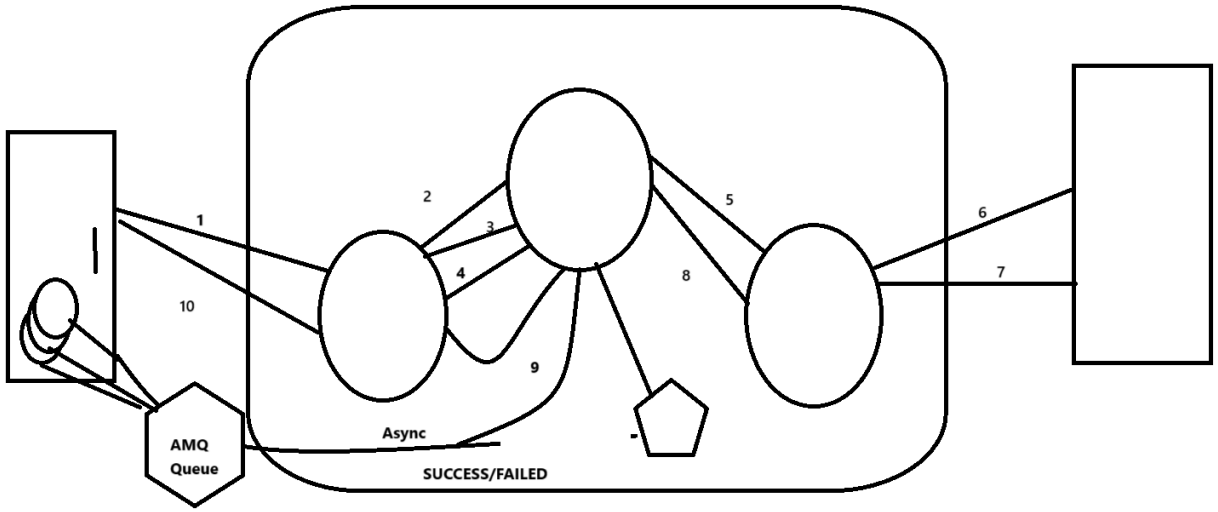
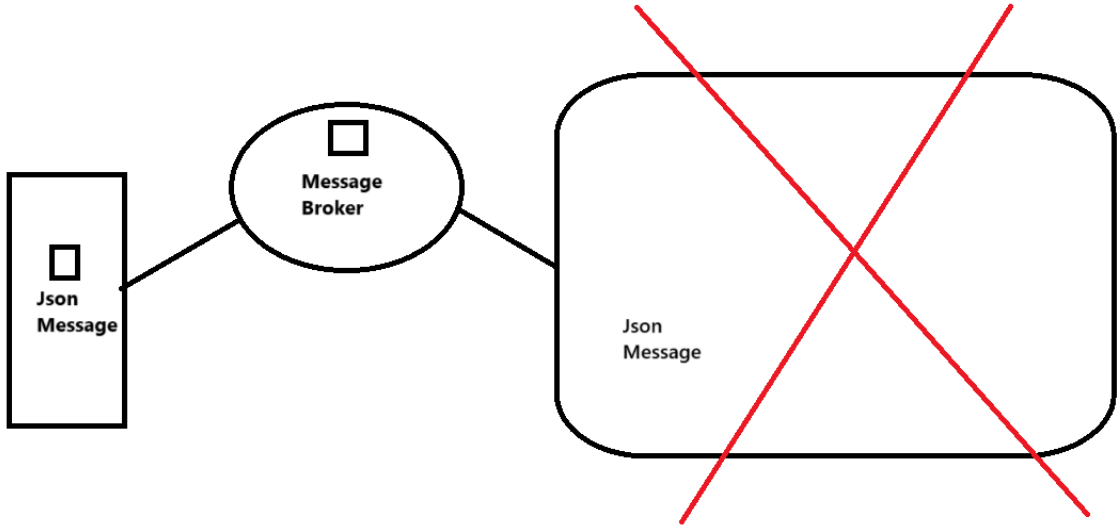
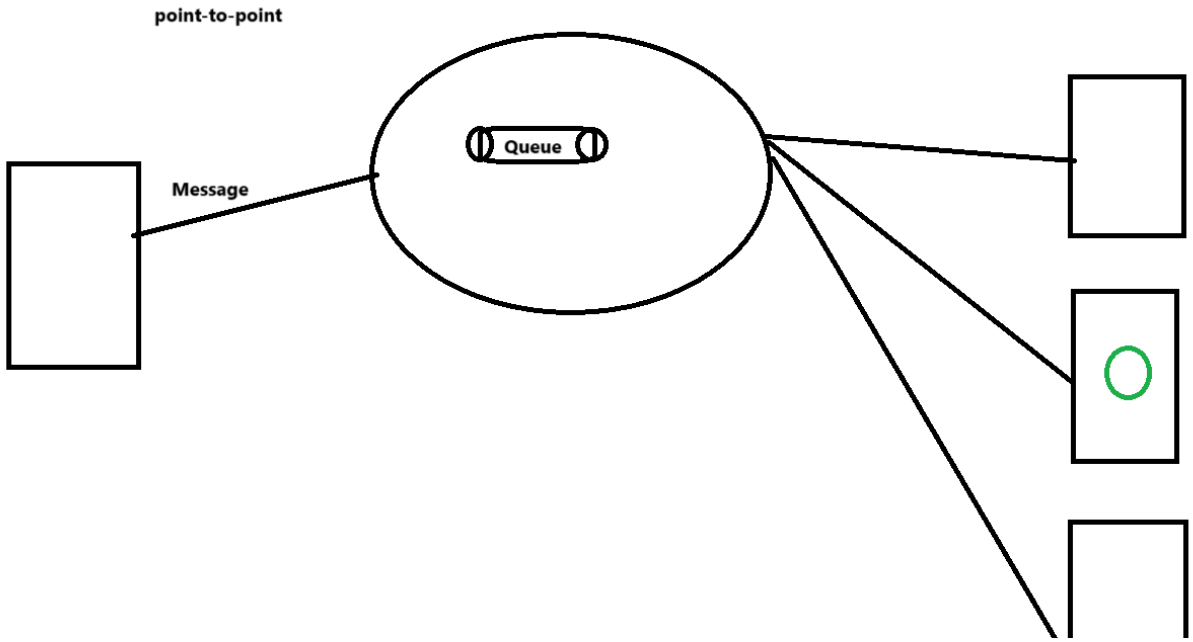
DIAGRAM

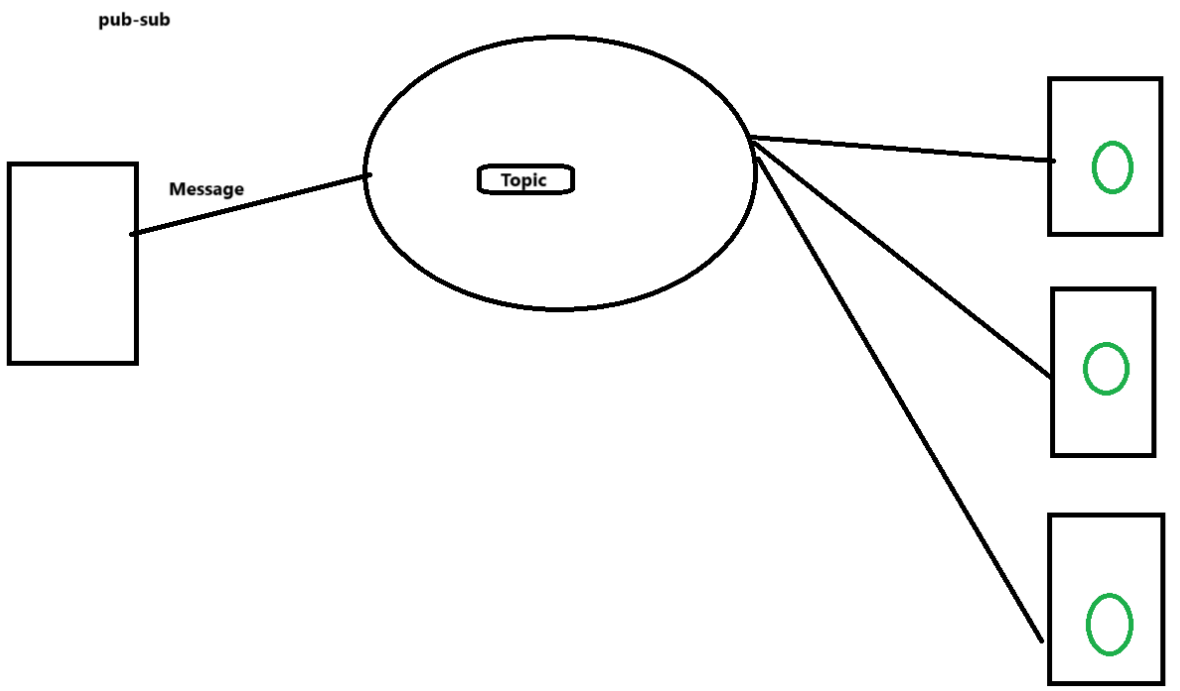
Sync & Asyn communication

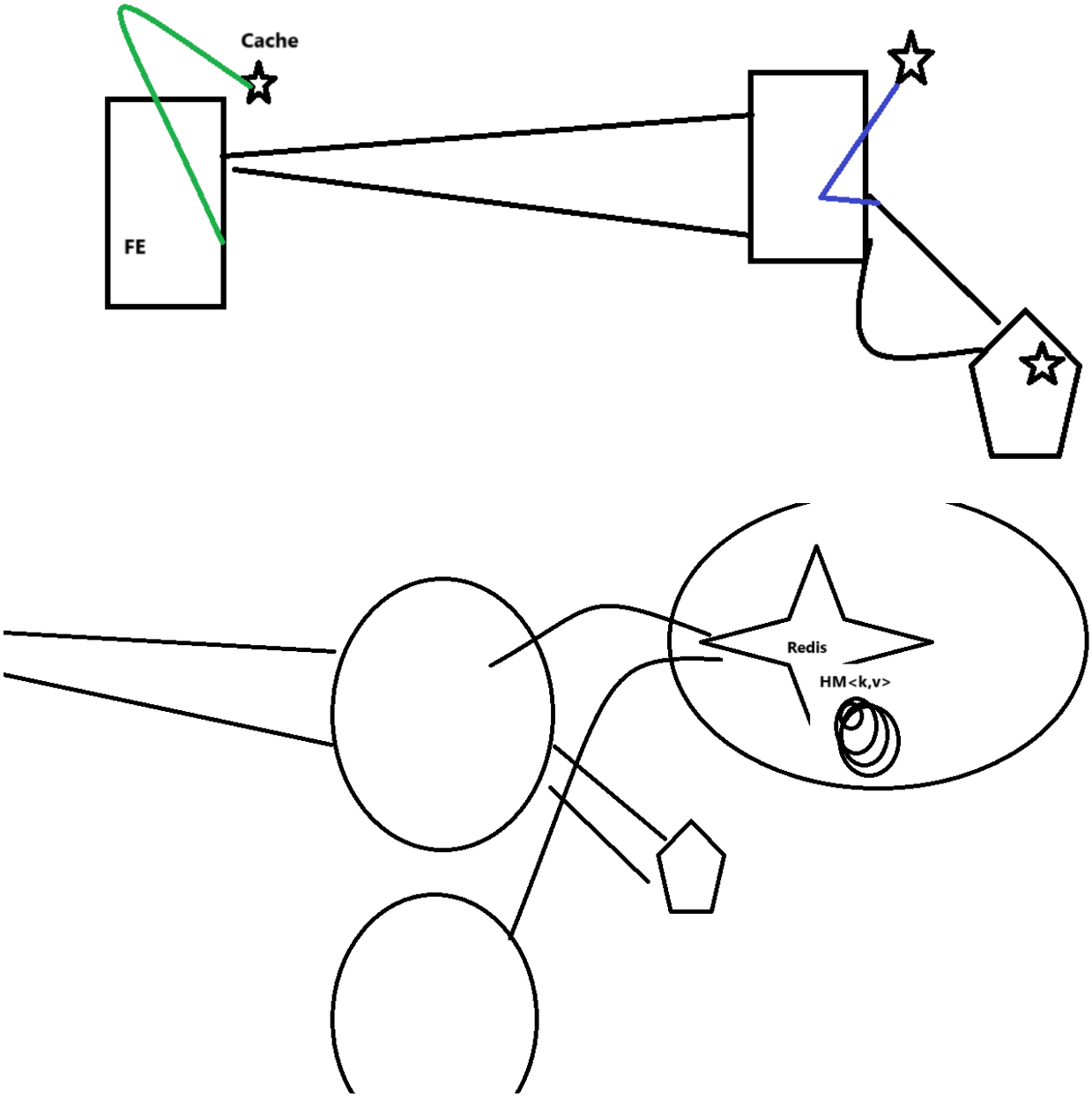


Async working while destination is down




point-to-point
Queue
Message


pub-sub
Topic
Message



LIVE NOTES

2 open source software.

- ActiveMQ - Async Communication
- Redis - Caching

Communication:
Sync - invoker waits for the response before getting started with next steps.
- If you want realtime responses. Immediate actions
- Making API call

Asyn - You invoke, & start executing next piece of logic
- This is applicable for systems, where it is ok to have some delay in execution.
- JMS

JDBC tech specification
    MySQL
    Oracle
    ..
    ..

JMS tech specification (For async)
    ActiveMQ

JMS
Java Message Service (JMS) is a Java API that allows applications to communicate asynchronously using messages. It provides a way for distributed systems to exchange messages reliably and loosely coupled. JMS is part of Java EE (Jakarta EE) and supports different types of messaging models:

Point-to-Point (P2P) – Uses a queue, where a sender sends a message to a queue, and only one consumer receives it.

Publish-Subscribe (Pub-Sub) – Uses a topic, where a sender (publisher) sends a message to a topic, and multiple subscribers receive it.

ActiveMQ:
ActiveMQ is an open-source, Java-based message broker that implements the JMS API. It acts as a middleware that enables communication between different applications by passing messages.

How JMS and ActiveMQ Work Together?
JMS is just an API specification, meaning it defines how messaging should work but doesn't provide an implementation.
ActiveMQ is an implementation of the JMS API, meaning you can use it as a message broker to send and receive messages using JMS.

Async:
- The destination system, need not immediate be available to process the message. It can do it later, when it gets available.
- Even if destination system, is down, still no issues. messages will be maintained in broker. And will be processed when destination system comes up.
- High performance system.

- When final status is updated, SUCCESS/FAILED, then inform client APP about the status in async manner.

JMS defines 2 models of communication
- Point to point
- pub-sub

Queue based communication
point-to-point.

processing-service will send the final payment status to AMQ.
To ActiveMQ "QUEUE"

in client app, multiple instances of receiving service will be listening to the above QUEUE for our status message.
But because of Queue based p-to-p communication, only 1 receiver will receive the message & process it further.

==========
We are building Async communication using ActiveMQ in our Spring Boot application. give us step-by-step actinos to do.

We want to send below message to "Queue" in AMQ for point-to-point communication.

```
public class StatusMessage {
	private String txnReference;
	private String txnStatus;

}
==============
```

Caching - Redis

Instead of again & again calling external system for certain data, you can simply keep in cache, & get it whenever you need it again.

Caches speed up the entire system.

what data to save in cache.

Data which doesn't change frequently, should be kept in cache.

if data gets modified in DB, your cache will have old data itself, and you will return old data to the invoker.

HashMap

List

How to create cache.

1. We can simply create cache as a HashMap in your memory of spring boot app
   - With time this will grow & spring boot app will have less memory to operate with.
   - When multiple instances of same service runs, then we have to maintain cache in all the service.
   - whenever restart, cache gets cleared.. & you need to re-populate it.

2. Store your cache data in external service like Redis
   - Spring boot app can run independently
   - all cache is maintained in redis, so service don't have to manage the updates of cache
   - even after restart, data is available.
   - PROD run redis as cluster, so no single point of failure
   - Redis primarily designed for fast access.

Redis has its own DS
- value
- List of value
- Map<key, value>

Install Redis & start redis

tool to connect to redis so you can run redis commads to work with the datastructure which redis offer.

redis-cli

learn redis commands

POC application - spring boot application, which works with redis
- saving data in cache & retriving data in cache
- RedisTemplate

In our project, what use-case we can speedup.

1. In initiatePayment call,1st step is to retrieve txnobject from DB & perform actions.
2. From create Payment to initiate payment, then txn object will not change.. so we can keep the data in redis for faster access.
3. When txn is saved in DB, in PSI, write code to save this TxnDTO in Redis

```
public void saveString(String key, String value, long timeout) {
      redisTemplate.opsForValue().set(key, value, timeout, TimeUnit.MINUTES);
   }
```

key: txnReference
value: JsonString TxnDTO
5 mins

4. Retrive, in initiatePayment, pass the txnRefernce to redis to read the value. if value doesn't exist make DB call to retrive the information & use it.

- data changed in db, doesn't apply
- data will not pileup, 5mins expiry

==========
REDIS prompt

I am new to redis.

I head its being used for caching solutions.

I have data as plain value, list, hash, i want to store & retrive in redis.

how to work with redis-cli to perform these operation.

give me step by step approach, i am using windows. start with setting up redis in my machine.

---
yes, give me below
Do you want to integrate Redis into your Spring Boot application for caching?
show with RedisTemplate library
