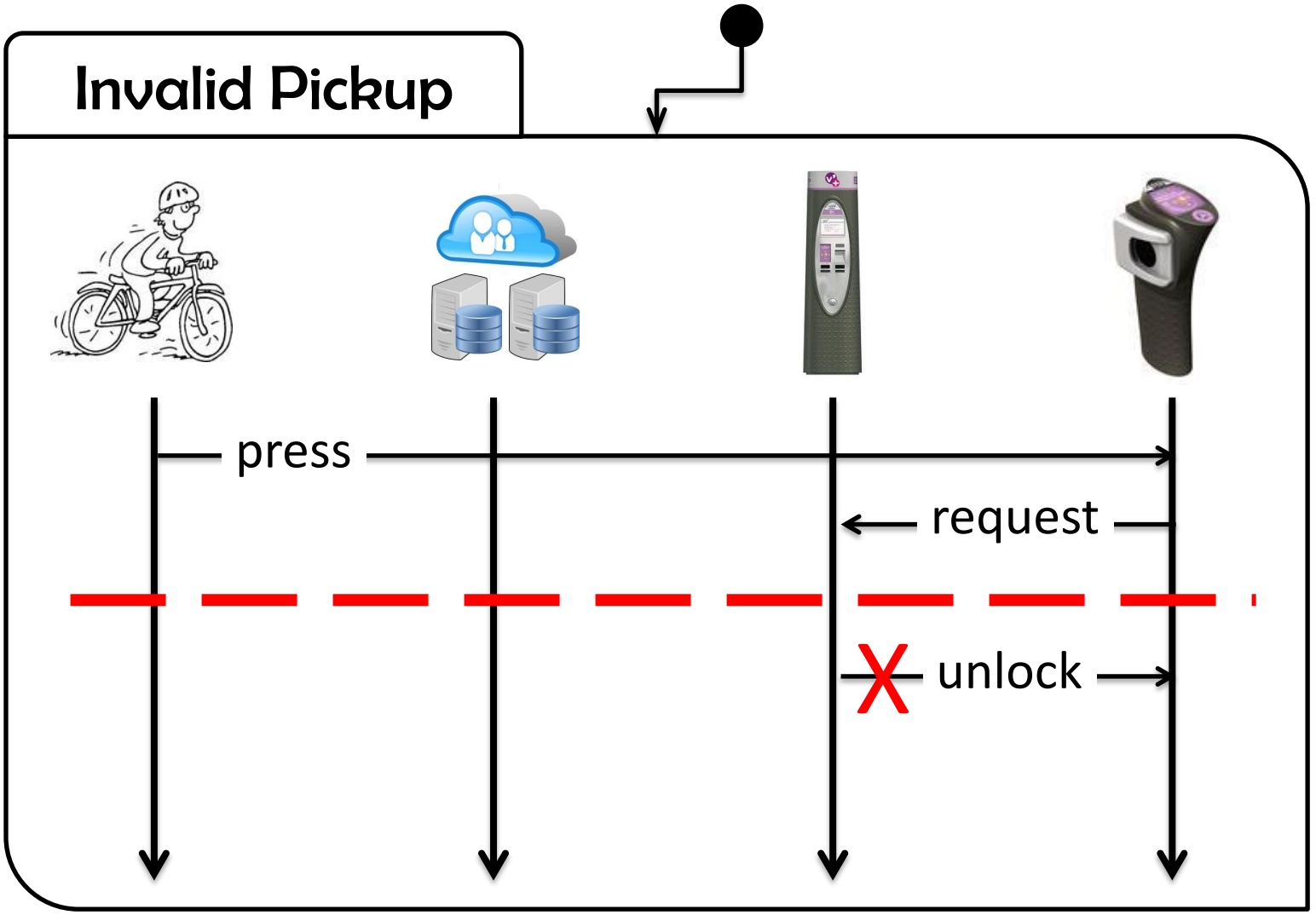
Scenarios


Subscribing and identification
subscribe
identify
authenticate
granted
welcome aboard

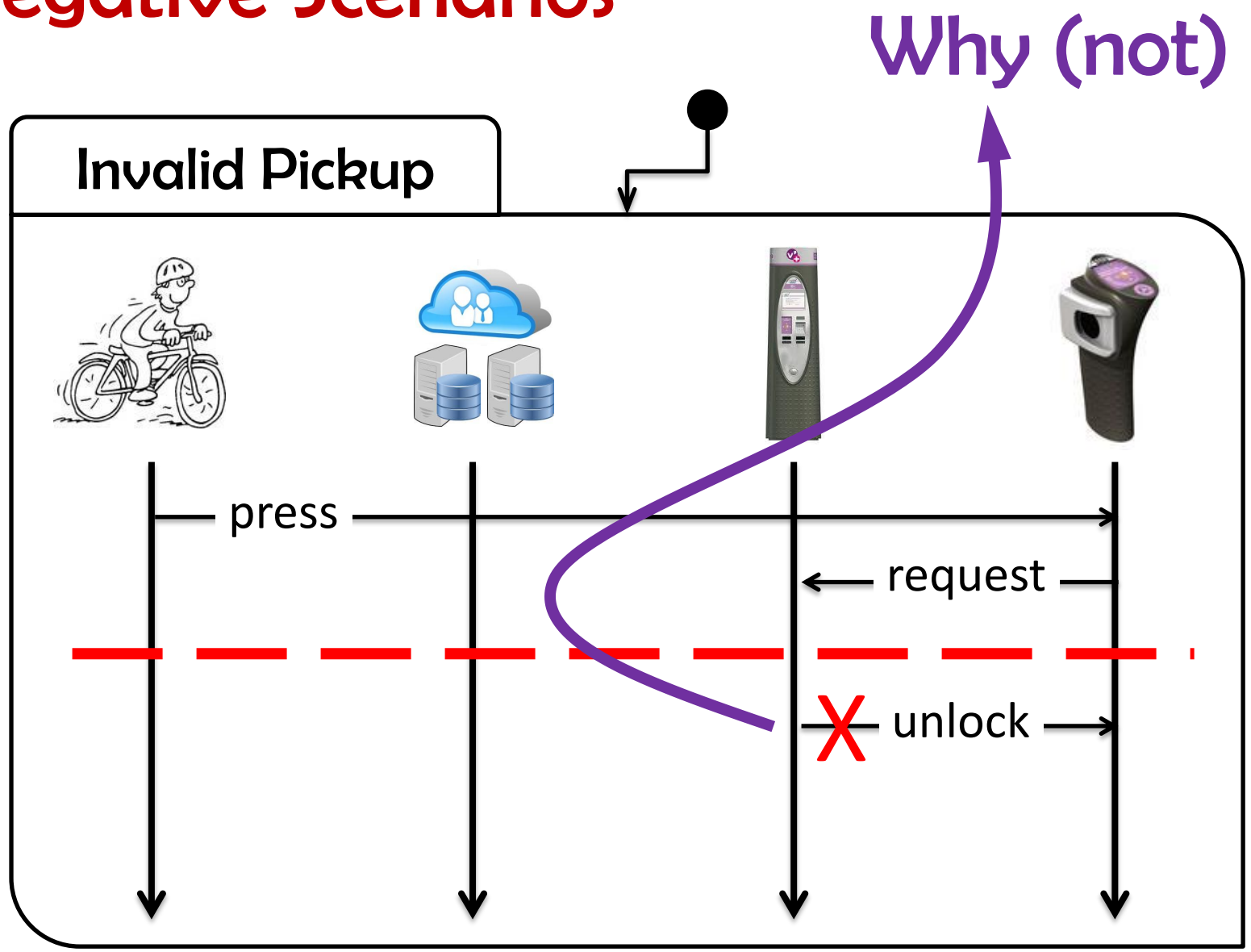
High-level Scenarios



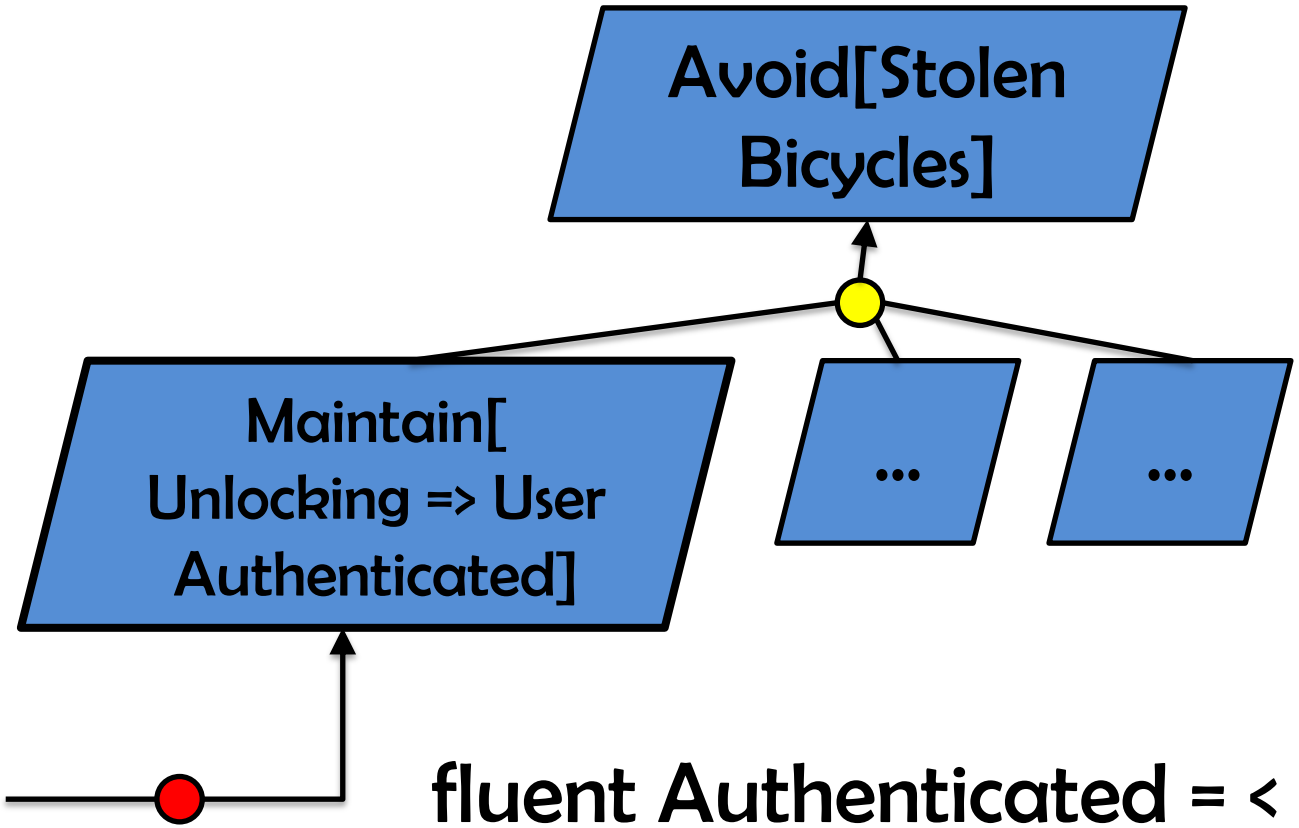
Negative Scenarios



Negative Scenarios

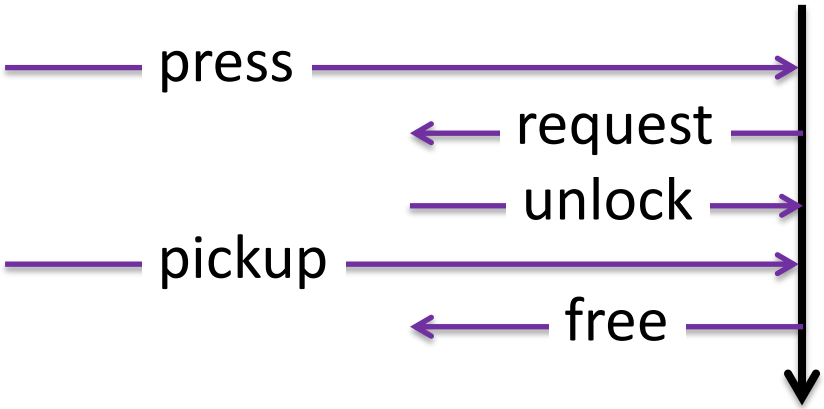


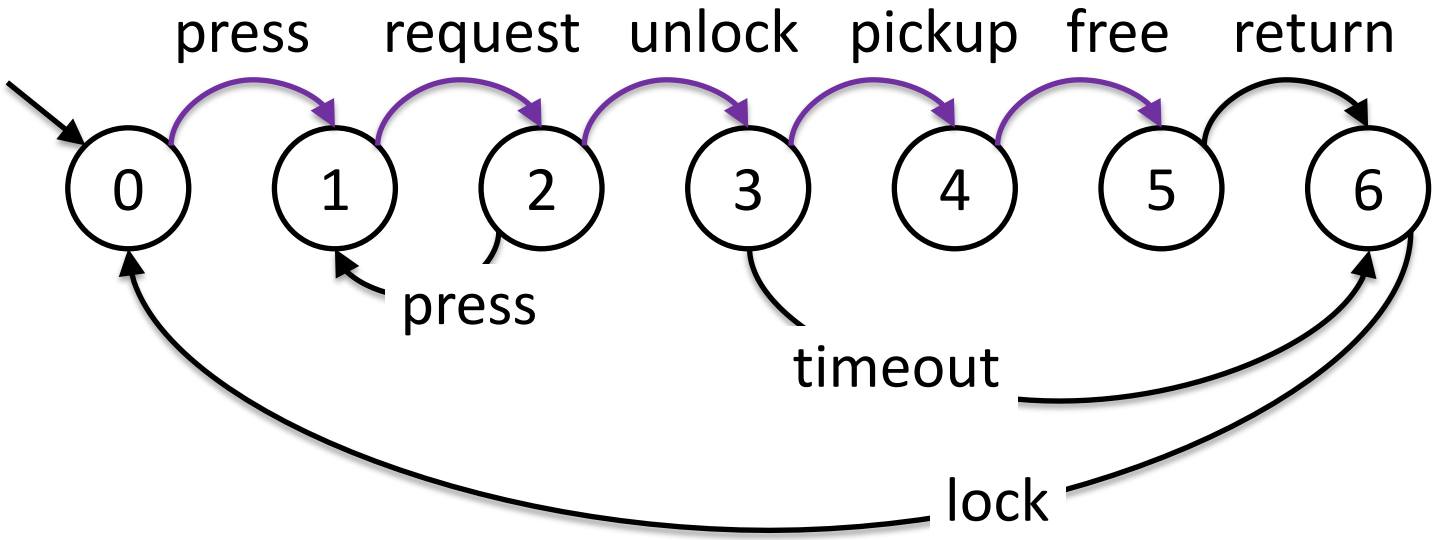
Goals & Requirements



fluent Authenticated = <
{ access granted },
{ ride start } > initially false

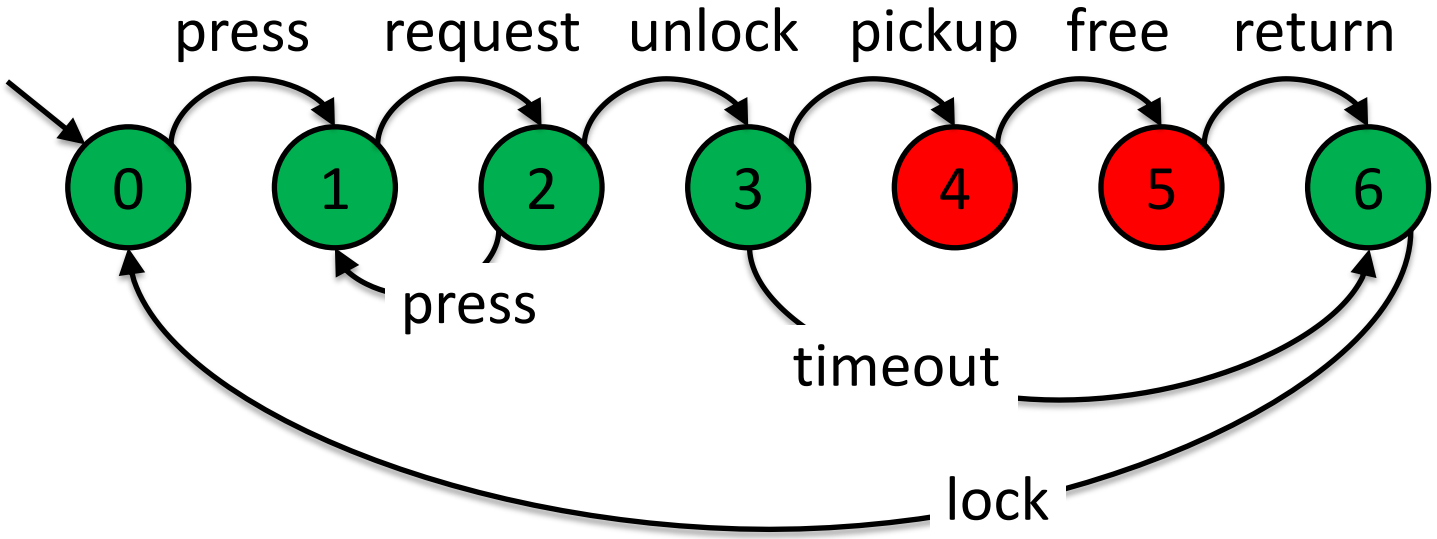
Agent state machines





Agent state variables

fluent BicyclePresent = <
{ return },
{ pickup } > initially true

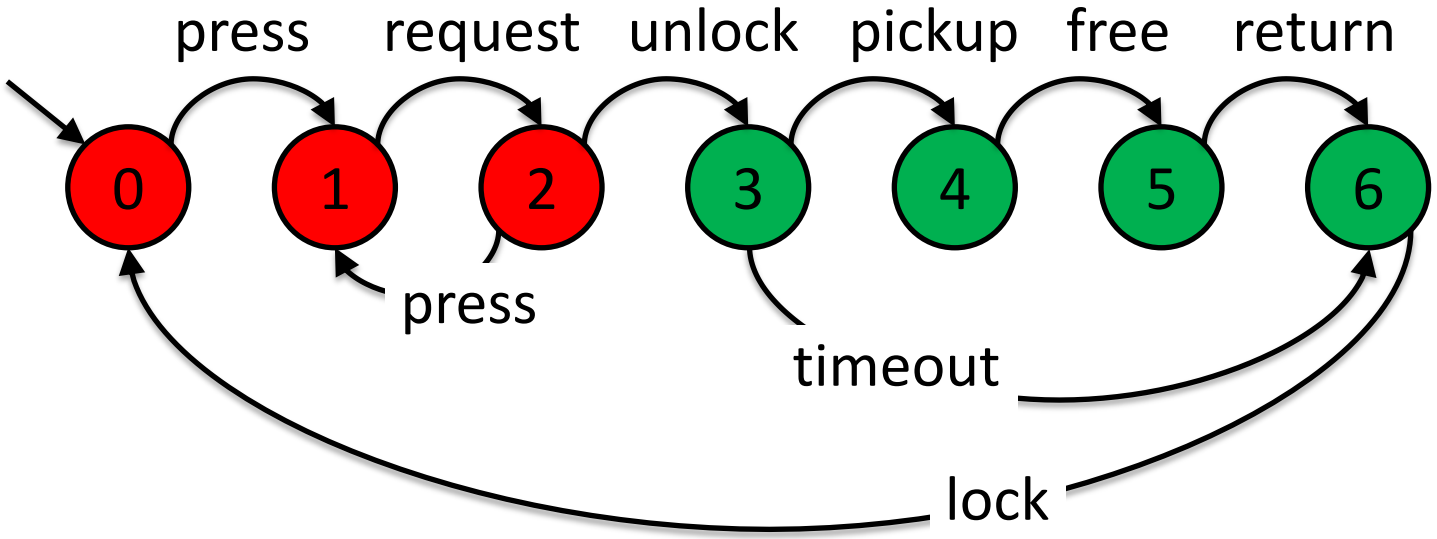


Agent state variables

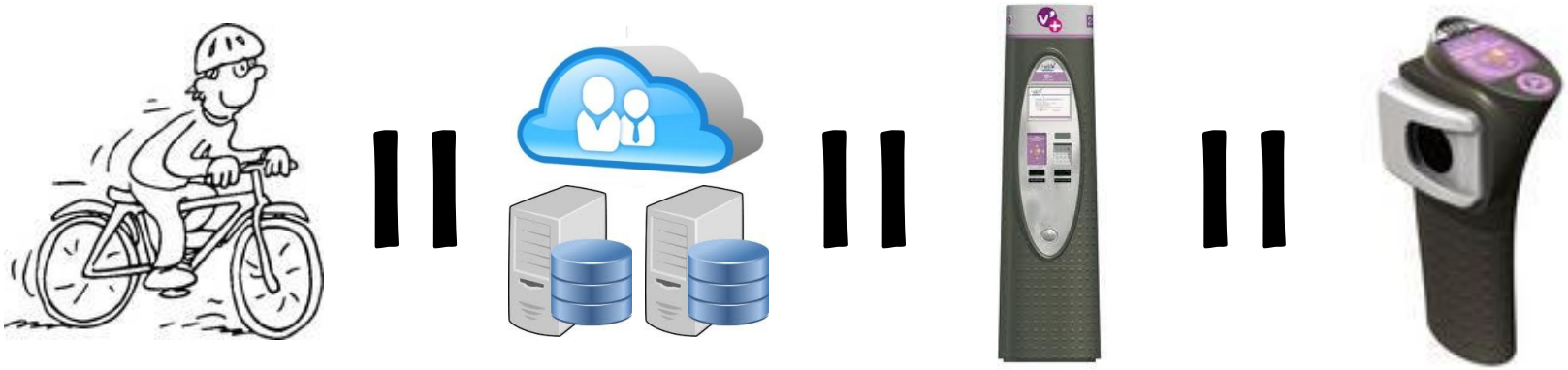
fluent Locked = <

{ lock },

{ unlock } > initially true

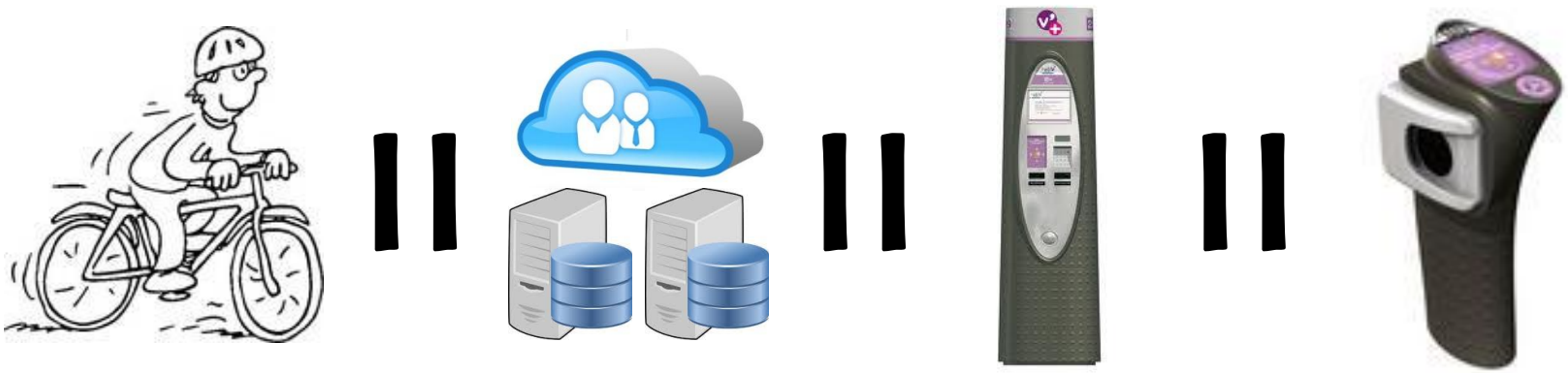


System behavior



- Parallel composition of agent behaviors
  - Agents behave asynchronously but synchronize on shared events
  - System behavior captured through a state machine

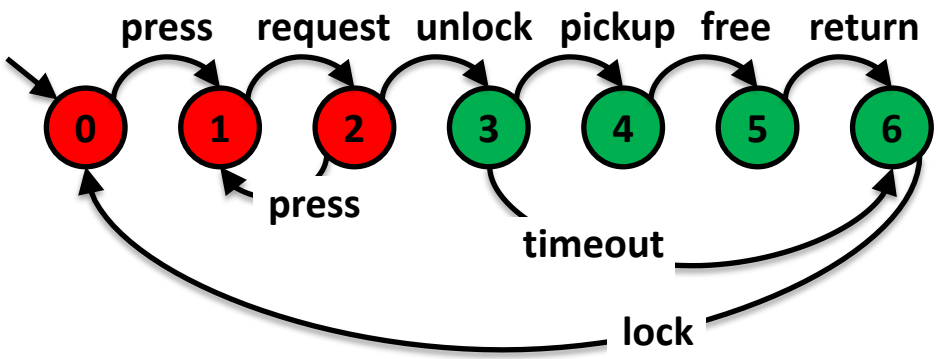
Modeling software systems is hard



- High-quality models should be
  - Adequate, consistent, complete, precise, analyzable, comprehensible, etc.
- Natural consequence of the "building software systems is hard" claim

Synthesizing Multi-view Models of Software Systems

Model synthesis for formal analysis



$\neg$ Locked $\Rightarrow$
o ($\neg$ BicyclePresent
$\vee$ Locked)