마이클 T 피셔의 The Art of Scalability

1. X 축: 수평 확장 (Horizontal Scaling)

- **설명**: 동일한 애플리케이션 인스턴스를 여러 대 추가하여 부하를 분산하는 방식입니다.
  즉, 동일한 기능을 수행하는 여러 복제본을 생성하여 처리량을 늘립니다.
- **특징**:
  - **로드 밸런싱**이 중요: 트래픽을 여러 서버로 균등하게 분배.
  - 인스턴스 추가만으로 확장이 가능하므로 상대적으로 구현이 간단.
  - 가장 일반적이고 초기 확장에서 자주 사용되는 방법.
- **예시**:
  - 애플리케이션 서버를 1 대에서 10 대로 늘려 트래픽을 분산.
  - 쿠버네티스에서 ReplicaSet 을 통해 파드를 복제.
  - AWS Auto Scaling Group 을 이용한 EC2 인스턴스 확장.
- **장점**:
  - 특정 서비스에 대한 부하가 늘어나더라도 동일한 인스턴스를 복제하여 대응 가능.
  - 초기 비용이 적음.
- **한계**:
  - 상태 유지(Stateful) 애플리케이션에서는 적용이 어려움.
  - 데이터베이스 등의 단일 장애 지점(SPOF)은 해결되지 않음.

---

2. Y 축: 서비스 분할 (Service Decomposition)

- **설명**: 시스템의 기능을 여러 개의 서비스로 분할하는 방식입니다.
  즉, 기능별로 독립적인 서비스(혹은 마이크로서비스)로 나누어 확장성을 확보합니다.
- **특징**:
  - 각 서비스는 독립적으로 배포, 확장, 유지보수 가능.
  - 서비스 간 **API 통신**(REST, gRPC, 메시지 큐 등)을 통해 상호작용.
  - MSA 의 핵심 철학.
- **예시**:
  - 전자상거래 시스템을 "상품", "주문", "결제"와 같은 독립적인 서비스로 분리.
  - Netflix 가 고객별 추천 시스템과 스트리밍 시스템을 각각 독립 서비스로 분리.
- **장점**:
  - 특정 서비스만 확장 가능, 리소스 활용 최적화.
  - 각 서비스가 다른 기술 스택을 사용할 수 있음 (폴리글롯 아키텍처).

    - 장애가 한 서비스에 국한되므로 전체 시스템의 안정성이 높아짐.
- **한계**:
    - 분산 시스템의 복잡성 증가 (서비스 간 통신, 데이터 일관성 등).
    - 초기 설계 비용이 높고, 조직적/기술적 준비가 필요.

---

3. Z 축: 데이터 분할 (Data Partitioning)

- **설명**: 데이터를 여러 분할(Shard)로 나누어 저장하고 처리하는 방식입니다. 즉, 데이터를 분산하여 각 노드의 처리 부하를 줄이고 확장성을 높입니다.
- **특징**:
    - 데이터는 특정 기준(예: 사용자 ID, 지리적 위치)에 따라 분할.
    - 각 데이터 파티션은 독립적인 서버나 데이터베이스에서 관리.
    - 글로벌 시스템에서 자주 사용됨.
- **예시**:
    - 트위터가 사용자 ID 를 기준으로 데이터를 여러 데이터베이스에 분산.
    - MongoDB, Elasticsearch 등에서의 샤딩(Sharding).
    - 지리적 분산: 미국 고객은 북미 데이터센터, 아시아 고객은 아시아 데이터센터에서 처리.
- **장점**:
    - 대규모 데이터를 효과적으로 처리 가능.
    - 특정 데이터 파티션만 영향을 받으므로 장애 격리성이 높아짐.
- **한계**:
    - 분할 기준 설계가 어렵고, 재분할(Resharding)은 매우 복잡.
    - 데이터 일관성 및 동기화 관리가 어려움.

세 축의 상호작용과 조합

- **X 축**은 시스템의 처리량이 급격히 증가할 때 가장 먼저 적용되는 확장 방법입니다.
- **Y 축**은 시스템이 복잡해지고 기능별로 트래픽 특성이 다를 때, 또는 개발 및 배포 속도를 개선하기 위해 사용됩니다.
- **Z 축**은 데이터가 방대해지고 처리 병목이 발생할 때 사용됩니다.

확장 전략의 조합

1. **X 축 + Y 축**:
    - 마이크로서비스를 독립적으로 분리(Y 축)하고, 각 마이크로서비스를 수평으로 확장(X 축).

    - 예: Netflix 는 수백 개의 마이크로서비스로 나누고, 각각을 독립적으로 확장.
2. **Y 축 + Z 축**:
    - 서비스 분리를 통해 각각의 서비스가 데이터를 독립적으로 관리(Y 축)하면서, 데이터도 분산 저장(Z 축).
    - 예: 전자상거래 시스템에서 "주문" 서비스는 사용자 ID 를 기준으로 데이터 분산(Z 축).
3. **X 축 + Z 축**:
    - 동일 애플리케이션을 수평 확장(X 축)하면서, 데이터도 분할 저장(Z 축).
    - 예: 트위터에서 데이터베이스 샤딩(Z 축)과 API 서버 복제(X 축) 조합.

MSA 구축 시 세 축의 적용

1. 초기 단계:
    - **X 축** 확장을 우선적으로 적용해 서버 부하를 분산.
2. 성장 단계:
    - **Y 축** 서비스 분리를 통해 팀별 독립성과 기능 확장을 지원.
3. 대규모 데이터:
    - **Z 축** 데이터 분산 및 파티셔닝으로 대규모 데이터 처리를 최적화.