소프트웨어 컴포넌트 관련 표준 개발 사례

전인걸 | TTA S/W 개발기술연구반 간사, 한국전자통신연구원 컴퓨터 · 소프트웨어연구소
장진호 | TTA S/W 응용기술연구반 의장, 한국전자통신연구원 컴퓨터 · 소프트웨어연구소
박창순 | TTA S/W 기술위원회 의장, 한국전자통신연구원 컴퓨터 · 소프트웨어연구소

1. 컴포넌트에 기반한 시스템 개발이란?

IT 관련 기술이 다양해지고 정보시스템의 규모와 복잡도가 증가함에 따라 이를 해결하기 위한 방법으로 컴포넌트 기반 개발(CBD : Component Based Development)이 대두되고 있다. 컴포넌트 기반 개발은 기존 개발 방식과는 달리 특정 프레임워크 상에서 실행되는 부품화된 컴포넌트를 바탕으로 이를 조립하여 더 큰 컴포넌트를 만들거나 애플리케이션을 개발하는 새로운 기법이다. 이러한 컴포넌트 기반 개발은 과거 구조적 방법이나 객체지향 기술이 제대로 해결하지 못한 개발 생산성, 소프트웨어 재사용성, 시스템 유지보수성을 향상시킬 수 있는 대안으로 주목받고 있으며, 이외에도 요구사항 획득 및 다른 소프트웨어의 생산, 납기 지연, 비용 초과 등 소프트웨어 위기를 초래한 고질적인 문제들을 해결할 수 있는 방안으로 각광받고 있다. 이러한 컴포넌트 관련 기술은 EJB, .NET, CORBA 등과 같은 컴포넌트 프레임워크 표준이 발표되고 이를 기반으로 관련업계에서도 다양한 솔루션과 개발 도구들을 출시함에 따라 폭넓게 확산되고 있다.

일반적으로 알려진 컴포넌트 기반 개발 방법의 장점은 다음과 같다. 첫째, 복잡한 개발 문제를 적당한 구현 단위로 분해하고 각 단위별로 개발하여 조립하는 방식으로 애플리케이션을 개발할 수 있어 개발 생산성을 높일 수 있다. 이는 컴포넌트의 경우 모든 접근이 인터페이스를 통해서만 이루어지므로 독립적인 개발과 배포가 가능하다는 특성에서 유래한다. 따라서 사용자들은 필요한 컴포넌트를 직접 개발하지 않고 요구사항에 부합하는 컴포넌트를 시장에서 구입하여 활용할 수도 있다. 기존에 개발되어 있는 컴포넌트를 이용할 경우 개발 생산성을 극대화할 수 있다. 둘째, 컴포넌트가 제공하는 완벽한 캡슐화 기능을 통해 변화 또는 에러의 영향을 해당 컴포넌트 내부로 한정시킬 수 있어 유지보수성이 높다. 예를 들면, 업무처리 방식이 바뀌어 비즈니스 로직이 달라질 경우, 전체 시스템을 재개발하지 않고 해당 컴포넌트를 수정하거나 새로운 항목으로 대체함으로써 이러한 변화를 수용할 수 있다. 셋째, 성능 극대화를 위해 개발된 애플리케이션을 시스템 · 네트워크에 최적으로 분산, 배치할 수 있는 단위로 나눌 수 있어 시스템의 업사이징(upsizing) 또는 다운사이징(downsizing)에 신속하게 대응할 수 있다. 이를 통해 새로운 기능 추가, 성능 개선, 시스템 스케일의 변화 등에 손쉽게 대처할 수 있다. 넷째, 표준화된 아키텍처를 준수하므로 내부 설계나 코드와 같은 컴포넌트 구현부를 더 나은 성능을 제공하는 다른 컴포넌트로 손쉽게 교체할 수 있다.

그러나 그간의 적용 경험과 여러 연구에서 지적된 바와 같이 이러한 컴포넌트 기반 개발의 장점을 살리고 널리 보급하기 위해서는 컴포넌트를 이용한 시스템 개발의 단순한 요소 기술이 아니라 시스템 개발 환경

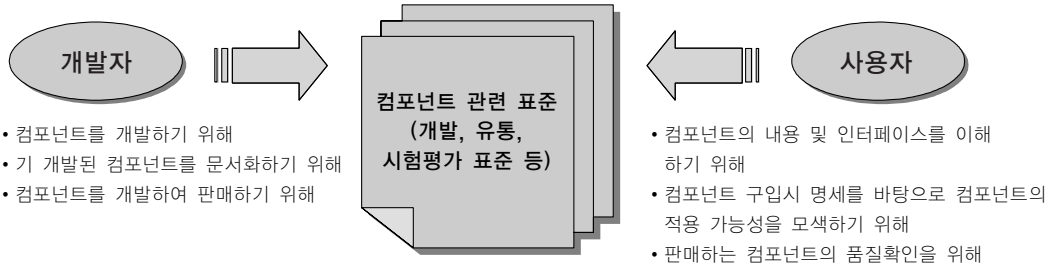
전반을 포괄하는 기반 환경으로 이해하고 이를 지원할 수 있는 체제를 구축해야 한다. 특히 컴포넌트의 특성상 다양한 개발 방식이 공존하게 되므로 이들간 최소한의 공통적인 부분을 유지할 수 있게 해주는 컴포넌트 관련 표준의 제정이 시급한 실정이다.

2. 컴포넌트 관련 표준 개발의 필요성

컴포넌트란 하나 이상의 기능을 가진 독립적인 소프트웨어이며, 조립을 통해 응용 프로그램을 생산할 수 있는 규격화된 부품을 의미한다. 이러한 컴포넌트는 대개의 경우 EJB, COM 등의 프레임워크 표준에 따라 만들어진 바이너리 코드, 즉 구현 컴포넌트만을 지칭하였지만 최근에는 개발 과정에서 만들어진 재사용 가능한 모든 산출물을 포괄하는 개념으로 확장되었다. 즉, 분석 및 설계 과정에서 만들어지는 문서 형태의 중간 산출물 역시 재사용의 대상이며, 경우에 따라서는 그 효과가 훨씬 크고 근본적일 수도 있기 때문이다. 이러한 컴포넌트 기반 개발은 기존의 소프트웨어 개발 방식과는 달리 부품화된 컴포넌트가 불특정 다수에 의해 사용되므로 목적이 서로 다른 다양한 주체들 간의 이해관계로 확장될 수 밖에 없다. 따라서 무엇보다 완성된 컴포넌트의 품질 보증을 위한 방안을 마련하고, 관련 주체들이 개발된 컴포넌트의 내용과 사용법을 정확하게 이해할 수 있도록 표준화된 명세를 제공하는 것이 매우 중요하다. 이를 위해서는 아래의 그림과 같이 구현 컴포넌트를 명세하기 위한 표준 뿐만 아니라 개발 및 유통 과정에서 적용할 수 있는 다양한 표준이 필요하다.

TTA 산하 IT S/W기술위원회(TC09)에서는 컴포넌트 기반 개발을 지원하기 위한 다양한 정책적 지원과 표준 개발을 위한 작업이 진행되고 있다. 그 결과 컴포넌트 유통 명세 표준, 개발 명세 표준, 시험평가 표준 등이 만들어지고 있으며, 이러한 컴포넌트 관련 표준은 다음과 같은 요건을 만족하고 있다.

- 방법론 및 프로세스 독립적 : 컴포넌트 관련 표준은 특정 방법론, 기법, 도구에 종속되지 않아야 한다. 이를 위해 실무적으로 활용되고 있는 다양한 방법론, 기법, 산출물의 내용을 비교 · 분석하여 공통적으로 언급하고 있는 요소들을 중심으로 명세 항목과 기술 방식을 정의하였다.
- 컴포넌트 개발, 유통, 시험평가를 위한 최소한의 산출물 : 컴포넌트 관련 표준의 내용이 복잡하여 개발자들에게 부담을 초래하는 역효과가 발생하지 않도록 컴포넌트의 내용을 이해하는데 필요한 최소한의 정보를 중심으로 필수요소를 정의하고, 기타 중요도가 떨어지거나 부차적인 항목들은 과감하게 선택요소로 정의하였다. 따라서 표준의 구성 항목은 컴포넌트 개발자와 사용자 입장에서 컴



[그림] 이해 당사자에 따른 컴포넌트 관련 표준의 필요성

포넌트를 개발하거나 활용하는데 필수불가결한 정보가 중심이 된다.

- 컴포넌트를 개발하고 유통하는데 필요한 정보를 충분히 포함 : 타사가 개발한 컴포넌트를 이용하여 시스템을 개발하거나 커스터마이징해야 하는 사용자 입장에서는 컴포넌트에 대해 가급적이면 다양하고 정확한 정보가 필요하다. 이는 위에서 지적한 최소 정보요건과는 상충된다. 따라서 컴포넌트를 이용하기 위해 반드시 필요한 정보가 간과되지 않으면서도 이러한 상충관계가 발생하지 않도록 실태조사와 전문가들의 의견을 반영하는 과정을 거쳐서 그 결과를 표준에 반영하였다.
- UML 표기법의 도입 : 컴포넌트 관련 표준을 이용하는 경우 산출물간의 호환성 및 적용가능성을 높이기 위해 대부분의 객체지향 및 컴포넌트 기반 개발 방법론에서 표준으로 채택하고 있는 UML 표기법을 준수하여 표준을 정의하였다. 이를 통해 시장에 출시된 다양한 지원 도구를 사용하여 산출물을 손쉽게 작성하고, 결과물을 용이하게 이해할 수 있다.
- 적은 시간과 노력으로 산출물에 대한 명세가 가능 : 산출물을 작성하는 데 너무 많은 시간과 노력이 소요된다면 표준을 적용하는 데 장애가 될 것이다. 따라서 이러한 문제가 발생하지 않도록 항목의 표현 방식을 텍스트 또는 그래픽으로 한정하고 두 가지 모두 가능할 경우에는 상대적으로 적용하기 쉬운 방식을 표준으로 정의하였다.

3. 소프트웨어 컴포넌트 관련 표준

3.1 컴포넌트 개발 명세 표준

컴포넌트는 구현과 명세가 철저하게 분리되고 인터페이스의 역할을 강조함으로써 데이터와 업무 기능의 통합, 캡슐화, 식별성과 같은 객체지향의 기본 원칙을 강화하고 있다. 이러한 컴포넌트의 개념을 정보시스템 개발에 적용하기 위해서는 사용자 입장에서 해당 서비스가 어떤 서비스를 어떤 방식으로 제공하는지를 이해해야 하고 개발자 입장에서 이를 위해 어떤 항목을 어떠한 방식으로 문서화해야 하는지를 잘 알고 있어야 한다. 이를 위해서는 컴포넌트 개발 명세에 관한 표준이 필수적이다. 왜냐하면 컴포넌트를 조립 또는 사용해야 하는 사용자 입장에서는 컴포넌트 구현에 관한 세부사항보다는 컴포넌트 자체를 설명하는 명세가 보다 중요하기 때문이다. 이는 시스템 전체에 영향을 주지 않으면서 동일한 명세를 제공하는 다른 컴포넌트로 손쉽게 교체할 수 있어야 한다는 컴포넌트 기반 개발의 특성에서 기인한다.

소프트웨어 컴포넌트 개발 명세 표준은 다음과 같은 특징을 가지고 있다.

- 기존 컴포넌트 기반 개발 방법론에서 사용하고 있는 기법과 산출물에 대한 분석결과를 바탕으로 표준화된 명세 방법을 제시한다.
- 컴포넌트 명세를 위한 기존의 접근방법은 재사용 잠재력이 월등히 높은 분석 및 설계 단계의 중간 산출물은 배제하고, 구현 컴포넌트에 대한 인터페이스를 명세하는데 초점을 맞추었다. 본 표준은 이러한 문제점을 해결할 수 있도록 인터페이스에 대한 명세 뿐만 아니라 컴포넌트 기반 개발의 전체 공정에서 작성해야 하는 주요 산출물과 이에 대한 명세 항목을 제시한다.
- 컴포넌트 기반 개발의 주요한 장점 중의 하나는 이미 만들어진 컴포넌트들을 조립 또는 합성하는 방법으로 시스템을 구현할 수 있다는 것이다. 이를 위해서는 컴포넌트에 대한 명세가 필수적인데,

본 표준에서는 사용자들이 컴포넌트 구현에 관한 세부사항을 손쉽게 이해할 수 있도록 표준화된 기술 체계를 제공한다.

이 표준은 컴포넌트 개발과 관련한 제반요소를 상세하게 규정하고 있는 방법론 형식의 개발을 위한 표준이 아니라 사용자의 입장을 고려하여 개발자들이 컴포넌트를 개발할 때 어떤 산출물을 어떠한 내용과 형식으로 작성하는 것이 바람직한지를 규정하고 있는 명세에 관한 표준이다. 소프트웨어 컴포넌트 개발 명세 표준은 구현 컴포넌트에 대한 인터페이스나 오퍼레이션과 등과 같은 세부사항을 묘사하기 위한 것으로 컴포넌트를 개발하는 과정에서 개발자들이 필수적으로 작성해야 하는 산출물을 핵심 산출물 위주로 정의하였다. 이는 구현 컴포넌트를 이용한 조립 및 유통을 위해서는 개발 절차보다는 개발 과정에서 작성해야 하는 산출물과 이를 작성하는 방법을 이해하는 것이 보다 중요하기 때문이다. 따라서 본 표준의 내용을 참조하여 컴포넌트 분석, 설계 및 구현 단계에서 반드시 작성해야 하는 표준 산출물과 이에 대한 명세 방안을 참조할 수 있다.

소프트웨어 컴포넌트 개발 명세 표준은 기존 CBD 방법론에 대한 분석 결과를 토대로 다음과 같은 다섯 가지 분야로 나누어, 이를 토대로 컴포넌트 명세를 위해 반드시 기술해야 하는 필수명세항목과 선택명세항목을 도출하고 이에 대한 구체적인 기술 방안을 정의하였다.

- 요구사항 정의(Requirements Specification) : 요구분석의 목적은 시스템이 무엇을 해야 하는지에 대한 관계자들의 이해와 의견을 일치시키고 시스템 개발자가 시스템을 제대로 이해할 수 있게 하며, 시스템의 범위를 결정하고 제약 사항을 파악하여 향후 시험평가를 위한 기준선으로 사용하기 위한 것이다. 요구사항 정의는 사용자와의 면담, 설문, 워크숍 등을 통해 컴포넌트로 구현할 시스템에 대한 개략적인 요구사항을 도출한 후, 이를 기술하기 위해 사용된다.
- 유즈 케이스 모형 기술(Use Case Model Description) : 컴포넌트 개발 초기 단계에서 개발자들은 구현하고자 하는 도메인에 대한 이해가 필수적이다. 이를 위해 개발자들은 요구 분석결과를 토대로 행위자를 식별한다. 그리고 행위자간의 행위 모델을 토대로 유즈 케이스를 정의하고, 이들 간의 관계를 결정한다. 이처럼 구현 대상 컴포넌트가 제공하는 기능을 행위자와 유즈 케이스 간의 관계로 묘사한 것이 바로 유즈 케이스 모델이다. 본 표준에서 유즈 케이스 모델은 컴포넌트로 구축하고자 하는 대상 혹은 도메인의 개념을 명세하기 위해 사용된다.
- 클래스 모델 기술(Class Model Description) : 컴포넌트는 여러 클래스와 객체로 이루어져 있다. 클래스 모델은 클래스 내부의 정적 구조를 표현하기 위한 것으로 객체지향기법에서 가장 중요한 산출물이다. 클래스는 공통 속성, 공통 행위, 객체들 간의 관계, 그리고 공통 객체 그룹에 대한 개념적 또는 구현된 실체를 의미한다. 클래스는 이전 단계에서 작성된 유즈 케이스 모델을 토대로 정의한다. 실제 클래스 모델은 복잡한 절차를 거쳐 작성되지만 본 명세 표준에서는 세부 절차에 대해서는 언급하지 않고 이의 구성요소만을 정의한다. 본 표준에서 클래스 모델은 컴포넌트 내부의 구성요소와 이들 간의 관계를 명세하기 위해 사용된다.
- 컴포넌트 아키텍처 명세(Component Architecture Specification) : 컴포넌트 아키텍처는 대상 컴포넌트의 구조를 상세화하고, 분석 관점의 모델을 구현 가능한 수준으로 발전시키기 위해 사용된다. 본 표준에서 컴포넌트 아키텍처는

개별적인 뷰로 표현된다. 이는 소프트웨어 컴포넌트를 관점에 따라 5개의 뷰, 즉 유즈 케이스 뷰(use-case view)를 중심으로 논리 뷰(logical view), 컴포넌트 뷰(component view), 프로세스 뷰(process view), 배포 뷰(deployment view)로 구분한다. 이를 "4+1 뷰" 아키텍처 모델(4+1 view architecture model)이라 한다. 본 표준에서 컴포넌트 아키텍처 정의서는 컴포넌트의 구조, 이들 간의 상호작용, 배치, 그리고 구현에 관한 제약조건을 명세하기 위해 사용된다.

3.2 컴포넌트 유통 명세 표준

컴포넌트 유통 명세 표준은 소프트웨어 컴포넌트의 유통을 효율적으로 지원하기 위하여 각 소프트웨어 개발사들의 개별적인 명세의 표기방식 및 문서의 형태를 통일하여 표준화 한 것이다. 이를 이용하여 개발자와 사용자 모두에게 효율적인 정보의 전달을 할 수 있는 매개가 된다. 현재 정부를 비롯한 각 민간 업체를 중심으로 컴포넌트에 대한 수요가 증가하고 있는데 이처럼 관심이 늘어나고 있는 이유는 그간 발표된 여러 문헌 등을 통해 알려진 바와 같이 CBD가 제공하는 여러 가지 이점, 예를 들면 개발 기간의 단축, 개발 인력 감소, 관리 효율성 향상 등에 기인한 바가 크다. 그러나 이러한 CBD의 이점을 실현하기 위해서는 무엇보다 컴포넌트를 재사용할 수 있는 환경이 구비되어야 한다. 이 중에서도 가장 중요한 요소는 바로 컴포넌트 유통 명세 표준이라고 할 수 있다. 컴포넌트 유통 명세가 표준화되고 이에 따라 컴포넌트가 개발 · 유통될 경우 컴포넌트 사용 환경 및 시장 활성화에 크게 기여할 것이다. 그러나 아직까지는 관련 시장이 충분히 성숙하지 않아 컴포넌트 유통 전문 업체들을 중심으로 별도의 기술체계를 만들어 활용하고 있어 시장 확대에 걸림돌이 되고 있다. 따라서 이해 관계자들이 모두 납득할 수 있는 단일한 유통 명세 체계를 확립하는 것이 시급하다.

유통 명세의 목적은 사용자 입장에서 시장에서 유통되고 있는 구현 컴포넌트의 내용과 인터페이스를 정확하게 이해하는데 필요한 정보를 제공하는 것이다. 따라서 유통 명세 표준은 개발자와 사용자 모두 납득할 수 있는 명세 항목, 기술 방식, 표기법 및 명세 범위를 표준, 지침, 템플릿, 작성 사례 등의 패키지 형식으로 제공하는 것이다.

3.3 컴포넌트 시험평가 표준

컴포넌트 시험평가 표준은 기존의 소프트웨어 품질 평가모델을 기반으로 컴포넌트의 일반적 특성을 적용하여 개발된 컴포넌트의 평가를 위한 표준이다. 컴포넌트 기반 소프트웨어 개발은 재사용에 의한 개발비용 감소의 효과를 거둘 수 있으나, 컴포넌트가 개발된 환경과는 다른 새로운 환경에 적용할 때 발생하는 여러 가지 위험 요소들을 고려하지 않을 수 없다. 또한 소스 코드를 공개하지 않는 컴포넌트의 특성상 성공적인 컴포넌트 기반 소프트웨어 개발을 위해서는 무엇보다도 컴포넌트에 대한 품질보증이 필수적이다. 본 표준의 목적은 컴포넌트를 사용자 입장에서 컴포넌트에 대한 신뢰성을 확보하기 위한 수단으로 기 개발된 컴포넌트의 품질평가를 위한 기준을 정의하는 데 있다. 특히 컴포넌트의 개발 및 유통 과정에서 필수적으로 고려해야 하는 일반적인 기준 요소들을 포함한다. 구체적으로는 각 기준별로 산출물을 평가하기 위한 지표나 배점 방법, 컴포넌트 품질을 인증하기 위해 개발자와 사용자들이 시험해야 하는 요소들을 정의하고 있다.

4. 결론

소프트웨어 개발자는 IT 기술의 급격한 발전으로 인해 빠르게 변화하는 다양한 사용자의 요구를 만족시키기 위해 점점 더 많은 노력을 기울이고 있다. 그러나 모든 요구사항을 만족시킨다는 것은 거의 불가능한 일이 되어 가고 있기 때문에 이를 해결할 수 있는 새로운 개발 패러다임인 컴포넌트 기반 개발이 일반화되고 있는 추세이다. 소프트웨어 개발자는 컴포넌트 기반 개발을 이용하여 기존에 지적되어 온 많은 문제를 해결할 수 있으나, 다양한 컴포넌트 개발 방법이 등장하여 서로간에 경쟁을 하게 됨에 따라 또 다른 문제를 개발자에게 안겨주고 있는 실정이다.

소프트웨어 개발에 있어서는 컴포넌트를 이용한 다양한 방법이 존재 할 수 있다. 그러나 컴포넌트의 기본 개념을 간과하지 말아야 한다. 컴포넌트란 특정 시스템 및 방법에 종속되는 소프트웨어 제품만을 말하는 것이 아니라 개발 과정에서 만들어진 재사용 가능한 모든 부산물을 포함하는 소프트웨어 단위라는 것이다. 따라서 컴포넌트 관련 표준은 어떠한 방법 및 시스템에도 사용될 수 있는 기본적이며 공통적인 내용을 표준화하는 것이다. 이러한 표준은 특정 집단의 이해관계를 대변하는 것이 아니기 때문에, 많은 시스템 설계자 및 개발자가 개발 중인 컴포넌트 관련 표준의 정제에 참여하여 누구나 쉽게 이용할 수 있는 표준이 되도록 해야 한다. 또한 컴포넌트 표준의 필요성을 인식하고 제정된 표준을 사용하여 컴포넌트를 개발 및 이용하여 컴포넌트 기술을 활성화시키기 위한 개개인의 노력이 필요하다.

디지털콘텐츠 유통표준 마련

정부가 디지털콘텐츠에 대한 범국가적 유통표준을 마련키로 했다. 이에 따라 온라인 디지털콘텐츠 유통에 새로운 전기를 마련하고 시장도 활성화할 것이라는 업계의 기대가 높아졌다. 정보통신부는 3월 10일 문화관광부 같은 관련부처가 참여하는 공동협의체를 구성하는 등 디지털콘텐츠 유통기반구축을 위한 추진체계를 정비하는 국가URN(Uniform Resource Names)기반구축 사업을 확정했다고 밝혔다. URN은 인터넷 접근체계인 URL(Uniform Resource Locators)과는 달리 디지털콘텐츠 자체에 상품 바코드와 같은 식별코드를 부착해 온라인상의 콘텐츠 유통에 활용하는 체계다. URL이란 현 방식의 인터넷 자원 식별체계는 정확한 식별기능이 떨어져 콘텐츠 유통에 적합하지 못하다는 지적을 받아 표준화 기구인 MPEG-21과 TV애니타임(Anytime) 등에선 URN을 권장하고 있다. 정통부는 표준식별체계를 정착시키면 콘텐츠 이용자는 등록된 콘텐츠 정보를 통해 원하는 정보를 더욱 빠르고 정확하게 검색, 이용할 수 있으며 콘텐츠 보유자 또는 유통사업자 역시 효율적으로 안전하게 콘텐츠를 유통시킬 수 있다고 설명했다. 공동협의체는 정통부, 문화부 같은 관계부처와 한국전산원, 한국소프트웨어진흥원, 한국문화콘텐츠진흥원, 한국디지털콘텐츠포럼 등 유관기관으로 구성된다. 협의체는 식별체계의 개발과 적용은 물론 거래인증 등 유통서비스, 권리자 보호, 표준화 등을 추진해나갈 계획이다. 정통부 관계자는 "디지털콘텐츠 유통기반이 체계적으로 정착되면 디지털콘텐츠 산업이 더욱 활기를 띠게 되며 장기적으로 우리나라가 국제 유통의 허브 노릇을 하게 될 것으로 기대한다"고 말했다