

Rethinking Software Systems
A Friendly Introduction to
Behavioral Programming

Michael Bar-Sinai
@michbarsinai

Behavioral Programming (BP)

Programming Paradigm

Introduced in 2010 by David Harel, Assaf Marron and Gera Weiss

Rooted in Scenario-Based Programming (Damm, Harel, Marelly)

Large body of scientific work around this: algorithms, visualizations, event selection, interoperation.

Implementations in Java, C++, Erlang, StateCharts



DEVOXX 2018

first major non-academic conference

hosting a BP talk

DEVOXX 2018

first major non-academic conference

hosting a BP talk

Behavioral Programming with React
Luca Matteis, ReactjsDay, Italy, October 2018



Programming

Modeling

Programming

Modeling



Programming

Telling a computer

What to do

Modeling

Programming

Telling a computer

What to do

Modeling

Telling a computer

What could be done

Programming

Telling a computer

What to do

Modeling

Telling a computer

**What could be done
can't be done**

Programming

Telling a computer

What to do

Modeling

Telling a computer

**What could be done
can't be done**



Hello, world!

```
bp.registerBThread("bt-1", function(){  
  bp.sync({request:bp.Event("hello")});  
});
```

```
bp.registerBThread("bt-2", function(){  
  bp.sync({request:bp.Event("world")});  
});
```

Using BPjs, <https://github.com/bThink-BGU/BPjs>

Hello, world!

```
bp.registerBThread("bt-1", function(){  
  bp.sync({request:bp.Event("hello")});  
});
```

```
bp.registerBThread("bt-2", function(){  
  bp.sync({request:bp.Event("world")});  
});
```

Using BPjs, <https://github.com/bThink-BGU/BPjs>

Hello, world!

```
bp.registerBThread("bt-1", function(){  
  bp.sync({request:bp.Event("hello")});  
});
```

```
bp.registerBThread("bt-2", function(){  
  bp.sync({request:bp.Event("world")});  
});
```

Using BPjs, <https://github.com/bThink-BGU/BPjs>

Hello, world!

```
bp.registerBThread("bt-1", function(){  
  bp.sync({request:bp.Event("hello")});  
});
```

```
bp.registerBThread("bt-2", function(){  
  bp.sync({request:bp.Event("world")});  
});
```

Using BPjs, <https://github.com/bThink-BGU/BPjs>

Hello, world!

```
bp.registerBThread("bt-1", function(){  
  bp.sync({request:bp.Event("hello")});  
});
```

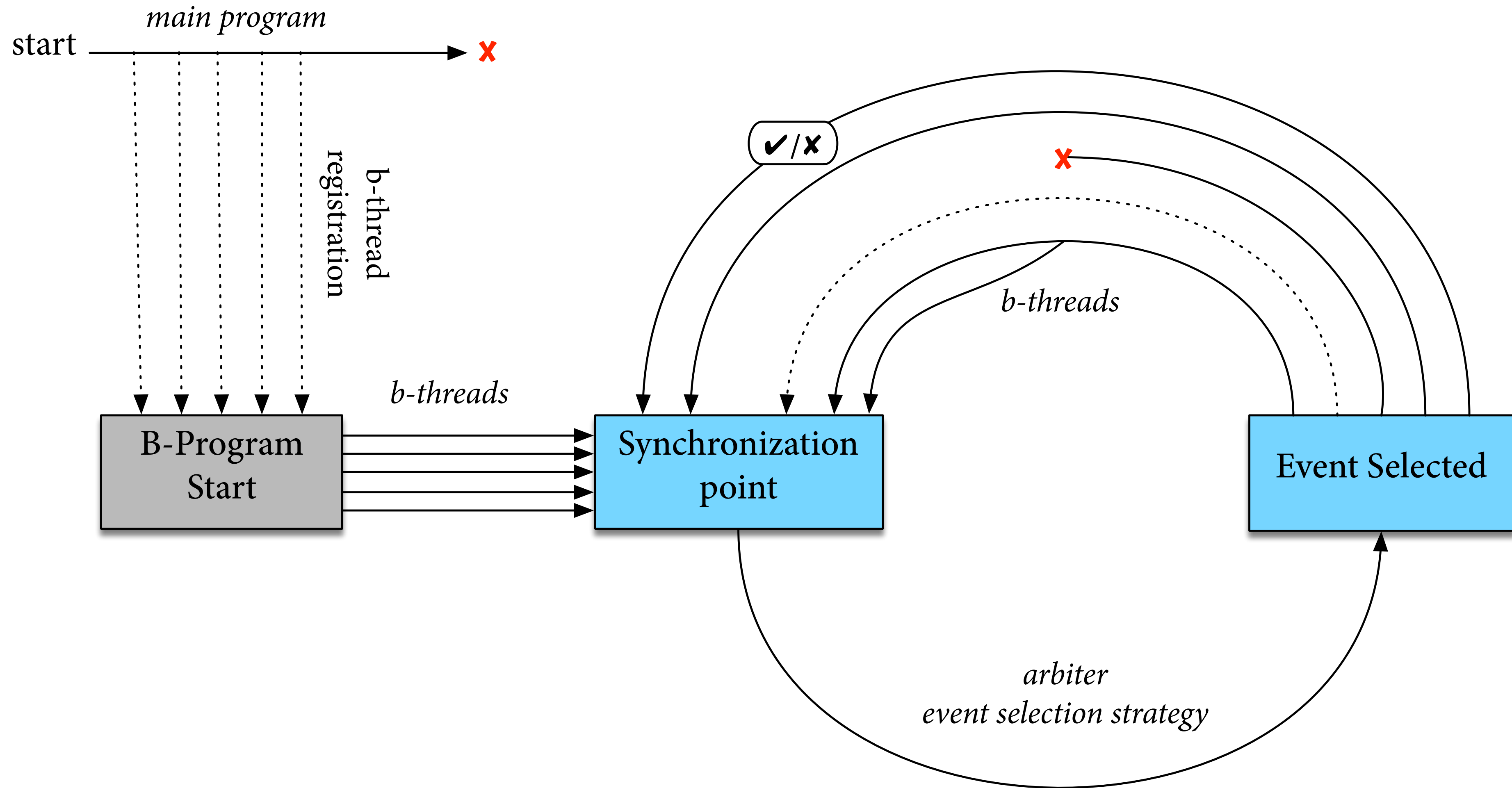
```
bp.registerBThread("bt-2", function(){  
  bp.sync({request:bp.Event("world")});  
});
```

Using BPjs, <https://github.com/bThink-BGU/BPjs>

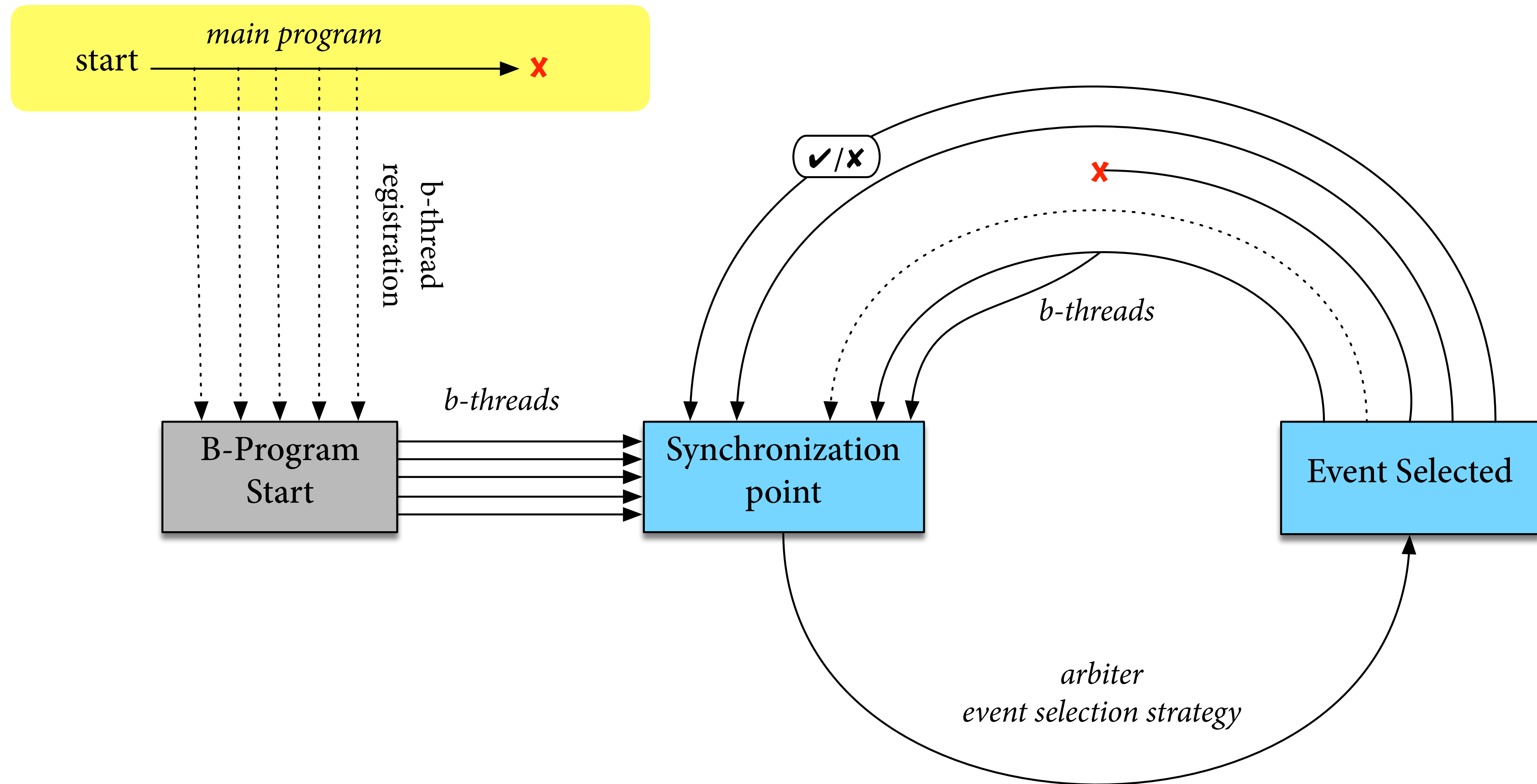


Demo

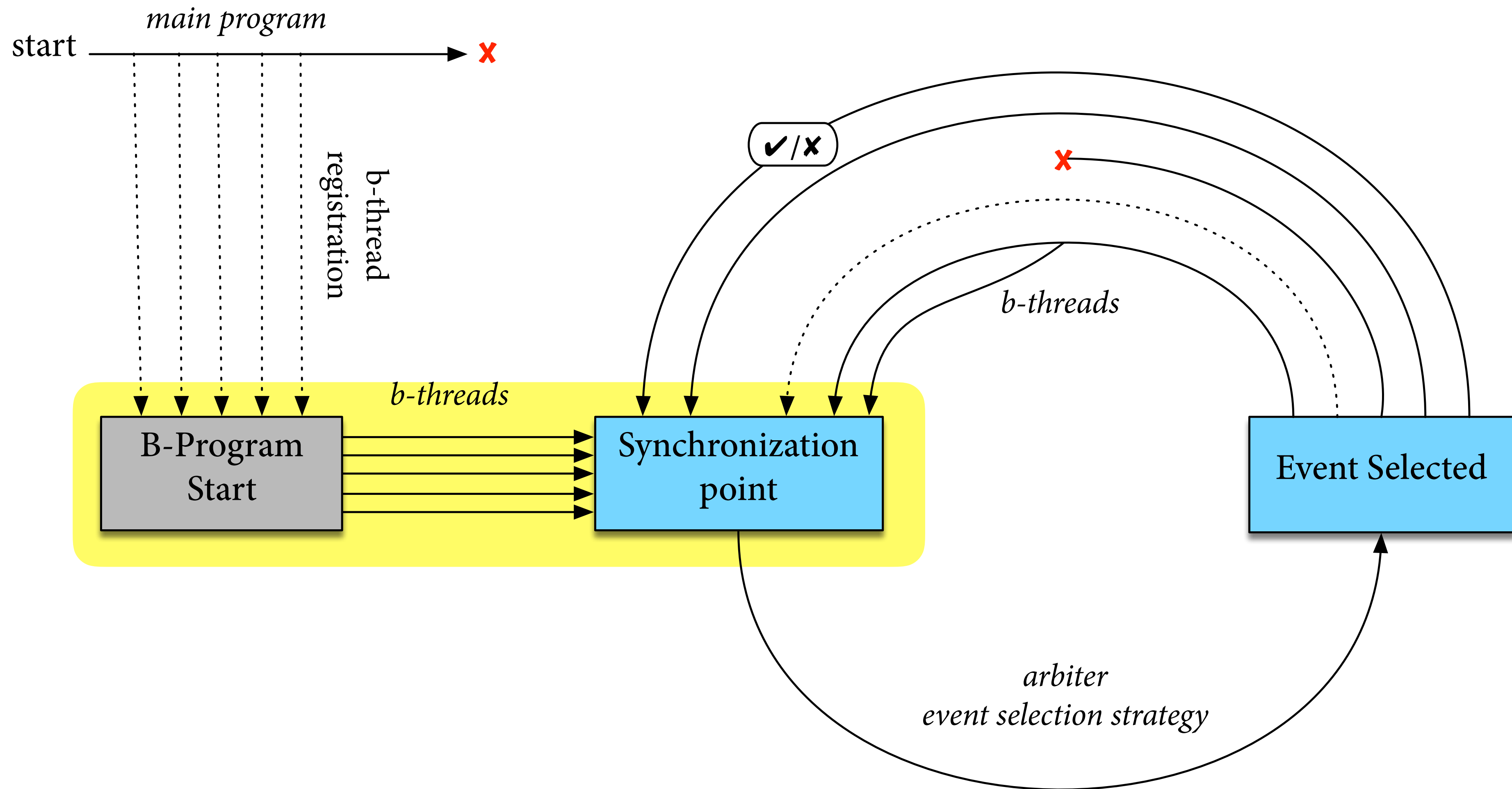
B-Program Life Cycle



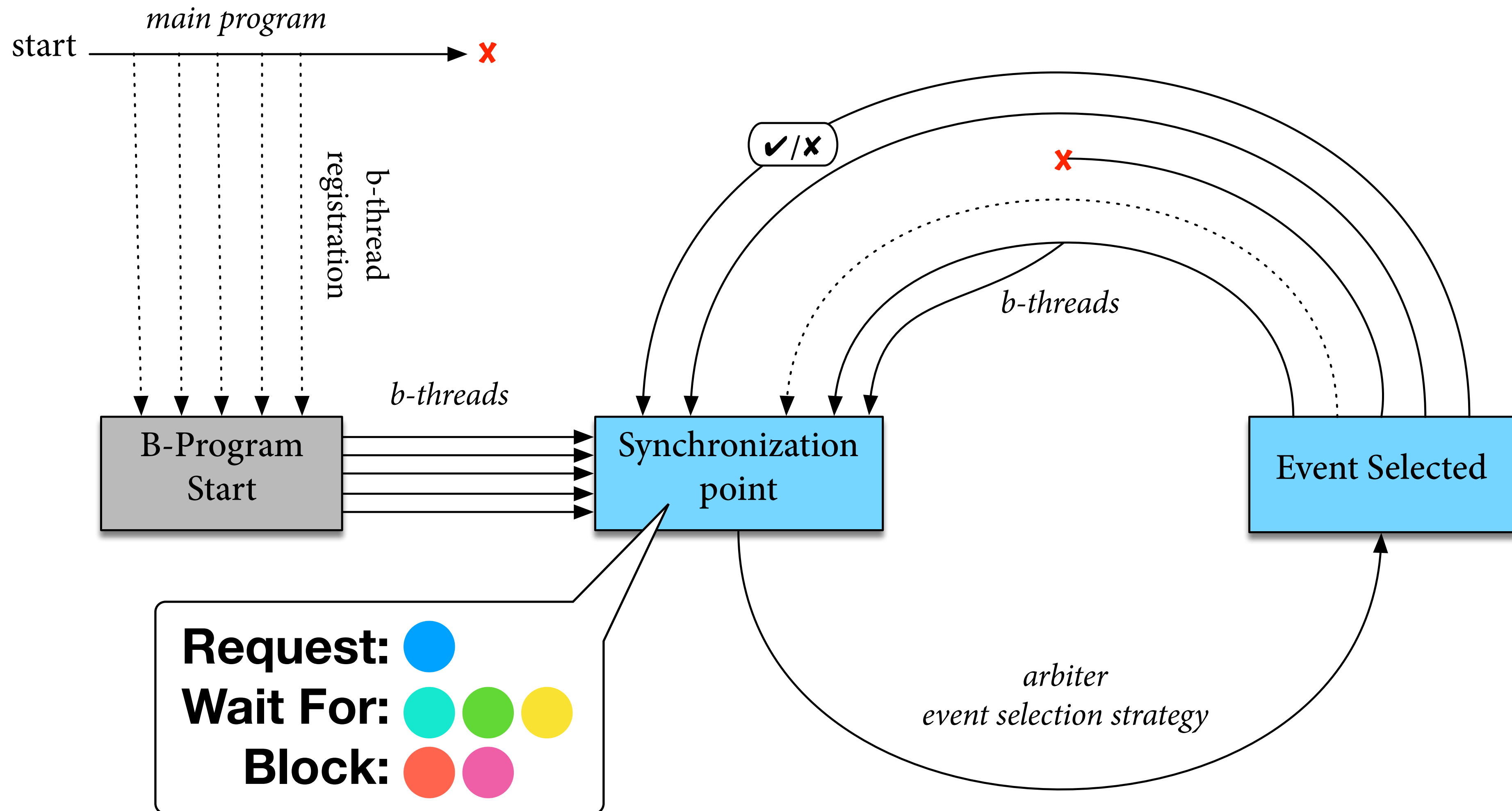
B-Program Life Cycle



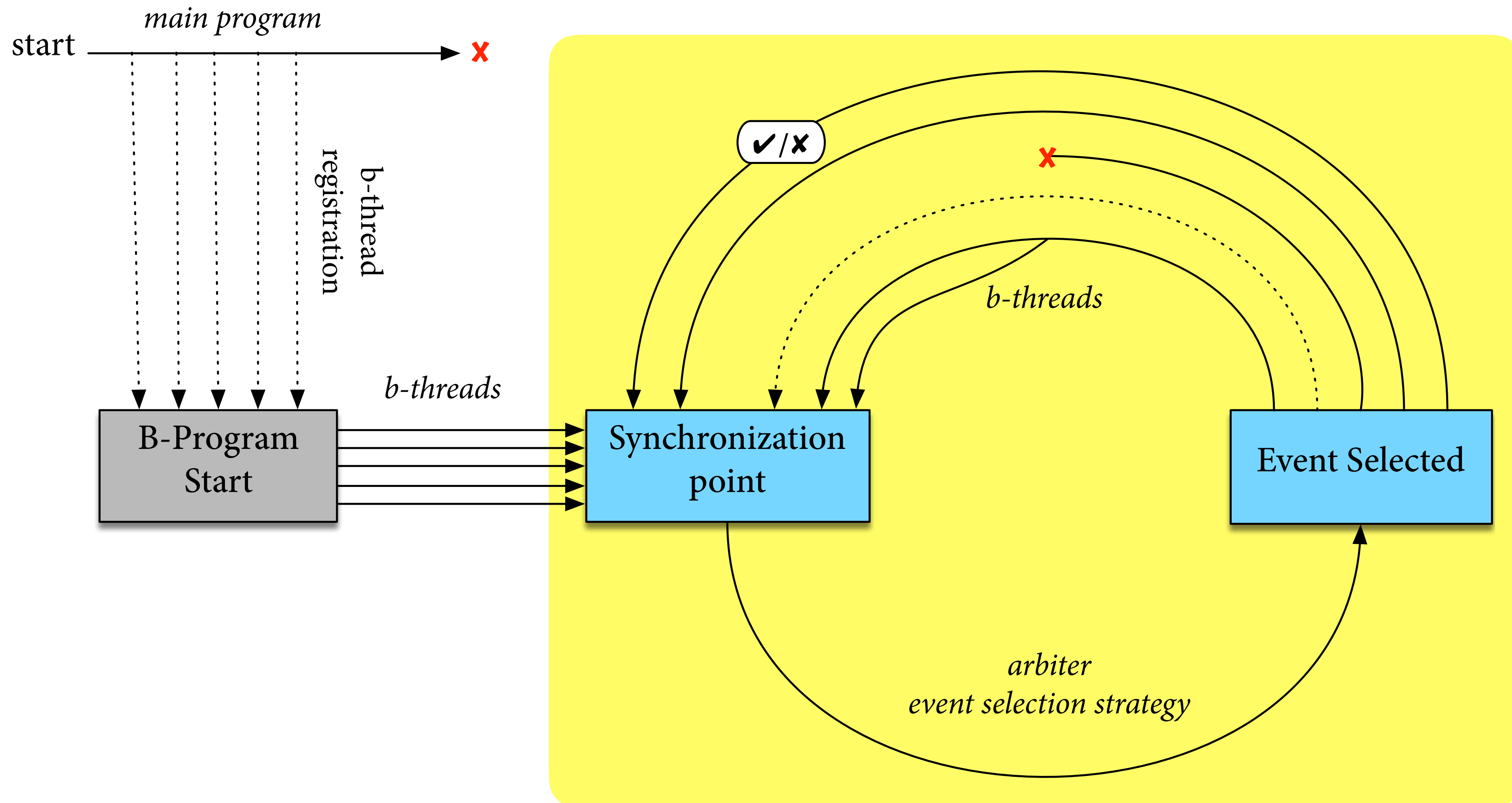
B-Program Life Cycle

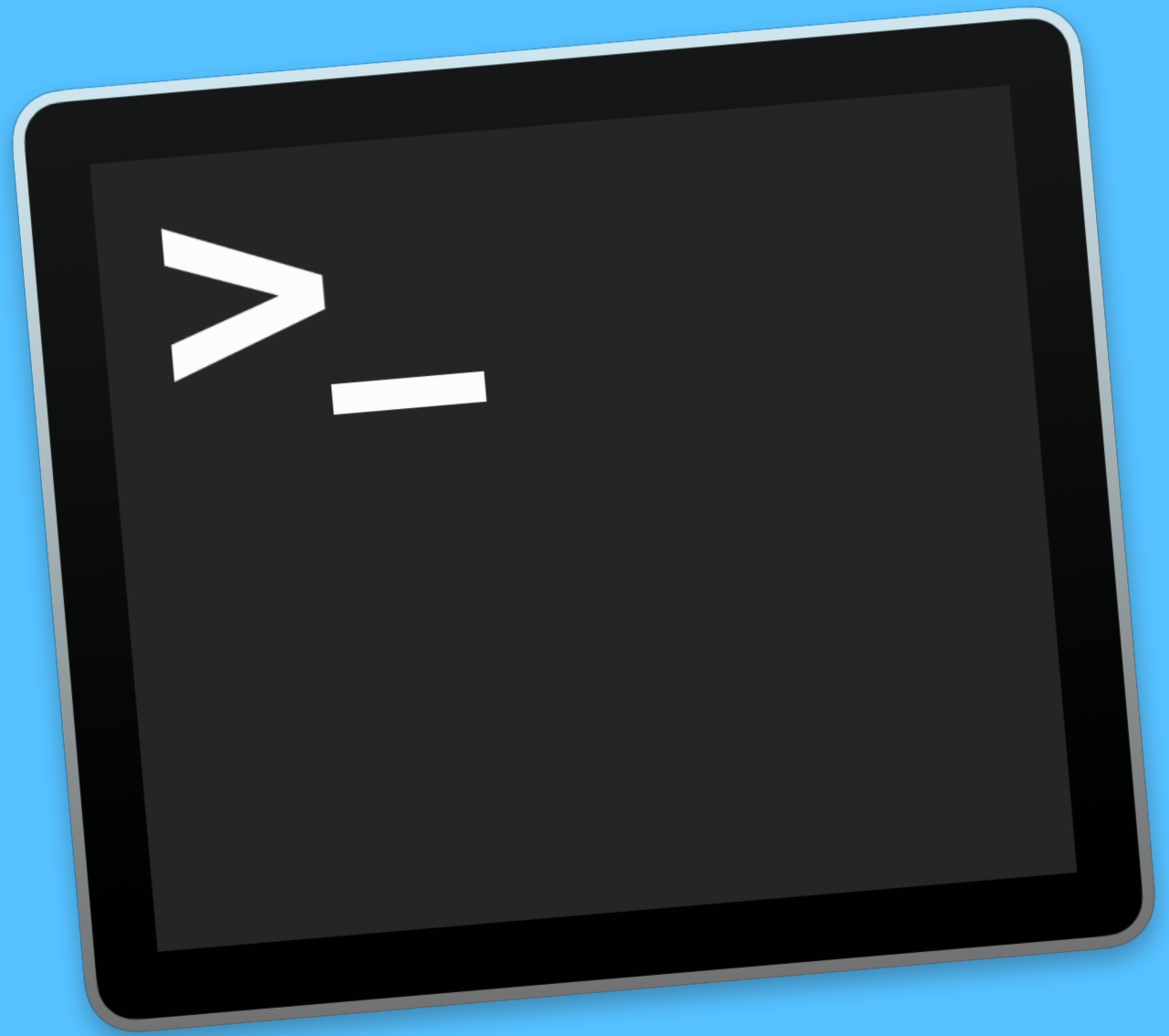


B-Program Life Cycle



B-Program Life Cycle





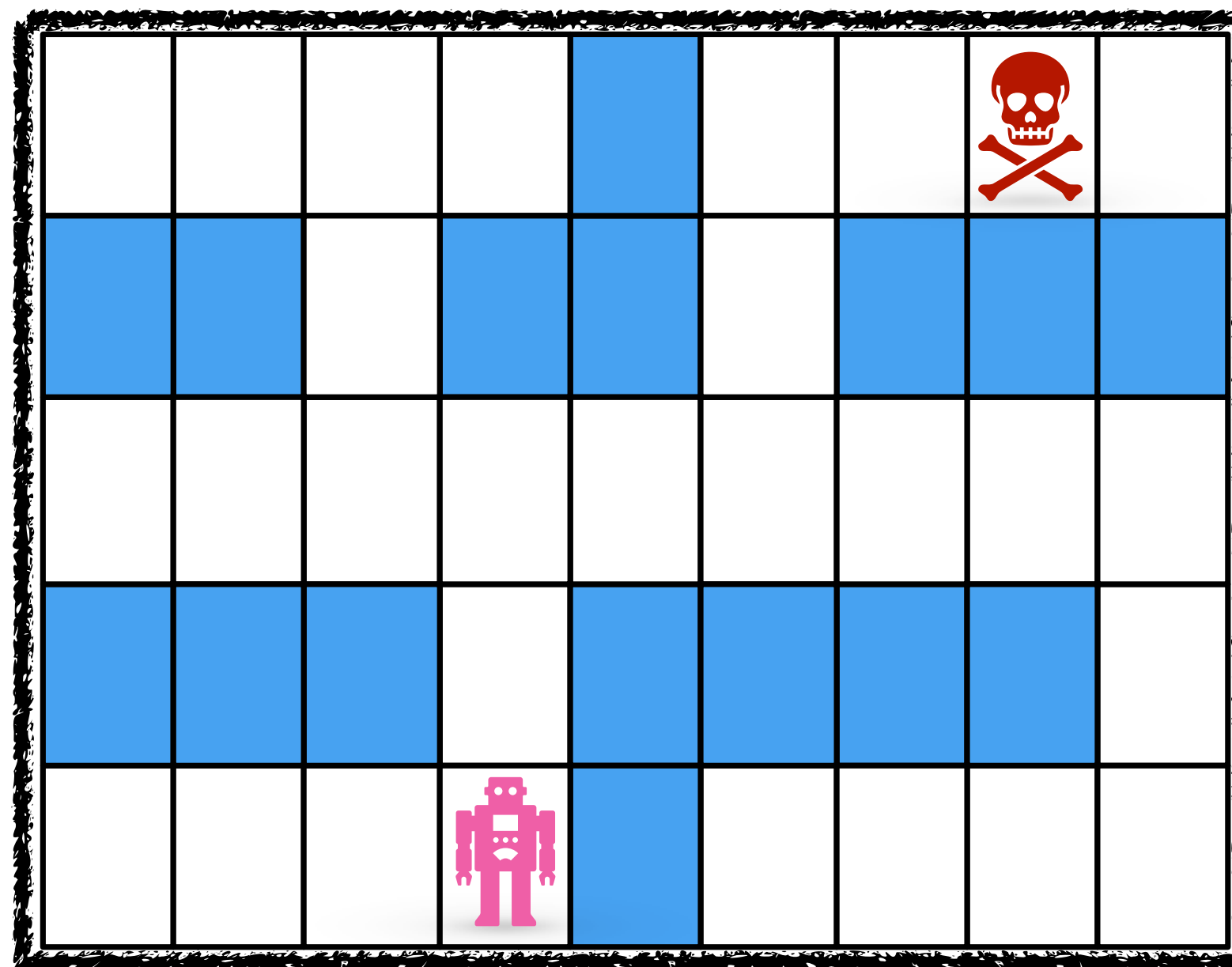
Demo

Hello World Takeaways

- Additivity / Composability
- Modularity
- Asynchronous-first approach

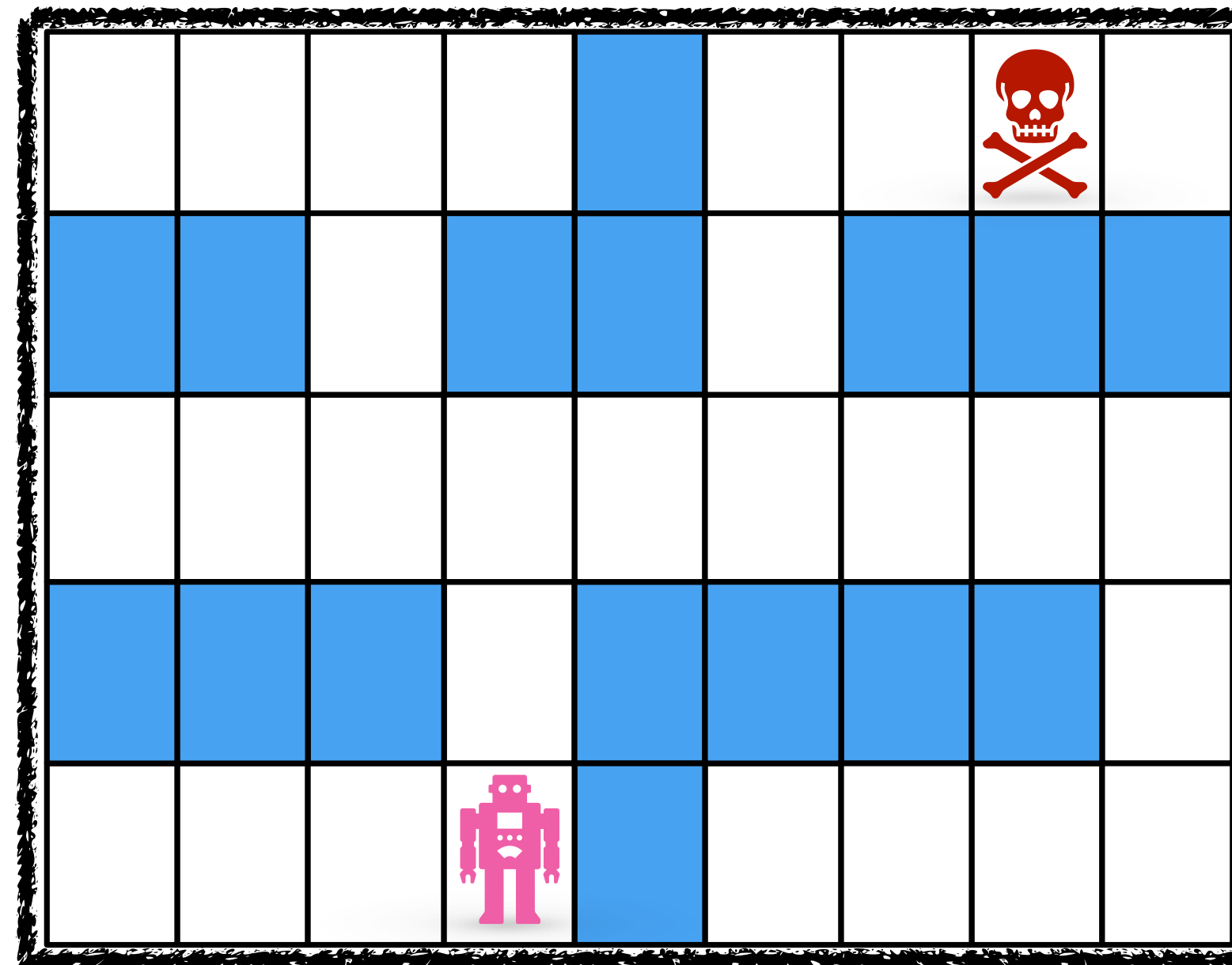
House Model

Simulate a robot moving inside a house



House Model

Simulate a robot moving inside a house



House Model

Simulate a robot moving inside a house

```
var house = [ "          #          t          " ,  
              " # #          # #          # # # " ,  
              "                               " ,  
              " # # #          # # # # " ,  
              "                               s #          " ] ;
```

Parsing into B-Threads: Idea

Parsing:

Iterate over characters in the house description string array.

Generate b-threads for each cell based on the character.

```
[ "      # t " ,  
  "##  ##  ###" ,  
  "      " ,  
  "###  ####" ,  
  "      s#  " ] ;
```

Parsing into B-Threads - Code

```
function parseMaze(mazeLines) {  
    for ( var row=0; row<mazeLines.length; row++ ) {  
        for ( var col=0; col<mazeLines[row].length; col++ ) {  
            var currentPixel = mazeLines[row].substring(col,col+1);  
            if ( currentPixel=== "  " ||  
                currentPixel=== "t" ||  
                currentPixel=== "s" ) {  
                addSpaceCell(col, row);  
                if ( currentPixel=== "t" ) {  
                    addTargetCell(col, row);  
                } else if ( currentPixel=== "s" ) {  
                    addStartCell(col, row);  
                }  
            }  
        }  
    }  
}
```

Parsing into B-Threads - Code

```
function parseMaze(mazeLines) {  
    for ( var row=0; row<mazeLines.length; row++ ) {  
        for ( var col=0; col<mazeLines[row].length; col++ ) {  
            var currentPixel = mazeLines[row].substring(col,col+1);  
            if ( currentPixel=== "  " ||  
                currentPixel=== "t" ||  
                currentPixel=== "s" ) {  
                addSpaceCell(col, row);  
                if ( currentPixel=== "t" ) {  
                    addTargetCell(col, row);  
                } else if ( currentPixel=== "s" ) {  
                    addStartCell(col, row);  
                }  
            }  
        }  
    }  
}
```

Parsing into B-Threads - Code

```
function parseMaze(mazeLines) {  
    for ( var row=0; row<mazeLines.length; row++ ) {  
        for ( var col=0; col<mazeLines[row].length; col++ ) {  
            var currentPixel = mazeLines[row].substring(col,col+1);  
            if ( currentPixel=== "  " ||  
                currentPixel=== "t" ||  
                currentPixel=== "s" ) {  
                addSpaceCell(col, row);  
                if ( currentPixel=== "t" ) {  
                    addTargetCell(col, row);  
                } else if ( currentPixel=== "s" ) {  
                    addStartCell(col, row);  
                }  
            }  
        }  
    }  
}
```

Parsing into B-Threads - Code

```
function parseMaze(mazeLines) {
  for ( var row=0; row<mazeLines.length; row++ ) {
    for ( var col=0; col<mazeLines[row].length; col++ ) {
      var currentPixel = mazeLines[row].substring(col,col+1);
      if ( currentPixel===" " ||
           currentPixel==="t" ||
           currentPixel==="s" ) {
        addSpaceCell(col, row);
        if ( currentPixel==="t" ) {
          addTargetCell(col, row);
        } else if ( currentPixel==="s" ) {
          addStartCell(col, row);
        }
      }
    }
  }
}
```

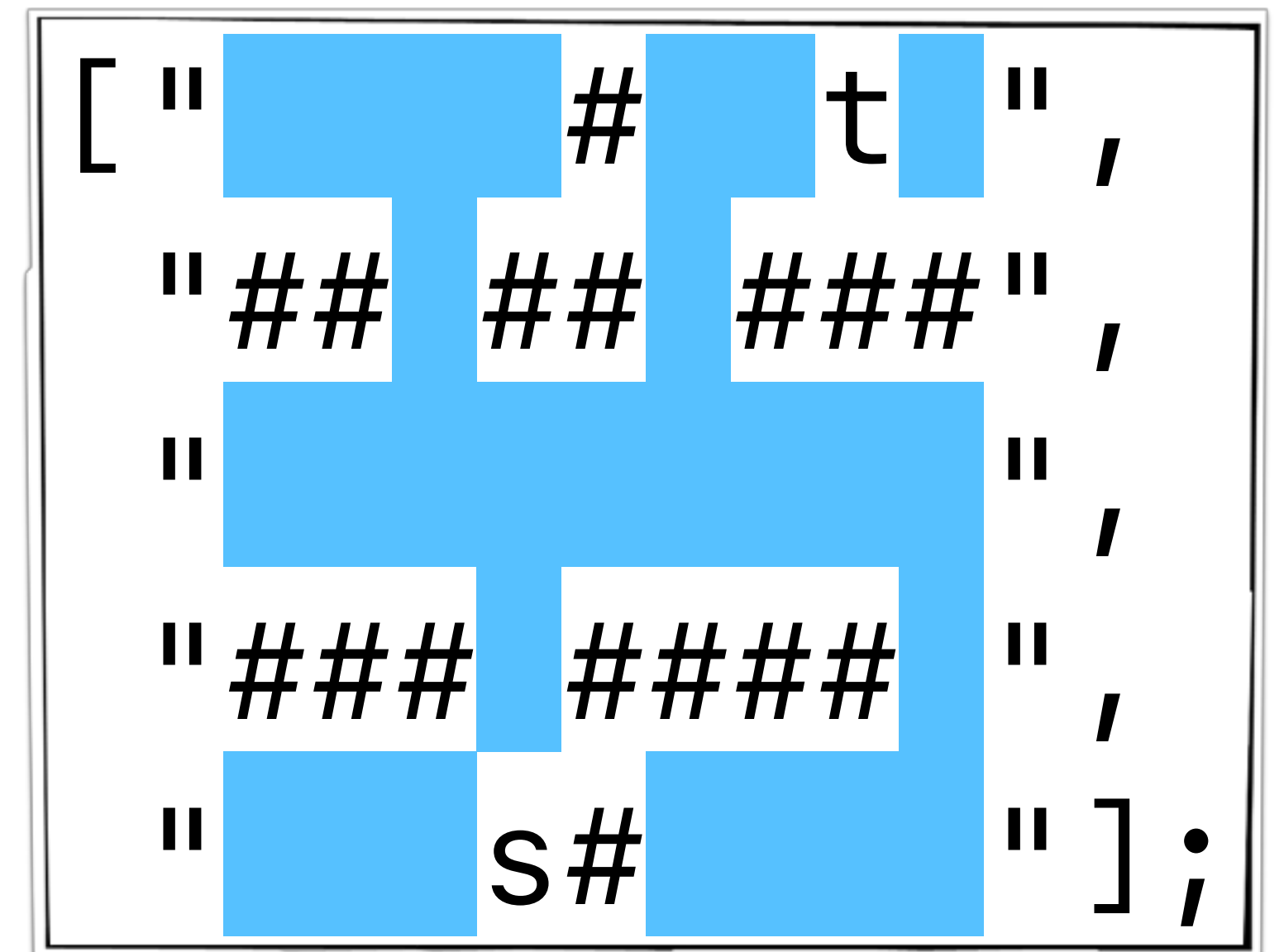
Parsing into B-Threads - Code

```
function parseMaze(mazeLines) {  
    for ( var row=0; row<mazeLines.length; row++ ) {  
        for ( var col=0; col<mazeLines[row].length; col++ ) {  
            var currentPixel = mazeLines[row].substring(col,col+1);  
            if ( currentPixel===" " ||  
                currentPixel==="t" ||  
                currentPixel==="s" ) {  
                addSpaceCell(col, row);  
                if ( currentPixel==="t" ) {  
                    addTargetCell(col, row);  
                } else if ( currentPixel==="s" ) {  
                    addStartCell(col, row);  
                }  
            }  
        }  
    }  
}
```

"Different types of cells have different behaviors"

Behaviors for a Space Cell

```
function addSpaceCell( col, row ) {  
  bp.registerBThread("cell(c:"+col+" r:"+row+")",  
    function() {  
      while ( true ) {  
        bp.sync({waitFor:adjacentCellEntries(col, row)});  
        bp.sync({  
          request: enterEvent(col, row),  
          waitFor: anyEntrance  
        });  
      }  
    }  
  );  
}
```



Behaviors for a Space Cell

```
function addSpaceCell( col, row ) {  
  bp.registerBThread("cell(c:"+col+" r:"+row+")",  
    function() {  
      while ( true ) {  
        bp.sync({waitFor:adjacentCellEntries(col, row)});  
        bp.sync({  
          request: enterEvent(col, row),  
          waitFor: anyEntrance  
        });  
      }  
    }  
  );  
}
```

```
[ " ■ ■ ■ # t " ,  
  " ## ## ### " ,  
  " ■ ■ ■ ■ ■ " ,  
  " ### #### " ,  
  " ■ ■ s# ■ " ] ;
```

Behaviors for a Target Cell

```
function addSpaceCell( col, row ) {...}
```

+

```
function addTargetCell(col, row) {  
  bp.registerBThread("Target(c:"+col+" r:"+row+")", function(){  
    bp.sync({  
      waitFor: enterEvent(col, row)  
    });  
  
    bp.sync({  
      request: TARGET_FOUND,  
      block: bp.allExcept( TARGET_FOUND )  
    });  
  });  
}
```

```
[ "      # t  ",  
  "###  ##  ###",  
  "          ",  
  "###  ####",  
  "      s#  " ];
```

Behaviors for a Target Cell

```
function addSpaceCell( col, row ) {...}
```

+

```
function addTargetCell(col, row) {  
  bp.registerBThread("Target(c:"+col+" r:"+row+")", function(){  
    bp.sync({  
      waitFor: enterEvent(col, row)  
    });  
  
    bp.sync({  
      request: TARGET_FOUND,  
      block: bp.allExcept( TARGET_FOUND )  
    });  
  });  
}
```

```
[ "      # t  ",  
  "###  ##  ###",  
  "          ",  
  "###  ####",  
  "      s#  " ];
```

Behaviors for a Target Cell

```
function addSpaceCell( col, row ) {...}
```

+

```
function addTargetCell(col, row) {  
  bp.registerBThread("Target(c:"+col+" r:"+row+" )", function(){  
    bp.sync({  
      waitFor: enterEvent(col, row)  
    });  
  
    bp.sync({  
      request: TARGET_FOUND,  
      block: bp.allExcept( TARGET_FOUND )  
    });  
  });  
}
```

```
[ "      # t  ",  
  "###  ##  ###",  
  "      " ,  
  "###  ####",  
  "      s#  " ];
```

Behaviors for a Target Cell

```
function addSpaceCell( col, row ) {...}
```

+

```
function addTargetCell(col, row) {  
  bp.registerBThread("Target(c:"+col+" r:"+row+")", function(){  
    bp.sync({  
      waitFor: enterEvent(col, row)  
    });  
  
    bp.sync({  
      request: TARGET_FOUND,  
      block: bp.allExcept( TARGET_FOUND )  
    });  
  });  
}
```

```
[ "      # t  ",  
  "###  ##  ###",  
  "                ",  
  "###  ####  ",  
  "      s#      " ];
```

Behaviors for a Start Cell

```
function addSpaceCell( col, row ) {...}
```

+

```
function addStartCell( col, row ) {  
  bp.registerBThread( "starter(c:"+col+" r:"+row+" )", function() {  
    bp.sync( {  
      request: enterEvent( col, row )  
    } );  
  } );  
}
```

```
[ "      # t ",  
  "##  ##  ###",  
  "      " ,  
  "###  #### " ,  
  "      s# " ] ;
```


Behaviors for a Start Cell

```
function addSpaceCell( col, row ) {...}
```

+

```
function addStartCell( col, row ) {  
    bp.registerBThread( "starter(c:"+col+" r:"+row+" )", function() {  
        bp.sync( {  
            request: enterEvent( col, row )  
        } );  
    } );  
}
```

```
[ "      # t ",  
  "##  ##  ###",  
  "                ",  
  "###  ####  ",  
  "      s#      " ];
```

Behaviors for a Start Cell

```
function addSpaceCell( col, row ) {...}
```

+

```
function addStartCell( col, row ) {  
  bp.registerBThread("starter(c:"+col+" r:"+row+")", function() {  
    bp.sync({  
      request:enterEvent( col, row )  
    });  
  });  
}
```

```
[ "      # t ",  
  "##  ##  ###",  
  "      ",  
  "###  #### ",  
  "      s#  " ];
```

Behaviors for a Start Cell

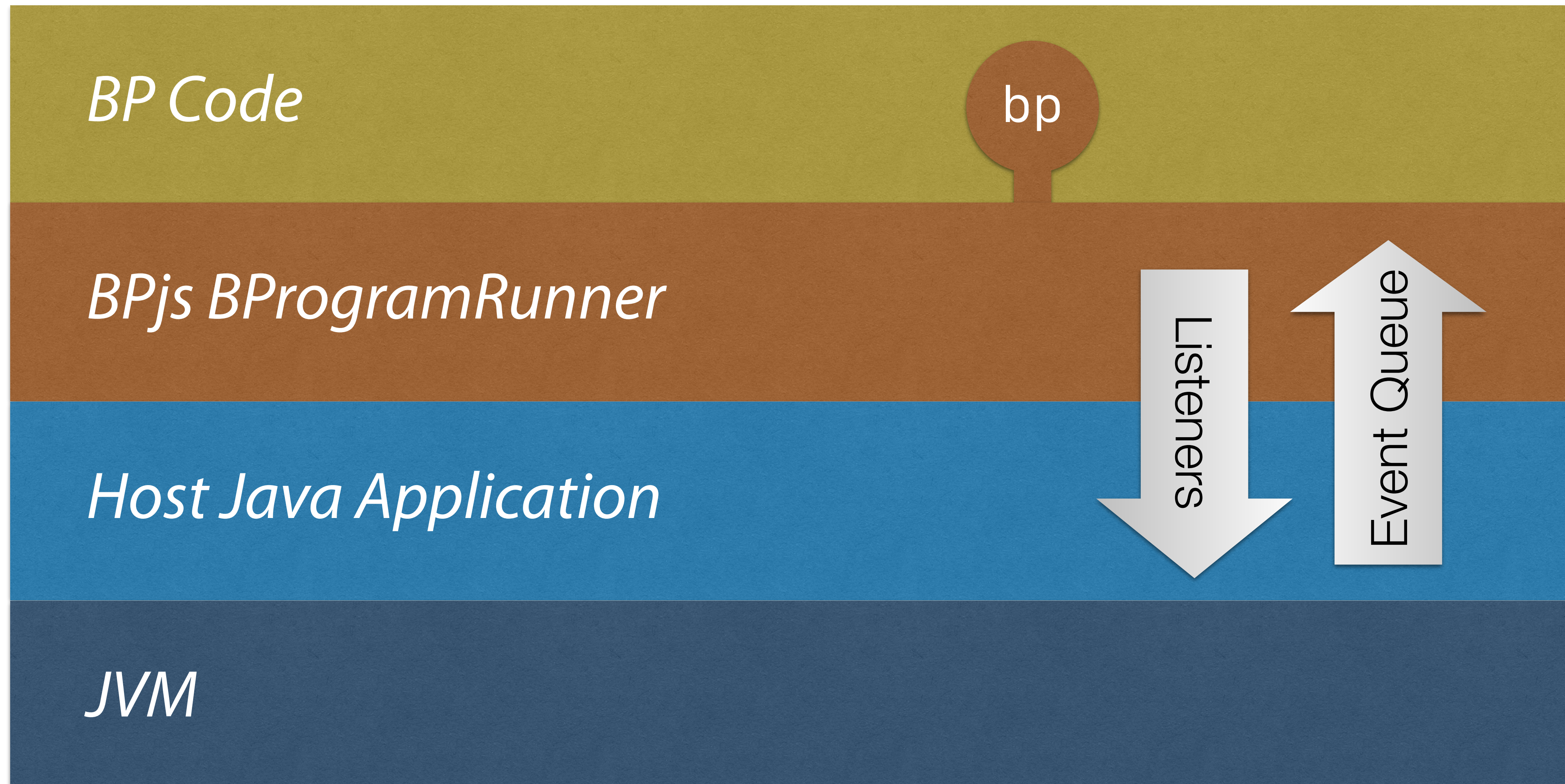
```
function addSpaceCell( col, row ) {...}
```

+

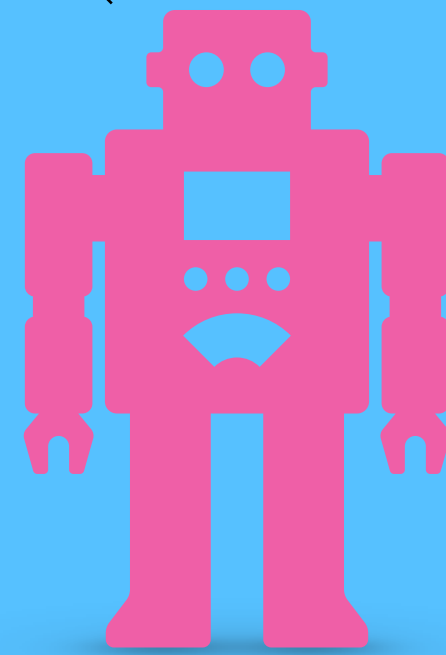
```
function addStartCell( col, row ) {  
  bp.registerBThread( "starter(c:"+col+" r:"+row+" )", function() {  
    bp.sync( {  
      request: enterEvent( col, row )  
    } );  
  } );  
}
```

```
[ "      # t ",  
  "##  ##  ###",  
  "      ",  
  "###  #### ",  
  "      s#  " ];
```

Runtime Stack




Let's Go!



Demo

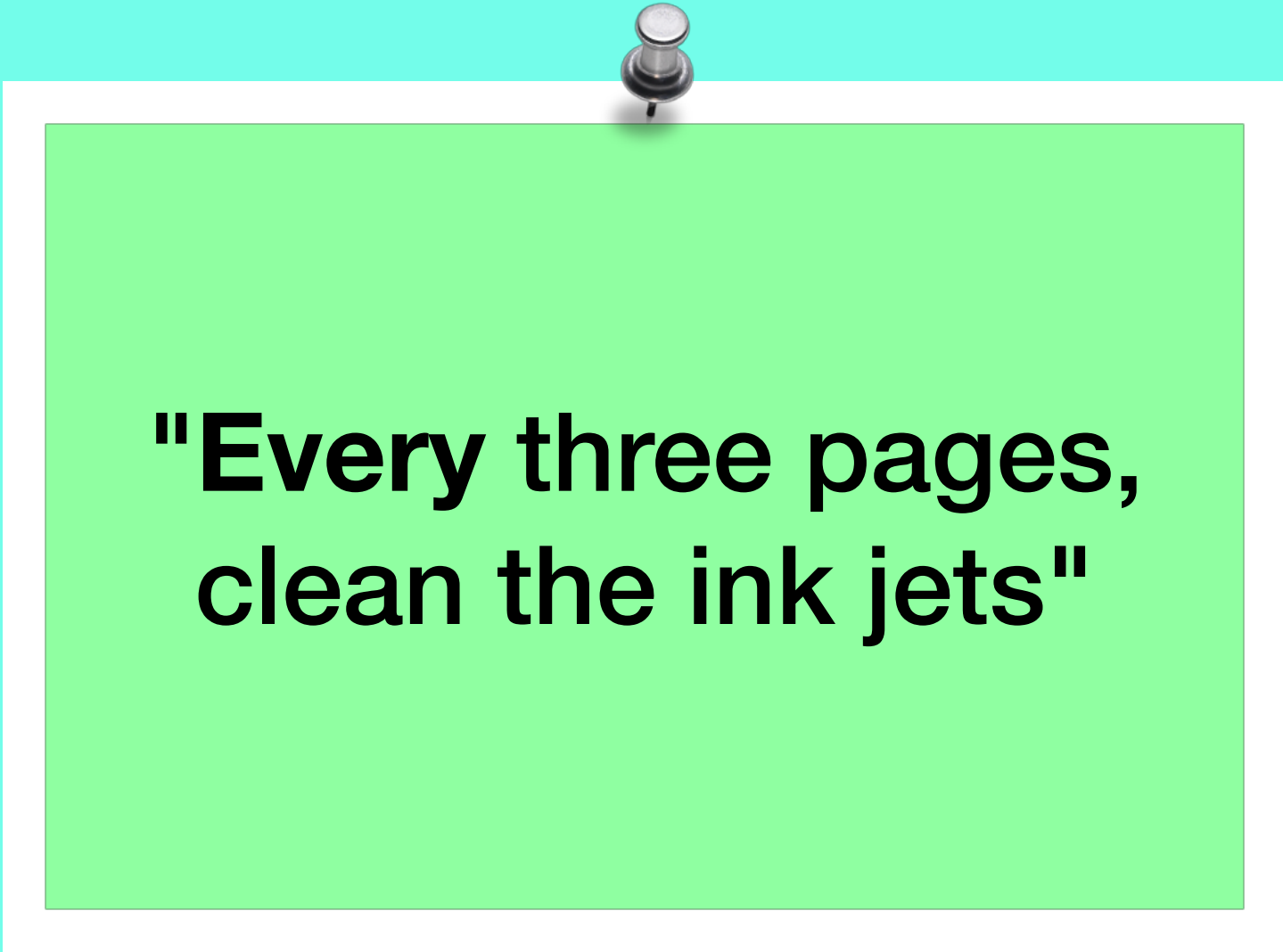


**"After the first three
pages, clean the ink
jets"**




"After the first three
pages, clean the ink
jets"

```
bp.registerBThread("cleaner", function(){  
    bp.sync({waitFor:PrintPage});  
    bp.sync({waitFor:PrintPage});  
    bp.sync({waitFor:PrintPage});  
    bp.sync{  
        request: CleanInkJets,  
        block: PrintPage  
    };  
});
```

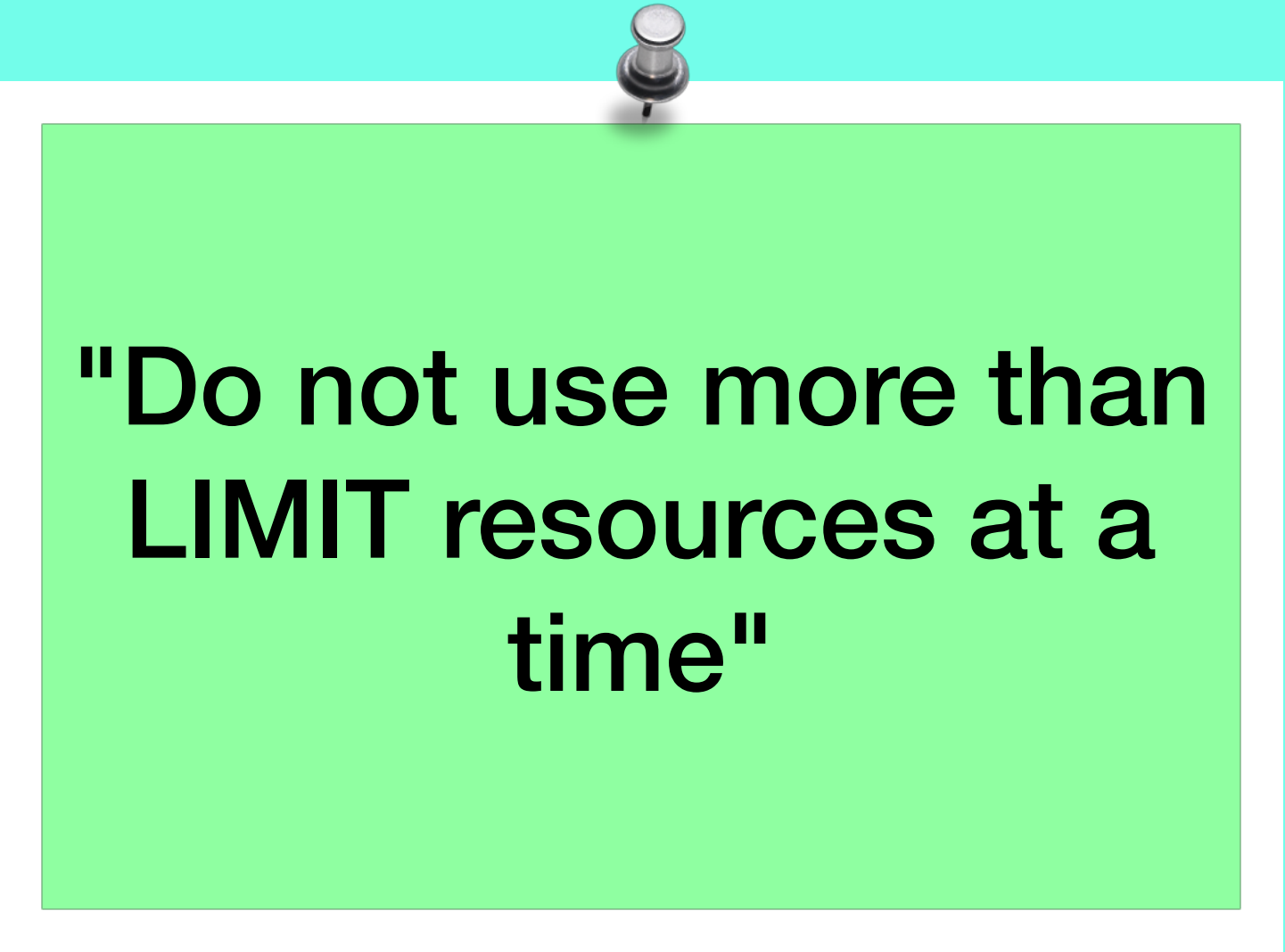


**"Every three pages,
clean the ink jets"**

```
bp.registerBThread("cleaner", function(){
    while ( true ) {
        bp.sync({waitFor:PrintPage});
        bp.sync({waitFor:PrintPage});
        bp.sync({waitFor:PrintPage});
        bp.sync({
            request: CleanInkJets,
            block: PrintPage
        });
    }
});
```

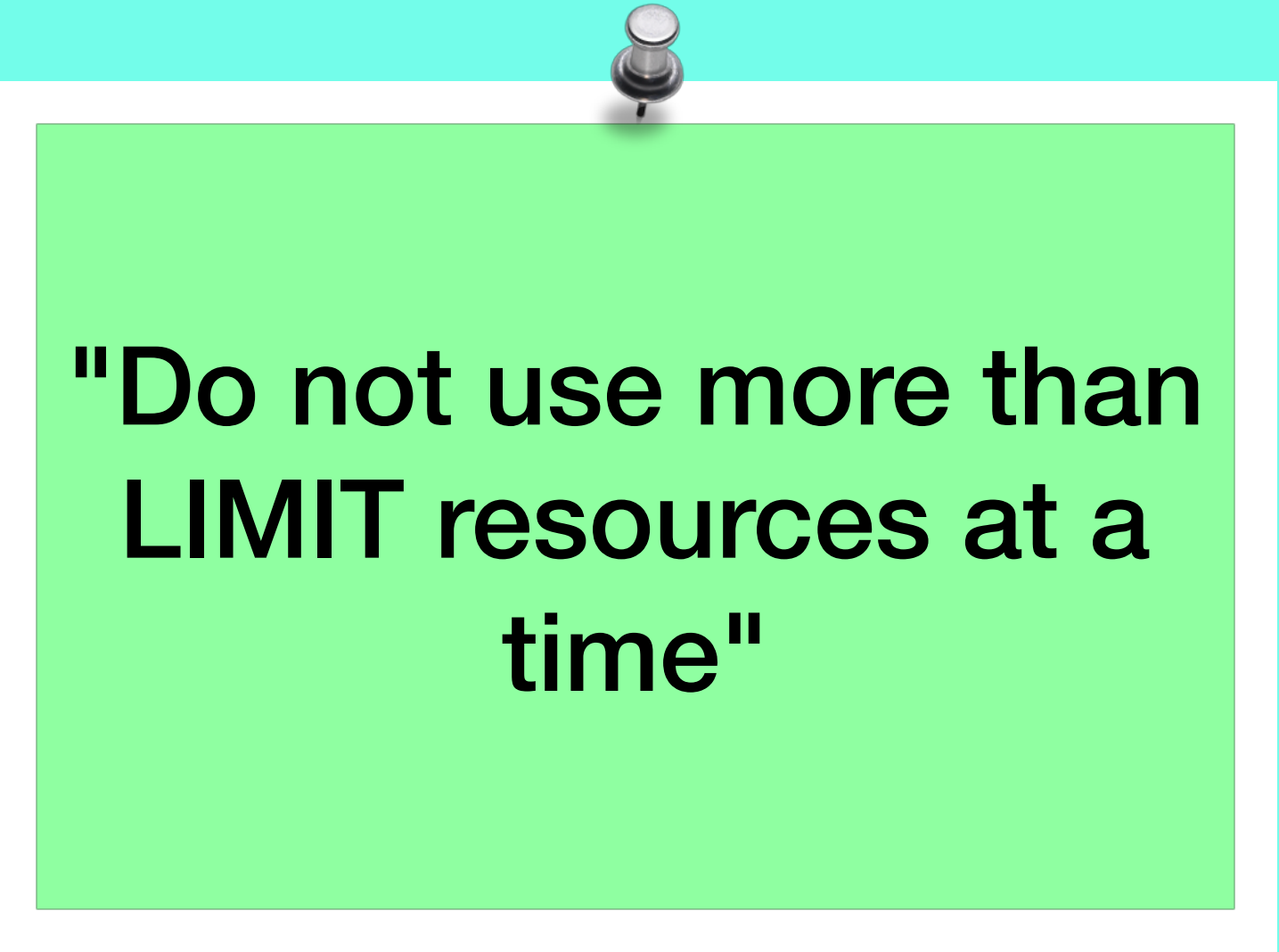



**"Do not use more than
LIMIT resources at a
time"**




"Do not use more than
LIMIT resources at a
time"

```
bp.registerBThread("limitConsumption", function(){
    var count = 0;
    while (true) {
        var evt = bp.sync({waitFor:[AnyUsageStart, AnyUsageEnd]});
        if ( AnyUsageStart.contains(evt) ) {
            count++;
        } else if ( AnyUsageEnd.contains(evt) ) {
            count--;
        }
        if ( count === LIMIT ) {
            bp.log.info("limiting...")
            bp.registerBThread("tempBlock", function(){
                bp.sync({
                    waitFor:AnyUsageEnd,
                    block:AnyUsageStart
                });
            });
        }
    }
});
```

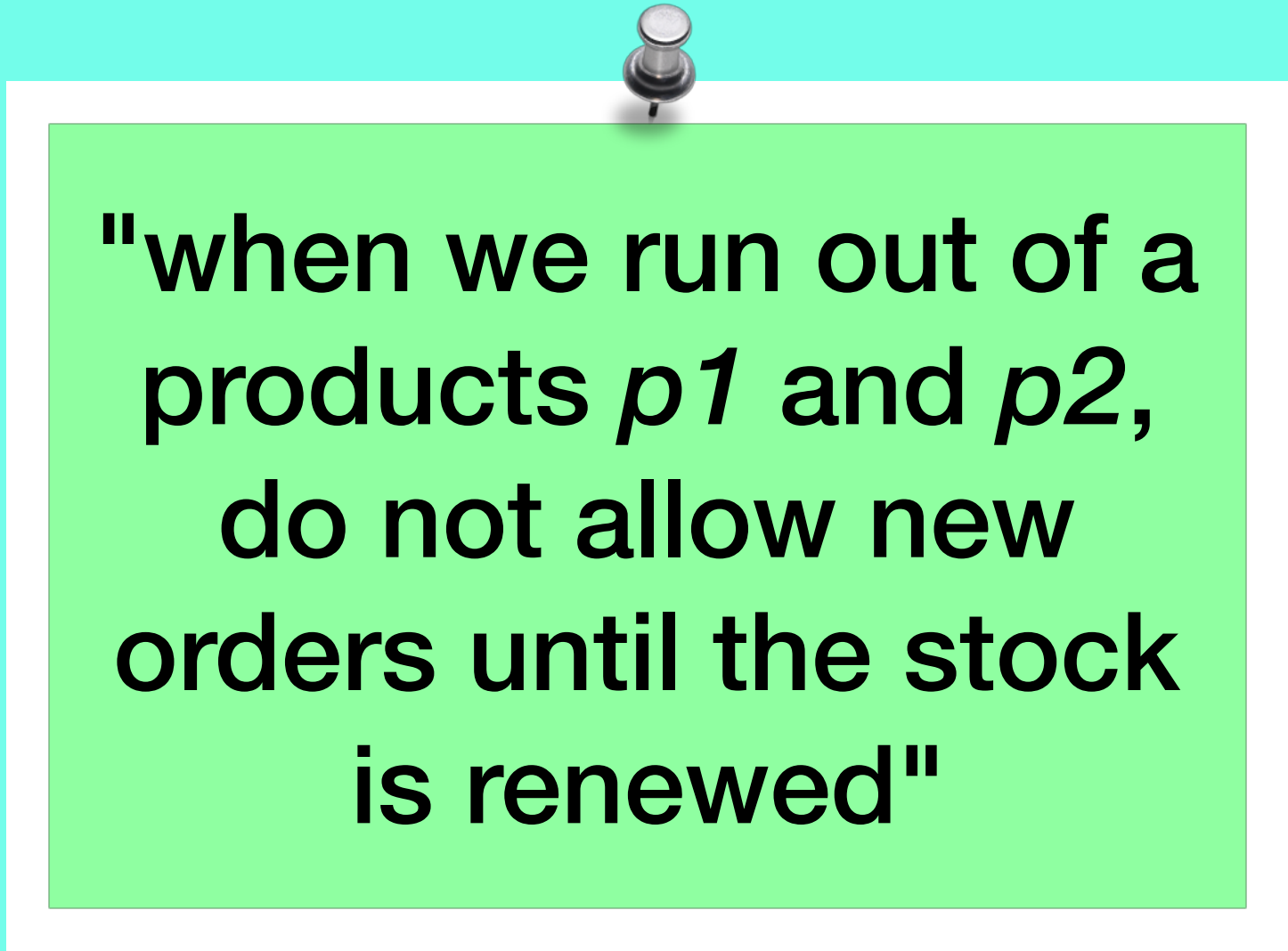


"Do not use more than
LIMIT resources at a
time"

```
bp.registerBThread("limitConsumption", function(){
    var count = 0;
    while (true) {
        var evt = bp.sync({waitFor:[AnyUsageStart, AnyUsageEnd]});
        if ( AnyUsageStart.contains(evt) ) {
            count++;
        } else if ( AnyUsageEnd.contains(evt) ) {
            count--;
        }
        if ( count === LIMIT ) {
            bp.log.info("limiting...")
            bp.registerBThread("tempBlock", function(){
                bp.sync({
                    waitFor:AnyUsageEnd,
                    block:AnyUsageStart
                });
            });
        }
    }
});
```



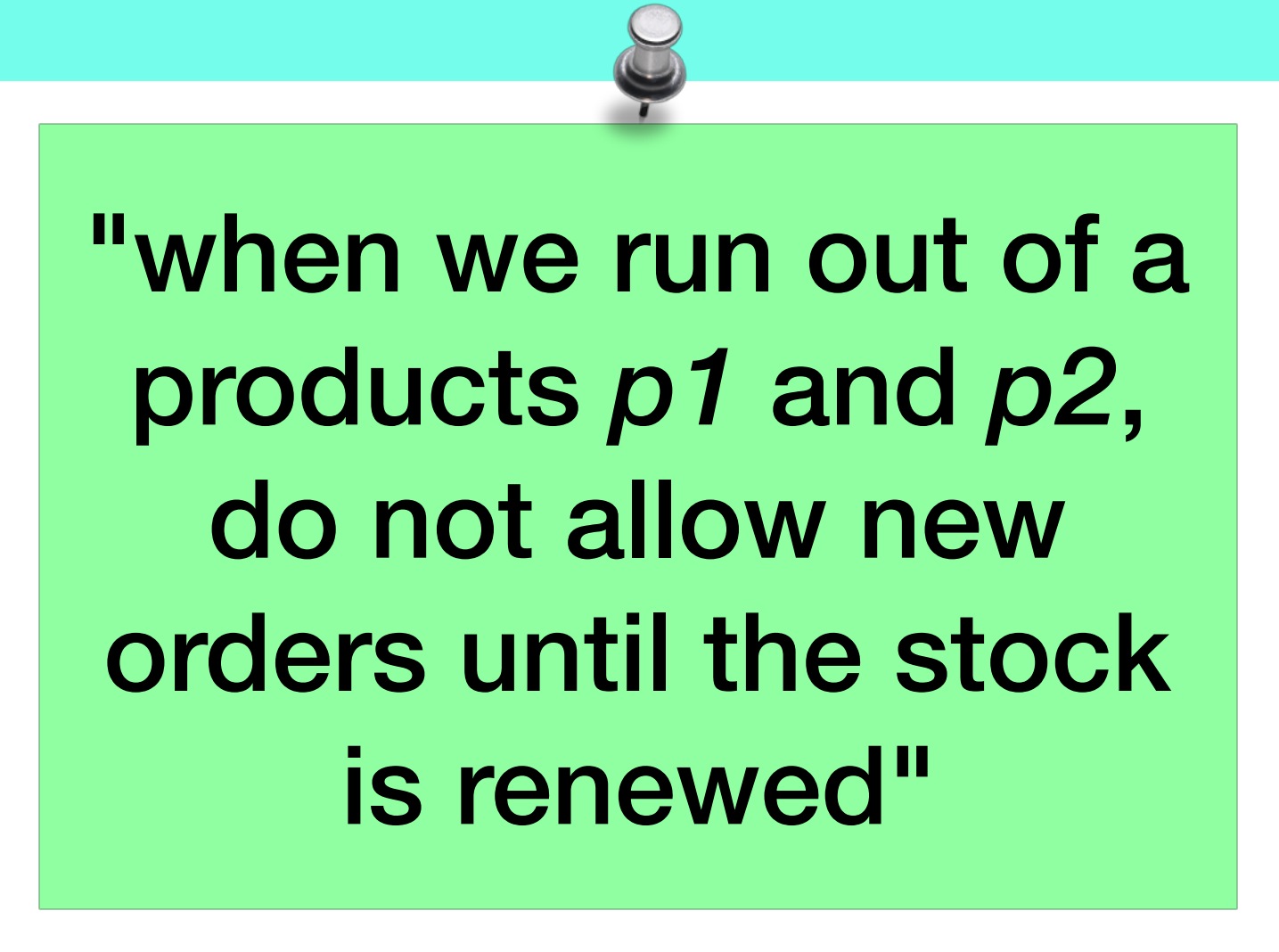
"when we run out of a
products $p1$ and $p2$,
do not allow new
orders until the stock
is renewed"



"when we run out of a products *p1* and *p2*, do not allow new orders until the stock is renewed"

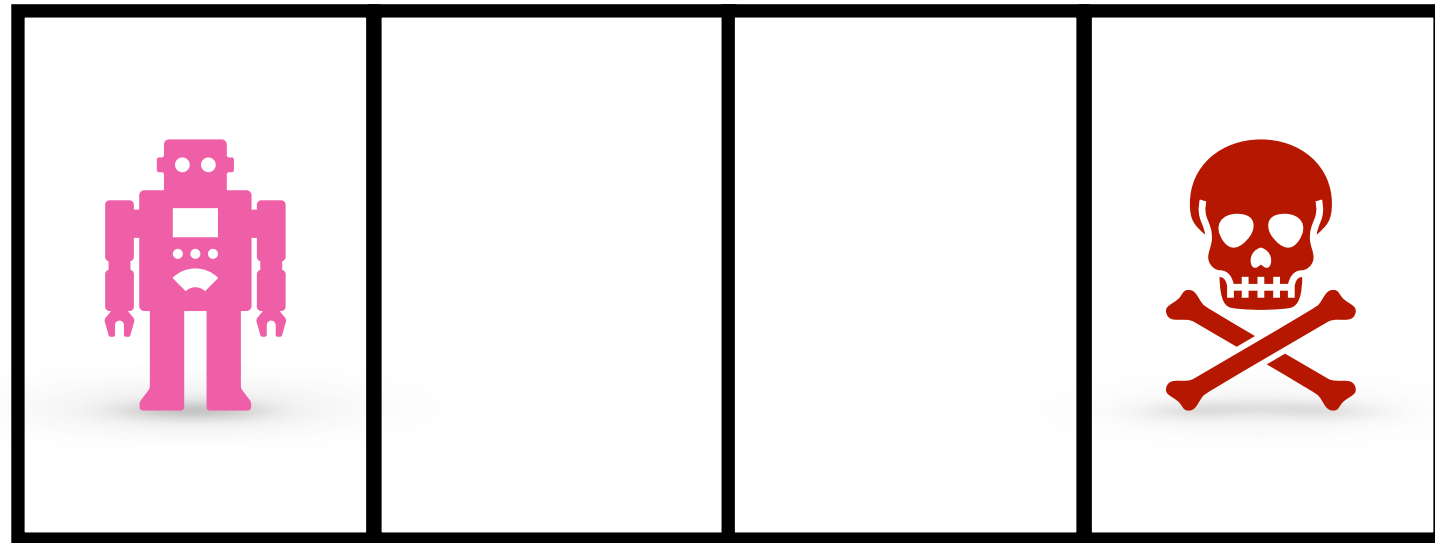
```
function blockOrdersWhenOutOf( p ) {  
  return function(){  
    while(true) {  
      bp.sync({waitFor: OutOfStock(p)});  
      bp.sync{  
        waitFor: RefillDone(p),  
        block: Order(p)  
      });  
    };  
  };  
}  
bp.registerBThread(blockOrdersWhenOutOf(p1));  
bp.registerBThread(blockOrdersWhenOutOf(p2));
```

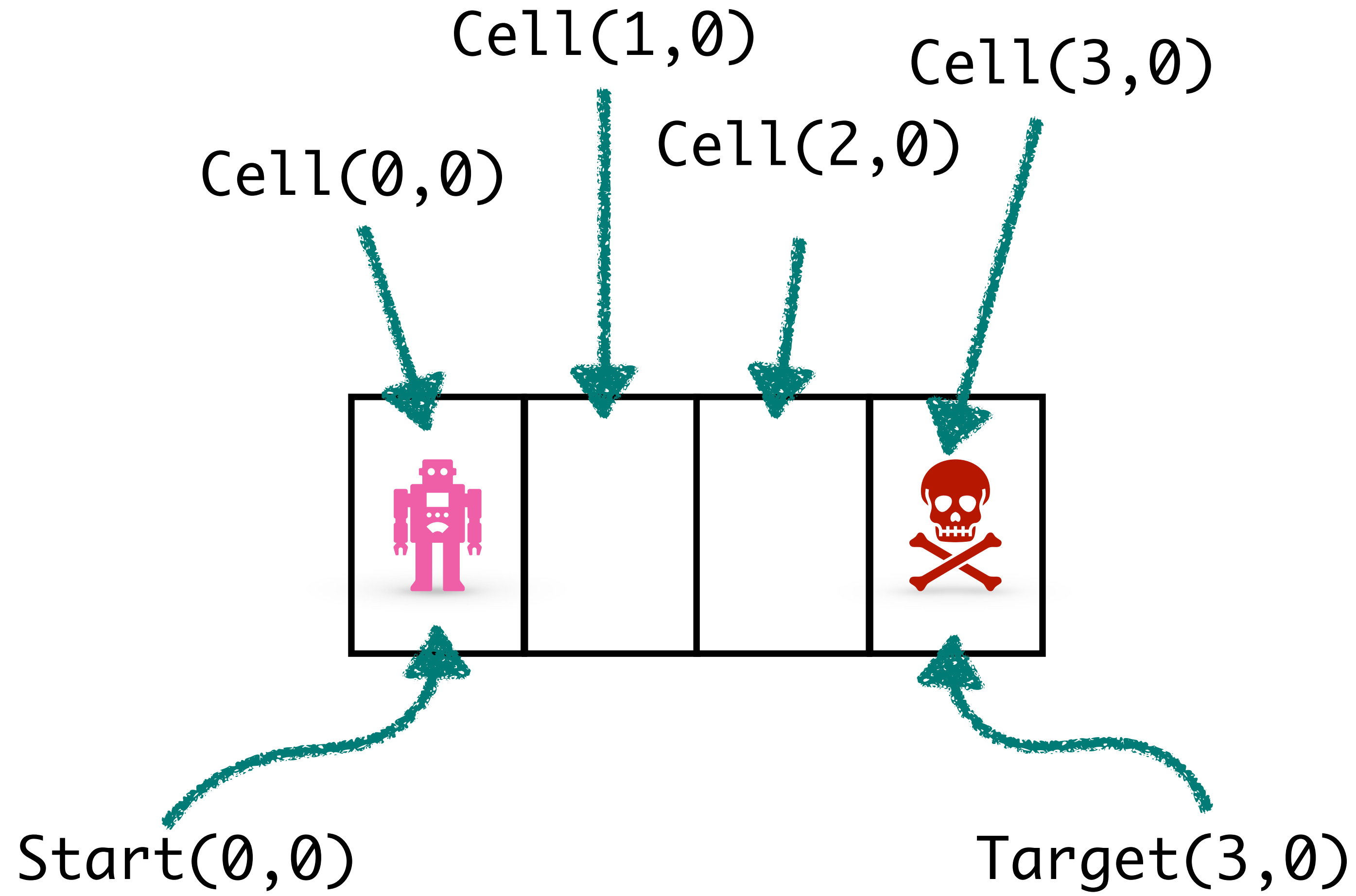
Requirement Alignment

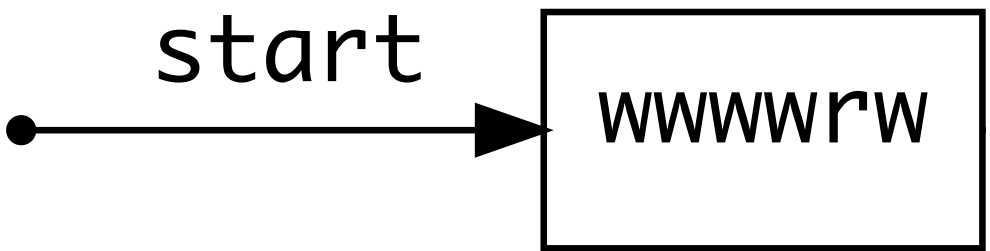
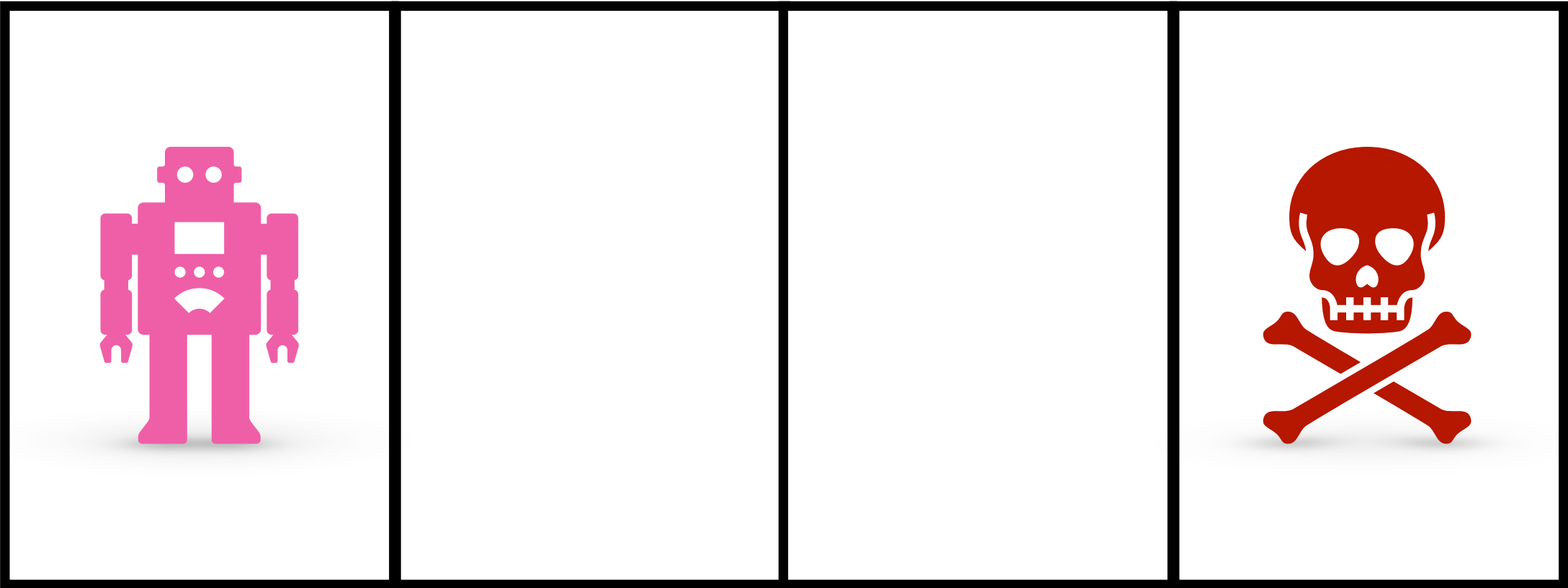
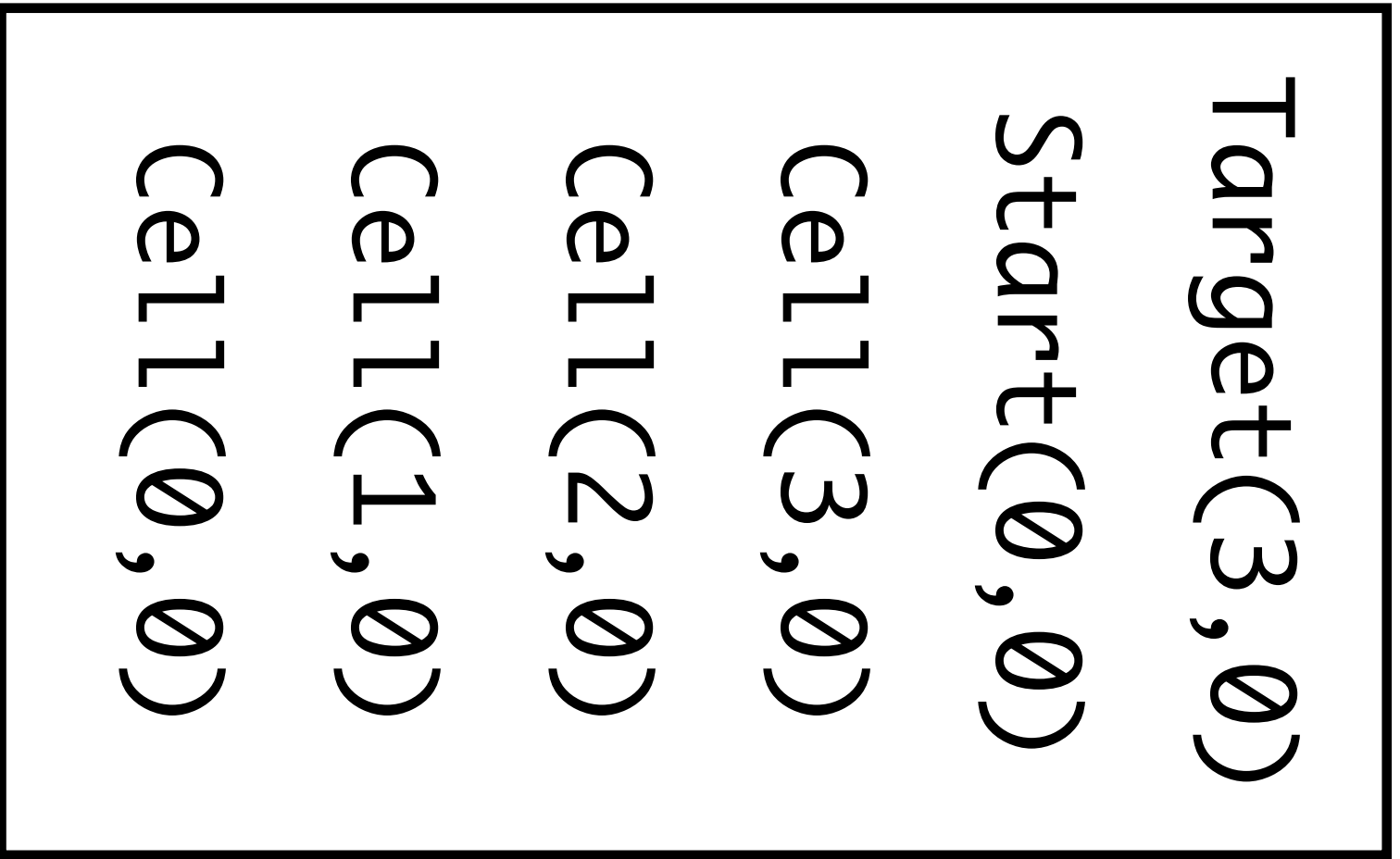


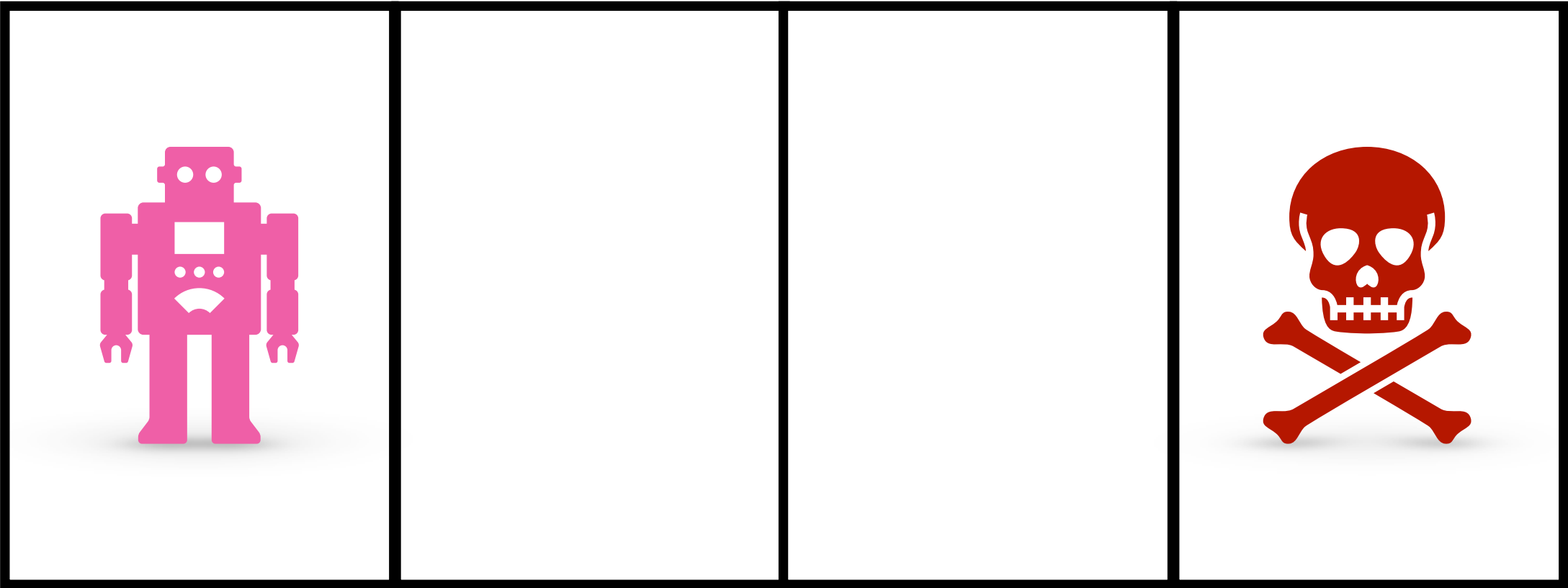
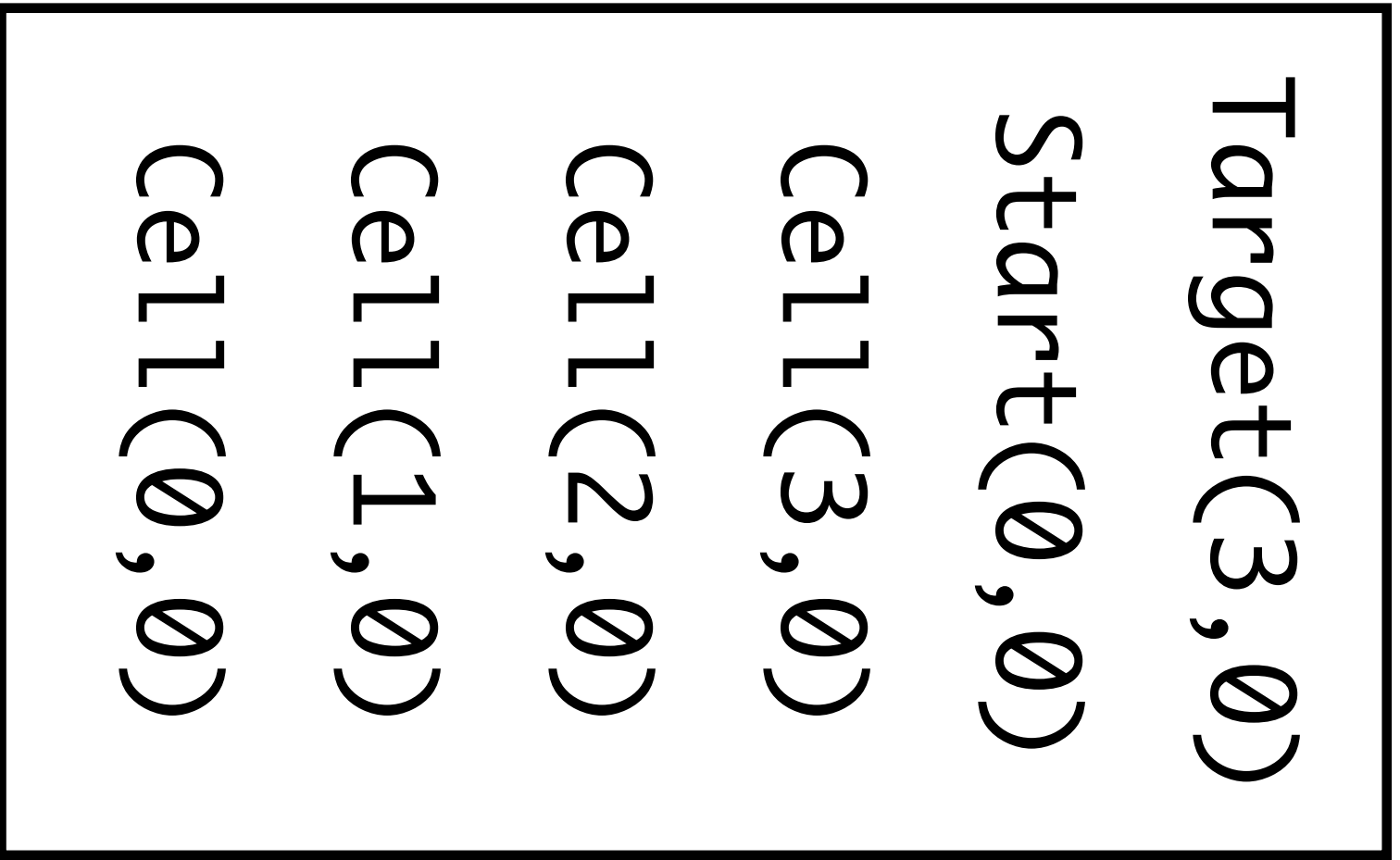
"when we run out of a products *p1* and *p2*, do not allow new orders until the stock is renewed"

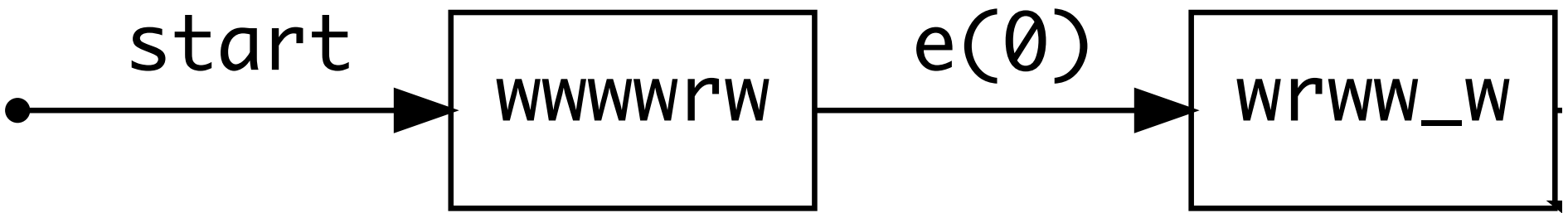
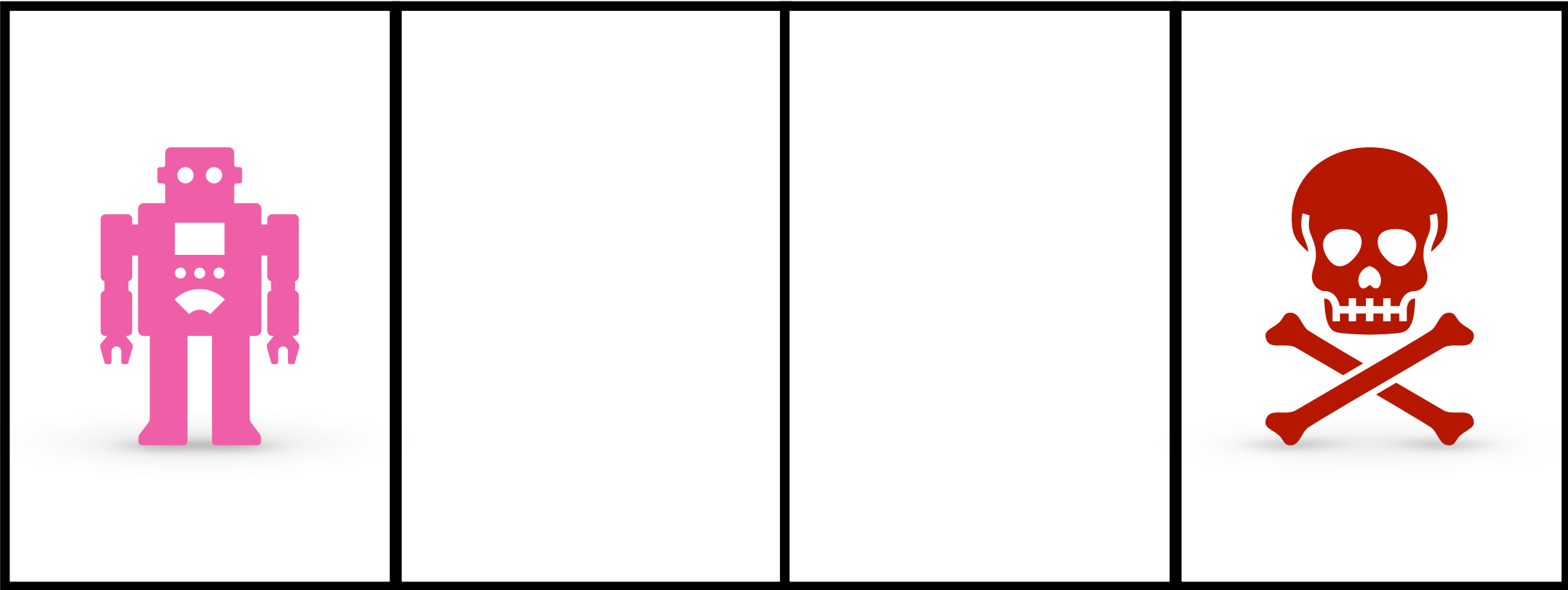
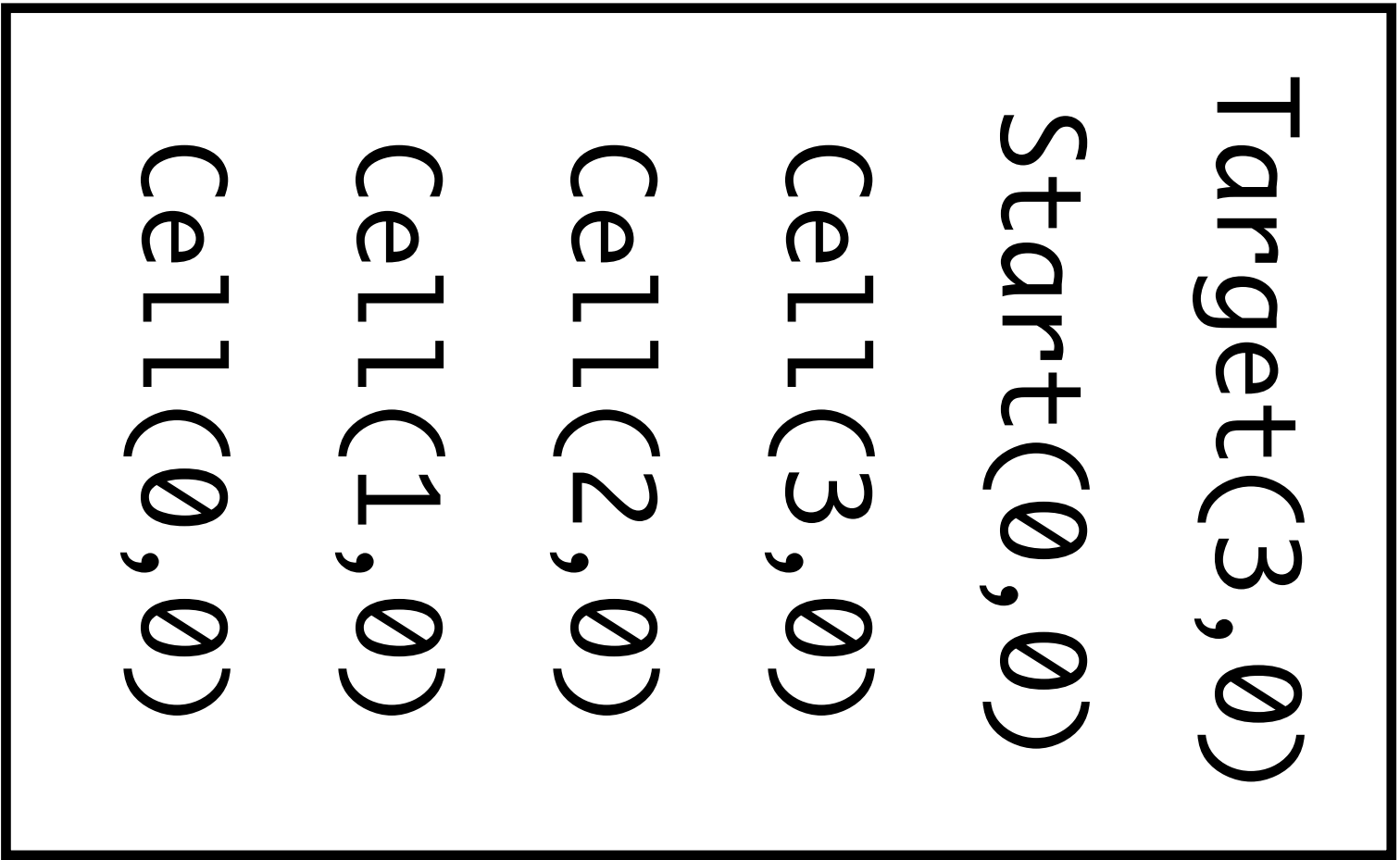
```
function blockOrdersWhenOutOf( p ) {  
  return function(){  
    while(true) {  
      bp.sync({waitFor: OutOfStock(p)});  
      bp.sync{  
        waitFor: RefillDone(p),  
        block: Order(p)  
      });  
    };  
  };  
}  
bp.registerBThread(blockOrdersWhenOutOf(p1));  
bp.registerBThread(blockOrdersWhenOutOf(p2));
```

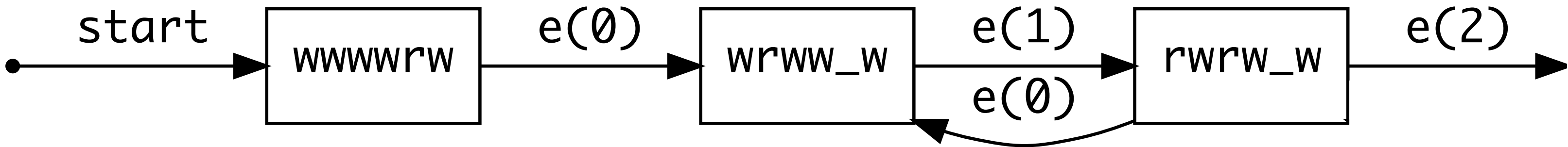
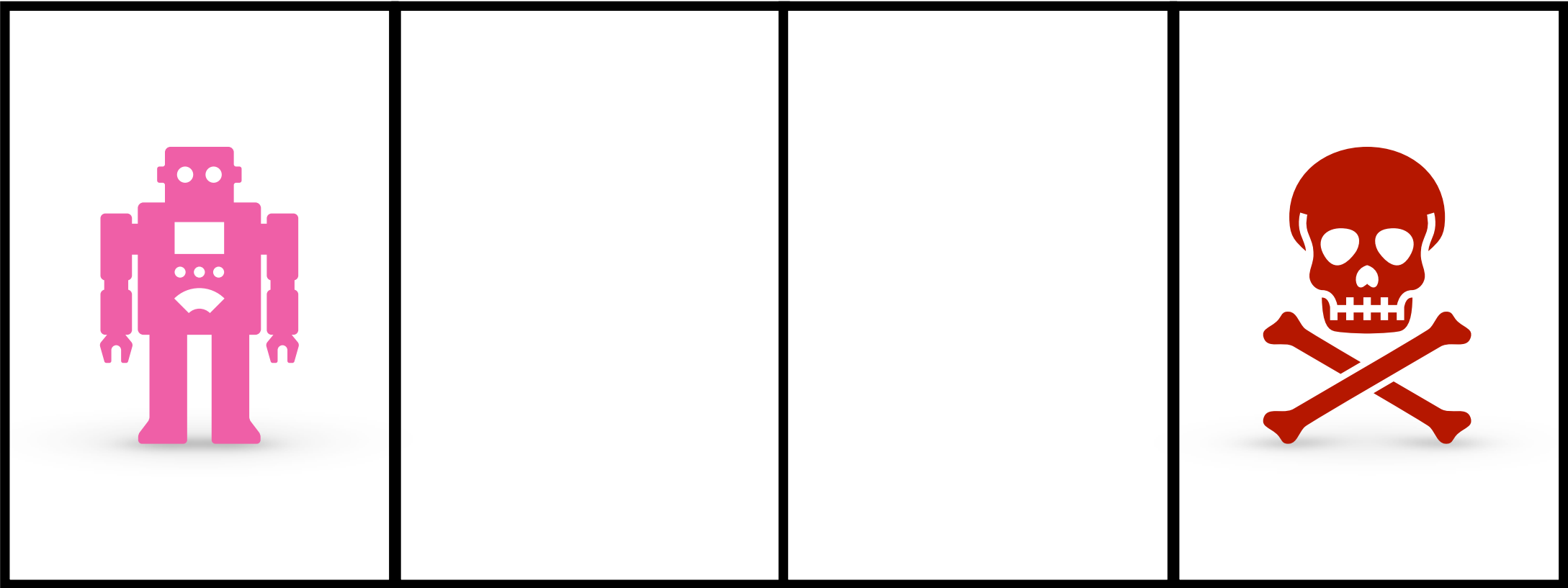
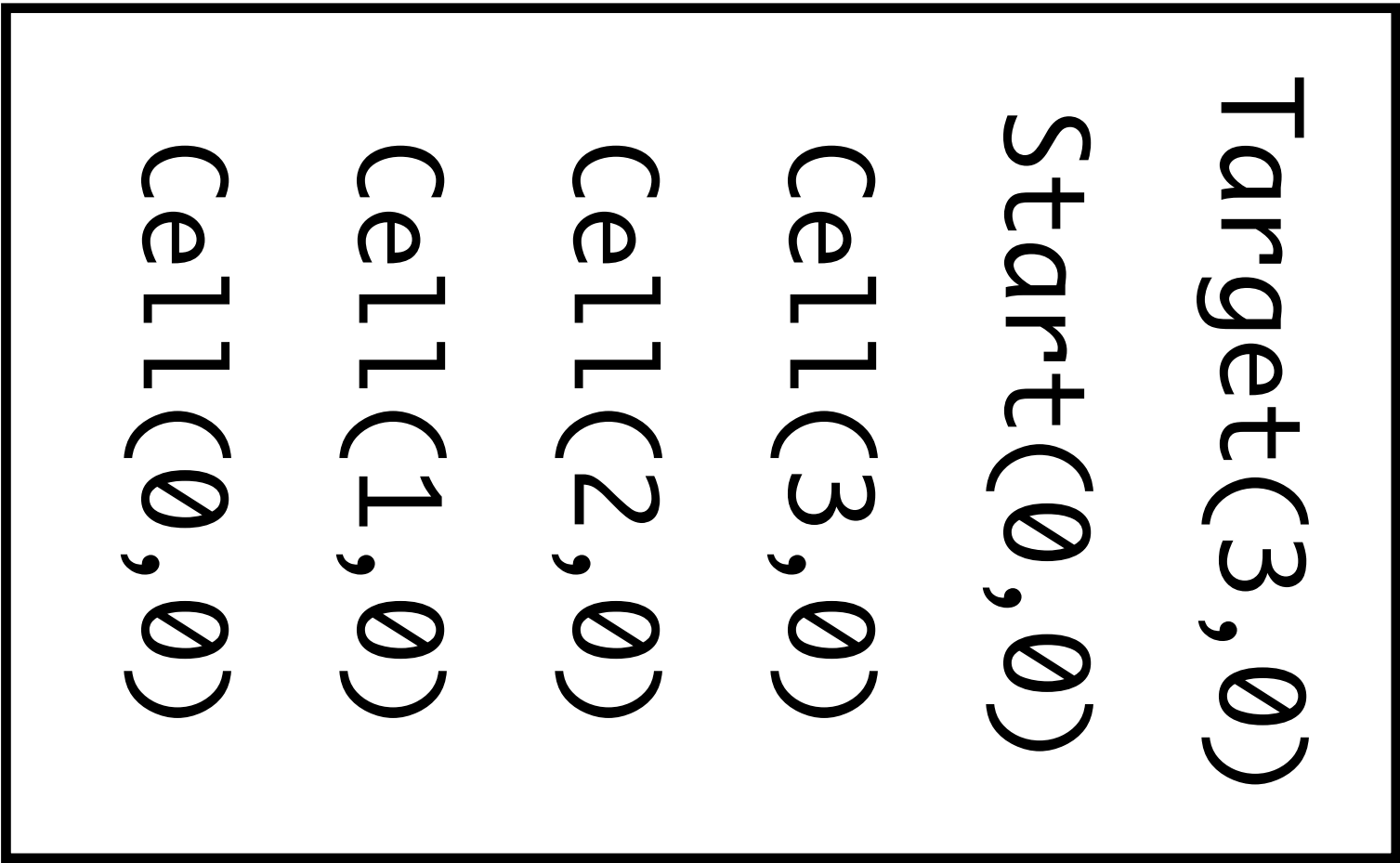


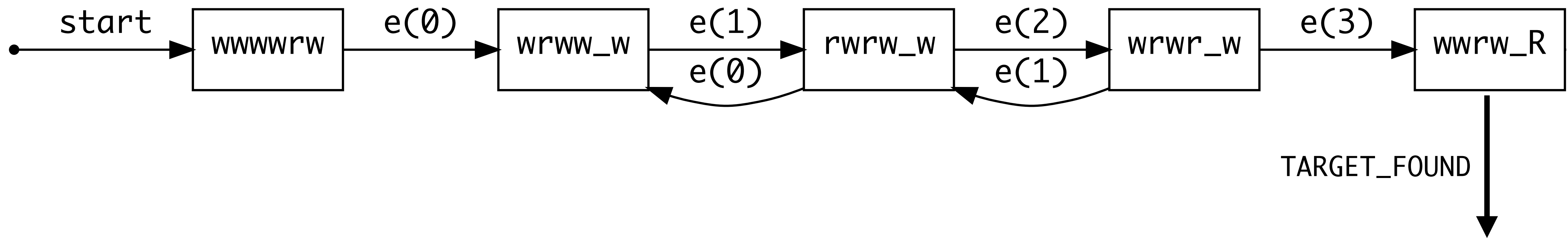
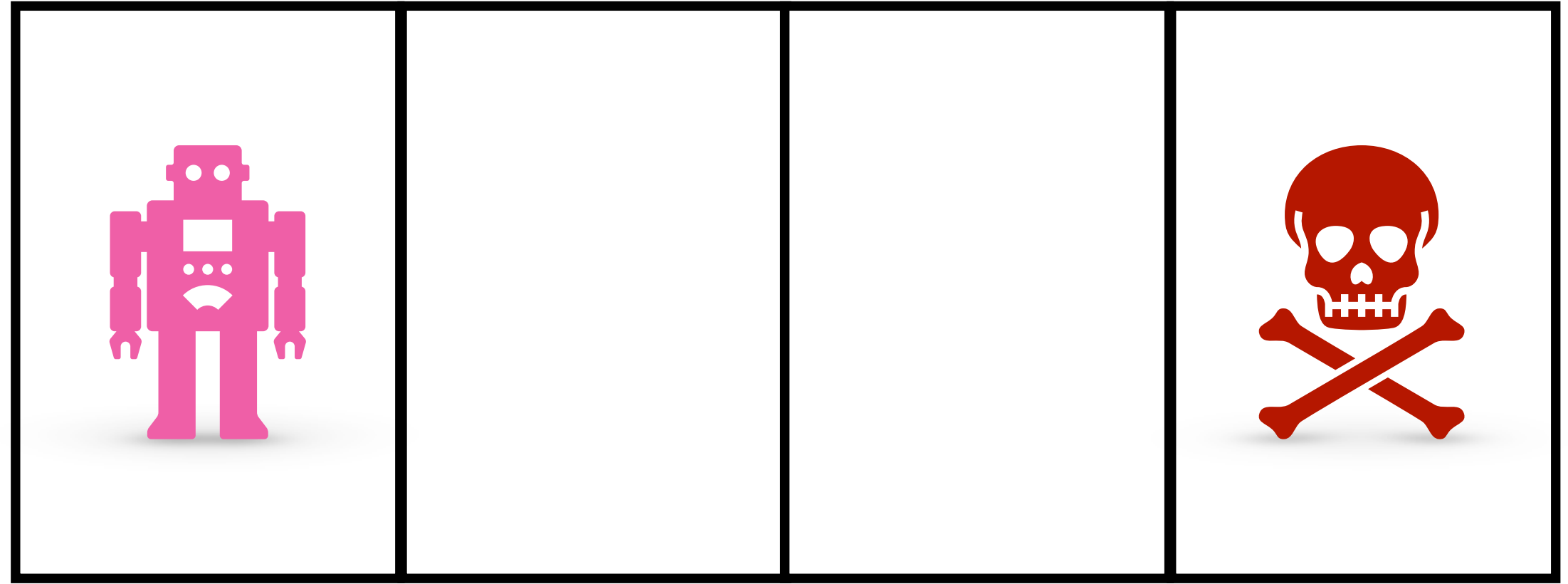
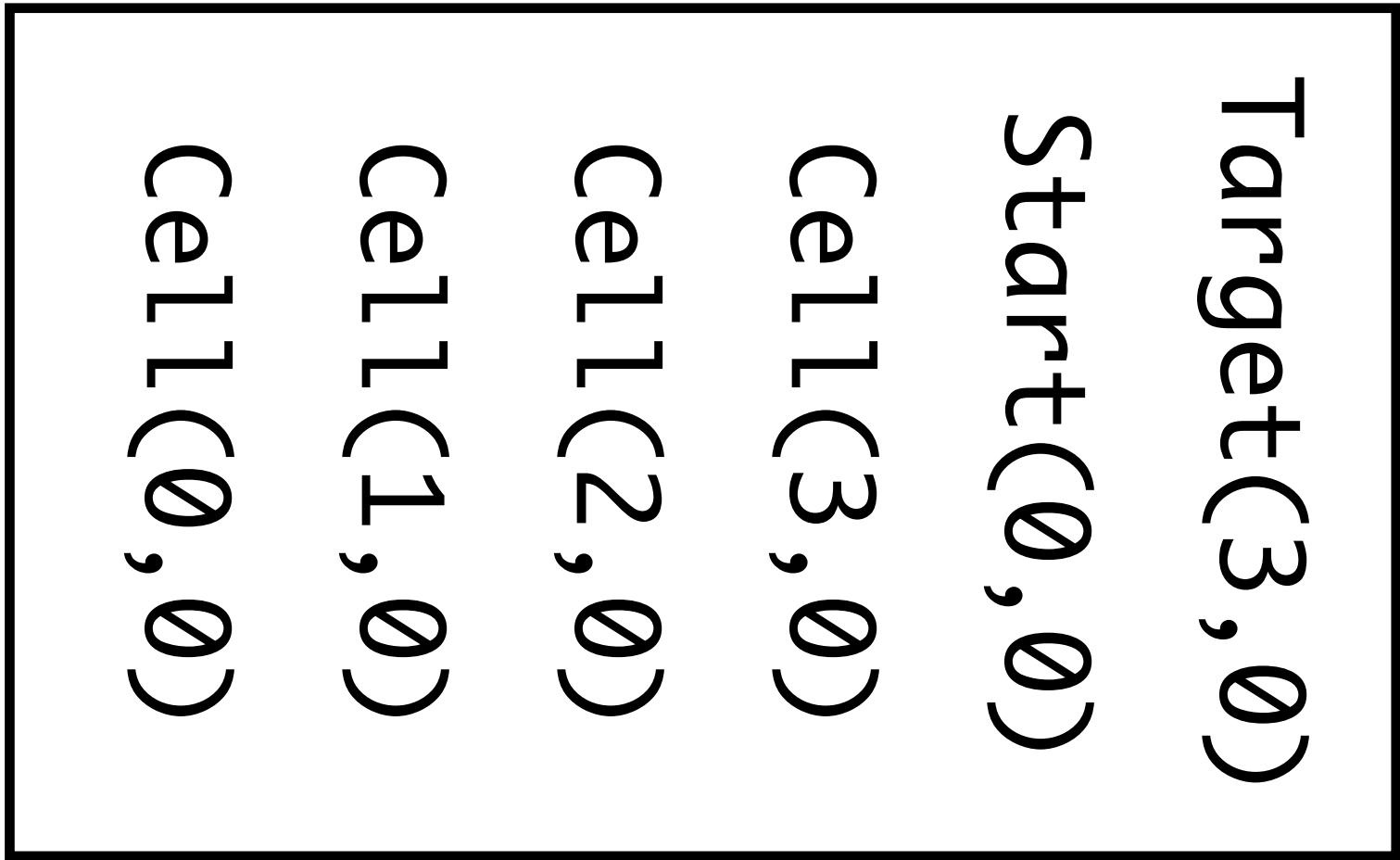


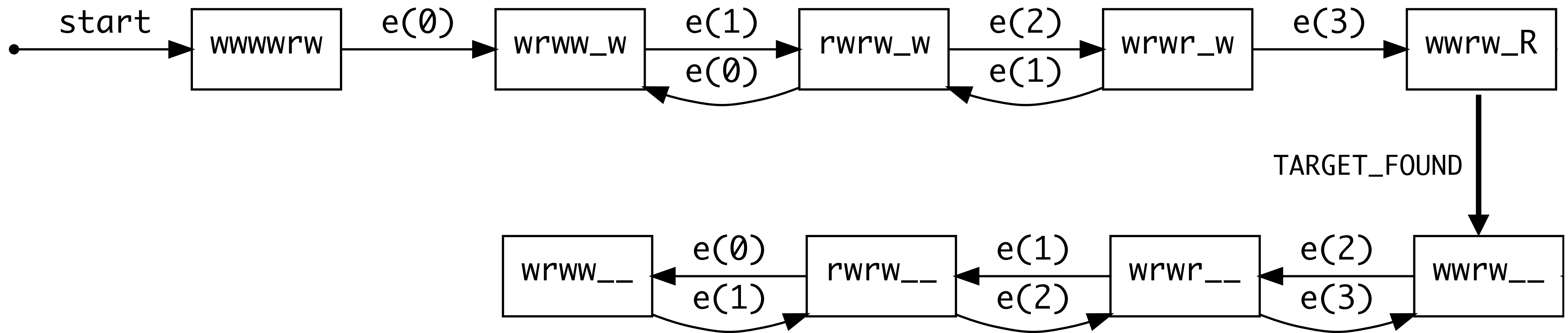
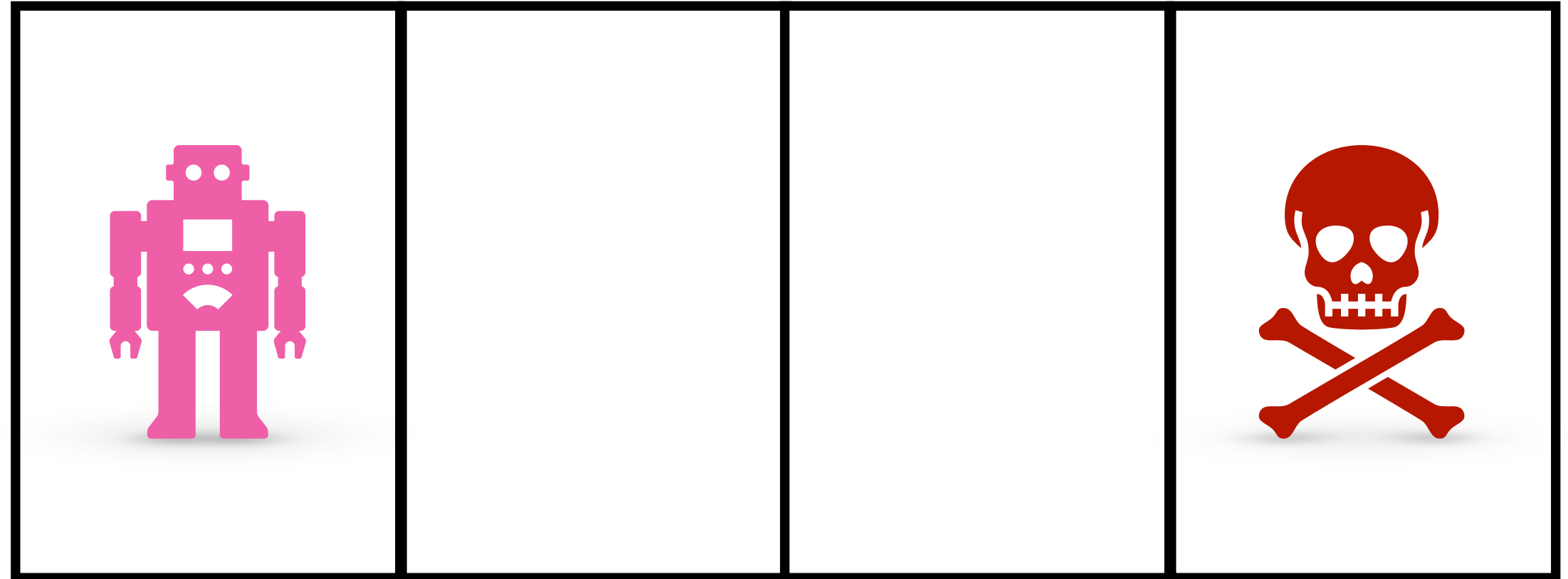
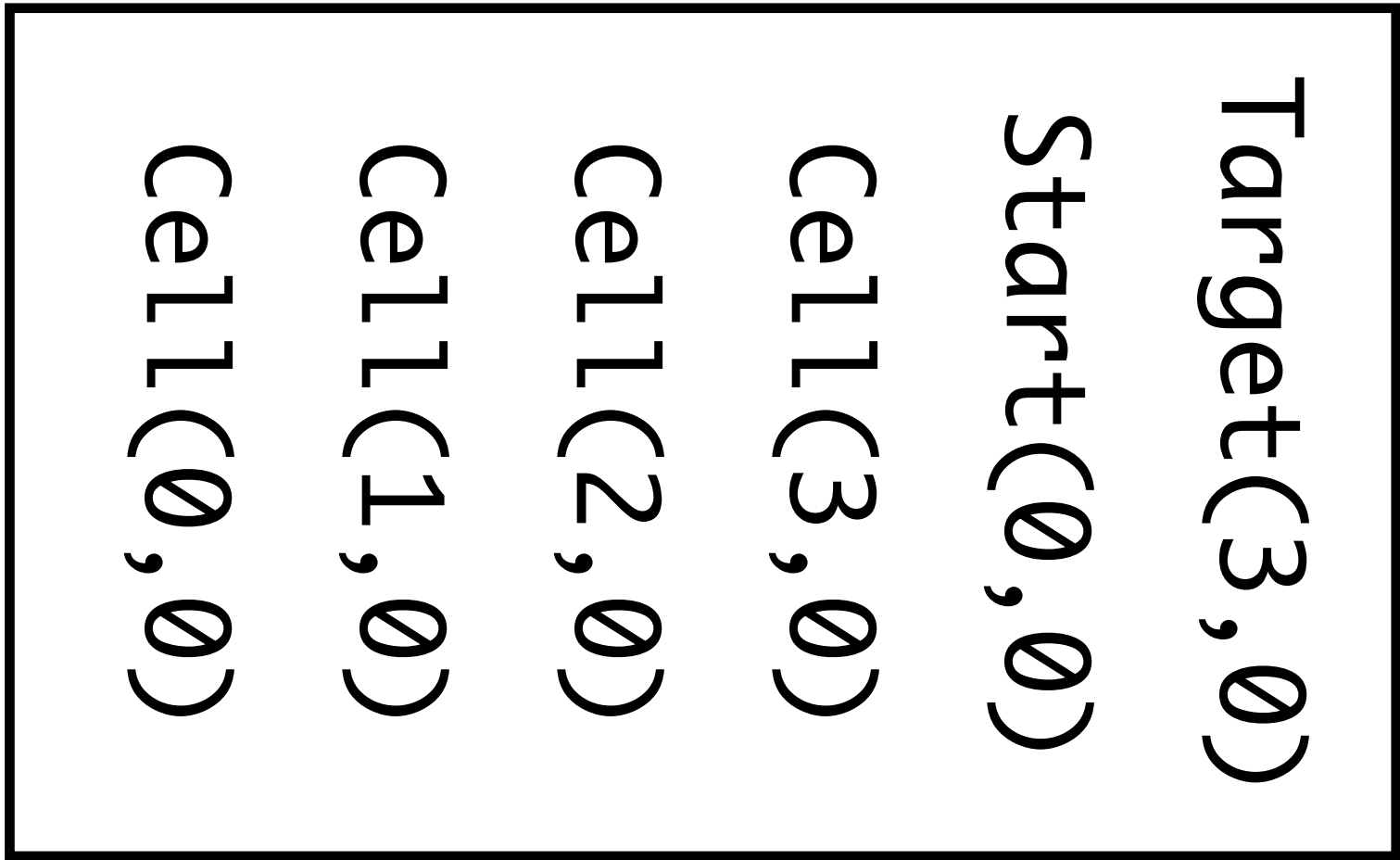


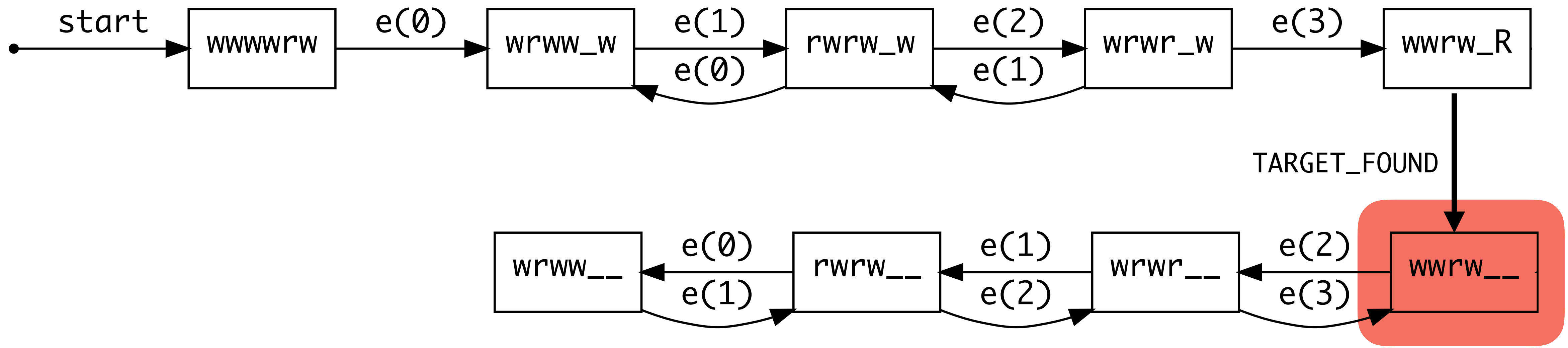
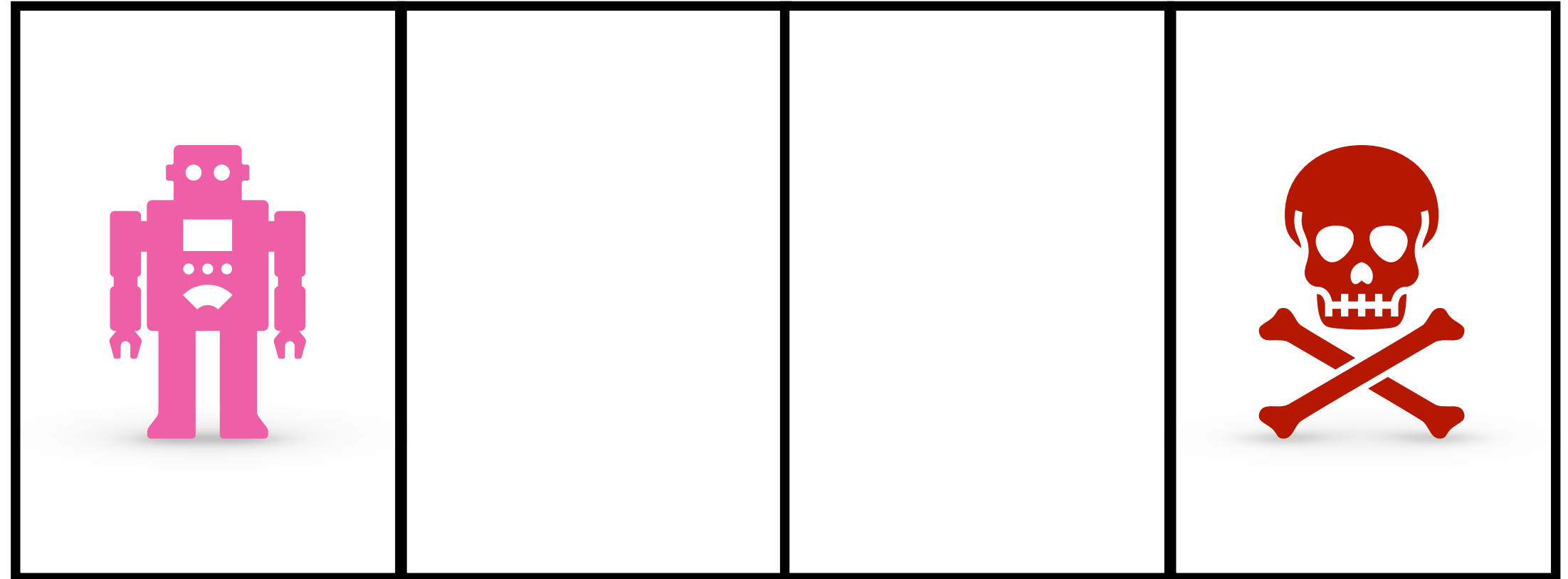
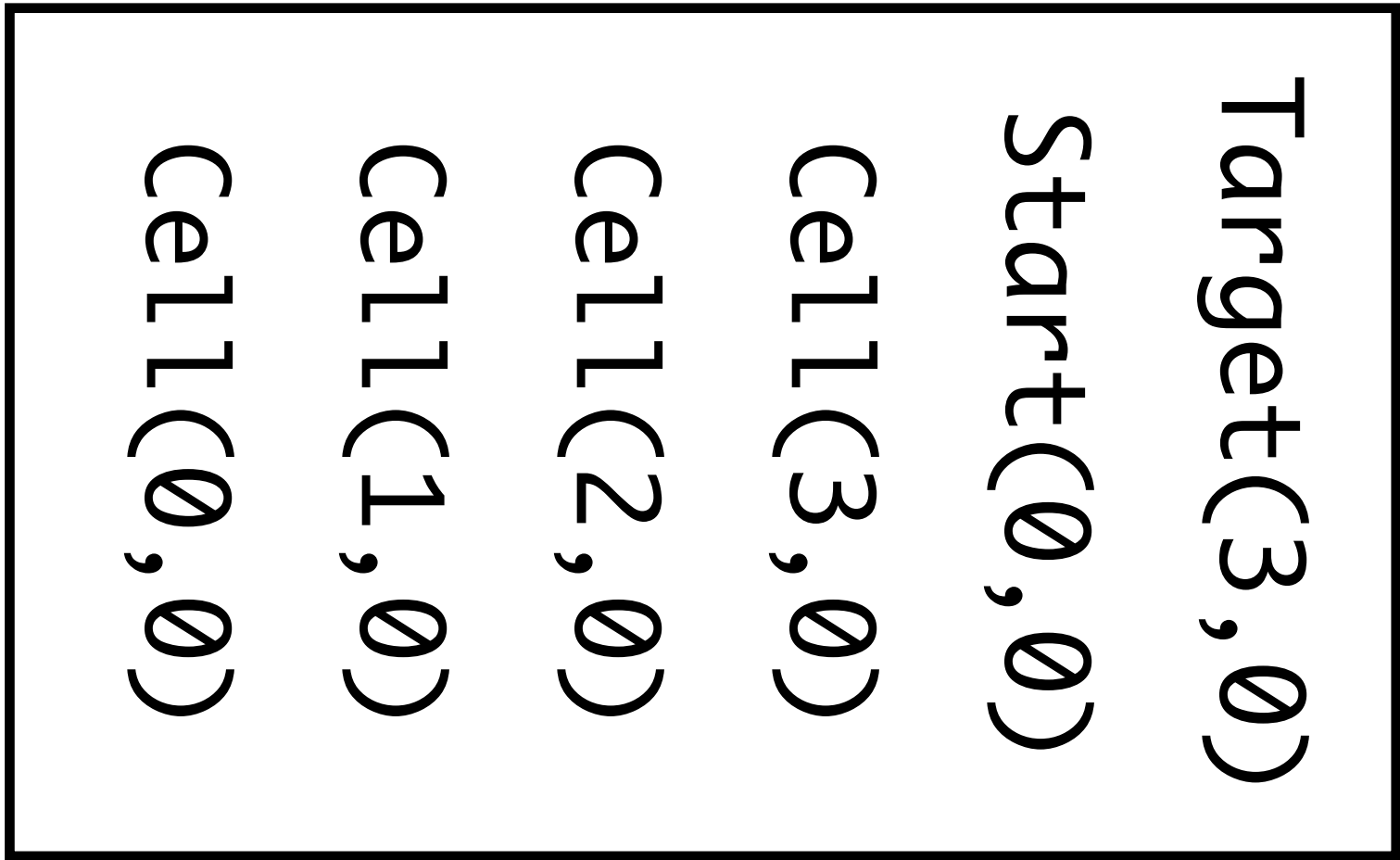






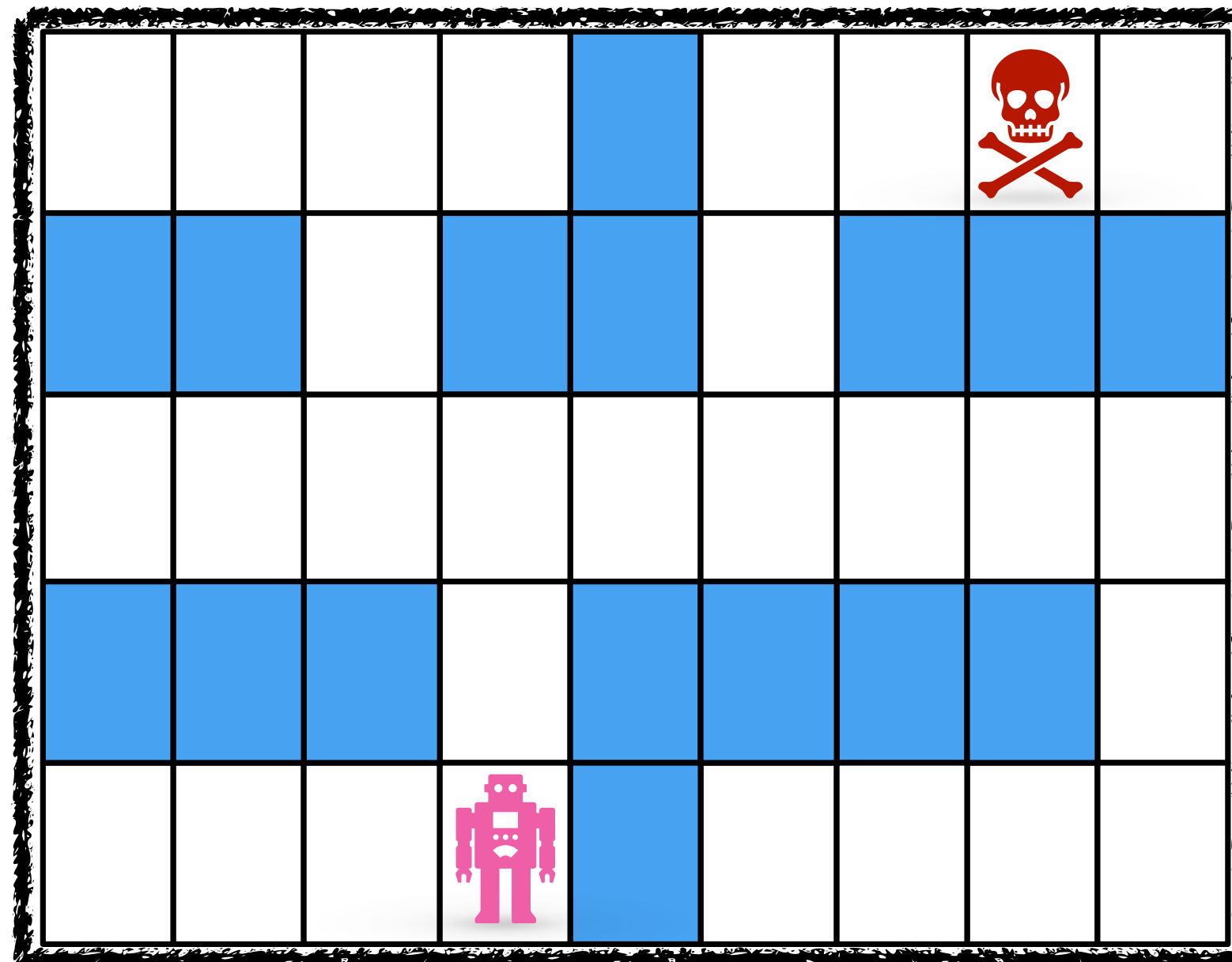






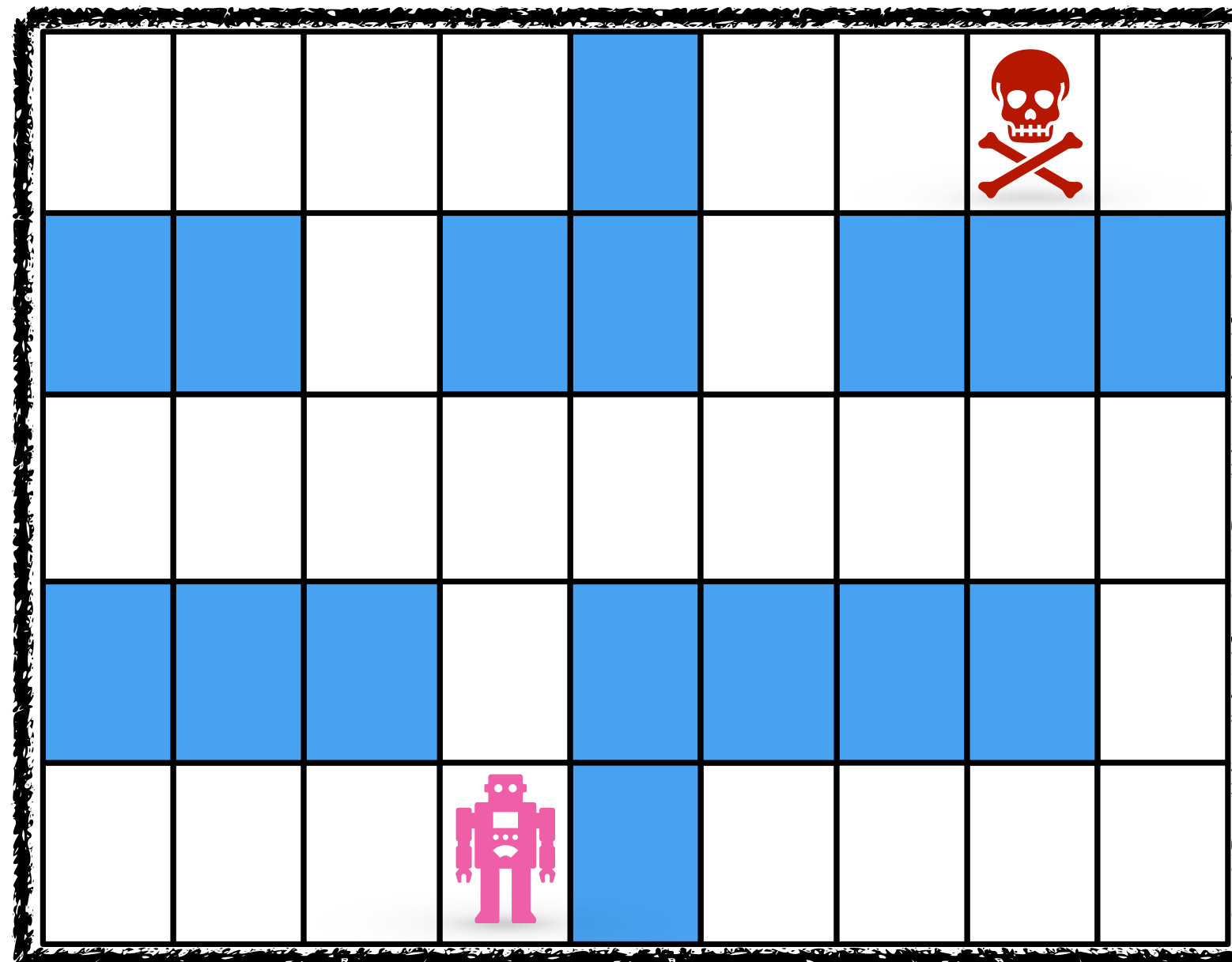
Verification Detecting "Bad States"

```
bp.registerBThread("robot-falling", function(){  
    bp.sync({waitFor:TARGET_FOUND});  
    bp.ASSERT(false, "Robot fell into trap");  
});
```



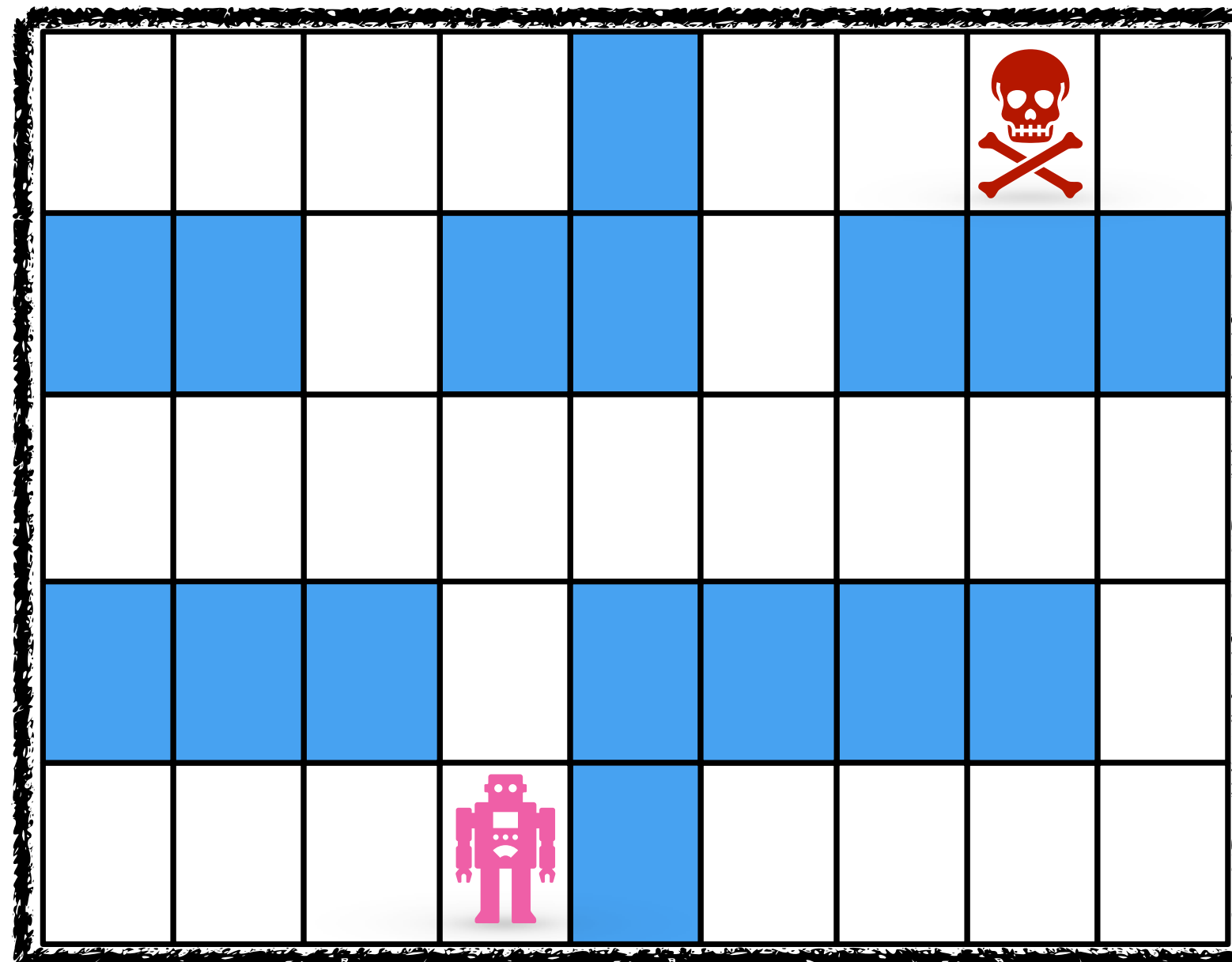
Verification Detecting "Bad States"

```
bp.registerBThread("robot-falling", function(){  
    bp.sync({waitFor:TARGET_FOUND});  
    bp.ASSERT(false, "Robot fell into trap");  
});
```



Verification Detecting "Bad States"

```
bp.registerBThread("robot-falling", function(){  
    bp.sync({waitFor:TARGET_FOUND});  
    bp.ASSERT(false, "Robot fell into trap");  
});
```

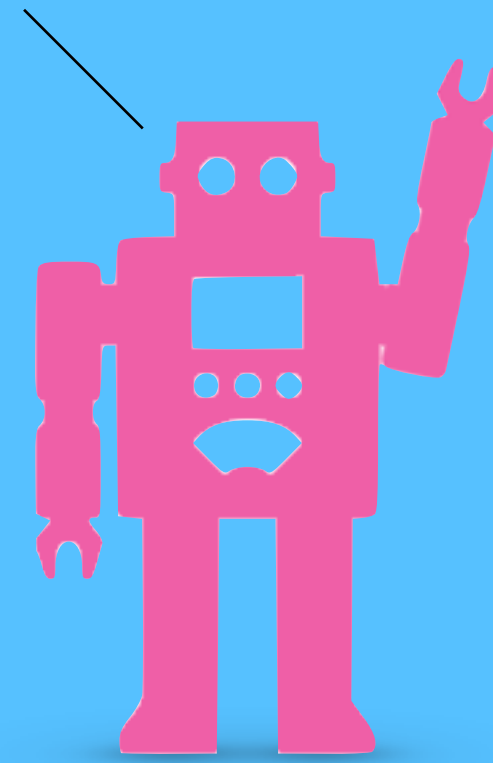
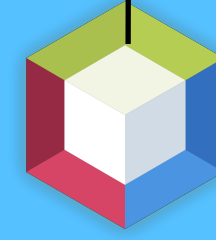


Keeping the Robot Safe

1. Add the "robot-falling" b-thread to the maze b-program
2. Traverse state graph (currently using DFS, A* demonstrated)
3. When hitting failed assertion, output trace
4. Use assumptions where possible (right)

```
bp.registerBThread("onlyOnce",
function(){
    var visited = [];
    while (true) {
        var evt = bp.sync({
            waitFor: anyEntrance,
            block: visited
        });
        visited.push(evt);
    }
});
```

Ah Ah Ah Ah
Staying alive
Staying alive



Demo

Verification in BPjs

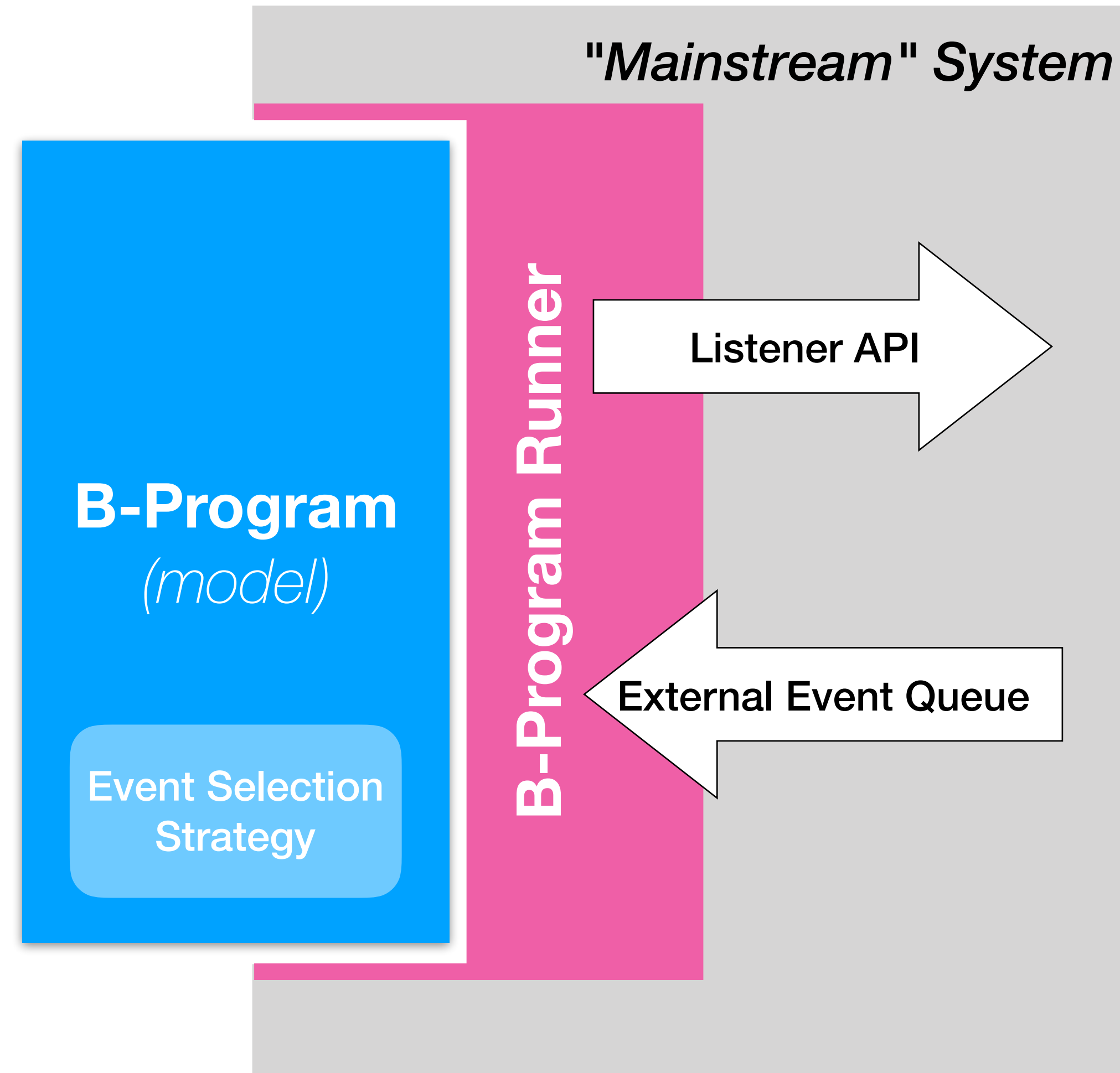
```
BProgram bprog = new StringBProgram(mazesSource);  
bprog.prependSource(mazeJs);  
bprog.appendSource(assumptionsSrc);  
bprog.appendSource(requirementSrc);  
DfsBProgramVerifier vfr = new DfsBProgramVerifier();  
vfr.setDetectDeadlocks(false);  
vfr.setProgressListener(...);  
VerificationResult res = vfr.verify(bprog);
```

SE  BP

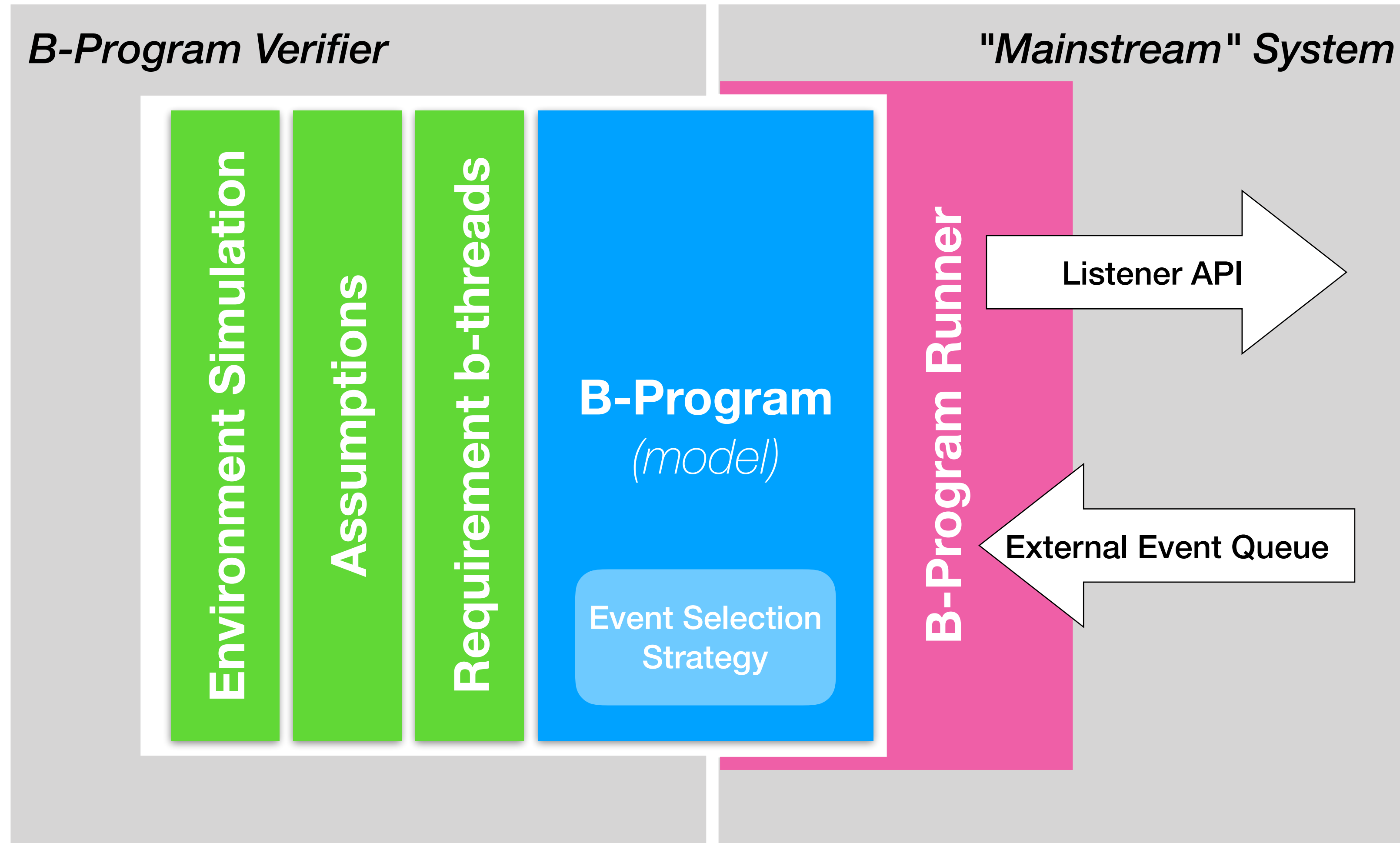
B-Program
(model)

Event Selection
Strategy

SE ? BP



SE BP



So What?

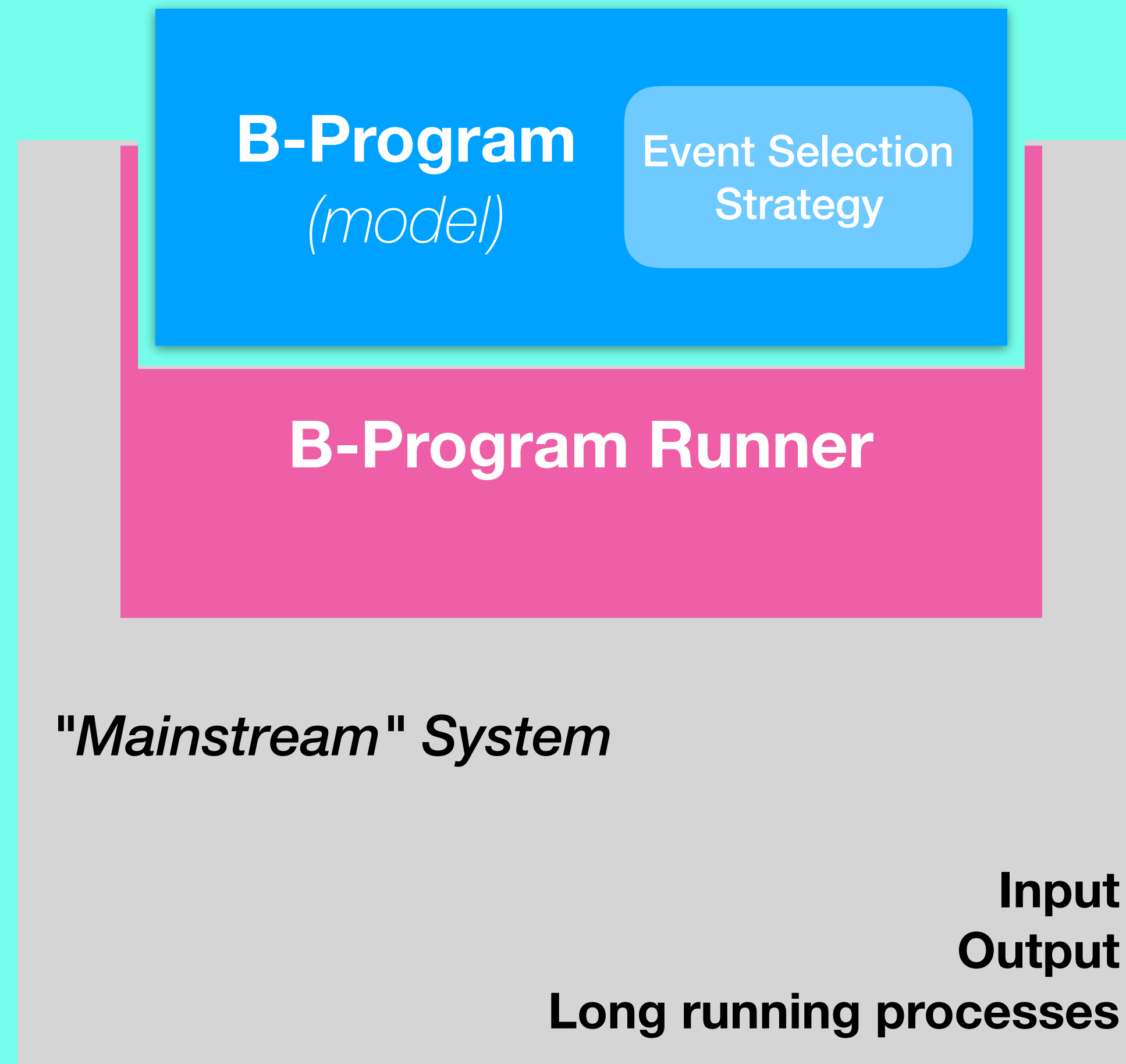
There are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies.

C.A.R Hoare, in
The Emperor's Old Clothes,
Turing Award lecture, 1980

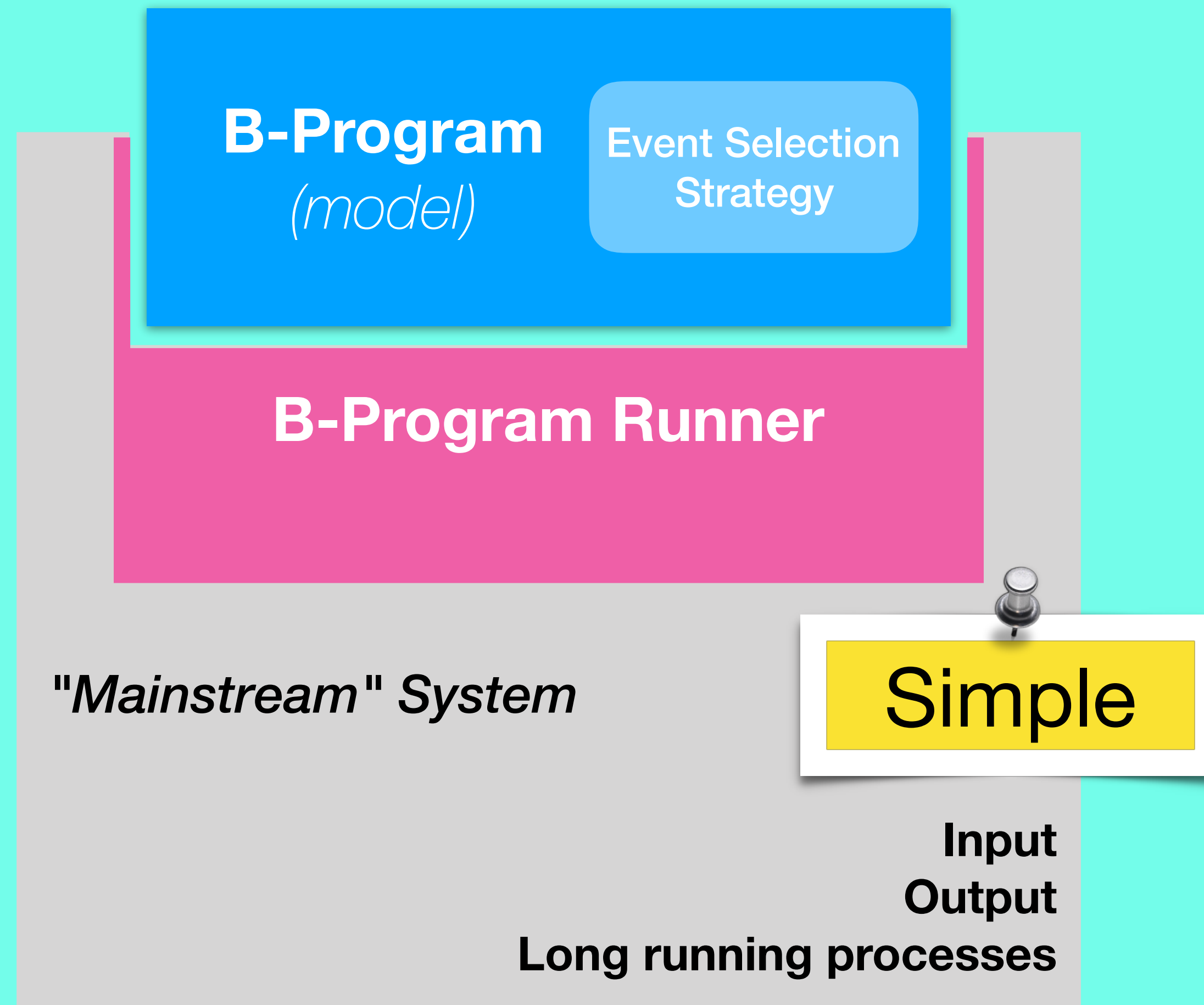


Photograph by Rama, Wikimedia Commons, Cc-by-sa-2.0-fr

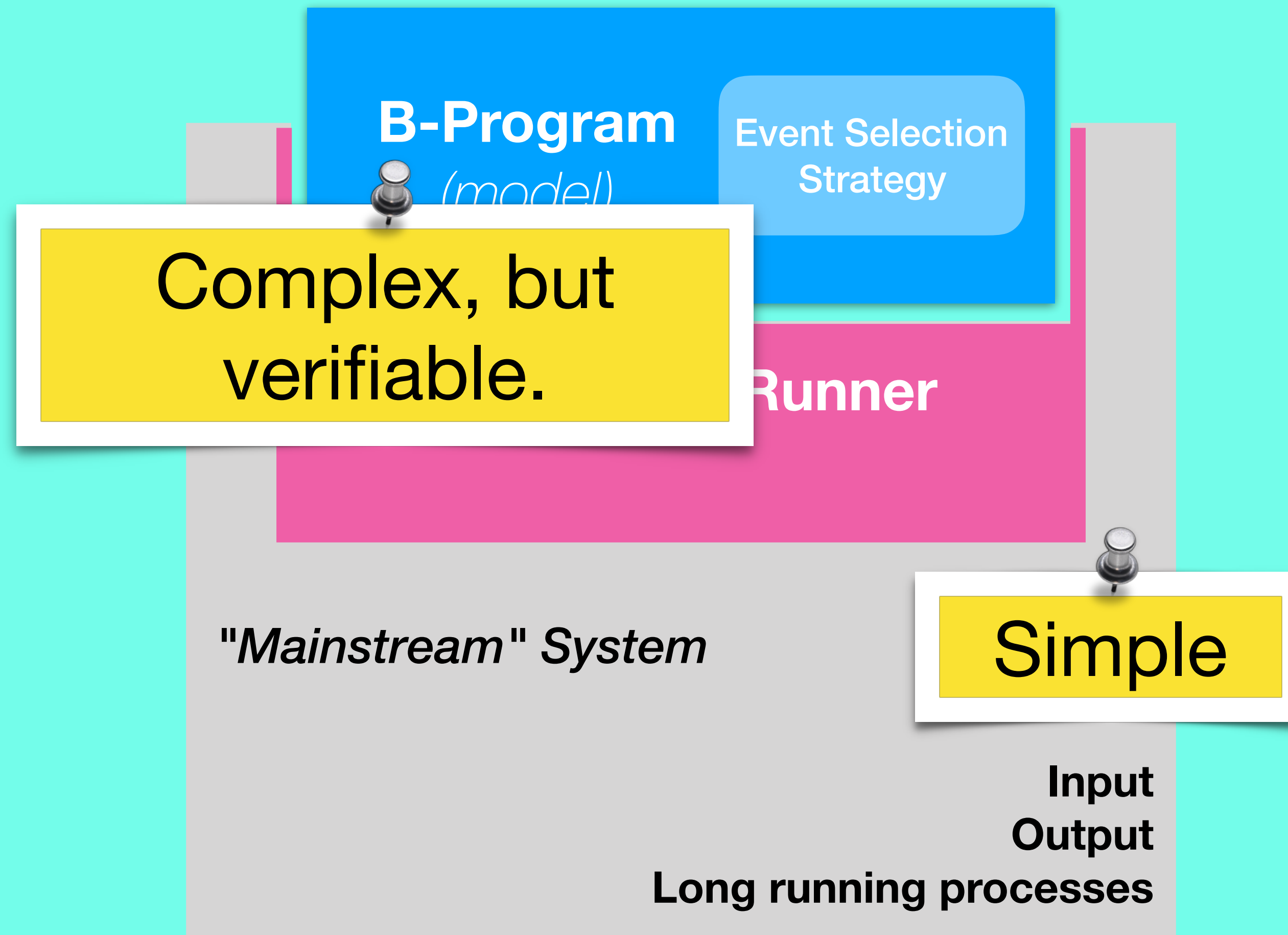
Why not Both?



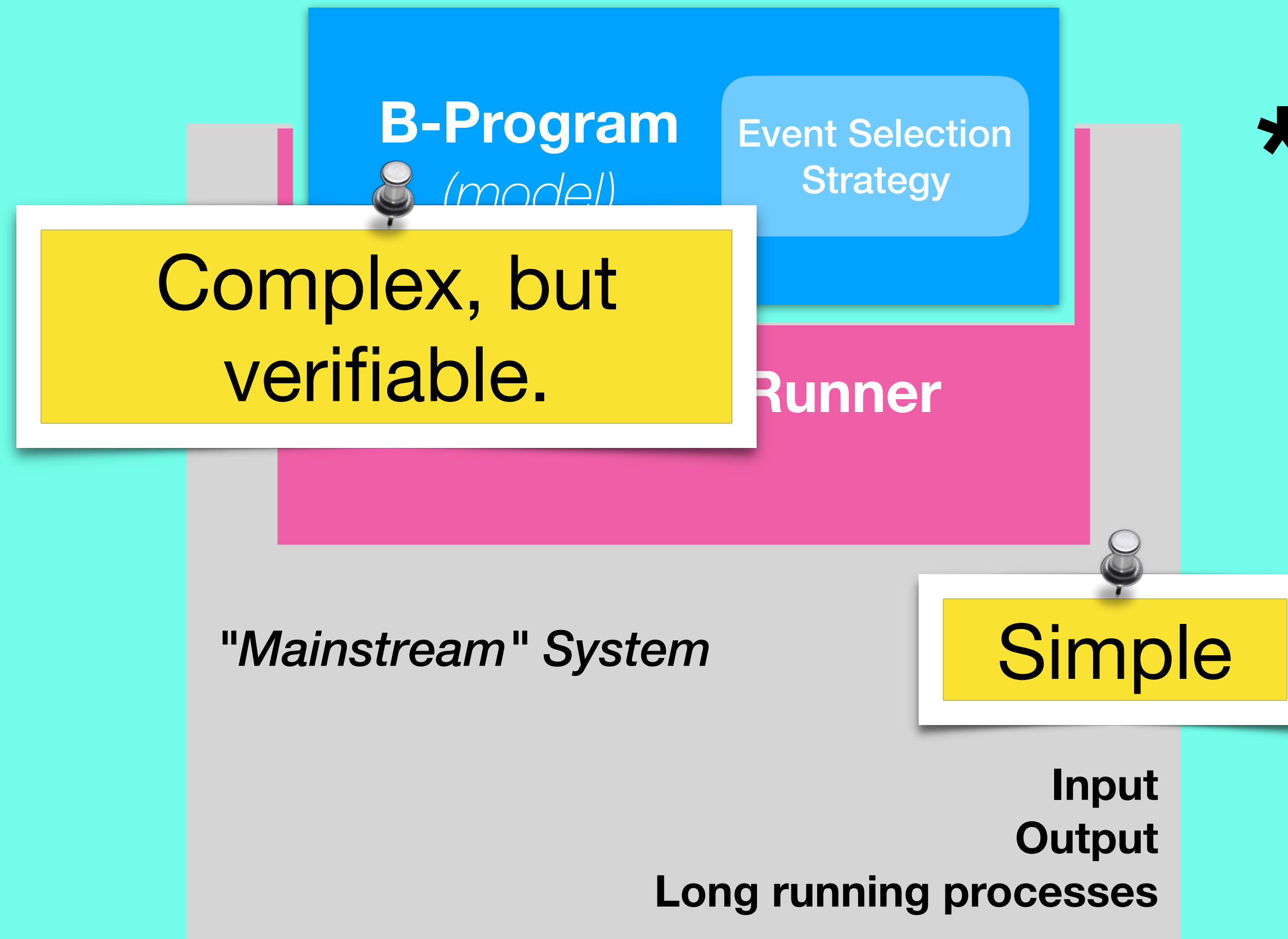
Why not Both?



Why not Both?



Why not Both?



*** Or, at least, partially verified, testable, and possibly runtime verified**



Bruno Borges @brunoborges · 1d



Write the scariest tech presentation title you can using only 4 words.

I'll start:

 356

 131

 274





Bruno Borges @brunoborges · 1d



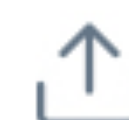
Write the scariest tech presentation title you can using only 4 words.

I'll start:

356

131

274



Simon Ritter

@speakjava



Replying to [@brunoborges](#)

Javascript for autonomous vehicles.

11/10/2018, 21:31

14 Retweets **179** Likes

```
BPjsLeaderFollower - Apache NetBeans IDE 9.0
Output - Run (BPjsLeaderFollower) x
48
---:ControllerLogic.js Event [BEvent name:Telemetry(0.800479590892792,13.132573127]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(34)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
49
---:ControllerLogic.js Event [BEvent name:Telemetry(0.776077151298523,13.360728265]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(38)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
50
---:ControllerLogic.js Event [BEvent name:Telemetry(0.749340057373047,13.597016334]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(41)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
51
---:ControllerLogic.js Event [BEvent name:Telemetry(0.720338642597198,13.842733385]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(43)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
52
---:ControllerLogic.js Event [BEvent name:Telemetry(0.689210103885651,14.098765375]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(43)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
53
---:ControllerLogic.js Event [BEvent name:Telemetry(0.656153559684753,14.365425105]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(44)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
54
---:ControllerLogic.js Event [BEvent name:Telemetry(0.617357671260834,14.673978805]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(44)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
55
---:ControllerLogic.js Event [BEvent name:Telemetry(0.580725431442261,14.962641710]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(44)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
56
---:ControllerLogic.js Event [BEvent name:Telemetry(0.542598009109497,15.261322975]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(42)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
57
---:ControllerLogic.js Event [BEvent name:Telemetry(0.503108859062195,15.569411275]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(40)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
58
---:ControllerLogic.js Event [BEvent name:Telemetry(0.46245229361038,15.865904555]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(37)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
59
---:ControllerLogic.js Event [BEvent name:Telemetry(0.420022411775589,16.208841325]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(37)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
60
---:ControllerLogic.js Event [BEvent name:Telemetry(0.377592211111111,16.547777777]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(34)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
61
---:ControllerLogic.js Event [BEvent name:Telemetry(0.335162010416667,16.886714285]
---:ControllerLogic.js Event [BEvent name:GoSlowGradient(34)]
---:ControllerLogic.js No Event Selected
---:ControllerLogic.js Event [BEvent name:Tick]
```



```
2.java
Current distance: 12.5946128156428
Current distance: 12.530860531152241
Current distance: 12.485946375074015
Current distance: 12.468915297233359
Current distance: 12.477892064763957
Current distance: 12.51051107961013
Current distance: 12.571489621366448
Current distance: 12.64952797423218
Current distance: 12.746063365553715
Current distance: 12.873708013512393
Current distance: 12.994594605723753
Current distance: 13.116232552414057
Current distance: 13.23792292617276
Current distance: 13.3572062093325
Current distance: 13.443025992312002
Current distance: 13.594301760385227
Current distance: 13.72019207272964
Current distance: 13.808595730733
Current distance: 13.87360657280038
Current distance: 13.9120412006
Current distance: 13.9245172104314
Current distance: 13.214737282697197
```

Robot Control Panel

RoverControl

start sim

Rover Gps X:	-1.401
Rover Gps Y:	29.453
Leader Gps X:	-1.401
Leader Gps Y:	29.453
Distance:	13.333
Deg2Target:	-0.463

Joel Greenyer, Michael Bar-Sinai, Gera Weiss, Aviran Sadon and Assaf Marron. Modeling and programming a leader-follower challenge problem with scenario-based tools. MDETools Workshop @ MODELS2018, Copenhagen

Space Applications

Control

Simulation

Pass

Angular Velocity

Battery Level Auto 77

Status

Simulation Status

Simulation Time

Battery Level

Mode

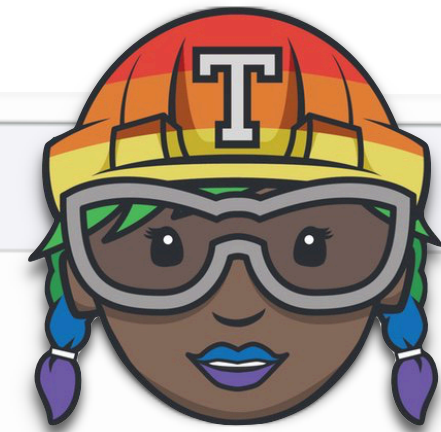
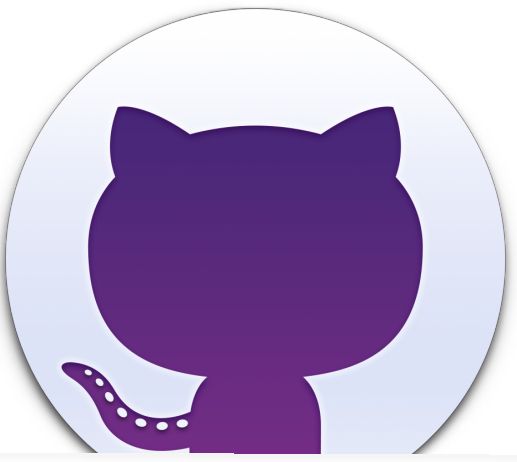
Mode

Angular Rate

Event Log

```
ADCSTelemetry currentADCMode:PayloadPointing, angularRate...
LocationTel... isOverTarget:true
EPSTelemetry vBatt:76, currentEPSMode:Good
ADCSTelemetry currentADCMode:PayloadPointing, angularRate...
LocationTel... isOverTarget:true
EPSTelemetry vBatt:76, currentEPSMode:Good
ADCSTelemetry currentADCMode:PayloadPointing, angularRate...
LocationTel... isOverTarget:true
EPSTelemetry vBatt:77, currentEPSMode:Good
ADCSTelemetry currentADCMode:PayloadPointing, angularRate...
LocationTel... isOverTarget:true
EPSTelemetry vBatt:77, currentEPSMode:Good
ADCSTelemetry currentADCMode:PayloadPointing, angularRate...
LocationTel... isOverTarget:true
EPSTelemetry vBatt:77, currentEPSMode:Good
ADCSTelemetry currentADCMode:PayloadPointing, angularRate...
LocationTel... isOverTarget:true
EPSTelemetry vBatt:77, currentEPSMode:Good
ADCSTelemetry currentADCMode:PayloadPointing, angularRate...
LocationTel... isOverTarget:true
```

BPjs a BP Tool Suite for JS



- Open Source (MIT)
- Embeddable / Commandline
- Continuous UT+CC, Documentation
- GitHub Repo, Maven Central, Jars
- Sample code
- JS using Mozilla Rhino

README.md

BPjs: A Javascript-based Behavioral Programming

This repository contains a javascript-based BP library.

build passing coverage 68% maven central 0.8.4 docs develop jav

COVERALLS

```
<dependency>
  <groupId>com.github.bthink-bgu</groupId>
  <artifactId>BPjs</artifactId>
  <version>0.9.3</version>
</dependency>
```

The Central Repository

Other tools out there too!

Read the Docs
Create, host, and browse documentation.

javadoc.io
104,770,580 page views since 2014-06-10 (updated daily).

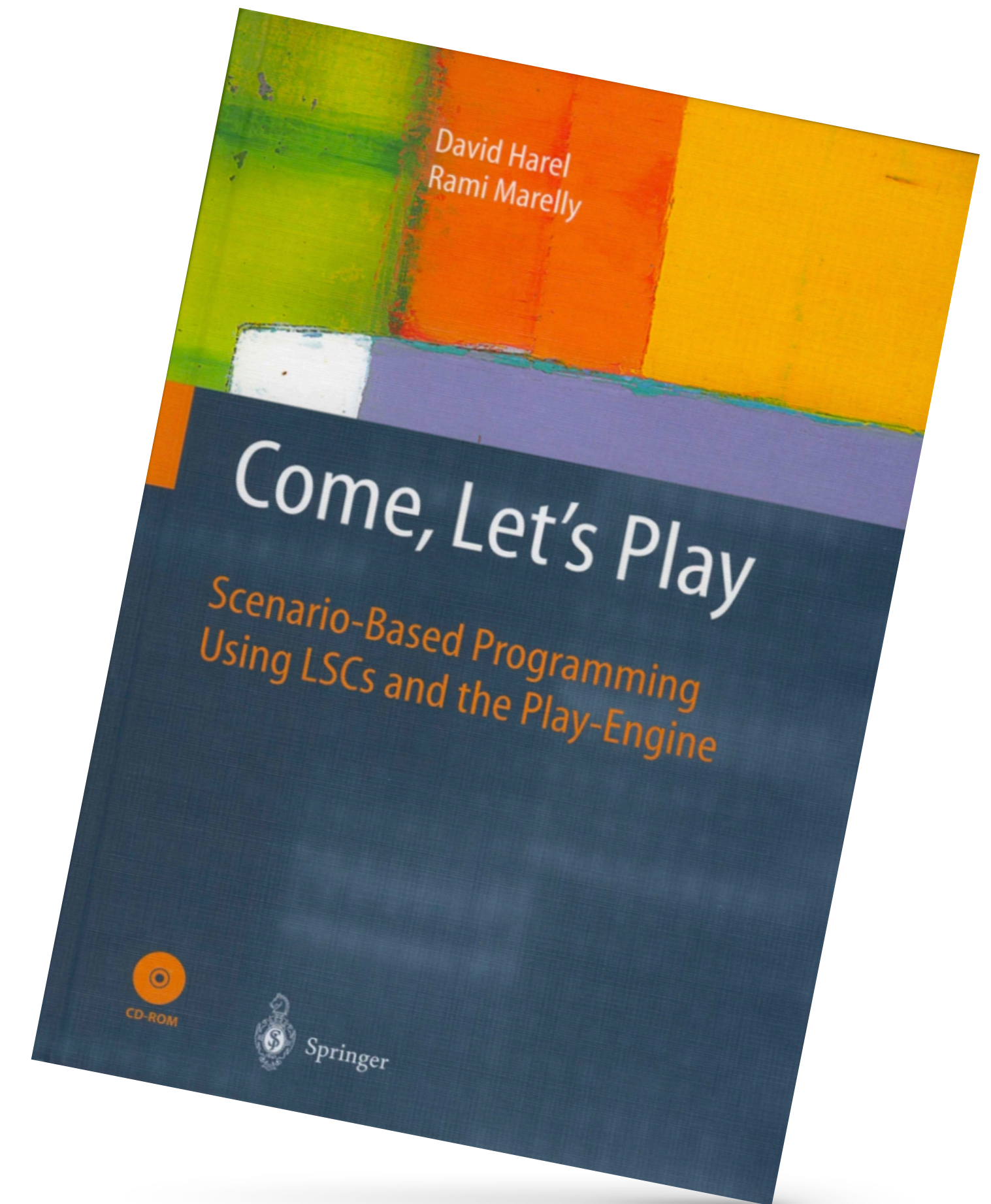
People and Works

- **Scenario Based Programming**

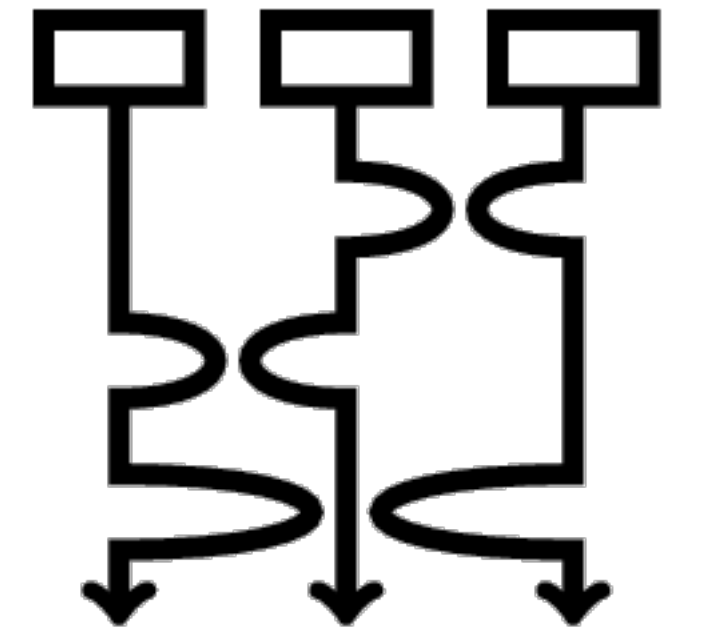
- Live Sequence Charts: **Werner Damm, David Harel** (1999, 2001)
- **David Harel, Rami Marelly**, Come, Let's Play: Scenario-Based Programming Using LSCs and the Play-Engine (2003)

- **Behavioral Programming**

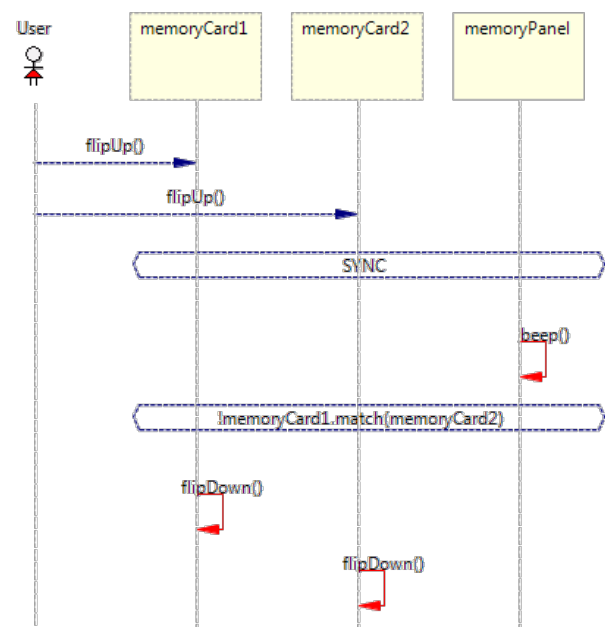
- **David Harel, Assaf Marron, Gera Weiss** (2010, 2012)
- *BPjs* developed at BGU Software Engineering Group, **Gera Weiss** (2017)
 - **Michael Bar-Sinai, Aviran Sadon, Reut Shmuel, Moshe Weinstock**
 - Collaborating with **David Harel's** Group, **Assaf Marron**
 - SE/CS Students
- Other BP Groups:
 - Weizmann Institute of Science (**David Harel**),
 - Leibniz Universität Hannover (**Joel Greenyer**)
 - JS/React: <https://github.com/lmatteis/behavioral> (**Luca Matteis**)



Resources



- b-prog.org: Main BP site, including other BP frameworks (C++, Java, Erlang, Blockly)
- **Academic Papers** (algorithms, event selections, visualizations, ...) *Behavioral Programming, Live Sequence Charts (LSCs)*
- github.com/bthink-BGU: BPjs, projects presented in this talk
- bpjs.readthedocs.io: BPjs tutorial and reference
- groups.google.com/forum/#!forum/bpjs: Google group
- scenariotools.org Live Sequence Charts take on this subject, Joel Greenyer





Please Rate Me!



If you enjoyed the talk
Or give feedback



Ben-Gurion University
of the Negev

```
// Thanks.  
bp.sync({  
  waitFor: questions  
});
```