Future Trends, AI and Predictive Technologies for Cloud Computing and HPC

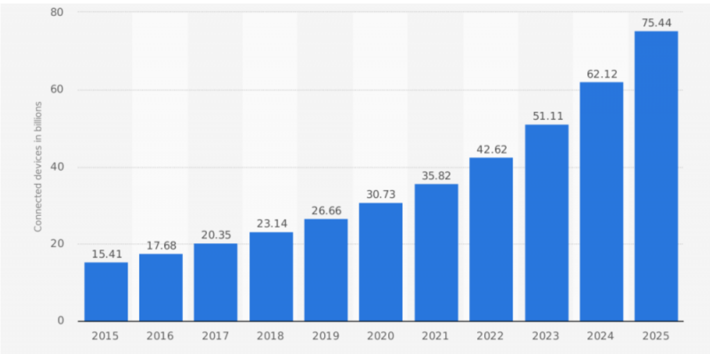
Ozalp Babaoglu

Technological and Application Trends

- Internet-of-Things
- Edge computing
- 5G
- Server-less computing
- Big data, Data analytics
- Cloud for Machine learning, Artificial Intelligence
- Machine learning, Artificial Intelligence for Cloud

Internet-of-Things

- IoT connected devices installed base worldwide



Internet of Things

- In IoT, what is interesting is not the number of devices but the data they generate

2020 — Amount of data generated per day by a	
Person	1.5GB
Smart Hospital	3TB
Self-driving car	4TB
Connected airplane	5TB
Connected factory	3PB

From Cloud to Edge Computing

- *Edge*: extremity of a typical enterprise network of IoT-enabled devices — sensors, actuators (billions)
- *Fog*: computing devices close to the edge of the IoT network to process the large amounts of generated data (millions)
- *Core*: backbone network connecting geographically dispersed fog networks (tens of thousands)
- *Cloud*: provides the storage and processing capabilities for the massive amounts of aggregated data originating at the edge, hosts applications to interact with and manage the IoT-enabled devices (thousands)

Edge Computing

- Cloud computing usually involves ***thousands of physical servers*** running in centralized data centers close to the ***core of the Internet***
- ***Edge computing*** directs specific processes away from centralized data centers to points in the network ***close to users***, ***devices*** and ***sensors***
- Edge computing is ***essential*** for IoT, as it allows ***collection and processing*** of huge amounts of data ***in real-time*** with ***low latency***
- Edge computing helps IoT systems ***lower connectivity costs*** by sending ***only the most important information*** to the cloud, as opposed to raw streams of sensor data

5G

- 5G is the ***fifth generation*** technology standard for cellular networks
- Evolution of 4G
- 5G is a ***key enabling technology*** for both IoT and edge computing
- 5G provides the fabric for ***device-to-device*** and ***device-to-edge*** communication
- 5G delivers the ***higher speeds*** and ***broader bandwidths*** required to support analytics and control functions in real-time where the data is created and actions are taken

5G

- With 5G, retailers will benefit from ***up-to-date information*** on consumer ***buying trends***, factories will be able to perform ***predictive maintenance*** on equipment that's about to fail, cellphone carriers will be able to support ***augmented reality***
- As 5G deployment takes place, hybrid cloud systems will increasingly take advantage of opportunities to ***perform computations at the edge***
- 5G frequency bands
  - Low-band 5G uses 600-700 MHz to achieve 30-250 Mbps speed
  - Mid-band 5G uses 2.5-3.7 GHz, to achieve 100-900 Mbps speed
  - High-band 5G uses 25-39 GHz to achieve 1 Gbps speed

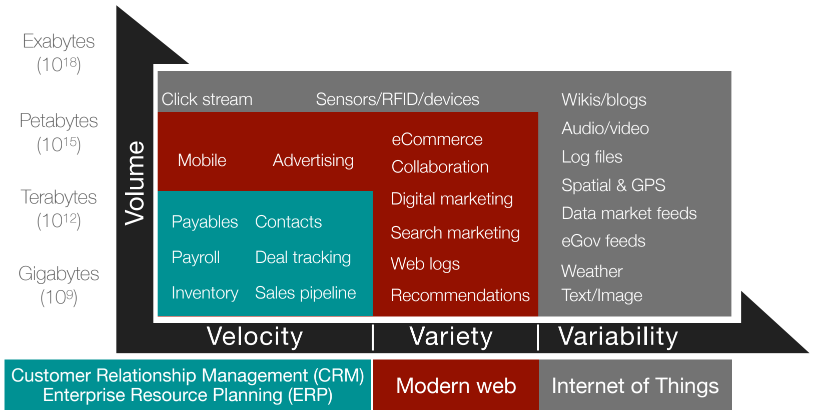
Big Data

- An important application of cloud computing is to perform ***data analytics*** — the science of ***extracting information*** from large volumes of data
- ***Cloud analytics*** refers to a service model where one or more components of data analysis and business intelligence operations are implemented in the cloud
- Combined, big data and the cloud can offer tremendous value to companies by making it easier to ***track***, ***analyze*** and ultimately ***act on*** insights

Big Data

- In the pre-cloud era,
  - It was tempting for business processes to become ***isolated*** in silos making coordination ***cumbersome***, sharing ***difficult*** and transferring data, especially large amounts of it, ***slow***
  - Big data processing was ***cumbersome***, and ***expensive*** meaning that big data efforts were ***reactive***, providing insights from out-of-date archived data whereas businesses need to be ***proactive*** and be able to access, analyze and act upon the most current data

Sources of Big Data



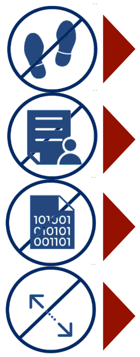
Big Data

- Big Data affects the organization of database systems
- Traditional ***relational databases*** are unable to satisfy some of the requirements of big data and ***NoSQL databases*** have emerged for many cloud applications
- ***Key-Value pair stores*** are simpler and can benefit from horizontal scaling to large clusters
- Many NoSQL stores compromise ***consistency*** in favor of ***availability***, ***partition tolerance***, and ***speed***

Machine Learning

- *Machine learning* (or *predictive analytics*) is a subfield of *Artificial Intelligence* and statistics that provides systems the ability to ***automatically learn*** and ***improve*** from experience without being ***explicitly programmed***
- The process of learning begins with a ***training*** phase where ***observations*** or ***data***, such as examples, direct experience, or instructions are processed to discover ***patterns*** so that better decisions can be made in the future

Why Learn?



1. *Learn* it if you cannot track it (e.g., AI gaming, robot control)
2. *Learn* it if you have to adapt/personalize (e.g., predictive typing)
3. *Learn* it if you cannot program it (e.g., recognizing speech/image/gestures, NL translation)
4. *Learn* it if you cannot scale it (e.g., recommendations, spam & fraud detection)

Cloud for Machine Learning and AI AWS

- *SageMaker* — fully managed platform to build, train, and deploy machine learning models
- *Rekognition* — a deep learning-based image recognition service
- *Lex* — for building voice and text chat chatbots
- *Polly* — convert text into lifelike speech
- *Comprehend* — continuously trained and fully managed natural language processing
- *Transcribe* — speech-to-text conversion with automatic speech recognition

Cloud for Machine Learning and AI Azure and Google

- *Azure Machine Learning Studio*
- *Google AI Platform*
- *TensorFlow*

Generative AI on Cloud
AWS

- AWS tools for building generative AI applications:
  - *Bedrock*: collection of foundation models from leading AI companies like AI21 Labs, Anthropic, Cohere, Meta, Mistral AI, Stability AI
  - Elastic Compute Cloud instances powered by *NVIDIA H100 Tensor Core* GPUs
  - Elastic Compute Cloud instances with up to 16 AWS *Trainium* accelerators
  - Elastic Compute Cloud instances powered by *Inferentia2* accelerators
  - *UltraClusters* consist of thousands of accelerated EC2 instances that are co-located in a given AWS Availability Zone and interconnected using a petabit-scale nonblocking network

Machine Learning and AI for Cloud Management

- ***Machine learning*** and ***predictive analytics*** provided through cloud computing are ***core capabilities*** that are useful throughout a business
- They can also be valuable in improving the ***operation*** and ***management*** of cloud and HPC systems themselves

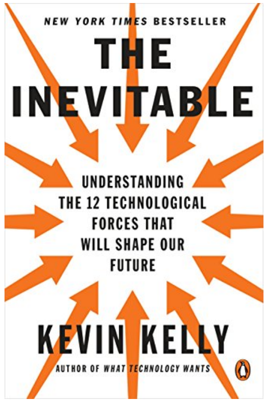
Machine Learning and AI for Cloud Management

- ***Energy efficiency***: Limit the power consumption of future HPC systems to 45MW (energy requirement of a small town of 80,000) by improving their energy efficiency
- ***Availability***: Limit the perceived failure rates of future HPC systems to the equivalent of once-per-week levels to improve their availability
- ***Management***: Improve manageability of future HPC systems by limiting reliance on human operators and facilitating semi-automatic control

Using Past Data to Predict the Future

- Scientific discovery is increasingly being driven by *data*
- The data-driven approach allows us to uncover interesting properties of processes without having to construct cause-effect mathematical models
- Easy access to *massive data sets*, *big-data analytics* tools and *HPC* have been factors fueling this trend
- The data-driven approach can be taken one step further by adding an intelligence component in the form of a *predictive computational model*
- Beyond gaining knowledge about the past from historical data, the “data-driven plus intelligence” approach can have *predictive capabilities* about future or unseen behaviors

Cognification



Penguin Books, 2016

Cognification

- *Cognification* — Transforming ordinary (dumb) objects, services or activities into their intelligent counterparts through *data* by *tapping* into existing services for the required *analytics* and *intelligence*
- Similar to *electrification* (of mechanical, manual tasks) that took place more than a century ago when electricity became ubiquitous as a commodity

Cognification

Cognified WSC Systems

- Cognification in the form of *predictive computational models* could be the software technology bridge that is necessary for achieving sustainable cloud computing and HPC
- Need *data* and *intelligence*
- Intelligence derived from *predictive* models for *workloads*, *power consumption*, *failures*
- If predictive models can be built online from *streamed data*, they can be used to *enact control* over the system in real time

Towards Cognified WSC
Predictive Models

- Technologies and services to be tapped:
  - Data analytics — streaming data management
  - Intelligence — machine learning, deep learning
- Need to build predictive models for
  - Power consumption
  - Workloads
  - Failures

Data-driven Energy Efficiency

- Based on predictive models for *power consumption*, *workloads* and *failures* built from online streamed data and on advanced hybrid optimization techniques using *Constraint Programming*
- Achieved through a *cognified dispatcher* — which jobs to run next (scheduling) and where to run them (allocation) based on
  - *intelligent consolidation*: gather as many active jobs/threads on as few physical nodes/cores as possible so that idle nodes/cores can be switched to low-power mode or powered off completely
  - *failure-aware allocation*: avoid assigning new jobs to nodes that are likely to fail in the near future
  - *energy-aware scheduling* through *power capping*: select the set of jobs to run such that their cumulative power needs do not exceed a threshold

Energy Efficiency

- Power consumption of HPC applications are often multidimensional, nonlinear and has large dynamic range
- Power-aware allocation schemes have to consider multiple measures for workloads (e.g., memory size in addition to CPU utilization) and take into account the effects of ***co-locating*** jobs on the same node
- In large Data Centers, consolidation has been facilitated to a large extent by the availability of ***virtualization*** and ***container*** technologies such as ***Docker*** and ***Kubernetes***
- Container technologies are not as widespread in current HPC systems which makes consolidation less common as an energy efficiency mechanism for them

Data-driven Availability

- Based on predictive models for *failures* and *workloads* built from online streamed data
- Achieved through
  - *adaptive checkpointing*: dynamically adjust the checkpoint interval based on predicted failure rates
  - *"just-in-time" checkpointing*: time proactive checkpoints to complete shortly before failures occur
  - *adaptive migration*: if a job that is predicted to take a long time to complete is started on a node with moderate failure probability, move it to a safer node; if the job is predicted to complete soon, leave it where it is (even if it is a node with moderate failure probability)
  - *adaptive replication:* hybrid mechanism that selects automatically between just-in-time checkpointing and *replication* while dynamically adjusting the checkpoint interval and the number of activated replicas

Data-driven Availability

- Process-level or hardware-level ***replication*** is an often-employed technique to increase availability in Data Centers
- It is less common in HPC systems for several reasons
  - *failure independence*, which is the foundation for replication is difficult to satisfy in HPC systems which tend to be more tightly coupled
  - replication often incurs high overhead in order to guarantee *replica equivalence* despite non-determinism in applications
  - hardware-level replication contributes to increasing *socket counts* and *power consumption*, which are already at elevated levels in HPC systems

Data-driven Availability

- When replication is in use to improve availability, the dispatcher tries to allocate replicas on nodes that exhibit the greatest *failure independence* as measured by predicted failure correlations between them

What We Have Done

- Built and tested *random forest ensemble classifiers* for node *failures* based on 416 features derived from a public Google dataset for a cluster of 12,453 machines over a 29-day period
- Built and tested predictive models for system *power consumption* based on data from a hybrid CPU-GPU-MIC HPC system called Eurora installed at a local data center (Cineca)
- Built and tested predictive models for *job duration* based on data from the Eurora system

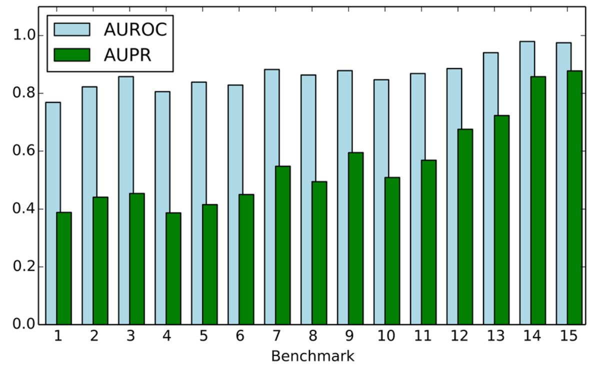
Evaluation

- Train the classifier on 15 benchmarks constructed from the Google trace
- If the “down time” of a node is longer than 2 hours, assume it is a node failure, otherwise assume node removed for a software update
- Continuous scores of the classifier are discretized (positive, negative) based on a threshold
- *Precision* — fraction of all classifications that are correctly classified as positive
- *Recall* or *True Positive Rate* — fraction of positive classifications that are correct

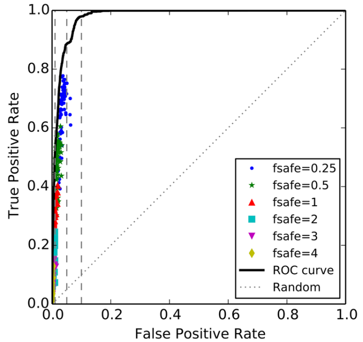
Evaluation

- The *Receiver Operating Characteristic* (ROC) curve plots the *True Positive Rate* (TPR) versus the *False Positive Rate* (FPR) of the classifier as the threshold is varied
- The *Precision-Recall* (PR) curve plots the *precision* versus *recall* (or TPR) of the classifier as the threshold is varied

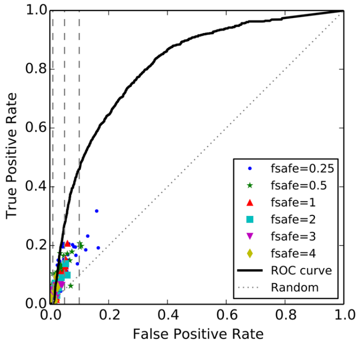
Failure Prediction
"Area under ROC" and "Area under PR"



Failure Prediction
Receiver Operating Characteristic



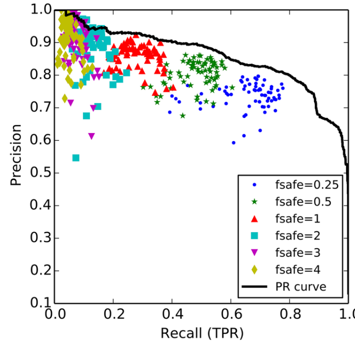
"Best case" benchmark



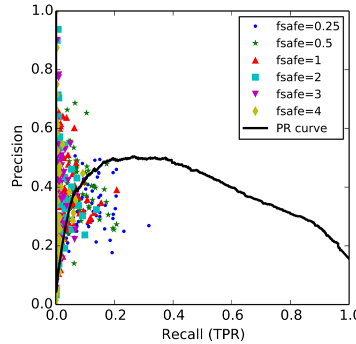
"Worst case" benchmark

- *fsafe* is a parameter of the classifier that governs sampling to compensate class imbalance

Failure Prediction
Precision Recall



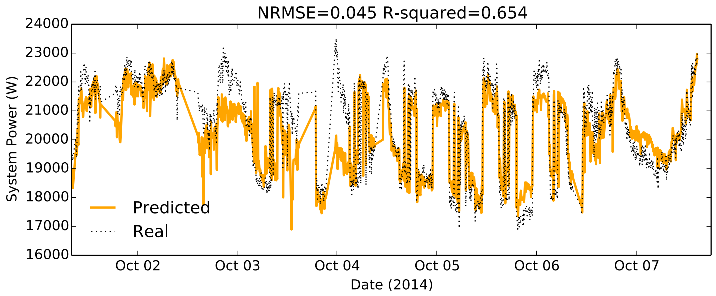
"Best case" benchmark



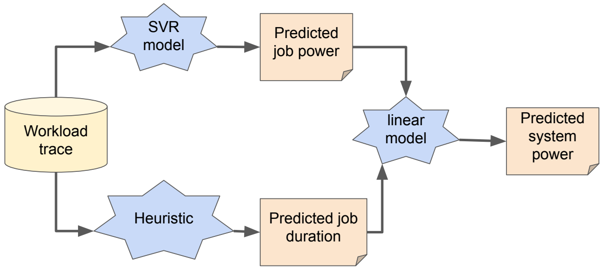
"Worst case" benchmark

Power Consumption Prediction

- Eurora HPC system at CINECA



Predicting System Power



Data-driven Availability

- Among the many software-based mechanisms for increasing availability, ***check-point/restart*** is by far the most widely used in current systems
- Check-pointing consists of taking a ***snapshot*** of the application in execution and saving it on nonvolatile media (usually a parallel file system)
- When a failure occurs, the application is ***restarted*** from the last check-point found on nonvolatile media and the application continues until the next check-point

Check-Point/Restart

- Check-pointing and restarting can be made ***automatic*** and ***transparent*** to applications by initiating them pro-actively through a system software layer
- This removes a big burden from users, but it comes at the cost of increasing ***overhead*** since the state that is saved and restored to/from nonvolatile memory cannot exploit application semantics (to reduce its size) and has to include the entire application state
- How to maintain the convenience of system-initiated check-pointing at a cost comparable to user-initiated check-pointing?
- Too frequent check-pointing with high overhead can slow down applications to a crawl and can also be detrimental for energy efficiency

Check-Point/Restart

- "Optimal" values for check-point intervals can be computed based on statistical averages for inter-failure times and check-pointing costs
- In HPC systems with high failure rates and large check-point/restart times, the mechanism can degenerate into a "pure overhead" scheme performing only check-points/restarts and no useful computation
- Under these conditions, failure masking through ***replication*** becomes a viable alternative for increasing availability
- The challenge is to devise ***dynamic*** and ***adaptive*** algorithms for adjusting the check-point interval and for ***switching between*** check-point/restart and replication as the appropriate availability mechanism

Just-in-Time Check-Pointing

- Ideally, a check-point should be taken only shortly before each failure
- Doing so will minimize the number of check-points taken as well as minimizing the amount of wasted computation
- Such a ***just-in-time check-pointing*** mechanism can be built based on a predictor for node failures

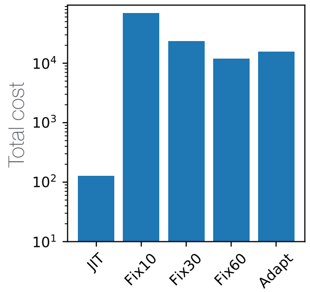
Just-in-Time Check-Pointing

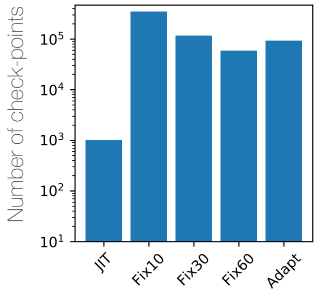
- Apply our node-failure prediction model every 5 minutes to decide which nodes are prone to fail in the following 5-minute time window
- Evaluate the strategy by simulating the workload from the Google trace, and using predictions to decide when to check-point running tasks
- The simulation covers the 10 days used for testing of our predictive model

Just-in-Time Check-Pointing

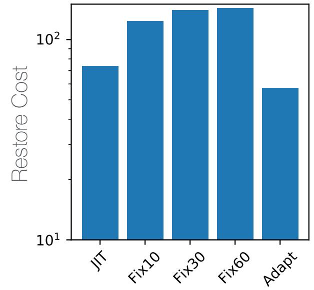
- Cost metrics:
  - Total check-pointing cost (the sum of the check-point sizes for all check-points created),
  - Number of check-points performed,
  - Restore cost (sum of the check-point sizes for all tasks restored after a failure),
  - Wasted CPU time (total CPU time used for all tasks that are interrupted by the failure between the last check-point and the failure)
- Compare six different strategies:
  - Just-in-time (JIT),
  - Fixed interval at 10 minutes (Fix10),
  - Fixed interval at 30 minutes (Fix30),
  - Fixed interval at 60 minutes (Fix60),
  - Adaptive based on Poisson distribution of inter-failure times (Adapt)
  - No check-pointing (NoCP)

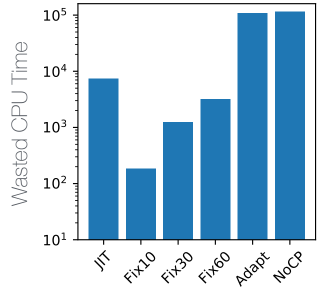
Just-in-Time Check-Pointing





Just-in-Time Check-Pointing





Towards Manageability

- Predictive models built for energy efficiency and resilience can be the basis also for a *management platform*
- The platform projects the system into the future and devises preventive actions based on control mechanisms in case of possible future anomalies
- Detecting anomalous behavior is viewed as a big data classification problem and is tackled using *Deep Learning* techniques for recognizing outliers among cluster patterns arising in streamed data

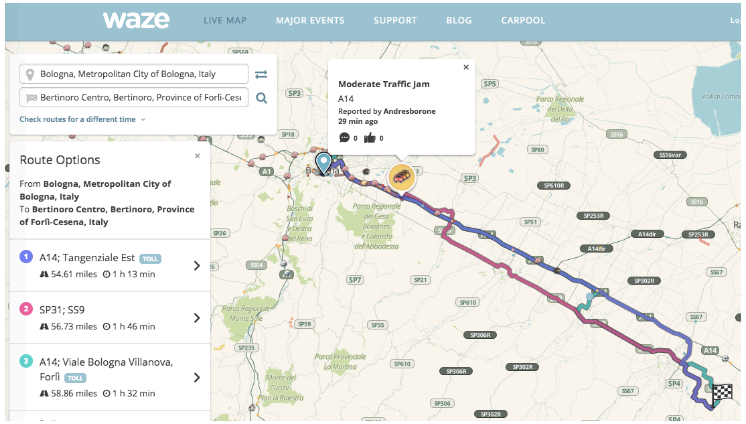
Towards Manageability

- User interface is based on a “route planner” metaphor similar to *Google Maps* or its “crowd-sourced” counterpart, *Waze*
- Administrators are presented a global view of the system indicating the actual loads at various system resources (analogs of vehicular traffic) along with other measures such as power consumption, temperatures, queue lengths and job delays
- The platform signals possible anomalous situations in the current system state and suggests preventive actions such as checkpointing, replication, migration, consolidation

Towards Manageability

- When invoked by an end-user, the service displays different options for executing her job (analogs of alternate routes for traveling from point *A* to point *B* on a map)
- The options are computed based on predictions of the job's demands along with predictions for future system states (including failures)
- For each option, she is given estimates for various metrics such as time-to-completion, cost and energy consumed
- The user may be given the option to select an alternative execution path for her job or alternative system responses to potential anomalies, sorted by their "popularity"

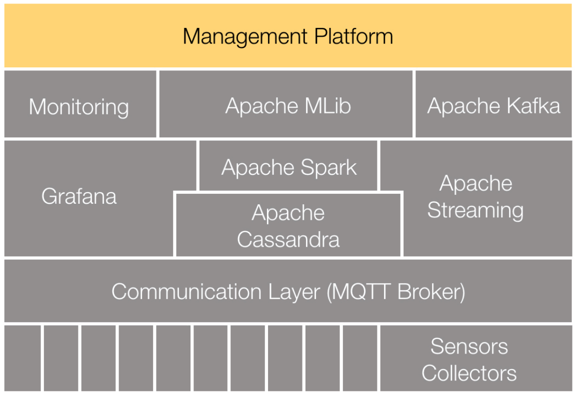
Route Planner Metaphor Waze



Prototype Architecture

- Based on a collection of open-source software technologies
  - *Apache Spark* for cluster computing
  - *Apache Cassandra* for database storage
  - *Apache MLib* for machine learning libraries
  - *Grafana* for data analytics and visualization
  - *Apache Streaming* and *Message Queuing Telemetry Transport* (MQTT) for communicating with sensors
  - *Apache Kafka* Publish/Subscribe service
  - *Google TensorFlow* technology on *Apache Spark ML Pipeline*
  - *Databricks* Deep Learning Pipelines

Streaming Data Monitoring



Management Platform Architecture