CAPITOLO 1

INTRODUZIONE

L’Infrastruttura è un principio fondamentale del processo di sviluppo software: essa è direttamente responsabile del funzionamento di un’applicazione software. Le infrastrutture possono spaziare da server, load balancers, firewalls e database a complessi cluster di container.

Le considerazioni sull’Infrastruttura sono valide al di là dei production environments, poiché si estendono all’intero processo di sviluppo. Includono strumenti e piattaforme CI/CD, ambienti di staging e strumenti di test. Queste considerazioni aumentano al crescere della complessità del prodotto software. Ciò porta alla conclusione che il tradizionale approccio per la gestione manuale delle infrastrutture diventa una soluzione non scalabile per soddisfare le esigenze dei moderni cicli di sviluppo software basati su DevOps.

È così che **Infrastructure as Code** (IaC) è diventata oggi la soluzione de facto in fase di sviluppo. IaC consente di soddisfare le crescenti esigenze di modifiche dell’infrastruttura in modo scalabile e tracciabile.

Infrastructure as Code, però, risolve in parte i problemi incontrati durante il periodo di gestione manuale delle infrastrutture.

Il lavoro di questa tesi va ad affrontare il problema legato all’inconsistenza linguistica nel contesto dell’IaC. La presenza di inconsistenze può fuorviare i developers, ad esempio portandoli ad equivocare il comportamento di una data porzione di codice o a spendere tempo eccessivo per comprenderlo. È importante, dunque, evidenziare la loro presenza, affinché sia prodotto codice pulito e facilmente comprensibile.

## Struttura della tesi

La tesi è organizzata in cinque capitoli, incluso il presente capitolo introduttivo e quello delle Conclusioni:

* il **Capitolo 2** fornisce una presentazione sui problemi collegati alle infrastrutture IT, proponendo una parziale soluzione iniziale, fino ad arrivare a quella fornita da Infrastructure as Code, proponendo Ansible come Configuration Manager.
* il **Capitolo 3** presenta i Linguistic Antipatterns (LA), ovvero cattive pratiche nella nominazione, documentazione e scelta degli identificatori nell’implementazione di una entità che potrebbero causare incomprensioni del programma.
* il **Capitolo 4** tratta degli esperimenti condotti in ambito di detecting di linguistic antipatterns nel contesto dell’Infrastructure as Code, sulla base di uno studio effettuato presso il *Jheronimus Academy of Data Science*.