5. 인터페이스 구현

시스템 인터페이스 요구사항 분석

1. 시스템 인터페이스 요구사항 분석 절차
    a. 요구사항 선별
    b. 요구사항 관련 자료 준비
    c. 요구사항 분류
    d. 요구사항 분석 및 명세서 구체화
    e. 요구사항 명세서 공유

모듈 연계를 위한 인터페이스 기능 식별

1. 모듈 연계
    - 내부 모듈과 외부 모듈 또는 내부 모듈 간 데이터 교환을 위해 관계를 설정하는 것

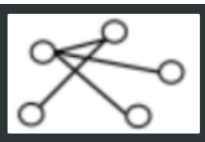
2. EAI (Enterprise Application Integration)
    - 기업 내 각종 애플리케이션 및 플랫폼 간의 정보 전달, 연계, 통합 등 상호 연동이 가능하게 해주는 솔루션

    - EAI 구축 유형

Point-to-Point

: 애플리케이션을 1 : 1로 연결

: 변경 및 재사용이 어려움

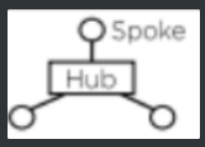


Hub & Spoke

: 단일 접점인 허브 시스템을 통해 데이터를 전송하는 중앙 집중형 방식

: 확장 및 보수가 용이

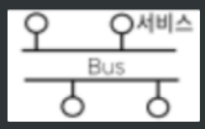
: 허브 장애 시 시스템 전체에 영향



Message Bus

: 애플리케이션 사이에 미들웨어를 두어 처리

: 확장성이 뛰어나고 대용량 처리 가능



Hybrid

: Hub & Spoke 와 Message Bus의 혼합 방식

: 그룹 내에는 Hub & Spoke, 그룹 간에는 Message Bus 사용

: 데이터 병목 현상 최소화

3. ESB (Enterprise Service Bus)
   - 애플리케이션 간 표준 기반의 인터페이스를 제공하는 수단
   - EAI와 유사하지만 애플리케이션 보다 서비스 중심의 통합을 지향
   - 애플리케이션과의 결합도를 약하게 유지

4. 웹 서비스 (Web Service)
   - 네트워크의 정보를 표준화된 서비스 형태로 만들어 공유하는 기술
   - 서비스 지향 아키텍쳐 (SOA) 개념을 실현하는 대표적 방법
   - 구성
     1. SOAP

        : HTTP / HTTPS, SMTP를 활용하여 XML 기반의 메시지를 교환하는 프로토콜
     2. UDDI

        : WSDL을 등록하여 서비스와 서비스 제공자를 검색하고 접근하는데 사용
     3. WSDL

        : 웹 서비스에 대한 정보를 XML 형식으로 구현

인터페이스 구현

1. 인터페이스 구현
   - 송 수신 시스템 간의 데이터 교환 및 처리를 실현하는 작업

2. 데이터 통신을 이용한 인터페이스 구현
    - 데이터 포맷을 인터페이스의 대상으로 전송하면 수신 측에서 파싱하여 해석
    - JSON이나 XML 형태의 데이터 포맷을 사용하여 인터페이스 구현

3. 인터페이스 엔티티를 이용한 인터페이스 구현
    - 인터페이스가 필요한 시스템 사이에 별도의 인터페이스 엔티티를 두어 상호 연계
    - 인터페이스 테이블을 엔티티로 사용

4. JSON (JavaScript Object Notation)
    - 데이터 객체를 Key-Value 쌍으로 표현하는 개방형 표준 포맷
    - AJAX에서 XML을 대체하여 사용

5. AJAX (Asynchronous Javascript And Xml)
    - Javascript를 이용하여 클라이언트와 서버간의 XML 데이터를 주고받는 비동기 통신 기술
    - 새로고침 없이 웹의 일부 영역 업데이트 가능

인터페이스 보안

1. 인터페이스 보안
    - 보안성 향상을 위해 인터페이스의 보안 취약점을 분석한 후 적절한 보안 기능을 적용하는 것

2. 인터페이스 보안 기능 적용
    a. 네트워크 영역

    : 인터페이스 송 수신 간 스니핑을 이용한 데이터 위협을 방지하기 위해 트래픽에 대한 암호화 설정

    : IPSec, SSL, S-HTTP

    b. 애플리케이션 영역

    : 소프트웨어 개발 보안 가이드를 참조하여 코드 상의 보안 취약점을 보완하는 방식

    c. 데이터베이스 영역

    : 데이터베이스 동작 객체의 보안 취약점에 보안 기능 적용

    : 개인 정보나 민감한 데이터의 경우 암호화나 익명화 같은 데이터 자체 보안 방안도 고려

3. 데이터 무결성 검사 도구
    - 인터페이스 보안 취약점을 분석하는데 이용
    - 시스템 파일의 변경 유무 확인 후, 파일 변경시 관리자에게 알림
    - Tripwire, AIDE, Samhain, Claymore, Slipwire, Fcheck

인터페이스 구현 검증

1. 인터페이스 구현 검증
    - 인터페이스가 정상적으로 작동하는지 확인

- 인터페이스 구현 검증 도구와 감시 도구를 이용하여 인터페이스 동작 상태 확인

2. 인터페이스 구현 검증 도구
   a. xUnit
      : Java, C++, .Net 등 다양한 언어를 지원하는 단위 테스트 프레임워크
   b. STAF
      : 서비스 호출 및 컴포넌트 재사용 등 다양한 환경을 지원하는 테스트 프레임워크
      : 크로스 플랫폼이나 분산 소프트웨어에서 테스트 환경 조성 가능하도록 지원
   c. FitNesse
      : 웹 기반 테스트 케이스 설계, 실행, 결과 확인 등을 지원하는 테스트 프레임워크
   d. NTAF
      : FitNesse의 장점 (협업 기능)과 STAF의 장점 (재사용 및 확장성)을 통합한 NHN의 테스트 자동화 프레임워크
   e. Selenium
      : 다양한 브라우저 및 개발 언어를 지원하는 웹 애플리케이션 테스트 프레임워크
   f. watir
      : Ruby를 사용하는 애플리케이션 테스트 프레임워크

3. 인터페이스 구현 감시 도구
   - 인터페이스 동작 상태는 APM (애플리케이션 성능 관리)를 사용하여 감시할 수 있음
   - 대표적인 APM
     1. 스카우터 (Scouter)
        : 애플리케이션 및 OS 자원에 대한 모니터링 기능 제공하는 오픈소스 APM 소프트웨어
     2. 제니퍼 (Jennifer)
        : 애플리케이션의 전 단계에 걸쳐 성능을 모니터링하고 분석해주는 소프트웨어

4. APM (Application Performance Management/Monitoring)
   - 애플리케이션 성능 관리를 위해 다양한 모니터링 기능을 제공하는 도구
   a. 리소스 방식 - Nagios, Zabbix, Cacti
   b. 엔드투엔드 방식 - VisualVM, 제니퍼, 스카우터