ActiveObject

ToasterOven

ToasterOven.Toast_Time_In_Sec = 10
ToasterOven.Bake_Time_In_Sec = 20
ToasterOven.Pre_Time_Sec = 1

toast_time_in_sec
bake_time_in_sec
get_ready_sec
history

buzz() # make a short buzzing sound
light_on()
light_off()
heater_on()
heater_off()
cook_time(time_in_sec)

ToasterOvenMock

ToasterOvenMock.prepend_trace_timestamp(string)
ToasterOvenMock.get_100ms_from_timestamp(string)
ToasterOvenMock.time_difference(time_1_string, ...)
ToasterOvenMock.instrumentation_line_of_match(...)

scribble(string) # overloading miros scribble
buzz()
light_on()
light_off()
heater_on()
heater_off()

Event Processor

Event Processor

```
get_ready_time_in_sec = time_in_sec - \
  self.get_ready_sec

self.post_fifo(
  Event(signal=signals.Get_Ready,
    payload=BuzzSpec(buzz_times=1)),
  times=1,
  period=get_ready_time_in_sec,
  deferred=True)

  self.post_fifo(
    Event(signal=signals.Done,
      payload=BuzzSpec(buzz_times=2),
    times=1,
    period=time_in_sec
    deferred=True)
```

```
BuzzSpec = namedtuple(
  'BuzzSpec', ['buzz_times'])
```

oven = ToasterOvenMock(name="oven")
oven.start_at(door_closed)

oven = ToasterOven(name="oven")
oven.start_at(door_closed)

common_features

Buzz /
  oven.buzz()

«state pattern»
Transition to History

door_closed

entry /
  oven.light_off(oven)

Collaboration

Door_Close /
  oven.trans(oven.history)

H*

door_open

entry /
  oven.light_on()

heating

entry/ oven.heater_on();
exit/ oven.heater_off();
Get_Ready as e /
  oven.post_fifo(
    Event(signal=signals.Buzz),
    times = e.payload.buzz_times,
    period = e.payload.buzz_period
    deferred=False)

Done as e /
  oven.post_fifo(
    Event(signal=signals.Buzz),
    times = e.payload.buzz_times,
    period = oven.done_buzz_period_sec
    deferred=False)

baking

entry /
  oven.history = baking
  oven.cook_time(oven.bake_time_in_sec)

exit /
  oven.cancel_events(
    Event(signal=signals.Get_Ready))
  oven.cancel_events(
    Event(signal=signals.Done))

toasting

entry /
  oven.history = toasting
  oven.cook_time(oven.toast_time_in_sec)

exit /
  oven.cancel_events(
    Event(signal=signals.Get_Ready))
  oven.cancel_events(
    Event(signal=signals.Done))

off

entry /
  oven.history= off

Door_Open

Baking

Toasting

Off