The communication interface that the VMM and the DS provide.

1. Communication between the AC and the DS

- AC_REGISTER_NEW_DS
  - When the AC registers to the DS for allocating a new VM.
  - Input: (byte command, long userId, int qosFlag)
    - command: RapidMessages.AC_REGISTER_NEW_DS as a byte value.
    - userId: -1 for the first use, otherwise the user ID that the AC has previously.
    - qosFlag: 0 if there is no QoS parameter, otherwise 1. *For the demo, it should be 0.*
  - Output: (byte status, long userId, Object ipList)
    - status: RapidMessages.OK for success, otherwise RapidMessages.ERROR.
    - userId: A new user ID issued by the DS.
    - ipList: The IP address array of available physical machines (ArrayList<String>).
- AC_REGISTER_PREV_DS
  - When the AC registers to the DS for allocating its previous VM.
  - Input: (byte command, long userId, int qosFlag)
    - command: RapidMessages.AC_REGISTER_PREV_DS as a byte value.
    - userId: The existing user ID that the AC has previously.
    - qosFlag: 0 if there is no QoS parameter, otherwise 1. *For the demo, it should be 0.*
  - Output: (byte status, long userId, UTF ipAddress)
    - status: RapidMessages.OK for success, otherwise RapidMessages.ERROR.
    - userId: The existing user ID that the AC has.
    - ipAddress: The IP address of the physical machine that has the VM of this client.

2. Communication between the AS and the DS

- AS_RM_REGISTER_DS
  - When the AS registers to the DS for retrieving its VM ID.
  - Input: (byte command, long userId)
    - command: RapidMessages.AS_RM_REGISTER_DS as a byte value.
    - userId: The user ID received by the AS_RM_REGISTER_VMM message. Please see Section 3.
  - Output: (byte status, long vmId)
    - status: RapidMessages.OK for success, otherwise RapidMessages.ERROR.
    - vmId: The VM ID issued by the DS for this AS.
- FORWARD_REQ
  - When the AS wants to forward its job to a helper VM. *The message will not return until there is an available resource.*
  - Input: (byte command, long vmId)
    - command: RapidMessages.FORWARD_REQ as a byte value.

        - ❖ vmId: The VM ID received by the AS_RM_REGISTER_DS message.
    - Output: (byte status, UTF ipAddress)
        - ❖ status: RapidMessages.OK for success, otherwise RapidMessages.ERROR.
        - ❖ ipAddress: The internal IP address of an available helper VM starting with "10.0."
- FORWARD_START
    - When the AS wants to notify the DS with the start of the forwarded job execution.
    - Input: (byte command, long vmId)
        - ❖ command: RapidMessages.FORWARD_START as a byte value.
        - ❖ vmId: The VM ID of this AS.
- FORWARD_END
    - When the AS wants to notify the DS with the end of the forwarded job execution.
    - Input: (byte command, long vmId)
        - ❖ command: RapidMessages.FORWARD_END as a byte value.
        - ❖ vmId: The VM ID of this AS.
- PARALLEL_REQ
    - When the AS wants to parallelize its job to multiple helper VMs. *The message will not return until there are available resources.*
    - Input: (byte command, long vmId, int helperVmNum)
        - ❖ command: RapidMessages.PARALLEL_REQ as a byte value.
        - ❖ vmId: The VM ID received by the AS_RM_REGISTER_DS message.
        - ❖ helperVmNum: The number of helper VMs that the AS wants to utilize.
    - Output: (byte status, Object ipList)
        - ❖ status: RapidMessages.OK for success, otherwise RapidMessages.ERROR.
        - ❖ ipList: The IP address array of available helper VMs (ArrayList<String>) each of which starts with "10.0."
- PARALLEL_START
    - When the AS wants to notify the DS with the start of the parallelized job execution.
    - Input: (byte command, long vmId)
        - ❖ command: RapidMessages. PARALLEL_START as a byte value.
        - ❖ vmId: The VM ID of this AS.
- PARALLEL_END
    - When the AS wants to notify the DS with the end of the parallelized job execution.
    - Input: (byte command, long vmId)
        - ❖ command: RapidMessages. PARALLEL_END as a byte value.
        - ❖ vmId: The VM ID of this AS.

3. Communication between the AS and the VMM

- AS_RM_REGISTER_VMM

- When the AS registers to the VMM for knowing the user ID of a client. *The AS should send this message to the VMM when it is both started and resumed.* Please refer to the Figure 5 of D3.3.
- Input: (byte command, UTF ipAddress)
  - command: RapidMessages.AS_RM_REGISTER_VMM as a byte value.
  - ipAddress: The IP address of the AS. *Please see the relevant email to extract the exact IP address.*
- Output: (byte status, long userId)
  - status: RapidMessages.OK for success, otherwise RapidMessages.ERROR.
  - userId: The user ID associated with this AS.

- AS_RM_NOTIFY_VMM
  - When the AS wants to suspend or stop its VM. *If the AS wants to stop its VM, the VM will be removed from the Openstack instances and the corresponding data will be erased.*
  - Input: (byte command, long userId, int actionType)
    - command: RapidMessages.AS_RM_NOTIFY_VMM as a byte value.
    - userId: The user ID received by the AS_RM_REGISTER_VMM message.
    - actionType: 1 if the AS wants to suspend, or 2 if it wants to stop.

4. Communication between the SLAM and the VMM

- SLAM_START_VM_VMM
  - When the SLAM starts a new VM or resumes the existing VM.
  - Input: (byte command, long userId, int vmType, int osType)
    - command: RapidMessages.SLAM_START_VM_VMM as a byte value.
    - userId: The user ID received by the client.
    - vmType: A virtual machine type, which is changed from the QoS parameter by the SLAM. *It should be 0 for the demo.*
    - osType: The type of OS to launch. Linux if 0 or Android if 1. Types can be added in the RapidConstants.Java
  - Output: (byte status, long userId, UTF ipAddress)
    - status: RapidMessages.OK for success, otherwise RapidMessages.ERROR.
    - userId: The user ID of the client.
    - ipAddress: The IP address of the created or resumed VM.