ProFrame
개발 안내서

ProFrame C v5.0 Fix#1

안내서 정보

안내서 제목: ProFrame 개발 안내서

발행일: 2014-05-15

소프트웨어 버전: ProFrame C v5.0 Fix#1

안내서 버전: v2.1.2

내용 목차

그림 목차

안내서에 대하여

안내서의 대상

본 안내서는 Tmax ProFrame®(C 기반의 Tmax ProFrame 이하 ProFrame)을 이용하여 서비스 또는 업무 모듈을 개발해야 하는 개발자에게 ProFrame를 개발측면에서 개념적으로 이해하는 데 도움이 되도록 작성되었다.

안내서의 전제 조건

본 안내서는 TmaxSoft의 ProFrame를 이용한 개발에 관한 설명과 본 솔루션을 적용하는데 필요한 기능을 개관하는 지침서이다. 본 안내서의 내용을 원활히 이해하기 위해서는 다음과 같은 사항을 미리 알고 있어야 한다.

- UNIX 계열(LINUX 포함)
- Windows™

안내서의 제한 조건

본 안내서는 ProFrame를 실무에 적용하거나 운용하는데 필요한 모든 사항을 포함하고 있지 않다. 따라서 ProFrame 설치, 환경설정 등 운용 및 관리에 대해서는 각 제품 안내서를 참고하기 바란다.

안내서 구성

ProFrame 개발 안내서는 총 6개의 장으로 구성되어 있다.

각 장의 주요 내용은 다음과 같다.

- 제1장: 개요

  ProFrame 개발 측면의 기본적인 개념에 대해서 간략히 설명한다.

- 제2장: 개발 기술

  ProFrame에서 사용되는 개발과 관련된 기술에 대해 설명한다.

- 제3장: 개발 구성요소

  ProFrame 개발에 필요한 구성요소에 대해 설명한다.

- 제4장: 개발 관련 도구

  ProFrame 개발에 필요한 도구에 대해 기술한다.

- 제5장: 개발 절차

  서비스 및 업무 모듈을 개발하는 절차에 대해 기술한다.

- 제6장: 안내서 구성

  ProFrame 개발과 관련된 안내서에 대한 내용을 간략히 소개한다.

안내서 규약

표기	의미
<<AaBbCc123>>	프로그램 소스 코드의 파일명
<Ctrl>+C	Ctrl과 C를 동시에 누름
[Button]	GUI의 버튼 또는 메뉴 이름
진하게	강조
" "(따옴표)	다른 관련 안내서 또는 안내서 내의 다른 장 및 절 언급
'입력항목'	화면 UI에서 입력 항목에 대한 설명
하이퍼링크	메일계정, 웹 사이트
>	메뉴의 진행 순서
+----	하위 디렉터리 또는 파일 있음
|----	하위 디렉터리 또는 파일 없음
참고	참고 또는 주의사항
주의	주의할 사항
[그림 1.1]	그림 이름
[표 1.1]	표 이름
`AaBbCc123`	Java 코드, XML 문서
`[ command argument ]`	옵션 파라미터
< xyz >	'<'와 '>' 사이의 내용이 실제 값으로 변경됨
|	선택 사항. 예) A|B: A나 B 중 하나
…	파라미터 등이 반복되어서 나옴

시스템 사용 환경

	요구 사항
Platform	IBM AIX 5.x
	HP-UX 11.xx
	Solaris 9 (SunOS 5.9)
	Linux kernel 2.4 이상
Hardware	최소 120MB 하드디스크 공간
	256MB 이상 메모리 공간
	1GB 이상 하드디스크와 512MB 이상 메모리 공간 권장
Database	Oracle 9i 또는 10g
	Tibero 3.0 sp2 이상
TP-Monitor	Tmax 4.0 이상
	Tuxedo 6.5 이상

관련 안내서

안내서	설명
ProFrame 시작하기 안내서	ProFrame 시스템의 아키텍처와 구성요소 등 실제로 시스템을 사용하기 전에 개념적인 이해를 돕기 위한 안내서이다.
ProFrame 설치 안내서	ProFrame 설치를 위한 시스템 요구사항 및 설치/제거 방법에 대한 안내서이다.
ProFrame 스튜디오 안내서	GUI 기반에서 각종 리소스 모듈을 생성하기 위해 스튜디오를 사용하는 방법에 대한 안내서이다.
ProFrame ProMapper 개발안내서	GUI 기반에서 ProMapper 모듈을 생성하는 방법에 대한 안내서이다.
ProFrame FileIO 개발 안내서	GUI 기반에서 FileIO 모듈을 생성하는 방법에 대한 안내서이다.
ProFrame DBIO 개발 안내서	GUI 기반에서 DBIO 모듈을 생성하는 방법에 대한 안내서이다.
ProFrame EMB 개발 안내서	GUI 기반에서 EMB 모듈을 생성하는 방법에 대한 안내서이다.
ProFrame 단위 테스트 안내서	ProFrame 시스템을 이용하여 개발이 완료된 서비스 또는 업무 모듈을 단위 테스트 하는 방법에 대한 안내서이다.
ProFrame 온라인 프로그래밍 안내서	ProFrame 시스템을 이용하여 온라인 프로그램을 작성하는 방법에 대한 안내서이다.
ProFrame 배치 프로그래밍 안내서	ProFrame 시스템을 이용하여 배치 프로그램을 작성하는 방법에 대한 안내서이다.
ProFrame 관리자 안내서	ProFrame 시스템을 관리하는 방법에 대한 안내서이다.
ProFrame 유틸리티 안내서	ProFrame이 제공하는 유틸리티에 관한 안내서이다.
ProFrame TCache 안내서	ProFrame이 제공하는 TCache에 관한 안내서이다.

연락처

Korea

TmaxSoft Co., Ltd
5, Hwangsaeul-ro 329beon-gil, Bundang-gu,
Seongnam-si, Gyeonggi-do, 463-824
South Korea
Tel: +82-31-8018-1000
Fax: +82-31-8018-1115
Email: `info@tmax.co.kr`
Web (Korean): http://www.tmax.co.kr
기술지원: http://technet.tmaxsoft.com

USA

TmaxSoft, Inc.
560 Sylvan Avenue Englewood Cliffs, NJ 07632
U.S.A
Tel: +1-201-567-8266
Fax: +1-201-567-7339
Email: `info@tmaxsoft.com`
Web (English): http://www.tmaxsoft.com

Japan

TmaxSoft Japan Co., Ltd.
5F Sanko Bldg, 3-12-16 Mita, Minato-Ku, Tokyo, 108-0073
Japan
Tel: +81-3-5765-2550
Fax: +81-3-5765-2567
Email: `info@tmaxsoft.co.jp`
Web (Japanese): http://www.tmaxsoft.co.jp

China

TmaxSoft China Co., Ltd.
Beijing Silver Tower, RM 1508, 2# North Rd Dong San Huan,
Chaoyang District, Beijing, China, 100027
China
Tel: +86-10-6410-6145~8
Fax: +86-10-6410-6144
Email: `info.cn@tmaxsoft.com`
Web (Chinese): http://www.tmaxsoft.com.cn

제1장 개요

본 장에서는 ProFrame를 개발 측면에서 이해하기 위해 기본적인 개념에 대해 설명한다.

1.1. 개발 사상

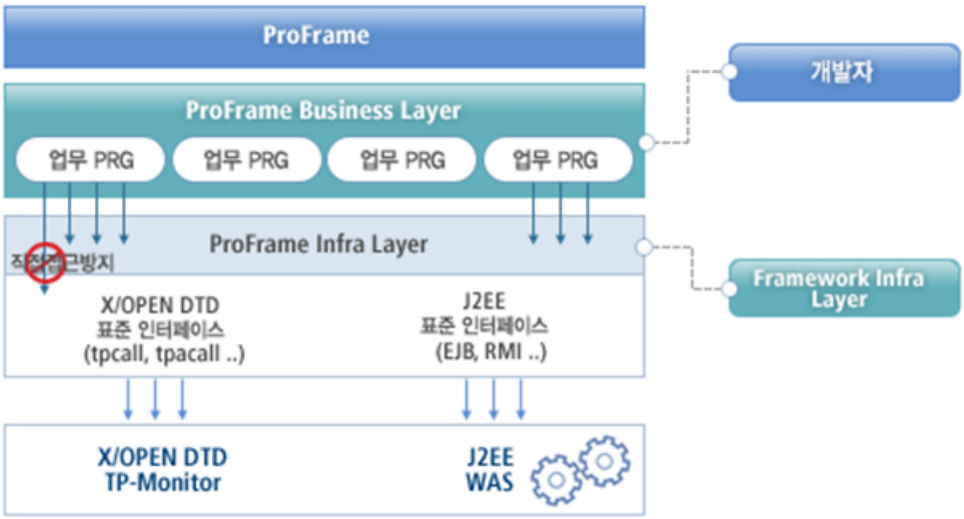
ProFrame는 온라인 서비스 또는 배치 서비스를 개발하기 위한 프레임워크이다. 즉, 공통적인 구조와 개발 구현으로 빠른 시간에 고객의 요구사항 변화에 맞는 유연하고 기능 확장이 뛰어난 결과를 이끌어 낸다. 이와 같이 ProFrame를 사용하여 개발된 서비스는 시스템의 안정적인 성능을 유지하고 유지보수를 쉽게 할 수 있도록 지원한다.

또한 ProFrame은 GUI(Graphical User Interface) 기반의 개발 툴인 ProFrame C 스튜디오(이하 스튜디오)를 사용하여 개발자가 쉽게 비즈니스 로직을 구현할 수 있도록 통합 개발환경을 제공하고 있다. 이를 통해 개발자는 EMB(Enterprise Module Bus) 기술을 이용한 비즈니스 로직 중심의 서비스 플로우를 개발할 수 있다.

1.2. 개발 특징

ProFrame은 아래 그림과 같이 개발자 영역과 프레임워크 영역으로 나뉜다.

[그림 1.1] ProFrame - 개발자 및 프레임워크 영역

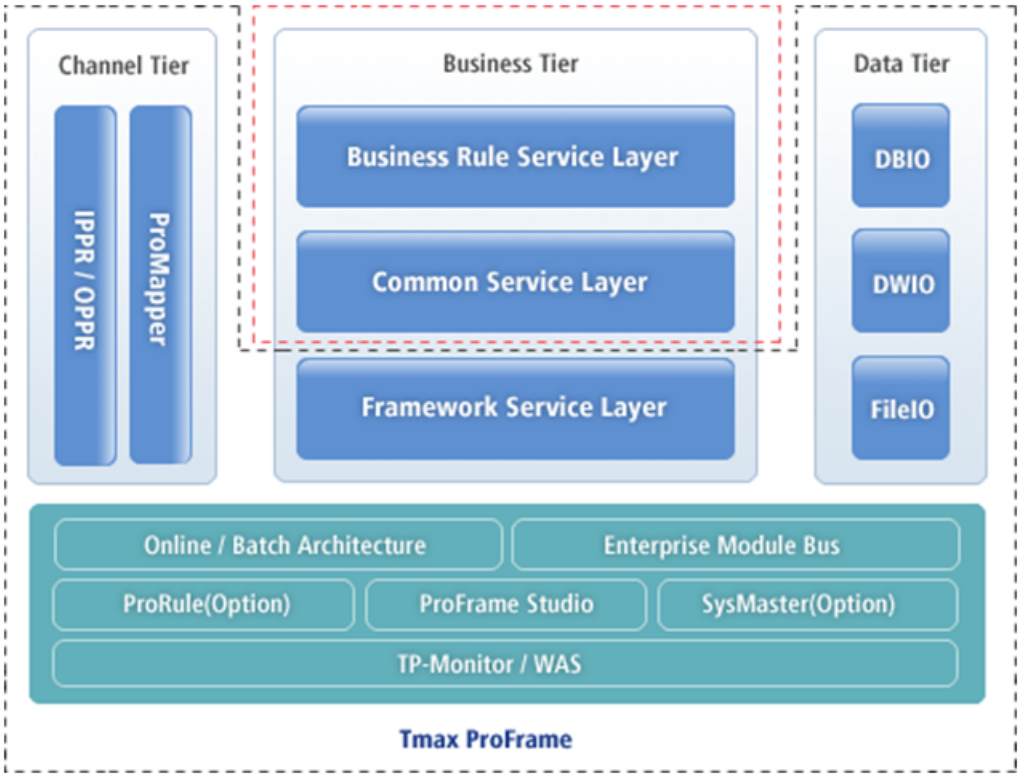


개발자는 **ProFrame Business Layer** 즉, 업무 영역에서 순수 업무 로직만을 구현하고 시스템의 안정성을 위한 기능이나 미들웨어 기능, 트랜잭션 관리, 인터페이스 처리 등의 기능들은 **ProFrame Infra Layer** 즉, 프레임워크 영역에서 직접 담당하거나 개발자에게 추상화된 API로 제공한다.

다음은 개발자 영역과 프레임워크 영역에 대한 특징이다.

- 개발자 영역(ProFrame Business Layer)
  - 순수 업무 로직만 구현
  - 거래 유형별 템플릿을 제공
  - API를 이용하여 시스템 자원 접근
- 프레임워크 영역(ProFrame Infra Layer)
  - 메모리 누수 관리
  - 트랜잭션에 대한 Commit/Rollback 처리
  - 코어 인터페이스 처리

[그림 1.2] ProFrame 아키텍처



ProFrame 아키텍처는 위 [그림 1.2]와 같이 3 Tier(Business, Channel, Data Tier) 구조로 나뉘어져 있다. 위 구조는 Business Tier를 Channel 및 Data Tier와 분리하여 개발자가 업무를 개발할 때 비즈니스 로직만을 고려한 시스템을 구축할 수 있도록 지원한다.

또한 Business Tier를 업무계(Business Rule Service Layer), 업무 공통(Common Service Layer), 시스템 공통(Framework Service Layer)의 3 Layer로 분리하여 개발의 표준화 및 편의성을 지원한다.

참고

본 안내서에서는 ProFrame 아키텍처 전반적인 개념을 설명하지 않고 개발과 관련된 요소만 설명한다.

개발자는 ProFrame 아키텍처를 기반으로 소프트웨어 설계 및 개발 단계에서 다음과 같은 역할을 수행한다.

- 설계 단계

  설계 툴로서 기능을 수행하는 EMB 기술을 이용하여 개발자는 업무 단위별 모듈을 구성하고 조합하여 새로운 온라인 또는 배치 서비스를 설계한다.

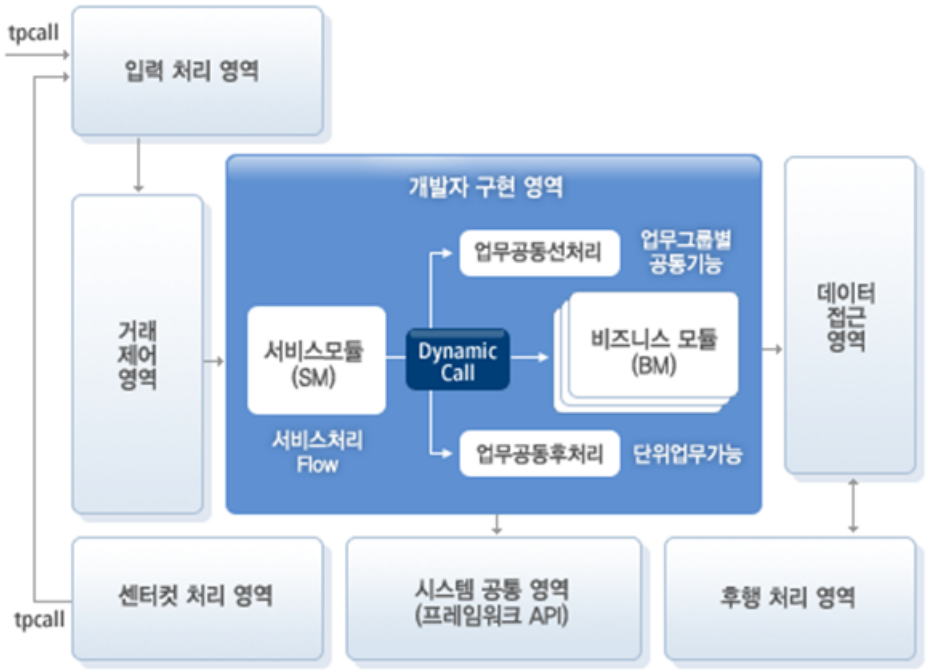
- 개발 단계

  코딩 툴로서 기능을 수행하는 EMB 기술을 이용하여 개발자는 업무 단위별 모듈을 개발하고 모듈 간의 조합을 통해 새로운 온라인 또는 배치 서비스를 개발한다.

  추가적으로 ProFrame은 개발 단계에서 소스 코딩의 일관성을 유지시켜 주며, 유지보수를 쉽게 할 수 있도록 Java 형태의 헝가리안 표기법을 사용한다. 이렇게 개발된 입출력 모듈, 업무 모듈, 데이터 처리 모듈 등은 기능 중심으로 조합하고 연동하여 해당 서비스를 재사용할 수 있다.

다음은 개발자 구현 영역을 중심으로 업무 개발이 구현되는 과정이다.

[그림 1.3] ProFrame Runtime 아키텍처



- 개발자 구현 영역

  스튜디오에 내장된 EMB Designer를 이용하여 개발자가 비즈니스 로직을 구현하는 영역이다.

- 입력처리 영역

  서비스의 입출력 변환과 관련된 처리를 담당하는 ProFrame의 엔진 영역으로 거래의 기본 정보 설정, 공통 파라미터의 로딩, 입력전문 로깅, 전문 헤더 구조체 변환 등을 입력정보를 전달하는 역할을 수행한다.

- 거래 제어 영역

서비스에 대한 거래 가능 여부, 일시 중단, 접근 권한 등에 대한 제어 및 트랜잭션 및 처리 흐름 제어를 담당하는 ProFrame의 엔진 영역이다.

- 센터컷처리 영역

  온라인 서비스를 배치로 대량 및 병렬처리하기 위해 제공된 ProFrame의 엔진 영역이다.

- 시스템 공통 영역

  ProFrame 엔진의 성능, 안정성 및 장애처리를 위해 애플리케이션 프로그램에 제공된 API 영역이다.

- 데이터 접근 영역

  개발자가 등록한 DBIO 모듈을 이용하여 Runtime의 데이터베이스 인터페이스를 담당하는 ProFrame 엔진 영역이다.

- 후행처리 영역

  정보성 데이터는 트랜잭션에서 분리하여 온라인 거래의 응답속도를 높이는데 사용한다.

  분리된 정보성 데이터를 실시간 처리에 가깝게 처리하고 거래정합성을 보장하기 위해 제공되는 ProFrame 엔진 영역이다.

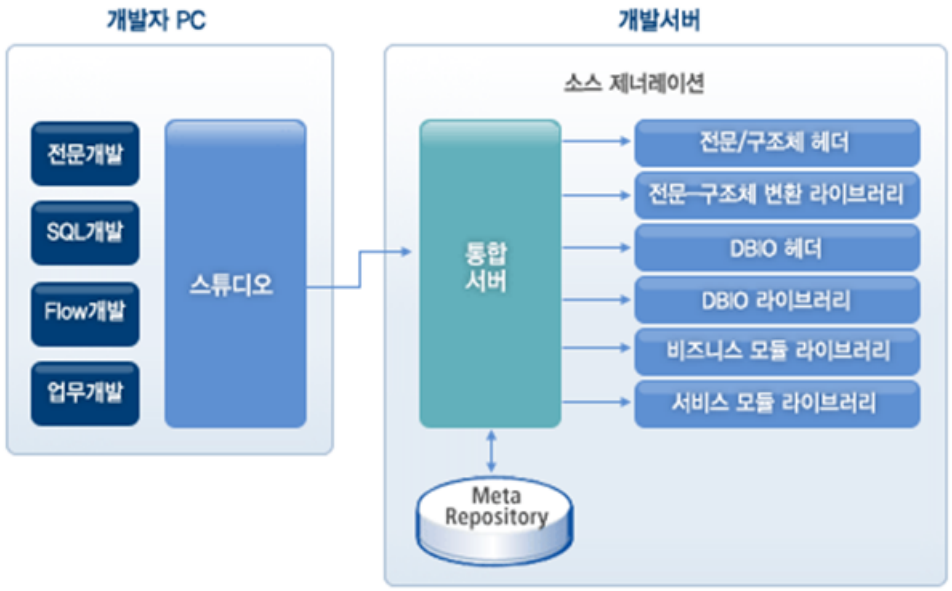
제2장 개발 기술

본 장에서는 ProFrame에서 사용되는 개발 기술에 대해 설명한다.

2.1. 개발 아키텍처

ProFrame은 개발자에게 통합 서버와 스튜디오가 통신할 수 있는 통합 개발환경을 제공한다. 통합 서버와 스튜디오 간의 상호연동을 통해 개발자는 서비스 또는 업무 단위의 모듈을 개발할 수 있다.

[그림 2.1] 통합 서버와 스튜디오 상관도



- 통합 서버

  통합 서버는 **개발자의 로컬 컴퓨터에 설치되어 있는 스튜디오와 통신**하며 개발 과정의 모든 리소스를 메타 기반으로 통합 관리한다. 이를 통해 리소스 버전 정보, 리소스 상호의존성 정보, 입출력 Property 정보, 모듈 목록 관리 기능 등을 수행하며 메타 정보를 기반으로 소스를 빌드하여 바이너리를 생성한다. 또한 개발자별로 권한 관리를 수행할 수 있다.

  ProFrame은 통합 서버가 있어 모든 애플리케이션 리소스에 대한 메타를 관리하고 제어할 수 있다. 또한 소스 버전 관리, Source Generation, 입출력 Property 관리 등을 메타 정보를 기반으로 통합하고 자동화할 수 있다.

- 스튜디오

스튜디오는 소프트웨어 설계 단계에서 개발 표준을 강제화하고 소스 코딩 최소화, 업무 모듈의 재사용 극대화라는 설계사상을 바탕으로 서비스 처리 흐름을 가시화하고 업무 모듈 간의 의존성(Dependency)을 최소화하는 툴이다.

개발자는 이러한 툴을 통해 구조체 및 전문 개발, SQL 개발, 서비스 플로우 디자인, 업무 모듈의 작성 등 모든 개발 작업을 수행할 수 있다.

또한 스튜디오는 **통합 서버와 XML 기반(Object 기반)으로 통신**하며 RI/WS 영역별로 서버의 메타 저장소에 있는 리소스 정보 및 목록을 조회하고 작성한 소스 및 정보를 서버 상에서 컴파일(또는 저장)을 요청한다. 또한 개발자 권한 관리, 소스 버전 관리 등의 리소스 관리가 통합적으로 이루어진다.

2.1.1. 통합 서버

ProFrame은 통합 서버를 통해 다음과 같은 역할을 수행한다.

- 서비스 및 공유 모듈의 버전을 관리한다.
- 소스를 자동으로 생성한다.
- 이력 및 권한을 관리한다.
- 리소스를 통합하여 관리한다.
- 리소스 영향도 분석 및 배포 기능을 수행한다.
- 사용자 및 그룹을 관리한다.
- 프로젝트별로 분기된 서비스를 제공한다.
- 운영 이미지와 개발 형상을 제어할 수 있는 트랜잭션을 컨트롤한다.

ProFrame은 각각의 개발도구가 스튜디오로 통합되어 개발도구 간의 상호 연동이 이루어짐으로써 통합 개발환경을 제공한다.

예를 들어 스튜디오에서는 입출력 구조체 및 전문 정보, 매핑 정보, 서비스 플로우 정보, 데이터베이스 접근 정보 등이 자동으로 소스가 생성되며, 통합 서버를 통해 모든 리소스가 XML 메타로 관리되고 이 메타에 템플릿을 적용하여 특정 플랫폼에 최적화된 소스를 자동 생성한다. 이와 같이 소스를 자동으로 생성하는 기능은 통합 서버에 Source Generation 엔진이 포함되어 있어 가능하다.

그리고 고객의 요구사항 또는 프레임워크의 최적화 작업 등과 같은 변경 요청이 발생하는 경우 변경사항에 유연한 대처를 위해 메타 기반의 템플릿 소스 생성방식을 설정하였다. 따라서 새로운 플랫폼 및 프로그래밍 언어에 대한 요구를 쉽게 충족시킬 수 있는 구조라고 할 수 있다.

또한, RI/WS 영역을 분리하여 개발자의 프로그램 수정이 다른 개발자들에게 주는 영향을 최소화하여 좀 더 유연한 개발을 할 수 있게 한다.

이러한 구조는 개발자의 특성에 따라 소스 코딩을 최소화하고 정형화하여 개발 표준에 맞는 최적화된 템플릿을 생성한다. 이는 개발 표준에 따라 소스를 생성하기 때문에 소스의 가독성이 높아지고 시스템 성능의 안정적인 서비스를 기대할 수 있다.

그리고 ProFrame에 추가된 멀티 프로젝트 기능을 이용하면 단일 통합 서버를 통해 동시에 여러 프로젝트를 진행할 수 있다.

2.1.2. 스튜디오

스튜디오는 온라인, 배치, 모듈 단위의 서비스 프로그램을 개발함에 있어 신속하고 안정된 개발을 지원하는 GUI 기반의 개발 툴이다.

구성요소

스튜디오는 크게 3가지로 구성된다.

- ProMapper 편집기

  ProMapper 편집기는 다양한 채널로부터 입출력 정보를 저장하고 해당 서비스가 이해할 수 있는 구조체 정보로 변환하는 등의 역할을 수행한다.

- DBIO 편집기

  개발자는 DBIO 편집기를 통해 SQL을 정의하고, Inspect 기능을 통해 미리 결과를 살펴 봄으로써 개발 생산성을 향상시킬 수 있다. 또한 생성된 SQL을 독립적으로 관리함으로써 관리의 용이성이 증대된다.

- EMB Designer

  ProFrame은 EMB 기술을 통해 업무를 처리하는 서비스 모듈 등이 만들어지면, 그 모듈들을 조합하여 다양하고 새로운 서비스를 만들 수 있다. 즉, 기존 서비스 모듈을 재사용하여 소스 코딩 없이 새로운 서비스를 개발할 수 있으므로 서비스 기반의 아키텍처 구현이 가능하다.

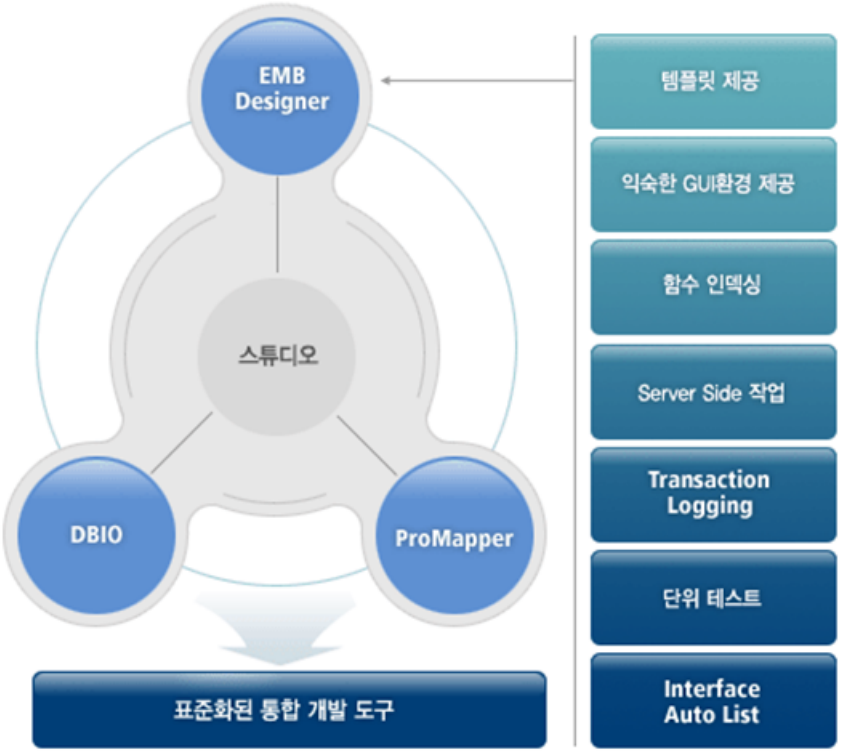
참고

ProMapper, DBIO, EMB Designer에 대한 상세한 내용은 본 안내서 중 “제3장 개발 구성요소”를 참고한다.

주요 기능

스튜디오가 제공하는 주요 기능은 다음과 같다.

[그림 2.2] 스튜디오 기능도



- 템플릿 제공

  정형화된 기본 템플릿 파일을 자동으로 생성해준다.

- 익숙한 GUI 환경 제공

  Eclipse 기반의 편집기를 이용하여 개발자들에게 익숙한 GUI 개발환경을 제공한다.

- 함수 인덱싱

  검색 창의 함수를 마우스로 더블클릭하면 편집 창에서 원하는 해당 함수를 쉽게 찾아가는 기능을 제공한다.

- Server Side 작업

  TP-Monitor 프로그램의 원격 컴파일, 서버 종료, 서버 재기동 등의 기능을 제공한다.

- Transaction Logging

  스튜디오 내에서 데이터베이스 트랜잭션에 대한 로그를 확인할 수 있다.

- 단위 테스트

  생성된 서비스 또는 업무 모듈이 정상적으로 동작하는지 단위 테스트할 수 있다.

- Interface Auto List

입출력 인터페이스 헤더 파일에 정의된 멤버들을 쉽게 검색할 수 있는 인터페이스 자동 목록 기능을 제공한다.

- 멀티 프로젝트

  동시에 여러 개의 프로젝트를 개발할 수 있는 기능을 제공한다.

- 개발 영역과 운영 이미지 분리

  리소스를 수정하는 개발 영역과 리소스 개발이 완료된 저장소인 운영 이미지를 분리하여 관리할 수 있다.

2.2. Eclipse 3.2™ 기반 플러그인 방식

스튜디오는 Eclipse 3.2™ 기반에 플러그인 방식으로 만들어졌다. 기본이 되는 Eclipse 3.2에 공통 모듈인 코어 플러그인과 코어 스튜디오에 각각의 프로바이더 즉, ProMapper, DBIO, EMB Designer 등을 설치하였다.

참고

Eclipse 플러그인 아키텍처에 대한 내용은 http://www.eclipse.org를 참고한다.

이와 같은 플러그인 방식을 통해 ProFrame은 다른 시스템과의 연계에도 뛰어난 확장성과 유연성을 가지고 있다. 기존에 설치된 프로바이더와 독립적으로 구분하여 설치할 수 있으며, 설치 방법 또한 쉽고 간편하다.

2.3. 다른 시스템 지원 여부

이전 절에서 설명하였듯이 스튜디오의 확장성과 유연성은 주로 다른 시스템과의 연계를 플러그인 방식으로 구성하게 된다.

예를 들어 TmaxSoft의 Tmax ProFactory® 또는 Tmax ProRule®과의 연계할 때 Tmax ProFactory® , Tmax ProRule®에서 제공하는 Eclipse 기반의 플러그인을 제공받음으로써 스튜디오의 EMB Designer에서는 상품 팩토리와 룰 정보 등을 사용할 수 있다. 더 나아가 TmaxSoft의 Tmax ProBus®와도 연계하여 다른 시스템의 각종 리소스들을 사용할 수 있다.

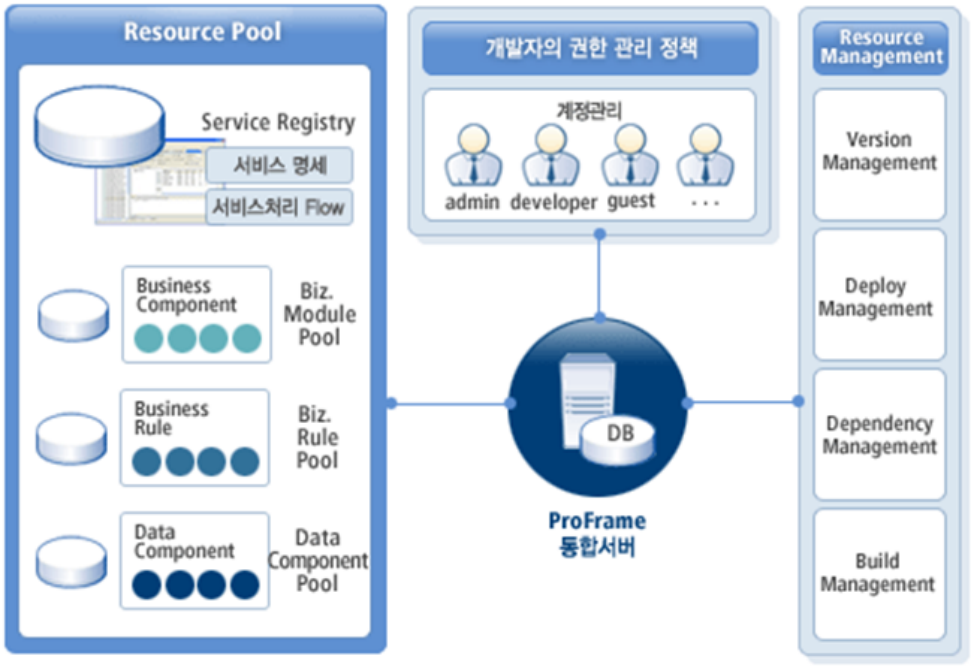
제3장 개발 구성요소

본 장에서는 ProFrame를 이용하여 서비스 또는 업무 단위의 모듈을 개발할 때 필요한 리소스와 Property에 대해 설명한다. 추가적으로 스튜디오에 내장된 대표 프로바이더 즉, ProMapper, DBIO, EMB Designer에 대해 설명한다.

3.1. 리소스

ProFrame은 개발되는 모든 리소스를 메타 관리를 통해 강력한 통합 개발환경 관리 기능을 제공한다.

[그림 3.1] 리소스 관리 및 사용



ProFrame의 모든 리소스 정보는 기본적으로 XML 메타 방식으로 표현하여 이를 관리하는 통합 서버의 데이터베이스에 저장되어 관리된다. 또한 XML 메타를 기반으로 템플릿 기반의 소스 생성 방식을 채택하여 특정 플랫폼에 종속되지 않고 관리할 수 있다. 템플릿 선택에 따라(C 또는 Java, 기타) 특정 플랫폼에 최적화된 소스를 생성하고 이를 통합 서버에 XML 메타와 함께 저장하는 구조이다. 또한, 선택적으로 배포 및 형상 관리를 위해 빌드된 바이너리도 함께 저장할 수 있는 구조를 제공한다.

효율적인 관리를 위해 XML에 표현된 기본 정보를 데이터베이스의 특정 테이블에 별도로 관리하여 영향도 분석, 의존성 검사, 버전 관리, 배포 관리, 형상 관리 등을 최적으로 수행할 수 있는 기능을 제공한다.

ProFrame에서 소스 생성 및 컴파일, 빌드는 통합 개발환경에서 처리할 수 있으며 통합 서버의 별도 명령어를 통해 XML 메타를 기반으로 템플릿 엔진을 이용하여 소스를 일괄 생성, 컴파일, 빌드, 배포할 수 있다.

즉, 모든 소스 및 분석, 설계 단계의 리소스를 통합 서버를 통해 관리하고 버전 및 이력 관리는 기본 기능으로 제공한다.

3.2. Property

ProFrame에서는 IT 용어, 업무 용어, 데이터베이스 명칭, 컬럼 명칭 등 다양한 형태의 용어들을 데이터 사전으로 등록하고 관리하여 애플리케이션 프로그램을 개발할 때 사용할 수 있다. 이를 **Property**라고 한다. Property는 RI/WS 동일하게 사용한다.

Property는 다음과 같은 역할을 수행한다.

- 코드 기반 모델링

  소스를 생성할 때에는 코드 값으로 Generation한다.

- 데이터 통합 관리

  데이터 설계자, 분석가, 개발자, 품질 담당자 등의 협의 체계를 기반으로 전사 차원의 통합 관리를 한다.

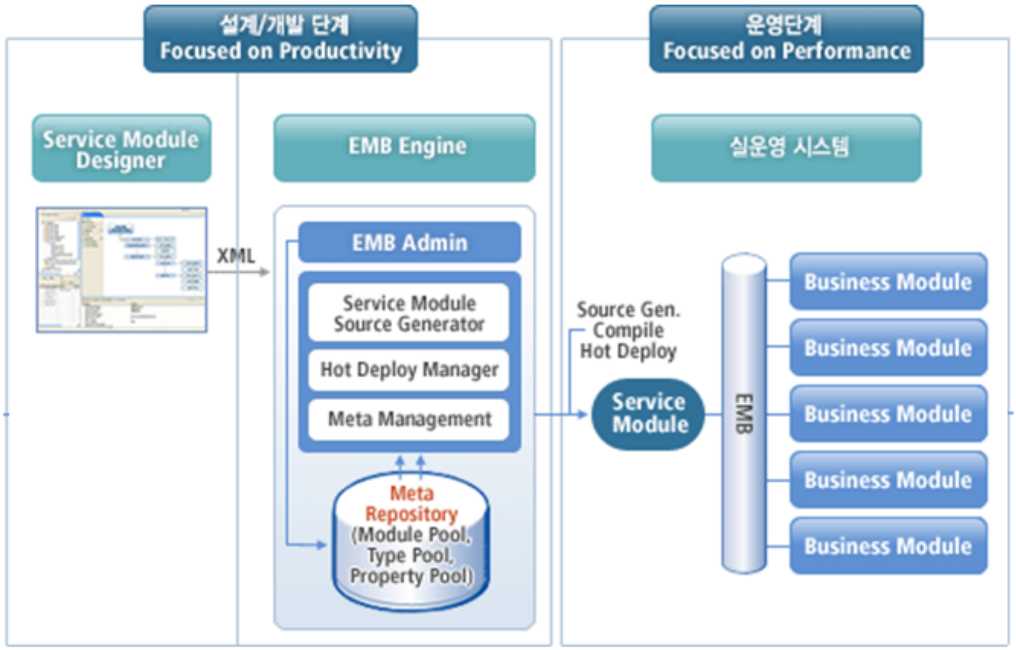
- 일관된 뷰 제공

  메타 데이터를 통합하여 전사적으로 공유하고 일관된 뷰를 제공받는다.

3.3. EMB

EMB는 서비스 처리 흐름을 가시화하고 업무 모듈 간의 의존성을 최소화하는 아키텍처이다. EMB는 SOA 아키텍처를 기반으로 하여 Loosely-Coupled 애플리케이션 모듈 간의 조합을 편리하게 하기 위해 개발되었다.

[그림 3.2] 설계/개발/운영 단계별 EMB 역할



EMB는 소프트웨어 구축 단계 중 설계 단계에서 설계 툴로서의 역할을 수행하며 개발할 때는 코딩 툴로서의 기능을 수행한다.

다음은 EMB를 설계 관점과 개발 관점으로 나누어 설명한다.

- 설계적 관점
  - 큰 단위의 업무 플로우가 보인다.
  - 트리 형태의 구조로 영향도 분석이 쉽다.
  - GUI 기반에서 마우스로 드래그 앤드 드롭 형식으로 작성이 가능하다.
- 개발적 관점
  - 기본적인 업무 플로우만으로 소스 생성이 가능하다.
  - 입출력 매핑을 할 때 ProMapper의 **Smart Mapping** 기능을 이용하여 자동으로 완성할 수 있다.
  - 설계 측에 변경이 필요하면 변경할 때 설계서에 실시간으로 반영된다.

또한 EMB 기반 시스템에서는 다음과 같은 차별화 기능을 제공한다.

- 비즈니스 로직을 자산화
  - 업무 서비스 플로우의 자산화
  - Module Pool, Property Pool 기능을 통해 업무 로직을 재사용한다.
  - Meta Repository 기반의 업무, 데이터, 인터페이스를 표준화한다.
- 유연한 Composite 서비스 개발
  - 업무 서비스의 분해와 조립이 쉬운 아키텍처를 지원한다.
  - 스튜디오를 이용하여 서비스 연동을 작성한다.
  - Module Pool 기반의 업무 로직 기능을 블록화한다.
- 성능을 고려한 아키텍처
  - Source Generation을 이용하여 서비스를 생성하므로 ProFrame 시스템의 성능 부하를 최소화한다.
  - Dynamic Call 기반의 호출 구조로 2PC 최소화 아키텍처를 제공한다.
  - CommBuff를 이용하여 업무 모듈 간의 데이터 공유로 디스크 IO를 최소화한다.

참고

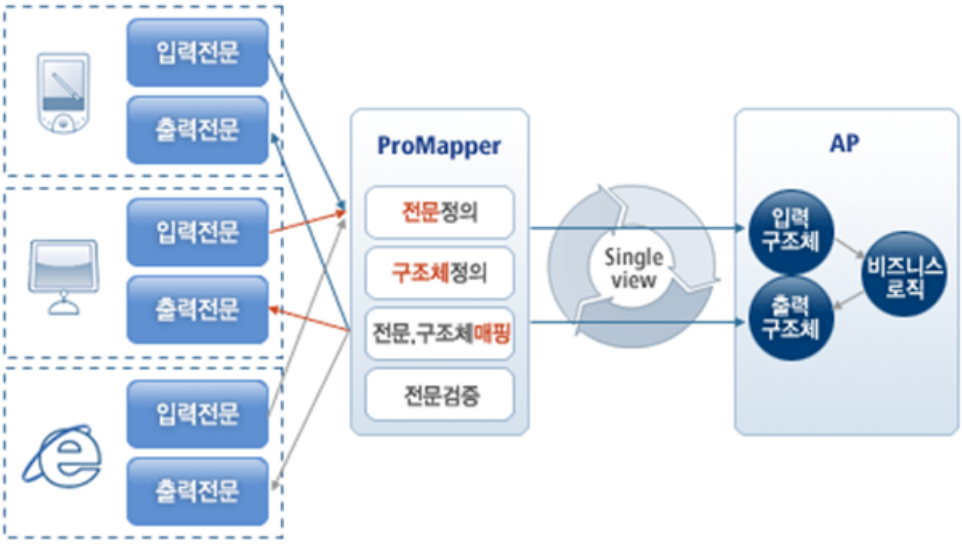
Dynamic Call은 시스템 운영 중에 동적으로 라이브러리를 교체하기 위해 ProFrame에서 제공하는 인터페이스 모듈이다.

CommBuff는 업무 모듈 간의 데이터 공유를 하기 위해 ProFrame에서 제공하는 인터페이스 모듈이다. 메모리 전달 및 관리를 수행한다.

3.4. ProMapper

ProMapper는 Channel Tier에서 전문 변환 및 모듈 간의 인터페이스 변환을 하기 위해 개발되었다.

[그림 3.3] ProMapper



개발자가 스튜디오에 내장된 ProMapper 편집기를 사용하여 개발중인 서비스에서 사용할 입출력 전문을 정의하면, ProMapper는 해당 전문을 C 언어로 표현할 수 있는 C 구조체와 C 소스를 작성하여 변환을 해준다. 그리고, 다양한 채널을 통해서 해당 서비스를 호출하면 ProMapper에 의해서 해당 채널의 전문은 서비스의 입력 구조체로 변환이 되며, 서비스에서 출력 구조체에 출력 값을 넣어주면 ProMapper에 의해서 출력 전문이 작성되어 해당 채널로 반환하게 된다.

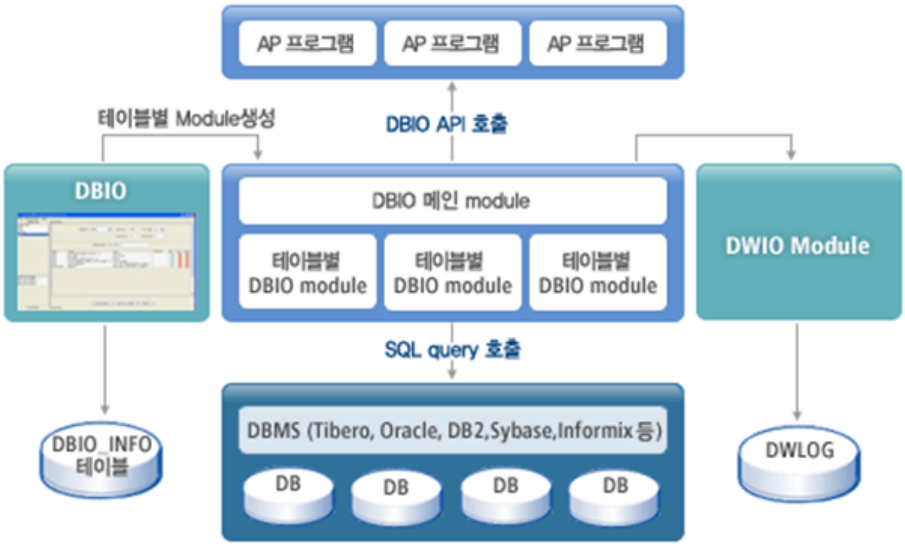
ProMapper는 서비스의 입출력, 모듈의 입출력 정의에 필요한 구조체의 생성 및 구조체와 구조체, 구조체와 전문 간의 변환 정의에 대한 정보를 관리하고 소스 생성을 지원한다. ProMapper에서 지원하는 전문 타입에는 FDL, FML, Delimiter이 있다.

전문 타입	설명
FLD (Fixed-Length-Data)	– 고속 해석 및 작성 가능 – 유연성이 낮음 – 중간 수준의 가독성
FDL/FML(Field Definition Language)	– 유연성이 높음 – 전문 해석 및 작성에 추가적인 부하 – 가독성이 매우 낮음
Delimiter	– 중간 수준의 유연성 – 중간 수준의 성능 – 높은 가독성

3.5. DBIO

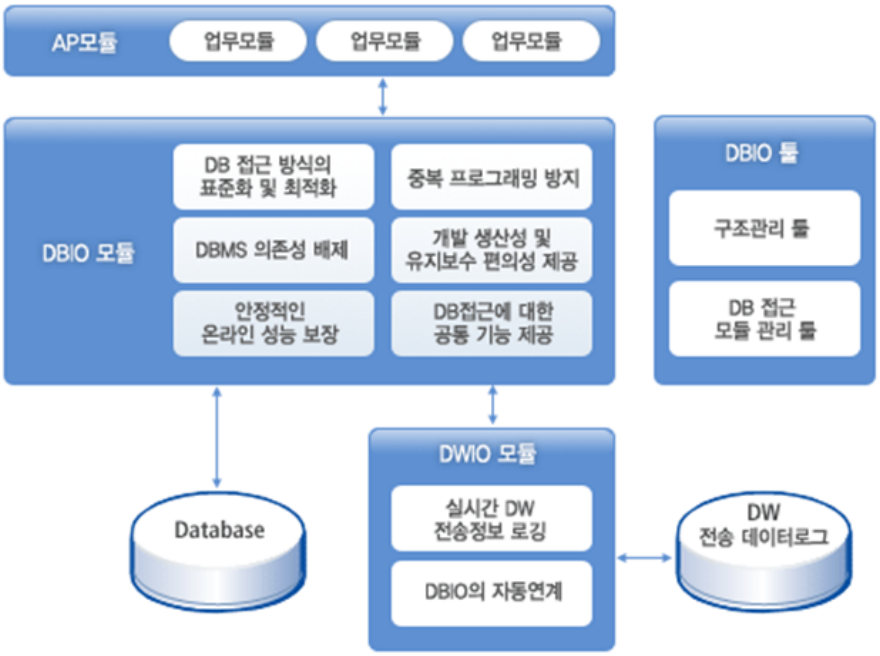
데이터 처리에 대한 개발 생산성을 높이기 위해 데이터베이스 접근에 대한 독립성과 성능을 보장해주는 계층이 필요하다. ProFrame은 데이터베이스 접근 통합을 위한 DBIO 모듈과 연계 시스템과의 데이터 전달을 위한 DWIO 모듈을 제공하여 데이터베이스 관리시스템과 독립적인 업무를 개발한다.

[그림 3.4] DBIO의 기능과 효과



DBIO는 데이터베이스 접근방식을 표준화하여 관리함으로써 중복된 SQL 프로그래밍을 방지하고, 데이터베이스 접근을 공통화하여 처리함으로써 개발 생산성 및 유지보수의 편의성을 제공하기 위한 목적으로 개발되었다.

[그림 3.5] DBIO 아키텍처



다음은 DBIO 편집기의 주요 기능에 대한 설명이다.

기능	설명
SQL 관리	개발된 모든 SQL의 변경 내용을 저장하여 형상 관리를 한다.
Runtime 권한 관리	개발된 SQL은 DBA의 승인 절차 없이 Runtime 시 시스템에 적용되지 못하도록 관리한다.
SQL동작 예측	개발된 SQL의 실행 계획 정보를 모니터링한다.
CRUD 모니터링 생성	프로그램 대 테이블 접근을 CRUD별로 모니터링한다.
개발자 권한 관리	개발된 SQL 편집을 그룹별로 권한을 관리한다. 조인 및 서브 쿼리 등은 DBA의 승인 관리를 한다. SQL을 다른 개발자에게 이관할 때 승인 관리를 한다.

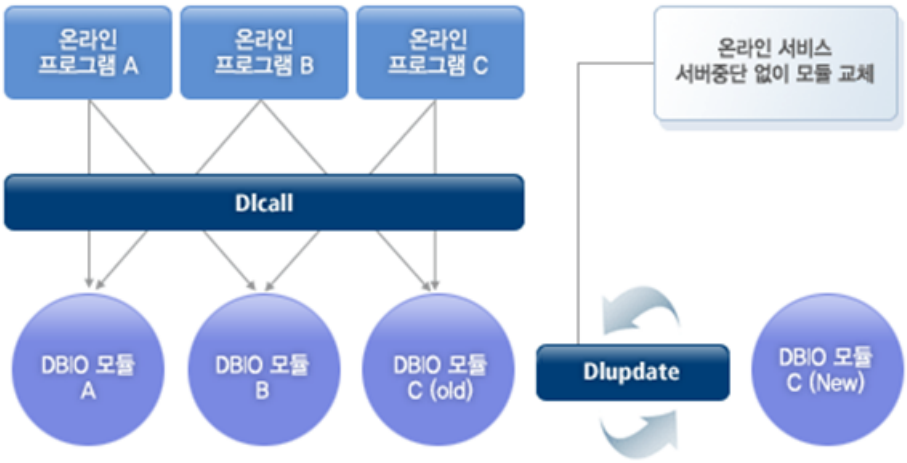
개발자는 DBIO 편집기를 통해 데이터베이스 접근 로직을 정의하고 ProFrame에서 소스를 자동 생성해 줌으로써 개발 생산성을 향상 시킨다.

[그림 3.6] DBIO 편집기

즉, DBIO 편집기에서 작성된 SQL은 Source Generator에 의해 간단한 Exec SQL 소스로 생성되고 컴파일 및 배포까지 개발자의 간단한 명령어 실행으로도 가능하다.

DBIO는 테이블 정보 또는 데이터베이스 접근 로직이 변경되는 경우 서비스 중단 없이 변경사항을 즉시 반영하기 위한 방안으로 ProFrame은 Runtime에 변경된 DBIO 모듈을 동적으로 반영할 수 있는 Hot Deploy 구조를 제공한다.

[그림 3.7] DBIO의 Hot Deploy 구조



ProFrame의 DBIO는 여러 유형의 SQL을 테이블과 컬럼 정보를 선택하여 생성하거나 입력하여 모듈의 메타 데이터를 생성하고 이를 이용하여 DBIO 모듈을 생성한다. 개발자는 품질이 검증된 기존에 개발된 SQL Query Pool을 바탕으로 개별 쿼리 설계 및 구현할 때 참조할 수 있으며 담당업무 모듈에 적용하기 전 SQL 쿼리를 시뮬레이션하여 실행 계획 정보 등을 참조하여 Runtime에 발생하는 문제점들을 미연에 예측하고 방지할 수 있다.

또한 **SQL의 테스트 및 실행 플랜보기** 등의 SQL 작성 및 테스트에 필요한 기능들을 지원하고, Repository를 기반으로 권한 관리, 버전 관리, 영향도 분석 등의 통합 개발환경과 통합된 기능들을 지원한다.

제4장 개발 관련 도구

본 장에서는 ProFrame에서 제공하는 개발 관련 도구에 대해 설명한다.

4.1. 스튜디오

스튜디오는 ProFrame 기반의 애플리케이션 프로그램을 개발을 위한 통합 개발환경을 제공하며 온라인 서비스 프로그램, 배치 서비스 프로그램, 모듈 서비스 프로그램을 개발에 대한 편의성을 제공한다.

스튜디오는 개발자가 응용 프로그램을 개발하기 위한 통합 개발환경 및 서비스, 모듈 프로그램 개발에 대한 템플릿 소스 코드 제공을 지원한다. 또한, 개발된 서비스 프로그램의 컴파일, 디버깅, Hot Deploy 기능을 통해 소스 코딩 후 보다 쉽게 프로그램을 등록, 디버깅할 수 있는 환경을 제공한다.

스튜디오의 특징은 다음과 같다.

- 통일된 개발환경을 제공한다.
- Property 메타 정보를 기반으로 한 입출력 변수의 표준화를 지원한다.
- 다양한 응용 프로그램 개발에 대한 편의성을 제공한다.
- 개발된 리소스와 EMB를 통해 개발자는 보다 쉽게 서비스와 업무 모듈을 개발할 수 있다.
- 복잡한 시스템의 서비스 코드로부터 업무 코드를 분리하여 개발자는 서비스 개발에 집중할 수 있다.
- 다양한 업무 - 온라인, 배치, 그리고 업무 모듈 프로그램 개발이 가능하다.
- GDB 디버그를 사용하여 오류를 보다 쉽게 찾아내고 수정할 수 있어 디버깅이 용이하다.
- 소스 Deploy 기능을 통해 보다 쉽게 서비스를 배포할 수 있다.
- EMB Designer를 통해 서비스를 쉽게 연동할 수 있다.

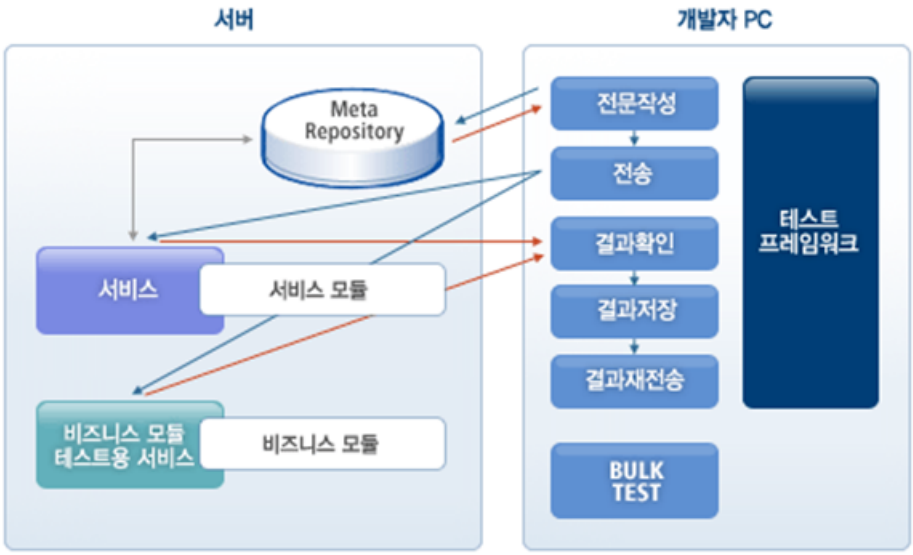
스튜디오는 Eclipse 기반의 통합 개발환경으로 “제3장 개발 구성요소”에서 언급된 다음과 같은 개발도구 **Perspective**를 포함하고 있다.

구분	설명
ProMapper 편집기	구조체 및 전문 정보 등록, 전문-구조체 변환 정보 등록
DBIO 편집기	SQL 등록
EMB Designer	서비스 및 모듈 플로우 디자인

4.2. 테스트 프레임워크

테스트 프레임워크는 개발된 서비스 모듈이나 비즈니스 모듈을 테스트할 수 있는 툴로서 모듈의 구조와 필드들을 시각적으로 확인할 수 있고 실제 값을 입력하여 전송하고 결과를 확인할 수 있다.

[그림 4.1] 테스트 프레임워크



또한 테스트 케이스들을 저장하고 저장된 케이스들을 일괄적으로 테스트하여 저장된 결과들과 현재의 결과들을 비교할 수 있는 기능을 제공한다. 이 외에 Excel로 작성한 하나의 데이터 파일을 가지고 **BULK TEST**를 수행할 수 있는 기능을 제공한다.

지원 기능	**설명**
단위 테스트	– 함수 단위의 비즈니스 모듈 테스트 가능 – 반환된 결과 확인 및 결과 상세내용 조회 – 헤더 필드도 수정가능
테스트 케이스 관리 (Fix2 지원 예정)	– 테스트 케이스 저장(입력, 출력 포함) – 저장된 입력 값 재전송 – 재전송 결과 값와 저장된 값과 상세 비교
BULK TEST (Fix2 지원 예정)	– 입력 데이터를 CSV 파일로 관리 – tpacall로 시스템 부하 테스트 – 결과 값 확인

제5장 개발 절차

본 장에서는 서비스 또는 업무 모듈 프로그램의 개발 절차 및 개발 툴에 대해 설명한다.

5.1. 서비스 프로그램 개발 절차

본 절에서는 서비스 프로그램을 개발자가 개발을 시작하는 단계부터 단위 테스트까지의 개발 절차를 설명한다.

다음은 서비스 프로그램의 개발 절차이다.

1. 설계

   개발자는 설계서를 작성하고 거래 파라미터, 입출력 정의, SQL 쿼리를 작성한다.

   – 거래 파라미터 등록

     개발자는 설계자가 작성한 설계서에 따라 거래 파라미터를 등록한다. 거래 파라미터 등록은 거래 파라미터 등록 화면을 이용하여 거래 코드, 서비스 ID, 입력 구조체명 및 기타 거래정보 등을 입력한다.

   – 입출력 정의

     ProMapper 편집기를 이용하여 설계서에 따라 해당 서비스 프로그램의 입출력 전문을 등록하고 입출력 전문의 구조체를 정의한 헤더 파일을 생성한다.

   – SQL 쿼리 작성

     DBIO 편집기를 이용하여 설계서에 따라 SQL 쿼리를 등록하고, 컴파일 후 DBIO 라이브러리 생성까지 완료한다. SQL 쿼리를 작성하기 전에 작성하려는 SQL 쿼리와 동일한 내용의 DBIO 맵이 등록되어 있는지 확인한다. 여기서 내용이 중복되는 DBIO 맵이 존재하지 않도록 한다.

2. 프로젝트에 서비스 추가 및 서비스 모듈 구현

   프로젝트에 서비스를 추가하고 설계서에 따라 서비스 모듈을 구현한다. 스튜디오를 이용하여 XA 또는 Non-XA를 구분하여 현재 프로젝트에 서비스를 추가하고, 설계서에 따라 EMB Designer에서 서비스 모듈을 구현한다.

3. 컴파일 및 Hot Deploy

   서비스 모듈 구현이 완료되면 스튜디오에서 **[컴파일]** 메뉴를 실행하여 소스 컴파일을 실행하고 Hot Deploy를 실행하기 위해 **[Dlupdate]** 메뉴를 실행한다.

4. 단위 테스트

   테스트 프레임워크가 제공하는 단위 테스트 기능을 이용하여 개발된 서비스 프로그램을 테스트한다. 테스트 후 프로그램에 문제가 있으면 1번 단계부터 다시 시작한다.

컴파일 및 Hot Deploy 실행 결과가 정상이면 TP-Monitor 서버를 재기동하여 TPM에 서비스를 등록하고, 테스트 프레임워크를 이용하여 단위 테스트를 수행한다. 스튜디오에서 **[단위 테스트]** 메뉴를 실행하면 테스트 프레임워크가 실행되면 단위 테스트를 수행할 수 있다.

5.2. 업무 모듈 프로그램 개발 절차

본 절에서는 업무 모듈 프로그램을 개발자가 개발을 시작하는 단계부터 단위 테스트까지의 개발 절차를 설명한다.

다음은 업무 모듈 프로그램의 개발 절차이다.

1. 설계

   개발자는 설계서를 작성하고 SQL 쿼리를 작성하며 업무 모듈 프로젝트를 설계한다.

   – SQL 쿼리 작성

     DBIO 편집기를 이용하여 설계서에 따라 SQL 쿼리를 등록하고, 컴파일 후 DBIO 라이브러리 생성까지 완료한다. SQL 쿼리를 작성하기 전에 작성하려는 SQL 쿼리와 동일한 내용의 DBIO 맵이 등록되어 있는지 확인한다. 여기서 내용이 중복되는 DBIO 맵이 존재하지 않도록 한다.

2. 입출력 정의 및 비즈니스 로직 구현

   업무에 대한 비즈니스 모듈을 생성하고 입출력 정의와 비즈니스 로직을 구현한다.

   – 비즈니스 모듈 생성

     스튜디오를 이용하여 업무 모듈 프로젝트를 생성한다. 이때 선택한 템플릿의 내용으로 .c, make 파일 및 헤더 파일이 자동으로 생성된다.

   – 입출력 정의 및 비즈니스 로직 구현

     업무 모듈 프로젝트를 생성할 때 자동으로 생성된 .h 파일에 설계서에 따라 ProMapper 편집기를 이용하여 입출력 구조체를 **신규 서비스모듈 생성** 대화상자에서 설정하고 EMB Designer에서 비즈니스 로직을 구현한다.

3. 컴파일

   업무 모듈 프로그램 구현이 완료되면 스튜디오에서 **[컴파일]** 메뉴를 실행하여 소스를 컴파일한다.

4. Hot Deploy

   컴파일이 정상으로 완료된 후에 해당 모듈이 Hot Deploy를 위한 Dlcall 대상 라이브러리이면 Dlupdate를 실행한다. 스튜디오에서 Hot Deploy를 실행하기 위해서 **[Dlupdate]** 메뉴를 실행한다.

5. 단위테스트

   컴파일 및 Hot Deploy 실행 결과가 정상이면 TP-Monitor 서버를 재기동하여 TPM에 서비스를 등록하고, 테스트 프레임워크를 이용하여 단위 테스트를 수행한다. 스튜디오에서 **[단위 테스트]** 메뉴를 실행하면 테스트 프레임워크가 실행되면 단위 테스트를 수행할 수 있다.

제6장 안내서 구성

6.1. 소개

제품에 대한 안내서가 있어도 제품에 대해 여러 가지 안내서가 존재하다 보니 제품을 처음 접하거나 제품에 익숙하지 않은 사용자는 원하는 안내서를 찾는데 문제를 겪을 수 있다.

예를 들어, ProFrame에서 DBIO 모듈을 생성하는 방법에 대한 정보를 얻고자 할 때 정확히 원하는 내용이 ProFrame DBIO 개발 안내서에 기술되어 있는지 아니면 함께 참고해야 할 안내서가 있는지를 파악하고 있어야 한다.

본 장에서는 개발과 관련된 안내서 사용에 있어 개발자의 편의를 돕기 위해 ProFrame 안내서를 어떻게 활용할 수 있는지 기술한다. 즉, ProFrame 개발과 관련된 안내서에는 어떤 것들이 있고, 각 안내서에서 어떤 내용을 기술하고 있으며, 각 안내서가 다른 안내서와 어떻게 연관을 맺고 있는지에 대해 기술한다.

ProFrame 개발과 관련된 안내서를 처음 접하는 개발자는 본 장을 주의 깊게 읽어볼 것을 권장한다. 비록 본 장이 실제 제품 사용법과 직접적인 관련이 있는 것은 아니지만, 안내서 구성을 전체적으로 이해하는데 도움이 되도록 작성하였다.

6.2. 안내서 구성과 내용

다음은 ProFrame 개발과 관련된 안내서 목록이다.

ProFrame	**ProFrame 안내서**
공통	개발 안내서
	스튜디오 안내서
모듈	ProMapper 개발 안내서
	FileIO 개발 안내서
	DBIO 개발 안내서
	EMB 개발 안내서
기타	단위 테스트 안내서
	유틸리티 안내서
	온라인 프로그래밍 안내서
	배치 프로그래밍 안내서

ProFrame은 개발과 관련된 총 10권의 안내서를 제공한다. 각 안내서에 대한 내용은 본 장에 설명되어 있으니 특정한 내용에 대해 빨리 찾기를 원한다면 아래의 내용을 먼저 확인하기 바란다.

- ProFrame 개발 안내서

본 안내서이다.

- ProFrame 스튜디오 안내서

  서비스 또는 업무 모듈 프로그램을 작성하고 배포할 수 있는 GUI 기반 툴인 스튜디오의 사용법에 대해서 상세히 기술한다.

  스튜디오 안내서의 구성은 다음과 같다.

  – 스튜디오 소개 및 화면 구성

  – 스튜디오 실행 방법

  – 스튜디오 환경설정

  – 단축키

- ProFrame ProMapper 개발 안내서

  스튜디오에 내장된 ProMapper 편집기를 이용하여 애플리케이션 프로그램의 입출력 설계 및 변환 규칙을 생성하는 방법에 대해 기술한다.

  ProMapper 개발 안내서의 구성은 다음과 같다.

  – ProMapper 소개 및 환경설정

  – 리소스 타입별 기본 정보 등록

  – 리소스 타입별 ProMapper 편집기 사용법

    • 구조체 편집기

    • 전문 편집기

    • 맵 편집기

  – 소스생성, 컴파일 및 Dlupdate

  – 생성된 ProMapper 리소스 사용법

  – 가변배열 구조체 사용법 및 사용예제

  – ProMapper 관련 API

  – ProMapper 관련 에러코드

- ProFrame FileIO 개발 안내서

  스튜디오에 내장된 FileIO 편집기를 이용하여 FileIO 모듈을 생성하는 방법에 대해 기술한다.

  FileIO 개발 안내서의 구성은 다음과 같다.

  – FileIO 소개 및 환경설정

  – FileIO 기본 정보 등록

  – FileIO 편집기 사용법

  – 컴파일 및 Dlupdate

- 생성된 FileIO 모듈 사용법 및 사용예제
- FileIO 관련 API

- ProFrame DBIO 개발 안내서

  스튜디오에 내장된 DBIO 편집기를 이용하여 DBIO 모듈을 생성하는 방법에 대해 기술한다.

  DBIO 개발 안내서의 구성은 다음과 같다.

  - DBIO 소개 및 환경설정
  - DBIO 기본 정보 등록
  - DBIO 편집기 사용법
  - 소스 생성, 컴파일 및 Dlupdate
  - DBIO 부가 기능
  - DWIO 기능 연계

- ProFrame EMB 개발 안내서

  스튜디오에 내장된 EMB Designer를 이용하여 EMB 모듈을 생성하는 방법에 대해 기술한다.

  EMB 개발 안내서의 구성은 다음과 같다.

  - EMB 소개
  - EMB Designer 소개
    - 환경설정
    - 화면 구성
  - EMB 모듈 생성 절차
    - EMB 모듈 기본 정보 등록
    - 플로우 작성
    - 플로우 편집
  - 컴파일 및 Dlupdate, 단위 테스트
  - EMB 기타 기능 및 소스 바로 가기 버튼 소개

- ProFrame 단위 테스트 안내서

  스튜디오에서 개발된 서비스 또는 업무 모듈을 단위 테스트 하는 방법에 대해 기술한다.

  단위 테스트 안내서의 구성은 다음과 같다.

  - 테스트 프레임워크 소개
  - 단위 테스트 기능 및 화면 구성
  - 단위 테스트 기능별 테스트 실행 방법

- 스튜디오에서 테스트 로그 보는 방법
- 로그 보기 환경설정 방법
- 단위 테스트 중 문제 해결

- ProFrame 유틸리티 안내서

  ProFrame가 제공하는 유틸리티에 대하여 기술한다.

  유틸리티 안내서의 구성은 다음과 같다.

  - ProFrame API 소개
  - Date 유틸리티 소개 및 관련 API 설명
  - Number 유틸리티 소개 및 관련 API 설명
  - String 유틸리티 소개 및 관련 API 설명
  - Long 유틸리티 소개 및 관련 API 설명
  - 에러처리 및 디버그 유틸리티 소개 및 관련 API 설명
  - 기타 유틸리티 소개 및 관련 API 설명
  - PfmNumber 활용 방법

- ProFrame 온라인 프로그래밍 안내서

  ProFrame를 이용하여 온라인 프로그램을 작성하는 방법에 대해 기술한다.

  온라인 프로그래밍 안내서의 구성은 다음과 같다.

  - 온라인 서비스 프로그램 소개
  - 온라인 서비스 프로그램 작성 절차
  - 온라인 서비스 프로그램 작성 방법
  - 서비스 연동 방법
  - 서비스 연동할 때 주의 사항 소개

- ProFrame 배치 프로그래밍 안내서

  ProFrame를 이용하여 배치 프로그램을 작성하는 방법에 대해 기술한다.

  배치 프로그래밍 안내서의 구성은 다음과 같다.

  - 배치 프레임워크 소개
  - 배치 프레임워크의 종류 소개
    - 일반 배치
    - POD 배치
    - 상주 배치

    - 배치 Job Information

– 배치 생성 절차 소개

– 배치 프레임워크의 종류별 배치 생성 방법

– 배치 파라미터 소개

색인