The architecture depicted in the diagram is a high-level design for a cloud-based task scheduling and resource management system, utilizing a reinforcement learning (RL) approach specifically employing the Qwhale or SARSAwhale algorithm. This architecture is structured to optimize the scheduling of tasks across a cloud infrastructure, ensuring efficient resource utilization and minimizing the overall Makespan, which represents the total time required to complete all submitted tasks.

At the core of this system is the **Hybrid RL Scheduler**, which leverages either Qwhale or SARSAwhale, both advanced RL algorithms. These algorithms are tailored to handle dynamic and complex task scheduling scenarios in cloud environments. Qwhale is a variant of Q-learning, a model-free RL algorithm that operates by learning a policy to maximize the cumulative reward by taking the best action based on the current state. SARSAwhale, on the other hand, is a variation of the SARSA algorithm, which stands for State-Action-Reward-State-Action, and it follows an on-policy learning approach, updating the policy based on the action actually taken rather than the optimal action.

Tasks are submitted by users through the **System Access Interface**, which serves as the entry point to the system. These tasks are then passed to the **Task Manager**, a central component responsible for monitoring task statuses, tracking available resources, and balancing the load across the system. The Task Manager is equipped with a **Task Monitor** to keep track of task execution, a **Resource Monitor** to manage the available computational resources, and a **Load Balancer** to distribute tasks evenly across these resources, ensuring no single resource is overburdened.

The **Hybrid RL Scheduler** interacts closely with the **Resource Provisioning** module, which is responsible for allocating the appropriate resources—such as virtual machines (VMs) within a **Data Center**—to the tasks based on the scheduler's decisions. This interaction is crucial as it allows for real-time adjustments to resource allocation, ensuring tasks are executed efficiently across the available VMs (VM1, VM2, …, VMn).

Once tasks are scheduled and resources provisioned, they are executed within the Data Center. The performance of this entire process is measured by the **Makespan**, which represents the total time taken to complete all the tasks. The system's primary goal is to minimize the Makespan, thereby improving overall efficiency and throughput.

This architecture, with its integration of Qwhale/SARSAwhale algorithms, represents a sophisticated approach to cloud task scheduling, emphasizing the importance of reinforcement learning in optimizing resource management and task execution in cloud computing environments. This system not only enhances resource utilization but also reduces operational costs and improves the responsiveness of cloud services by minimizing task completion times

**\*Components in next page**

.

**Components:**

1. **System Access Interface**:
   * This is the entry point for users (User 1, User 2, …, User n) to submit tasks to the system. The interface collects and forwards tasks to the Task Manager.
2. **Task Manager**:
   * **Task Monitor**: Monitors the status of tasks that have been submitted.
   * **Resource Monitor**: Keeps track of the available resources in the system.
   * **Load Balancer**: Distributes tasks evenly across resources to ensure optimal performance.
   * **Task List**: Represents the collection of tasks (T1, T2, T3, …, Tn) that are currently managed by the Task Manager.
3. **Hybrid RL Scheduler**:
   * This component utilizes a hybrid reinforcement learning approach to optimize the scheduling of tasks. It interacts with both Resource Provisioning and the Task Manager to allocate tasks efficiently.
4. **Resource Provisioning**:
   * Allocates resources (e.g., virtual machines) based on the tasks' requirements and the scheduling decisions made by the Hybrid RL Scheduler.
5. **Data Center**:
   * Contains Virtual Machines (VM1, VM2, …, VMn) that execute the scheduled tasks. The VMs are managed and provisioned based on the scheduling and resource provisioning decisions.
6. **Makespan**:
   * Represents the total time taken to complete all the tasks. It is an important metric for evaluating the efficiency of the scheduling algorithm.

**Data Flow:**

1. **Task Submission**:
   * Users submit tasks through the System Access Interface, which then forwards them to the Task Manager.
2. **Task Management**:
   * The Task Manager monitors tasks, resources, and load balancing, ensuring that tasks are ready to be scheduled.
3. **Scheduling and Resource Provisioning**:
   * The Hybrid RL Scheduler interacts with Resource Provisioning to allocate the necessary resources for each task.
   * Resource Provisioning then allocates the required resources (e.g., VMs) within the Data Center.
4. **Task Execution**:
   * Tasks are executed on the allocated VMs in the Data Center. The execution time contributes to the overall Makespan.
5. **Makespan Calculation**:
   * The system monitors the completion of tasks to calculate the Makespan, which is used to assess the performance of the scheduling process.

This architecture seems designed to optimize task scheduling and resource allocation using a reinforcement learning approach, potentially improving performance metrics like Makespan and resource utilization.