|  |
| --- |
| **포스코 조업시스템(MES) 프로젝트** https://blogimgs.pstatic.net/imgs/nblog/spc.gif [퍼온글(나만의 공간)](https://blog.naver.com/PostList.nhn?blogId=h_tiger&categoryNo=5&from=postList&parentCategoryNo=5) https://blogimgs.pstatic.net/imgs/nblog/spc.gif  2005. 10. 14. 9:39  복사<https://blog.naver.com/h_tiger/20018068997>  [번역하기](https://blog.naver.com/PostView.nhn?blogId=h_tiger&logNo=20018068997&redirect=Dlog&widgetTypeCall=true&directAccess=false) [전용뷰어 보기](https://blog.naver.com/PostView.nhn?blogId=h_tiger&logNo=20018068997&redirect=Dlog&widgetTypeCall=true&directAccess=false) |

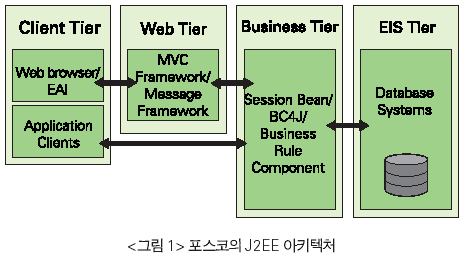
국내 최대 J2EE 신화 창조: 포스코 조업시스템(MES) 프로젝트

박정배 | 한국오라클

ERP 기반의 전사적 통합 패키지 도입, 기업 포탈의 성공적 구축으로 눈부신 경영시스템 혁신을 일구어온 포스코가 이제 조업시스템 재구축 프로젝트의 완성을 눈앞에 두고 있다.   
선도적인 IT 기술 도입이 미래 경쟁력의 바탕임을 확신하는 포스코는 새로 구축한 조업시스템을 통해 광양과 포항 두 개 제철소를 하나의 관리 체계하에서 표준화된 프로세스로 운영할 수 있게 되었으며, ERP 시스템과의 연계를 통해 경영시스템의 혁신은 물론, 생산시스템의 혁신도 이룰 수 있게 되었다.   
포스코 신화 창조의 현장으로 독자들을 안내한다.

레가시를 대체하는 새로운 시스템의 필요

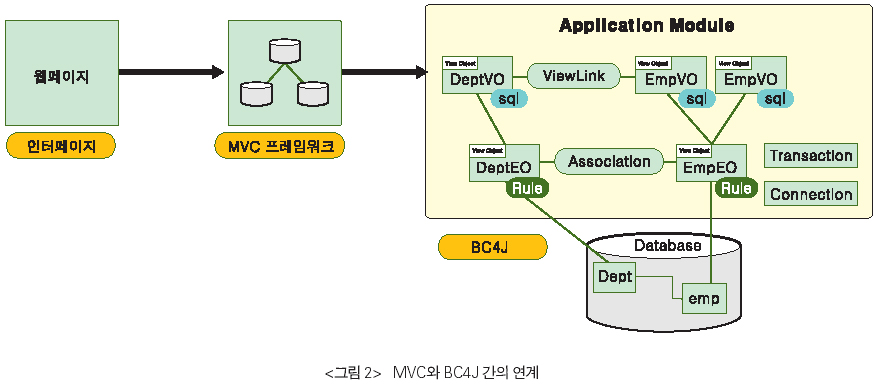
포스코의 기존 조업시스템(MES : Manufacturing Execution System)은 메인프레임으로 구축되어 있었는데 장비의 노후화와 전문인력의 부재 등 으로 유지보수 비용이 계속 증가하고 있었다. 이는 장비뿐 아니라 소프트 웨어도 마찬가지였는데, 광양과 포항 양 제철소의 애플리케이션이 상호 이 질적인 데다가 새로이 추가되는 시스템과도 잘 어울리지 못했다. 스테인리 스강 공장 시스템과 같은 새로운 시스템이 계속 추가되는 상황에서, 결국 하드웨어와 소프트웨어 양쪽 모두에서 시스템을 계속 유지는 것이 비경제 적이나 판단하게 되었다. 포스코는 사업 초기부터 시스템에 많은 투자를 해 온 회사로서, 지금이 바로 시스템 교체가 필요한 시기라고 판단하고, PI 1기의 성공을 발판 삼아 PI 2기 프로젝트의 일환으로 새로운 조업시스템의 구축 작업에 들어갔다.   
새로이 구축하는 조업시스템은 하드웨어와 소프트웨어 모두에 있어서 기존의 것을 뒤집는 매우 파격적인 것이었다. 하드웨어로는 메인프레임을 버리고 슈퍼컴퓨터급의 Unix 장비를 도입했고, 여기에 소프트웨어로는 오 라클의 애플리케이션 서버와 데이타베이스를 탑재하여 Java를 기반으로 하는 J2EE 엔터프라이즈 애플리케이션으로 시스템을 구축했다.   
이러한 파격적인 시스템 교체에 대한 우려도 적지 않았는데, 가장 큰 우려는 Java가 성능이 좋지 못해 조업시스템으로서는 적절치 못하다는 것 이었다. 그러나, 이것은 초창기 Java가 보여주었던 성능적 결함을 침소봉 대한 것으로, 최근의 Java는 J2EE에 기반한 견고하고 안정된 아키텍처 아 래 안정된 성능을 보여주고 있다.<그림1>



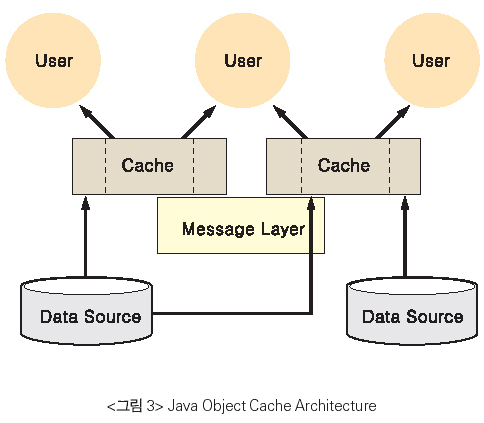
문제는 Java나 J2EE에 기반한 아키텍처가 아니라 어떻게 조업시스템 을 구현할 것인가 하는 방법이었다. 오라클은 조업시스템 개발에 사용할 프레임워크를 제공함으로써 성공적인 조업시스템 구축에 기여하였다.

강력한 애플리케이션 개발 프레임워크의 도입

조업시스템은 공장의 설비로부터 데이타를 받아 이를 처리하고 새로운 작업 을 지시하는 등의 일을 수행한다. 문제는 각 공장이 가지고 있는 작업의 다양 성에서 발생한다. 각 공장마다 설비가 다르고 공정이 다르고 작업내용이 다 르기 때문에 패키지로 되어 있는 소프트웨어로는 시스템의 요구조건을 만족 시키는 데 한계가 있기 마련이다. 결국 포스코는 조업시스템을 직접 구현하 여 구축하기로 결정했다.   
포스코가 선택한 방법은 CBD(Component Based Development) 를 기반으로 하여 주요 모듈을 컴포넌트로 구성하여 개발하는 것인데, 이 를 위해서는 강력한 개발 프레임워크의 뒷받침이 필수적이었다.   
포스코는 조업시스템 구축을 위해 엔터프라이즈 애플리케이션 개발에 사용할 수 있는 모든 기술에 대해서 검토해 봤다고 해도 과언이 아닐 정도 로 많은 투자를 했다. 초기에는‘J2EE 기반 애플리케이션’이라고 하면 금 방 떠오르는 EJB(Enterprise Java Beans) 기술을 사용해서 애플리케이션 을 구축하려고 많은 노력을 기울였다. 그러나, EJB는 데이타베이스와 연동 하여 많은 양의 데이타를 처리해야 하거나, 복잡한 비즈니스 로직을 수행 해야 하는 일이 많은 조업시스템에는 그리 적합한 기술이 아니었다. 포스 코의 조업시스템에는 보다 강력하고 사용하기 쉬운 애플리케이션 개발 프 레임워크가 더 적합했다. 여기에 사용된 것이 바로 오라클의 여러 강력한 애플리케이션 개발 프레임워크들이다.   
오라클은 데이타베이스 회사일 뿐만 아니라 비즈니스 애플리케이션 회사이기도 하다. 오라클의 비즈니스 애플리케이션은 기본적으로 Oracle Application Framework라는 프레임워크를 기반으로 제작되어 있는데, 이 프레임워크는 대량의 데이타 처리나 많은 사용자를 지원하는 데 있어 안정적인 기능을 제공한다. 특히 Oracle Application Framework가제공 하는 여러 프레임워크 기능 중 데이타베이스 연동에 사용되는 BC4J는 그 기능이나 성능, 사용 편이성 등에 있어서 타의 추종을 불허하는 최고의 솔 루션이라 할 수 있다. 포스코 조업시스템은 대량의 데이타 처리 작업과 비 즈니스 로직 구성 부분에서 바로 이 BC4J를 사용했다.   
이 뿐만 아니라 웹 애플리케이션 개발 프레임워크도 사용되었다. 포스 코 조업시스템의 사용자 인터페이스는 기본적으로 웹을 기반으로 하여 시 스템에 쉽게 접근해 운영할 수 있도록 되어 있다. 웹을 기반으로 하는 사용 자 인터페이스의 개발에는 오라클의 웹 애플리케이션 개발 프레임워크인 Oracle MVC Framework(일반적인 MVC 패턴의 프레임워크가 아니라 오라클의 제품 이름)가 사용되었다. Oracle MVC Framework는 모델링 툴과 잘 연계되어 있어 모델을 통해서 바로 사용자 화면의 흐름을 정의할 수 있는데, 사용자 화면이 많은 웹 기반 애플리케이션을 개발하는 데 아주 유용하다. <그림2>



또한 ERP 기술도 사용되었다. 포스코의 조업시스템은 실시간 데이타 를 바로 처리해야 하는 것을 기본으로 하고 있지만, 배치(batch)로 처리해 야 하는 일도 많이 있다. 이러한 배치 작업의 처리는 Oracle E-business Suite의 Concurrent Manager를 이용했으며, 이 덕분에 안정적인 배치 관 리와 수행이 가능해졌다. Oracle E-business Suite의 메인 컴포넌트로 구 성되어 있는 Oracle Concurrent Manager는 자체적인 작업수행 엔진과 스케줄링 엔진, HA 기능, 모니터링 기능을 가지고 있는, 자체적으로 완전 한 하나의 애플리케이션이라 할 수 있다.   
조업의 자동 수행과 주기적 수행을 요구하는 많은 애플리케이션의 수 행을 위해 프로젝트 초기에 오라클은 Oracle Concurrent Manager의사용 을 권고했고, 경제성과 안정성을 고려하여 조업시스템의 일부분으로 사용 하기로 결정했다. 대규모의 ERP를 구동하기 위한 배치 프로세싱 시스템으 로서의 Concurrent Manager 사용은 경제성과 안정성을 조업시스템에 가 져다주게 되었다. EAI와 스케줄링에 의해서 처리되는 Oracle Concurrent Manager의 처리량은 전체TC의 60% 이상으로 예상되고 있다.   
조업시스템은 성능에 대한 요구사항이 매우 엄격해서 그 성능을 달성 하기 위해 아키텍처 수립 단계에서부터 많은 고려가 있었다. 포스코의 조 업시스템은 시스템에서 사용하는 주요 기준 데이타나 룰 데이타, 메타데이 타 등을 데이타베이스에 저장해 두고 필요할 때마다 조회해서 사용하도록 <그림 2> MVC와 BC4J 간의 연계 했는데, 이 구조는 데이타베이스로의 라운드트립이 많아져 필연적으로 시 스템이 좋은 성능을 내기가 힘든 약점이 있다.   
이를 해결하기 위해 JOC(Java Object Cache)라는 객체 캐쉬 메커니즘을 사용했다. JOC는 객체를 캐쉬에 저장하여 애플리케이션이 필요로 하 는 데이타를 보다 신속하게 사용할 수 있도록 하는 기능을 제공한다. 포스 코의 조업시스템은 기준 데이타와 룰 데이타, 메타데이타 등을 JOC의 캐 쉬에 저장하여 사용하도록 함으로써 보다 빠른 성능을 낼 수 있도록 구성 되어 있다. <그림3>



앞에서 말한 네 가지 프레임워크는 오라클 애플리케이션의 토대가 되 는 것들로서, 특히 Java를 기반으로 한 애플리케이션을 개발하는 데 아주 유용한 것들이다. 또한, 이 프레임워크들은 여러 비즈니스 영역 전반에 걸 쳐 다양하게 사용될 수 있는 것들이다. 그러나 포스코의 조업시스템 프로 젝트에서는 이들 프레임워크를 그냥 사용하는 데 그치지 않고, 여기서 한 걸음 더 나아가, 이들 프레임워크 위에서 운영될 수 있는 컴포넌트를 개발 하여 컴포넌트의 재사용을 통해 조업시스템을 구축하는 방식으로 프레임 워크를 확장했다.

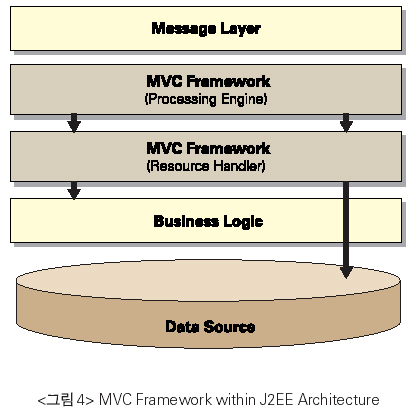
조업시스템의 기둥, 리유즈 컴포넌트의 개발

오라클의 프레임워크는 그 자체만으로도 애플리케이션을 개발하는 기반구 조로 매우 훌륭했지만 포스코의 조업시스템은 그 규모가 크고 개발에 참여 하는 인원이 많았기 때문에 보다 쉽게 사용할 수 있는 방법이 필요했다. 포 스코 조업시스템 프로젝트에서 사용한 방법은 공통의 기능을 수행하는 컴 포넌트를 개발하고, 각 리유저블(Reusable) 컴포넌트의 세부 기능을 메타 데이타로 정의할 수 있도록 해서 모델링 만으로도 애플리케이션 개발이 가 능하게 하는 방법이었다. 리유저블 컴포넌트를 기반으로 개발을 하는 방법 인데, 이는 애플리케이션을 더욱 쉽게 개발할 수 있는 밑바탕이 되었다.   
좀더 자세히 설명하면, 데이타 조회, 데이타 저장, 값의 비교를 통한 분기 등과 같이 일반적이고 공통적으로 자주 발생하는 작업을 수행하는 컴포넌트를 구현하고, 각 컴포넌트가 동작할 때 실제로 사용하는 데이타, 즉 어 떠한 테이블을 조회할지, 어떠한 값을 저장할지, 어떠한 값을 비교하여 어 디로 분기할지와 같이, 실제 기능을 결정하는 데이타를 메타데이타로 뽑아 내어 관리함으로써 리유저블 컴포넌트의 배치와 메타데이타 모델링만으로 도 애플리케이션 개발을 가능하게 하는 방법이다.   
포스코의 조업시스템 개발에 사용한 Java는 기존의 조업시스템에 익 숙한 개발자들이 쉽게 익혀서 사용할 수 있는 것이 아니었고 또한 개발에 참여하는 인원이 많았기 때문에, 이러한 리유저블 컴포넌트의 개발은 시스 템 구축에 꼭 필요한 과정이었다.   
오라클 컨설팅팀은 포스데이타와 함께 리유저블 컴포넌트를 개발했는 데, 오라클 컨설팅팀이 각 리유저블 컴포넌트에 대한 설계와 개발에 대한 방법론과 아이디어를 제시하고 포스데이타가 구현을 맡았다.   
포스코 조업시스템을 구성하는 리유저블 컴포넌트는 다음의 네 가지 로 나뉜다.

● **애플리케이션 로직   
● 비즈니스 로직   
● 사용자 인터페이스   
● 비기능(Non-functional)**

포스코 조업시스템의 핵심 - 애플리케이션 로직 컴포넌트

애플리케이션 로직 컴포넌트는 조업시스템의 업무 흐름에 대한 처리를 담 당한다. 애플리케이션 로직 컴포넌트는 포스코 조업시스템의 중추를 이루 는 컴포넌트로서, 크게 사용자 인터페이스(UI)를 처리하는 부분과 그 외 의 부분을 처리하는 Non-UI로 나뉜다. UI 컴포넌트는 웹을 기반으로 하는 사용자 인터페이스와 연관된 부분을 다루는 것으로, Oracle MVC Framework 위에서 동작한다. Oracle MVC Framework의 기본 구조는 Struts를 비롯한 여타 Model-2 기반의 웹 애플리케이션 개발 프레임워크 와 크게 다르지 않다. UI 컴포넌트는 Oracle MVC Framework 위에서 조 업시스템이 일반적으로 수행하는 데이타 조회, 데이타 저장, 전문 작성, 전 문 전송 등과 같은 작업을 처리한다. 포스코의 조업시스템은 자동화가 되 어 있어 사용자가 직접 개입해서 처리해야 하는 작업은 별로 없지만, 유사 시 백업시스템으로서 UI를 통한 제어가 가능하도록 시스템을 구성하였다 <그림4>.



UI 컴포넌트가 사용자 인터페이스와 관련된 일을 하는 데 반해, Non- UI 컴포넌트는 조업설비가 보낸 데이타를 EAI를 통해 전달 받아 처리하는 데 사용된다. 조업설비가 보내는 데이타는 표준화된 전문의 형태로 나타나 는데, 이 전문의 표준화는 조업시스템 구축의 매우 중요한 작업 중 하나였 다. 포스코 조업시스템 프로젝트팀은 Non-UI 컴포넌트 부분의 구현을 위해 프레임워크를 직접 만들어 사용했다. ‘메시지 프레임워크’라 명명된 이 프레임워크는 그 개념에 있어 Oracle MVC Framework와 크게 다르지 않 고, 다만 많은 수의 전문을 처리해야 하는 요구사항을 만족시키기 위해 가 볍게 설계된 것이 특징이다. 이 프레임워크 위에 UI 컴포넌트와 마찬가지 로 데이타 조회, 데이타 저장, 전문 작성, 전문 전송 등의 작업을 처리하는 컴포넌트를 만들었다.   
애플리케이션 로직 컴포넌트는 포스코 조업시스템의 중추로서 매우 중요한 역할을 수행한다. 초당 수백 건에 달하는 사용자 명령과 전문은 모 두 애플리케이션 로직 컴포넌트에 의해 처리되며 애플리케이션 로직 컴포 넌트를 중심으로 다른 리유저블 컴포넌트가 사용된다. 포스코의 조업시스 템은 애플리케이션 로직 컴포넌트를 바탕으로 비즈니스 액티비티 모델링 (BAM)과 메타데이타 세팅으로 그 대부분의 기능을 구현해냈다. 이것은 BC4J와 Oracle MVC Framework라는 강력한 프레임워크가 있었기에 가 능한 일이라고 할 수 있다.

비즈니스 룰 처리를 위한 비즈니스 로직 컴포넌트

비즈니스 로직 컴포넌트는 포스코 조업시스템의 비즈니스 룰이 담겨 있는 비 즈니스 룰 컴포넌트와 애플리케이션 로직 레벨에서 데이타베이스에 접근해 서 데이타를 조작하는 데 인터페이스 역할을 하는 비즈니스 로직 인터페이스 컴포넌트로 나뉜다. 이 두 컴포넌트는 모두 BC4J의 기본 기능을 확장하여 구 현했다.   
비즈니스 룰 컴포넌트는 포스코의 조업시스템 각 공정마다 다양하게 존재하는 업무 규칙을 BC4J의‘엔티티 객체’라고 하는 하나의 체계를 통 해 정립한 것이다. 조업시스템은 비즈니스 룰 컴포넌트를 통해 업무 규칙 을 일관되고 효과적인 방법으로 관리할 수 있게 되었다. 비즈니스 룰 컴포 넌트를 통해 수집된 업무 규칙은 지속적으로 관리되고 개선됨으로써 시스 템이 시간이 갈수록 안정화되는 기반이 된다.   
비즈니스 로직 인터페이스 컴포넌트는 애플리케이션이 데이타베이스 에 접근하는 부분을 정의하는데, 포스코 조업시스템은 비즈니스 로직 인터 페이스를 통해서만 데이타베이스에 접근하여 데이타를 검색하거나 조작 할 수 있다. 포스코 조업시스템은 비즈니스 로직 인터페이스로 데이타 조 회와 수정에 대한 표준을 정의하고 애플리케이션 로직 컴포넌트가 메타데 이타에 정의된 값에 따라 비즈니스 로직 인터페이스를 사용하도록 되어 있 다. 포스코 조업시스템은 많은 양의 데이타를 다루기 때문에 데이타베이스 를 많이 사용하게 되는데, 데이타베이스에 접근하는 부분을 비즈니스 로직 인터페이스를 통해 제어함으로써 애플리케이션을 보다 안정적으로 구축 할 수 있게 되었다.

태그라이브러리 컴포넌트들로 꾸며진 사용자 인터페이스

포스코 조업시스템의 모든 분야는 개발방법론을 바탕으로 진행되어 대부분 의 개발 라이프사이클을 충실하게 준수하였으며, 사용자 인터페이스도 마찬 가지였다. 개발 초기에 사용자 요구사항을 수용하여, 사용자가 원하는 화면 들에 대한 템플릿화 작업이 선행되었다. 재사용이 필요한 사용자 화면의 요 소들, 특히 JSP 화면에서 데이타를 렌더링하는 영역에 해당 컴포넌트를 중심 으로 JSP 태그라이브러리화하여 개발자들의 생산성 향상과 사용자 인터페이 스의 통일화를 추구할 수 있도록 한 것이다. 이 작업은 프로젝트에 투입된 개 발자가 수백 명인 대규모 프로젝트에서 당연히 거쳐야 하는 과정이었으며, 프로젝트 성공을 위한 하나의 요건이 될 수 있다.

시스템의 보조자인 비기능 컴포넌트(Non-functional)

포스코 조업시스템의 핵심적인 부분은 아니지만, 전체 시스템에 공통으로 사용되는 유틸리티성 기능들을 구현하는 것도 중요한 일이었다. 포스코 조 업시스템에서는 이러한 주요 항목들을 비기능 컴포넌트로 정의하고, 재사 용 가능한 컴포넌트로 개발했다. 여기에 해당하는 것은 메시징/로깅/에러 핸들링, 보안 등이었다.   
메시징/로깅/에러 핸들링 등의 비기능 컴포넌트는 개발 단계에서 디버깅 및 모니터링을 손쉽게 할 수 있도록 도와주었으며, 소스 코드의 에러를 신속히 조치 가능하도록 해주었다. 또한 향후 운영시에도 운영자들이 비즈니스 및 데이타의 정합성 체크를 신속히 처리하는 데 도움이 될 수 있도록 설계되었다.   
조업시스템에서 중요한 부분 중의 하나는 사용자 관리, 권한 관리 등 조업시스템 내의 보안과 관련한 기능이다. 조업시스템에서 개발된 화면을 사용하는 주사용자는 포항, 광양 양 제철소의 작업자들이며, 15,000명 이 상의 사용자들이 모든 화면을 사용 가능하도록 하는 것은 결코 바람직하지 않기 때문이다. 각 작업자는 자기의 주어진 업무 및 권한에 따라, 특정 화면 들에만 접근 가능하여야 하며, 화면 내에서도 단순 조회, 입력/수정 권한의 여부까지 세분화된 권한으로 분류되어 있어야 한다. 조업시스템의 사용자 및 권한관리의 기본 모델은 Oracle E-Business Suite의 사용자 관리체계 (User Responsbility)이며, 사용자는 개인별로 접속할 수 있는 메뉴 체계 를 갖도록 구현했다(Responsibility , Menu , Function 등으로 분류). 권 한관리에 해당하는 부분은 조업시스템이 자체적으로 가지고 있으며, 이미 구축되어 있는 POSCO Enterprise Portal 시스템에 연계되어 있다.

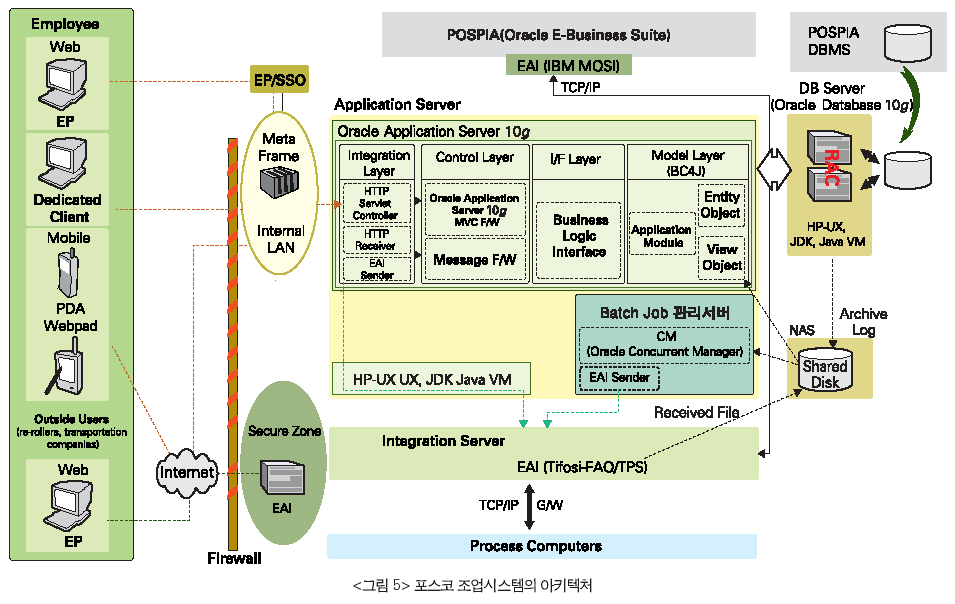
**포스코 조업시스템의 진기록**  
• 프로그램 수 : 6,500여종  
• 프로그램 라인 : 200만줄(300페이지 책으로360권분량)  
• 하루 데이타 처리량 : 2기가바이트 (36면일간지2년6개월 분량)  
• 프로젝트의 투입 공수 : 15,999M/M

시스템의 보이지 않는 근간이 된 마스터데이타/메타데이타

연간 2,800만 톤의 조강생산 능력이 있는 포스코 조업시스템의 운영을 위해서는 표준화 및 통합화가 기본 사상이다. 이미 구축된 Oracle EBusiness Suite 시스템(POSPIA)으로부터 작업지시를 받아 포항, 광양 양소의 수십 개 공장에서 생산라인에 명령을 하달하는 수많은 프로세스 컴퓨터들, 그리고 생산실적 처리 업무들, 기타 업무에 필요한 품질판정, 출하, 야드 등의 물류처리 정보 등 포스코 조업시스템이 쏟아내는 정보의 양은 엄청나다. 이 엄청난 정보를 정확히 그리고 조강생산에 지장이 없을 정도 의 시간 내에 처리하기 위해서 마스터 코드, 업무판단기준, 계산식, 인터페 이스 전문 에러체크 기준 등을 표준화하고 통합 관리하는 체계를 구축했 다. 이것을 포스코에서는‘마스터데이타’라고 지칭하였다. 또한 조업시스 템을 위해 개발된 리유저블 컴포넌트는 개발 생산성 향상과 향후 운영성 향상을 위해 업무처리에 필요한 데이타를 메타데이타로 관리하도록 했다.   
이러한 마스터데이타와 메타데이타는 성능향상을 위해 JOC(Java Object Cache) 메커니즘을 사용하도록 했으며, 소리없이 포스코의 조업 시스템을 움직이는 근간으로서의 역할을 수행하고 있다.

조업시스템의 아키텍처

포스코 조업시스템의 아키텍처는 <그림 5>와 같이 구성되어 있다.



조업시스템의 기반이 되는 미들티어

오라클의 미들티어 서버인 Oracle Application Server 10g는 조업시스템 애플리케이션과 함께 미들티어의 고가용성(HA)을 제공한다. HTTP 프로토 콜이 근간을 이루는 웹에서 고가용성은 애플리케이션 서버 단독으로 이루어 질 수 없는 것이며, 애플리케이션에서 세션과 관련된 다양한 기술을 차용함 으로써 완성된다. Oracle Application Server 10g는 클러스터 기능과 OC4J의 세션 공유 기술인 세션 복제(session replication), 그리고 JOC (Java Object Cache) 라이브러리를 통한 JVM간 객체 동기화 기능을 제공 한다. 이는 포스코 조업시스템의 까다로운 고가용성 조건을 만족시킬 수 있 는 인프라를 완벽하게 제공하는 것이다.

클러스터

고가용성의 기본은‘다중화’라할수있다. ‘다중화’라는 바탕 위에 세션 공유 나 객체 공유, 로드 밸런싱 알고리즘 등의 기능이 그려지게 된다. 엔터프라이 즈 환경에서 다중화는 크게 두 가지 의미를 가지는데, 하나는 자원의 동기화이 고 다른 하나는 다중화 개체들 사이의 페일오버를 통한 무정지 서비스이다.   
Oracle Application Server 10g에서는 Oracle9i Application Server 에서 제공하던 데이타베이스 기반 클러스터 방식 외에 파일 기반 클러스터 방식을 제공한다. 파일 기반 클러스터는 소규모의 파일 데이타베이스를 기 반으로 하여 WAS 자원이 관리되는 방식으로서, 포스코 조업시스템은 이 방식을 차용하였다.   
이 방식은 자원 동기화를 목적으로 데이타베이스를 관리해야 했던 부 담을 덜어주고 하드웨어를 절약하게 해주며 오퍼레이션 시간을 단축시켜 많은 호응을 얻고 있는 Oracle Application Server 10g의 신기능이다.   
클러스터 개체들간 로드 밸런싱에는 다양한 알고리즘이 제공되고 있 는데, 조업시스템에서는 특히 Local Affinity 설정(OHS에서 OC4J로 라우 팅할 때 로컬 OC4J에 우선순위가 부여되는 설정)을 사용하여 사용자 화면 이 아닌 공장 단말에서 올라오는 트랜잭션 처리에 적합하도록 되어 있다. 이는 특히 조업의 특성상 아주 유용한 알고리즘으로 성능에 많은 기여를 하고 있다. 사용자 화면에 대해서는 서블릿 세션이 Local Affinity보다 우 선되기 때문에 달리 고려할 점이 없다.

OC4J간 세션 복제

이 기능은 JVM간 세션을 공유하는 기능으로, OC4J에서 설정해 준다.   
<그림 5>에서 알 수 있듯, 조업시스템의 애플리케이션은 Oracle Application Server 10g MVC Framework를 컨트롤러 부분으로 사용하 고 있는데, 이 프레임워크는 서블릿 세션을 기반으로 동작한다. 프레임워 크 내부에서 사용하는 세션의 규모가 세션 복제를 하기에는 무리가 있다고 판단되어 조업시스템에서는 이것을 사용하지 않는다. 대신 조업시스템 애 플리케이션은 쿠키와 HTML 폼 파라미터를 이용하여 세션을 재구성 할 수 있도록 설계되었기 때문에 서비스 관점에서는 완벽한 페일오버가 구현되 고 있다.

Java Object Cache

조업시스템 애플리케이션은 세션 복제를 사용하는 대신 메타데이타와 마 스터데이타 간 동기화가 필요한 부분은 JOC(Java Object Cache) 라이브 러리를 사용하여 JVM간 객체 동기화를 실현한다. JOC는 데이타베이스 트 래픽을 현격하게 줄여주는 것과 더불어 JVM 간 객체 동기화 기능을 수행 함으로써 하드웨어를 충족시키는 애플리케이션 설계를 더욱 편리하게 하 도록 도움을 준다.   
이와 같이 조업시스템의 미들티어는 Oracle Application Server 10g 에서 제공하는 인프라적 기능과 웹 본연의 기술들을 접목하여 조업시스템 의 가장 중요한 비즈니스 요구사항을 한치의 틀림도 없이 제공한다.   
업무적으로 무정지성과 무정지 상태에서의 애플리케이션의 배포를 가 능케 하고자 하는 POSCO 조업시스템의 미들티어에 대한 선택은 현재 가 동중인 시스템에서 그 기능을 증명하고 있다.

포스코 조업시스템의 의의와 기대효과

포스코는 메인프레임에 기반한 기존의 조업시스템을 애플리케이션 서버 와 J2EE를 기반으로 하는 새로운 엔터프라이즈 애플리케이션 시스템으로 재구축해 냄으로써 메인프레임을 대체하는 새로운 시스템의 방향을 제시 했다. 그 동안 많은 J2EE 프로젝트가 있었지만, 포스코의 조업시스템 구축 프로젝트처럼 큰 규모의 프로젝트는 없었다. 그것도 생산설비를 제어하고 관리하는 조업시스템을 구축하는 것이기에 그 의의가 더욱 크다고 하겠다.   
통상적으로 J2EE 시스템을 이야기할 때 대부분의 경우에는 기간계 업 무의 중심이 아닌 서비스 중심의 시스템을 그 대상 영역으로 설정하는 경 우가 많다. 초기에 포스코가 J2EE를 생산계의 기반 기술로 선정했을 때 기 간계, 즉 생산 시스템을 J2EE로 한 경우는 거의 없다는 것이 문제로 대두되 었다. 하지만 새로운 패러다임을 도입하고자 하는 의식과 구현이 가능하다 는 자신감을 가지고 진행해온 3년에 걸친 프로젝트는 이제 1차 Iteration의 포항/광양 가동에 이은 2차 광양 본가동이 11월 23일에 시작됨으로 결실 을 맺어가고 있다. 이제 포항의 2차 본가동만을 남겨 놓은 상태이다.   
이를 통해 생산설비의 확장이나 새로운 업무의 도입에 기존 기술 요소 를 사용함으로써 경제적인 이득은 물론 시간과 인원의 절감 효과도 기대하 고 있다. 경제적 의미뿐만 아니라 상징적인 의미에서도 포스코 조업시스템 은 탁월한 가치를 가진다. 현재까지 알려진 생산계 시스템으로는 최대 규 모 J2EE로서 단순히 프로젝트에 참여한 인원이나 시간만이 아니라 단일 시스템이 한번에 개발되어 커버하고 있는 업무 영역으로도 최대 규모를 자랑한다.   
포스코의 조업시스템 구축은 생산설비시스템부터 경영정보시스템 전반을 아우르는 진정한 e-business의 완성으로서, J2EE가 과연 거대 조업시스템에도 적용이 가능한 것인가와 전사 시스템에 대한 해답으로서 기억될 것이다.

[**제공 : DB포탈사이트 DBguide.net**](http://www.dbguide.net/)

출처명 : 한국오라클

**[출처]** [포스코 조업시스템(MES) 프로젝트](https://blog.naver.com/h_tiger/20018068997)|**작성자** [행복 범](https://blog.naver.com/h_tiger)