Scheduling and Reservation Modules

The following pseudo-code represents my current thoughts on the direction to be taken for providing scheduling functionality in PWE.

```
1.  process partitions for initial start time and resource requirements
2.  for each partition:
3.     query for matching resources, prioritized {NOP when resource is specified}
4.     if no matches:
5.        go to 13
6.     for each rule (the first of which is empty):
7.           for each match:
8.             try to schedule/reserve partition
9.             if successful:
11.             go to 2
12.    if no partial scheduling allowed:
13.       clear all reservations (both in partitions and by canceling on host)
14.    break
```

NOTES:

1. There should be some delta, based on the expected latency of the algorithm itself, added to the start time of the first node; the actual start time (after adjustments through rules) should be visible for successive steps to compute each partition's required start time.
3. MODULE based on an extension point; should use hard requirements (if any) such as platform, minimum resources, etc., to check to see if these could be theoretically met by the hosts listed in our Host Information service; these should be ordered by probability of obtaining the resources required by the partition; right now we would use something simple like (available nodes / total nodes), but later we might be able to plug in something like Batch Queue Predictor.
6. We start the iteration with an empty rule: that is, try to get exactly the resources specified.
7. We prioritize rules over the resource-list: i.e, we assume it is more desirable to get an unmodified match on any available resource than to run on a given resource even with modified requirements; of course, if the resource is specified by the workflow rather than coming from a potential list, this does not apply.
8. MODULE based on an extension point: the module uses what is appropriate to determine whether something can be scheduled and/or reserve slots for it.
12.-13. If a partition fails to get scheduled, and we allow only full-workflow scheduling, then make sure we have told the batch system to cancel the reservations we hold. In any case, we back out of the algorithm at the first failure.

FURTHER POINTS:

- If partial scheduling is allowed, we will need to be able to do this for a single node (partition) at a time when each node comes up (this possibility is already provided for in PWE, but has not been fully implemented yet).
- Failure to schedule a workflow node (in the case of an entire workflow, the root nodes) should cause its state to become ATTEMPTED; such nodes "go dormant" and are awakened after a given time interval. I will modify the API to allow the user to include policies regarding timeouts and the number of possible attempts for any given workflow (but will also provide a reasonable back-off default: try after 15 twice; after 30 once; after an hour once; quit).
- Hard requirements for finding the initial matches:
    - user account on the machine
    - architecture, OS if expressed
    - minimum requirements on cpus/nodes, memory, etc. cannot exceed total available on machine (the user should set maximum on these as an upper bound that is soft)
    - any *ad hoc* property values that pertain to the host must match if given by workflow