Разработка планировщика

Легенда

В рамках некоторой системы, были выделены следующие компоненты:

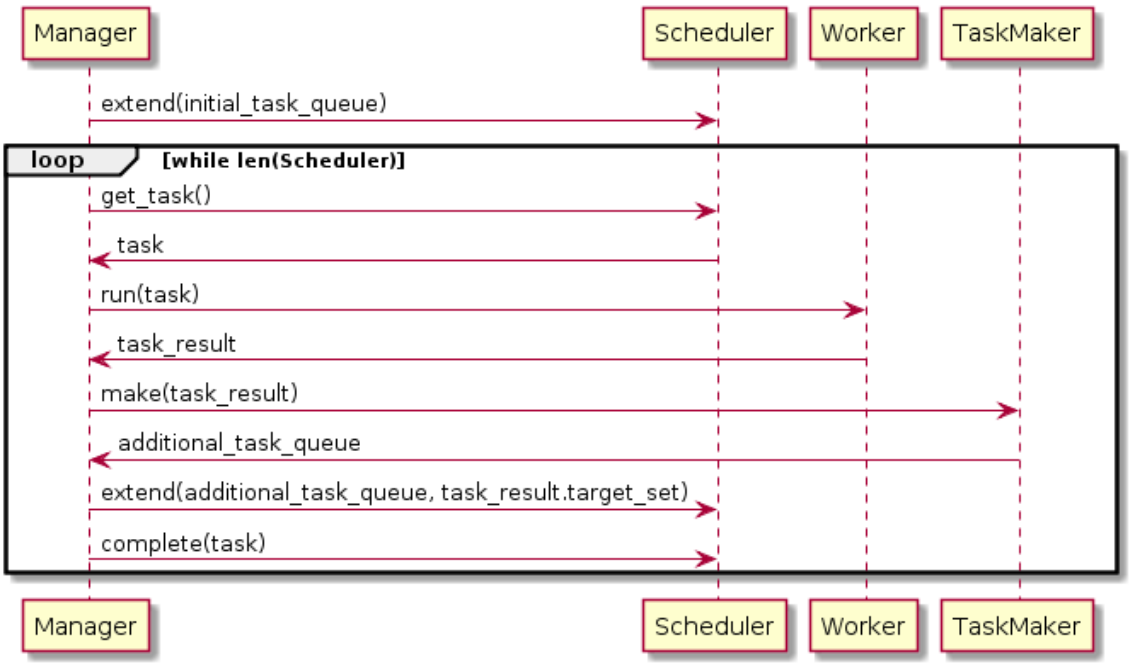
- Исполнитель - выполнение поступившей задачи
- Поставщик задач - создание новых задач
- Планировщик - упорядоченная выдача задач
- Менеджер - обеспечение взаимодействия остальных компонентов

Task - единица работы

- id - идентификатор
- priority - приоритет
- target_set - множество целей достигаемых после выполнения задачи
- read_set - множество ресурсов используемых для чтения
- write_set - множество ресурсов используемых для записи

Сценарий работы компонентов:

1. Менеджер загружает в Планировщик начальный набор задач
2. Пока есть задачи в Планировщике
    a. Менеджер получает у Планировщика очередную задачу и передаёт её Исполнителю
    b. Исполнитель выполняет задачу и передаёт результат Менеджеру
    c. Менеджер передаёт результат в Поставщик Задач
    d. Поставщик Задач создаёт дополнительные задачи и передаёт их Менеджеру
    e. Менеджер загружает в Планировщик дополнительные задачи и список достигнутых целей из результата
    f. Менеджер сообщает Планировщику что задача выполнена



Задание

В качестве задания необходимо поэтапно реализовать Планировщик задач

Каждый этап сформулирован в виде дополнения к условиям влияющим на порядок выдачи задач

Последовательность этапов

1. Базовая версия - Задачу добавленную раньше следует выполнить раньше
2. Приоритеты - Задачу с большим приоритетом (Task.priotity) следует выполнить раньше
3. Цели - Задача исключается, если хотя бы одна из её целей (Task.target_set) уже достигнута
4. Монополия - Запрещено одновременное выполнение задач, использующих одинаковый ресурс для записи (Task.write_set)
5. Конкуренция - Запрещено одновременное выполнение задачи использующей ресурс для записи (Task.write_set) и задач использующих его для чтения (Task.read_set)
6. Зависимости - Задача использующая ресурс для записи (Task.write_set) должна быть выполнена раньше, чем задача использующая этот ресурс для чтения (Task.read_set)
    a. Если есть только один Исполнитель
    b. Если есть много Исполнителей

Для этапа 6 допустимо предоставить описание решения, вместо самого решения

Реализация каждого этапа должна отдельно фиксироваться в git

Эмулятор

Для упрощения решения, вместо реальных Исполнителя, Поставщика Задач и Менеджера предоставлен их эмулятор

EmulatorTask - единица работы эмулятора

- task - объект задачи
- duration - длительность выполнения
- complete_target_set - множество целей которые будут достигнуты после выполнения
- additional_task_id_queue - список идентификаторов задач, которые нужно добавить после выполнения данной

EmulatorConfig - конфигурация эмулятора, определяющая его работу, в формате YAML

```
config:
  %task_id%:
    ?priority: 0
        ?target_set: [%target%]
        ?read_set: [%resource%]
        ?write_set: [%resource%]
    ?duration: 0
    ?complete_target_set: [%target%]
      ?additional_task_id_queue: [%task_id%]

initial_task_id_queue: [%task_id%]
?thread_count: 1
```

Пример конфигурации

```
config:
  1:
    pririty: 100500
    duration: 10
    target_set: [goal_a]
          read_set: [file_a]
    write_set: [file_b]
  2:
    additional_task_id_queue: [1]
  3:
    pririty: 3
    complete_target_set: [goal_a]

initial_task_id_queue: [2, 3]
```