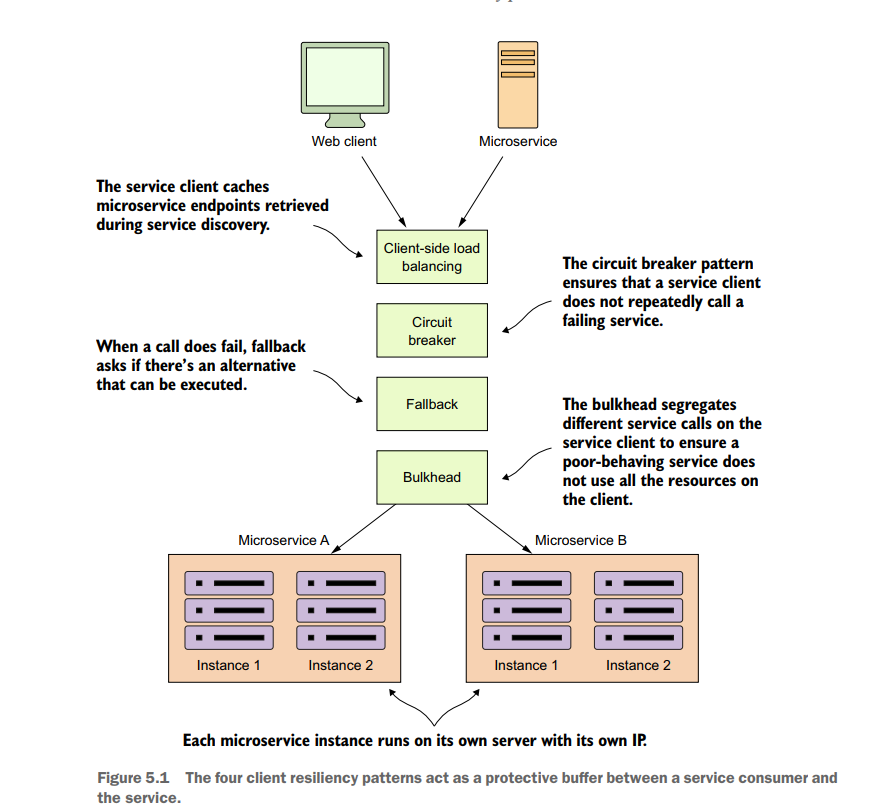
# Client resiliency pattern with spring Cloud and Netflix Hystrix

* Khi một service bị crash thì có thể dễ dàng phát hiện, tuy nhiên khi service bị chậm hoặc hiệu năng kém thì khó bị phát hiện bởi vì:
* Sự xuống cấp của service có thể bị gián đoạn : đầu tiên có một nhóm người dùng nhỏ phàn nàn về service , và đột nhiên toàn bộ ứng dụng bị quá tải và crash
* Gọi đến các service khác (remote service) thường là đồng bộ và không bị ngắt nếu khoảng thời gian đợi quá timeout
* Ứng dụng được thiết kế để xử lý với lỗi hoàn toàn do tài nguyên từ xa chứ không xử lý với sự xuống cấp 1 phần

1. Client-side resiliency pattern

* Client resiliency pattern mục đích để bảo vệ các tài nguyên từ xa của client khỏi bị crash khi yêu cầu tài nguyên ngày fail do remote service trả về lỗi hoặc hiệu năng kém
* Mục đích chính của pattern này là cho phép client nhanh chóng nhận được lỗi để không tiêu tốn tài nguyên của serve như connection DB, thread pool và ngăn chặn việc remote service tiêu tốn luồng kết nối của client.
* 4 Pattern: các impl của pattern này đứng giữa client request resource và resource trên server:
  + Client-side load balancing
  + Circuit breaker
  + Fallbacks
  + Bulkheads



* 1. Client side load balancing
* Liên quan đến việc client sẽ tìm các thông tin instance của service từ service discovery agent(Eureka) và caching các thông tin (địa chỉ vật lý) của các instance đó.
* Khi cần gọi đến service instance đó client-side load balacing sẽ trả về thông tin từ pool
* Khi phát hiện các service instance trả ra lỗi nó có thể dễ dàng loại bỏ nó khỏi pool để ngăn chặn client gọi đến service instance này
  1. Circuit Breaker