침입감내 소프트웨어 모델링을 위한 요구사항 추출 및 명세

조은숙*, 이강신**

Extraction and Specification of Requirements for Intrusion Tolerant Software Modeling

Eun Sook Cho, Kang Shin, Lee

Abstract

Current distibuted systems are attacked from the outside as well as in which new intrusions are occurred. In particular there is a growing but largely unnoticed intrusion threat due to the emerging middleware technologies such as CORBA, WAP, XML support, and enterprise application integrators. In order to cope with these attacks, intrusion tolerance technology is introduced. Intrusion tolerance technology means that it can provide services normally although attacks are occurred into system. There are intrusion tolerance architectures such as ITUA, HACQIT, SITAR, and so on as a part of DARPA project. In this paper, we analyze and discuss existing intrusion tolerance architectures with respect to intrusion tolerance technology. Also, we extract intrusion tolerant requirements, which are required to develop intrusion tolerant system. We propose UML-IT(Intrusion Tolerance) profiles and specify intrusion tolerant software by applying UML-IT profiles.

Key Words: 컴포넌트기반 개발, 컴포넌트 참조 모델, 컴포넌트 메타 모델

* 동덕여자대학교

** 한국정보보호진흥원

1. 서론

최근에 보고되는 침해사고들은 점점 더 복잡하고 교묘한 공격 방법들을 사용하여 피해 규모를 키워가고 있다. 또한 침해 사고의 원인이 되는 시스템, 네트워크, 소프트웨어의 취약성들이 연간 3000여개 이상 발견되는 등 잠재적인 침해사고의 원인들도 매우 급속하게 증가하고 있다. 이러한 침해사고를 예방하고 효과적인 대응방법을 마련하기 위하여 침입차단 기술, 침입탐지기술 등의 다양한 정보보호기술들이 개발되고 있다. 이런 기술들은 알려진 취약점에 대한 공격 예방과 탐지를 제공하며, 알려지지 않은 취약점에 대한 공격에는 적절하게 대응하지 못하는 단점이 있다. 그러므로 이와 같이 알려지지 않은 취약점이나 공격 방법에 의한 침해 사고를 방지하기 위한 기술이 필요하며 침입감내기술(Intrusion Tolerance Technology)이 이에 대한 한가지 해결책으로 제시될 수 있다[1,2].

침입감내 기술이란 중요한 서비스를 제공하는 시스템에 대한 공격이 발생하더라도 정상적인 서비스를 제공할 수 있도록 하는 기술이다. 이러한 침입감내 기술은 결함허용기술과 침입차단 기술이나 탐지 기술 등의 정보보호 기술들이 결합된 형태로 미국과 유럽에서 비교적 최근에 시작되었다[3]. 침입감내 기술에 대한 연구는 오래전부터 수행되어 왔으나 많은 연구자들이 집중적으로 연구를 시작한 것은 비교적 최근의 일이다. 유럽에서는 FP5의 IST 프로그램을 통하여 관련 연구를 진행하였고, 미국에서도 DARPA(Defense Advanced Research Project Agency)의 지원을 통하여 연구들을 진행하고 있다.

그러나 현재 이러한 침입 감내 기술을 적용한 소프트웨어 모델링에 대한 연구는 거의 이루어지지 않은 실정이다. 특히 객체지향 소프트웨어 모델링 언어 표준인 UML은 이러한 침입 감내와 관련된 요구사항들을 명세하는 장치들이 부족하며, 여러 모델링 기법들에서도 침입 감내에 특화된 명세 기법들이 제시되어 있지 않다.

본 논문에서는 이러한 침입 감내 기반 소프트웨어를 개발하는 데 있어서 필요한 요구사항들을 추출하고 추출된 요구사항들을 소프트웨어 모델링 과정에 명세하기 위한 기법을 제시한다.

논 논문은 다음과 같이 구성된다. 2장에서는 DARPA 프로젝트의 일환으로 연구된 침입감내 아키텍처들에 대해 살펴본다. 3장에서는 이러한 비교를 통하여 침입 감내 시스템을 구축하는데 있어서 침입 감내와 관련된 요구사항들을 도출한 결과를 제시한다. 4장에서는 추출된 요구사항들을 소프트웨어 모델링에 반영하기 위한 UML-IT 프로파일과 명세 기법을 제시한다. 마지막으로 5장에서 결론으로 맺는다.

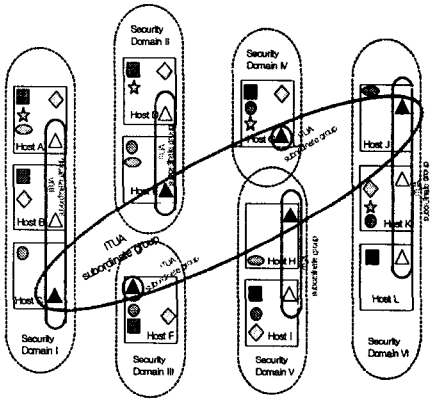
2. 관련 연구

2.1. ITUA(Intrusion Tolerance by Unpredictable Adaptation)

ITUA 프로젝트는 2000년 7월에 시작해서 2003년 12월에 종료되는 프로젝트로서[4] ITUA의 주된 특징으로는 두 가지를 들 수 있다. 첫째는 혼합 결합 복제 메커니즘이고 둘째는 예측 불가능에 대한 대응 메커니즘이다. 또 다른 특징으로는 ITUA는 여러 그룹들 간 통신이 가능하도록 한다.

ITUA 침입 모델은 4가지 요소 즉, 항목(Item), 행동(Action), 가정(Assumption), 그리고 명세로 구성되어 있다. 또한 ITUA 아키텍처는 복제자(replica), 관리자(manager), 그리고 subordinate들로 구성되어 있다. ITUA는 이러한 컴포넌트들로 구성되어 있으면서 in-band와 out-of-band 적응을 통해 지원하는 여러 종류의 적응들을 관리하기 위해 분산화된(decentralized) 인프라스트럭처를 구현하고 있다[5]. 각각의 호스트는 하나의 관리자나 하

나의 종속자를 실행할 수 있다. 각각의 보안 도메인은 하나의 관리자를 실행하는 하나의 호스트를 갖는다. 따라서 결과적으로 동일 도메인에 있는 다른 호스트들은 종속자들을 실행하게 된다. 모든 관리자들은 "관리자 그룹"이라고 하는 하나의 그룹을 형성하게 된다. 한 도메인에 있는 종속자들과 그 관리자는 하나의 "종속자 그룹"을 형성한다. subordinate는 "보안 권고(security advising)"와 "복제 관리(replication management)"라는 두 가지 의무를 갖는다.



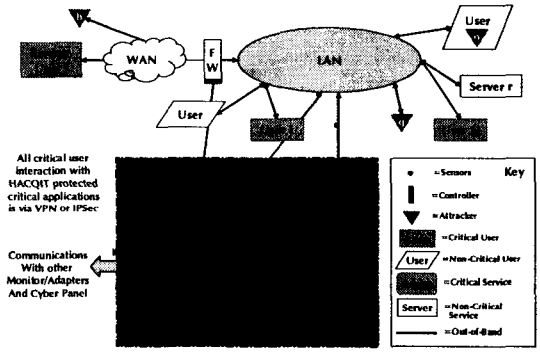
<그림 1> ITUA 아키텍처

ITUA에서는 침입 감내와 관련된 특성들 가운데 가용성은 QuO(Quality Objects) 적응 미들웨어를 통해 제공하며, 신뢰성은 Intrusion Tolerant GCS(Group Communication System)와 복제 그룹을 통해 이루어지며, 기밀성은 보안 도메인을 통해 이루어지고, 무결성은 센서 작동 장치를 통해 제공된다.

2.2 HACQIT

HACQIT(Hierarchical Adaptive Control of Quality of service for Intrusion Tolerance) 프로젝트는 공격 시에 사용자 성능 25% 이상의 저하를 방지하면서 네 시간 동안의 침입 감내 제공을 목표로 하고 있다[6].

HACQIT 시스템은 또한 다음 사항들을 목표로 한다; 알려지지 않은 공격(서명이 확실하지 않은 공격이나 완전히 새로운 공격)에 대해 방어할 수 있는 능력을 갖추는 것, 주요한(critical) 컴포넌트들은 따로 떨어진 장소에 두어서 공격으로부터 지키는 것, 이전에 관찰되었던 공격을 식별하고 방어하거나 피하는 것, 롤백을 위해 잘 유지된 백업들을 사용하는 것, 공격당한 서버를 회복시키는 것, 결함의 양을 최소화하는 것, 그리고 시스템 부하로 인한 HACQIT의 수고를 최소화하는 것이다[7].



<그림 2> HACQIT 아키텍처

<그림 2>는 HACQIT 아키텍처로서, 중요 어플리케이션들, 중요 서버들 그리고 다른 서버(sandbox)들이 함께 보호된 서브 도메인 내에 있는 것을 강조하여 설명하고 있다.

HACQIT 시스템 내의 두 대의 서버, 주 서버와 백업 서버가 제공하는 서비스는 침입 차단기능을 수행하는 게이트웨이를 통해서 사용자와 연결된다. 이들을 모니터하고 제어하는 모니터 & 어댑터는 결함 진단 소프트웨어를 포함하고 있다. 그림에서 점선으로 나타낸 연결은 별도의 out-of-band 네트워크로 구성되어 일반 사용자가 접근할 수 없도록 되어 있다. 샌드 박스(Sandbox)는 주 서버 및 백업 서버의 데이터 복제를 가지고 있어서, 두 서버의 결과가 일치하지 않는 경우 일시적인 문제인지, 공격에 의한 것인지를 파악하는데 이용

된다. HACQIT은 오류 검출과 실패 차단을 위해 중복성과 다양성을 복합적으로 이용한다. 즉 한 개의 요청은 주 서버 뿐만 아니라 백업 시스템에도 전달되며 모니터와 어댑터는 각 시스템에서 오는 응답을 비교하여 두 응답이 서로 같으면 고장이나 침입이 없는 것으로 간주한다. 만일 결과 값이 다르면 고장으로 간주되어 더 이상의 피해가 확산되지 않도록 하기 위해 어떤 웹 서버가 공격받은 것인가를 판단하고 수리 및 복구 절차를 행한다.

HACQIT 아키텍처의 샌드 박스는 가용성을 제공하며 결함 발생시 복제된 데이터나 서비스를 통해 공격이 계속적으로 일어남에도 불구하고 어느 정도의 성능을 유지할 수 있게 해주어 신뢰성을 제공한다. 모니터와 어댑터는 결함을 진단하는 기능을 포함하고 있어 요청이나 들어오는 패킷에 대한 무결성을 검사해 주는 기능을 포함한다. HACQIT은 센서를 통해 침입을 감지하여 공격이 전파되는 것을 최소화시킬 수 있기 때문에 신뢰성과 무결성을 제공한다.

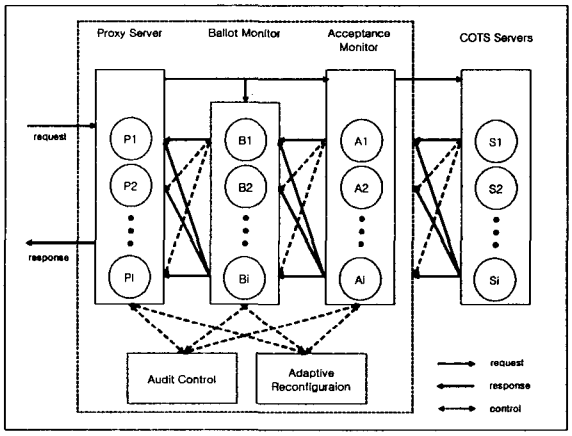
게이트웨이와 방화벽을 통해 기밀성과 무결성을 제공한다.

2.3 SITAR

SITAR(Scalable Intrusion Tolerance Architecture)는 분산 서비스를 제공하는 고가용성의 침입 감내 시스템을 위한 확장성 있는 아키텍처 및 프레임워크이다[8].

프로젝트의 목적은 침입 감내 시스템을 구축하기 위한 아키텍처를 설계하고, 프로토타입을 개발하여 실제로 사례 연구를 통하여 평가를 하는 것이다. 이 과정을 통해 고가용성의 서비스를 구축할 수 있는 새로운 기술들을 고안해 내어 모든 새로운 시스템을 개발하거나 COTS를 통해 시스템을 개발하거나 혹은 기존의 시스템을 더욱 견고하게 만들어 주는데 유용 하도록 하는 것이다. SITAR의 접근 방법에서 잘 알려지지 않은 공격을 극복할 수 있도록 침입에 대한 관점을 공격자와 공격들 자체로부터 보호해야 할 대상, 즉 본질적으로 제공하는 기능들과 서비스들로 이동시켰다. 이것은 위협의 주체가 악의적인 공격이거나, 자연적인 실패이거나, 혹은 인위적인 사고이거나를 결정하기 이전에 그것에 대한 영향에 대항하여 시스템이 지속적으로 살아남을 수 있도록 한다는 것이다[9].

<그림 3>은 서비스 아키텍처의 논리적인 뷰이다.



<그림 3> 일반적인 침입 감내 서비스 아키텍처

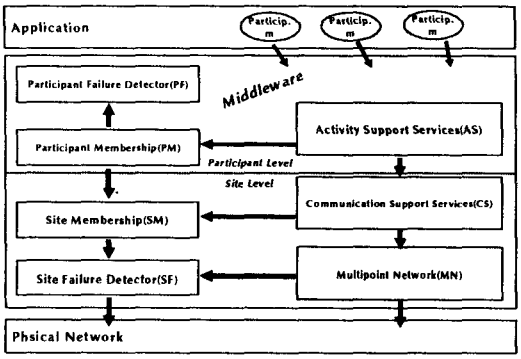
SITAR는 적응 재설정 모듈(Adaptive Reconfiguration Module)을 통해 기밀성과 가용성 서비스를 제공한다. 프락시 서버 또한 COTS 서버와 사용자의 직접적인 접근을 막고 침입을 감내하기 위한 서비스를 제공하여 높은 가용성을 보장한다. 투표 모니터(Ballot Monitor)는 처리된 서비스의 결과값, 즉 응답값을 최종적으로 결정하여 높은 신뢰성을 지원한다. 응답 모니터를 통해 무결성을 제공하며, 감시 제어를 통해 기밀성을 제공한다.

2.4 MAFTIA

MAFTIA(Malicious-and Accidential-Fault

Tolerance for Internet Applications)는 대규모 분산 시스템에서 우발적인 결함과 악의적인 공격을 감내하기 위하여 포괄적인 접근 방법의 조사가 유도된 프로젝트이다[10,11].

MAFTIA의 아키텍처는 <그림 4>에서 보는 바와 같이 로컬 토폴로지와 모듈간의 의존성 관계를 표현하고 Site 계층과 Participant 계층으로 분류된 계층을 보여준다[12].



<그림 4> MAFTIA 아키텍처

각 계층별을 구성하는 요소를 통해 다음과 같은 특징을 가지고 있다. 첫째, 멀티포인트 네트워크를 위한 멀티포인트 어드레싱을 제공한다. 이를 위해 로컬넷과 글로벌넷 개념을 적용하고 있다. 둘째, 최선의 노력을 시도하는 전송(Best-effrot Delivery) 기법을 제공한다. 그리고 라우팅 메커니즘을 동적으로 제공한다. 따라서 이를 통해 가용성을 향상시킨다. 그리고 기본적인 보안 채널과 메시지 엔빌로프를 제공함으로써 통신 연결에 대한 보호와 메시지 보호를 위한 메커니즘으로 제공하고 있다. 셋째, 통신 지원 서비스로 분산 암호 해독법, 그룹통신, 시간과 시계의 동기화 메커니즘 등을 제공하고 있으며, 활동 지원 서비스로 중복성 관리, 키 관리, 트랜잭션 관리 서비스들을 제공하고 있다.

MAFTIA 는 멀티 포인트 어드레싱을 통해 가용성을 제공하고, 무결성은 통신 지원 서비스의 그룹통신과 활동 지원 서비스대칭키 관리와 트랜잭션 관리를 통해 만족된다. 기밀성은 활동 지원 서비스의 대칭키 관리와 영구적인 서비스, 동시성 서비스를 통해서 제공된다.

3. 침입 감내 시스템을 위한 요구사항

이 장에서는 2장에서 분석한 각각의 아키텍처들의 특징들과 요소들을 바탕으로 침입감내 시스템을 구축하는 데 있어서 침입 감내와 관련된 요구사항들을 도출하여 정의한다. 특히 침입 감내와 관련된 서비스를 비기능적인 측면과 기능적인 측면으로 나누어 정의하며, 최종적으로 이렇게 각각 도출된 비기능적 요소와 기능적 요소간의 대응관계로 비기능적 요소에 필요한 기능 요소들을 파악할 수 있도록 정의한다.

3.1 비 기능적 요구사항

3.1.1 가용성(Availability)

가용성은 "Readiness of Usage" 를 의미하는 것으로, 데이터의 부분 혹은 서버의 변경 및 삭제에도 불구하고, 시스템이 서비스의 품질 요구사항을 이행하면서 적법한 사용자에게 서비스를 제공할 수 있는 정도 혹은 시스템이 클라이언트에게 서비스할 수 있는 시간의 정도이다.

가. 미들웨어 가용성

미들웨어 가용성은 각기 분리된 두 개의 어플리케이션 사이에서, 매개 역할을 하거나 연합시켜주는 미들웨어의 정보가 변경되거나 미들웨어 자체에 비정상적인 상태가 발생함에도 불구하고, 적법한 사용자에게 서비스를 제공할 수 있는 정도를 의미한다. 아래와 같은 미들웨어 가용성이 존재한다.

- 계층별 독립성
- 메시지 가용성

o 네이밍 서비스 가용성
o 서비스 가용성

나. 어플리케이션 가용성

어플리케이션 가용성은 적법하지 않은 오퍼레이션 사용자로 인한 공격이나 전달되는 메시지가 유효하지 않은 경우가 발생하더라도 서비스의 클라이언트에게 계속적으로 기대 수준의 서비스를 제공할 수 있는 정도를 의미한다. 다음과 같은 어플리케이션 가용성이 필요하다.

o 사용자의 인증 가용성
o 예외 핸들링 가용성
o 충돌 방지 가용성
o 코딩 결함 방지 가용성
o 비밀번호의 복잡성
o 저장소 침입 가용성
o 데이터 타입 가용성
o 데이터 길이의 가용성
o 데이터 값의 보존성

3.1.2 신뢰성(Reliability)

신뢰성은 명시된 운영 환경 및 조건에서 서비스가 일정시간 동안 운영될 수 있는 정도를 의미한다.

가. 미들웨어 신뢰성

미들웨어 신뢰성은 미들웨어를 구성하는 각 계층에서 데이터 혹은 관련 기능들에 대한 결함이 발생되더라도 주변 계층에게 영향을 미치지 않고, 명시된 성능 및 서비스를 일정기간 동안 제공하는 정도를 의미한다.

o 계층별 독립성
o 상호작용 신뢰성
o 성능 신뢰성

나 어플리케이션 신뢰성

어플리케이션 신뢰성은 요구사항 명세서에 정해진 기능들을 정확하고 충실하게 제공하며, 한 기능의 오류 혹은 실패가 다른 기능에 악영향을 미치지 않으면서 운영되는 정도를 의미한다.

o 서비스 독립성
o 계산의 정확성
o 워크플로우 신뢰성
o 데이터 신뢰성

3.1.3 무결성(Integrity)

무결성은 어플리케이션의 상태가 부적절하게 변화되지 않는 정도를 의미한다. 여기서 '부적절하게' 의 의미는 '권한이 없는, 허가받지 않은' 의 의미를 포함한다.

가. 어플리케이션 무결성

어플리케이션 무결성은 어플리케이션 수준에서 포함하고 있는 데이터나 여러 기능에 대해 부적절한 상태 변화가 없는 정도를 의미한다. 어플리케이션 무결성은 어플리케이션 설계에 의해 상당 부분 영향을 받는다. 설계 시의 오류를 줄임으로써 어플리케이션 무결성을 향상시킬 수 있다. 어플리케이션 무결성에는 다음과 같은 세부 속성이 있다.

o 메시지 무결성
o 메소드 무결성
o 데이터 무결성
o 파일 무결성
o 워크플로우의 비정상적인 변경 정도
o 코드의 무결성
o 어플리케이션 오퍼레이션 메타 정보 무결성
o 어플리케이션 정보에 대한 메타 정보 무결성
o 어플리케이션 비거부성 (Non-repudiation)
o 어플리케이션 접근 제어 (Access Control)
o 어플리케이션 접근 검사 (Access Audit)

나. 미들웨어 무결성 (Middleware Integrity)

미들웨어 무결성은 미들웨어 계층에 대한 무결성으로 미들웨어가 제공하는 여러 시스템 레벨의 서비스나 미들웨어 데이터에 대한 부적절한 변화가 없는 정도를 의미한다. 미들웨어 무결성에는 다음과 같은 세부 속성이 있다.

o 메시지 무결성
o 서비스 모듈 무결성
o 데이터 무결성
o 미들웨어 오퍼레이션 메타 정보 무결성
o 미들웨어 정보에 대한 메타 정보 무결성
o 미들웨어 비거부성 (Non-repudiation)
o 미들웨어 접근 제어성 (Access Control)
o 미들웨어 접근 검사성 (Access Audit)

3.1.4 기밀성(Confidentiality)

기밀성은 어떠한 정보에 대한 비승인된 유출이 없는 정도를 의미한다.

가. 어플리케이션 기밀성 (Application Confidentiality)

어플리케이션 기밀성은 공격으로 인해 어플리케이션 계층에서 가지고 있는 정보(데이터, 메소드, 또는 메타 정보 등)에 대한 비승인된 유출이 없는 정도를 의미한다. 어플리케이션 기밀성에는 다음과 같은 세부 속성이 있다.

o 어플리케이션 데이터 기밀성
o 어플리케이션 메시지 기밀성
o 오퍼레이션 수행 결과의 기밀성
o 익명성 (Anonymity)
o 비밀성 (Privacy)
o 어플리케이션의 메타 정보 기밀성

나. 미들웨어 기밀성

미들웨어 기밀성은 미들웨어가 공격을 받았을 때 미들웨어 수준에서 이루어진 서비스 요청이나 어플리케이션으로 전달될 정보에 대한 비승인된 유출이 없는 정도를 나타낸다. 미들웨어 기밀성에는 다음과 같은 세부 속성이 있다.

o 미들웨어 서비스 기밀성
o 미들웨어 메시지 기밀성
o 미들웨어 데이터 기밀성
o 익명성 (Anonymity)
o 비밀성 (Privacy)
o 미들웨어의 메타 정보 기밀성

3.2 기능적 요구사항

침입 감내를 위한 기능적 요구 사항은 침입 자체에 대한 대응뿐만 아니라 침입으로 인한 파급 효과인 결함과 오류에 대한 대응 기능도 포함되어야 한다. 다음은 침입 감내를 위한 기능적 요구 사항들이다.

3.2.1 침입 탐지

침입의 존재나 그에 대한 결과들을 측정하는 기능과 요소를 포함한다.

o 네트워크 모니터링
o 시스템 자원 모니터링
o 서비스 모니터링
o 응답 검증(Responce Validation)
o 요청 검증(Request Validation)
o 트리거에 대한 분석(Auditing)
o 네트워크 로그 분석(Network Log Auditing)
o 트랜잭션 로그 분석(Transaction Log Auditing)
o 센서 설정(Sensor Deploying)

3.2.2 침입 방지

외부로부터 악의적인 목적으로 시스템에 접근하는 것을 차단하기 위한 기능과 요소들이 포함된다.

o 데이터의 무결성 검사
o 메시지의 무결성 검사
o 요청 검증
o 응답 검증
o 인증 검사(Authentication Validating)
o 권한 검사(Authorization Validating)
o 암호화(Encryption)
o 플랫폼의 다양성(Platform Diversity)
o 정보 혹은 서비스 소스 파악(Source Identifying)
o 데이터의 분리(Fragmentation)

<표 1> 미들웨어 수준에서의 기능적/비기능적 요구사항 매핑 표

비기능적 / 기능적	가용성		신뢰성		무결성		기밀성	
	M	A	M	A	M	A	M	A
네트워크 모니터링	o		o					
시스템 자원 모니터링	o	o	o	o	o	o		
서비스 모니터링	o	o	o	o	o	o	o	o
응답 검증	o	o	o	o	o	o		o
요청 검증	o	o	o	o	o	o	o	o
트리거에 대한 분석	o	o	o	o	o	o		
네트워크 로그 분석								
트랜잭션 로그 분석	o			o	o	o	o	o
센서 설정	o		o	o	o	o	o	o
데이터의 무결성 검사				o	o	o		
메시지의 무결성 검사			o	o	o	o		
인증 검사				o	o	o	o	o
권한 검사				o	o	o	o	o
암호화					o	o	o	o
플랫폼의 다양성	o							
정보 혹은 서비스 소스 파악			o	o	o	o	o	o
데이터의 분리								o
데이터의 분산	o	o		o		o		o
데이터의 복제	o	o		o				
서비스의 분산								
서비스의 중복								
멀티포인트 어드레싱	o	o						
서비스 이주		o		o				
손상된 요소의 고립	o	o			o	o	o	o
브로드케스팅	o	o			o	o	o	o
정보흐름 제어	o	o	o	o	o	o	o	o
워크플로우 제어	o	o	o	o	o	o	o	o
서비스 및 데이터 재전송	o	o	o	o		o		
데이터 접근 권한 재설정	o	o			o	o	o	o
서비스 접근 권한 재설정	o	o			o	o	o	o
시스템 운용 환경 재설정	o			o	o			
서비스 영향 추적 및 회복	o	o	o	o	o	o		
데이터 변경 추적 및 회복	o	o	o	o	o	o		

3.2.3 침입 감내

침입에 의한 손상을 억제하거나 완화하기 위한 기능과 요소들이 포함된다.

- o 데이터의 분산(Scattering)
- o 데이터의 복제(Redundancy, Copy)
- o 서비스의 분산(Scattering)
- o 서비스의 복제(Redundancy, Copy)
- o 멀티포인트 어드레싱

ㅇ서비스 이주(migration)
ㅇ손상된 요소의 고립(isolation)
ㅇ브로드케스팅
ㅇ정보흐름 제어
ㅇ워크플로우 제어

3.2.4 침입 회복

침입에 의한 결함 및 손상의 파급 효과를 제거하여 침입으로 인한 부정적 영향을 없애기 위한 기능들을 포함한다.

ㅇ서비스 및 데이터 재전송
ㅇ파일 접근 권한 재설정
ㅇ서비스 접근 권한 재설정
ㅇ시스템 운용 환경 재설정
ㅇ서비스 회복
ㅇ데이터 회복

3.3 요구사항간의 대응관계

이 절에서는 표1에 나타난 바와 같이 비기능적 요구사항들과 기능적 요구사항들 간의 대응관계를 제시한다.

4. UML-IT 기반 모델링

이 장에서는 3장에서 정의한 요구사항들을 소프트웨어 모델링 과정에 적용하기 위한 기법을 제시한다. 특히 현존 UML의 메커니즘을 확장한 UML-IT 프로파일을 정의하고, 이를 기능적,구조적,행위적 관점에서 명세하기 위한 기법을 제시한다.

4.1 UML-IT Profile

침입 감내 기반 소프트웨어 개발에 있어서 침입 감내 관련 요구사항들을 모델링에 반영하기 위해 현존 UML 메커니즘에 침입 감내와 관련된 사항들을 표현하기 위해 UML-IT Profile을 정의한다.

·{IT_Type[IT_Topic(Specific_Value)]} (IT_Mechanism)

여기서 IT_Type 은 비기능적인 요구사항의 종류를 의미한다. 비기능적인 속성의 종류는 가용성, 신뢰성, 무결성, 기밀성이 된다.

IT_Topic은 IT_Type의 세부 속성으로서, 세부 속성은 3장에서 정의된 가용성, 신뢰성, 무결성, 기밀성의 세부 속성들인 파일 기밀성, 메시지 기밀성, 어플리케이션 메타 정보 기밀성 등이 된다.

Specific_Value는 IT_Topic에 대한 구체적인 값이나 조건을 기술하는 것으로서 사용자가 요구한 비기능적 요구사항 중 추출될 수 있는 특정한 값이나 조건이다. 추출되지 않을 경우도 있으므로 선택적으로 표기한다.

{pre: (IT_condition)}

UML의 Note를 이용하여 UML 다이어그램의 대상 요소에 선조건을 명시한다. 선조건 표기의 예는 다음과 같다.

- {pre: (employee.sector=test.sector AND test.result in est.saferange) OR monitor.state = "ON" }

{post: (IT_condition)}

UML의 Note를 이용하여 UML 다이어그램의 대상 요소에 후조건을 명시한다. 후조건 표기의 예는 다음과 같다.

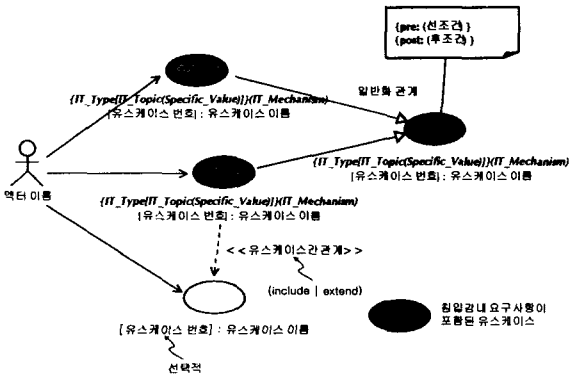
{post: (employee.sector=test.sector AND test.result in est.saferange) OR monitor.state = "OFF" }

IT_Mechanism 을 지원하기 위한 침입감내 메커니즘은 정의된 여러 기능성에서 추출할 수 있으며 비기능적인 요구사항을 만족시키기 위한 기법이 된다. IT_Mechanism은 분석 및 설계 단계에서 사용된다.

4.2 UML-IT를 적용한 유스케이스 명세

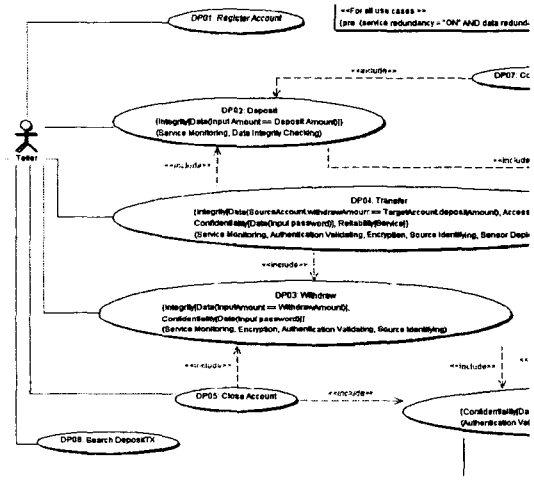
본 논문에서 제공하는 표기법에 따라 메카니즘을 기술하여 유스케이스에 침입감내 요구사항을 반영하도록 한다. 기존 유스케이스 다

이어그램은 단지 기능적인 부분만 명세하지만 본 논문에서는 침입 감내와 관련된 요구사항들을 유스 케이스에 반영하여 명세하도록 제시한다.



<그림 5> UML-IT를 적용한 유스케이스 명세

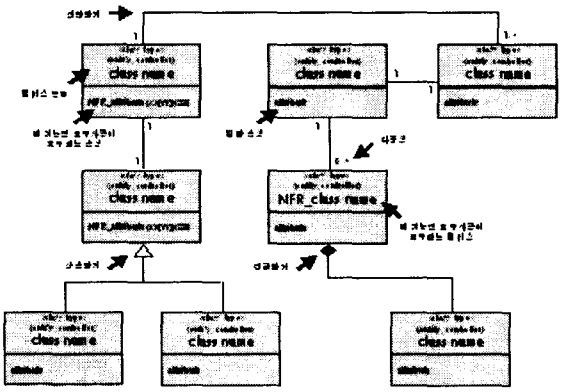
<그림 6>은 은행 업무의 수신관리 시스템에 대한 유스케이스 모델링의 일부를 보여주고 있다.



<그림 6> 수신 관리 유스케이스 다이어그램

4.3 UML-IT를 적용한 객체 모델 명세

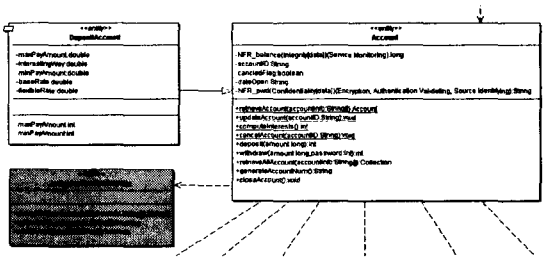
<그림 7>은 기존의 UML의 클래스다이어그램에 침입 감내와 관련된 요구사항들을 UML-IT Profile을 이용해서 확장 적용하기 위한 표현이다.



<그림 7> UML-IT를 적용한 객체모델 명세

기존 UML의 클래스 다이어그램에 침입 감내 부분을 담당하는 클래스 혹은 객체에 대해서는 <<MW>>를 정의하여 표현하는데 이는 미들웨어의 약어로서 침입 감내 서비스를 담당하는 클래스나 객체들이 미들웨어 계층에 존재하기 때문이다. 시스템에서 비 기능적인 요구사항을 만족해야 하는 속성 혹은 오퍼레이션인 경우, 각 이름 앞에 NFR_ 가 추가되어 표기된다. 속성에 대한 비 기능적인 요구사항을 상세하게 기술하기 위해서 클래스 식별 작업과 동일한 방법인 Tagged values를 사용한다.

<그림 8>은 UML-IT를 적용하여 수신 관리 시스템에 대한 클래스 다이어그램의 일부분을 보여주고 있다.

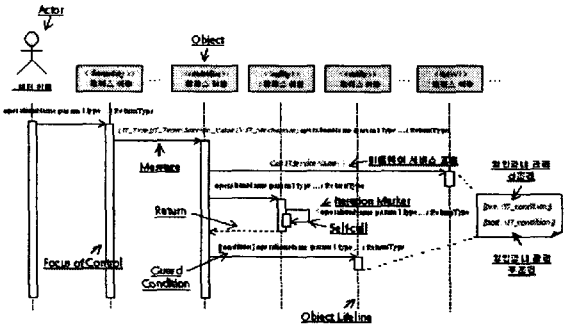


<그림 8> 수신 관리 클래스 다이어그램

4.4 UML-IT를 적용한 행위 모델 명세

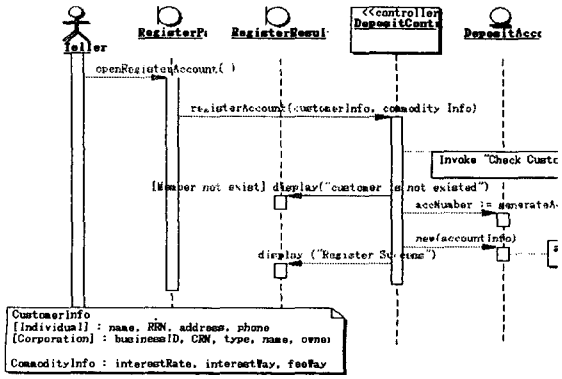
<그림 9>는 행위 모델링에 있어서 침입 감내 관련 서비스와 이를 제어하는 제어 객체 그리고 다른 객체들과의 메시지 전송에 있어서 UML-IT Profile을 적용한 것이다.

행위 모델링을 위해 상호 작용도를 이용하였는데 이 다이어그램에도 침입 감내와 관련된 비기능적인 요구사항들과 사전, 사후 조건들을 메시지에 함께 명세하고, 특히 침입 감내 서비스들을 담당하는 객체들을 <<MW>>로 표현하였고, 이들 객체들에 대한 메시지는 거의 침입 감내 관련 서비스 호출로 명세된다.



<그림 9> UML-IT를 적용한 행위 모델링

<그림 10>은 수신 관리 업무에서 계좌 등록 유스케이스에 대한 행위 모델링을 한 예제이다.



<그림10> '계좌 등록' 행위 모델링

5. 결론 및 향후 연구과제

지금까지 ITUA, HACQIT, SITAR, MAFTIA들에 대해 살펴보았다. 또한 각각의 아키텍처들에 대해서 각각의 아키텍처를 구성하는 요소나 메커니즘들을 분석하여 의존성 특성과 관련하여 각각의 아키텍처들이 어떻게 지원하고 있는지를 비교 평가하였다. 향후 연구과제로는 이러한 비교 평가를 토대로 침입 감내 기반 응용 소프트웨어 개발에 필요한 기능적, 비기능적 요구사항들을 도출하여 명세하고 이를 바탕으로 침입 감내 기반 소프트웨어를 위한 모델링 기법을 제시하는 것이다.

참고문헌

[1]Sames, D., et.al, "Developing a Heterogeneous Intrusion Tolerant CORBA System", *Proceedings of the 2002 International Conference on Dependable Systems & Networks, Washing, D.C., June 23-26, 2002.*

[2]Deswarte, Y., et.al, "Intrusion Tolerance in Distributed Systems", in IEEE Symp. on Research in Security and Privacy, Oakland, CA, USA, pp.110-121, 1991.

[3]Dutertre, B., Crettaz, V., Stavridou, V.: Intrusion-tolerant Enclaves. In: Proceedings of the IEEE International Symposium on Security and Privacy, 2002.

[4]Courtney, T., *et.al,* Providing Intrusion Tolerance with ITUA, *Proc. of the ICDSN 2002 Supplementary Vol.,* p.C-5-1, June 2002, IEEE CS.

[5]Cukier, M., et. al, "Intrusion Tolerance Approaches in ITUA", In Supplement of the 201 International Conference on Dependable Systems and Networks, pp. 64 ~ 65,G¨oteborg,Sweden, July 2001.

[6]Reynolds, J., et. al, "The Design andImplementation of an Intrusion Tolerant System", Proceedings of the International Conference on Dependable Systems and Networks (DSN'02), 2002.

[7]Just, J.E, and Reynolds, J.C., "HACQIT (Hierarchical Adaptive Control of QoS for Intrusion Tolerance)". In 17th Annual Computer Security Applications Conference, 2001.

[8]Wang, F. and Killian, C., "Design and Implementation of SITAR Architecture: A Status Report", *Proc, of the ICDSN 2002 Supplementary Vol.,* p.C-3-1, June 2002.

[9]Wang, F., Gong, F., Sargor, C., Goseva-Popstojanova, K., Trivedi, K., and Jou, F., "SITAR: A Scalable Intrusion Tolerance Architecture for Distributed Server," IEEE 2nd SMC Information Assurance Workshop, West Point, New York, 2001

[10]Neves, N. F. and Verissimo, P., The Middleware architecture of MAFTIA: A blueprint. DI/FCUL TR 99-6, Department of Computer Science, University of Lisboa, Sept. 2000.

[11]Adelsbach, A., et. al, "Conceptual Model and Architecture of MAFTIA. Project MAFTIA IST-1999-11583 deliverable D21. 2002. http://www.research.ec.org/maftia/deliverables/D21.pdf.

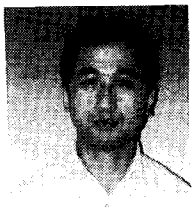
[12]Powell, D., et.al, "MAFTIA (Malicious- and Accidental-Fault Tolerance for Internet Applications), *Sup. of the 2001 International Conference on Dependable Systems and Networks (DSN2001)*, G¨oteborg (Sweden), 1-4 July 2001, IEEE, pp. D-32-D-35.

주 작 성 자 : 조 은 숙
논문투고일 : 2004. 01. 14
논문심사일 : 2004. 01. 19(1차), 2004. 01. 26(2차), 2004. 01. 27(3차), 2004. 02. 18(4차), 2004. 02. 23(5차), 2004. 03. 15(6차),
심사판정일 : 2004. 03. 16

● 저자소개 ●

조은숙(escho@dongduk.ac.kr)

1993. 2 동의대학교 자연과학대학 전산통계학과 졸업(이학사)
1996. 2 숭실대학교 공과대학 컴퓨터학과 졸업(공학석사)
2000. 2 숭실대학교 공과대학 컴퓨터학과 졸업(공학박사)
2002. 1 ~2003.10 한국전자통신연구원 초빙연구원
2000. 1 ~현재 동덕여자대학교 정보학부 강의전임교수
관심분야 소프트웨어 모델링, CBD 방법론, 소프트웨어 아키텍처, ,소프트웨어 품질 및 테스팅



이강신

1987. 2 한양대학교 수학과 졸업(이학사)
1989. 8 한양대학교 수학과 졸업(이학석사)
2002. 9 ~ 현재 고려대학교 정보보호학과 박사과정
1990. 7 ~ 1992. 6 데이콤 종합연구소 연구원
1992. 7 ~ 2000. 8 한국전산원 부장
2000. 9 ~ 현재 한국 정보보호 진흥원 팀장
관심분야 정보 보호, 보안정책, 침입 감내, 미들웨어