전력품질 감시 시스템 개발(2): 객체지향 방법론을 적용한 전력품질 분석 소프트웨어 설계

김 영 일*, 방 순 정, 한 진 희, 윤 태 욱
LG산전 전력연구소

Development of Power Quality Monitoring System(PQMS) : Object-Oriented Design for Power Quality Analysis Software

Young-Il Kim*, Soon-Jung Bahng, Jin-Hee Han, Tae-Wook Yun
LGIS R&D Center

Abstract - In this papers, it will be introduced that power quality analysis software design including investigation of mutual relation between power quality data and mechanism for getting data from power quality database according as based on object-oriented methodology.
The UML which is only a language and so is just part of a software development, may be used to visualize, specify, construct, and document the artifacts of software-intensive system.

1. 서 론

본 논문에서는 전력품질 감시 시스템 개발을 위한 전력계통 분야의 전력품질 도메인 분석결과를 토대로, SE(Software Engineering)의 한 분야인 객체지향 방법론을 적용하여 전력품질 데이터 간의 상호관계(데이터 구조 및 관계) 및 프로그램에서 전력품질 분석을 수행하기 위해 전력품질 데이터베이스로부터 관련 정보(데이터)를 읽어오는 메커니즘(Mechanism)을 포함한 전력품질 분석 소프트웨어 설계를 소개하고자 한다.
이곳에서 소개되는 객체지향 설계의 모델링은 UML(Unified Modeling Language)로서 표현되는데, UML은 모델링 언어로서 소프트웨어 개발 방법의 한 분야이고 소프트웨어 산출물을 규정하고 시각화하며 구현하고 문서화(Documentation)하는 언어이다.
방법론 vs 객체지향 방법론 vs UML : 실세계의 현상을 소프트웨어로 구현하는데 있어 "어떻게" 구조화 할 것인가의 방법을 제시하는 것이 방법론이고, 객체지향 방법론은 여러 방법론 중에서 모델을 객체 상호간의 관계로 구조화하는 방법론이며, UML은 구조화된 모델을 표현하는 언어(모델링 언어) 이다.

2. 본 론

2.1 객체지향 방법론

소프트웨어 개발의 생산성과 개발된 제품의 품질문제가 소프트웨어 위기현상으로 대두되면서, 객체지향기술은 소프트웨어 위기 문제를 해결하기 위해 추상화, 캡슐화, 상속성 등의 개념을 기반으로 확장성과 재사용성을 높이는 핵심기술로 부상하고 있다. 객체지향 기술은 소프트웨어를 효율적으로 개발하고 소프트웨어 품질을 높여 유지보수 노력을 최소화하는 소프트웨어 개발 기술로서, 실세계를 소프트웨어로 구현하는 과정에서 실세계의 모델을 소프트웨어에서 객체들의 집합으로 구성하는 것이다.

2.2 전력품질 분석 소프트웨어 객체지향 분석, 설계 (Analysis and Design)

먼저, 전력품질 도메인 분석결과를 토대로 용어 및 사양 등을 정의하고 Use-Case Model(사용사례 모델)을 도출하는 단계를 통하여 요구사항을 정리한다. 그리고 분석단계에서는 요구사항 결과물을 토대로 전력품질 분석을 위한 Key Abstraction(주요정보)을 정의하고 문제이해를 위한 개념적인 분석이 수행된다. 설계단계에서는 분석단계에서의 개념적인 결과를 실제 코딩수준의 세부적인 모델(객체) 도출 및 모델간의 관계를 표현하는 과정이 수행된다.

2.3 Use-Case Model - 요구사항 정의

Use-Case Model은 소프트웨어 기능 중심의 요구사항을 표현하는 모델로서, Use-Case Model을 통해 사용자와 개발자와의 의사소통 및 소프트웨어의 기능을 명확히 정의할 수 있다.

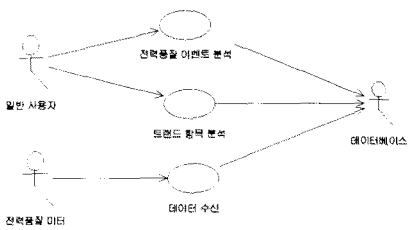

그림 1. 사용사례 모델

그림 1은 전력품질 분석 소프트웨어의 Use-Case Model 일부를 나타낸 것으로, 사용자는 소프트웨어를 통해 이벤트 분석 및 트랜드 분석을 수행하고, 전력품질 미터로부터 데이터를 수신하여 데이터베이스에 저장하는 3가지 Use-Case를 볼 수 있다.

2.4 분석(Analysis)

분석단계에서는 소프트웨어에서 관리하는 Key Abstraction을 정의하고 각 Use-Case에 대한 Realization을 수행한다.

2.4.1 Key Abstraction

Key Abstraction은 도메인을 표현하기 위한 주요정보를 정의하는 것으로, 도메인에서 용어나 사용자 요구사항으로부터 도출된다. 분석/설계가 진행되는 각 단계별 추가 및 배제가 가능하므로 초기에 너무 명확히 정의할 필요는 없다. 그림 2는 전력품질 분석 도메인의 Key Abstraction 중 일부를 나타낸 그림으로, Key Abstraction은 분석/설계 과정에서 클래스와 1: 1 연관될 확률이 높다.
그림 2에서와 같이 도메인을 표현할 수 있는 주요 키워드(Key-Word)를 정의하는데, PQEvent, VolCurData와 같이 간단히 용어만을 표현할 수 있고, 쉽게 속성을 도출할 수 있는 경우 PQMeter와 같이 속성까지 표현할 수 있다.

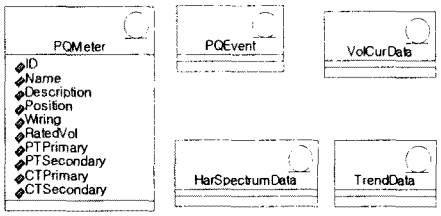

그림 2. Kev Abstraction

2.4.2. Use-Case Realization

Use-Case Model의 각 Use-Case별 Sequence/Collaboration/Class 모델을 통하여 Realization을 수행한다.

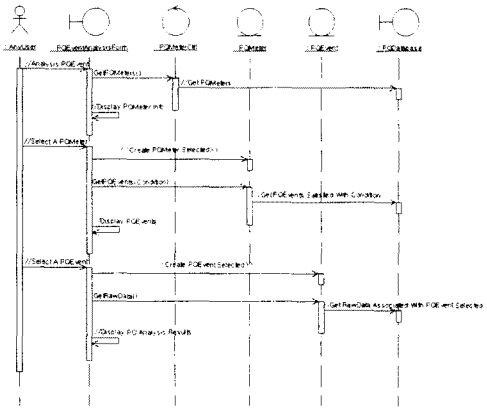

그림 3. Use-Case Realization(Sequence Diagram) - 전력품질 이벤트 분석

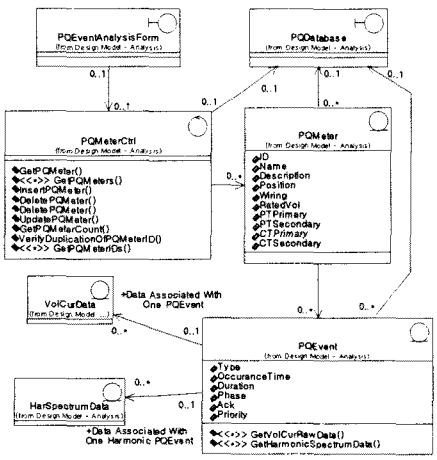

그림 4. Use-Case Realization(Class Diagram) - 전력품질 이벤트 분석(분석)

그림 3과 4는 이벤트 분석을 위한 Sequence Diagram과 Class Diagram을 나타낸 것으로, Sequence Diagram은 흐름의 순서를 표현하고 Class Diagram은 객체 상호간의 관계를 표현한다.

분석단계에서는 세부적인 표현보다는 전체적인 흐름과 객체간의 관계를 표현하는데 중점을 둔다. 그림에서 PQMeter, PQEvent 등의 클래스는 데이터베이스와 연결되어 관련 정보를 읽어오는데 "데이터베이스에 어떻게 접속하여 관련 정보를 읽는지"의 세부적인 내용은 포함하지 않는다. 이러한 세부적인 흐름 및 관계는 설계단계에서 좀더 깊이 다루게 되고 표현되게 된다.

분석단계에서는 각 모델을 통하여 단순히 객체간의 관계성 여부를 표현하는 것이 목적이므로 Class Diagram에서 객체간의 관계 방향성이나 다수성(Cardinality)이 명확하지 않을 경우 단순히 관계(Association)만으로 표현할 수 있다.

2.5 설계(Design)

설계단계에서는 개념적 접근의 분석결과를 토대로 구현수준의 세부적인 설계가 수행되는 단계이다. 따라서 설계단계에서는 단순히 객체간의 관계 뿐 아니라 Architecture, Mechanism 등 소프트웨어 전반적인 내용이 세부적으로 표현되어야 한다. 본 논문에서는 전력품질 이벤트 분석 Use-Case에 대한 세부설계와 데이터베이스로부터 관련 정보를 얻기 위한 메커니즘(Mechanism)을 소개하고, Architecture의 하나인 Layer에 대해 간단히 소개한다.

2.5.1 Architecture - Layer

그림 5에서와 같이 전력품질 분석 소프트웨어의 Layer는 Application, Business, Middle Layer의 3 Layer로 구성하였다. 일반적으로 Application Layer에서는 애플리케이션 도메인에 한정된 서비스 등이 포함되며, Business Layer에서는 Application Layer보다 구체적으로 추상화된, 즉 재사용 가능성이 있는 컴포넌트 등이 포함된다. 그리고 Middle Layer에서는 데이터베이스 인터페이스나 플랫폼(Platform)에 의존적인 OS 서비스 등이 포함된다. 전력품질 분석 소프트웨어의 Business Layer에서는 전력품질 미터와 통신, RDBMS 인터페이스 등이 포함된다.

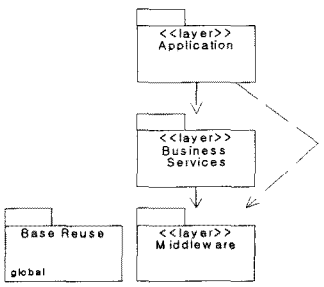

그림 5. Architecture - Layer

2.5.2 Persistency Mechanism

데이터베이스와 연관된 대표적인 메커니즘인 Persistency Mechanism은 DAO(Data Access Object)를 사용할 경우 다음과 같이 설계할 수 있다.

그림 6에서, 데이터베이스와 연계하여 관련 데이터를 얻는 클라이언트 클래스-여기에서 PersistencyClient-는 DBClass를 통하여 데이터를 얻고, DBClass는 여러 DAO 클래스와 연관되어 데이터베이스로부터 관련 데이터를 읽는다.

PersistencyClientClassList 클래스는 데이터베이스의 여러 레코드 정보를 리스트로 관리하기 위한 리스트 클래스이며, PersistencyClass는 데이터베이스의 각 레코드 정보를 갖는 클래스 이다. 따라서 전력품질 분석 소프트

웨어에서 전력품질 이벤트 정보를 데이터베이스로부터 읽어오기 위한 클래스에 Persistency Mechanism을 적용하면 그림 7과 같다.

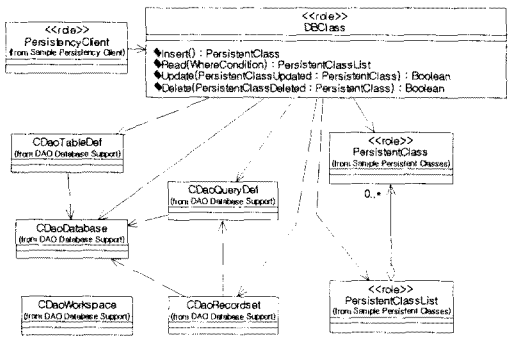

그림 6. RDBMS - DAO/Persistency

그림 7은 CPQMeter 클래스를 통해 데이터베이스로부터 이벤트 정보(CPQEvent)를 얻는 메커니즘으로, CPQMeter 클래스가 PersistencyClient 이며, CDaoRecordset 클래스를 통해 데이터베이스로부터 읽어들인 각각의 이벤트 정보는 CPQEvent 클래스에 저장되고, CBiList 클래스를 통해 여러 이벤트 정보-CPQEvent 클래스-가 관리된다.

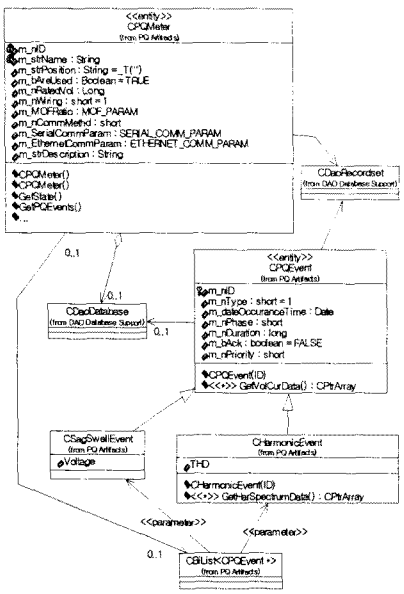

그림 7. 전력품질 이벤트 정보를 얻기 위한 Class Diagram

분석단계(그림 4)에서 PQEvent 클래스는 설계단계에서 CSagSwellEvent 클래스와 CHarmonicEvent 클래스, CPQEvent 클래스로 세분화된다. 즉, 발생시간, 지연시간 등의 이벤트 공통속성은 CPQEvent 클래스에서 정의되며 Sag, Swell 이벤트의 크기(전압 크기) 및 고조파 종합 왜곡율(THD)은 상이한 데이터 속성이므로 CSagSwellEvent 클래스와 CHarmonicEvent 클래스 속성으로 정의하고 CPQEvent 클래스를 상속받도록 한다.

2.5.3 Use-Case Realization

그림 4에서와 같이 분석단계에서 전력품질 이벤트 분석을 위한 Use-Case Realization은 설계단계에서는 Persistency Mechanism을 포함하여 최종적으로 그림 8과 같이 세부적으로 표현된다.

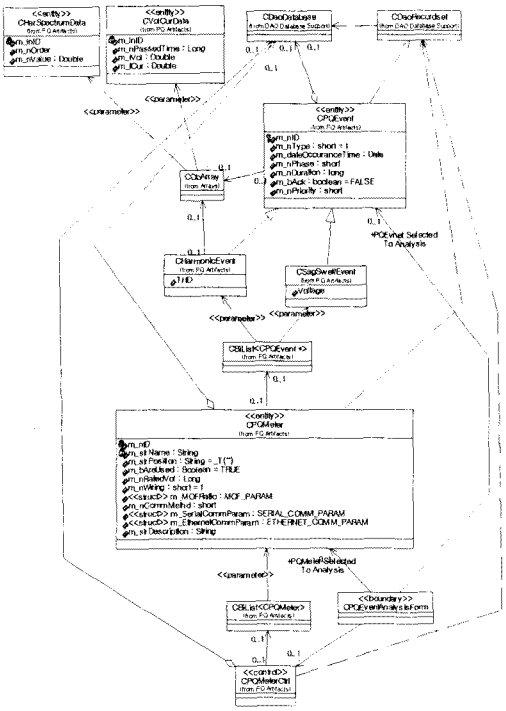

그림 8. Use-Case Realization(Class Diagram) - 전력품질 이벤트 분석(설계)

3. 결 론

전력품질 분석 소프트웨어 설계 중 전력품질 이벤트 분석기능 중심으로 요구사항, 분석, 설계 단계별 일부 내용을 나열하였다. 설계단계에서는 데이터베이스와 연계한 메커니즘과 Layer를 포함하였고, 각 단계별 모델링은 UML로 표현되었다.

UML을 이용한 체계적인 소프트웨어 설계는 소프트웨어 위기를 극복할 수 있는 대안으로 소프트웨어 개발의 중심을 이루고 있는데, 이상에서 소개된 내용은 체계적인 분석, 설계를 통하여 소프트웨어 품질을 높이고 유지보수 노력을 최소화하기 위해 전력품질 분석 소프트웨어 개발에 적용된 일부 내용을 소개한 것이지만, 최종적으로는 소프트웨어 개발에 있어 체계적인 분석, 설계의 중요성을 소개하기 위한 것이다.

[참 고 문 헌]

[1] Mastering Object-Oriented Analysis and Design with UML - Rational University
[2] The Unified Modeling Language User Guide - Addison Wesley