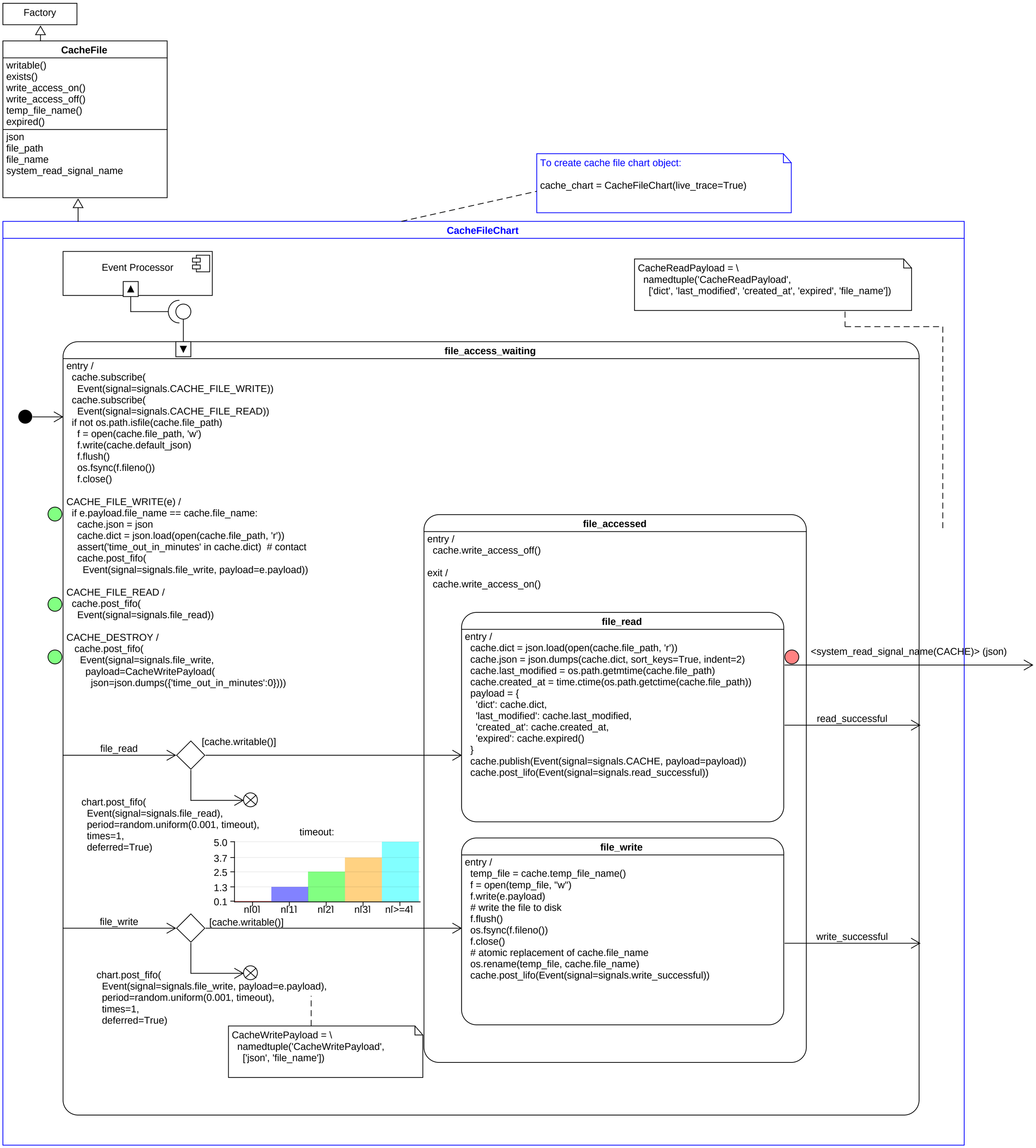
Factory
CacheFile
writable()
exists()
write_access_on()
write_access_off()
temp_file_name()
expired()
json
file_path
file_name
system_read_signal_name
To create cache file chart object:
cache_chart = CacheFileChart(live_trace=True)
CacheFileChart
Event Processor
CacheReadPayload = \
namedtuple('CacheReadPayload',
['dict', 'last_modified', 'created_at', 'expired', 'file_name'])
file_access_waiting
entry /
cache.subscribe(
Event(signal=signals.CACHE_FILE_WRITE))
cache.subscribe(
Event(signal=signals.CACHE_FILE_READ))
if not os.path.isfile(cache.file_path)
f = open(cache.file_path, 'w')
f.write(cache.default_json)
f.flush()
os.fsync(f.fileno())
f.close()
CACHE_FILE_WRITE(e) /
if e.payload.file_name == cache.file_name:
cache.json = json
cache.dict = json.load(open(cache.file_path, 'r'))
assert('time_out_in_minutes' in cache.dict) # contact
cache.post_fifo(
Event(signal=signals.file_write, payload=e.payload))
CACHE_FILE_READ /
cache.post_fifo(
Event(signal=signals.file_read))
CACHE_DESTROY /
cache.post_fifo(
Event(signal=signals.file_write,
payload=CacheWritePayload(
json=json.dumps({'time_out_in_minutes':0})))
file_accessed
entry /
cache.write_access_off()
exit /
cache.write_access_on()
file_read
entry /
cache.dict = json.load(open(cache.file_path, 'r'))
cache.json = json.dumps(cache.dict, sort_keys=True, indent=2)
cache.last_modified = os.path.getmtime(cache.file_path)
cache.created_at = time.ctime(os.path.getctime(cache.file_path))
payload = {
'dict': cache.dict,
'last_modified': cache.last_modified,
'created_at': cache.created_at,
'expired': cache.expired()
}
cache.publish(Event(signal=signals.CACHE, payload=payload))
cache.post_lifo(Event(signal=signals.read_successful))
<system_read_signal_name(CACHE)> (json)
read_successful
file_read
[cache.writable()]
chart.post_fifo(
Event(signal=signals.file_read),
period=random.uniform(0.001, timeout),
times=1,
deferred=True)
timeout:
5.0
3.7
2.5
1.3
0.1
n[0]
n[1]
n[2]
n[3]
n[>=4]
file_write
[cache.writable()]
chart.post_fifo(
Event(signal=signals.file_write, payload=e.payload),
period=random.uniform(0.001, timeout),
times=1,
deferred=True)
CacheWritePayload = \
namedtuple('CacheWritePayload',
['json', 'file_name'])
file_write
entry /
temp_file = cache.temp_file_name()
f = open(temp_file, "w")
f.write(e.payload)
# write the file to disk
f.flush()
os.fsync(f.fileno())
f.close()
# atomic replacement of cache.file_name
os.rename(temp_file, cache.file_name)
cache.post_lifo(Event(signal=signals.write_successful))
write_successful