사례 연구 : 속성기반설계(ADD)를 적용한 하나로 연구로 방사선감시시스템(RMS) 개발

서용석[O*], 홍석봉[*], 김성진[**], 김종명[**], 김현수[***]
한국원자력연구소 계측제어•인간공학연구부[*], 삼창기업(주) 제어기술연구소[**], 충남대학교 컴퓨터공학과[***]
{yssuh[O], boong}@kaeri.re.kr[*], {ksj1330, kjm1332}@samchang.com[**], hskim401@cnu.ac.kr[***]

Case Study : A Development of Radiation Monitoring System for HANARO Nuclear Research Reactor by Applying the Attribute Driven Design

Suh Y.S.[O*], Hong S.B.[*], Kim S.J.[**], Kim J.M.[**], Kim H.S.[***]
Div. of I&C and HF, Korea Atomic Energy Research Inst.[*], Control Tech. Research Inst., Samchang Enterprise Co., Ltd.[**], Dept. of Computer Engineering, Chungnam National University[***]

요 약

본 논문은 한국원자력연구소 내에서 가동 중인 하나로 원자로의 방사선감시시스템 소프트웨어 개발에 있어서 속성기반설계를 적용한 사례를 소개한다. 본 논문에서 채택한 속성기반설계는 시스템의 기능요건 및 품질요건 도출, 이를 만족하기 위한 전술 설정, 설정된 전술에 근거하여 시스템 아키텍처 결정, 확정된 아키텍처를 구현 및 검증하는 과정으로 이루어진다. 하나로 방사선감시시스템의 개발요건으로부터 사용성, 가용성, 유지보수성, 호환성, 확장성과 같은 품질속성을 추출하였으며, 개발 전술로는 이중화된 서버에 다수의 클라이언트가 연결되는 클라이언트-서버 클러스터링 전술과 객체지향적 데이터 처리 및 디스플레이 설계 전술을 채택하였다. 단기간 내에 개발을 완수해야 하는 방사선감시시스템 개발에 속성기반설계를 적용함으로써 보다 효율적으로 과제를 성공시킬 수 있었다.

1. 서 론

한국원자력연구소 내에는 연구용 원자로인 하나로가 가동 중에 있으며 이 연구로에서 누출될 우려가 있는 방사선을 감시하고 통제하는 방사선감시시스템(RMS, Radiation Monitoring System)이 설치되어 있다. 이 RMS는 하나로 내•외부 방사선 누출여부를 현장에서 감시하는 30개의 현장감시기기시스템과 데이터를 종합 감시하는 5개의 원격감시컴퓨터시스템으로 구성되어 있다. 이 RMS는 1999년에 MS-SQL™과 Delphi™로 개발되었으며[1], 원시코드는 약 만줄 정도이고 죽은 코드가 산재되어 있었으며 개발 문서가 빈약하여 해석이 어려운 상태였다. 이것은 RMS를 확장하거나 유지보수 하는데 많은 어려움을 초래하고 있었기 때문에 이를 개선하기 위한 과제가 2005년 11월에 시작되었다. 이 때 주어진 요건으로는 개선될 RMS는 기존의 RMS보다 우수한 기능을 제공해야 한다는 것이다. 예를 들어 저장된 데이터의 편리한 검색기능, 자동 백업기능, 명료하고 정확한 정보전달이 가능한 유저인터페이스 등이다. 문제는 우수하다는 기능을 개발자가 판단하여 제공하여야 한다는 것인데 이러한 추상적인 의미의 요건들은 개발자에게 구현의 융통성이라는 장점이 주어지는 반면 얼마나 구현해야 사용자를 만족할 수 있겠는가에 대한 판단의 어려움을 안겨준다. 이 과제는 적은 예산으로 6개월이라는 단 시간 내에 완료해야 하는 제한사항이 주어졌기 때문에 시작단계서 기존 RMS를 역공학 하여 개선하는 방법과 새로 개발하는 것에 대한 결정이 필요하였는데 복잡한 원시코드의 상태로 인해 새로 개발하는 전략이 불가피하였다. 소프트웨어 개발에서 분석, 설계, 구현, 시험과정은 피할 수 없기 때문에 이 과정을 수행하는데 있어서 RMS 기능요건의 가변성을 고려하여 폭포수 개발모델보다는 반복적 및 점진적 개발모델을 채택하고 사용자에게 프로토타입을 보여줌으로써 유저인터페이스 편리성을 확인하는 전략을 채택하였다. 이러한 전략을 효율적으로 수행하기 위해 속성기반설계(ADD, Attribute Driven Design)[2]를 적용하였다. Bass가 제시한 ADD 개발과정 중에 RMS 개발에서 변형하여 채택한 대체(alternate)과정을 "-Ⓐ"로 아래와 같이 표시하였다.

① 아키텍처 드라이버 도출
② 아키텍처 드라이버를 만족하는 아키텍처 패턴 결정
②-Ⓐ 아키텍처 드라이버를 만족하는 전술(tactic) 결정
③ 패턴으로부터 구성요소를 인스턴스화
③-Ⓐ 아키텍처 구현
④ 인스턴스로부터 제품계열화가 가능한 공통인자 규명
④-Ⓐ 재사용 가능한 공통인자 규명
⑤ 아키텍처 드라이버의 만족성 검증
⑥ 필요시 (추가적인 분해가 필요하면 이를 위한) 정제과정 수행

위 ADD의 각 세부과정에 대해 분석, 설계, 구현, 시험업무를 통해 수행한 내용을 다음 절에서 설명한다.

2. ADD을 적용한 개발과정

2.1 아키텍처 드라이버 도출

분석업무를 통해 RMS 아키텍처 드라이버로서 아래와 같은 기능요건과 품질요건을 도출하여 소프트웨어 요건명세서에 명시하였다.

① 기능요건: 기존 RMS보다 우수한 기능 제공
② 사용성요건: 편리한 유저인터페이스 제공
③ 가용성요건: 시스템 동작불능 상태 최소화
④ 유지보수요건: 24시간 이내에 유지보수 완료
⑤ 호환성요건: 미래의 다른 시스템과 데이터 호환가능
⑥ 확장성요건: 현장감시기, 원격감시컴퓨터 추가가능

RMS의 기본 기능은 그림 1과 같으며 세부적으로는 많은 기능이 포함되어 있고 이러한 기능은 운전의 편리성을 감안하면 많은 변화가 예상된다. 따라서 기능요건과 사용성요건은 운전원과 많은 협의를 통해 확정할 수밖에 없다.

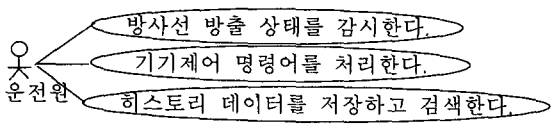


그림 1 RMS의 개략적 쓰임새도

2.2 전술 결정

분석과정에서 도출한 품질요건을 근거로 그에 적절한 전술을 채택하고 아키텍처를 결정하는 일이 설계과정에서 발생한다. 본 논문은 시스템, 하드웨어, 소프트웨어 아키텍처를 모두 포괄하는 RMS 아키텍처 완성에 중점을 둔다. 도메인 특성은 아키텍처 결정에 중요한 영향을 준다. RMS는 컴퓨터시스템들이 지역적으로 분산되어 있다. 이러한 특성은 RMS가 분산시스템 아키텍처로 결정하여야 하는 제한사항이 된다. 그림 2는 RMS의 분산시스템 아키텍처를 보여준다. 그림 2에서 서버의 이중화 구조는 RMS 아키텍처 드라이버를 만족하기 위해 채택한 전술의 결과이다.

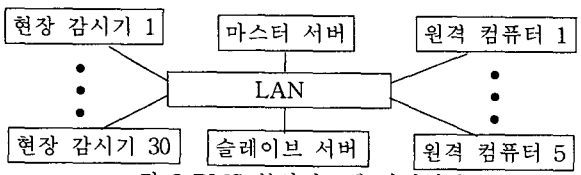
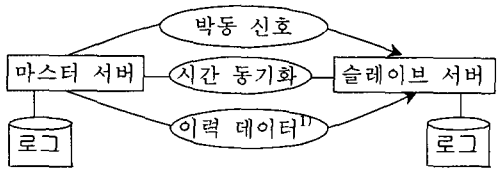


그림 2 RMS 분산시스템 아키텍처

전술은 아키텍처 드라이버를 만족하는 아키텍처를 형성하기 위해 필요한 요소를 결정하고 선택하는 일을 의미한다. 하나의 아키텍처를 결정하기 위해서는 하나 이상의 전술이 필요할 수 있으며 전술 상호간의 trade-off를 조정하는 과정에서 아키텍처가 완성된다. RMS 아키텍처 드라이버 중 사용성요건 만족을 위해서는 윈도우, 메뉴, 3D 이미지 등으로 구현한 유저인터페이스를 사용자에게 시연하는 전술을 채택하였다. 호환성요건 만족을 위해서는 OPC 국제산업표준을 채택하였다. OPC는 "OLE (Object Linking Embedding) for Process Control"의 약어이다. 이는 많은 나라의 산업체가 플랜트 제어 및 감시 시스템들 간의 인터페이스를 Microsoft사의 OLE 기반으로 국제표준화 한 것이다. 범용 데이터베이스와의 연결은 ODBC 또는 OLE DB 방식을 이용한다. 응용프로그래밍 개발이 필요하다면 DLL 또는 ActiveX 방식을 이용한다. 다음 절에서는 RMS 아키텍처 결정에 중요하게 작용한 가용성과 유지보수성 전술만을 선택하여 설명한다.

2.2.1 가용성 전술

가용성(availability)요건은 시스템의 연속운전을 장기화하고 최대한 데이터 무손실 보장을 요구한다. 이를 위한 전술로 그림 3과 같이 마스터 서버와 슬레이브 서버로 구성된 hot-standby 클러스터링 전술을 채택한다. 이 전술에서 클라이언트는 항상 마스터 서버와 통신한다. 마스터는 클라이언트의 요청을 수행하며 동시에 자신에게 발생한 변화를 슬레이브에게 즉시 전송하여 마스터와 슬레이브 간의 데이터 일치성이 실시간으로 유지된다. 슬레이브는 마스터로부터 받은 실시간 데이터를 박동신호(heartbeat)로 인지한다. 마스터로부터 박동신호가 오지 않는 경우 슬레이브 자신이 마스터가 되고 이를 클라이언트에 알린다. 고장 난 마스터가 복구되어 클러스터에 연결될 때에는 슬레이브로서 연결된다. 이때 마스터는 슬레이브와 데이터 일치를 위해 자신이 보관하고 있는 과거의 이력(history) 데이터 중 슬레이브가 갖고 있지 않은 부분만을 슬레이브에게 전송한다.



1) 고장복귀 후

그림 3 마스터/슬레이브 서버의 hot-standby 전술

2.2.2 유지보수성 전술

유지보수성(maintainability)요건은 소프트웨어의 수정, 확장, 제거 등의 용이성을 요구한다. 이를 위한 전술 결정은 RMS 데이터 처리 특성과 연관이 있다. 30개의 현장감시기로부터 취득되는 RMS 데이터는 상호 연관이 없다. 따라서 독립적인 처리가 가능하며 각각 독립적인 객체로 운전원에게 상황을 보여주면 된다. 이러한 특성에 적합한 전술로는 높은 응집도와 낮은 결합도를 적용한 객체지향적 설계 전술을 채택한다. RMS 클래스 타입의 종류로는 아날로그, 디지털, 알람, 로그, 익스프레션, 커스텀, 스캔 등이 있다. 그림 4는 RMS를 위해 설계된 클래스의 일부를 보여준다. UDR과 LCU는 RMS를 위해 설계된 고유 클래스이며 RE002, RE003, RE019는 고유의 감시기 이름을 갖는 객체이다. 이 객체들은 실시간 현장 데이터를 저장하고 있으며 클라이언트의 유저인터페이스 그래픽 화면에 해당 그래픽 객체와 연결되어 실시간 값 또는 명령어 등을 교환한다. 유저인터페이스는 객체지향적 그래픽 설계 전술을 채택한다. 그래픽 객체들은 서로 독립적으로 동작하며 이를 위해 데이터와 행위가 템플릿 형태로 정의되어 있다. 그래픽 객체가 필요로 하는 동적 데이터는 서버의 객체로부터 받는다. 그래픽 객체와 데이터 객체의 중계역학(intermediary)을 하는 인터페이스

를 두어 분리함으로써 유지보수가 용이하도록 한다.

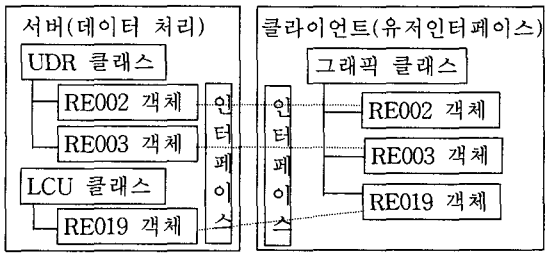


그림 4 RMS 클라이언트-서버의 인터페이스 전술

2.3 아키텍처 구현

분석과 설계의 결과, 구현 업무를 그림 5와 같이 명확히 구분할 수 있었다. 구현의 중심은 데이터베이스를 구축하는 것이다. 데이터베이스 구축에서는 클래스 설계와 각 객체의 활용을 정의한다. 데이터베이스 구축 업무는 마스터와 슬레이브 간의 데이터 일치와 외부 시스템과의 데이터 교환도 포함한다. 현장감시기와 데이터 교환을 위한 통신모듈은 독립적으로 구현될 수 있으며 데이터베이스와의 인터페이스만 일치시키면 된다. 응용프로그램은 복잡한 계산 알고리즘 수행을 위해 개발한다. 응용프로그램은 독립적으로 구현될 수 있으며 계산 결과를 사용하는 객체와 인터페이스만 일치시키면 된다. 유저인터페이스는 많은 그래픽 객체를 개발하는 업무이며 각 객체는 데이터베이스에 정의된 해당 객체와 인터페이스를 갖게 된다.

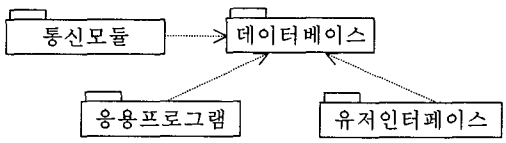


그림 5 RMS 소프트웨어 패키지도

RMS 개발 과제는 단 시간 내에 개발되어야 하는 제약사항 때문에 프로그래밍을 최소화하는 전략을 채택하였다. 이를 위해 SCADA(Supervisory Control and Data Acquisition) 시스템 구축 도구인 Adroit[TM1)]를 채택하였다. Adroit는 분산 실시간 시스템을 객체지향적으로 구현할 수 있도록 도와주는 도구이다. 이 도구를 사용함으로써 데이터베이스의 구축과 유저인터페이스 그래픽 화면을 효율적으로 구현할 수 있었다.

2.4 재사용 가능한 공통인자 규명

RMS 개발 과제에서 재사용이 가능한 공통인자는 이중화된 서버의 클라이언트-서버 아키텍처와 객체지향적 데이터처리 및 유저인터페이스 아키텍처이다. Adroit 적용은 가변인자이지만 아키텍처 자체는 공통인자이다. 통신모듈과 응용프로그램도 가변인자이다. 가변인자는 사용자 요구에 따라 변경이 필요한 부분을 의미한다. 본 과제를 통해 축적된 엔지니어링 산출물의 틀(framework)도 공통인자이다.

1) www.adroit.co.za, www.bnftech.com

2.5 아키텍처 드라이버의 만족성 검증

아키텍처 드라이버의 만족성은 시험으로 검증하였다. 기능 및 사용성에 대한 만족성은 프로토타입을 만들어 검증하였다. 소프트웨어 요건명세서는 기능요건을 추상적으로 명시하였기 때문에 알파 테스트를 위해 사용되었고, 사용자에게 프로토타입을 시연 및 동작하게 함으로써 사용성을 검증하였다. 가용성에 대한 만족성은, 마스터 서버와 슬레이브 서버의 실시간 데이터 일치성 시험과, 마스터 서버에 고장을 발생시켜 슬레이브 서버에 의한 운전 연속성 시험과, 고장이 복구된 서버가 클러스터에 연결되었을 때 마스터 서버의 데이터가 슬레이브 서버에 완전히 복구됨을 시험을 통해 검증하였다. 유지보수성에 대한 만족성은 임의의 고장 또는 변경 사례를 만들어 그 임무가 24시간 이내에 완료됨으로 검증하였다. 호환성에 대한 만족성은 이기종으로 개발한 외부 시스템과의 데이터 교환 시험과 MS-SQL™ DB와의 호환성 시험을 통해 검증하였다. 확장성에 대한 만족성은 새로운 화면 추가, 현장감시기 추가, 클라이언트 추가 등의 시험을 통해 검증하였다.

2.6 정제 과정 수행

정제 과정은 RMS 아키텍처를 분산 구조와 이중화 서버로 확정한 후 데이터베이스의 클래스와 객체의 설계, 유저인터페이스의 그래픽 객체의 설계에서 반복적으로 발생하였다.

3. 결론

본 논문은 소프트웨어 공학적 개발 과정을 준수하면서 단시간 내에 개발을 완료하여야 하는 RMS 개발과제에 ADD을 적용함으로써 효율적으로 개발을 완성한 사례를 소개하였다. 본 논문이 소프트웨어 아키텍처 설계를 상세히 보여주지 못한 것은 Adroit라는 SCADA 구축 도구를 사용하여 프로그래밍을 최소화했기 때문이다. 비록 도구를 사용할지라도 소프트웨어 공학적으로 ADD을 적용하여 개발 공정을 따른 사례로써 본 논문의 의의가 있다. 아직은 산업계에서는 ADD 적용사례가 많이 발표되고 있지 않은 것 같으나 소프트웨어 개발과정에서 목표를 수립한 후 각각의 목표를 달성하기 위한 전술을 수립하고 아키텍처를 정제하는 과정은 소프트웨어를 구현에만 집중하였을 때 발생할 수 있는 위험을 줄일 수 있는 효과가 있을 것이다. 본 과제를 수행하면서 RMS 시스템이 요구하는 요건을 만족하는 아키텍처를 얻을 수 있었으며 추후 한국원자력연구소 내 다른 시설에서의 RMS 구축 및 통합하는 과제에 재사용할 예정이다.

참고문헌

[1] KAERI/RR-2059/99, "Y2k 문제 해결을 위한 하나로 및 부속설비의 방사선감시계통 전산설비 개선", 한국원자력연구소, 2000.

[2] Len Bass, et al., "Software Architecture in Practice", 2nd Ed., Addison Wesley, 2003.