Mini Multi-agent System

Write a ros node in c++ to simulate a multi-agent system.

- Description
  1. Create a compilable ROS package (10 points)
  2. In the first ROS node, name it as node_mini_factory_server, create a ROS topic subscriber (topic name "/agent_feedback") to receive agent feedback info. (10 points)
  3. In the second ROS node, name it as node_robot_1, create a ROS service server (topic name "/agent_task_1") to receive task from mini_factory_server. In the node_mini_factory_server ROS node, create a ROS service client (topic name "/agent_task_1") to send agent task info. (10 points)
  4. In the third ROS node, name it as node_robot_2, create a ROS service server (topic name "/agent_task_2") to receive task from mini_factory_server. In the node_mini_factory_server ROS node, create a ROS service client (topic name "/agent_task_2") to send agent task info. (5 points)
  5. Implement your node_robot_1 with at least 2 states. When started, robot is in "ready" state. After receive a task from node_mini_factory_server, it transits to "executing" state. After about 5 seconds, robot transits from "executing" to "ready". (10 points)
  6. Implement your node_robot_2 with at least 2 states. When started, robot is in "ready" state. After receive a task from node_mini_factory_server, it transits to "executing" state. After about 3 seconds, robot transits from "executing" to "ready". (5 points)
  7. Implement your node_mini_factory_server with 5 initial tasks identified with id=1,2,3,4,5; Your node_mini_factory_server will know there are 2 robots and their state after receiving robot feedback from "/agent_feedback" topic. Your node_mini_factory_server should then send task to a robot if that robot is in "ready" state. Notice that your node_mini_factory_server should only send ONE task to each robot at a time. When a robot finished a task(state transits from "executing" to "ready"), your node_mini_factory_server should send another task to that robot if not all 5 tasks are done.(25 points)
  8. Create ROS message and ROS service for your "/agent_feedback" and "/agent_task_x" (x=1,2) to fulfill above need. (10 points)
  9. Create a roslaunch that will launch all three nodes aforementioned and result should be printed onto screen clearly. The result should show each robot states, which factory task is done or pending, etc. (10 points)
  10. Create a README.md in your package to show necessary steps or caveats about your repo. (5 points)
  11. Attention. Your ROS package should be compilable under typical ROS kinetic environment. We will clone your package, compile and evaluate. Compilation error will cause a 0 score.

- Skills
  1. C++
  2. ROS
  3. Clear logics
  4. A good code structure and style is a must.

- Final Submission:
  1. Create a github repo and upload your package to a repo.
  2. Please push frequently after each step.
  3. Develop each feature in a separate branch and merge back to master after module test is a bonus.
  4. Final submission is a link to the repo.