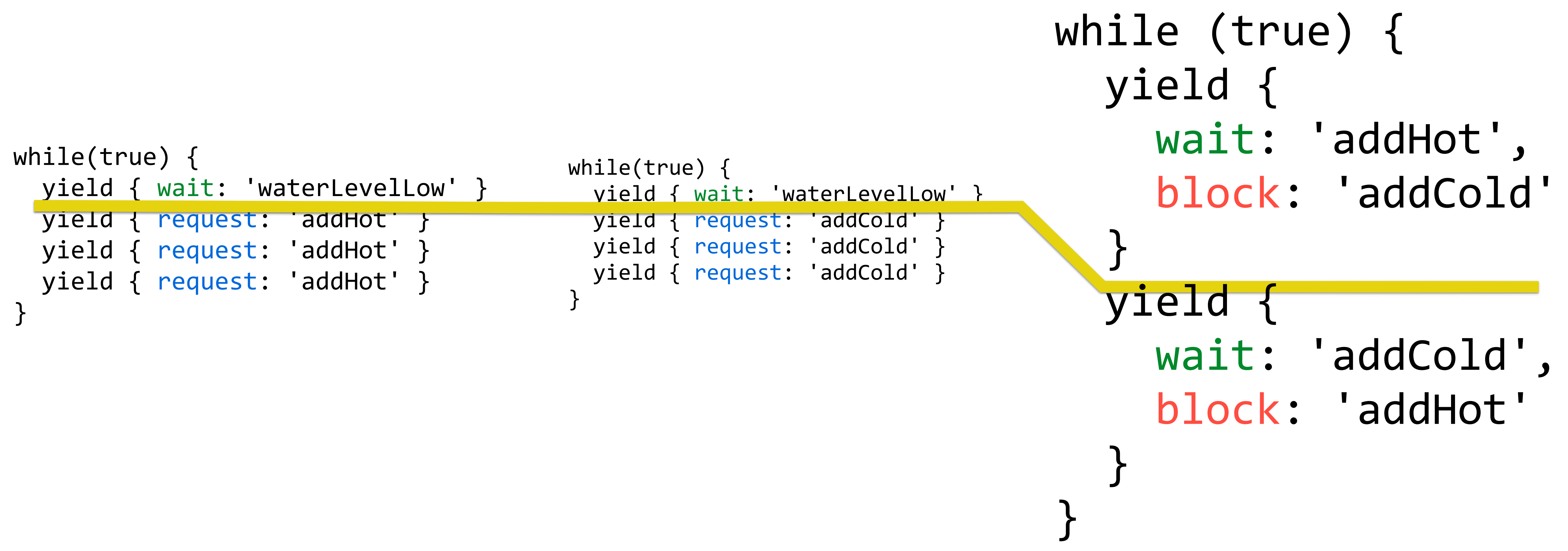


```
while(true) {
  yield { wait: 'waterLevelLow' }
  yield { request: 'addHot' }
  yield { request: 'addHot' }
  yield { request: 'addHot' }
}
```

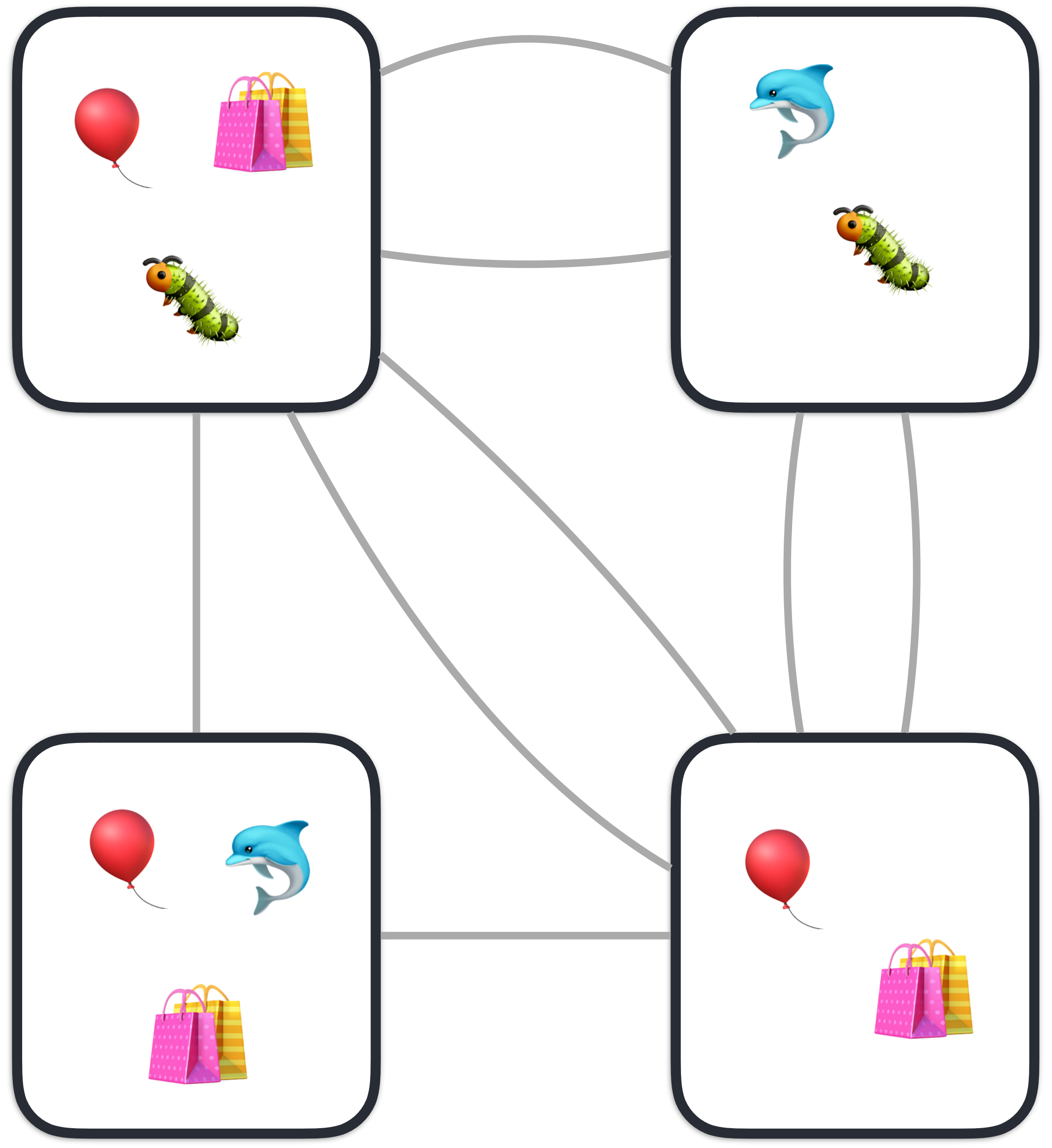
```
while(true) {
  yield { wait: 'waterLevelLow' }
  yield { request: 'addCold' }
  yield { request: 'addCold' }
  yield { request: 'addCold' }
}
```

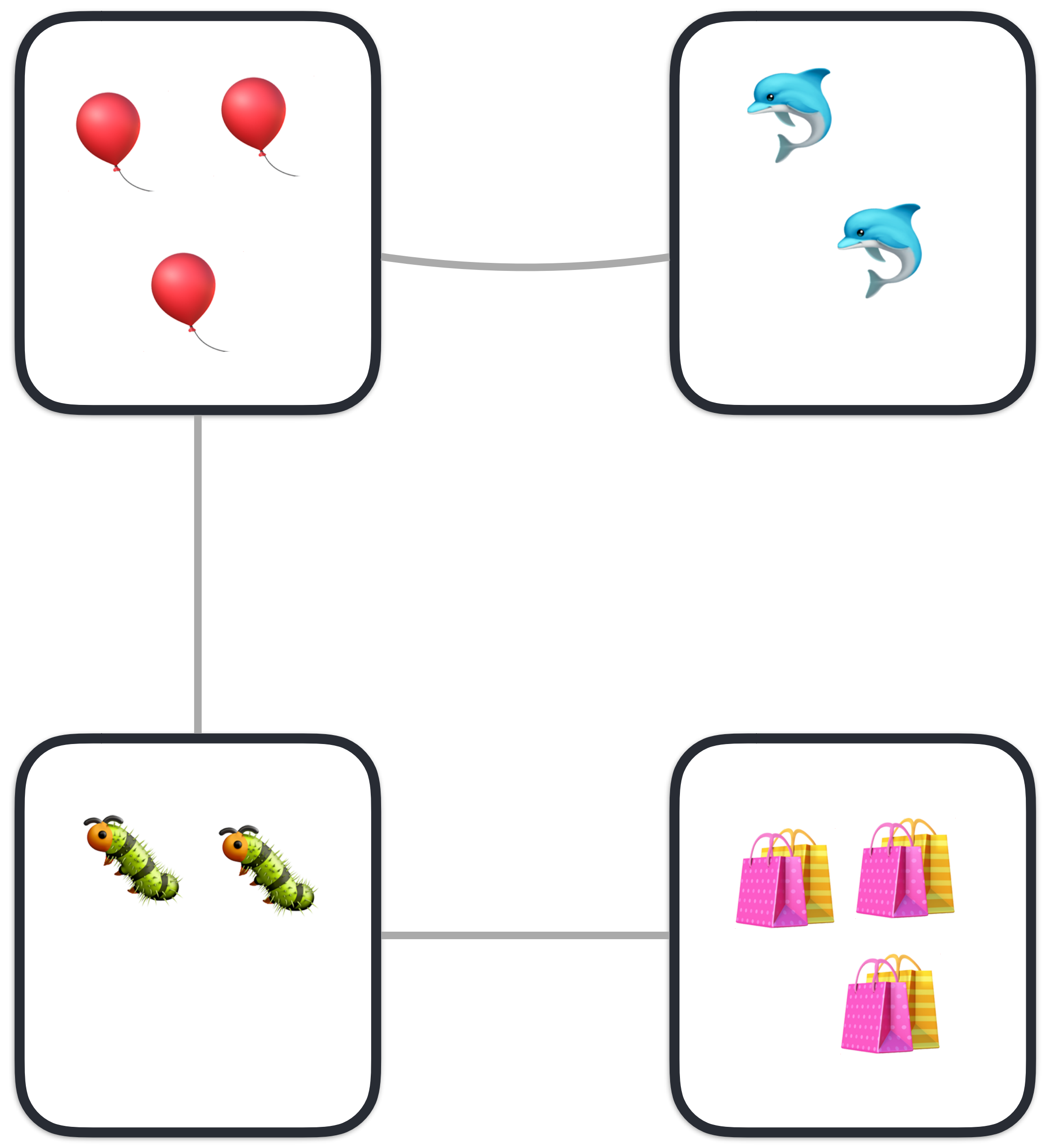
```
while (true) {
  yield {
    wait: 'addHot',
    block: 'addCold'
  }
  yield {
    wait: 'addCold',
    block: 'addHot'
  }
}
```

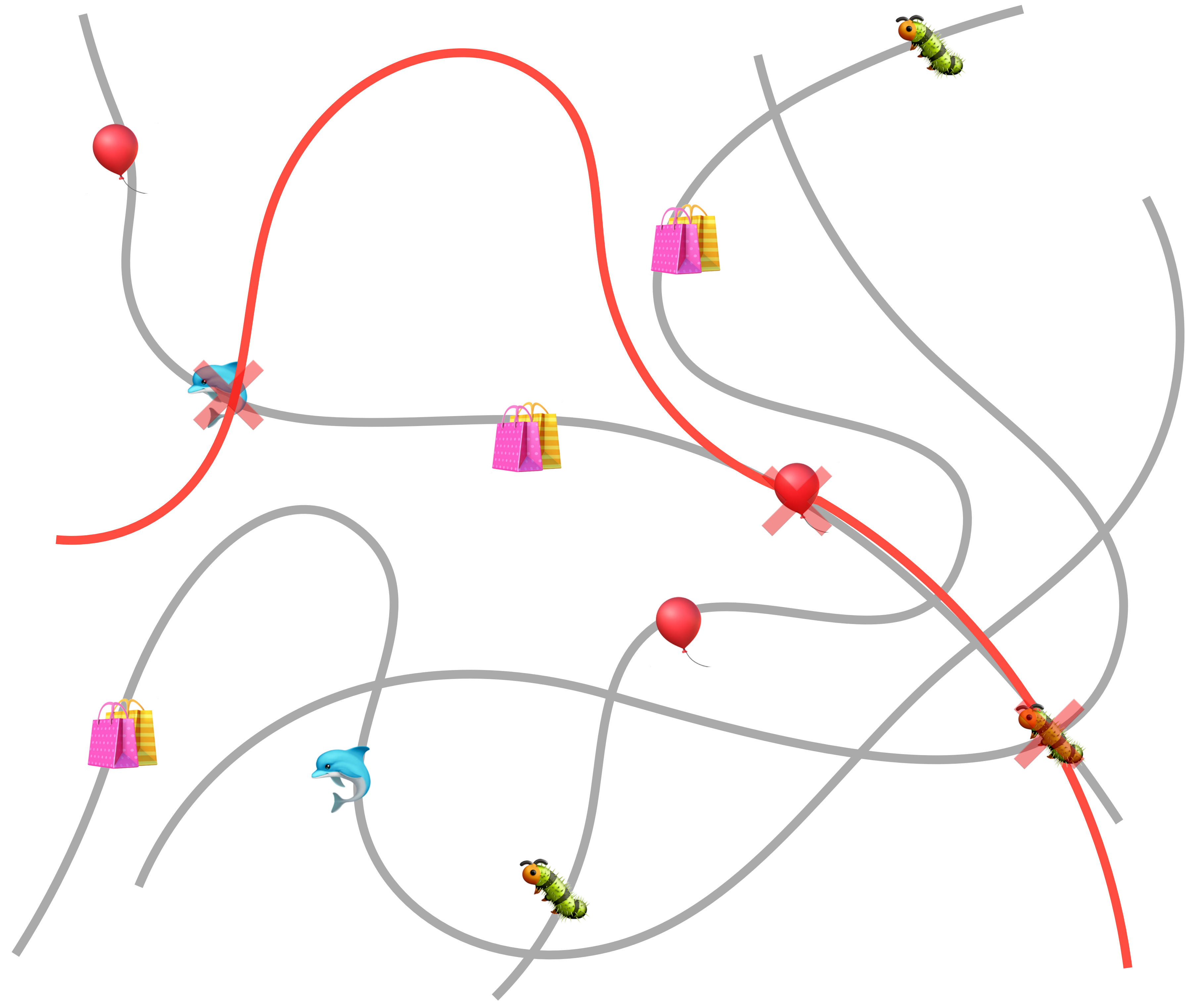
Event trace

waterLevelLow
addHot
addCold
addHot
addCold
addHot
addCold

In this approach, modularity is not necessarily achieved by the structure, but can be done by behaviours. You don't have to think of your system's behaviour as being "chopped up" into objects or tasks or components; you can chop it up any way you want according to the way you like to think about the behaviour.


Low Cohesion & High Coupling


High Cohesion & Low Coupling


Behavioral Programming