# *Modeling the Applet Life Cycle and Security Protocols in JML*

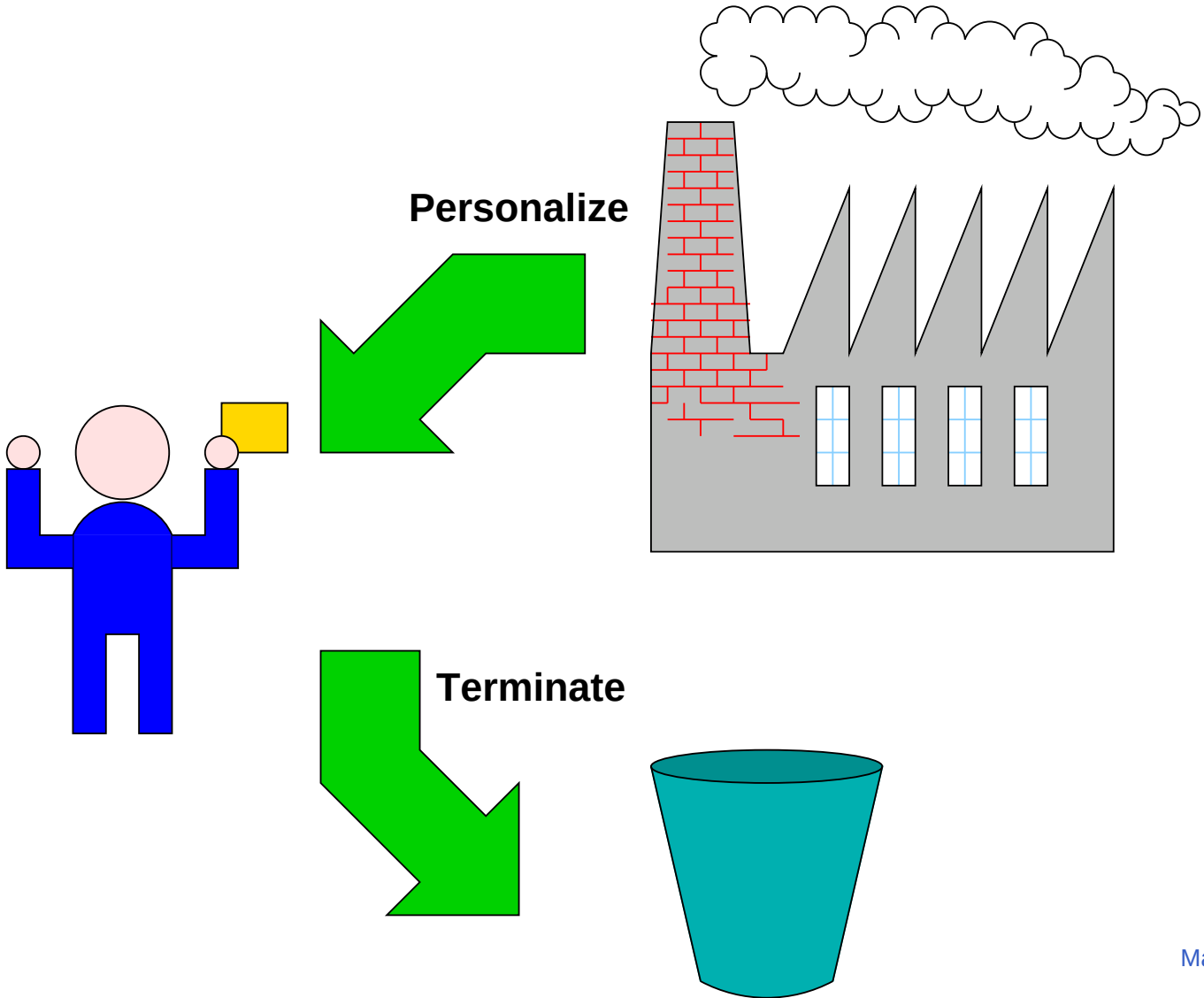
***Martijn Oostdijk***

***University of Nijmegen***

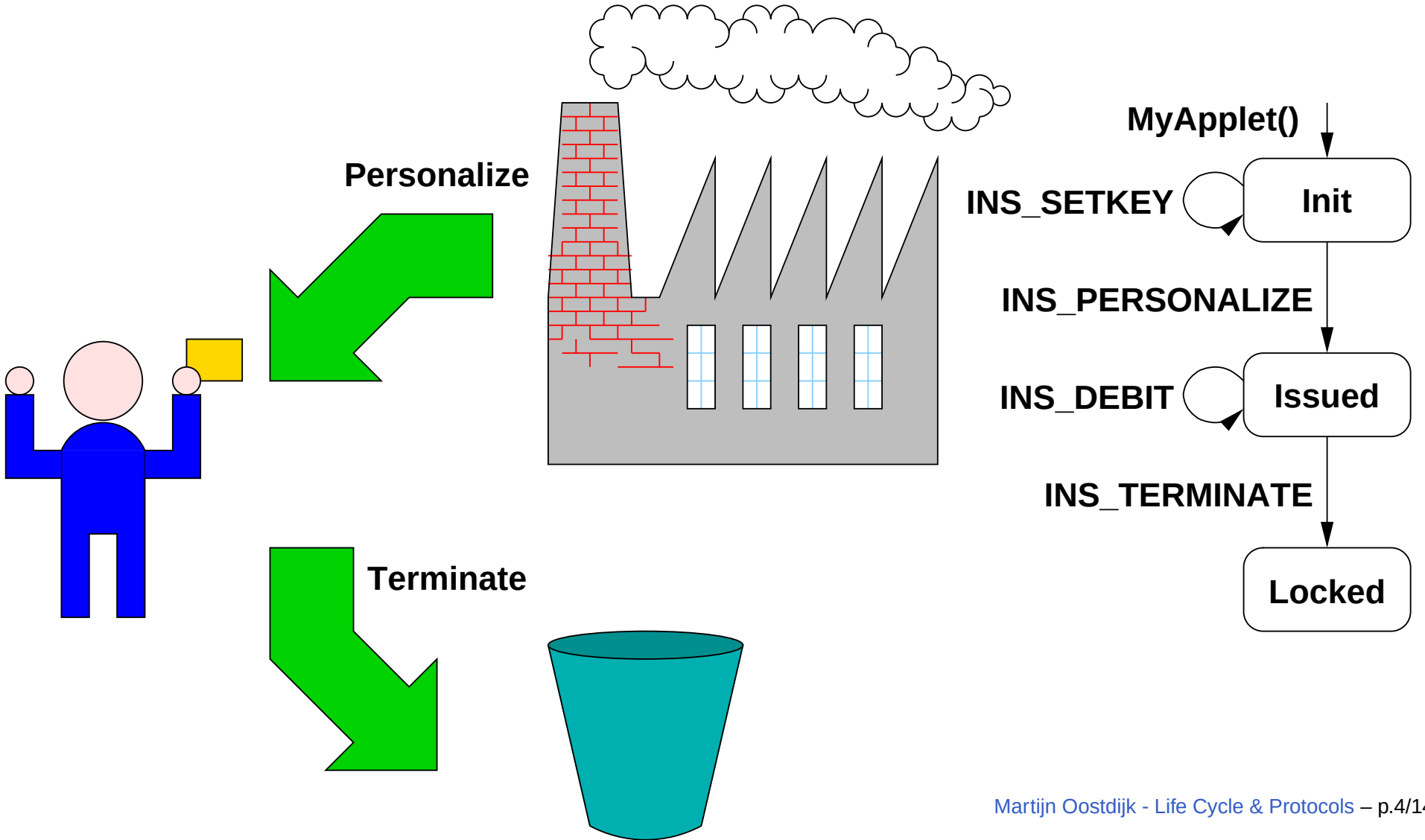
# Overview of this talk

- The Applet Life Cycle
  - Applet Life Cycle: Implementation
  - Applet Life Cycle: Specification
- Security Protocols
  - Security Protocols: Implementation
  - Security Protocols: Specification
- Conclusions

# The Applet Life Cycle



# The Applet Life Cycle



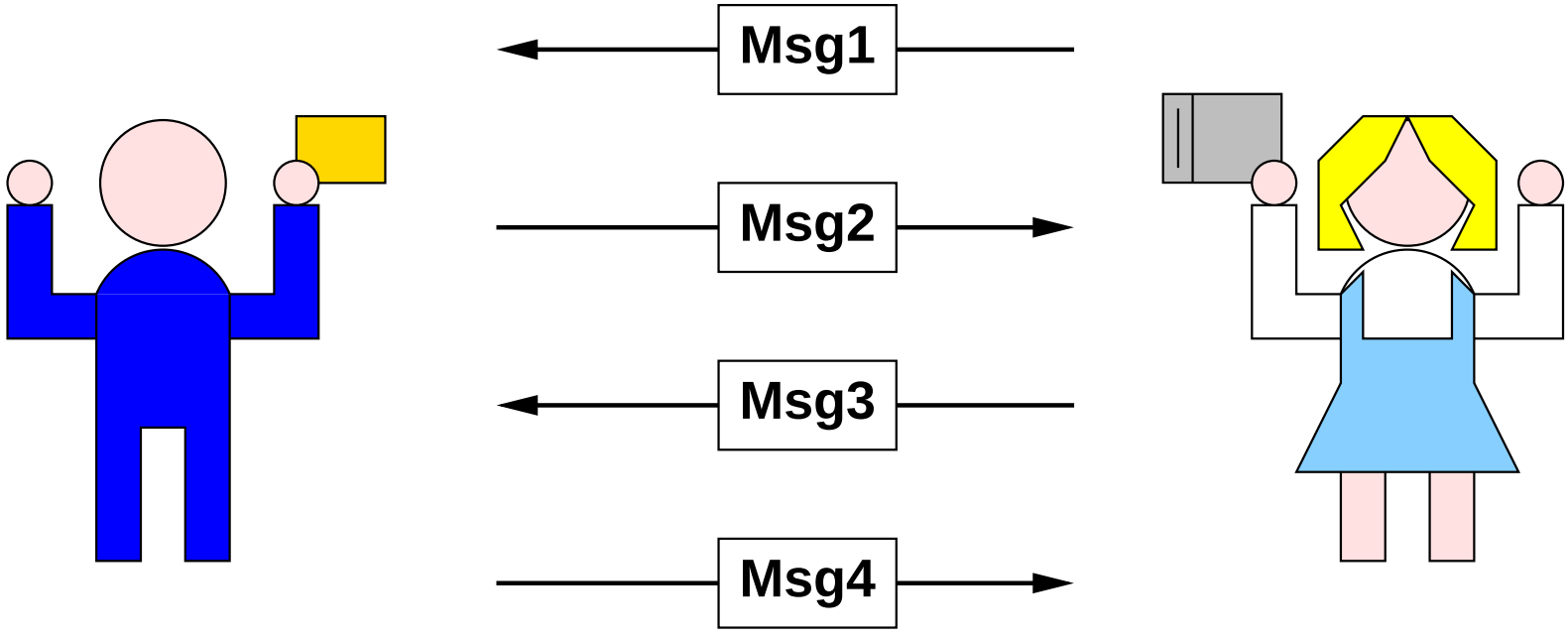
# Applet Life Cycle: Implementation

```
static final short STATE_INIT = 0;
static final short STATE_ISSUED = 1;
static final short STATE_LOCKED = 2;
short state;

public void process(APDU apdu) {
  switch (state) {
    case STATE_INIT:
      switch (buffer[OFFSET_INS]) {
         case INS_SETKEY: processSetKey(apdu); break;
         ...
      }
      break;
    case STATE_ISSUED:
      switch (buffer[OFFSET_INS]) {
        case INS_DEBIT: ...; break;
```

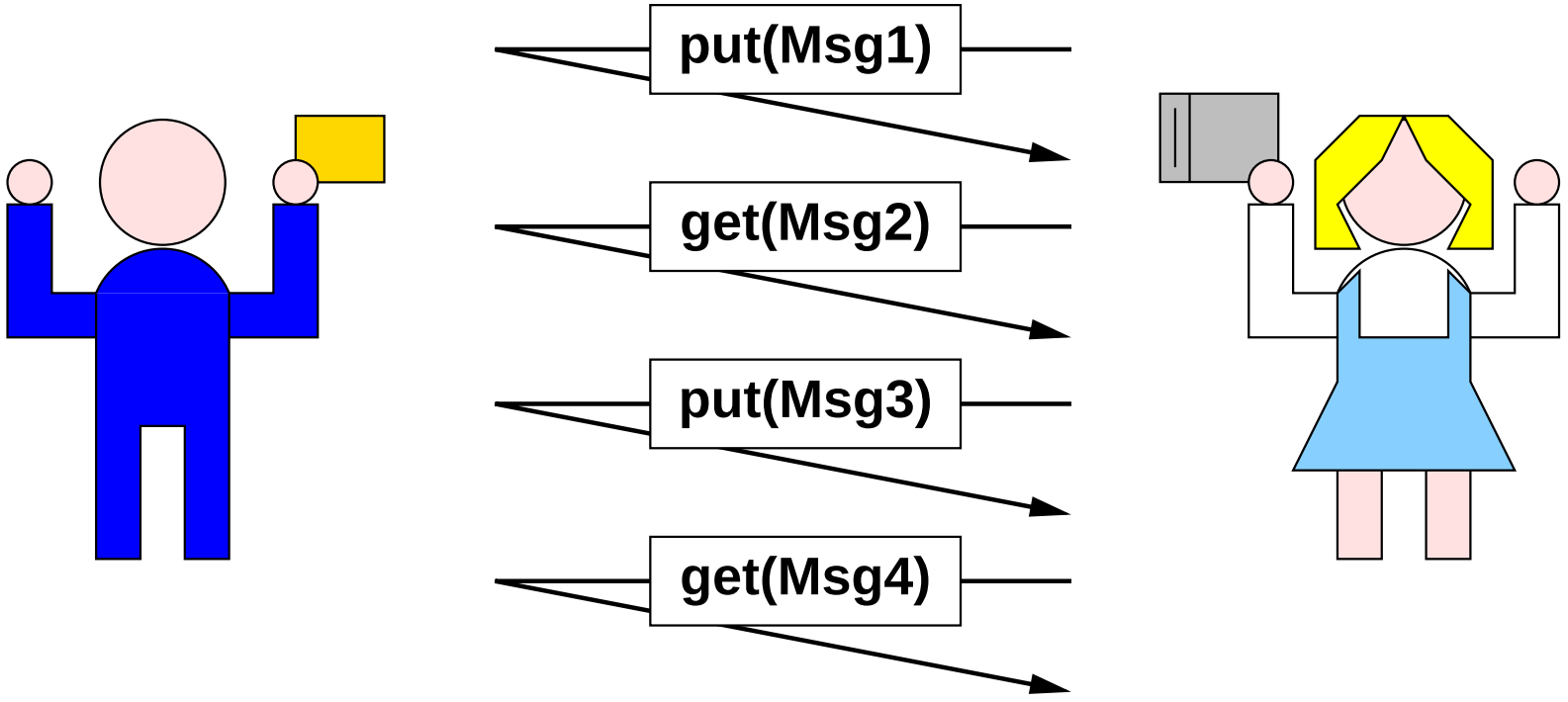
# Applet Life Cycle: Specification

```
/*@ invariant state == STATE_INIT ||
  @   state == STATE_ISSUED || state == STATE_LOCKED; */

/*@ constraint \old(state == STATE_INIT) ==>
  @   state == STATE_INIT || state == STATE_ISSUED; */

/*@ public behavior
  @   requires state == STATE_INIT;
  @   assignable state;
  @   ensures state == STATE_INIT;
  @   signals (ISOException) state == \old(state); */
public void processSetKey(/*@ non_null */ APDU apdu) {
```

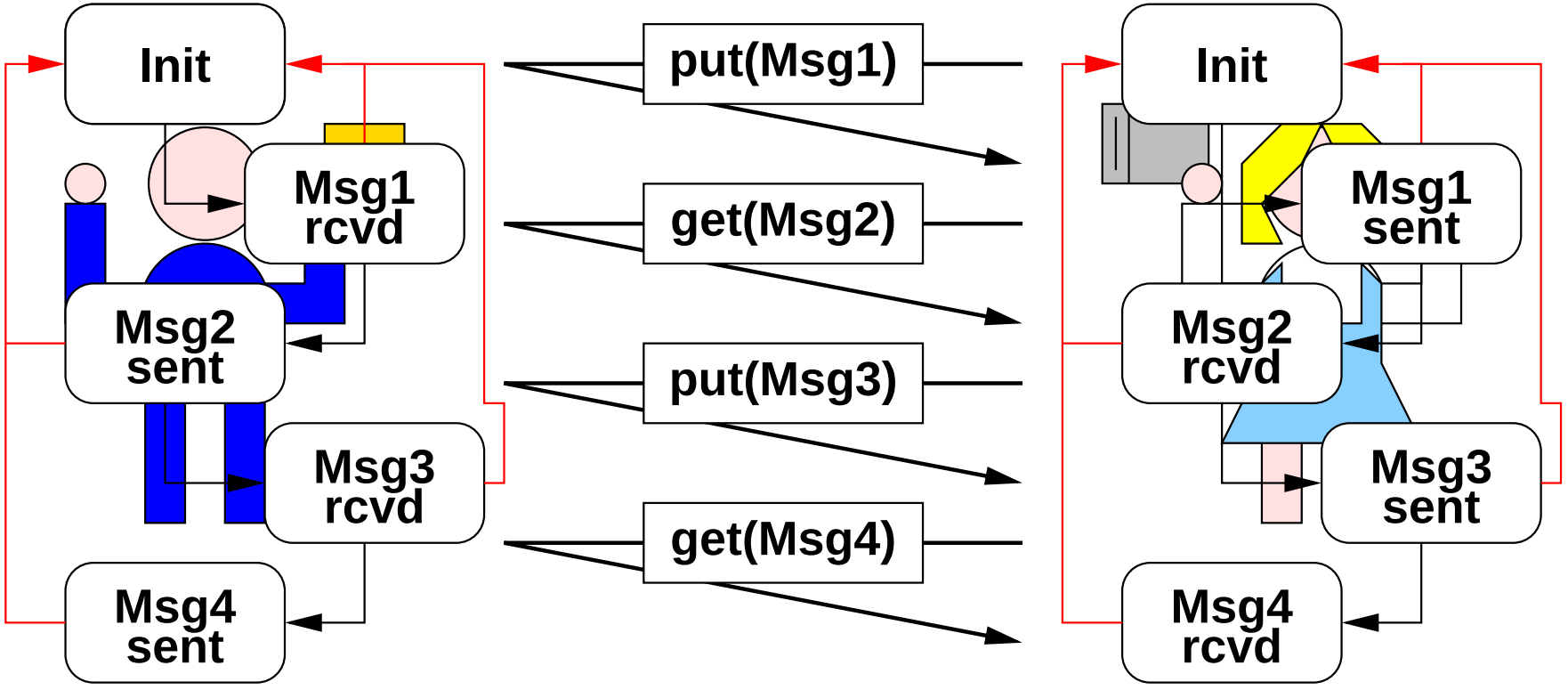
# Security Protocols



# Security Protocols



# Security Protocols



# Security Protocols: Implementation

```
static final short PSTATE_INIT = 0;
static final short PSTATE_MSG1_RECEIVED = 1;
static final short PSTATE_MSG2_SENT = 2;
static final short PSTATE_MSG3_RECEIVED = 3;
static final short PSTATE_MSG4_SENT = 4;
short[] protocolState; // points to CoR trans.mem...

public void processMsg1(APDU apdu) {
  if (protocolState[0] == PSTATE_INIT &&
    apdu.getBuffer()[OFFSET_INS] == INS_PUT_MSG1) {
    ...;
    protocolState[0] = PSTATE_MSG1_RECEIVED;
  } else {
    protocolState[0] = PSTATE_INIT;
    ISOException.throwIt(SW_SECURITY_STATUS_NOT_SATISFIED);
  }
}
```

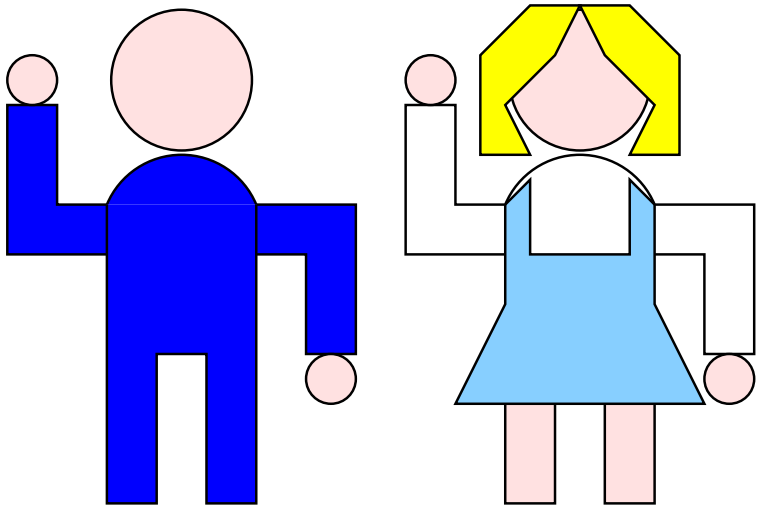
# Security Protocols: Specification

```
/*@ public behavior
  @    assignable protocolState[0];
  @    ensures \old(protocolState[0] == PSTATE_INIT);
  @    ensures protocolState[0] == PSTATE_MSG1_RECEIVED;
  @    signals (ISOException)
  @       \old(protocolState[0] != PSTATE_INIT);
  @    signals (ISOException)
  @       protocolState[0] == PSTATE_INIT;
  */
public void processMsg1(/*@ non_null */ APDU apdu) {
   ...
}
```

# Conclusions

- Applet Life Cycle
  - Simple finite state machine
  - Implement using a `state` field
  - Transitions correspond to instructions/methods
  - Pre- and post-condition clauses
  - Invariant: no way back
- Security Protocols
  - Each principal is a finite state machine
  - Usually protocol state can be in transient memory
  - Messages need to be mapped on APDUs
  - Go back to initial state in case of exception

# That’s all!



# That’s all!