Questo breve documento contiene semplicemente qualche idea volutamente descritta in modo sintetico. Il lavoro effettivo verrà poi concordato con lo studente in base agli interessi specifici e ai tempi d'esecuzione. La tesi potrà essere scritta/svolta in inglese, ma per semplicità le tracce proposte sono in italiano.

Lista aggiornata al 21/05/2019

Analisi e valutazione delle caratteristiche di diversi ambienti per container

Anche se molto spesso si tende a "confondere" container e Docker, ovvero uno dei principali motori oggi disponibili per l'esecuzione e la gestione dei container, oggi esistono varie soluzioni che in forme e modi diversi supportano i container. Ad esempio, ma la lista non è esaustiva, Cloud Native Application Bundles (CNAB), Balena, Firecracker, Kata, Moby, open container e Singularity sono alcune delle soluzioni oggi esistenti. La tesi avrebbe quindi l'obiettivo di studiare, provare ed organizzare le diverse soluzioni al fine di creare un'analisi precisa e dettagliata dello stato dell'arte, identificare i maggiori problemi rimasti aperti e definire possibili linee guida per l'evoluzione dei container e per la ricerca futura.

Gestione dinamica di risorse in ambienti di computazione eterogenei

Da anni ormai ci occupiamo della gestione dinamica delle risorse per l'esecuzione di sistemi software. Gli ultimi risultati ottenuti riguardano Spark e la gestione "intelligente" di container, abilitando la loro scalabilità sia in orizzontale (nuove istanze) sia in verticale (nuove risorse assegnate alle istanze esistenti). Il lavoro fatto ha sempre riguardato infrastrutture di calcolo omogenee, mentre oggi in molti contesti si adottano soluzioni eterogenee (cloud, edge, HPC e magari anche IoT) e la gestione delle risorse si complica. La tesi si propone quindi di continuare/estendere i lavori fatti considerando infrastrutture eterogenee e nello specifico la gestione dinamica delle risorse per infrastrutture per il calcolo parallelo (HPC).

Framework model-driven per applicazioni IoT

Ogni framework per applicazioni IoT (ad esempio, UML IoT, AndroidThings, NodeRED) ha la propria notazione, il proprio runtime ed il proprio linguaggio di programmazione. Una notazione unica ed un approccio model-driven aiuterebbero a semplificare il problema e consentirebbero al progettista/programmatore di concentrarsi sul cuore dell'applicazione e non sui problemi relativi alla sua implementazione. In questo modo si darebbe anche la possibilità di fare facili sperimentazioni con tecnologie diverse. La tesi dovrebbe studiare le soluzioni disponibili, realizzare un meta-modello unificato, ovvero un'unica notazione di progetto, e realizzare le trasformazioni richieste per generare il codice richiesto dai framework target a partire dal modello (istanze del meta-modello) dell'applicazione di interesse.

Ambiente di emulazione/test di applicazioni Android su container

Il lavoro affronta il problema di parallelizzare l'esecuzione di app Android su dispositivi diversi. Poiché spesso i diversi dispositivi non sono disponibili, questi vengono emulati ed oggi si usano il cloud e le macchine virtuali per consentire l'esecuzione parallela di emulatori con caratteristiche diverse e quindi poter collaudare l'app Android con dispositivi ed in contesti diversi. Il lavoro di tesi aggiunge l'uso dei container come soluzione per migliorare la gestione delle risorse e/o dei costi. Si tratterebbe quindi di fare due cose: (a) sistematizzare l'esecuzione di un emulatore Android all'interno di un container (esistono esempi in merito, ma nulla di sistematico) e (b) realizzare un sistema, magari attraverso Kubernetes, per l'orchestrazione dei diversi container e la creazione automatizzata di contesti d'esecuzione diversi.

Cruscotto del ricercatore

La valutazione oggettiva e quantitativa della ricerca è oggi fondamentale in diversi contesti. Diverse organizzazioni forniscono un insieme di informazioni parziali, ma non esiste un'unica soluzione che presenti in modo organico e riassuma i diversi dati (ad esempio, numero di pubblicazioni, numero di citazioni, hindex, qualità delle conferenze, qualità delle riviste, ecc.). A volte queste informazioni sono disponibili attraverso API dedicate, mentre altre volte devono essere estratte in modo opportuno da pagine web. La tesi si propone di organizzare i dati disponibili, capire come acquisire le informazioni di interesse e realizzare poi un cruscotto del ricercatore che presenti le informazioni raccolte in modo univoco, semplice e personalizzabile.

Analisi formale di modelli BIM

Un modello BIM (Building Information Modeling) cattura tutti gli aspetti di un edificio. Poiché nella sostanza è possibile creare un modello ad oggetti dell'edificio di interesse e di tutte le sue parti, la tesi si propone di creare un linguaggio specifico (domain-specific) per la definizione di vincoli sul modello (ad esempio, per le distanze dai vicini, per le dimensioni minime di una camera da letto o per il volume di aria scambiata da un impianto di condizionamento) ed un motore di verifica ispirato ai tanti metodi e modelli formali disponibili per l'analisi e la progettazione del software. L'integrazione con un tool di progettazione (ad esempio, AutoCAD o Revit) potrebbe essere un plus. L'idea di fondo è di iniziare a pensare ad un gemello digitale (digital twin) vivo dell'edificio di interesse.

JML e Java11

Java Modeling Language (JML) è il linguaggio di riferimento usato per insegnare la programmazione per contratto nel corso di Ingegneria del Software. Purtroppo, dopo diversi anni, e dopo varie versioni di Java, oggi non esiste un tool utilizzabile per definire e valutazione asserzioni scritte in JML; esistono diverse soluzioni parziali che funzionano solamente in casi molto particolari. La tesi si propone sia di dare una definizione precisa, compiuta ed eventualmente estendibile di JML sia di realizzare un'opportuna libreria che consenta la valutazione delle asserzioni definite. La libreria deve essere indipendente dai diversi IDE e compilatori Java utilizzati e deve anche sfruttare le caratteristiche innovative di Java 11 per fornire meccanismi di valutazione semplici e flessibili.