**Elasticsearch: distributed, portable, RESTful search engine**



## Abstract

## Introduction

In the past, we stored information in files, books. They were importable and hard to maintain. And when we tried to take advantage of some information, it would took substantial time and resource to look up and analyze all the files.

Now, things are quite different. We live in a world of data now, everything can be stored as bits in disks. We have developed many sophisticated databases to reliably store and look up data. However, when we talk about efficient and highly customized search and analysis, those databases are incapable. Then some amazing applications come to rescue. Elasticsearch is one of them.

Elasticsearch is an open source, distributed and RESTful search engine. It provides you the power to make full use of your data, whether it is used to support searching in applications with giant data storage (PB level) or be involved in some cutting edge areas like machine learning.

Its distributed feature makes it to meet the high-performance requirement. It is nearly real-time, which means the data can be searched almost instantly after being stored. What’s more, Elasticsearch provides all kinds of APIs (RESTful, Java, Python…) for outer applications to communicate with this powerful search engine. Due to all of these amazing features, Elasticsearch becomes popular among developers and is widely used production environment.

In this document, we’ll give an overview of the overarching architecture of the Elasticsearch project. We’ll start by introducing the project and discussing its stakeholders. Then focus on different viewpoints and perspectives to analyze Elasticsearch's architecture. In the end, we’d like to discuss the technical debt hidden in the depths of the codebase.

## Stakeholders view

## Context view

## Evolution view

## Stakeholders view

## Development view

## Deployment view

## Technical debt

## Conclusion