

Orchestratori

In startInternal of SimulationManager there is a check if enable_orchestrators=false, in that case the device is elected orchestrator.

Dopo di che si schedula il task con tag **SEND_TO_ORCH**

In ProcessEvent the case **SEND_TO_ORCH** is taken and sendTaskToOrchestrator is called

- Check if the task is failed

- Se gli orchestratori sono abilitati

- Scorro la lista degli orchestratori (costruita in ServersManager)

- Ogni datacenter del tipo specificato da DEPLOY_ORCHESTRATOR nel metodo selectOrchestrators della classe ServersManager e' aggiunto nella lista degli orchestratori.

- Escludo gli orchestratori Cloud e cerco quello più vicino

- Se la lista fosse vuota termino

- Assegno l'orchestratore assegnato

- Schedulo immediatamente al networkModel il task con tag **SEND_REQUEST_FROM_DEVICE_TO_ORCH**

In NetworkModel the processEvent case **SEND_REQUEST_FROM_DEVICE_TO_ORCH** is caught and sendRequestFromDeviceToOrch is called

- Verifica se l'orchestratore del task e' il device stesso

- Se così non fosse inizia il trasferimento dei files del task come REQUEST

- Il trasferimento dei files segue l'iter del NetworkModel fino ad arrivare al metodo transferFinished

- Se il trasferimento è una REQUEST chiamo offloadingRequestReceivedByOrchestrator

- Controlla se l'orchestratore è di tipo Cloud

- Se così fosse schedula al SimulationManager il task con delay WAN_PROPAGATION_DELAY e tag

- SEND_TASK_FROM_ORCH_TO_DESTINATION**

Se non fosse Cloud schedula immediatamente senza
WAN_PROPAGATION_DELAY

Altrimenti manda la richiesta direttamente alla destinazione (se stesso) e
schedula immediatamente per il SimulationManager con tag
`SEND_TASK_FROM_ORCH_TO_DESTINATION`

In ProcessEvent the case `SEND_TASK_FROM_ORCH_TO_DESTINATION` is taken
and `sendFromOrchToDestination` is called

there is a check on the failure of the task

then `edgeOrchestrator.initialize(task)` is called and a suitable VM is chosen

`initialize` of Orchestrator is called and, depending on the architecture of
the simulation, the appropriate method that insert a string corresponding
to the architecture in a String array

Then `assignTaskToVm` is called on `findVM`

`findVM` is abstract and must be implemented by the class that
extends orchestrator

In `DefaultEdgeOrchestrator` `findVM` utilize 2 algorithms
(`ROUND_ROBIN` and `TRADE_OFF`)

In `CustomEdgeOrchestrator` è presente solo l'algoritmo
`INCREASE_LIFETIME`

In entrambi viene controllato se l'offloading è possibile
(`offloadingIsPossible` di Orchestrator)

Si controlla l'architettura è Cloud e la VM è Cloud (l'offload è
sempre possibile)

Si controlla se l'architettura è Edge e la VM è Edge

In questo caso si verifica anche se:

Il dispositivo che genera il task e l'Edgedatacente della VM
sono nel raggio d'azione di ques'ultimo

Oppure gli orchestratori sono abilitati e l'orchestratore del
task è nel range dell'Edgedatacente della VM

Si controlla se l'architettura è Mist e la VM è di un `EdgeDevice`

Se così fosse si controlla che il device della VM non sia
morto

E

O Il datacenter della VM (un edge device) deve trovarsi nel raggio del device che genera il task

Oppure l'orchestrazione è attiva e il device che genera il task si trova nel raggio di azione dell'EdgeDataCenter della VM

The found VM (if any, otherwise if (vmlIndex == -1) {
simLog.incrementTasksFailedLackOfResources(task); }) is assigned to the task

Dopo l'initialize si fa un controllo se la VM è null

Se così fosse incrementTasksFailedLackOfResources e returno

QUI POTREI INSERIRE LO SWITCH A CLOUD SE L'ARCHITETTURA
FOSSE EDGE

If the task is offloaded **and** the orchestrator is not the offloading destination

Schedule Now on the networkmodel with

SEND_REQUEST_FROM_ORCH_TO_DESTINATION the task

NetworkModel calls sendRequestFromOrchToDest that begins the transfer of files with the tag TASK

Upon completing transferring executeTaskOrDownloadContainer is called

Here is checked whether:

- The ENABLE_REGISTRY in parameters is true

and

- The registry mode is Cloud

and

- The orchestrator isn't offloaded on the Cloud

If is it the above case a DOWNLOAD_CONTAINER is scheduled immediately

addContainer case is switched and a file transfer with tag CONTAINER is added to transferProgressList

upon finishing to download the container in transferFinished containerDownloadFinished is called and the task with tag

EXECUTE_TASK is immediately scheduled to the SimulationManager
Otherwise

If the task is offloaded on the cloud an EXECUTION_TASK is
scheduled to SimulationManager with a WAN_PROPAGATION_DELAY
delay

else the task is immediately executed on the simulation manager

Else schedule now the execution of the task on this (the Simulation Manager)
with the tag EXECUTE_TASK

The case EXECUTE_TASK is switched

There is a check for failure

Chiama submitCloudlet del broker ottenuto con getBroker()

In DatacenterBrokerAbstract.java nel metodo processCloudletFinish
c'e' un controllo nel quale se le istruzioni effettuate fin'ora sono 0 allora il
task dovrebbe risultare fallito per mancanza di risorse

viene chiamato

cloudlet.getVm().getCloudletScheduler().cloudletFail(cloudlet); ma
non riesco a trovare un'implementazione

Il broker crea una lista e ci aggiunge il Cloudlet

Chiama submitCloudletList (con un solo elemento?)

La lista viene aggiunta a cloudletWaitingList
(this.cloudletWaitingList.addAll(list);)

Se il broker e' started allora chiamo
requestDatacentersToCreateWaitingCloudlets

Iterando su tutta la lista:

Verifico se la VM sia creata (perche' l'ultima?)

Se cosi' non fosse postpongo l'esecuzione

Altrimenti rimuovo "FreePESNumber" (dovrebbero essere i cores)

Imposta l'ultima VM selezionata al Cloudlet

Faccio una send con argomento la VM, il submission delay (non so a
come venga settato), un tag e il cloudlet stesso

Aggiungo al delay il delay del network

Schedulo col delay trovato

aggiungo il cloudlet alla lista dei cloudlet creati

Stampo un log

aggiungo l'utilizzo della cpu

aggiorno l'uso dell'energia

In CloudletAbstract di CloudSim c'e' il metodo `returnToBrokerIfFinished` che schedula `CLOUDLET_RETURN`, non ho approfondito

CustomBroker una volta che la riceve schedula immediatamente un evento `TRANSFER_RESULTS_TO_ORCH` destinato al SimulationManager

The case `TRANSFER_RESULTS_TO_ORCH` is taken

- the CPU utilization is removed and the method `sendResultsToOchestrator` is called

- check if the task failed

- compare the id of the device that generated the device and the id of the device that owns the VM

- if they are different then the result must be sent to the orchestrator

- an event is scheduled immediately to the networkmodel with the tag `SEND_RESULT_TO_ORCH`

- the case is switched and the method `sendResultFromDevToOrch` is called

- if the orchestrator of the task isn't the one that generated it a file transfer is started with tag `RESULTS_TO_ORCH`

- on transferFinished the `RESULTS_TO_ORCH` is caught and `returnResultToDevice` is called

- if the orchestrator is the Cloud or the DataCenter of the VM is the Cloud an event with tag

- `SEND_RESULT_FROM_ORCH_TO_DEV` and delay `WAN_PROPAGATION_DELAY` is scheduled

- else the same event is scheduled without delay

- else is scheduled immediately an event `SEND_RESULT_FROM_ORCH_TO_DEV` to this (NetworkModel)

The event `SEND_RESULT_FROM_ORCH_TO_DEV` is caught, as always, in `processEvent` of `NetworkModel` and `sendResultFromOrchToDev` is called

a file transfer is initialized with tag `RESULTS_TO_DEV`
on `transferFinished` no branch match and the final `else` is caught

`resultsReturnedToDevice` is called where is scheduled immediately an event to `SimulationManager` with tag `RESULT_RETURN_FINISHED`

if they are the same (no offload) an event of `RESULT_RETURN_FINISHED` is immediately scheduled to this (`SimulationModel`)

The switch for the tag `RESULT_RETURN_FINISHED` is caught

check for failure

```
this.edgeOrchestrator.resultsReturned(task);
```

This method is customizable

the counter of tasks is increased