Power On The Roof

Microservices Architecture Proposal

Executive Summary-

Power on the Roof is company that provides safe and reliable alternative Energy Platforms, that too for the half of the Cost which energy companies charge per Kilowatt hour. It provides and manages many solar energy products including Solar cells, Solar roof tiles, Batteries, Power inverters and many safety devices including interconnectors and isolators.

PoTR is a global company having market presence in Australia, United States and Spain, with plans to expand worldwide.

PoTR is not only eying to expand geographically, It is wanting to expand into new products, product variations, product extensions and what not. But they are restricted due to their monolithic IT architecture which is not allowing them to scale in all terms and fulfill the present IT requirements.

Existing System Analysis

On closely analyzing the Existing system used by PoTR I have found some points to be considered for improvement:

1-Customer Support and Interaction

As customer contact is using very old IVR system for customer Interaction, they require more human power in support team to address all the inbound and outbound customers and to resolve verifications for MOSIEs identity. This can be a big challenge keeping scaling in mind. And Customer Experience is also very bad in this case as todays nobody prefers to get information on calls and SMS.

2-Centralised System-

As PoTR works on Solar Energy products the configurations for devices may vary with the geographical locations, the centralized system is not solving the problem for the same. With wider reach (around the word) and customer satisfaction they need distributed Servers and Application architecture.

3-Lack of Automation:

Presently most of the data base transections and information transfer is done manually whether it is COST module or Authentication module or Credit module which always adds the Vulnerability of Human error and on top of that the most critical dependency will be manpower requirement which is not good in various aspects including cost effectiveness and scalability. For example, PoTR wants to start their business in Some trivial country first and crucial requirement will be trained man power which can be tedious at times.

4-Complexity In application Maintenace and Bug Fixes:

It is very difficult with single server instance to manage the production code and Quality assurance of the product at the same time, It can cause either the larger downtime which can cause the direct revenue loss or the buggy code which is eventually a loss too.

5-Mobile or Table Application for MOSIE's

Improvements in authentication, registration and information gathering can be possible in MOSIE's application, as this app is performing various functions and it's a good point of contact from Control System, it can be leveraged more as Application. It can have various features like, incident creation, Task Management module, Alert system for instant action required on the Installed devices and many more.

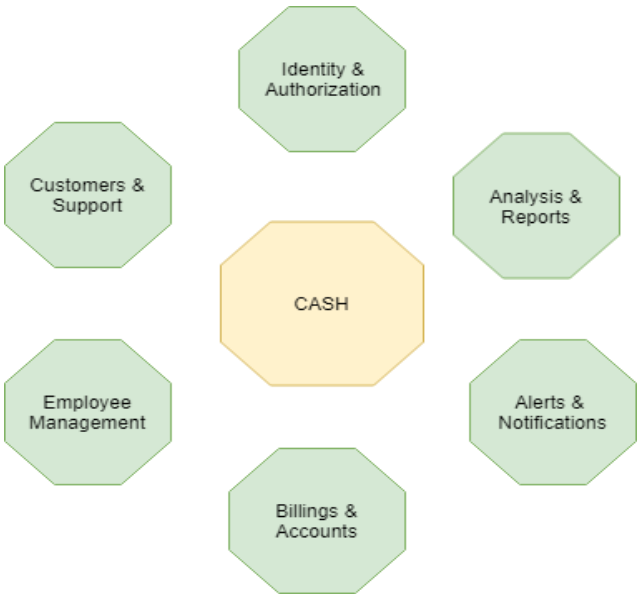
Microservices As A Solution

Looking at the current system architecture, there are few big modules Like CASH and CREDIT which are responsible for delivering almost complete business logic, It can be good If there is no scope of **further scaling**, Database is **centralized,** there is minimal **requirement change** and no **technology change** is expected.

But looking at the PoTR scenario and business goals current monolithic architecture is not suffice.

CASH

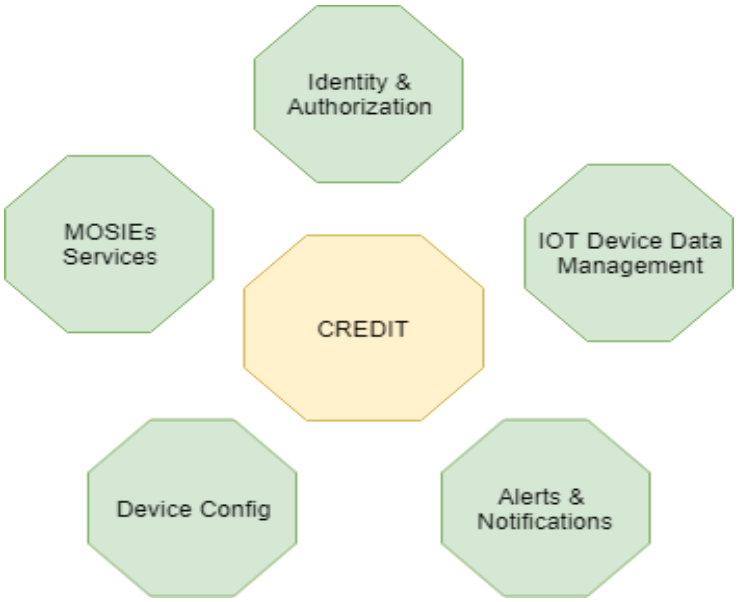
Consider the CASH module It can be modularized in many micro services according to their respective functionalities like:



It is found that sometime CASH module faces the downtime for 30 mins which is letting the direct revenue loss and security loss. If we use disintegrated and independent microservices the **Fault isolation is easy**. Even if one service goes down, other can continue to function. And with distributed Data base according to the geographical location, performance of the overall system can be increased. As It will allow each of the microservice to adopt a data model best suited for its needs. And change in data model for one microservice will not affect the other ones.In this Architecture all the modules will be independent from each other and can interact with well-defined Interfaces and messaging tools. Which makes easier deployments.

CREDIT

Now let's Consider the CREDIT module, It also is performing a lot of operations and the most tedious part is fetching and maintaining data coming from IOT devices consistently and also the MOSIEs Applications, this module can be divided in many microservices like.



CREDIT is the system in which Scaling is a big challenge, using microservices architecture, more hardware resources could be allocated to the service that is frequently used.

In the PoTR case, a greater number of IOT devices will send the data to the system as compared to the traffic on MOSIE's Applications. So, more resources could be allocated to the "IOT Device Management Microservice" as compared to other microservices, which can be great raise in system performance.

Hence using microservices architecture this whole system can be changed to a better performing and scalable system, but It is bit more complex to set up microservice architecture Technology stack will we larger and so it the cost.

But as it can automate most of the processes and Deployments will be easier to maintain and less time taking the human and developer resource requirement will be lower which can enable the PoTR to take this architecture change.

Recommendations

I will recommend the microservices architecture as a solution to all the problems PoTR is facing as a worldwide solution that I found in the analysis. Let's take this point wise:

1-Customer Support and Interaction

We can create a module which can replace the traditional IVR system for customer interaction, which will be a kind on unified portal for Inbound as well as the outbound customers and MOSIEs. This will contain all the details including billing details, device details, customer balance details, MOSIE's job descriptions and registrations.

It will be a role based portal in which user can log in with role based credentials provided to him and he'll get all the information relevant for him.

2- Distributed System

Instead of using centralized system, there will be distributed servers which will be configured according to the geographical location and requirements.

3-Automation

There are various process automation tools can be used to reduce manual efforts such as replacing the "calling and getting the billing information" with unified and automated portal.

Automated incident and notification alert features can be given for better service of the devices. SAP workflows can be used for managing the different processes.

4- Agile Development methodology

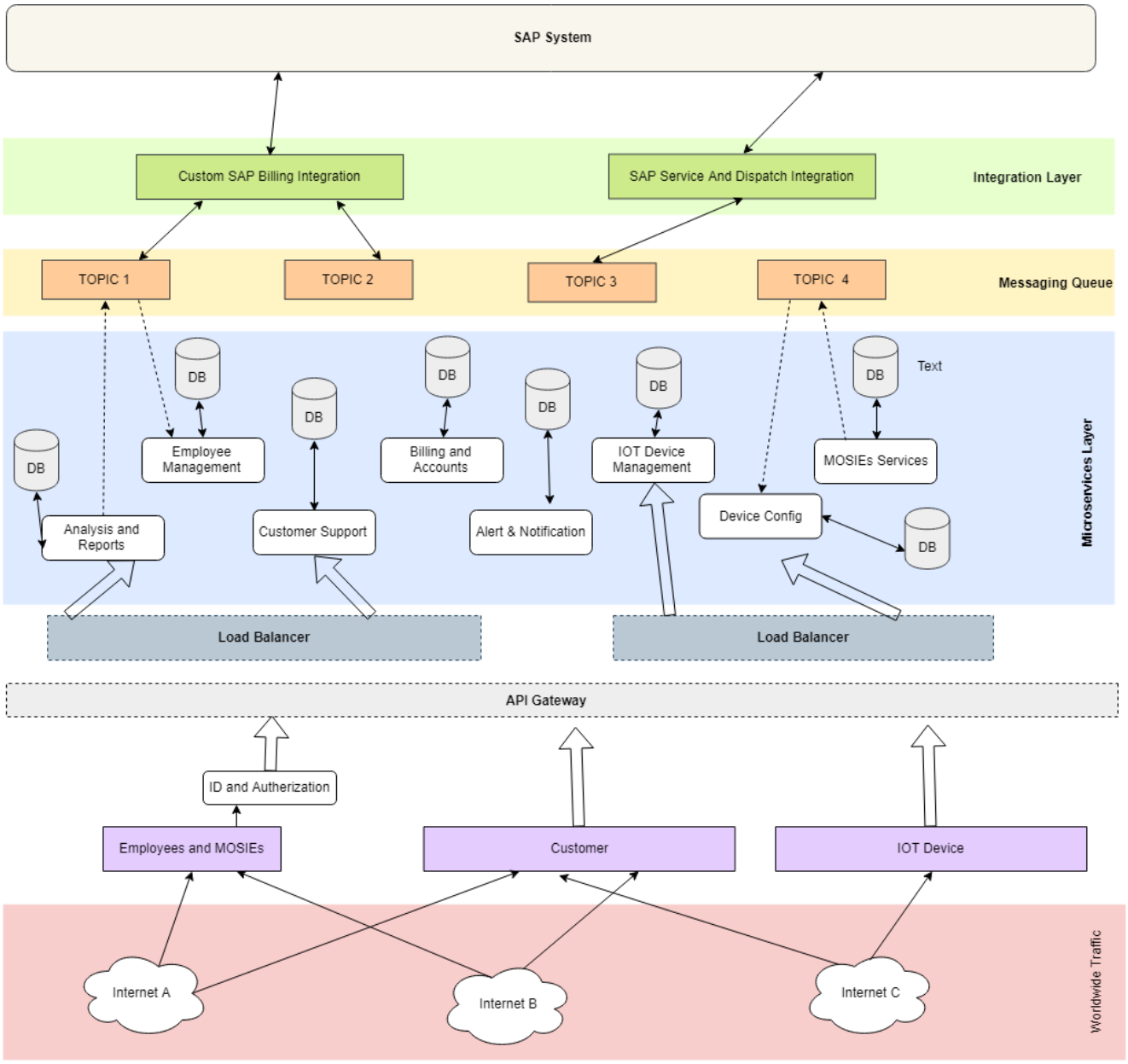
Use of agile methodology should be used for better application maintenance and replica of production should be maintained for proper testing and Quality assurance. Each sprint should be planed and deployments should be done according to it.

5-New Features

Some new features like unified Portal, Incident and notification for MOSIEs, customer support and complain portal and Administration portal for various analysis can implemented as the microservices using suitable technology and data model for each of the microservice.

MicroService Architecture Design

Here is the Proposal Design that can fulfill the PoTR business Requirements and Future Scalability Goals.



Let's understand this design from bottom to top.

1- Traffic from all over the world comes in the form of Rest Calls from **Android devices or web UI's.**

2- This traffic can be the from external **customers** who wants to see the plan, billing and support related stuffs or it can be from **MOSIE's or other employees** who wants their job details or analysis related Information, or it can the **IOT devices** which are sending the information in real time in every 5 mins.

3- After Authentication of employees which can be done via module, these requests will go to **API Gateway** which will filter the request to the various modules, reduce the request roundtrips and filter the request to various service instances on the basis of location.

4-Now these requests are managed by **load balancer** and more resources are provided to the module with larger traffic, for example "IOT Device Management" module will get request from very large number of IOT devices hence traffic will be more as compared to other modules like "customer support" or "device Config module" so this load balancer can assign more resource to that module. (This will be a great application of using microservices over the traditional monolithic architecture)

5-Now Request will come to Module Layer in which each of the module will have its own Database, each can use different data Model or language according to their requirement. So here is the short description of all the microservices:

- **ID and Authentication** –User Authentication using PIC's for Employees and MOSIE's.
- **Analysis and report** – It will keep the analysis at the admin level for the sales and profit and generate report for the administrators and employees according to their roles.
- **Employee management**- It will manage the employees Jobs description, performance and Billing.
- **Customer Support**- It will manage support system for customer, like customer can raise change request, complains, get Information about their plan and Billing information.
- **Billing And Accounts**- It will manage the customer as well as Employees Accounts and Billing information for inventory management.
- **IOT Device Management**- This will be responsible for Fetching and maintaining the data coming from the IOT devices according to the location.
- **Device Configuration**- This will be a master module which will handle he configuration for all the devices according to their type and deployment location.

- **MOSIEs Services**- This will be responsible for Showing information for MOSIE's on android devices given to them and their configuration.
- **Alert and Notification**- this can be additional functionality which can be used in multiple applications like, showing new offers and plans to customer, showing notification to MOSIEs if any device needs to be replaced.

6-All these modules will talk to each other and maintain the data consistency using middleware tools and messaging queues like **RABITMQ**.

7- Now from these Queues Data will be transferred to Integration Layer which has **Custom SAP billing Integration** and **SAP Service and Dispatch Information** and data will be transferred to **SAP System.**