Google Cloud

Edge Architecture

Technology Stack

Agenda

- Capabilities
- Edge:
  - Technology stack and Components
- API BaaS:
  - Technology stack and Components

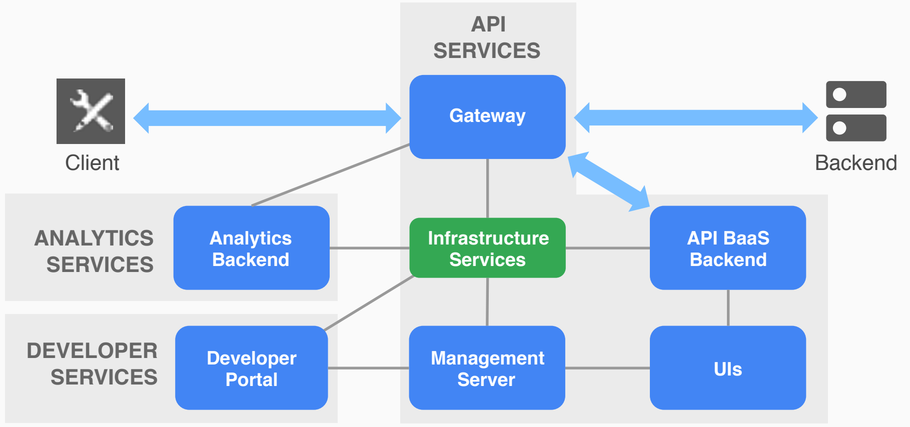
Edge - Capabilities

Edge – Capabilities

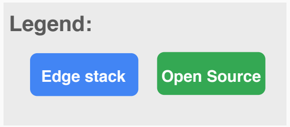
Edge is comprised of several stateless components that use infrastructure services to persist data:

- Gateway: Routing and API calls processing.
- UIs: Enterprise UI, Developer Portal.
- Infrastructure Services: Persistence of runtime, analytics and management.
- Management Server: Provides REST API for all configuration and management tasks.

Note: Monetization, not showed in the diagram below, is part of Developer Services and leverages Gateway, Analytics Services and Management Server.

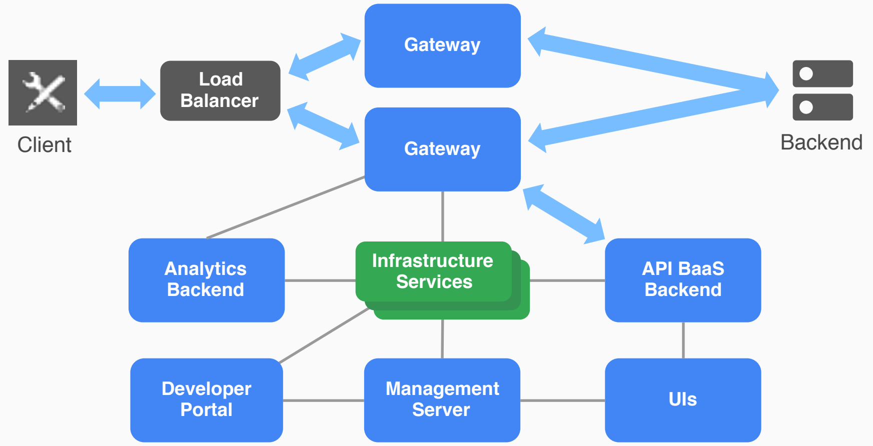


Legend:



Edge – Scalability

- Horizontally scalable.
- Additional Gateway components can be added to keep up with API volume, high availability and resiliency requirements
- As the number of Gateway increases, some of the supporting infrastructure services may need to scale out.

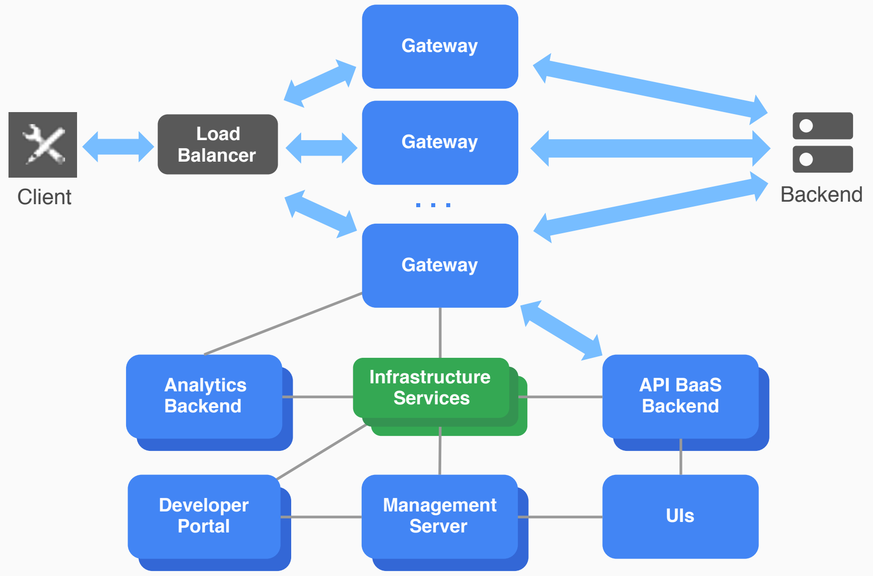


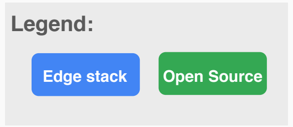
Legend:

Edge stack   Open Source

Edge – Scalability

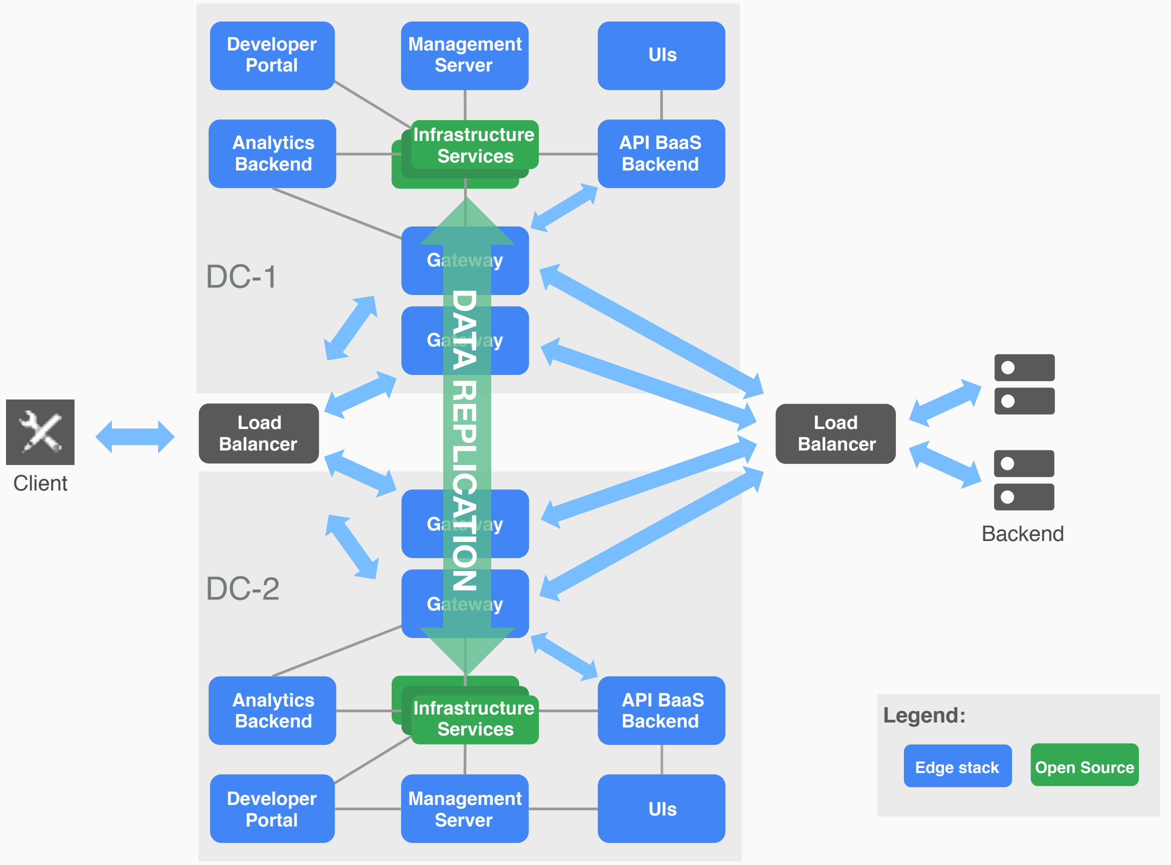
- Management Server, Analytics Backend, API BaaS Backend and Developer Portal can also be set up in an HA way. Multiples instances of these capabilities within a single zone or across zones is possible.
- Gateway, Infrastructure services, analytics and other capabilities can scale-out independently from each other.





Edge – Scalability

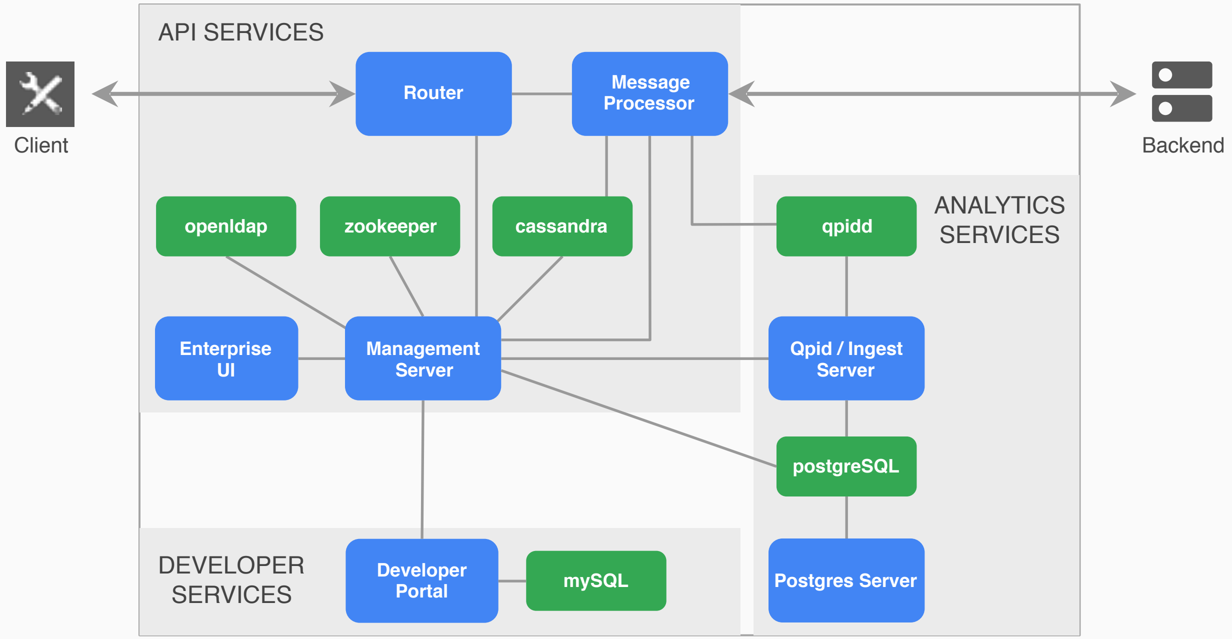
- Multi-DC and DR scalability.
- Edge is capable of scaling across multiple DCs/Regions in a active/ active fashion.
- Active data replication between sites using eventual consistency.



Edge - Technology Stack and Components

Edge – Component View

Each box represents a process. These processes can be can be run on independant of each other or collocated across a limited number of VM/servers.



Legend:

🟦 Edge stack
🟩 Open Source

Google Cloud

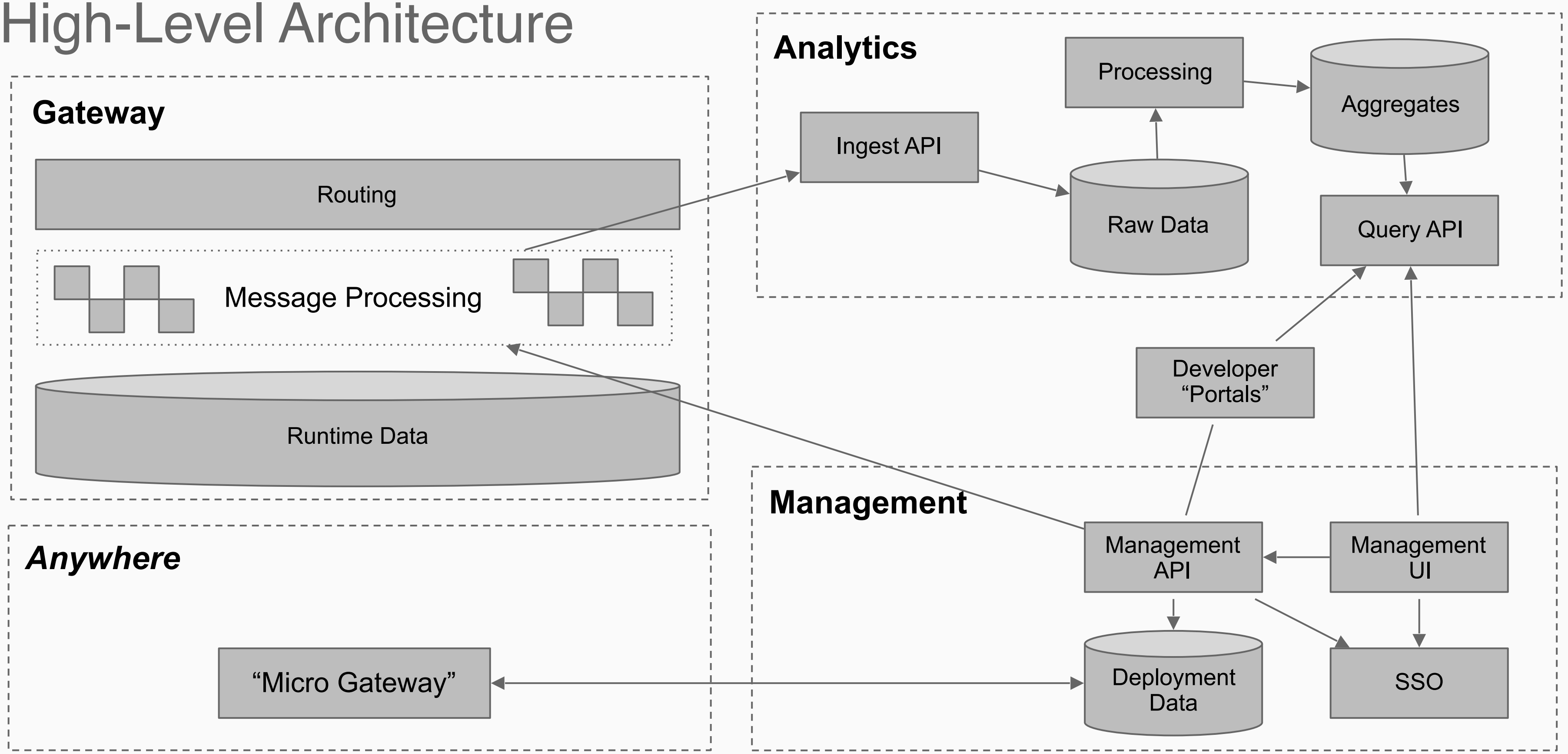
Edge - Technology Stack

Edge components are, in general, Java based. Most components are based on a homegrown technology stack that leverages best in class open source technology under it. Below we highlight some of the underlying technologies used as building blocks.

Message Processor (MP)	Homegrown Java based
Enterprise UI (UI)	Play Framework
Management Server (MS)	Homegrown Java based
Postgres Server (PS)	Homegrown Java based
Qpid/Ingest Server (QS)	Homegrown Java based

Router (R)	Nginx (C++)
Zookeeper (ZK)	Apache Zookeeper
Cassandra (CS)	Apache Cassandra
LDAP (OL)	Open LDAP
Postgres (PG)	PostgreSQL
Qpid (QD)	Apache Qpid

High-Level Architecture



Scaling by capability

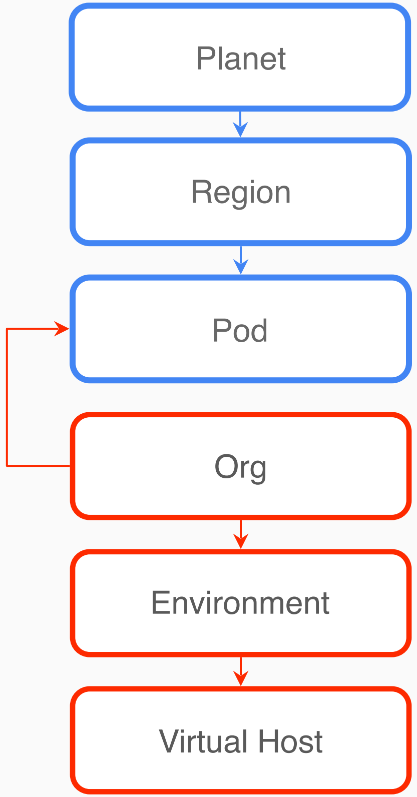
Capability Involved	Components
API Traffic	R, MP, CS
Analytics	QS, QD, PS, PG
Management	UI, MS, OL, ZK
Developer	DP, MY

- Given the responsibility and capabilities offered by each component, scalability requirements and how they are implemented may vary.
- In most scenarios, scaling to accommodate higher API volume may impact only components serving live API traffic.
- Analytics data components may have to be scaled in response to increased API traffic and/or raw analytics data retention policy.
- Other components may grow in number mostly driven by high availability requirements for those capabilities.

Legend:

R	Router	MS	Management Server	DP	Developer Portal	MY	MySQL	OL	Openldap	ES	Elastic Search
MP	Message Processor	PS	Postgres Server	BA	API BaaS Stack	ZK	Zookeeper	PG	PostgreSQL	☐	Server/Virtual Machine
UI	Enterprise UI	QS	Qpid/Ingest Server	BP	API BaaS Portal	CS	Cassandra	QD	Apache Qpid	⬚	POD

Multitenancy



Planets represent an entire physical installation and it can encompass multiple regions and pods

Regional Services are shared by many pods across a single geographical region and maintain state and provide an API that works across the pods in a region

Pods are a collection of servers that share logical functions such as a Gateway Pod or an Analytics Pod.

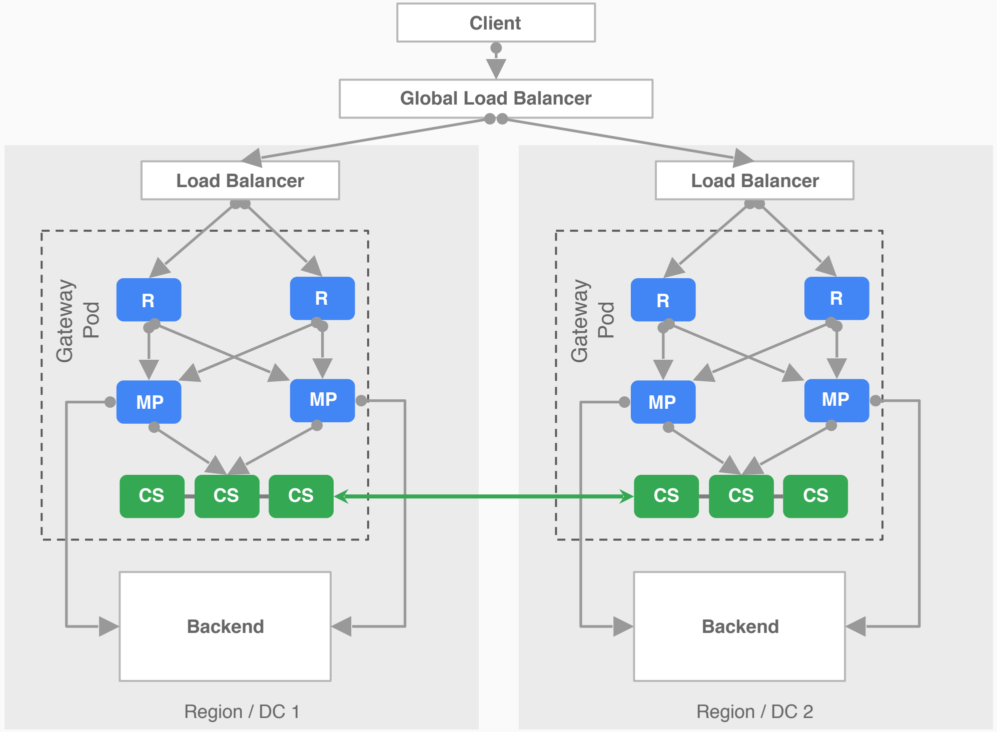
Orgs provide logical grouping to secure access to API management services. Orgs are associated with pods for servers – they can either share hardware or be isolated

Environments are virtual routes that allow API bundles to be deployed and tested within an Org. Environments can be associated with pods for servers independently of the Org.

Virtual Hosts are similar to Apache Virtual hosts and route traffic to environments based on ports and domain names.

API traffic data flow

- Routers send requests to Message Processors within the same Gateway pod.
- If there are two or more gateway pods in a region, then routers will ignore message processors in the other gateway pods.
- Message Processors respect the region as their scope.
- All Edge components are configured to only use the Cassandra nodes in their region/DC.
- Communication between Message Processors and backend systems is driven by API Proxy implementations.

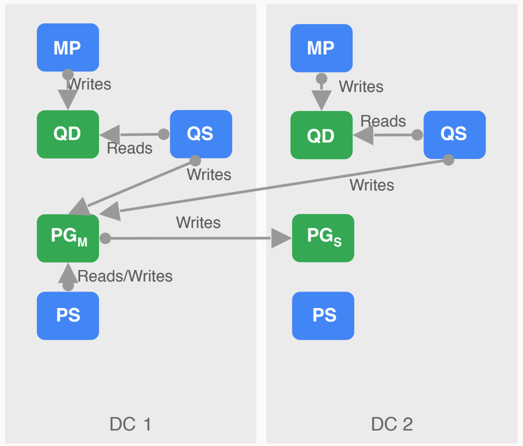


Legend:

R	Router	MS	Management Server	DP	Developer Portal	MY	MySQL	OL	Openldap	ES	Elastic Search
MP	Message Processor	PS	Postgres Server	BA	API BaaS Stack	ZK	Zookeeper	PG	PostgreSQL	☐	Server/Virtual Machine
UI	Enterprise UI	QS	Qpid/Ingest Server	BP	API BaaS Portal	CS	Cassandra	QD	Apache Qpid	⬚	POD

Analytics data flow

Master/Slave



- Analytics data is generated by Message Processor and asynchronously send to Qpid.
- Ingest process consumes analytics raw data from Qpid and stores it on PostgreSQL.
- Data on PostgresSQL is logically partitioned by organization and environment.
- PostgresSQL Master/Slave. Multiple slaves can be added on each region.
- PostgresServer aggregates data.
- Analytics record size is about 2kb. It will vary if custom variables are captured.
- Analytics data is partitioned by Organization and Environment.

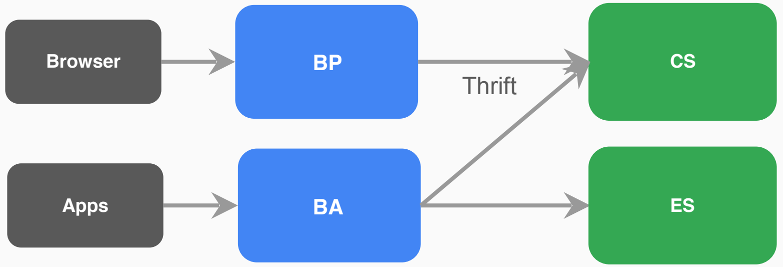
Google Cloud

Legend:

R	Router	MS	Management Server	DP	Developer Portal	MY	MySQL	OL	Openldap	ES	Elastic Search
MP	Message Processor	PS	Postgres Server	BA	API BaaS Stack	ZK	Zookeeper	PG	PostgreSQL	☐	Server/Virtual Machine
UI	Enterprise UI	QS	Qpid/Ingest Server	BP	API BaaS Portal	CS	Cassandra	QD	Apache Qpid	⬚	POD

API BaaS - Technology Stack and Components

API BaaS



API BaaS Stack

- Data storage & management
- Flexible data querying
- Social
- User management
- Geolocation
- Push notifications
- Configuration management

API BaaS Portal

- Management UI.
- Allows creation and maintenance of organization, collections, users and other entities.
- Error & performance monitoring

Cassandra

- Apache Cassandra provides distributed eventual consistent data storage.

Elastic Search

- Elasticsearch provides indexing and searching.
- https://github.com/elastic/elasticsearch

http://usergrid.apache.org/

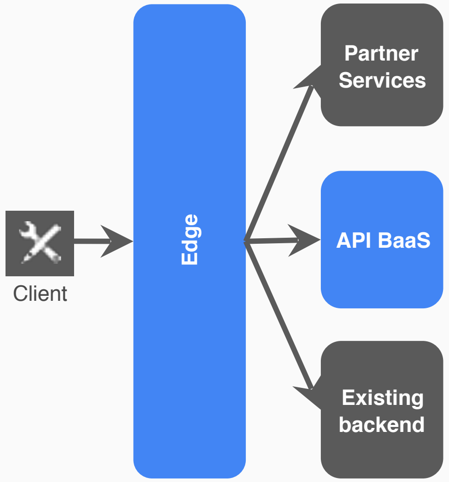
Legend:

R	Router	MS	Management Server	DP	Developer Portal	MY	MySQL	OL	Openldap	ES	Elastic Search
MP	Message Processor	PS	Postgres Server	BA	API BaaS Stack	ZK	Zookeeper	PG	PostgreSQL	▭	Server/Virtual Machine
UI	Enterprise UI	QS	Qpid/Ingest Server	BP	API BaaS Portal	CS	Cassandra	QD	Apache Qpid	⬚	POD

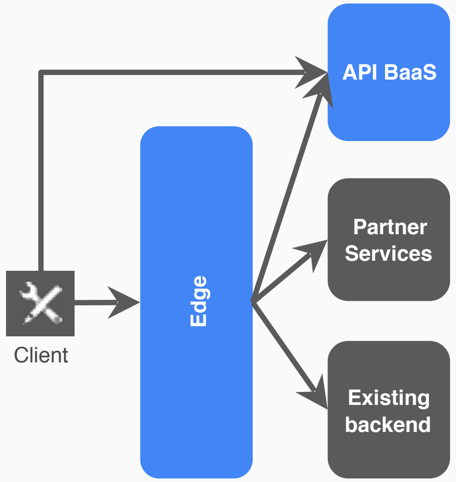
Google Cloud

API BaaS – Where it fits on the architecture?

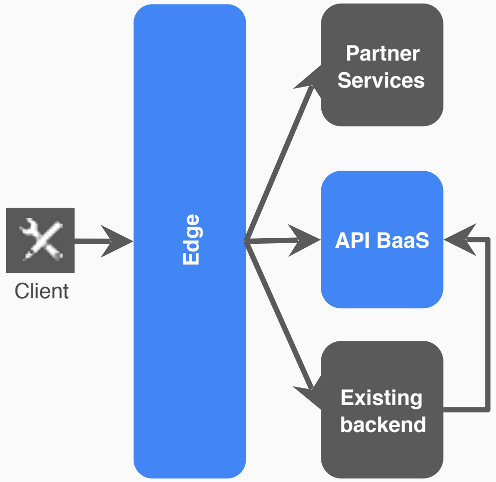
1. As Edge target



2. Direct access from Mobile

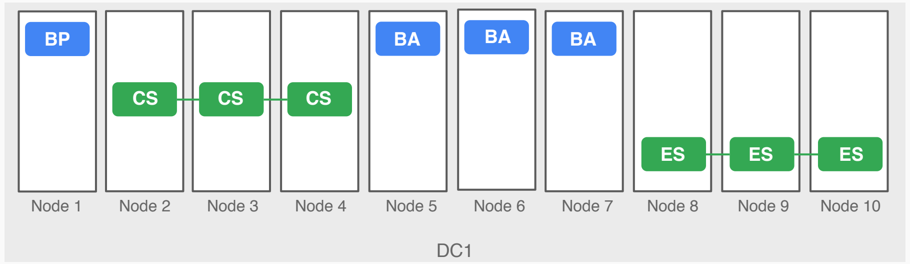


3. As complement to existing backend systems



API BaaS Components

- Footprint is driven by requirements. Transaction volumes, availability and reliability among others drive components stacking and number of nodes.
- BA, ES and CS are critical components to handle live API BaaS API traffic.
- API BaaS Analytics provided by BP.
- Cassandra can be a dedicated ring for API BaaS or shared with Edge.



Legend:

R	Router	MS	Management Server	DP	Developer Portal	MY	MySQL	OL	Openldap	ES	Elastic Search
MP	Message Processor	PS	Postgres Server	BA	API BaaS Stack	ZK	Zookeeper	PG	PostgreSQL	☐	Server/Virtual Machine
UI	Enterprise UI	QS	Qpid/Ingest Server	BP	API BaaS Portal	CS	Cassandra	QD	Apache Qpid	⬚	POD

THANK YOU