# ACME Mining Simulation

## MineOverlord

* Observer Pattern
* Sends out ticks (5-minute intervals) that drive the simulation
* Manages a list of MineMinion objects that receive ticks
* All other objects are MineMinion objects
* Application calls MineOverlord::attach(MineMinion&) to attach MineMinions
* Simulation runs for 72 hours (864 ticks);
* One true source for timestamps (HH:MM:SS), passed to MineMinions during update()

## MineMinion

* Observer Pattern Observer
* Receives a timestamp (HH:MM:SS) string every update()
* All objects inherit from MineMinion

## MineTruck

* N MineTrucks in simulation
* N MineSites since a MineTruck never has to wait for a MineSite
* Each visit to a MineSite takes between 1 and 5 hours (12 and 60 ticks), randomly generated each visit
* Transiting between a MineSite and a MineStation takes 30 minutes (6 ticks) each way
* MineTruck states (MineTruckState):
  + MINING: Starting state. Starts when a MineTruck arrives at a MineSite, ends when the mining duration reaches 0
  + INBOUND: En route to a MiningStation, starts when mining ends, ends when the trip duration reaches 0. Receives a destination MineStation from the StationDispatcher, which assigns the MineStation with the shortest queue. The MineTruck adds itself to the MineStation queue.
  + QUEUED: In a MineStation queue, if applicable. Starts when MineTruck reaches the MineStation, ends in QSIZE – 1 ticks
  + UNLOADING: Unloading, takes 1 tick
  + OUTBOUND: En route to a MineSite. Receives a destination MineSite from the SiteDispatcher, which assigns the MineSite at the front of the MineSite queue. Starts after unloading, ends in 6 ticks.

## MineStation

* M MineStations in a simulation
* MineStations reside in the StationDispatcher’s min priority queue, sorted by their queue size; each time their queue changes size, they reinsert themselves in the priority queue
* It takes one tick to unload a MineTruck
* MineStation states (MineStationState):
  + IDLE: Starting state. No MineTrucks in queue.
  + READY: 1 or more MineTrucks in queue, but the front truck in the queue is still INBOUND; ends when the MineTruck at the front of the queue is QUEUED
  + UNLOADING: Unload, takes 1 tick per truck in the queue. Ends when the queue is empty(). Removes the MineTruck at the front of the queue, and updates the StationDispatcher.

## MineSite

* N MineSites, equal to the number of MineTrucks
* Randomly generates a visit duration (between 12 and 60 ticks) when a MineTruck arrives
* MineSites reside in the SiteDispatcher’s queue; the SiteDispatcher removes a MineSite from the front of the queue, and pushes it into the back, once a MineTruck is finished
* MineSite states (MineSiteState):
  + MINING: Starting state, MineTruck is mining, ends when the mining duration expires
  + IDLE: No MineTruck attached, MineSite is pushed into the SiteDispatcher queue

## SiteDispatcher

* Maintains a queue of idle MineSites. Initially empty, since all MineTrucks start at a MineSite
* Assigns the MineSite at the front of the queue to a MineTruck upon request

## StationDispatcher

* Maintains a min priority queue of MineStations; initially contains M MinStations with empty queues, and therefore equal priority
* When a MineTruck adds itself to a MineStation queue, the MineStation is reinserted into the priority queue, to maintain the MineStations with the shortest queue at its front

## Statistics

* For each truck:
  + For each state, get time spent in state; Turnaround = total
* For each station:
  + For UNLOADING state, get time spent in state; Utilization = (time / 72) \* 100
  + Total trucks served = total time / 5 minutes (unloading time)
* Efficiency ratio: total unloading time / turnaround time