Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 08, August-2020

Introduction to Cloud Design Patterns

Nachiket Vaidya
Bharti Vidyapith College of Engineering

Abstract:- **Software Design Patterns have long been popular and viewed as a reusable solution to a commonly occurring problem in software design.A design pattern describes a collection of general techniques, actions, and/or tasks for developing object-oriented software.**
This paper will reinforce and reiterate some of these traditional concepts and discuss how they may evolve in the context of cloud computing. It will also discuss some unprecedented concepts that have emerged due to the dynamic nature of the cloud.
This paper is targeted towards cloud architects who are gearing up to move an enterprise-class application from a fixed physical environment to a virtualized cloud environment. The focus of this paper is to introduce the different patterns that need to be considered in creating new cloud applications or migrating existing applications to the cloud.

Keywords: Cloud Computing, Software Design Patterns,Cloud Design Patterns, Enterprise Cloud Development

1. INTRODUCTION

In the modern era, software is delivered as a service(SAAS).Modern software applications including cloud are built on a twelve-factor methodology that

- [1]Use **declarative** formats for setup automation, to minimize time and cost for new developers joining the project;

2. CHALLENGES **WITH CLOUD APPLICATIONS**

Availability

Availability is the proportion of time that the system is functional and working, usually measured as a percentage of uptime. [2]It can be affected by system errors, infrastructure problems, malicious attacks, and system load. Cloud applications typically provide users with a service level agreement (SLA), so applications must be designed to maximize availability.

Design and Implementation

Good design encompasses factors such as consistency and coherence in component design and deployment, maintainability to simplify administration and development, and reusability to allow components and subsystems to be used in other applications and in other scenarios.Decisions made during the design and implementation phase have a huge impact on the quality and the total cost of ownership of cloud hosted applications and services.

Performance & Scalability

Performance is an indication of the responsiveness of a system to execute any action within a given time interval, while scalability is the ability of a system either to handle increases in load without impact on performance or for the available resources to be readily increased.Cloud applications typically encounter variable workloads and peaks in activity. Instead, applications should be able to scale out within limits to meet peaks in demand, and scale in when demand decreases. Scalability concerns not just compute

designed and deployed in a way that protects them from malicious attacks, restricts access to only approved users, and protects sensitive data often open to the public, and may serve untrusted users. Applications must be

Messaging

The distributed nature of cloud applications requires a messaging infrastructure that connects the components and services, ideally in a loosely coupled manner in order to maximise scalability. Asynchronous messaging is widely used, and provides many benefits, but also brings challenges such as the ordering of messages, poison message management, idempotency, and more.

Resiliency

Resiliency is the ability of a system to gracefully handle and recover from failures. The nature of cloud hosting, where applications are often multi-tenant, use shared services, compete for resources and bandwidth, communicate over the Internet, and run on commodity hardware means there is an increased likelihood that both transient and more permanent faults will arise. Detecting failures, and recovering quickly and efficiently, is necessary to maintain resiliency.

3. DESIGN PATTERNS FOR CLOUD

Cloud design patterns are useful for building reliable, scalable, secure applications in the cloud and thus mitigate the challenges associated with cloud application development. There are many design patterns available for Azure AWS Google Cloud and other cloud environments depending on the problem you are trying to resolve , an appropriate choice can be made. Below are some the popular ones

- Minimize cost and maximize scalability and performance

Cache Aside Pattern

- Load data on demand into a cache from datastore
- Helps improve performance
- Helps in maintain consistency between data held in the cache and data in the underlying data store

When to use this pattern

- Resource demand is unpredictable.
- This pattern enables applications to load data on demand
- It makes no assumptions about which data an application will require in advance

When not to use this pattern

- Don’t use it for data that changes very often

Federated Identity Pattern

- Delegate authentication to an external identity provider.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 08, August-2020

- Simplify development, minimize the requirement for user administration
- Improve the user experience of the application
- Centralized providing MFA for user authentication
- cache and data in the underlying data store

When to use this pattern

- When you have multiple applications and want to provide SSO for applications

When to use this pattern

- Sensitive information (Health care, Authentication)
- Distributed System where perform request validation separately

When not to use this pattern

- Performance vs Security

Circuit Breaker Pattern

- To handle faults that might take a variable amount of time to recover
- When connecting to a remote service or resource

When to use this pattern

- To prevent an application from trying to invoke a remote service or access a shared resource if this operation is highly likely to fail
- Better user experience

When not to use this pattern

- Handling access to local private resources in an application, such as in-memory data structure
- Creates an overhead
-

4. CONCLUSION

Design patterns offer guidance and provide best practice solutions to the common problems faced during designing cloud-hosted applications. As you have seen,

- Have a **clean contract** with the underlying operating system, offering **maximum portability** between execution environments;
- Are suitable for **deployment** on modern **cloud platforms**, obviating the need for servers and systems administration;
- **Minimize divergence** between development and production, enabling **continuous deployment** for maximum agility;
- And can **scale up** without significant changes to tooling, architecture, or development practices.

don't deliver the values enterprises expect often resulting in cost overflows and lower KPIS for the organization.

The cloud reinforces some old concepts of building highly scalable Internet architectures and introduces some new concepts that entirely change the way applications are built and deployed. As a cloud architect, it is important to understand the challenges associated with Cloud before designing the solutions to reap on the benefits that cloud computing brings to the table.Cloud applications that are not designed with Cloud design Patterns(CDP) can lead to poorly designed applications.

Management and Monitoring

Cloud applications run in a remote datacenter where you do not have full control of the infrastructure or, in some cases, the operating system.[3] This can make management and monitoring more difficult than an on-premises deployment.Applications must expose runtime information that administrators and operators can use to manage and monitor the system, as well as supporting changing business requirements and customisation without requiring the application to be stopped or redeployed.

Data Management

Data management is the key element of cloud applications, and influences most of the quality attributes. Data is typically hosted in different locations and across multiple servers for reasons such as performance, scalability or availability, and this can present a range of challenges. For example, data consistency must be maintained, and data will typically need to be synchronised across different locations.

Security

Security is the capability of a system to prevent malicious or accidental actions outside of the designed usage, and to prevent disclosure or loss of information. Cloud applications are exposed on the Internet outside trusted on-premises boundaries, are

[4]External Configuration Store pattern

Move configuration information out of the application deployment package to a centralized location. This can provide opportunities for easier management and control of configuration data, and for sharing configuration data across applications and application instances.

When to use this pattern

- When you have shared configuration, multiple application
- You want to manage configuration centrally by DevOps
- Provide audit for each configuration

When not to use this pattern

- When you only have a single application there is no need to use this pattern it will make things more complex

When not to use this pattern

- You already have a single application and have custom code that allows you to login

Value Key Pattern

- Use a token that provides clients with restricted direct access to a specific resource
- Provide offload data transfer from the application
- Federated identity in SAAS application
- Federated identity with multiple partners

When to use this pattern

- The application has limited resources
- To minimize operational cost
- Many interaction with external resources (upload, download)
- When the data is stored in a remote data store or a different datacenter

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 08, August-2020

When not to use this pattern

- When you need to transform the data before upload or download
- Not a substitute for handling exceptions in the business logic of your applications

GateKeeper Pattern

- Using a dedicated host instance that acts as a broker between clients and services
- Protect applications and services
- Validates and sanitizes requests, and passes requests and data between them
- Provide an additional layer of security, and limit the attack surface of the system

Azure Cloud Design Patterns are rich in useful content, therefore, a foundational knowledge of the design patterns will help you to design and build modern cloud-based applications seamlessly and with precision.
Most of the patterns include code samples or snippets that show how to implement the pattern on Azure. However, most of the patterns are relevant to any distributed system, whether hosted on Azure or on other cloud platforms.

5. REFERENCES

[1] Twelve Factor App (https://12factor.net/)
[2] Azure Architecture Center | Microsoft Docs https://docs.microsoft.com/en-us/azure/architecture/patterns/category/design-implementation
[3] Management and Monitoring patterns - Cloud... | Microsoft Docs https://docs.microsoft.com/en-us/azure/architecture/patterns/category/management-monitoring
[4] Cloud Design Patterns | Microsoft Docshttps://docs.microsoft.com/it-it/azure/architecture/patterns/