IoT – Scale & Performance, planning for the real world

Yogendran Gopalakrishnan Gopithenna – IoT and Big Data Architect, SAP Labs
Sunil Kumar Verma – Senior IoT and Big Data Engineer, SAP Labs

Quick Introduction and takeaways from the tutorial

Yogendran Gopalakrishnan Gopithenna

LinkedIn: https://www.linkedin.com/in/yogendran-gopalakrishnan/

Twitter: @YogiTweets

Organization: SAP Labs

Role: IoT and Big Data Architect and Technical Product Owner

Sunil Kumar Verma

LinkedIn: https://www.linkedin.com/in/vermasunil21/

Twitter: @SunilVe98863214

Organization: SAP Labs

Role: Senior IoT and Big Data Engineer

Audience and Key Takeaways

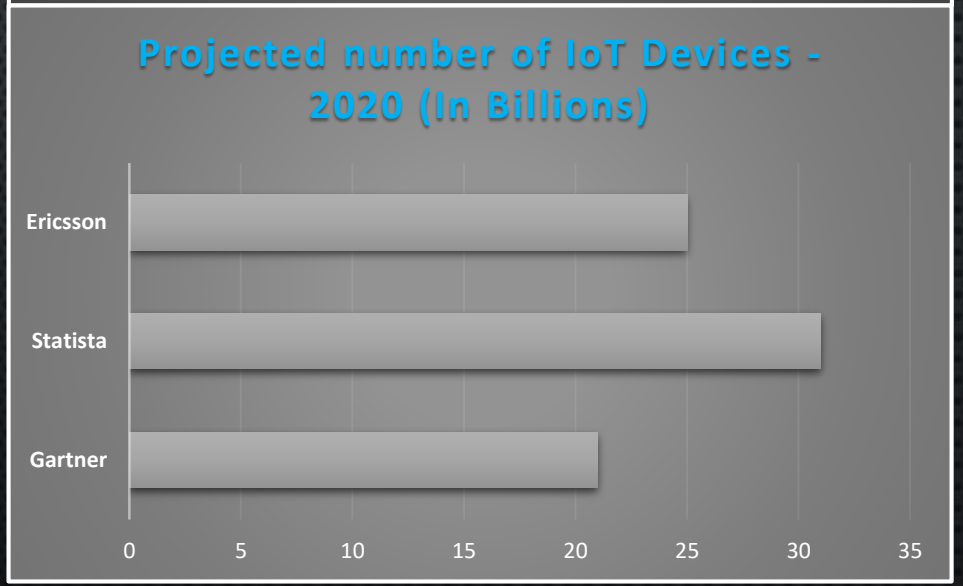
- Audience: Technologists, Architects and executives who will plan/design IoT systems and products
- Questions to ask relevant to IoT and scalability
- Key items to look out for which can be planned early in the development lifecycle
- Items to look out for existing products/systems
- Overall architectural and development practices to enable cost effective scalability

Getting Started

- Successfully rode hype wave and is now in every aspect of our daily life. Disruption at its best with very few analogues!
- Business use cases with exponential growth in number of devices. Volumes kick in at any time! Configuration based scaling is a must.
- Pronounced Performance requirements along with scale is the norm
- Scale is not economy if not done right! All the more so in IoT.
- Unconventional approaches to planning and realization
- Be prepared for volatility and rapidly changing reality on the ground – Plan early, very early!
- Strong foundations - Open to extension but closed to modification
- Best ROI is money making its way into resources – not towards redesign, reconstruction and focus away from business features!

Questions to ask and account for

1) *Why do I have to plan for scale and performance in IoT?*
2) *Does every IoT project necessitate careful planning for scale and performance?*
3) *Do scaling and performance go together?*
4) *Economy of scale?*
5) *How early do I have to plan?*
6) *RoI?*



Building the foundation - I

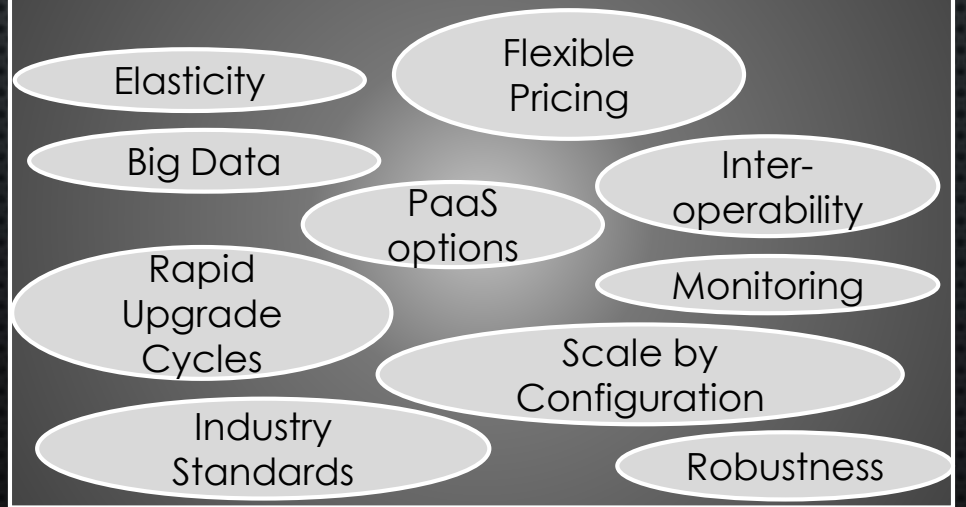
- Industry precedents and analogues – Cloud first strategy!
- Too much cloud jargon – really, now, what does IoT really need in the cloud?
- Cloud brings – predictable pricing models, industry standards and best practices, faster time to market +++ elasticity (rapid scale up and down)!
- Cloud isn't the only ingredient in the foundational alchemy! IoT demands more than just cloud capabilities. Public, Private, Hybrid, On-premise!
- Big Data and IoT are about as conjoined as things get. With growing device volumes comes growing data volumes, be prepared for large data volumes

Tiered Big Data

Questions to ask and account for

1) *What is this "foundation", Where should I start?*
2) *What are the essentials for my foundation?*
3) *Can cloud meet all my needs? Does it always have to be cloud?*
4) *Will any cloud provider do?*
5) *IoT and Big Data? Jargon or necessity?*
6) *How should my big data be tiered?*

Foundational Essentials

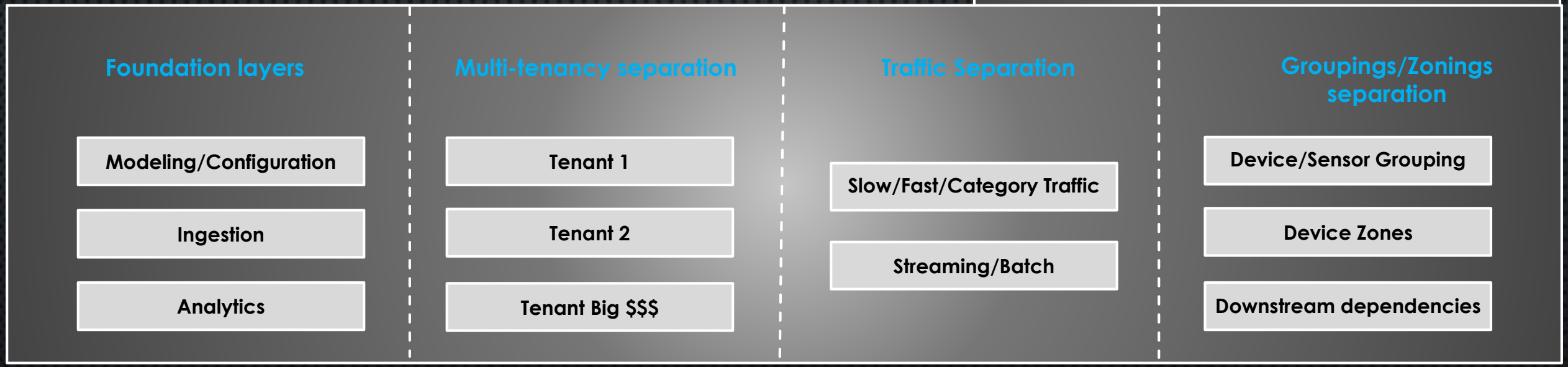


Building the foundation - II

- Foundation should enable 1 customized instance per tenant where possible
- Multi-tenancy + horizontal scaling strategy is a must for scalability
- Some horizontal scaling strategies – Device groupings, nature of traffic (slow Vs fast Vs scheduled vs Ad-hoc), real-time Vs near real-time etc
- Custom tech stack per layer. Microservices! Scale by choice & demand
- Don't forget the APIs and the basics – for example on-boarding

Questions to ask and account for

1) *What should be my multi-tenancy strategy?*
2) *My product/system doesn't necessitate multi-tenancy, what should I do?*
3) *What should be my plan to scale horizontally?*
4) *Does every layer of the IoT stack need its own scaling strategy?*



Leveraging IoT PaaS

- Foundation, Platform – they all matter – Your foundation/platform is another layer on top of the best practices condensed as a platform!
- Security, protocols, connectivity, import/export etc
- Don’t reinvent the wheel! Your domain has been “PaaS” ed but don’t pass on PaaS!
- Vendors vying for your attention with ever growing bells and whistles! Numerous IoT PaaS solutions out there!
- IaaS Muscle backing PaaS – potent combination

Some PaaS features/KPIs to keep an eye on

Multi layer Security	Quick onboarding	Throughput Vs Plan
Easy Customization	Connectivity Extensibility	Interoperability
Custom Retention	Close IaaS support	Feature stability

Questions to ask and account for

1) *What exactly is PaaS and why do I need it?*
2) *Why PaaS when I have my own “foundation”?*
3) *Do I need IoT PaaS offerings?*
4) *What does PaaS bring with it?*
5) *What is IaaS?*
6) *What are the KPIs?*

Some Cloud PaaS Offerings

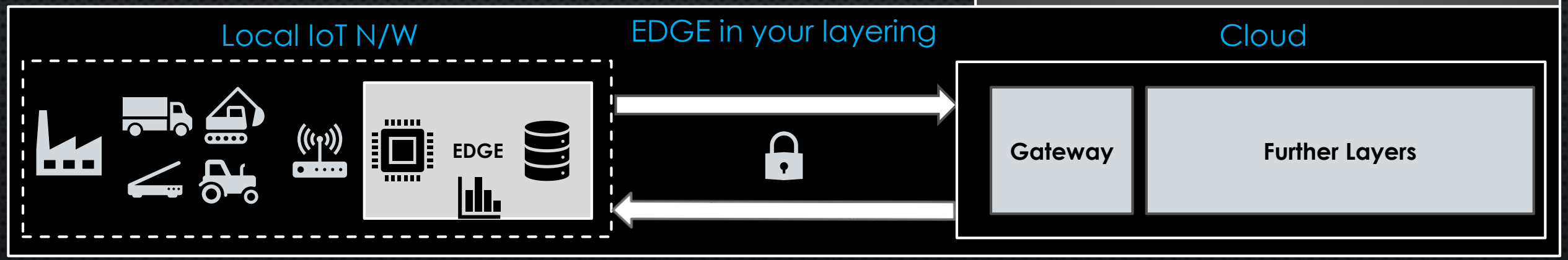
- Azure IoT Hub
- SAP IoT Platform
- AWS IoT Platform
- Google IoT Platform

Get that edge – IoT Loves EDGE

- In simple terms, processing not happening in the cloud typically happen in the last layers of the of the network wherein data is actually generated.
- EDGE when used correctly, not only ensures high performance but also brings great cost savings to boot.
- Local processing of data in the form of collation, filtering and before dispatch
- Do NOT store data for too long on the EDGE
- Do NOT replace cloud processing and analytics with EDGE analogues
- Do NOT use EDGE if you as a pass through. Isolate traffic

Questions to ask and account for

1) *What exactly is EDGE and why do I need it?*
2) *Cloud, PaaS and now the EDGE?*
3) *How do I really use the EDGE for performance?*
4) *What should I avoid doing with EDGE?*
5) *Can EDGE processing replace cloud processing?*



The first and the last miles

- Chain and its weakest links
- EDGE and cloud ensure that the major paths are built towards the proven patterns but do not discount the other aspects which can slow down things
- Some IoT scenarios involve devices talking directly to the cloud bypassing EDGE and some involve full duplex between the cloud and the device
- Robustness when not done right can also hit performance and scalability

Questions to ask and account for

1) *Why do I have to worry about the first and the last miles?*
2) *I already have EDGE, why do I need to worry about the first mile?*
3) *I have the EDGE and the CLOUD, why do I need to worry about the small stuff?*
4) *How do I avoid the slow first and last miles?*

Things which can slow down the first and last miles

- Low throughput security mechanisms and layers
- Interfaces with low baud rates
- Weak microcontrollers/processors
- Low signal/weak GSM/wireless networks

How to avoid slow first and last miles

- Leverage EDGE and PaaS out-of-box capabilities
- Test robustness scenarios
- Test devices with peak scenarios
- Staging and buffered consumption to handle weak links

Plan for peak scenarios

- As pointed out earlier, volume growths in IoT can be extremely volatile and unpredictable
- Peak scenarios involve both scheduled and unscheduled ones
- Auto scaling is your friend, close PaaS and IaaS integration is a life saver
- Always run performance tests for known scheduled peak scenarios and involve each layer including devices
- Add randomly salted peak scenarios in your performance tests

Questions to ask and account for

1) *Why do peak scenarios have higher significance in the context of IoT?*
2) *Is it even possible to plan peak scenarios given your notes on IOT volumes growth?*
3) *How do I handle unexpected peaks and sudden exponential growth in volumes?*
4) *Do I have to spend a fortune for the ability to handle peak scenarios?*

Things to keep an eye on for peak scenarios and auto scaling

- Ensure native auto scaling from the stack and the PaaS as against implementing your own
- Individual components have algorithms which best handle the peak scenarios
- Ensure a pay-as-you-use plan to make sure you pay only for what was scaled up
- Implement a feedback cycle of the unscheduled peak scenarios onto your performance tests

Architectures and Dev Practices

- Architectures of the foundation which do not follow open architecture approaches and are tied to particular stacks bring upgrade and migration costs which in IoT scenarios can be significant
- Architectures which do not scale by configuration and need redesign/releases contribute significantly to development costs
- Constantly evolving technology landscapes – for e.g. 5g is knocking on the door

Questions to ask and account for

1) *How will bad architecture and development practices impact scalability and costs?*
2) *What are the development practices to include by the engineering teams?*
3) *Why should I plan to keep my architecture open?*

Recommended Architectural Practices

- Open Architectural approach
- Scaling by configuration – not by code changes or redesign
- Well defined performance testing plans and feedback cycle
- Compliant with PaaS guidelines and best practices
- NFRs for every layer

Recommended Development Practices

- Performance tests with every release/deployment
- Integration tests involving every layer with real life like mocking involving latencies, throughput and defined NFRs
- 12 Factor Application Development
- Backing services from PaaS