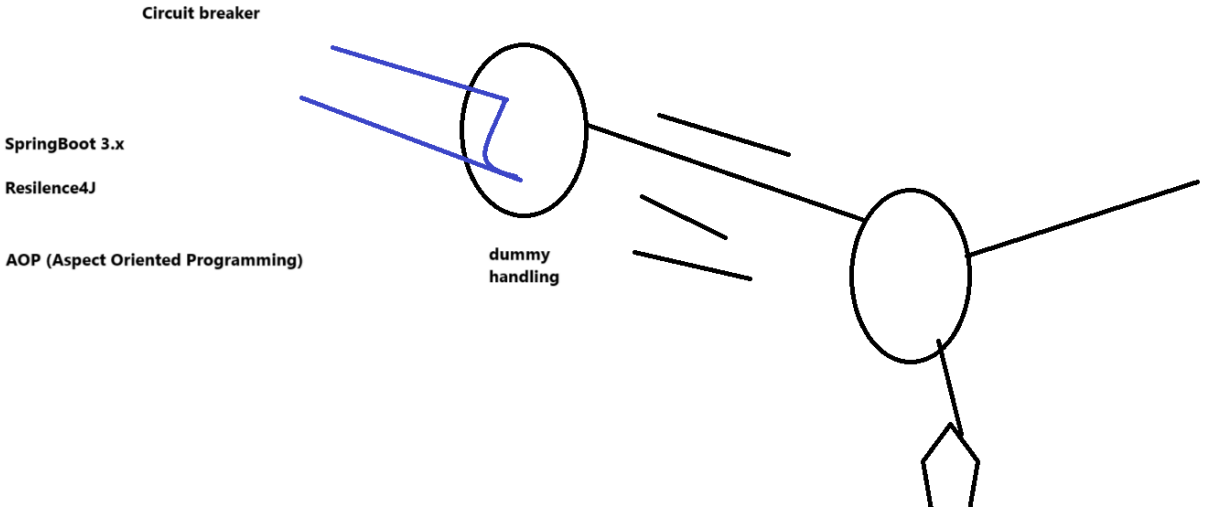
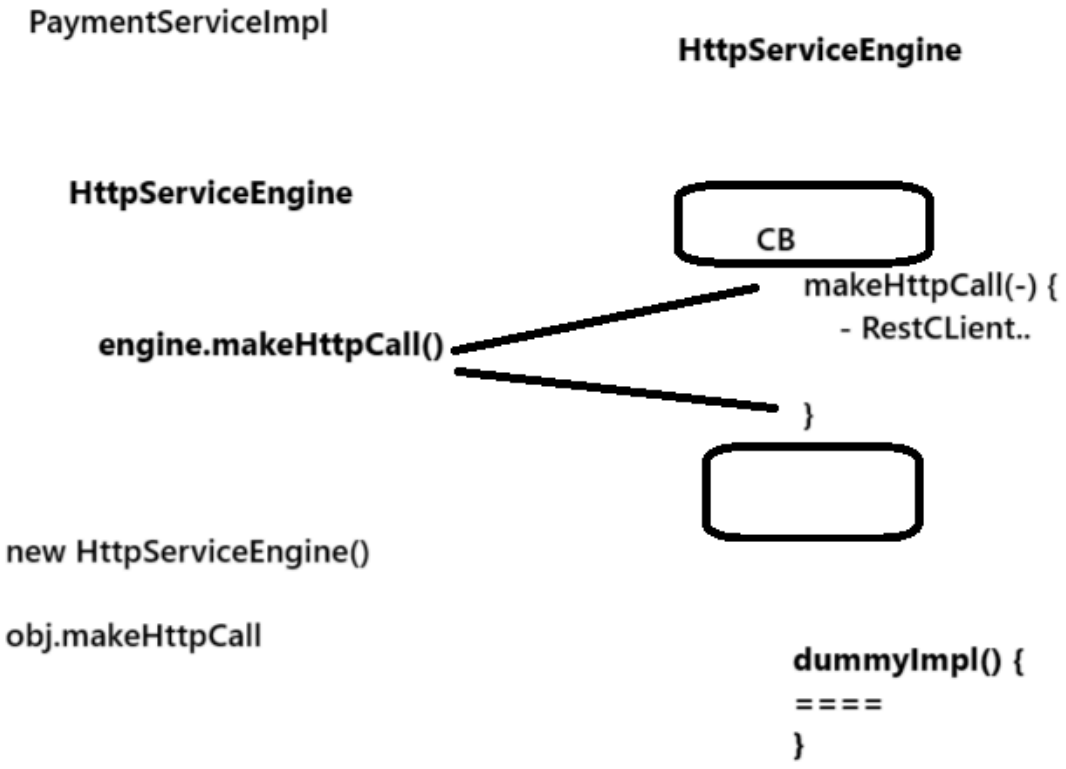
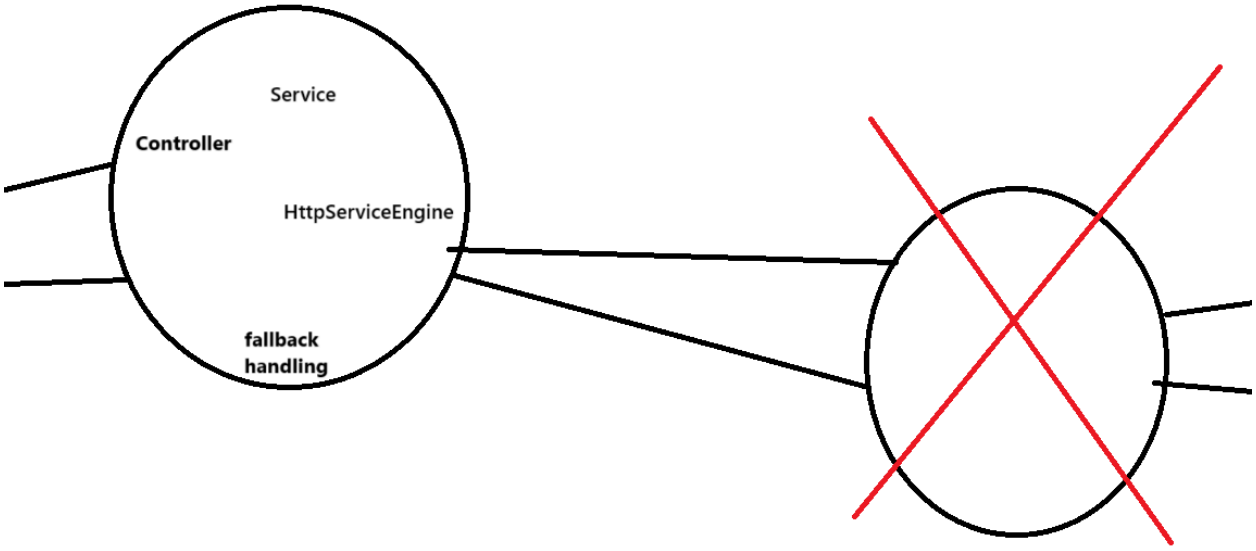
DIAGRAM



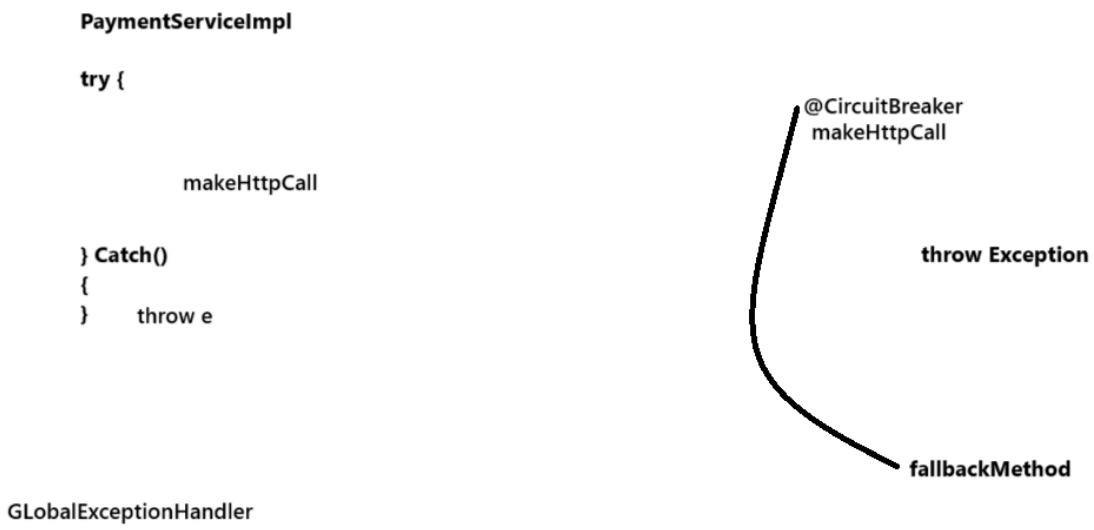
AOP mechanism



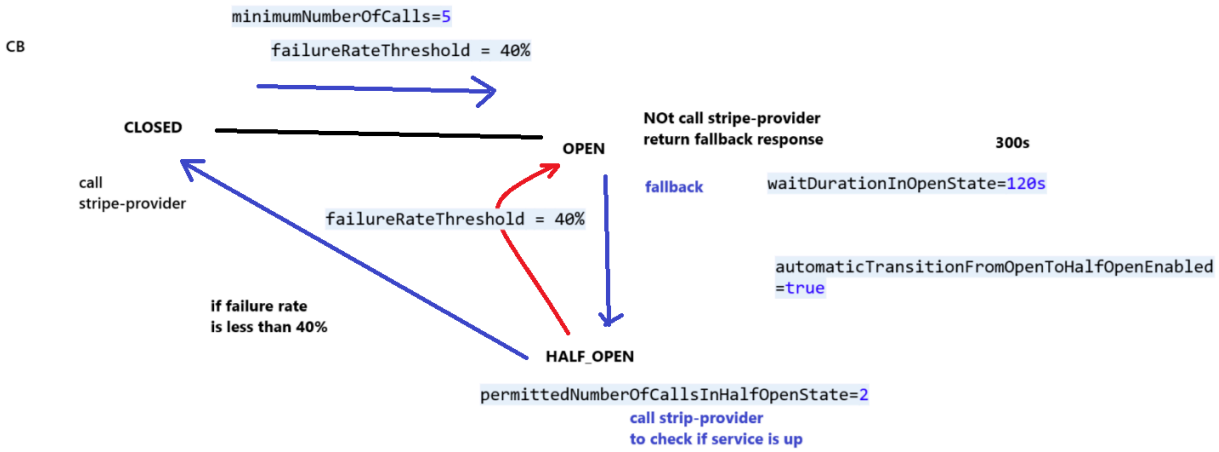
Scenario stop stripe-provider-service



Circuit breaker wraps the exception being thrown



How states are managed:



LIVE NOTES

Microservices..

as maximum as these 9 attributes.

1. Componentization via services
2. Organized around business capabilities
3. Decentralized governance
4. Decentralized data management (When possible)
5. Infrastructure automation
6. Smart Endpoints and Dumb Pipes:

===
7. Products, Not Projects:
8. Design for Failure:
9. Evolutionary Design:

Products, Not Projects:

Product based org.

working on products

designing

tech descisions

sense of responsibility will be more.
more sense of ownership required.

projects
1-2yrs

Build product that it should run very stably even when you are not there in the project.

==========

Design for Failure

processing => stripe-provider

stripe-provider => StripePSP
Success
Error

turn off the feature..

Circuit Breaker

Circuit Breaker

Once you realise that other system is not responding, or not behaving as expected. THen even if you call them for more request processing, it is highly likely that those new requests will also fail.

As soon as you noticed that other system is not behaving properly, instead of calling it, you do dummy handling at your end. (Assume that already system is not working).

How to implement in SpringBoot 3.x
Java library Resilence4j

AOP to accomplish CB mechanism.

Actuator
- realtime health monitoring of your application.
- check the internal state of the application.
- pom.xml dependency needs to be added.
- health-monitoring URL.

AOP - CB

======

CB + AOP + Actuator

3 dependencies in POM.xml

1. End-to-end successful API Call. with breakpoints
2. bring stripe-provider down. & check with breakpoint.

when you are not getting responses from external system - only these cases are eligible for CB logic

CB Reslience4J

properties
required configuration for CB

java code
@CircuitBreaker

fallbackMethod
logic of what to do when unable to get response.
throw new ProcessingException(

ErrorCodeEnum.UNABLE_TO_CONNECT_TO_STRIPE_PS.getErrorCode(),

ErrorCodeEnum.UNABLE_TO_CONNECT_TO_STRIPE_PS.getErrorMessage(),
HttpStatus.INTERNAL_SERVER_ERROR);

How will Circuit Breaker Know that we are unable to connect to external system???

CB, checks if end system is working.
tracks the exceptions that you throw in your method.
and uses this exception count to interprete that end system is working or not.

If its working, then call the functional method
if its not working, then call fallback method

Currently even tough CB is catching exception & calling fallback method.. still for next request its still calling stripe-provider.. So CB logic is not bypassing the stripe-provider invocation.

We need to additionally configure & tell CircuitBreaker when to stop calling stripe-provider.

properties

Status management

1 time you tried to call stripe-provider. & it failed.
it means 100% failure rate.
Just deciding to go in open state based on 1 time failure is not good.

"payment-processing-service": {

"status": "CIRCUIT_OPEN",
"details": {
"failureRate": "100.0%",
"failureRateThreshold": "40.0%",
"slowCallRate": "0.0%",
"slowCallRateThreshold": "100.0%",
"bufferedCalls": 5,
"slowCalls": 0,
"slowFailedCalls": 0,
"failedCalls": 5,
"notPermittedCalls": 0,
"state": "OPEN"
}
}

In OPEN state, when you invoke, directly fallback method is called, not the functional method. & in acutally, we notice below field incrementing.
notPermittedCalls": 7,

After 300s, automatically move to HALF_OPEN state
"payment-processing-service": {
"status": "CIRCUIT_HALF_OPEN",
"details": {
"failureRate": "-1.0%",
"failureRateThreshold": "40.0%",
"slowCallRate": "-1.0%",
"slowCallRateThreshold": "100.0%",
"bufferedCalls": 0,
"slowCalls": 0,
"slowFailedCalls": 0,
"failedCalls": 0,
"notPermittedCalls": 0,
"state": "HALF_OPEN"
}
}

==========
In HALF_OPEN STATE

- we tried 2 times ("bufferedCalls": 2)
- both the 2 calls failed ("failedCalls": 2)
- 100% failurerate ("failureRate": "100.0%")
- Is failureRather more than equal to ("failureRateThreshold": "40.0%")
- yes, then move status to OPEN
=> HALF_OPEN => OPEN

"payment-processing-service": {
"status": "CIRCUIT_OPEN",
"details": {
"failureRate": "100.0%",
"failureRateThreshold": "40.0%",
"slowCallRate": "0.0%",
"slowCallRateThreshold": "100.0%",
"bufferedCalls": 2,
"slowCalls": 0,
"slowFailedCalls": 0,
"failedCalls": 2,
"notPermittedCalls": 0,
"state": "OPEN"
}
}

WHILE we are in OPEN state, we dont check with external service, its its running. we directly return back fallback response.

even if external service is up & functional, we will still return fallback response

CLOSED

keep calling 3rd party stripe-provider endpoint

end-to-end functional system

OPEN

you cannot call 3rd party

you have to return response from fallback method only.

HALF_OPEN