* Introduction
  + I think you can simplify the description of the service by stating something along the lines of “Alerts are produced when an ownship is projected to enter/exit a keep out/in zone at any point from the current time up until a configured time window. “
  + Otherwise, a definition of violation of the zone is probably needed.
* Service Parameters
  + Iteration Period
    - What is meant by “…is best effort and is neither hard nor soft-real time…”?
  + Path Deviation Tolerance
    - Are you defining the currently assigned pat as a straight line between the previous waypoint and the current waypoint the vehicle is attempting to attain?
    - I still think you should always alert when a violation is projected. It would be the concern of the response to filter those messages as appropriate for the intended behavior. Additionally, though the claim is that OpenUxAS will not violate no fly zones enroute to tasks, that hasn’t been formally proven or sufficiently tested.
    - I am not sure how you are planning on computing this path deviation and then checking it in comparison to the deviation tolerance you define.
  + Look-ahead Time Window
    - I am assuming the “last reported linear trajectory” means a linear projection starting from the most recently reported vehicle stated.
      * It might be useful to add to the introduction that the service operates on ownship state messages and the resulting iteration window is resultant from 2 successive operational tics of the service.
  + Based on cases examined, it is probably prudent to define some relations between the parameters.
    - Ts + Tw should be at least X times the iteration period. This would provide some assurance that the projection is far enough ahead of the latest state update as to not miss potential violations should the iteration period have disturbances.
* Zone Definitions and Interpretation Consistency
  + Nice handling of padding to encompass vehicle dimension, hard limits, etc.
* Semantic Consistency with Other Zone Services
  + Acceptable design decision supported by argument justification.
* Only Considering Zones of a Vehicle’s Assigned Operating Regions.
  + I don’t agree with the argument presented here on all levels. If talking about keep out zones, they should be enforced irrespective of being in the vehicle allowed operating region. When talking about keep in zones, it does make sense that only those in the allowed operating region should be enforced as to the best of my knowledge, a vehicle should never have an operating region and a keep in zone that are disparate. (Though this isn’t checked…it could easily be made into a configuration requirement).
  + Additionally, the automation request validator is responsible to making sure task requests are valid in that operating regions, tasks, and vehicles have be previously defined. This is the more logical place to add a capability for enforcing the correctness of request (i.e. not only have zones been defined but the correct zone is attached to the correct set of vehicles). The zone alert service should assume that only properly formed requests are handled and thus the globally defined zones should all be enforced.
* Basic Behaviors
  + Case 1: Report Potential Zone Violations Based on Current Linear Trajectory
    - I think you should change the relationship between the times such that Ts <= Tp. Using the simulation model, there is .5s between state updates. I would anticipate not much time to actually have passed between reception of a message and then beginning to process on that message. Additionally, if the time of computation is = to Ts+Tw, it would also be problematic.
    - Past Projected Violations Are Not Reported
      * I am not sure what this section is communicating. I am not seeing the figure illustrating the two cases mentioned.
      * I want to emphasize that a model where Tp is significantly different from Ts would not be of any utility in prevention of zone violations
    - Repeating Reports
      * The behavior here is expected as described by the mission statement of the service produced in the introduction. Doesn’t need a separate section.
    - Potentially Report While Following a Route.
      * Unless you are planning on tracking performance, OpenUxAS doesn’t have a metric for path deviation already described. Using this metric to influence reporting can be done, but would add complexity to the service beyond stakeholder requirements. I think this type of processing is more appropriate for a service that responds to the alerts. Using this deviation from expected path might provide a filter for which alerts to respond to.
      * In the simulation environment, the additional messages may be burdensome on faster than real time simulations, but shouldn’t present a problem for real time computations. (Using the DAIDALUS response service as a surrogate).
  + Case 2: Violation of a Zone Boundary
    - I was expecting the service to be driven by state updates. Perhaps you are intending the service to fire off at some fixed rate that is independent of the state update rate?
    - Even the example you provide, an alert to a violation happening before Tp should be given.
    - An Impact of Computation Interval on Reporting
      * Iteration interval should not affect reporting. If there is a projected violation from Ts to Ts+Tw, a report should be generated. Here again maybe the additional constraint that the iteration time period is defined as the time between successive state updates and that Ts+Tw = X \* Iteration\_state\_update will make this section moot. Or if you want to have a separate iteration period for the service vs the state updates, Iteration\_state\_update <= Iteration\_zone\_alert
  + Case 3: Reports Failure to Be Able to Compute
    - This service should inherit the context of OpenUxAS in the simulation environment, which includes things like well-formed requests and expectations on state updates
  + Case 4: No Reporting Regarding Operating Region Boundary.
    - Expectation is that if a keep in zone is provided it should be within an operating region. Stated behavior is fine.
* Complex Behaviors
  + Statements here are true, but given a full definition of parameters should not need to be separated from Basic Behaviors.