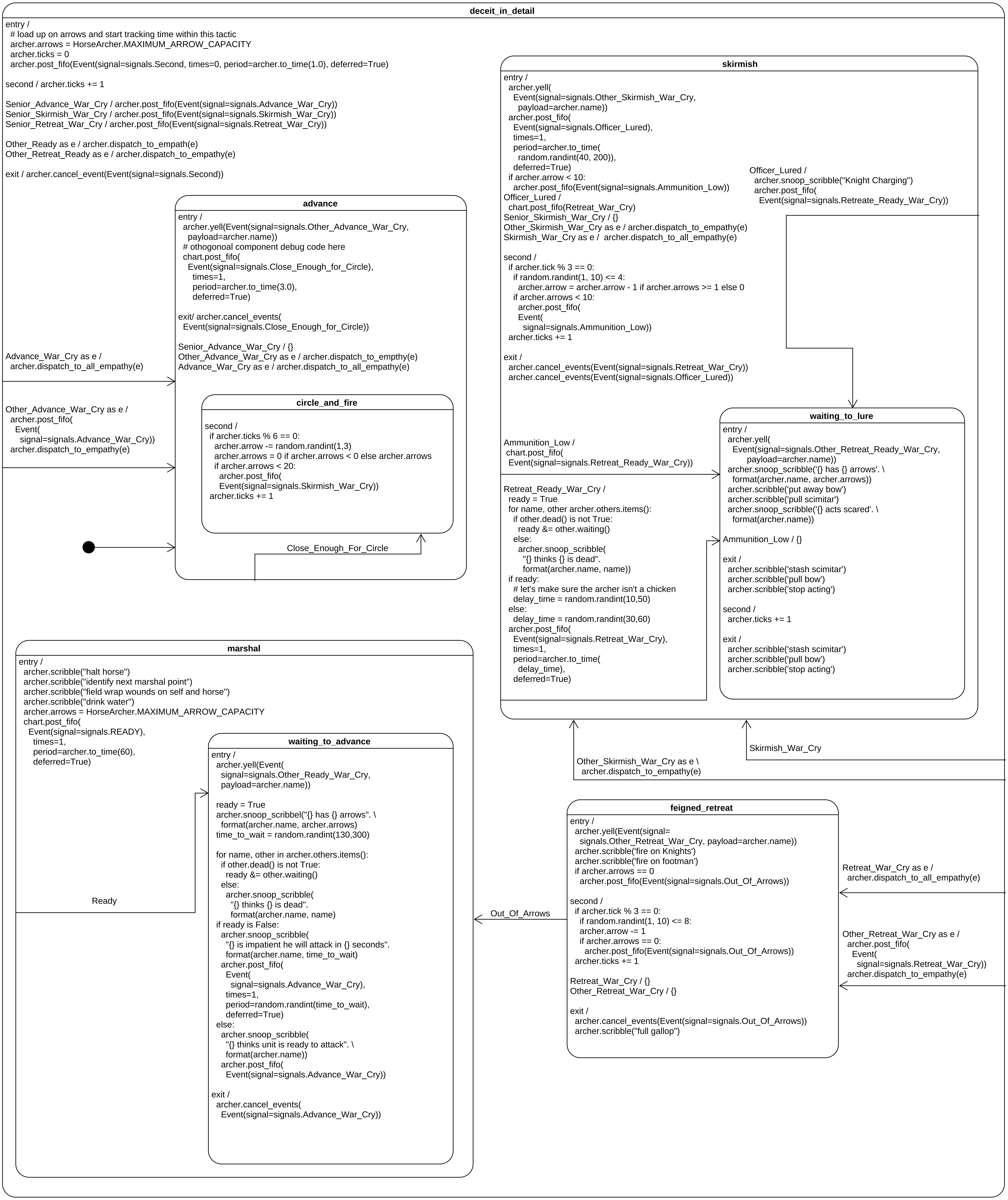

deceit_in_detail

```
entry /
  # load up on arrows and start tracking time within this tactic
  archer.arrows = HorseArcher.MAXIMUM_ARROW_CAPACITY
  archer.ticks = 0
  archer.post_fifo(Event(signal=signals.Second, times=0, period=archer.to_time(1.0), deferred=True)

second / archer.ticks += 1

Senior_Advance_War_Cry / archer.post_fifo(Event(signal=signals.Advance_War_Cry))
Senior_Skirmish_War_Cry / archer.post_fifo(Event(signal=signals.Skirmish_War_Cry))
Senior_Retreat_War_Cry / archer.post_fifo(Event(signal=signals.Retreat_War_Cry))

Other_Ready as e / archer.dispatch_to_empath(e)
Other_Retreat_Ready as e / archer.dispatch_to_empathy(e)

exit / archer.cancel_event(Event(signal=signals.Second))
```

```
Advance_War_Cry as e /
  archer.dispatch_to_all_empathy(e)
```

```
Other_Advance_War_Cry as e /
  archer.post_fifo(
    Event(
      signal=signals.Advance_War_Cry))
  archer.dispatch_to_empathy(e)
```

advance

```
entry /
  archer.yell(Event(signal=signals.Other_Advance_War_Cry,
    payload=archer.name))
  # othogonoal component debug code here
  chart.post_fifo(
    Event(signal=signals.Close_Enough_for_Circle),
      times=1,
      period=archer.to_time(3.0),
      deferred=True)

exit/ archer.cancel_events(
  Event(signal=signals.Close_Enough_for_Circle))

Senior_Advance_War_Cry / {}
Other_Advance_War_Cry as e / archer.dispatch_to_empthy(e)
Advance_War_Cry as e / archer.dispatch_to_all_empathy(e)
```

circle_and_fire

```
second /
  if archer.ticks % 6 == 0:
    archer.arrow -= random.randint(1,3)
    archer.arrows = 0 if archer.arrows < 0 else archer.arrows
    if archer.arrows < 20:
      archer.post_fifo(
      Event(signal=signals.Skirmish_War_Cry))
  archer.ticks += 1
```

Close_Enough_For_Circle

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.Retreate_Ready_War_Cry))
```

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

```
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)
```

waiting_to_lure

```
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')
```

Skirmish_War_Cry

```
Other_Skirmish_War_Cry as e \
  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)
```

Ready

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))
```

Out_Of_Arrows

feigned_retreat

```
entry /
  archer.yell(Event(signal=
    signals.Other_Retreat_War_Cry, payload=archer.name))
  archer.scribble('fire on Knights')
  archer.scribble('fire on footman')
  if archer.arrows == 0
    archer.post_fifo(Event(signal=signals.Out_Of_Arrows))

second /
  if archer.tick % 3 == 0:
    if random.randint(1, 10) <= 8:
    archer.arrow -= 1
    if archer.arrows == 0:
      archer.post_fifo(Event(signal=signals.Out_Of_Arrows))
  archer.ticks += 1

Retreat_War_Cry / {}
Other_Retreat_War_Cry / {}

exit /
  archer.cancel_events(Event(signal=signals.Out_Of_Arrows))
  archer.scribble("full gallop")
```

```
Retreat_War_Cry as e /
  archer.dispatch_to_all_empathy(e)
```

```
Other_Retreat_War_Cry as e /
  archer.post_fifo(
    Event(
      signal=signals.Retreat_War_Cry))
  archer.dispatch_to_empathy(e)
```