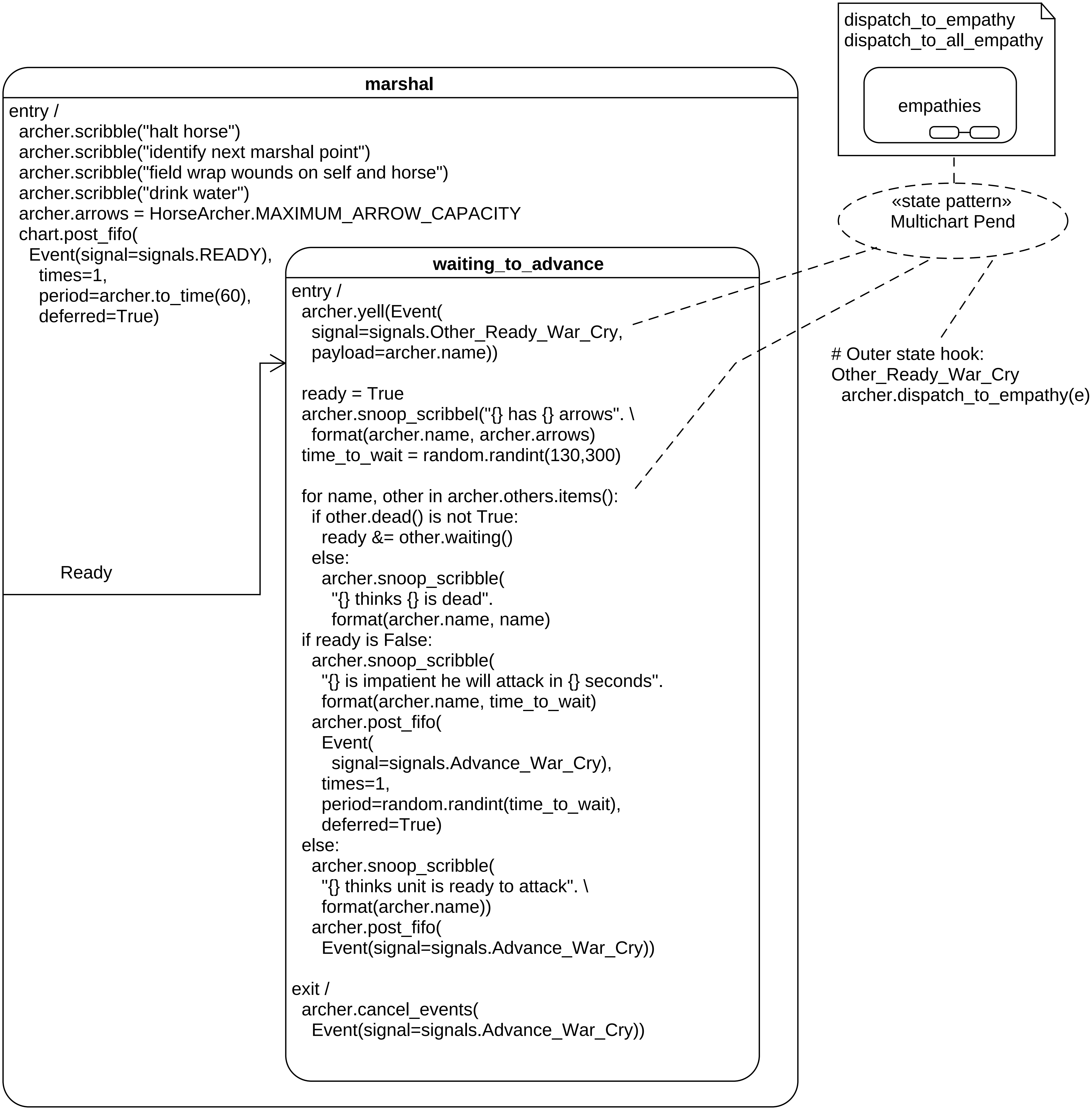

dispatch_to_empathy
dispatch_to_all_empathy

«state pattern»
Multichart Pend

```
# Outer state hook:
Other_Ready_War_Cry
  archer.dispatch_to_empathy(e)
```

marshal

```
entry /
  archer.scribble("halt horse")
  archer.scribble("identify next marshal point")
  archer.scribble("field wrap wounds on self and horse")
  archer.scribble("drink water")
  archer.arrows = HorseArcher.MAXIMUM_ARROW_CAPACITY
  chart.post_fifo(
    Event(signal=signals.READY),
      times=1,
      period=archer.to_time(60),
      deferred=True)
```

waiting_to_advance

```
entry /
  archer.yell(Event(
    signal=signals.Other_Ready_War_Cry,
    payload=archer.name))

  ready = True
  archer.snoop_scribbel("{} has {} arrows". \
    format(archer.name, archer.arrows)
  time_to_wait = random.randint(130,300)

  for name, other in 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 is False:
    archer.snoop_scribble(
      "{} is impatient he will attack in {} seconds".
      format(archer.name, time_to_wait)
    archer.post_fifo(
      Event(
        signal=signals.Advance_War_Cry),
      times=1,
      period=random.randint(time_to_wait),
      deferred=True)
  else:
    archer.snoop_scribble(
      "{} thinks unit is ready to attack". \
      format(archer.name))
    archer.post_fifo(
      Event(signal=signals.Advance_War_Cry))

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