



MongoDB for **ENEA**

Architecture and Technology Overview

Safe Harbor Statement

This presentation contains “forward-looking statements” within the meaning of Section 27A of the Securities Act of 1933, as amended, and Section 21E of the Securities Exchange Act of 1934, as amended. Such forward-looking statements are subject to a number of risks, uncertainties, assumptions and other factors that could cause actual results and the timing of certain events to differ materially from future results expressed or implied by the forward-looking statements. Factors that could cause or contribute to such differences include, but are not limited to, those identified our filings with the Securities and Exchange Commission. You should not rely upon forward-looking statements as predictions of future events. Furthermore, such forward-looking statements speak only as of the date of this presentation.

In particular, the development, release, and timing of any features or functionality described for MongoDB products remains at MongoDB’s sole discretion. This information is merely intended to outline our general product direction and it should not be relied on in making a purchasing decision nor is this a commitment, promise or legal obligation to deliver any material, code, or functionality. Except as required by law, we undertake no obligation to update any forward-looking statements to reflect events or circumstances after the date of such statements.

Agenda

- MongoDB: overview tecnologica
- Sicurezza: GDPR e gestione ruoli
- Stitch: back-end as a service
- Use case di gestione documentale
- Architettura e best practice per IOT



Why MongoDB



MongoDB Use Cases



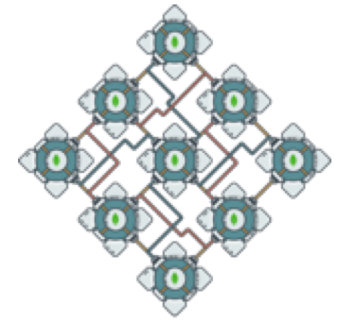
**ANALYTICS
BIG DATA**



**ARTIFICIAL
INTELLIGENCE**



IOT



BLOCKCHAIN



**CONTENT
MANAGEMENT**



SINGLE VIEW



GDPR



**OPERATIONAL
DATA LAYER**

Why MongoDB?

Intelligent Operational Data Platform



**Best way to work
with data**



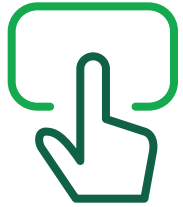
**Intelligently put data
where you need it**



**Freedom
to run anywhere**

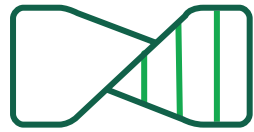
Overview Tecnologica

Best way to work with data



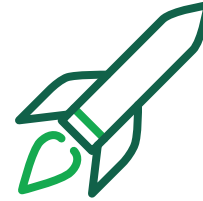
Easy:

Work with data
in a natural,
intuitive way



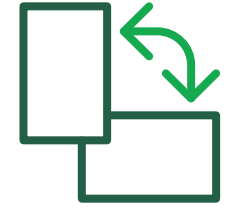
Flexible:

Adapt and
make changes
quickly



Fast:

Get great
performance
with less code



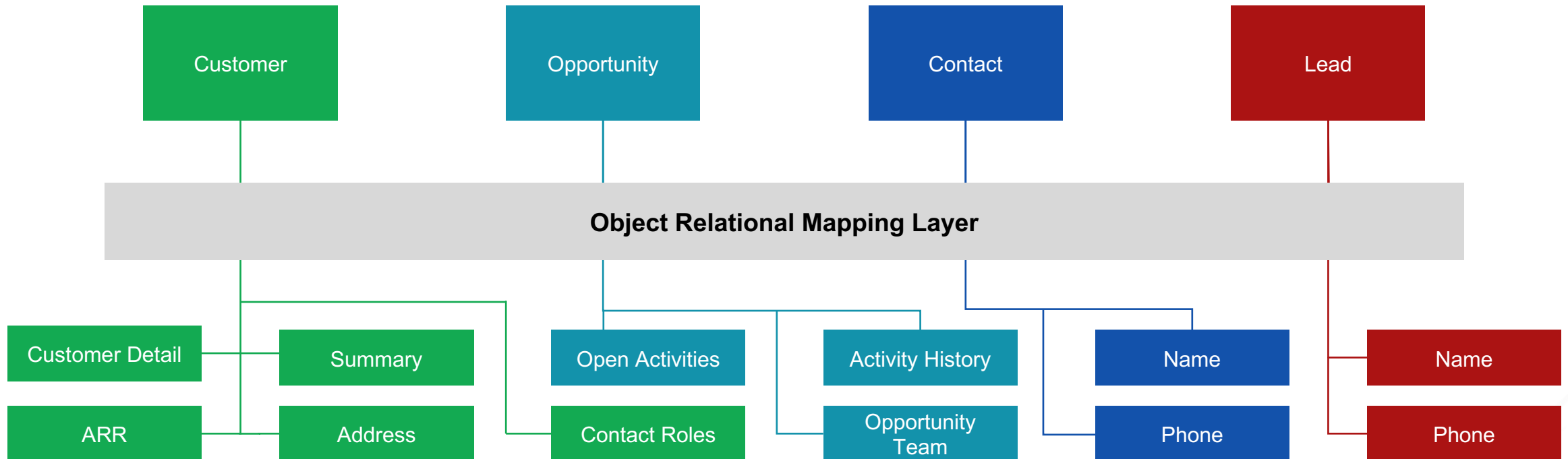
Versatile:

Supports a
wide variety of
data models
and queries



Go from this....

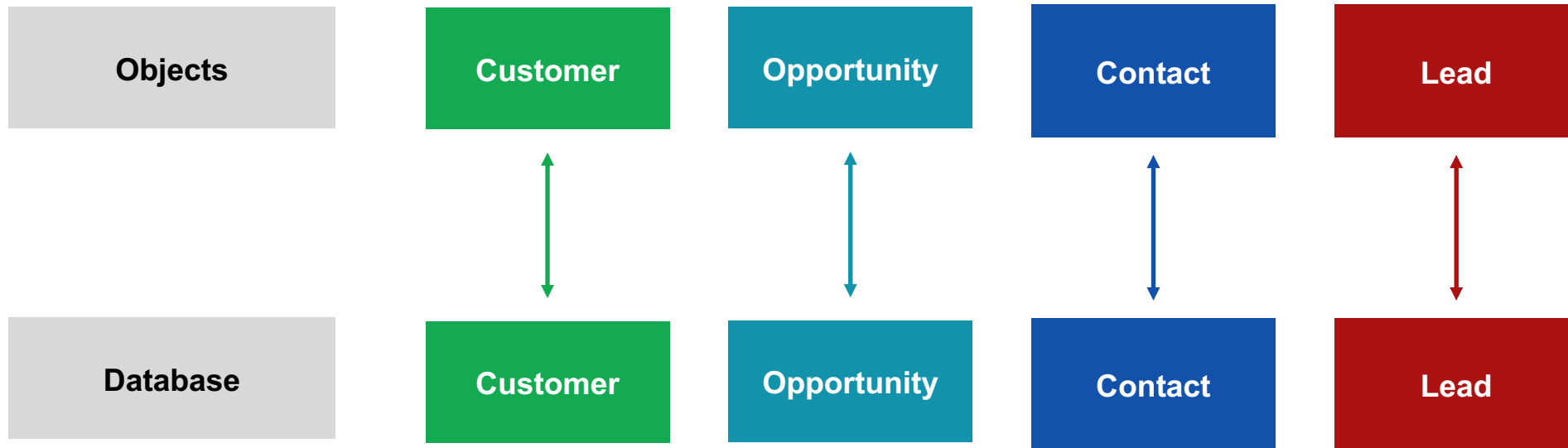
Objects



Tables

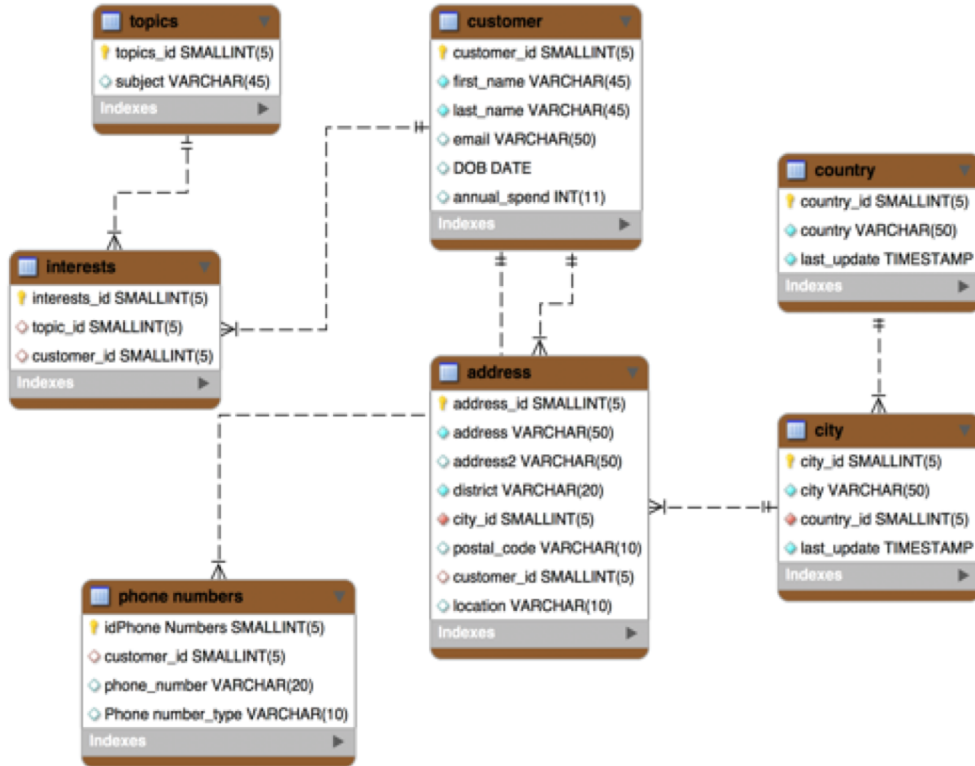


To this: store objects directly...





Easy: Contrasting data models



Tabular (Relational) Data Model

Related data split across multiple records and tables

```
{
  "_id" : ObjectId("5ad88534e3632e1a35a58d00"),
  "name" : {
    "first" : "John",
    "last" : "Doe" },
  "address" : [
    { "location" : "work",
      "address" : {
        "street" : "16 Hatfields",
        "city" : "London",
        "postal_code" : "SE1 8DJ"},
      "geo" : { "type" : "Point", "coord" : [
        51.5065752, -0.109081] } } },
    { ... }
  ],
  "phone" : [
    { "location" : "work",
      "number" : "+44-1234567890" },
    { ... }
  ],
  "dob" : ISODate("1977-04-01T05:00:00Z"),
  "retirement_fund" : NumberDecimal("1292815.75")
}
```

Document Data Model

Related data contained in a single, rich document



Easy: Document data model

- Naturally maps to objects in code
- Represent data of any structure
- Strongly typed for ease of processing
 - Over 20 binary encoded JSON data types
- Access by idiomatic drivers in all major programming language

```
{
  "_id" : ObjectId("5ad88534e3632e1a35a58d00"),
  "name" : {
    "first" : "John",
    "last" : "Doe" },
  "address" : [
    { "location" : "work",
      "address" : {
        "street" : "16 Hatfields",
        "city" : "London",
        "postal_code" : "SE1 8DJ"},
      "geo" : { "type" : "Point", "coord" : [
        51.5065752, -0.109081 ]}},
    + { ... }
  ],
  "phone" : [
    { "location" : "work",
      "number" : "+44-1234567890"},
    + { ... }
  ],
  "dob" : ISODate("1977-04-01T05:00:00Z"),
  "retirement_fund" : NumberDecimal("1292815.75")
}
```



Easy: Drivers and Frameworks

Drivers



Frameworks





Flexible: Adapt to change

```
{
  "_id" : ObjectId("5ad88534e3632e1a35a58d00"),
  "name" : {
    "first" : "John",
    "last" : "Doe" },
  "address" : [
    { "location" : "work",
      "address" : {
        "street" : "16 Hatfields",
        "city" : "London",
        "postal_code" : "SE1 8DJ"},
      "geo" : { "type" : "Point", "coord" : [
        51.5065752,-0.109081]}}],
+   {...}
  ],
  "dob" : ISODate("1977-04-01T05:00:00Z"),
  "retirement_fund" : NumberDecimal("1292815.75")
}
```

```
{
  "_id" : ObjectId("5ad88534e3632e1a35a58d00"),
  "name" : {
    "first" : "John",
    "last" : "Doe" },
  "address" : [
    { "location" : "work",
      "address" : {
        "street" : "16 Hatfields",
        "city" : "London",
        "postal_code" : "SE1 8DJ"},
      "geo" : { "type" : "Point", "coord" : [
        51.5065752,-0.109081]}}],
+   {...}
    { "location" : "work",
      "address" : {
        "street" : "16 Hatfields",
        "city" : "London",
        "postal_code" : "SE1 8DJ"},
      "geo" : { "type" : "Point", "coord" : [
        51.5065752,-0.109081]}}],
+   {...}
  ],
  "phone" : [
    { "location" : "work",
      "number" : "+44-1234567890"},
+   {...}
  ],
  "dob" : ISODate("1977-04-01T05:00:00Z"),
  "retirement_fund" : NumberDecimal("1292815.75")
}
```

Add new fields dynamically at runtime



Flexible: Govern



JSON Schema

Enforces strict schema structure over a complete collection for data governance & quality

- Builds on document validation introduced by restricting new content that can be added to a document
- Enforces presence, type, and values for document content, including nested array
- Simplifies application logic

Tunable: enforce document structure, log warnings, or allow complete schema flexibility

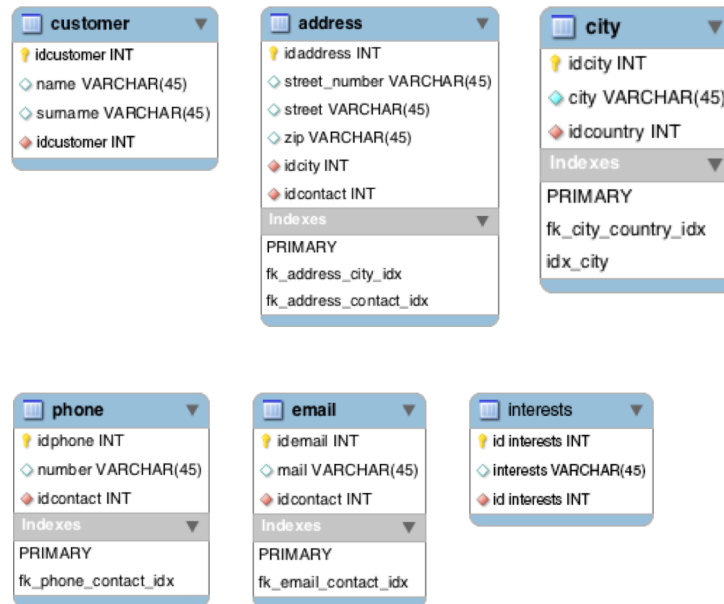
Queryable: identify all existing documents that do not comply



Fast: To work with data

Compared to storing data across multiple tables, a single document data structure:

- Presents a single place for the database to read and write data
- Denormalized data eliminates JOINS for most operational queries
- Simplifies query development and optimization



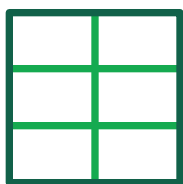
```
_id: 12345678
> name: Object
> address: Array
> phone: Array
  email: "john.doe.@mongodb.com"
  dob: 1966-07-30 01:00:00:000
v interests: Array
  0: "Cycling"
  1: "IoT"
```



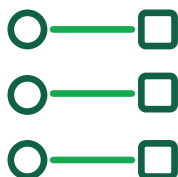
Versatile: Multiple data models, rich query functionality



JSON Documents



Tabular



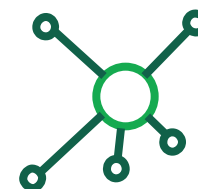
Key-Value



Text



Geospatial



Graph

Rich Queries

Point | Range | Geospatial | Faceted Search | Aggregations | JOINS | Graph Traversals



Versatile: Rich query functionality

Expressive Queries	- Find anyone with phone # "1-212..." - Check if the person with number "555..." is on the "do not call" list
Geospatial	Find the best offer for the customer at geo coordinates of 42nd St. and 6th Ave
Text Search	Find all tweets that mention the firm within the last 2 days
Aggregation	Count and sort number of customers by city, compute min, max, and average spend
Native Binary JSON Support	- Add an additional phone number to Mark Smith's record without rewriting the document - Update just 2 phone numbers out of 10 - Sort on the modified date
JOIN (\$lookup)	Query for all San Francisco residences, lookup their transactions, and sum the amount by person
Graph Queries (\$graphLookup)	Query for all people within 3 degrees of separation from Mark

MongoDB

```
{
  customer_id : 1,
  first_name  : "Mark",
  last_name   : "Smith",
  city        : "San Francisco",
  phones: [
    {
      number : "1-212-777-1212",
      type   : "work"
    },
    {
      number : "1-212-777-1213",
      type   : "cell"
    }
  ]
}
..... . . .
```




Fully Indexable

Fully featured secondary indexes

Index Types

- Primary Index
 - Every Collection has a primary key index
- Compound Index
 - Index against multiple keys in the document
- MultiKey Index
 - Index into arrays
- Text Indexes
 - Support for text searches
- GeoSpatial Indexes
 - 2d & 2dSphere indexes for spatial geometries
- Hashed Indexes
 - Hashed based values for sharding

Index Features

- TTL Indexes
 - Single Field indexes, when expired delete the document
- Unique Indexes
 - Ensures value is not duplicated
- Partial Indexes
 - Expression based indexes, allowing indexes on subsets of data
- Case Insensitive Indexes
 - supports text search using case insensitive search
- Sparse Indexes
 - Only index documents which have the given field



Aggregations

Advanced data processing pipeline for transformations and analytics

- Multiple stages
- Similar to a unix pipe
 - Build complex pipeline by chaining commands together
- Rich Expressions

Collection

```
db.orders.aggregate( [  
  $match stage  
  $group stage  
  "$cust_id",total: { $sum: "$amount" } } ] )
```

"A123",	cust_id:
500,	amount:
"A",	status:
"A123",	cust_id:
250,	amount:
"A",	status:
"B212",	cust_id:
200,	amount:
"A",	status:
"A123",	cust_id:
300,	amount:
"D",	status:

Orders

\$match

{	cust_id:
"A123",	amount:
"A",	status:
"A123",	cust_id:
250,	amount:
"A",	status:
"B212",	cust_id:
200,	amount:
"A",	status:

\$group

{	id: "A123",
	total: 750
}	
{	id: "B212",
	total: 200
}	

```
{ $match: { status: "A" }  
{ $group: { _id:  
  ]
```



Aggregation Features

A feature rich analytical framework

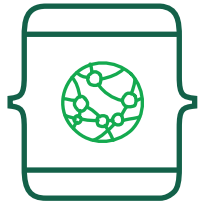
Pipeline Stages

- \$match
- \$group
- \$facet
- \$geoNear
- \$graphLookup
- \$lookup
- \$project
- \$sort
- \$unwind

Operators

- Mathematical
 - \$add, \$abs, \$subtract, \$multiply, \$divide, \$log, \$log10, \$stdDevPop, \$stdDevSam, \$avg, \$sqrt, \$pow, \$sum, \$zip, \$convert, etc.
- Array
 - \$push, \$reduce, \$reverseArray, \$addToSet, \$arrayElemAt, \$slice, etc.
- Conditionals
 - \$and, \$or, \$eq, \$lt, \$lte, \$gt, \$gte, \$cmp, \$cond, \$switch, \$in, etc
- Date
 - \$dateFromParts, \$dateToParts, \$dateFromString, \$dateToString, \$dayOfMonth, \$isoWeek, \$minute, \$month, \$year, etc.
- String
 - \$toUpper, \$toLower, \$substr, \$strcasecmp, \$concat, \$split, etc.
- Laterals
 - \$exp, \$let, \$literal, \$map, \$type, etc

Intelligently put data where you need it



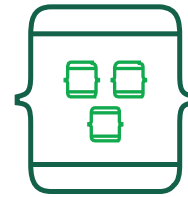
Highly Available

Built-in **multi-region** high availability, replication & automated failover



Workload Isolation

Ability to run both **operational & analytics workloads** on same cluster, for timely insight and lower cost



Scalability

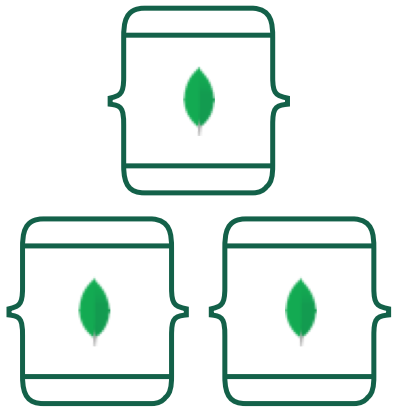
Elastic horizontal scalability – add/remove capacity dynamically **without downtime**



Locality

Declare **data locality rules** for governance (e.g. data sovereignty), class of service & local low latency access

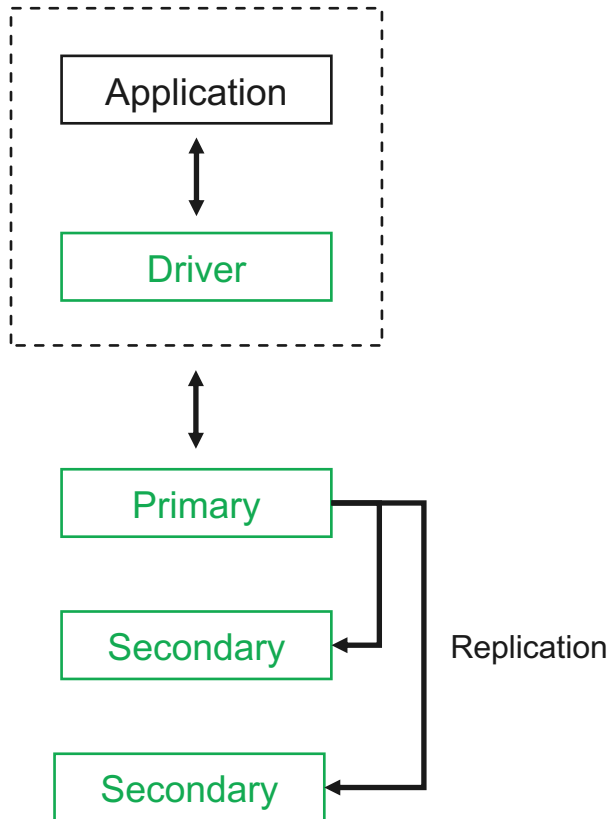
Put data where you need it: Availability



Replica Sets

- Up to 50 replicas, distributed across racks, data centers, and regions
- Self-healing
- Tunable durability and consistency controls
- Always-on write availability with retryable writes

MongoDB Replica Sets



Replica Set – 2 to 50 copies

Self-healing

Data Center Aware

Addresses availability considerations:

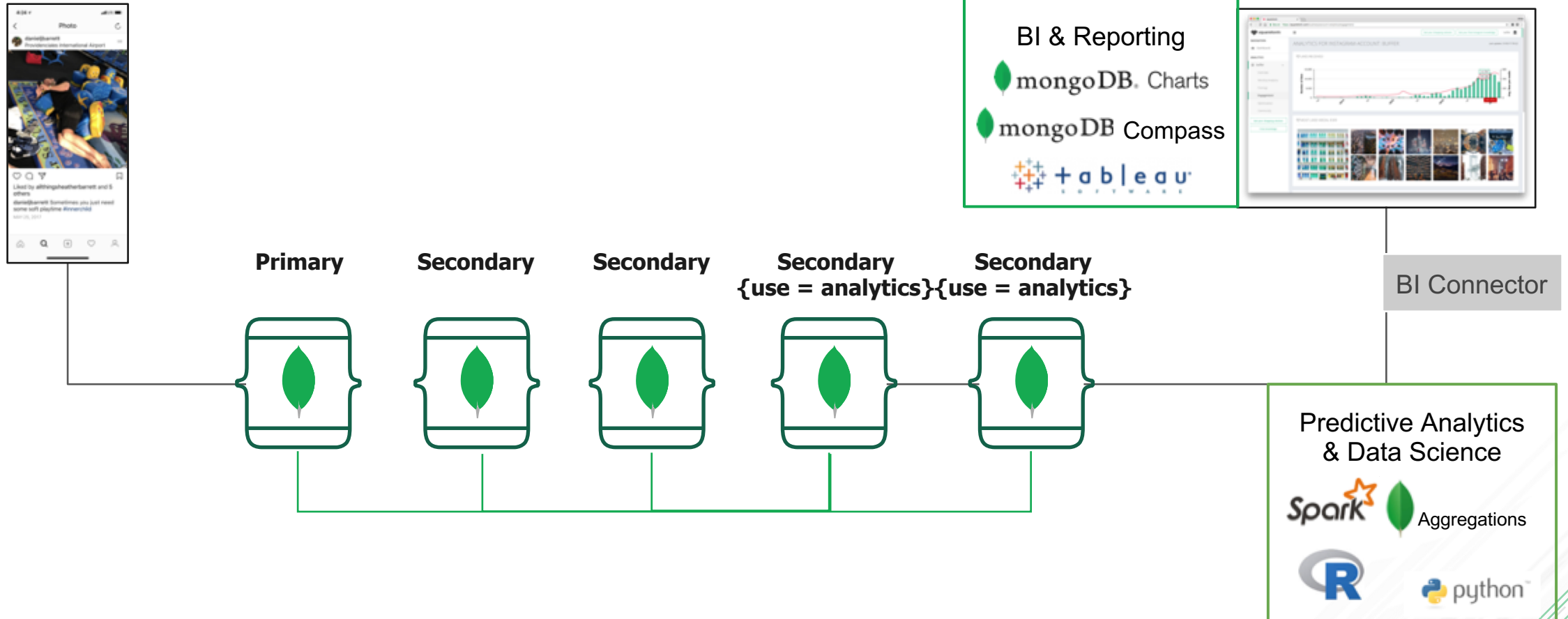
- High Availability
- Disaster Recovery
- Maintenance

Workload Isolation: operational & analytics

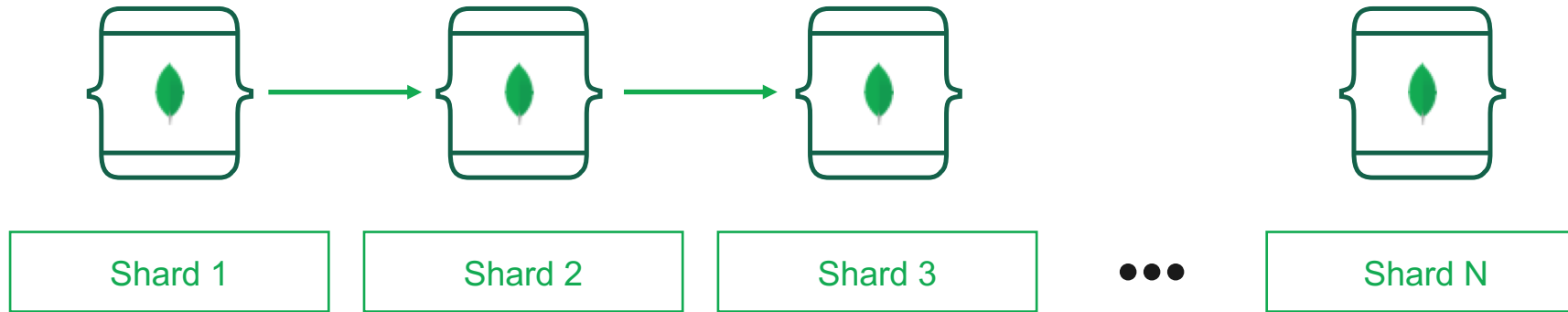
Co-locating operational and analytical workloads

Operational

Analytical



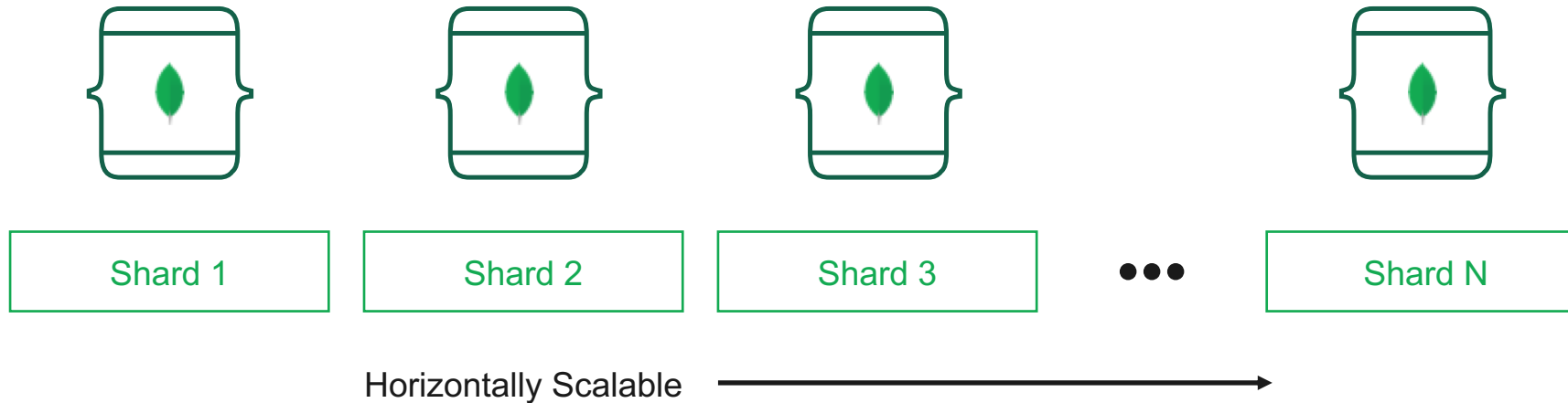
Put data where you need it: Scalability



Auto-Sharding

- Automatically scale beyond the constraints of a single node
- Application transparent
- Scale and rebalance incrementally, in real time
- Unlike NoSQL systems that randomly spray data across a cluster, MongoDB exposes multiple data distribution policies to optimize for query patterns and locality

Scaling MongoDB: Automatic Sharding

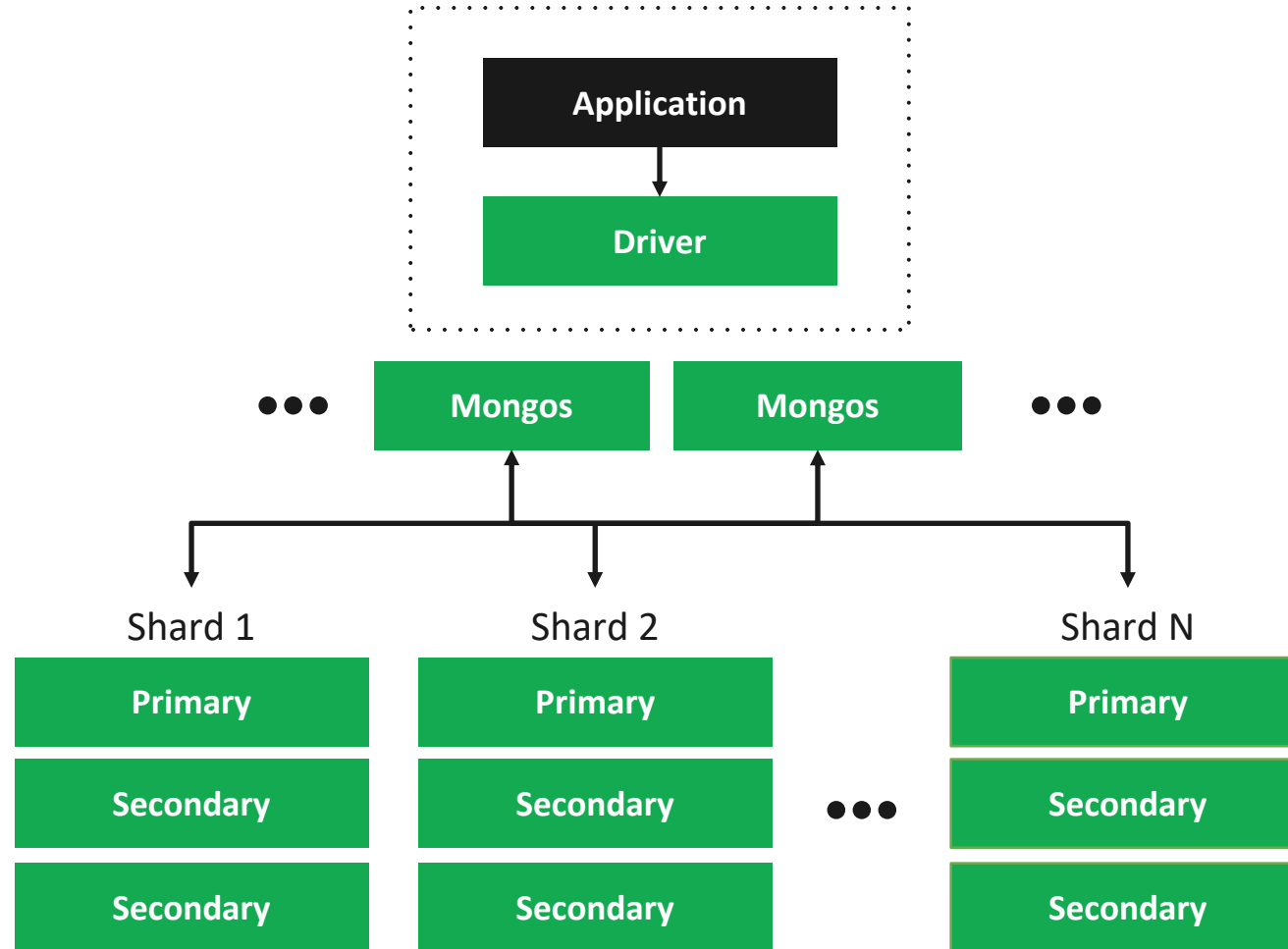


Multiple sharding policies: hashed, ranged, zoned

Increase or decrease capacity as you go

Automatic balancing for elasticity

Sharding Architecture



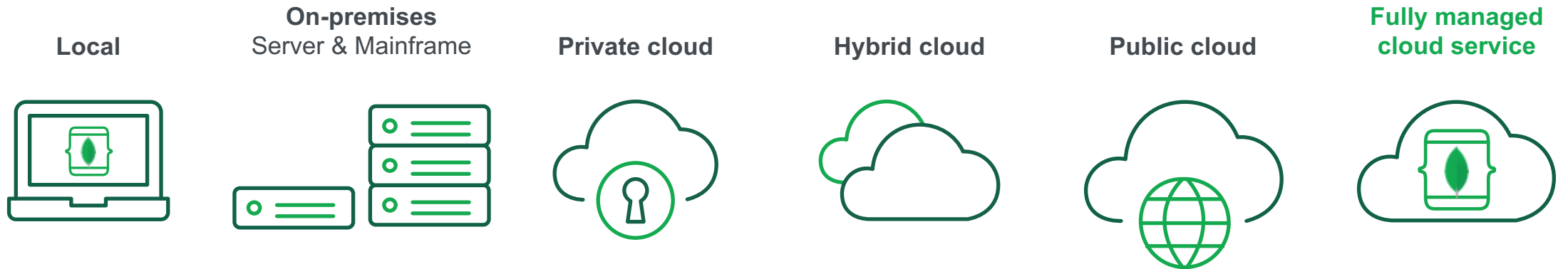
High availability

- Replica sets

Horizontal scalability

- Sharding

Freedom to run anywhere



- Database that runs the same everywhere
- Leverage the benefits of a multi-cloud strategy
- Global coverage
- Avoid lock-in

Convenience: same codebase, same APIs, same tools, wherever you run

Manage & Run MongoDB Yourself Or Use DBaaS

Laptop



Mainframe



Server



Self-managed
in the cloud



Database as a Service



Self-Managed: Ops Mgr

- Automation
- Monitoring Alerting
- Backup/Restore
- Patching
- Scaling
- Performance Advice
- Kubernetes integration

Database as a Service

- Self-service & fully automated
- Global, elastic clusters
- Highly available and secure out of the box
- Operational tooling built in: monitoring, backups
- **On AWS, Azure, GCP, 50+ regions total**

Atlas

unlocks **agility**
and **reduces**
cost



Self-service and
elastic



Global and highly
available



Secure by default



Comprehensive
monitoring



Managed backup



Cloud agnostic

mongoDB® Atlas : The only *true* multi-cloud database as a service



Database that runs the same everywhere

Consistent experience across AWS, Azure, and GCP

Coverage in any geography

Deploy in over 50 regions worldwide
Create globally distributed databases with a few clicks

Leverage the benefits of a multi-cloud strategy

Exploit the benefits of AWS, Azure, or GCP services on your data

Avoid lock-in

Easily migrate data between cloud providers

MongoDB Atlas: Database as a Service

Self-service and elastic

- Deploy in minutes
- Scale up/down without downtime
- Automated upgrades

Global and highly available

- 54 Regions worldwide
- Replica sets optimized for availability
- Global clusters for read/write everywhere deployments

Secure by default

- Network isolation and Peering
- Encryption in flight and at rest with key management
- Role-based access control; auditing
- SOC 2 / Privacy Shield / HIPAA

Comprehensive Monitoring

- Performance Advisor
- Dashboards w/ 100+ metrics
- Real Time Performance
- Customizable alerting

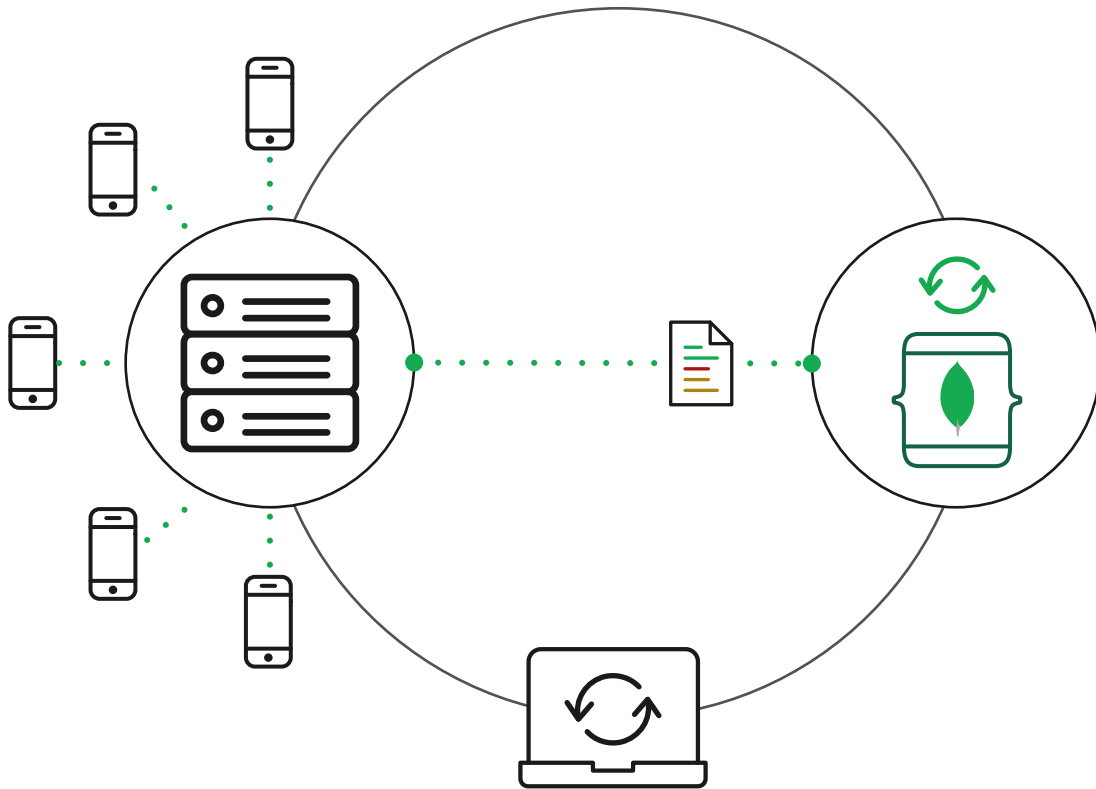
Managed Backup

- Point in Time Restore
- Queryable backups
- Consistent snapshots

Cloud Agnostic

- AWS, Azure, and GCP
- Easy migrations
- Consistent experience

Freedom to run Anywhere: Cloud Migration



Migrate existing deployments running anywhere into MongoDB Atlas with minimal impact to your application.

Live migration works by:

- Performing a sync between your source database and a target database hosted in MongoDB Atlas
- Syncing live data between your source database and the target database by tailing the oplog
- Notifying you when it's time to cut over to the MongoDB Atlas cluster

GDPR e Sicurezza

Secure: Wherever you put your data

Business Needs

Security Features

Authentication	SHA-2, LDAP, Kerberos, x.509 Certificates
Authorization	RBAC, Read-Only Views, Field-Level Redaction
Auditing	Admin, DML, DDL, Role-based
Encryption	Network: TLS (with FIPS 140-2), Disk: Encrypted Storage Engine or Partner Solutions



Business Transformation with Event-Driven Architecture Powered By MongoDB

UK's Leading Commercial Property Data Service Delivers 50x More Releases Per Month, Achieving Always-On Availability

Problem	Solution	Results
Need to develop a new platform for the company to move from traditional print media to a digital business delivering market intelligence and tools across multiple online channels Monolithic application architecture and rigid relational database prevented IT team pushing new updates any more than once per month	Moved to MEAN stack powered by a microservices-based architecture on the cloud MongoDB Enterprise Advanced for access to Ops Manager, Compass and Support MongoDB Encrypted storage engine to meet demands of EU GDPR data privacy legislation Geospatial queries and indexes to allow rich query & search across property data	Transformed business: now digital is driving revenue growth Supports 50x more releases per month, with always on availability Faster development velocity to build custom services for multiple market segments Reduced operational overhead with fault tolerant replica sets

GDPR Data Protection Requirements



DISCOVER

Identify all PII
in your systems



DEFEND

Implement appropriate security
controls



DETECT

Monitor to identify suspicious
behavior, remediate gaps

Mapping Required Capabilities to GDPR

Discover



Defend



Detect



Identify Personal Data	Access Control	Monitor & Report
Implement Retention Policies	Pseudonymisation & Encryption	Audit
	Resilience & DR	
	Data Sovereignty	

How MongoDB Supports GDPR

Discover



Defend



Detect



Identify Personal Data

- MongoDB Compass
- Expressive Queries & Analytics

Access Control

- Authentication (i.e. LDAP, Kerberos)
- Authorization (RBAC)
- IP Whitelisting & VPC Peering

Monitor & Report

- Real-Time Alerting

Personal Data Retention

- TTL Indexes

Pseudonymisation & Encryption

- Read-Only Views
- Log Redaction
- TLS/SSL Network Encryption
- Encrypted Storage Engine

Audit

- MongoDB Audit Log

Resilience & DR

- Replica Sets
- MongoDB PIT Backup & Recovery

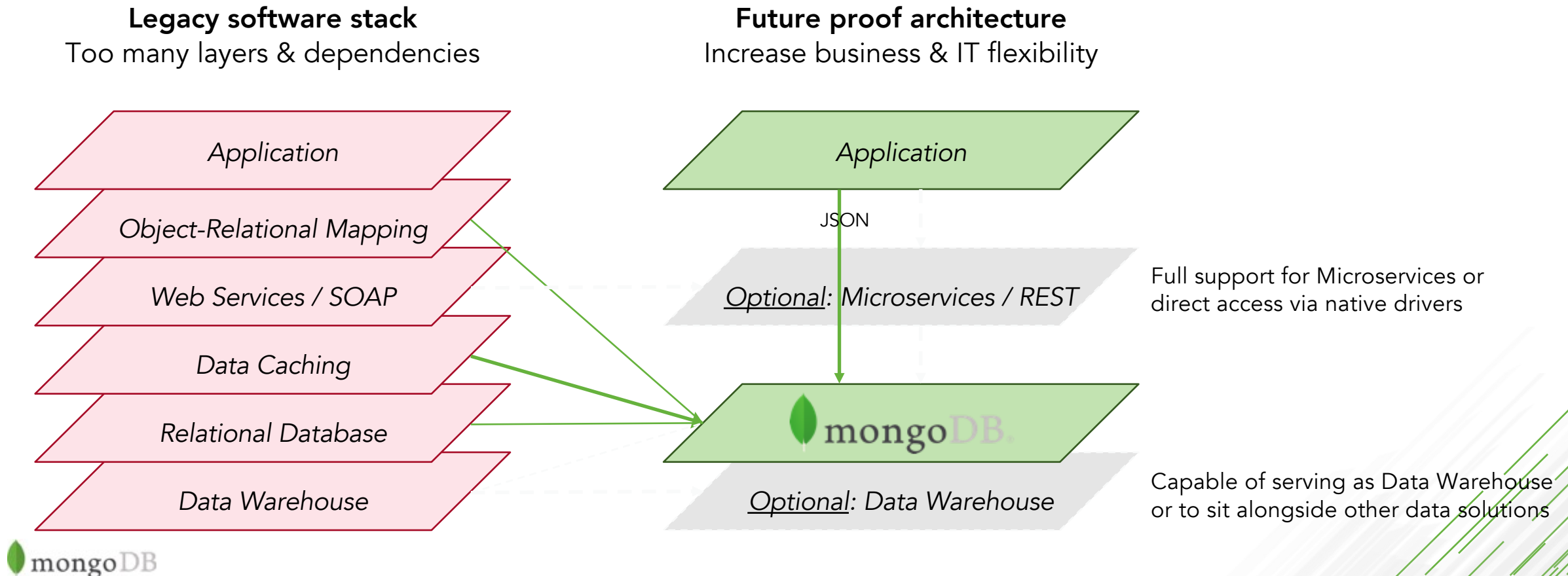
Data Sovereignty

- MongoDB Zones

Stitch: back-end as a service

Simplify technology stacks

Legacy stacks have too many layers, driving complexity & time to market. MongoDB enables you to collapse several legacy layers, as the required capabilities can all be provided directly by MongoDB.



The Broader IODP Accelerates Everything

3-5x increase in productivity by leveraging the Intelligent Operational Data Platform

MongoDB Mobile The power of MongoDB in your device

Client Application or Service

Application Logic

Data

MongoDB Stitch

Serverless platform that allows developers to focus on innovation rather than plumbing, services orchestration, and boilerplate code

Application Logic

Services and APIs

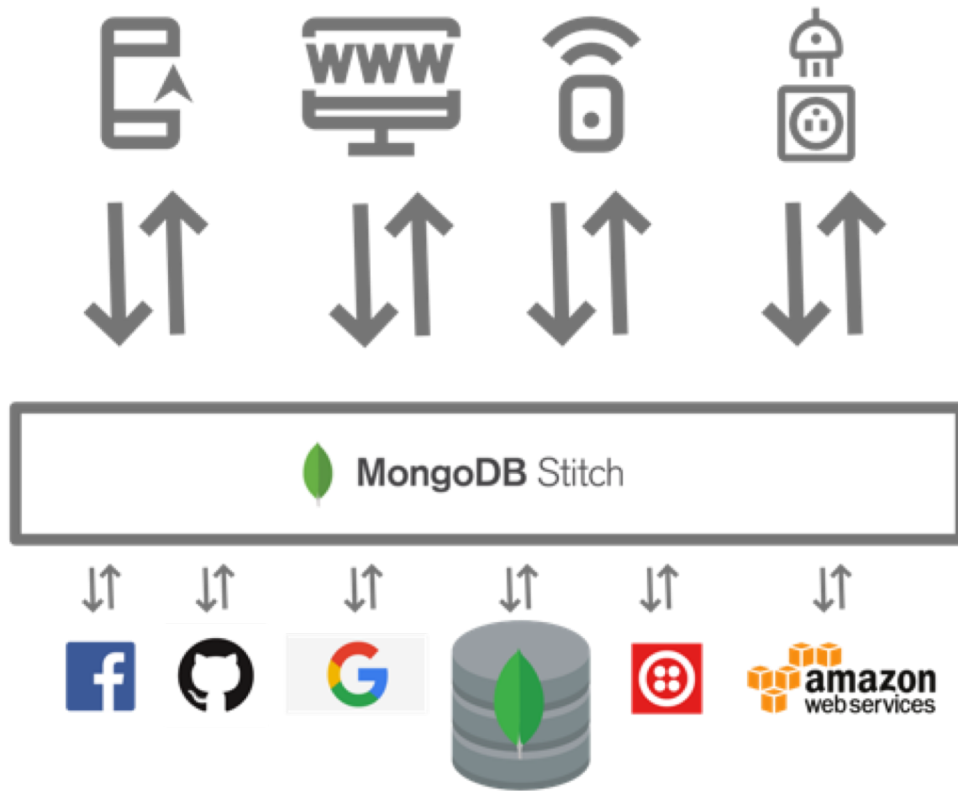
MongoDB Atlas

Rapidly deploy, dynamically scale, and distribute databases across regions and cloud providers



Cloud Infrastructure

MongoDB Stitch Serverless Platform



Streamlines app development with simple, secure access to data and services from the client with thousands of lines less code to write and no infrastructure to manage.

Getting your apps to market faster while reducing operational costs.

MongoDB Stitch Serverless Platform – Services



Stitch QueryAnywhere

Brings MongoDB's rich query language safely to the edge

iOS, Android, Web, IoT



Stitch Functions

Integrate microservices + server-side logic + cloud services

Build full apps, or Data as a Service through custom APIs



Stitch Triggers

Real-time notifications let your application functions react in response to database changes, as they happen



Stitch Mobile Sync

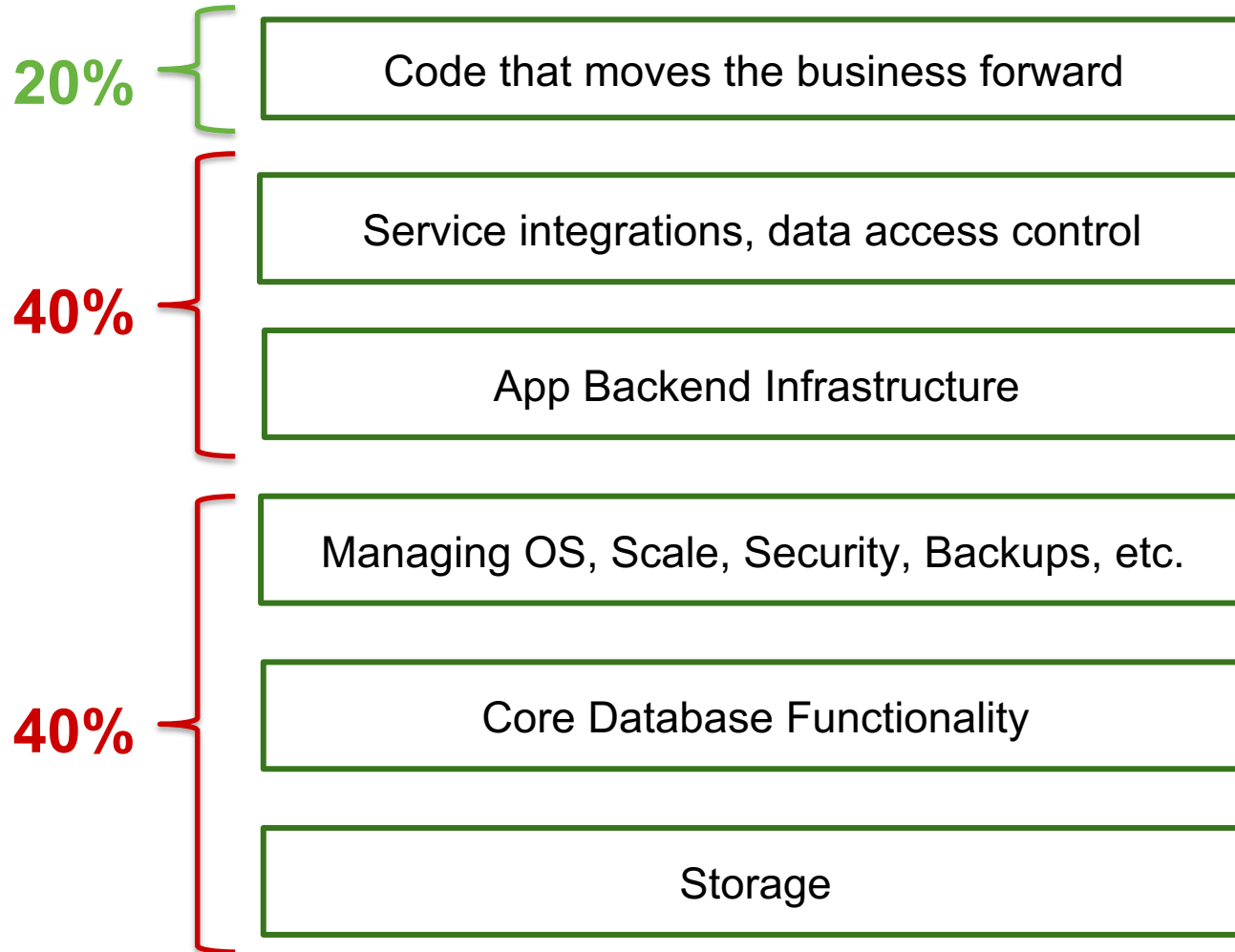
Automatically synchronizes data between documents held locally in MongoDB Mobile and your backend database

(beta)

Streamlines app development with simple, secure access to data and services from the client with thousands of lines less code to write and no infrastructure to manage – getting your apps to market faster while reducing operational costs.

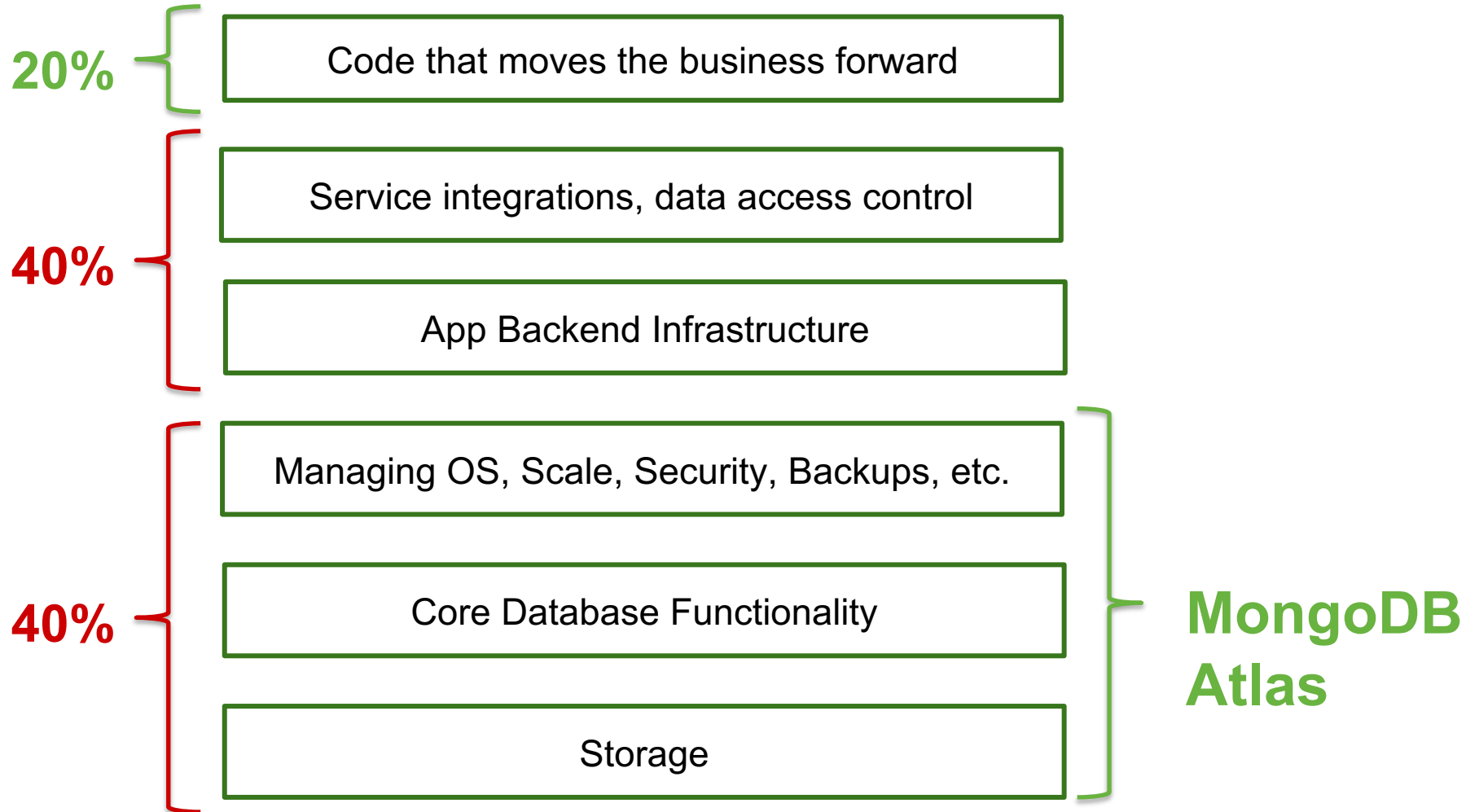
Focus Your Energy Where You Can Make a Difference

Time



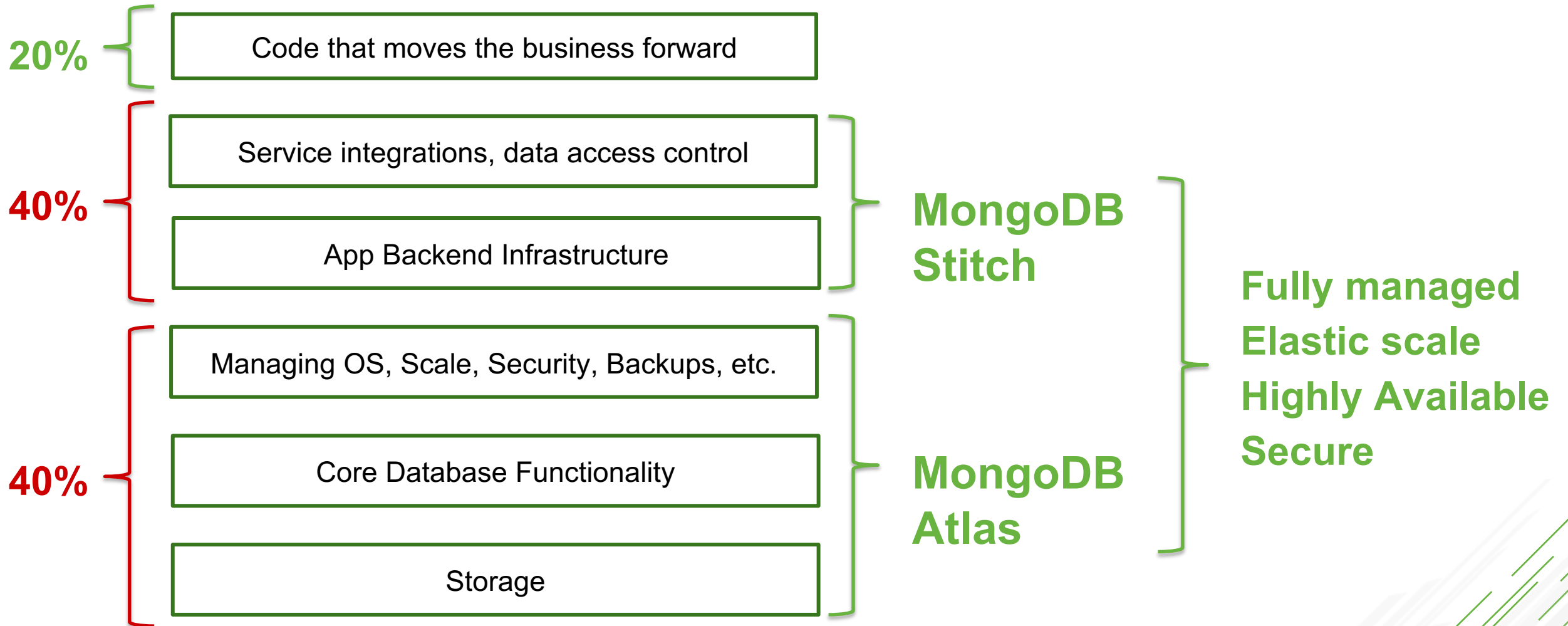
Focus Your Energy Where You Can Make a Difference

Time



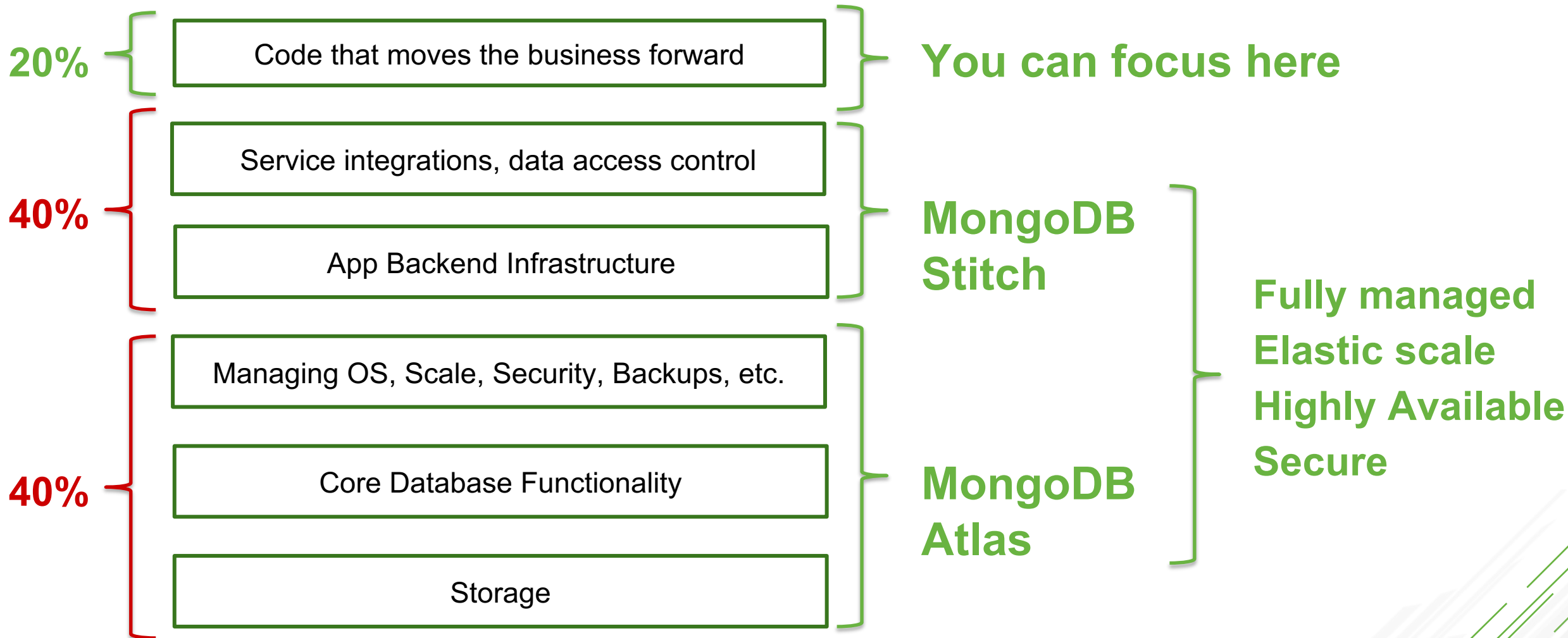
Focus Your Energy Where You Can Make a Difference

Time



Focus Your Energy Where You Can Make a Difference

Time

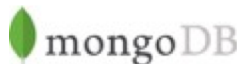


Save weeks of development and thousands of lines of code

Backend

Data Access

Frontend



Without Stitch

- Provision backend server
- Install runtime environment
- Add code to make backend HA
- Add code to scale backend
- Monitor & manage backend infrastructure
- Code REST API for frontend to use backend
- Code backend application logic

- Code user authentication
- Code data access controls

- Code application frontend
- Code against each external service API
- Continuously poll database for changes

With Stitch

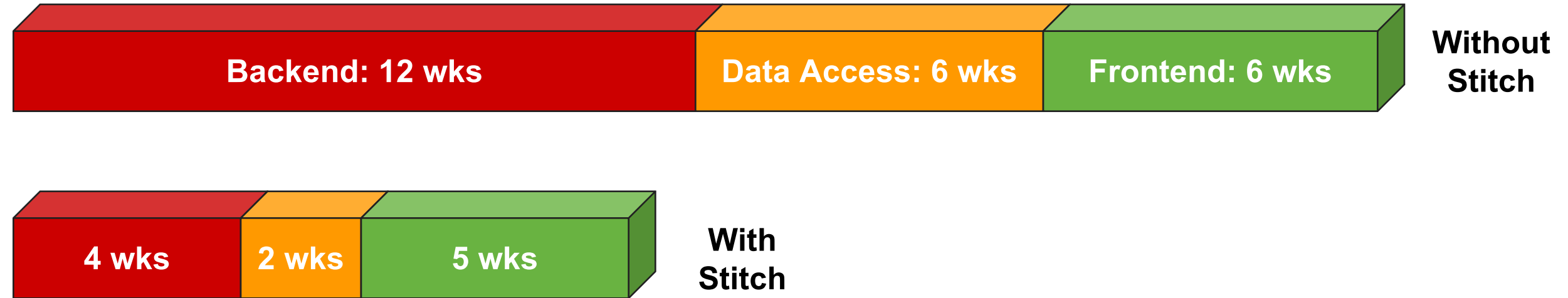
Handled automatically by Stitch and Atlas

Provide JS code for Stitch Functions

Simple JSON Config

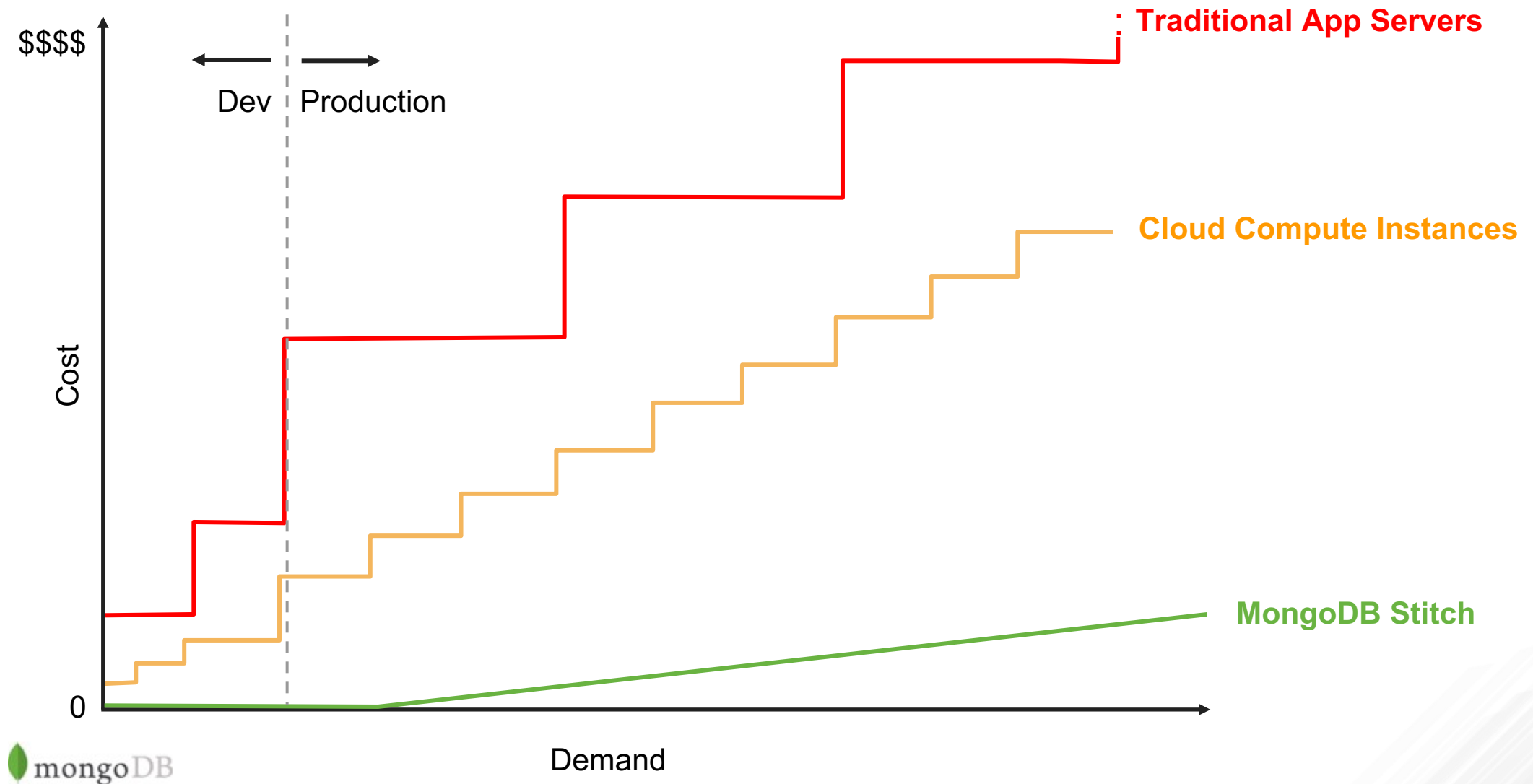
Code frontend using single SDK/API to access backend services

Development Time With/Without Stitch

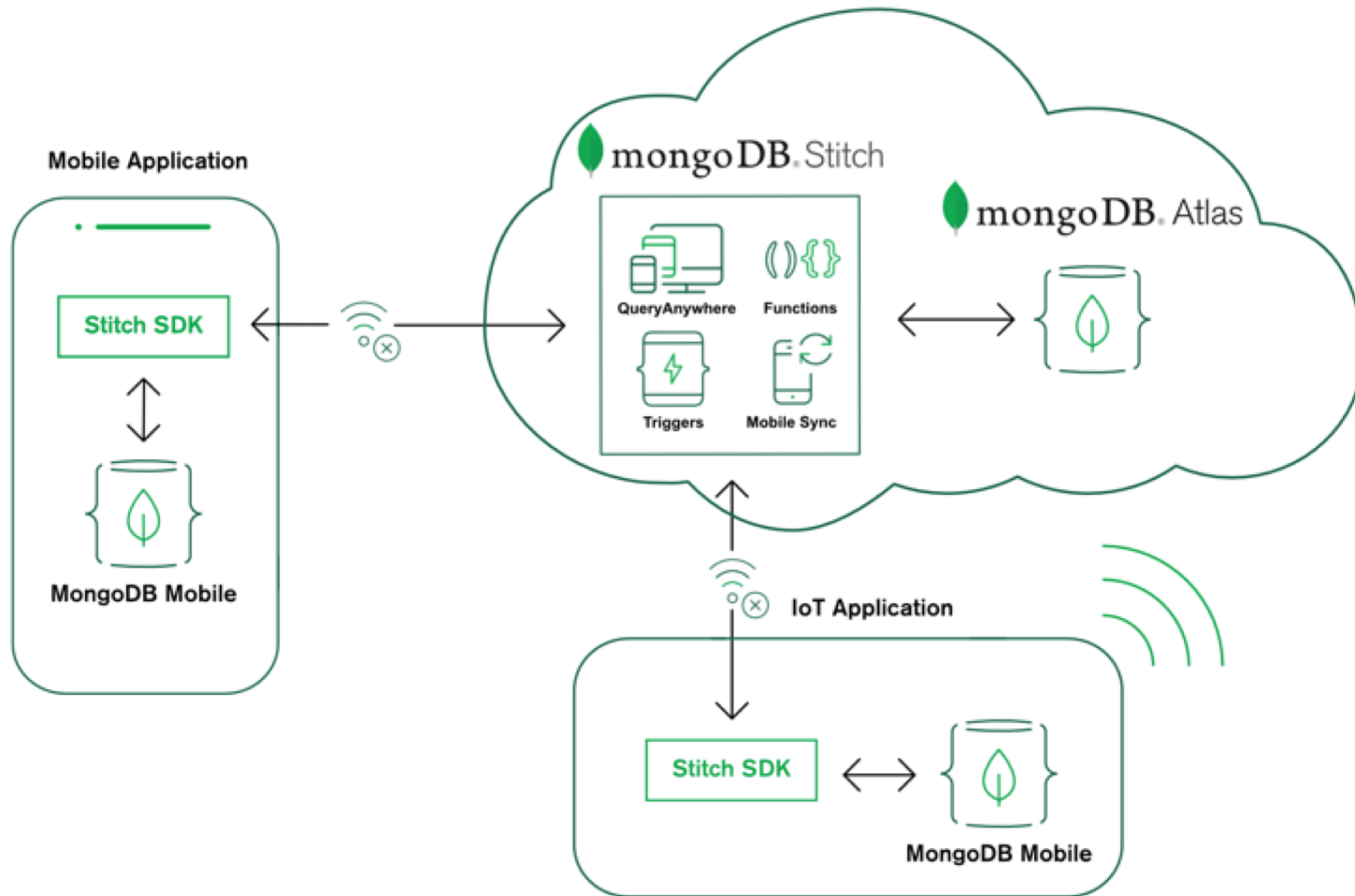


Cut development time in half

MongoDB Stitch Reduces Infrastructure Costs



MongoDB Mobile



Brings the power of MongoDB to your device

Same database, same access patterns – from IoT to Mobile to Web.

MongoDB Mobile Benefits



Use the #1 modern database everywhere

Same database in your client device and backend

Full MongoDB query language, including aggregations

Learn a **single query syntax**, use everywhere



Code once, run anywhere

Same query syntax for frontend and backend
SDKs available for **iOS** and **Android**



Build faster, online/offline apps

Local data access avoids network delays, delivering responsive apps

Secondary indexes allow you to efficiently query the same data in many ways

Progressive apps continue to run key features when offline



Sync your data everywhere

Stitch Mobile Sync

Mobile → backend

Backend → mobile

Always-on, offline changes synced when back online

Conflict detection

Get Started

Try out MongoDB Stitch [on the free tier](#)

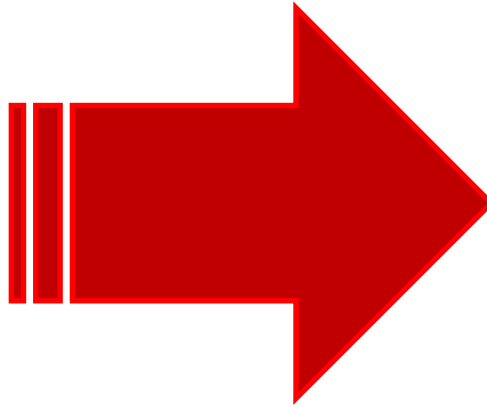
Take MongoDB Mobile for a [spin](#)

Gestione Documentale

Content Management - Contest

Solutions:

- Document Mgt.
- Web
- Collaboration



Challenges:

- Variety of content
- Volume of content
- Dynamic content

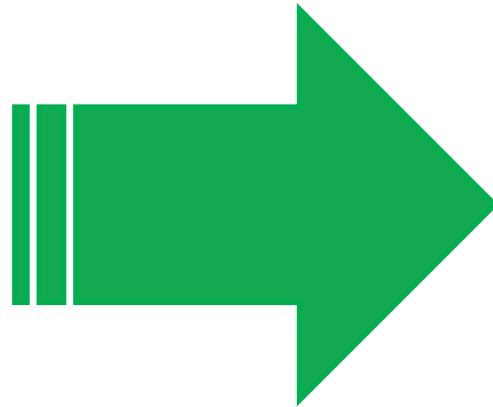
Problems:

- Mobile App
- Big Data
- Variety

How MongoDB can help with these problems?

Challenges:

- Variety of content
- Volume of content
- Dynamic content

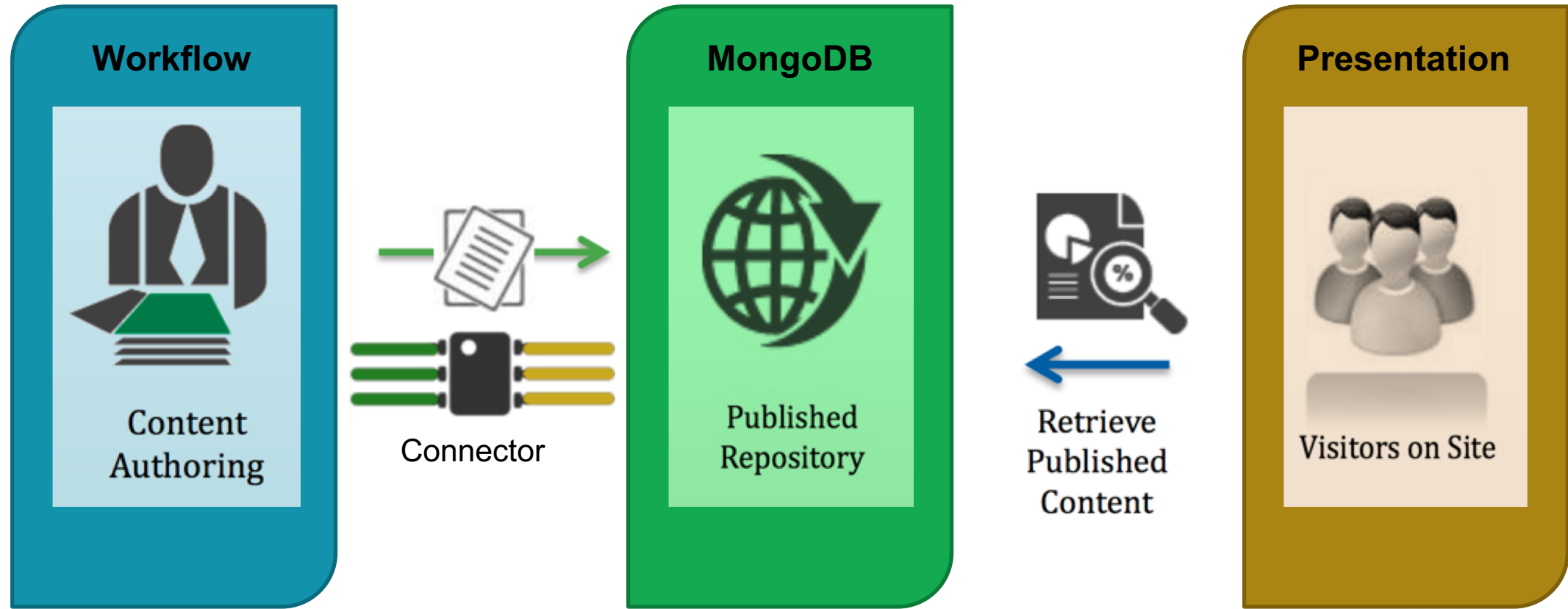


Why MongoDB?

Solution:

- Dynamic Schema + Flexible Data Model
- Sharding (horizontal scaling)
- High performance + native replication (locality and HA)

Solution Architecture





Social Publishing Platform

Media company relies on MongoDB consulting services to handle Super Bowl traffic

Problem

Adobe Social needed to ensure smooth-running ops for their MongoDB app during high-volume activities

Before Super Bowl, needed to confirm that the app would stand up to the load

Potential implications for new business, including MLB

Solution

Enlisted MongoDB consulting services to ensure uptime and smooth running ops during game time

MongoDB consulting engineer reviewed the deployment and made recommendations for optimization

Results

Adobe Social met the high volumes of the Super Bowl, its biggest traffic day, without incident



Modern Publishing Platform

Leading source of business news overhauls their entire CMS in a fraction of the time and cost of their old approach

Problem

Forbes struggled to keep up with the quickening pace of modern media with old, closed systems

System outages were common, and changes to the underlying architecture of their content management system were costly and challenging

Solution

Rebuilt their entire content management system on MongoDB; they were able to launch a brand new mobile site with just 1 full-time engineer

Leverage MongoDB's JSON document model to continuously and painlessly evolve their sites as they please with less downtime

Use MongoDB to scale across geographies to accommodate distributed writers, leverage MongoDB analytics for analysis on readership

Results

CMS built on MongoDB was built in 2 months; mobile site in 1. Both with small development teams.

Mobile traffic went from 5% of forbes.com to 50%

Forbes can now analyze reader behavior as it happens and put the right content in front of readers within seconds



Archive Database

To better archive every post in history, Craigslist migrated over 2 billion documents (5 TB) to MongoDB from MySQL

Problem

Every post in the history of Craigslist was originally stored in MySQL, but simple schema changes on the vast archive would take months to complete

New features that the business wanted had to be delayed due to continued maintenance work on MySQL

Solution

Craigslist migrated to MongoDB for its flexible data model, which was easily able to handle the various different formats of Craigslist postings

MongoDB's auto-sharding capabilities eased operational pain points and allowed for scale-out on commodity hardware

Replica sets provided automatic failover and the ability to isolate analytical workloads on secondaries

Results

Craigslist currently has a MongoDB deployment designed to hold over 5B documents & > 10 TB of data

Schema change times reduced from months to just hours

Increased performance with the ability to isolate analytical workloads on secondaries in a MongoDB replica set

Internal Use Only



Content Management

Centralized data hub for all data sources for lab and clinical analysis

Problem	Solution	Results
Need to develop a central repository for R&D teams to share and extract internal research to increase productivity RDBMS not able to meet requirements for storing and accessing large amounts of unstructured data Unable to slice and dice the research data in many different ways for labs and clinical analysis	Flexible data model made it easy to add new data types and sources Expressive query language made it possible to query arrays; ability to update data in place allowed developers to modify data by any field or data type, including geospatial and text Given the criticality of this application, GlaxoSmithKline enlisted a Dedicated Consulting Engineer from MongoDB	Launched a centralized data hub called “Target Validation Platform” to create a center of excellence around unstructured data from a variety of diverse research sources to speed up R&D Organization plans to open source data hub Reduced risk as the DCE reviewed application design patterns and implemented deployment best practices

Internal Use Only



Vehicle History Database

Replatforms core vehicle history database on MongoDB, increasing performance 10x

Problem

Core business – vehicle history info— built on an obsolete OpenVMS relational database; key/value structure didn't translate to modern RDBMS

Couldn't build modern apps on this obsolete core

Analytic workloads slowed down the production system

Solution

Vehicle data accommodated by extensible MongoDB document model

Elastically scalable to hundreds of servers with sharding for 13B+ documents

Replication separates workloads and provides built-in redundancy

Results

10x increase in performance

Scales to handle 13B+ documents, growing at 1B+ per year

100% database uptime

Reference Architecture per IoT



IoT App Running on MongoDB Atlas

Biotechnology giant uses MongoDB Atlas to allow their customers to track experiments from any mobile device

Problem

Thermo Fisher is developing Thermo Fisher Cloud, one of the largest cloud platforms for the scientific community on AWS

For scientific IoT applications, internal developers need a database that could easily handle a wide variety of fast-changing data

Each experiment produces millions of “rows” of data, which led to suboptimal performance with incumbent database

Thermo Fisher customers need to be able to slice and dice their data in many different ways

Solution

MS instrument Connect allows Thermo Fisher customers to see live experiment results from any mobile device or browser

MongoDB’s expressive query language and rich secondary indexes provide flexibility to support both ad-hoc and predefined queries to support customers’ scientific experiments

Deployed MongoDB using MongoDB Atlas, a hosted DB service running on Amazon EC2

Results

Thermo Fisher customers now can obtain real-time insights from mass spectrometry experiments from any mobile device or browser; not possible before

Improved developer productivity with 40x less code in testing with MongoDB when compared to incumbent databases

Improved performance by 6x

Easy migration process & zero downtime. Testing to production in under 2 months



IoT / Inventory Management

SureCheck mission-critical app powered by MongoDB
enables business owners to meet FDA regulated standards

Problem

SureCheck combines software, a mobile tablet and built-in temperature measuring devices to collect, track and store food safety data for anytime and anywhere access

Struggled with SQL Server to ingest data and generate reports for clients and FDA within required SLA

Unable to scale to meet platform growth, or roll out new features at speed demanded by the business

Solution

Re-built platform on MongoDB, using flexible model to ingest and store time series food safety data

Met performance and uptime goals by implementing sharding for horizontal scalability and replication for HA

Deployed Ops Manager to provision, monitor, back up, and scale MongoDB

Results

Faster business innovation

Utilized sharding to ingest the data quickly; secondary indexes and aggregation framework to analyze large data sets and generate the reports within required SLA

Fully scalable, future-proof platform built to grow predictably with their customer base

Increased customer retention and improved customer satisfaction

Internal Use Only



Real-Time Analytics

Reduced the time to introduce new drug discovery tests from 6 months to 3 weeks

Problem

Testing instruments have gotten better at capturing complex and varied genetic data, but legacy databases struggle to store and use it all

Team needed to change the database schema every time a new lab instrument was introduced, holding up research by 3-6 months

Solution

MongoDB can easily capture the variety of data generated by genetic tests via flexible data model

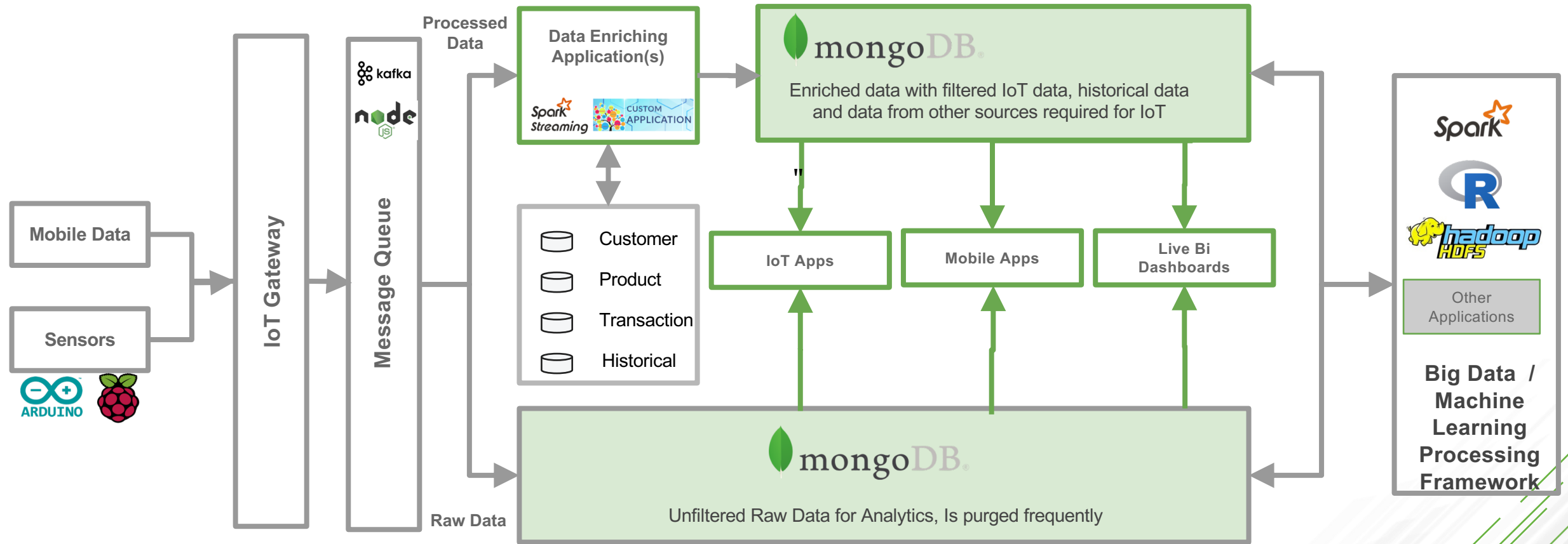
MongoDB redesign means that database does not hold up ability to add new genetic test instruments

Results

New tests can be run in weeks versus 3-6 months using just Oracle

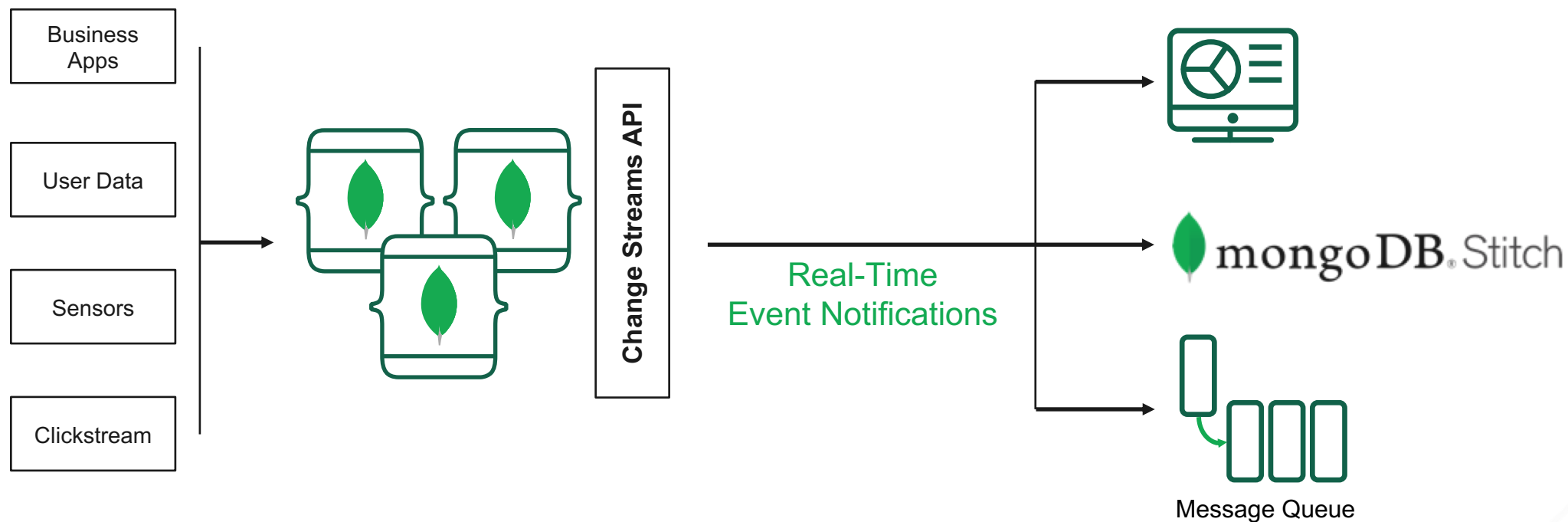
Reduce the time it takes to introduce a new drug – a big difference to patients

IoT Reference Architecture





Versatile: MongoDB Change Streams



Enabling developers to build
reactive, real-time services



BOSCH

Invented for life

IoT

Leading engineering company turns to MongoDB consultants to restore its primary storage backend for IoT projects

Problem

Bosch uses MongoDB as the backend for all Internet of Things projects

A bad ad hoc test aggregation query took down the whole deployment

Performance slowed to a crawl, with single queries running for 10 hours

Solution

Bosch turned to MongoDB consulting services to perform a thorough performance review and analysis in order to resolve the crisis

MongoDB provided a schema review, and analysis of query structure, indexing, and overall deployment health

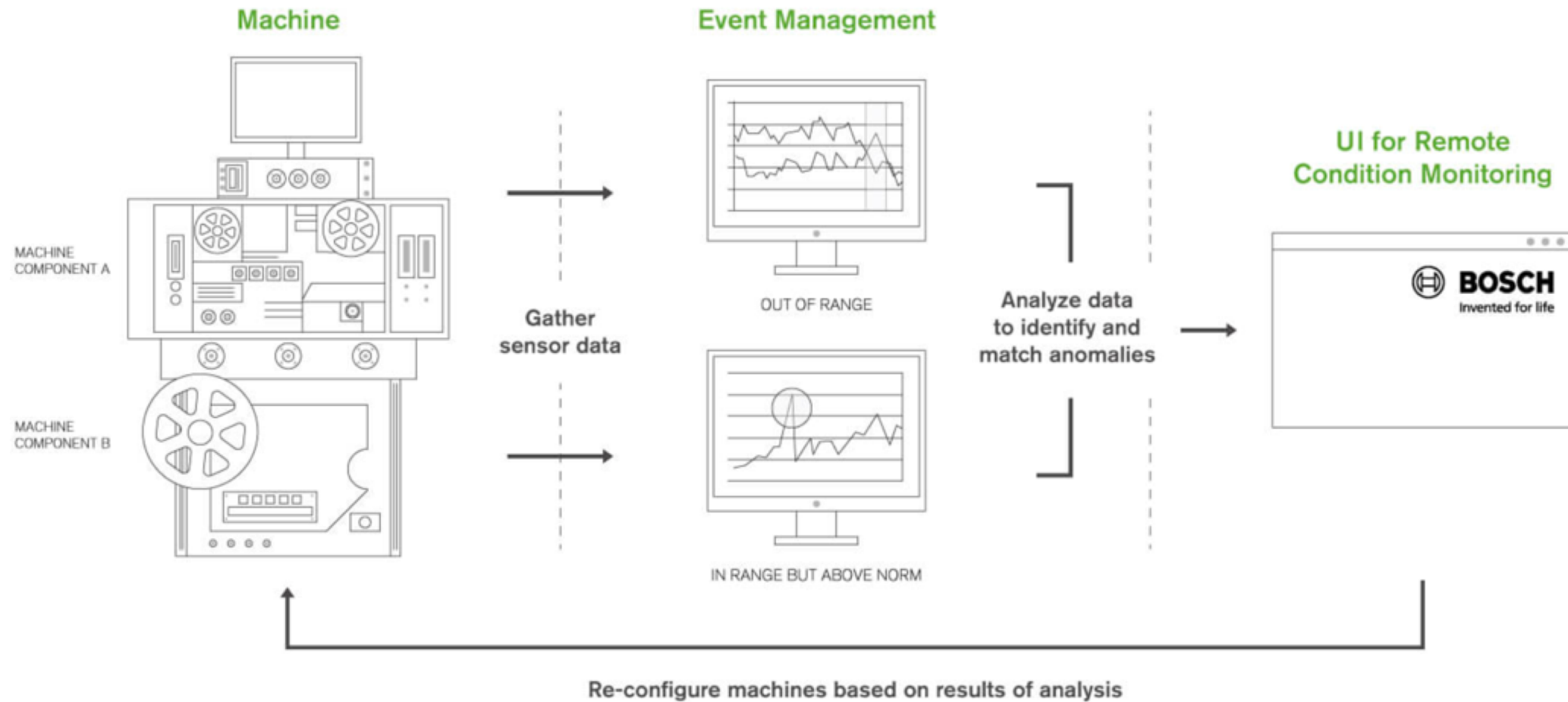
Results

MongoDB's tuning stabilized the deployment

Aggregation query time on 2.5TB of data reduced from 10 hours to 3 seconds

High confidence in MongoDB restored across Bosch Software Innovations

Bosch IoT - Flow



Rich Functionality for IoT/Time Series

Sample Document

Expressive Queries & Secondary Indexes

- Find all devices of type Thermostat in Newark
- Find the time samples for yesterday from 12:00-12:05pm for all Thermostats in sector 8

Geospatial

- Find temperatures for all devices within 100 ft of [40.7, -74.2]
- Find the Location of all devices within the polygon representing zip code 07114

Text Search

- Find all log entries that mention “hot day”

Aggregation

- Find the devices that recorded an average temperature from 12-12:05pm of over 65 degrees

Native Binary JSON support

- Add an additional temp entry without sending the whole document
- Select just the time entry for 1:05pm
- Find all device entries and sort all of their date types properly

Left outer join (\$lookup)

- Find the manufacture date (in another collection) for all devices with error statuses, sorted by date

Graph queries (\$graphLookup)

- Find all devices and subcomponents for DeviceId 100

```
Device {
  DeviceId : 100,
  Name : "Temp8",
  Type: "Thermostat",
  Location : "EWR",
  Coordinates: [40.702675, -74.179471]
  PlantName : "EWR 10",
  Sector: 8
  ForemanLogEntry: "Very hot day caused..."
  Samples: {
    StartTime : ISODate("2016-09-05"),
    EndTime: ISODate("2016-09-06"),
    Entries: [
      {t: ISODate("...12:00"), temp: 65, ...}
      {t: ISODate("...12:01"), temp: 66, ...}
      ... ]
  }
  SubcomponentIDs: [105, 207, 308, ...]
}
```

```
{
  plane_id: "3209",
  ts: ISODate("2017-07-03T16:00:00.000Z")
  metrics: { engine_fuel: 99 }
},
```

```
{
  plane_id: "3209",
  ts: ISODate("2017-07-03T16:01:00.000Z")
  metrics: { engine_fuel: 98.5 }
},
```

```
{
  plane_id: "3209",
  ts: ISODate("2017-07-03T16:02:00.000Z")
  metrics: { engine_fuel: 98 }
}
```

. . .

```
{
  plane_id: "3209",
  ts: ISODate("2017-07-03T16:59:00.000Z")
  metrics: { engine_fuel: 69 }
}
```

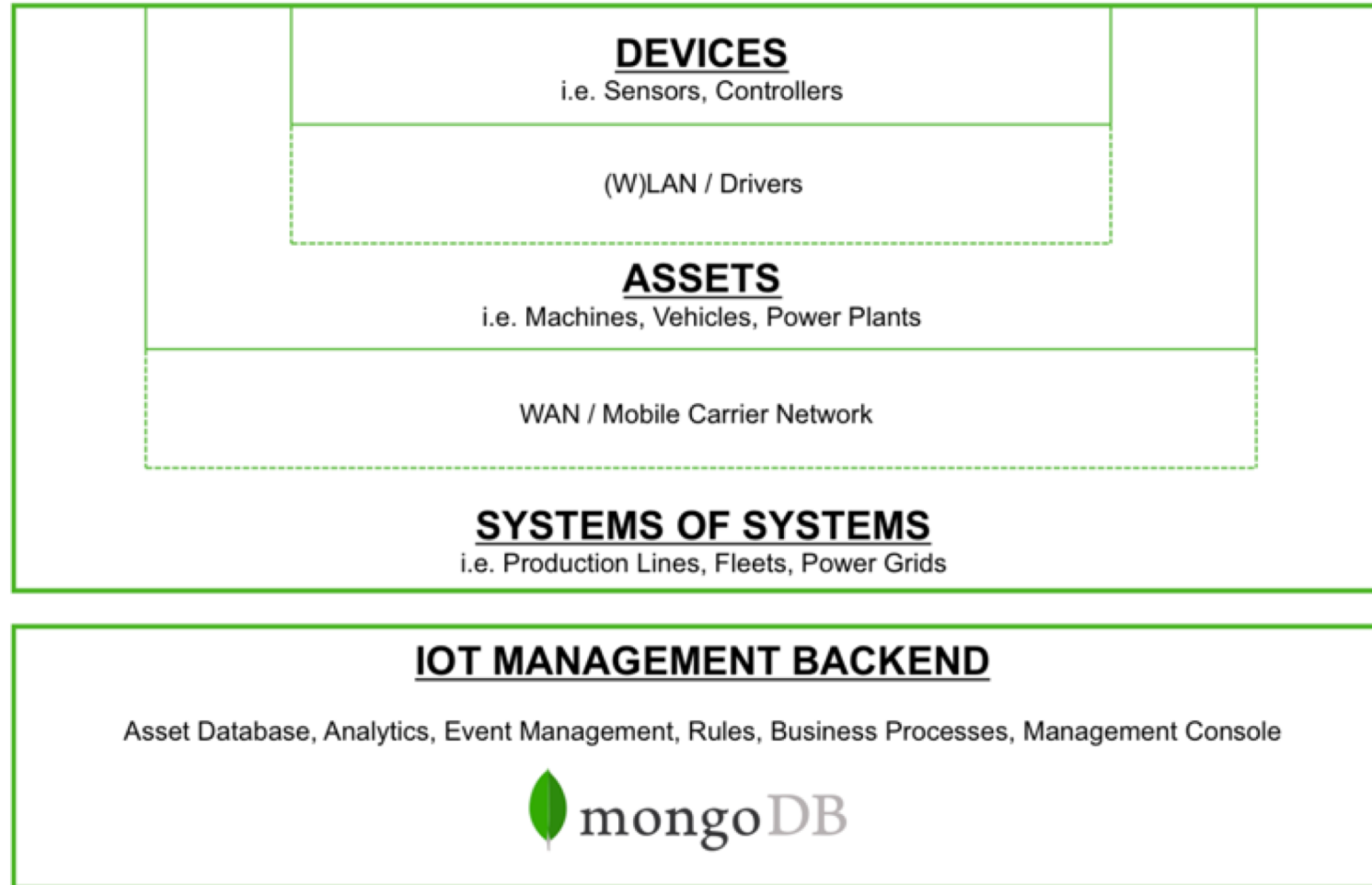
60:1

```
{
  plane_id: "3209",
  hour: ISODate("2017-07-03T16:00:00.000Z"),
  incInSeconds: 60,
  metrics: {
    engine_fuel: {
      values: [ 99, 98.5, 98, ... 69 ],
      avg: 81.4
    },
    turbine_temp: {
      values: [ 1900, 2200, 2150, ... ]
      avg: 2170
    },
    flight_mode: {
      values: [ "A", "A", "A", "X", "X"... ]
    }
    ...
  }
}
```

- Less docs – space savings
- Write performance - less index entries
- Queryable & better analytics support

BUCKETING OPTIMIZATION of TIME SERIES DATA

Bosch IoT - Architecture





Predictive & Curative Maintenance

French leader airline company leverages data to optimize the duration of flight of their A380 aircrafts.

Problem

Air France needs to increase revenue coming from A380 aircrafts. It means increase the duration of flight for each 380.

Maintenance, Repair and Operation (MRO) is the most time consuming process which sticks the aircraft "on ground".

Legacy databases unable to manage high volume and different types of data.

Hadoop unable to ensure real time analytics.

Solution

IoT platform built on MongoDB leveraging its flexible data model, horizontal scalability and aggregation framework to manage.

Each year, 9 To data are generated from ten A380 and 24k sensors on each A380. Each flight creates 1,6 Go data. It will be 10x more when all sensors will be used.

The data stored in MongoDB feeds nightly batch jobs in Hadoop, the results of which go back into MongoDB to enhance real time analysis.

Results

After each flight, data are sent in less than 1 hour to maintenance team in Toulouse.

Since July 2015, one important failure was anticipated (gas pump) → several millions euros saved.

For maintenance curative, breakdown identifications time decreases from 6 hours to 5 minutes.

See public article here:

<http://www.usine-digitale.fr/article/air-france-klm-anticipe-les-pannes-de-ses-avions-a380-au-big-data.N365483>

Appendix - GDPR

Discover Retention of Personal Data



“Information to be Provided”

GDPR Article 13 (clause 2a)

“the period for which the personal data will be stored, or if that is not possible, the criteria used to determine that period.”

Time to Live (TTL) Indexes



- Automates the expiry of data from the database
- Define TTL index against a date field, specify the expiration period
- Background process deletes the document once retention period expires
- Simplifies enforcement, with lower overhead

Defend General Security Requirements



“Security of Processing”

GDPR Article 32 (clause 1)

“....the controller and the processor shall implement appropriate technical and organisational measures to ensure a level of security appropriate to the risk, including inter alia as appropriate:

- a. **the pseudonymisation and encryption** of personal data;*
- b. the ability to ensure the ongoing **confidentiality, integrity, availability and resilience** of processing systems and services;*
- c. the ability to **restore the availability and access to personal data** in a timely manner in the event of a physical or technical incident;*
- d. a process for regularly testing, assessing and evaluating the effectiveness of technical and organisational measures for ensuring the security of the processing.”*

Defend Access Control



“Data Protection by Design and by Default”

GDPR Article 25 (clause 2)

“The controller shall implement appropriate technical and organisational measures for ensuring that, by default, only personal data which are necessary for each specific purpose of the processing are processed”

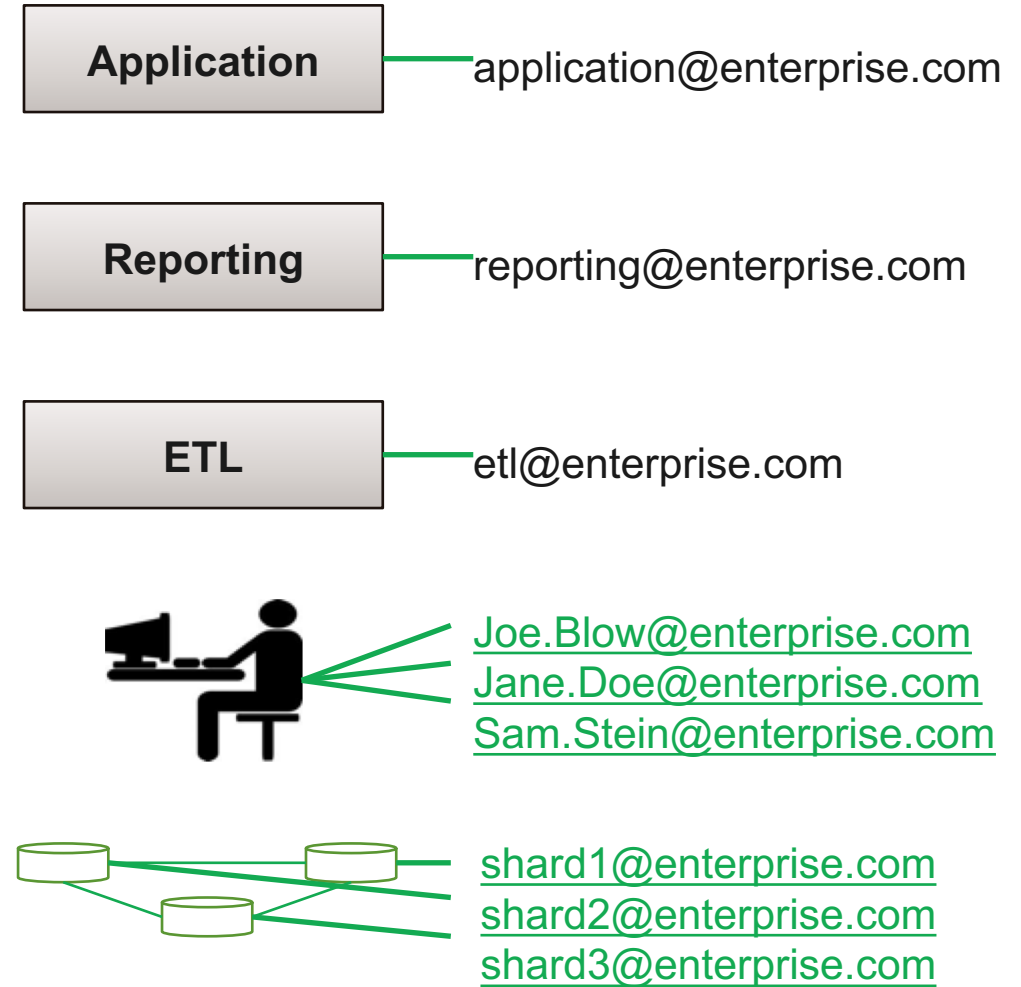
“Processing Under the Authority of the Controller or Processor ”

GDPR Article 29

“The processor and any person acting under the authority of the controller or of the processor, who has access to personal data, shall not process those data except on instructions from the controller....”

Authentication

- Confirming identity of any client accessing the database
- Multiple options
 - Built in authentication: challenge/response (SCRAM IETF RFC 5802)
 - x509 certificates
 - Integration with corporate authentication infrastructure (Enterprise Advanced)
 - Enforce centralized access controls
 - LDAP
 - Kerberos



Authorization

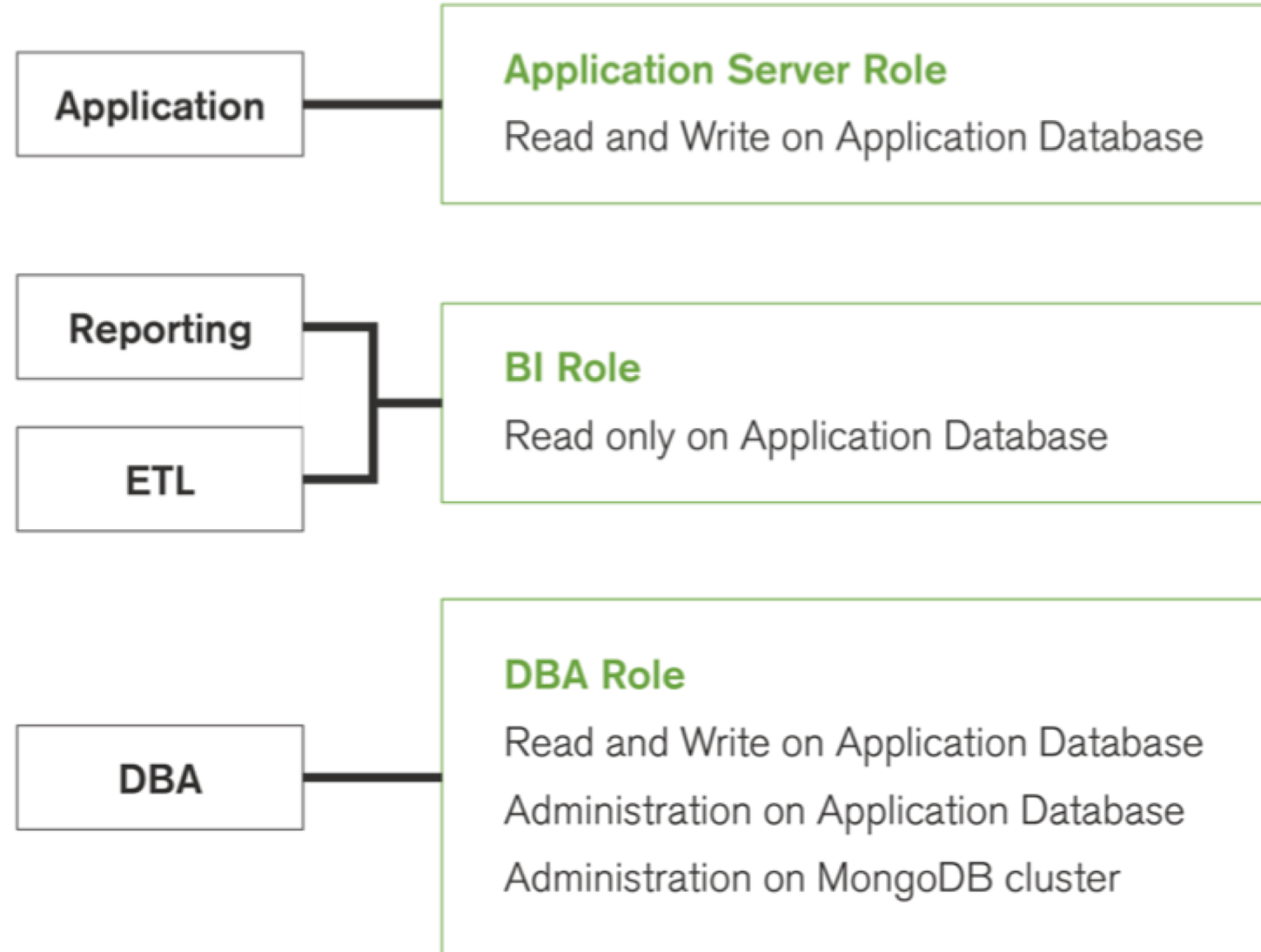


Role Based Access Control

- read, readWrite, dbAdmin, clusterAdmin, root, etc..

User defined roles

- Customized roles based on existing roles and privileges
- Delegate across teams



Defend Pseudonymisation & Encryption



“Security of Processing”

GDPR Article 32 (clause 1)

“.... shall implement appropriate technical and organisational measures to ensure a level of security appropriate to the risk....:

*a. the **pseudonymisation** and **encryption** of personal data;”*

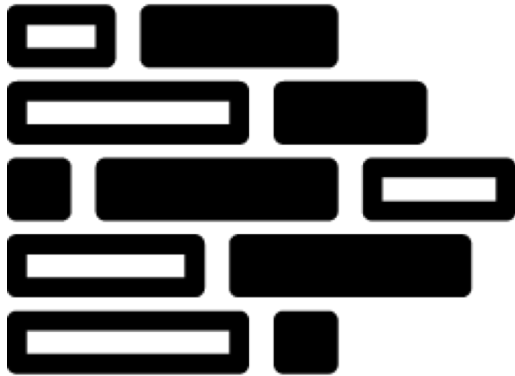
“Communication of a Personal Data Breach to the Data Subject”

GDPR Article 34 (clause 3a)

Communication of a breach to a data subject **is not required** if the data is rendered unintelligible, i.e. via encryption



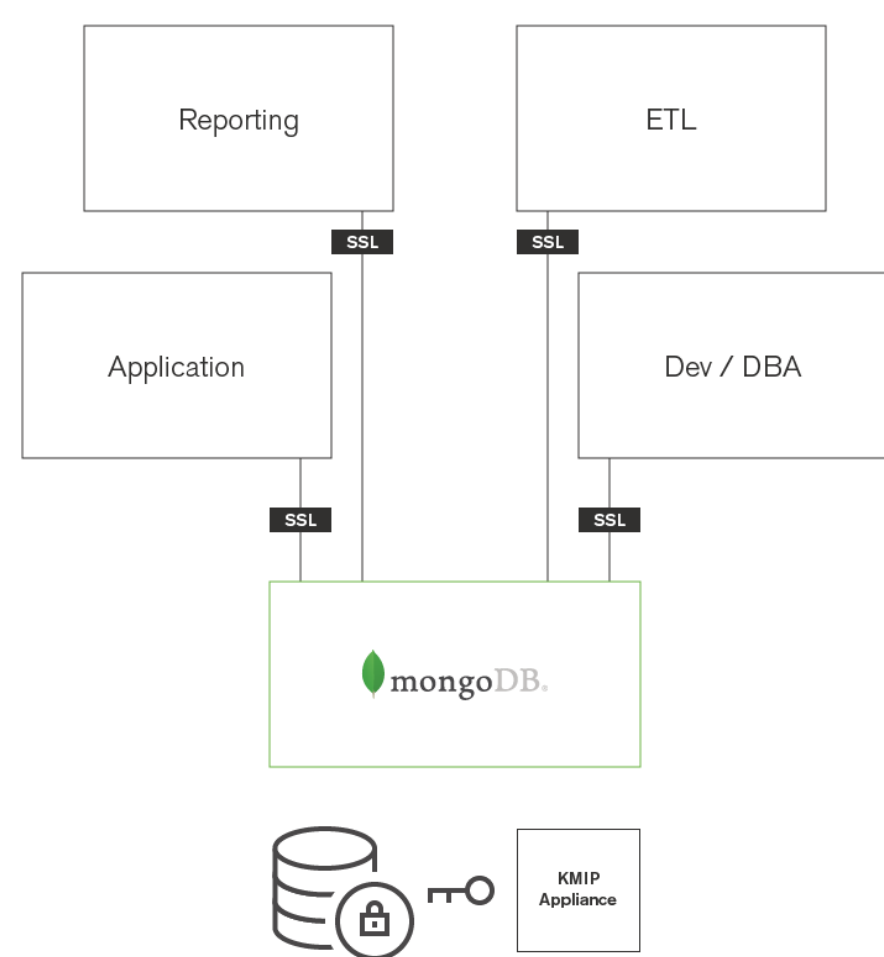
Read-Only Views & Log Redaction



- MongoDB allows administrators to define dynamically generated views that expose a subset of data from the underlying collection
 - Exclude or mask fields
 - Reduces risk of sensitive data exposure
 - Views do not affect source collections
 - Separately specified permissions levels
- Log redaction strips PII from diagnostic logs used by technical staff (Enterprise Advanced)



Encryption: In Motion & At Rest



- TLS encryption to protect network traffic
- MongoDB Atlas offers encryption at rest with cloud providers disk & volume encryption
- MongoDB Enterprise Advanced offers Encrypted Storage Engine
 - Reduces surface attack area
 - Reduces performance & management overhead
 - AES 256 + FIPS compliant
 - 1 master key per server, 1 key per database
 - KMIP or keyfiles

Defend Sovereignty: Data Transfers Outside of the EU



GDPR Article 45 (clause 1)

“A transfer of personal data to a third country or an international organisation may take place where the Commission has decided that the third country, a territory or one or more specified sectors within that third country, or the international organisation in question ensures an adequate level of protection.”



MongoDB Zones



- Partition data across distributed clusters based on data locality policies
 - Adhere to data sovereignty requirements
 - If policies change, update the shard key range and data is automatically migrated
 - Support distributed local writes
 - Enable deployment patterns such as tiered storage
- Can be configured visually from MongoDB Ops Manager
- Cross-region replication with MongoDB Atlas

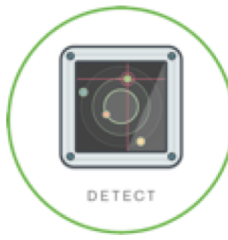
Detect Monitoring & Alerting



“Notification of a Personal Data Breach to the Supervisory Authority”

GDPR Article 33 (clause 1)

“In the case of a personal data breach, the controller shall without undue delay and, where feasible, not later than 72 hours after having become aware of it, notify the personal data breach to the supervisory authority....”



Monitoring & Alerting



- Over 100+ database-related metrics
- Visualized across charts and dashboards
- Push telemetry data via API into APM platforms
- Custom alerts so incidents do not become emergencies: pushed to collaboration & management platforms

Detect Auditing



“Data Protection by Design and by Default”

GDPR Article 25 (clause 2)

“....Each controller and, where applicable, the controller's representative, shall maintain a record of processing activities under its responsibility”

Auditing



Console, Syslog and File



Audit Trail

- MongoDB Enterprise Advanced audit log records all actions taken against the database
 - Configurable filters (commands, IP, etc) & role-based auditing
- MongoDB Atlas maintains audit log of administrative actions, exposes logs of database connections