«state pattern>
Multichart Race

skirmish

```
entry /
  archer.yell(
    Event(signal=signals.Other_Skirmish_War_Cry,
      payload=archer.name))
  archer.post_fifo(
    Event(signal=signals.Officer_Lured),
    times=1,
    period=archer.to_time(
      random.randint(40, 200)),
    deferred=True)
  if archer.arrow < 10:
    archer.post_fifo(Event(signal=signals.Ammunition_Low))
Officer_Lured /
  chart.post_fifo(Retreat_War_Cry)
Senior_Skirmish_War_Cry / {}
Other_Skirmish_War_Cry as e / archer.dispatch_to_empathy(e)
Skirmish_War_Cry as e /  archer.dispatch_to_all_empathy(e)

second /
  if archer.tick % 3 == 0:
    if random.randint(1, 10) <= 4:
      archer.arrow = archer.arrow - 1 if archer.arrows >= 1 else 0
    if archer.arrows < 10:
      archer.post_fifo(
      Event(
        signal=signals.Ammunition_Low))
  archer.ticks += 1

exit /
  archer.cancel_events(Event(signal=signals.Retreat_War_Cry))
  archer.cancel_events(Event(signal=signals.Officer_Lured))
```

```
Officer_Lured /
  archer.snoop_scribble("Knight Charging")
  archer.post_fifo(
    Event(signal=signals.Retreat_War_Cry))
```

Other_Skirmish_War_Cry as e /
archer.dispatch_to_empathy(e)

Skirmish_War_Cry

«state pattern»
Multichart Pend

dispatch_to_empathy
dispatch_to_all_empathy

empathies

Outer state hook:
Other_Retreat_Ready_War_Cry
  archer.dispatch_to_empathy(e)

waiting_to_lure

```
Retreat_Ready_War_Cry /
  ready = True
  for name, other archer.others.items():
    if other.dead() is not True:
      ready &= other.waiting()
    else:
      archer.snoop_scribble(
        "{} thinks {} is dead".
        format(archer.name, name))
  if ready:
    # let's make sure the archer isn't a chicken
    delay_time = random.randint(10,50)
  else:
    delay_time = random.randint(30,60)
  archer.post_fifo(
    Event(signal=signals.Retreat_War_Cry),
    times=1,
    period=archer.to_time(
      delay_time),
    deferred=True)
```

```
Ammunition_Low /
 chart.post_fifo(
  Event(signal=signals.Retreat_Ready_War_Cry))
```

```
entry /
  archer.yell(
    Event(signal=signals.Other_Retreat_Ready_War_Cry,
          payload=archer.name))
  archer.snoop_scribble('{} has {} arrows'. \
    format(archer.name, archer.arrows))
  archer.scribble('put away bow')
  archer.scribble('pull scimitar')
  archer.snoop_scribble('{} acts scared'. \
    format(archer.name))

Ammunition_Low / {}

exit /
  archer.scribble('stash scimitar')
  archer.scribble('pull bow')
  archer.scribble('stop acting')

second /
  archer.ticks += 1

exit /
  archer.scribble('stash scimitar')
  archer.scribble('pull bow')
  archer.scribble('stop acting')
```