'96 춘계학술발표회 논문집
한국원자력학회

CASE Tool을 이용한 Safety Critical 소프트웨어 개발 방법론

김장열, 권기춘
한국원자력연구소

요 약

본 논문은 Computer Aided Software Engineering (CASE) Tool을 이용할 경우의 Safety Critical 소프트웨어 개발 방법론인 구조적 분석 및 구조적 설계 모델링 방법론을 Teamwork CASE tool의 예를 중심으로 제안하고자 한다. 제시된 사례는 NSIS(Nuclear Safety Information System)으로서 Essential Modeling과 Implementation Modeling을 제시하였는데 Teamwork CASE 환경하에서의 분석 및 설계 절차, 지침 등을 제시하였다. Essential Modeling에서는 NSIS의 MMIS 분석범위 및 External Interface를 제시하는 환경 모델(Environmental Model)과 MMIS의 기능을 계층구조적으로 분할하는 행위모델링(Behaviroal Modeling)을 각각 Context Diagram과 Data Flow Diagram (DFD)으로 그 과정을 제시하였다. Implementation Modeling에서는 Essential Modeling으로 부터 나온 결과물을 토대로 Boss Rule, Transform Rule과 Transaction Rule 등을 거쳐 NSIS MMIS의 설계 근간이 되는 Structured Chart(SC)를 제시하였다.
본 논문에서 제시된 모델링 방법론을 통하여 Safety Critical 소프트웨어 개발시 Teamwork CASE Tool을 활용할 수 있음과 동시에 분석 및 설계의 일치성을 통하여 Safety Critical 소프트웨어의 안전성 확립과 품질보증 목표에 기여할 수 있다.

1. 서 론

본 논문에서는 Teamwork CASE Tool을 이용한 Safety Critical 소프트웨어 개발방법론의 하나인 구조적 분석 및 구조적 설계 모델링 방법론을 제시한다. 일반적으로 소프트웨어 엔지니어링 절차는 크게 Essential Modeling과 Implementation Modeling 으로 나눌 수 있다. Essential Modeling은 분석과정의 모델링 방법이며 Implementation Modeling은 설계과정의 모델링이다. 요구사항명세서 작성을 위한 Essential Modeling은 크게 3가지 관점으로 볼 수 있다. 즉, 정보흐름의 관점, 환경의 관점, 행위의 관점이다. 설계사양서 작성을 위한 Implementation Modeling은 모듈 형태의 Struictured Chart의 관점에서 볼 수 있다.
Essential Modeling이나 Implementation Modeling에 앞서 모델링해야 될 것은 Information Modeling인데 이는 다른 용어로 데이터 모델링이라고도 한다. Information Modeling에서는 NSIS의 MMIS data의 관계 및 속성을 정의하는 과정으로서 Entity Relationship Diagram(ERD) 형태로 표현한다. 여기서는 MMIS data를 Data Dictionary Entry (DDE) operator를 사용하여 후보대상이 되는 Entity와 그에따른 Attribute를 정의한다. 이러한 ERD 모델링이 끝나면 MMIS의 Event Response List (ERL)을 작성한다. 그다음 과정이 Context Diagram을 이용하여 Environmental Modeling을 하는데 Context Diagram은 ERL을 토대로 작성한다.
두 번째 상세한 분석과정인 Bebavioral Modeling은 Context Diagram으로 부터 level down하면서 application logic, data processing, data dependencies, intermediate data, application의 분할 등의 과정을 거쳐 더 이상 프로세스를 분할할 수 없을때 까지 기능을 분할한다.
이때 프로세스가 leaf process가 될 때 peocess spec(또는 mini spec)을 작성한다.
이와 같이 하여 Essential Modeling 과정이 끝나면 Essential Modeling 자료를 설계자 관점에서

분석자료 할당(boss, subgrouping, transform 및 transaction) 과정을 거쳐 Implementation Modeling 산물인 Structure Chart를 작성하게 된다.

2. Information Modeling

Information Modeling은 Teamwork CASE 환경 모델링의 첫 번째 과정으로서 ERD를 이용하여 Entity와 Attribute를 정의한다. 예를들어 원자력발전소와 압력센서 사이의 관계를 ERD로 모델링한다면 발전소와 센서의 관계는 1 대 N의 관계를 가질 수 있으며 발전소인 NPP라는 Entity는 plant_no + model_type + construction_date + life_span 으로 그 속성을 정의할 수 있으며 sensor_detector는 sensor_id+location+date_installed의 속성을 가진다. 이들 두 Entity사이의 포함관계를 종합하면 그림 1과 같다. 그림1에서 @표시는 키워드를 의미한다.

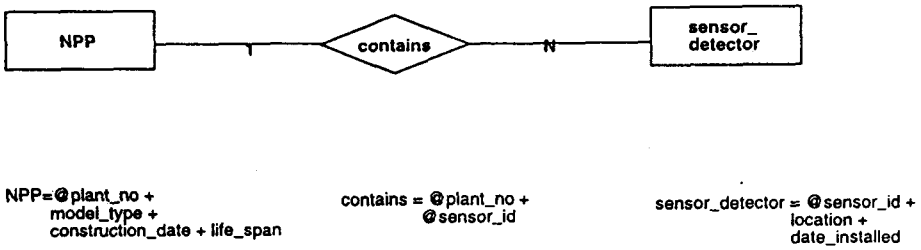

그림 1. NPP와 pressure_detector 사이의 Information Modeling

이외 NPP, registration, mechnic, sensor_detector 관계를 ERD로 모델링한 예는 그림 2와 같다.

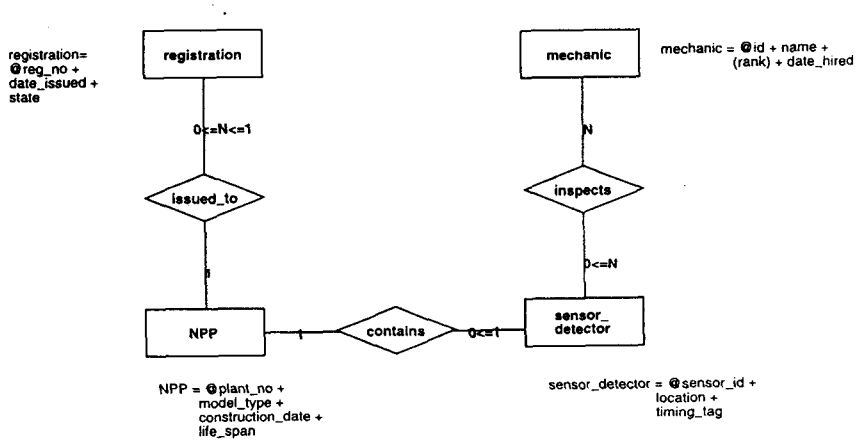

그림 2. MMIS에 있어서 ERD Cardinality

3. Environmental Modeling

Environmental Modeling은 논리적으로 분석대상을 overview할 수 있게 하는 것으로서 분석대상 범위와 주변환경 및 목표시스템(NSIS)이 가져야 할 성능 및 제약조건 등을 도식화 한다.
예를들면 여러 가지 MMIS 소프트웨어중(Information Processing System, Safety Parameter Display System, Post Accident Monitoring Instrumentation, Process Component Control System 등)의 분석범위를 정하게 된다. NSIS가 위와같은 MMIS 소프트웨어를 포괄적으로 포함하는 종합안전정보시스템이라고 가정하면 Environmental Modeling은 그림 3과 같이 정의할 수 있다.

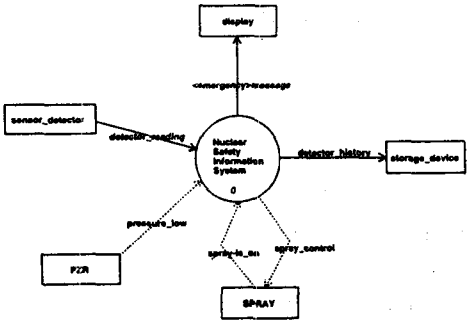

그림 3. NSIS의 Context Diagram

그림 3의 Environmental Modeling 과정을 요약하면 다음과 같다.

① ERL을 토대로 Context Diagram을 작성한다.

② 분석대상 프로세스의 명칭(NSIS)을 부여하고 주변환경이 되는 Terminator를 정의한다.

③ 정의한 Terminator에 대해서는 Input, Output, Body 부분을 기술한다.

④ ①-③ 번의 과정을 ERL 사건목록을 토대로 반복하면서 NSIS의 Context Diagram을 완성한다.

NSIS Context Diagram 작성시의 지침사항은 NSIS 는 최소한 하나의 input과 하나의 output을 가져야만 한다. Terminator는 오직 하나의 input 또는 하나의 output 또는 양방향의 data flow 또는 control flow를 가질 수 있다. 또한 Terminator와 Terminator 사이에는 data 또는 control flow를 그려서는 안되며 Context Diagram상에 data store 기호를 사용해서도 안된다.

4. Behavioral Modeling

Behavioral Modeling은 Essental Modeling의 마지막 과정으로서 Data Flow Diagram 을 이용하여 상세한 기능을 분석하게 된다. NSIS중 Data Acquisition 부분의 DFD를 예로 보면 센서로부터 압력값을 읽어 임계값(Threshold value)과 비교하여 임계값을 넘으면 spray를 on시켜 살수계통시스템을 작동시키도록하고 임계값을 넘지는 않으나 경고 메시지를 주어야 할 경우 Emergency message를 주며 센서로 부터 읽은 모든 데이터는 history 파일로 저장한다.

이를 DFD로 모델링한 것은 그림 4와 같다.

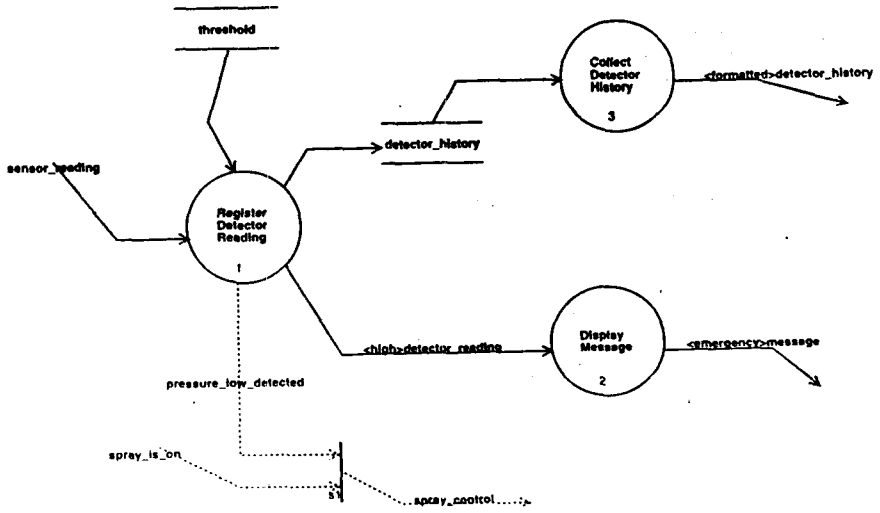

그림 4. NSIS의 BM

그림 4에서 예로든 Behavioral Modeling시의 지침 사항을 요약해 보면 다음과 같다.

① 모든 프로세스는 DFD로 표현되거나 P-spec 형태로 분할될 수 있어야 한다.

② 부모 DFD와 자식의 DFD는 상호 match를 이루어야한다. (즉, 상위 DFD에서의 input 및 output flow와 하위단계에서의 input 및 output flow가 일치하여야 한다.)

③ 모든 data flow와 data store는 primitive level까지 정의하여야 한다.

④ control flow의 경우, C-spec을 작성할 때 Decision Table(DT) 또는 STD(State Transition Tanle) 로 표현해야 한다.

5. Structured Design

Implementation Modeling에서는 overview부터 detail한 부분까지 top-down 형식으로 top level module(super class), class modules(subclass module), leaf module 순으로 structuted chart를 작성해 나가는데 앞서 분석단계의 DFD 분석자료를 토대로 boss rule, transform rule, transaction rule 등을 적용하여 설계하게 된다. 분석된 자료를 토대로 모듈을 나누고 할당할 때 safety 개념과 defense-in-depth 개념을 고려하여 적용한다.

NSIS의 모니터링 시스템을 예로 든다면 그림 5와 같이 초기화 하는 부문, 센서로 부터 자료를 취득하는 부문, 취득한 자료값이 임계값을 넘었을 경우 경보를 생성하는 부분, 수집된 자료를 저장하는 부분 등으로 모듈을 나눌 수 있다.

일반적으로 모듈을 설계할 때 소프트웨어 공학의 관점에서 모듈내의 cohension은 높게 모듈간의 coupling은 가능한한 낮게하여 설계한다. 설계시 사용되는 데이터들은 분석단계에서 이미 분석한 DDE(Data Dictionary Entry)를 사용하기 때문에 재정의할 필요는 없다.

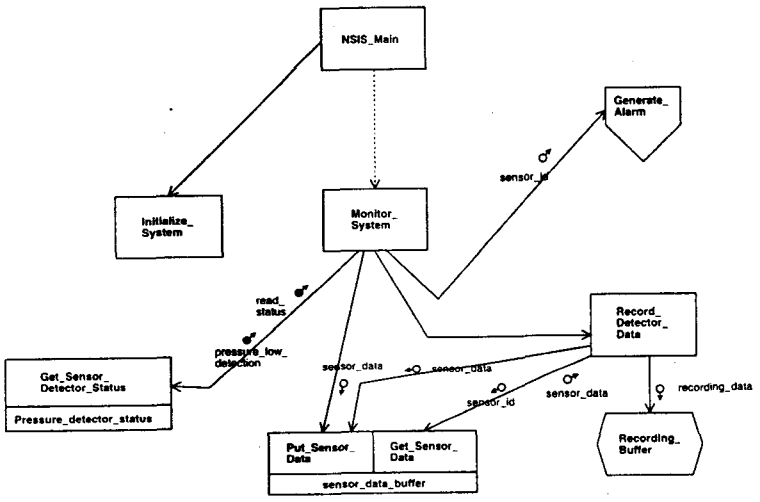

그림 5. NSIS 모니터링 시스템의 Structured Chart

6. 결 론

본 논문에서는 Teamwork CASE Tool을 이용하여 Safety Critical 소프트웨어를 개발할 경우의 모델링 방법론을 NSIS의 예를 중심으로 기술하였다. 본 논문에서 제시된 모델링 절차는 Information Modeling에서 Event Response List(ERL), Essential Modeling인 Environmental Modeling과 Behavioral Modeling 그리고 Implementation Modeling 까지의 과정을 NSIS 시스템의 일부분을 하나의 예로들어 Safety Critical 소프트웨어 모델링 과정을 제시하였다. 제시된 모델링은 대부분의 CASE tool을 이용할 경우 유사하게 적용할 수 있다. 이와같이 제시된 방법론으로 Safety Critical 소프트웨어를 모델링할 경우 분석과 설계 전과정의 Common database화로 일치성 확인 과 설계자료의 재사용성 등 자동화된 tool에 의한 품질보증이 가능하여 소프트웨어의 안

전성 측면을 향상시킬 수 있음과 동시에 정형화된 모델링 기법을 사용하기 때문에 계측제어 소프트웨어의 확인 및 검증 업무에 기여할 수 있다.

[참고문헌]

1. Demarco, T., Structured Analysis and System Specification, Yourdon Press, NY, 1978
2. IEEE, IEEE Transactions on Software Engineering, SE-3,1 (January 1977), (Entire issue devoted to Structured Analysis)
3. Cadre Technologies Inc., Requirements Analysis with Teamwork Release 6.0, June 1994
4. Allworth, S. T., Introduction to Real Time Software Design, Springer-Verlag, 1981
5. Gomaa, H., "Software Development of Real-Time Systems," Communications of the ACM, 29, 7 (July 1986), PP 657-668.