# Sistemi Distribuiti e Cloud Computing - A.A. 2021/22
# Progetto A1: Funzioni serverless nel Compute Continuum

Docente: Valeria Cardellini
Dipartimento di Ingegneria Civile e Ingegneria Informatica
Università degli Studi di Roma "Tor Vergata"
cardellini@ing.uniroma2.it

## Requisiti del progetto

Lo scopo del progetto è realizzare in un linguaggio di programmazione a scelta tra Python e Go un sistema distribuito che permetta l'esecuzione di funzioni serverless in ambito edge e in ambito cloud, ovvero nel cosiddetto Compute Continuum [1]. Si consideri un server che supporta l'esecuzione di funzioni serverless in uno scenario di edge computing [2], di seguito denominato *nodo edge*, essendo localizzato in prossimità degli utenti e dotato di risorse computazionali e di storage tipicamente limitate. L'obiettivo del progetto è realizzare un nodo edge che sia in grado di eseguire localmente funzioni serverless e di decidere se eventualmente effettuarne l'offloading verso un servizio Cloud di serverless computing, allo scopo di migliorare la qualità di servizio (QoS) percepita dagli utenti (ad es., il tempo di risposta sperimentato dagli utenti nel caso di funzioni computazionalmente onerose oppure di funzioni latency-sensitive).

Il sistema deve soddisfare i requisiti elencati di seguito.

- Il nodo edge esegue il codice relativo alla funzione serverless invocata dall'utente all'interno di un container Docker o di una micro VM (ad es. Firecracker [5]). Terminata la funzione, il container (o la microVM) può rimanere in uno stato di warm per un tempo limitato, in modo da ridurre il fenomeno dei cold start [4]. Esso si verifica quando, prima di poter eseguire la funzione, il nodo edge deve ottenerne il codice, istanziare un nuovo container ed avviare l'ambiente di esecuzione.

- Il nodo edge utilizza una *politica di decisione* che permette di stabilire se eseguire localmente la funzione oppure demandarne l'esecuzione (ovvero effettuare l'*offloading* della funzione) ad un servizio Cloud di serverless computing, interagendo con esso per l'invocazione della funzione e la gestione dell'output verso l'utente (ovvero il nodo edge agisce come un intermediario o proxy tra l'utente ed il servizio Cloud). Si richiede quindi l'implementazione delle funzioni sia localmente sul nodo edge, sia come funzioni supportate dal servizio AWS Lambda [3]. La politica di decisione implementata deve tenere conto di almeno due tra i seguenti fattori: parametro in input alla funzione, stato di carico del nodo edge, capacità computazionale del nodo edge, stima del tempo di cold start della funzione, stima del tempo di esecuzione della funzione. Il tempo di esecuzione della funzione serverless nel Cloud dipende generalmente dalla quantità di memoria associata alla funzione (e quindi dalla quantità di risorse che vengono assegnate al container o microVM che esegue la funzione).

- Si richiede di implementare due differenti funzioni (di tipo CPU-intensive, memory-intensive o I/O-intensive, ovvero che usano in modo intensivo CPU, memoria o I/O), che richiedono un carico di lavoro configurabile tramite un parametro di input della funzione. Esempi di funzioni includono: il

calcolo dei primi $N$ numeri primi, dove $N$ è il parametro in ingresso (ad es. se $N = 100$, la funzione restituisce 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97); il calcolo del valore di $\pi$ (ad es. con il metodo Montecarlo) con un certo grado di accuratezza, dove il grado di accuratezza è il parametro in ingresso; la manipolazione (ad es. il ridimensionamento) di un'immagine, il cui file oppure l'URL da cui effettuarne il download viene fornito come parametro di input; il riconoscimento di un volto o di un oggetto all'interno di una foto, il cui file oppure l'URL da cui effettuarne il download viene fornito come parametro di input.

- *Obbligatorio per gruppi composti da 3 studenti:* Supportare la *migrazione* di una funzione in esecuzione verso un altro nodo edge disponibile nel sistema, sfruttando i meccanismi di migrazione supportati dall'ambiente di virtualizzazione scelto ed implementando una *politica di migrazione*, che permetta ad un nodo di decidere se completare localmente l'esecuzione di una funzione oppure effettuarne la migrazione (ad es. perché il nodo edge sta diventando sovraccarico e quindi il tempo di completamento della funzione potrebbe aumentare in modo eccessivo, non permettendo quindi di soddisfare il requisito di QoS richiesto dall'utente).

- *Opzionale* Per aumentare la scalabilità del sistema, integrare un servizio di load balancing che permetta di utilizzare più nodi edge che eseguono le funzioni serverless.

Si richiede di effettuare il testing del sistema realizzato utilizzando le due funzioni serverless implementate e al variare di 3 diversi scenari di carico di lavoro del nodo edge (carico leggero, medio e pesante, che può essere ottenuto variando il parametro di input della funzione e/o il numero di richieste concorrenti inviate dagli utenti al nodo edge) e di riportare i risultati dei relativi tempi di risposta misurati lato utente. Per effettuare il testing, a seconda delle informazioni usate dalla politica di decisione implementata, è possibile anche variare la quantità di risorse a disposizione del nodo edge e/o utilizzare un tool (ad es. `stress` in Linux) per generare un carico artificiale sul nodo edge.

Si progetti il sistema ponendo particolare cura al soddisfacimento dei requisiti sopra elencati. Si richiede inoltre che gli eventuali parametri relativi al sistema e al suo deployment siano configurabili.

È possibile usare librerie e tool di supporto allo sviluppo del progetto, non sovrapposti con gli scopi del progetto; le librerie ed i tool usati devono essere esplicitamente indicati e brevemente descritti nella relazione.

## Scelta e consegna del progetto

Il progetto è dimensionato per essere realizzato da **2 o 3** studenti. Essendo un progetto di tipo di A, il voto del progetto peserà il 50% della valutazione complessiva dell'esame.

Per poter sostenere l'esame nell'A.A. 2021/22, **entro il 26/8/2022** è necessario prenotarsi per il progetto, comunicando alla docente in una email avente come oggetto **[SDCC scelta progetto]** le seguenti informazioni:

- nome, cognome e numero di matricola dei componenti del gruppo;

- progetto scelto.

Nel caso in cui il numero di prenotazioni per il progetto scelto abbia raggiunto la soglia massima prevista, sarà necessario effettuare una nuova scelta tra i progetti ancora disponibili.

Eventuali modifiche nella scelta del progetto devono essere tempestivamente comunicate alla docente e con lei concordate. Non è possibile cambiare in corso di svolgimento la tipologia del progetto (ad es. passare da progetto di tipo B a progetto di tipo A).

Per ogni comunicazione via email è necessario specificare **[SDCC]** nell'oggetto della email. Il progetto è valido **solo** per l'A.A. 2021/22 e deve essere consegnato **entro il 30/11/2022**. La prova d'esame scritta deve essere superata **entro la sessione autunnale 2021/22** (appelli di settembre 2022).

La consegna del progetto deve avvenire almeno 5 giorni lavorativi prima della data (da concordare con la docente) in cui si intende sostenere la discussione del progetto. La consegna del progetto consiste nell'invio di una email alla docente avente come oggetto **[SDCC consegna progetto]** e con il corpo della mail contenente un link a spazio di Cloud storage o repository (e.g., Dropbox, Google Drive, GitHub) che contiene:

1. il codice sorgente (opportunamente commentato);
2. relazione (in formato pdf), senza codice;
3. breve howto per l'installazione, la configurazione e l'esecuzione del sistema.

La relazione contiene:

- la descrizione dettagliata dell'architettura del sistema e delle scelte progettuali effettuate, opportunamente motivate;
- la descrizione dell'implementazione realizzata;
- la descrizione delle eventuali limitazioni riscontrate;
- l'indicazione della piattaforma software usata per lo sviluppo del sistema (incluse eventuali librerie).

Si consiglia di redarre la relazione in forma di articolo scientifico di lunghezza massima pari a 5 pagine, usando il formato ACM proceedings (`https://www.acm.org/publications/proceedings-template`) oppure il formato IEEE proceedings (`https://www.ieee.org/conferences_events/conferences/publishing/templates.html`).

## Valutazione del progetto

I principali criteri di valutazione del progetto saranno:

1. rispondenza ai requisiti;
2. originalità;
3. organizzazione del codice (leggibilità, modularità, ...);
4. organizzazione, chiarezza e completezza della relazione.

## Riferimenti bibliografici


[1] M. Aldinucci, N. Bombieri, P. Dazzi, B. Di Martino, and M. Fazio. Compute continuum, nuove opportunità di calcolo efficiente e pervasivo: sfide e vantaggi. *Agendadigitale.eu*, 2021. https://www.agendadigitale.eu/infrastrutture/compute-continuum-nuove-opportunita-di-calcolo-efficiente-e-pervasivo-sfide-e

[2] M. S. Aslanpour, A. N. Toosi, C. Cicconetti, B. Javadi, P. Sbarski, D. Taibi, M. Assuncao, S. S. Gill, R. Gaire, and S. Dustdar. Serverless edge computing: Vision and challenges. In *2021 Australasian Computer Science Week Multiconference*, ACSW '21, 2021. https://qmro.qmul.ac.uk/xmlui/bitstream/handle/123456789/70411/Gill%20Serverless%20Edge%20Computing%202021%20Accepted.pdf.

[3] AWS Lambda, 2022. https://aws.amazon.com/lambda/.

[4] P. Castro, V. Ishakian, V. Muthusamy, and A. Slominski. The rise of serverless computing. *Commun. ACM*, 62(12):44–54, Nov. 2019. https://www.researchgate.net/profile/Vatche_Isahagian/publication/337429660_The_rise_of_serverless_computing/links/6157be764a82eb7cb5e24b02/The-rise-of-serverless-computing.pdf.

[5] Secure and fast microVMs for serverless computing, 2022. https://firecracker-microvm.github.io/.