한국게임학회 논문지
제3권 제2호 2003년 9월

게임엔진 개발 공정의 정형화를 지원하는 컴포넌트 저장소의 설계

송의철
세경대학 멀티미디어과
songec@saekyung.ac.kr

A Component storage Design Supporting formalization of Game Engine Development Process

Eui Cheol Song
saekyung college Dept. of Multimedia

요 약

게임엔진에서 처리하는 속성과 절차가 매우 유사한 많은 게임 소프트웨어 들이 다른 게임에서 참조하거나 재사용하지 않고 새로운 게임을 개발할 때 게임엔진 부분에 대한 중복투자 문제가 발생한다. 특히 현재 게임 소프트웨어 개발사들이 게임을 개발할 때 매우 다양한 소프트웨어 개발 프로세스를 사용하고 있는 것이 중복투자의 주된 문제점 중 하나이다. 그러므로 게임엔진에 대한 프로세스의 표준화가 되어있지 않아 현재 개발 중인 게임 소프트웨어에 다른 소프트웨어 개발과정에서 생성된 산출물을 이해하고 재사용할 수가 없다. 따라서 어느 게임 소프트웨어 개발사가 특정 게임을 개발할 때 다른 게임 소프트웨어와 동일한 게임엔진 처리에 대하여 새롭게 분석하고 설계하는 것이 현재의 게임 소프트웨어가 안고 있는 커다란 문제점이다.

이러한 문제점을 해결하기 위해 본 논문에서는 컴포넌트 기반 개발방법을 적용할 수 있도록 게임엔진 개발에 대한 공정개선, 구조와 관계성 분석, 계층별 모듈별 분류와 조합 방법, 저장소 구현, 프로세서 모형을 제시하였다.

Abstract

There arose problems of double investment about the game engine part when a lot of game software similar to the property and procedure processed in the game engine develop new game without the reference or reuse in the other games. In particular, using various software development processes is one of main problems of double investment when the enterprises for the game software development develop games now. Accordingly, because it does not make standardization of process about the game engine, it does not understand and reuse products created in process of the other software development process in development now. Accordingly, the newly analyzed and designed software was big problems with the present game software about the game engine process similar to the other game software when the enterprises for any game software develop a special game.

For solving these problems, this study is to suggest the process improvement about the game engine development, analysis of structure and relation, classification and combination method by the class and module, implementation of storage, and processor model in order to apply the development method based on the component.

1. 서 론

1.1 연구 배경

게임 소프트웨어 이용자의 사용환경은 매우 빠른 속도로 발전하고 있으나 이러한 환경에 빠르게 적응할 수 있는 게임엔진의 개발은 코드중심의 개발로 인하여 유지보수에 많은 비용이 소요되는 개발공정을 가지고있다. 따라서 새로운 시스템 개발시 이전에 개발된 엔진부분들을 분석단계부터 반복하고 있는 실정이며, 이러한 과정을 반복하는데는 개발자의 독창성을 극대화하기 위한 이유도 있지만 시스템의 규모와 환경 등이 대형화되어 가는 개발환경에서는 부적합 할 뿐 아니라 게임성이 뛰어난 새로운 시스템개발에 많은 문제점을 가지고 있다.

1) 개발공정의 미비로 개발 기간과 개발비용 산출의 정확성이 떨어지므로 과다한 개발비용 부담.
2) 일반적인 응용소프트웨어에 비해 재사용성의 매우 저조하며, 재사용의 범위도 코드 중심의 참조 수준에 머물고 있음.
3) 컴포넌트 재사용을 위한 적용방법과 게임개발 방법에서 잘못된 적용방법으로 재사용성이 매우 저조함.
4) 게임엔진 개발 방법에서의 컴포넌트 종류와 검정방법, 관계성 등은 일반적인 응용소프트웨어 개발방법으로는 적용이 불가능함.

1.2 연구 목표

컴포넌트 기반 개발 방법을 적용하기 위하여 게임 소프트웨어 개발공정의 정형화 방안을 제시하고, 새로운 게임 소프트웨어 개발에 컴포넌트를 참조 적용할 수 있는 단계별 분류와 적용방법을 제시하여 재사용성을 향상 시킨다. 이러한 개발 방법의 효과적인 운영과 적용의 편리성, 수월성을 제공할 수 있는 저장소를 설계 및 구현한다.

1.3 연구 내용

본 논문의 연구내용은 다음과 같다.

1) 게임 개발공정의 재구성과 단위 엔진의 관련성 분석
2) 게임엔진 구조와 관계성 분석을 통하여 컴포넌트 유형을 추출
3) 컴포넌트 적용을 위한 엔진의 계층과 모듈 분류방법
4) 소프트웨어의 복잡도를 증가시키는 데이터의 속성 관리와 게임 컴포넌트 등록 및 저장방법
5) 게임엔진 개발에 필요한 컴포넌트 분류와 단위클래스 분류 방법

2. 기반연구

2.1 컴포넌트 기반 개발

컴포넌트 기반 개발이란 독립적인 기능을 담당하는 다양한 소프트웨어 컴포넌트의 집합으로부터 해당 업무 수행에 필요한 기능을 담당하는 하나 이상의 컴포넌트들을 결합하여 해당 업무를 위한 소프트웨어를 개발하는 기술을 말한다[1]. 현재 컴포넌트에 기반 한 응용 소프트웨어를 개발하는데 이용될 수 있는 상업적인 기술로는 Microsoft사의 COM/DCOM과 ActiveX, Sun사의 Javabeans와 Enterprise JavaBeans, 그리고 OMG의 CORBA 등이 있다. 일반적으로 컴포넌트 기반 개발의 절차는 6단계로 구분하는데, 각 단계에 대한 간략한 활동 내용을 소개하면 다음과 같다[2].

- 요구 분석: 시스템이 만족 시켜야 할 사항을 기술하는 단계로 영역 분석기법을 사용하여 소요되는 컴포넌트의 확인이 가능하도록 하는 근거를 제공한다.
- 컴포넌트 기반 설계: 컴포넌트에 기반하는 설계를 수행하며 이 단계에서는 구조 설계를 주로 수행한다. 해당 응용 소프트웨어를 구현하는데 필요한 컴포넌트의 인터페이스와 적용 컴포넌트를 추출한다.
- 소요 컴포넌트의 획득: 제3자가 개발한 컴포넌트를 획득하거나 적합한 컴포넌트를 찾지 못할 경우는 해당 컴포넌트를 구현한다.
- 소프트웨어 조립: 획득된 컴포넌트의 집합에 대해 전체적으로 원하는 기능을 수행할 수 있도록 통합한다.
- 수행 및 평가: 목표한 수준의 성능과 기능이 제공되는지 실제 수행을 통해 평가한다
- 유지 보수: 각 개발 단계에서 나온 산출물들을 관리하며 재사용을 향상시키기 위한 방법으로 품질 매트릭스를 통한 컴포넌트 품질 관리를 수행한다.

2.2 게임개발 공정

게임개발 전체공정의 형태는 (그림 1)과 같으며, 전체 공정 중 본 논문에서 고려하는 부분은 게임엔진과 관련된 부

분으로 제한 한다[6].

엔진부분에 대한 세부적인 구현단계는 다음과 같다.

1) 프로그램에 의한 그래픽통합

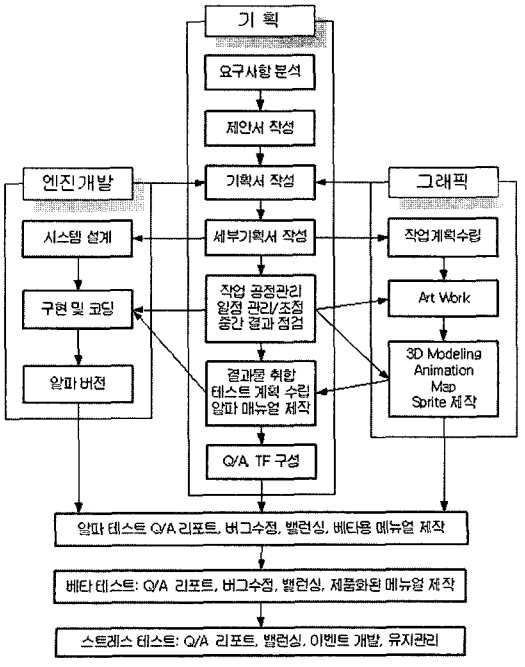
완성된 결과물을 프로그램으로 통합시키는 단계이다. 전체적인 그래픽 (MAP, 캐릭터, 아이템, 애니메이션)등을 프로그래머가 관계성을 정의하여 연결시키는 최종 작업이다.

2) 알파 테스트

완성된 게임을 테스트하는 단계로 프로그램 코딩에 대한 버그 테스트, 사운드 밸런스 조절, 그래픽 처리에 대한 문제점 테스트, 네트워크, 플레이에 대한 테스트 등을 통하여 일반적이 소프트웨어 테스트보다 여러 가지 복잡한 문제를 수정하는 단계이다.

3) 버그 수정 작업

테스트 단계에서 발견된 문제점을 수정 보완하는 단계이다. 특히 온라인 게임의 경우에는 많은 시간이 소요될 수 있는 단계이다.



(그림 1) 게임개발 프로세스

4) 베타 테스트

베타 서비스를 위하여 내부적으로 수정 보완한 부분들을 중심으로 최종적인 테스트를 실시한다.

5) 게임 서비스

베타 테스트 버전으로 외부에 공개하여 게임사용자들 플레이 할 수 있도록 환경을 구축하고 반응을 확인하는 단계이다.

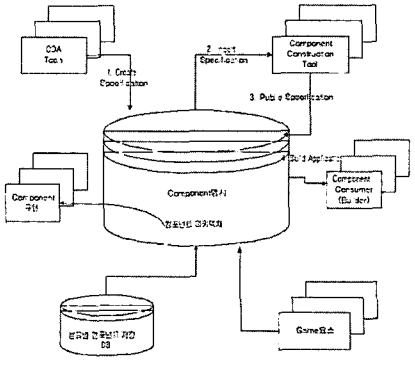
6) 정식 서비스

완성된 게임을 최종적으로 정식 서비스를 실시하는 단계이며, 계속하여 패치 업그레이드 및 이벤트 추가 등을 실시하여 게임을 더욱 재미있고 알차게 만들어 간다.

3. 컴포넌트 저장소

3.1 컴포넌트 저장소 기본형태

컴포넌트 저장소는 비즈니스 응용 구축을 위해 요구되어지는 컴포넌트를 찾고 공급함으로써 응용 개발을 위한 환경을 지원하는 라이브러리 시스템이다. 따라서 컴포넌트 저장소는 (그림 2)와 같이 컴포넌트의 분석, 설계 결과물에서 구현 결과물에 이르는 컴포넌트 라이프 사이클의 모든 정보를 정의되어진 아키텍쳐에 따라 저장 등록 관리하며 나아가 보다 진보된 검색 서비스의 부라우징 기능을 통해 컴포넌트에 의한 재사용(Reuse With Component)을 지원하는 도구이다. 컴포넌트 저장소는 (그림 2)와 같이 두 가지 관점에서 컴포넌트 생성과 활용을 위한 중심 매개체로서 도식화될 수 있다[9, 10].

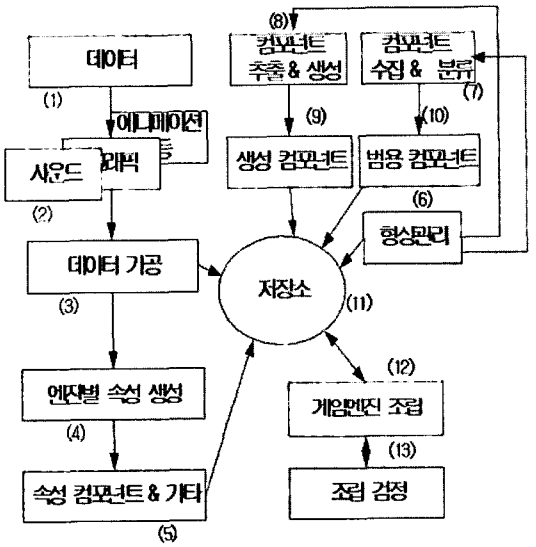


(그림 2) 저장소 운영의 기본 형태

즉, 정의된 컴포넌트 아키텍쳐에 바탕을 둔 컴포넌트의 명세와 구현을 객체지향 도구를 사용함으로써 생성하고 저장소에서 저장, 관리하며 이는 다시 컴포넌트 개발자에게 새로운 게임 개발에 필요한 요소로서 재사용되기 위해 배포활용 된다[3].

3.2 게임 컴포넌트관리 프로세스

본 논문의 CBD 방법 적용을 위한 데이터와 컴포넌트 저장소 관리 프로세스는 (그림 3)과 같이 정의한다.



(그림 3) 데이터와 컴포넌트 저장소 관리 프로세스

이는 프로젝트 수행에서 요구되는 필수적인 작업을 중심으로 재구성한 것이다. 따라서 컴포넌트 저장소 중심의 재사용 요소의 "획득-이해-적용"이라는 기본적인 재사용 원칙에 준하며, 특히 컴포넌트 확보과정에서 설계패턴의 식별 및 추출을 위한 일련의 세부 프로세스를 포함하고 있다. 저장소는 컴포넌트와 게임엔진 개발에 필요한 요소들을 동시에 관리, 제공하고 새로운 게임개발에 적용할 수 있도록 커스터마이즈를 위한 융통성을 제공한다.

(그림 3)에 표시된 ①-⑫ 프로세스의 과정을 상세하게 살펴보면 다음과 같다.

① 수집단계로 이전단계에서 분류한 엔진별 계층별로 분류하여 관리 한다.

② 분류된 자료는 상세 기획서를 참고하여 구체적인 가공계획을 수립하고 예상되는 동작 및 버퍼의 규모 해상도 등 데이터의 기본적인 비 기능적인 요소를 확정한다.

③ 분류된 자료를 기준으로 앞의 단계에서 체계화된 기획서의 내용에 의해서 데이터에 대한 가공을 실시하는 단계이다. 예를 들어 애니메이션 데이터의 동작추 가, 사운드 기능추가 부분 등을 이 단계에서 실시하고 분류된 모듈에 따라 각각의 저장소에 저장한다. 일반적으로 게임 개발사의 독자적인 통합 툴을 이용하는 경우와 일반적인 그래픽툴 또는 사운드툴 등을 이용하기도 한다.

④ 해당 게임에 적합한 기능적이 속성을 부여하는 단계이다. 앞 단계까지 저장된 비 기능적인 데이터 부분보다 엔진별로 분류하여 부여한 데이터의 속성이 게임개발에서는 재사용성이 뛰어난 부분으로 볼 수 있다. 예를 들어 그래픽의 모양은 차이가 있지만 움직이는 모션이나 움직임에 의한 속도, 버퍼의 변동값 등은 다른 게임개발에서도 재사용이 가능한 속성이므로 구체적인 정보까지 저장소에서 관리 운영해야 한다. 특히 이 단계에서는 캐릭터의 중요도와 그래픽 사용목적에 따라 속성과 레벨을 조절하며, 개발단계에서 API의 속성에 적합한 스크립트 형태로 변환하여 재사용할 수 있도록 저장소에서 관리한다.

⑤ 본 논문에서는 스크립트 형태의 컴포넌트와 데이터의 각종 동작의 좌표, 지연시간, 해상도, 동작의 시작위치 종료 관계 등을 메인 프로그램의 코딩 시에 고려하지 않고, 저장된 스크립터 타입의 컴포넌트를 참조하여 사용할 수 있도록 한다.

⑥ 앞 단계에서 분류된 계층별, 모듈별 클래스에 대하여 생성되는 형상들과 추출된 컴포넌트에 대한 계층의 범위에 따라서 생성 컴포넌트와 범용 컴포넌트 별로 분류한다. 범용 컴포넌트의 경우에는 시스템에 종속적인 부분들이 많이 존재하며, 일반적인 소프트웨어에서 사용되는 형태의 컴포넌트로 관리한다.

⑦ 형상관리단계에서 분류된 컴포넌트에 대하여 컴포넌트를 수집하고 관리하는 기능을 추진한다.

⑧ 이 단계의 컴포넌트는 게임에 관련된 부분들이 대부분이며, 이러한 부분들은 상대적으로 일반 소프트웨어 개발에서 사용하는 컴포넌트보다 재사용성이 다소 떨어질 수 있다.

⑨ 게임엔진의 개발에 직접적으로 사용되는 컴포넌트들로 엔진별 계층별로 분류하여 저장한다.

⑩ 게임엔진의 직접적인 기능에 관계되는 컴포넌트 보다

는 일반적인 응용 소프트웨어에서 사용하고 있는 컴포넌트가 대부분을 차지하므로 상세기획서와 요구명세에 있는 순서에 의해 참조형태 중심으로 저장 관리한다.

⑪ 저장소에 관한 부분은 저장소 구현 부분에서 구체화한다.

⑫ 컴포넌트를 참조하여 엔진을 조립하고 개발하는 단계로 조립방법에 따라서 형상화된 다이어그램을 참조하여 코드화 할 수도 있고, 바이너리 형태, 스크립트 등 다양한 형태의 컴포넌트들을 참조하여 조립하는 단계이다.

4. 컴포넌트 저장소 구현

4.1 시스템 환경

4.1.1 시스템 정의

본 시스템은 앞장에서 서술한 이론적 배경을 바탕으로 분석, 설계된 정보들을 기반으로 엔진개발 환경에서 이용할 수 있는 데이터, 스크립터, 속성, 컴포넌트를 식별, 저장, 응용하는 저장소지원 시스템으로 정의한다.

4.1.2 시스템 개발환경

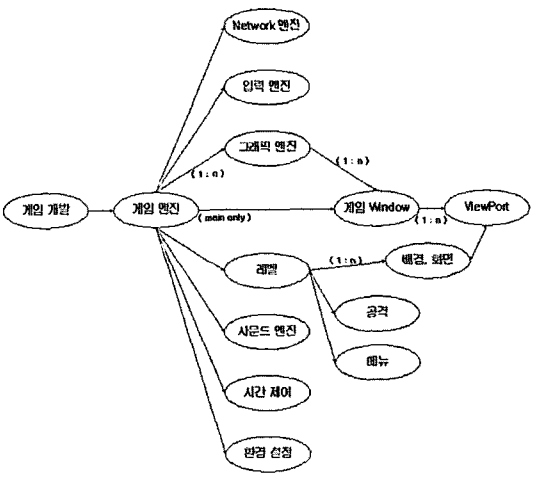
시스템 개발환경 중 저장소 서버의 운영체제는 Windows 2000를 사용했고, 데이터베이스는 MS-SQL, 사용자 인터페이스는 ASP와 HTML을 사용했다. 지원 시스템은 게임 컴포넌트의 공용뱅크 역할을 지원하기 위하여 구축하였으며, 현재 운영되고 있는 상업적인 유통 사이트와는 차별되게 게임개발에 필요한 광의의 컴포넌트를 관리운영, 개발을 지원할 목적으로 개발된 것이다.

1) 시스템 인터페이스

웹에서 이용할 수 있는 본 시스템의 사용자 인터페이스는 대상 컴포넌트를 등록, 검색, 이해를 지원하고, 컴포넌트 기반의 게임개발에 적용할 수 있는 저장소이다. 뿐만 아니라 게임엔진 개발에 필요한 다양한 멀티미디어 요소들도 동시에 관리 운영할 수 있는 편리성도 제공하고 있다.

2) 컴포넌트 저장소 코드 정의

컴포넌트를 식별, 저장하고 필요할 때 검색하여 기능적으로 관련된 컴포넌트 사이를 브라우징하고 네비게이트 할 수 있기 위해서는 컴포넌트 분류체계의 제공은 필수적이다. 따라서 컴포넌트 자산들을 저장시킨 후 요구가 발생할 때마다 검색, 이해를 통해 재사용을 실현하는 저장소 시스템은 컴포넌트의 인덱싱의 기본이 되는 코드 체계로부터 시작할 수 있다[4, 5].



(그림 4) 엔진분류를 위한 계층 구조

본 논문에서 제시하는 컴포넌트 아키텍처에 따른 명세 분류코드는 (그림 4)의 엔진분류를 위한 계층구조에 의해 다음과 같은 원칙에 의해 작성된다[8].

① 아키텍처 계층을 대분류로 표기하며 각 계층은 영역별로 재 표기 한다.

② 중분류는 규모에 따라 상세하게 세분화고 소분류를 표기한다.

③ 소분류는 기능적인 하위레벨의 단계를 표시한다.

④ 중분류와 소분류는 각각 네자리로 정하여 표시한다.

⑤ 소분류는 4단계이하로 기능적로 레벨링 한다.

⑥ 중분류 및 소분류의 잠재적 의미는 컴포넌트 기능별 규모별 표시한다.

⑦ 영역의 공통부분 표현을 위해서 중 분류와 필요 부분에 "0000"은 공용으로 정의한다.

⑧ 각 영역마다 그룹화하기 모호한 컴포넌트들은 "XXX"로 표기하여 기타로 정의한다.

3) 게임 컴포넌트 기본 코드 형식

게임 컴포넌트 저장소의 분류를 위한 상위레벨의 코드형식은 〈표 1〉와 같이 한 자리로 구분했다. 그리고 오브젝트와 컴포넌트의 하위레벨은 〈표 2〉와 같이 두 자리로 기타

코드의 요소들은 〈표 3〉과 같이 구성되어 있다[7].

Top level Engine	T
Graphics and Scene Graph	G
Input and Configuration	I
Networking	N
Sound	S
Animation	A
Effects	E
User Interface	U
Base Object	B

〈표 1〉 상위레벨의 기본 코드분류

캐릭터	CH
맵	MA
아이템	IT
몬스터	MO
예비1	
예비2	

〈표 3.3〉 하위레벨의 코드구성

이벤트 사운드	ES
배경 사운드	BS
예비1	
예비2	

〈표 3.4〉 기타 요소 코드

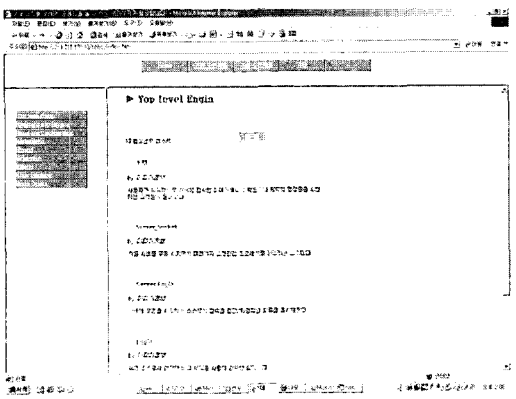
4.2 K-RPG 저장소의 기본 구조

데이터베이스의 인덱싱 형식은 순차적인 형식으로 검색할 수 있도록 구현되었으며, 유사 컴포넌트를 동시에 참조할 수 있도록 설계되었다. (그림 5)와 같이 리스트에 의해 검색된 컴포넌

트의 속성은 상세 검색이 가능하도록 구성되었으면 일반 사용자모드와 관리자 모드로 구분하여 운영할 수 있도록 구현되어 있다.

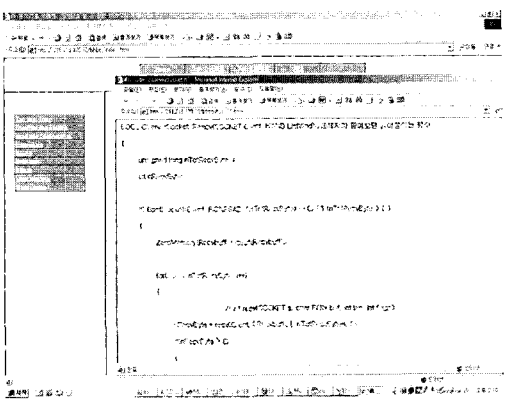
(그림 5)은 저장소의 기본화면으로 각 주제별 컴포넌트의 리스트와 컴포넌트에 대한 기본설명, 제조사, 등 컴포넌트 관리와 운영에 필요한 정보를 검색 및 조회할 수 있다.

(그림 6)은 컴포넌트를 상세하게 검색한 경우를 볼 수 있으며, 컴포넌트에 대한 소스를 참조할 수 있도록 설계하여 컴포넌트 적용시 안정성을 기할 수 있도록 구현하였다. (그림 6)은 오브젝트 형식으로 구성된 게임 컴포넌트를 게임엔진의 적용 형식별로 분류하여 사용자는 시각적으로 확인과 속성을 저장소로부터 검색할 수 있도록 구현되었다. 게임의 그래픽적인 요소와 레벨, 플레이의 속성을 고려한 것으로 볼 수 있다.

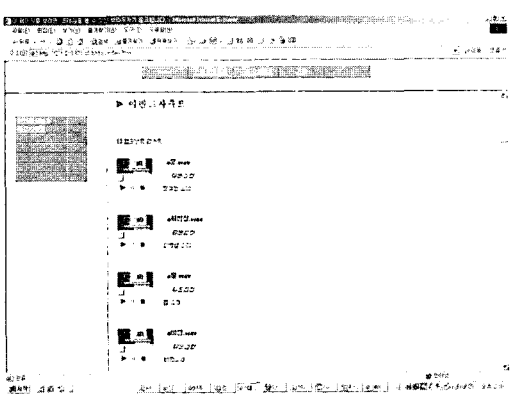


(그림 5) 저장소의 기본구조

(그림 7)은 사운드 컴포넌트의 검색형식과 기타 게임개발에 필요한 컴포넌트의 저장 구조이다.



(그림 6) 컴포넌트 소스코드 검색



(그림 7) 사운드 컴포넌트 검색형식

5. 결 론

컴포넌트 기반의 개발방법의 적용에는 많은 장단점을 가지고 있다. 그러나 개발 기간과 비용적인 측면, 코드의 신뢰성 확보, 유지보수의 편리성을 고려할 경우 기존의 방법과 비교하여 몇 가지 단점을 가지고 있지만 이러한 단점들은 소프트웨어 개발비용과 신뢰성측면을 고려한다면 적용하는 편이 우수하다고 평가할 수 있다.

본 논문에서 제시한 컴포넌트 기반 게임엔진 개발방법을 위한 저장소는 앞에서 살펴본 바와 같이 다음과 같은 문제점을 해결할 수 있다.

전체시스템의 개발시간의 단축과 각 단위별 공정산출의 정확성을 증대시킬 수 있다. 특정한 범용 모듈개발에 집중화와 재사용성의 증대로 단순화된 개발방법의 적용이 가능하므로 정형화된 엔진개발이 가능하다.

따라서 플랫폼으로부터 독립성을 유지할 수 있으므로 코드의 안정적 유지가 가능하며, 재사용 가능성과 관리, 개발자의 지식공유, 시스템 개발주기의 공정이 보다 가시적인 적용이 가능하므로 특화된 시스템개발 분야인 게임엔진 개발에 효율성을 가져올 수 있다.

물론 위와 같은 여러 가지 장점을 가지고 있지만 컴포넌트의 수집과 최초 적용 시 기술의 탄력성이 떨어진다는 문제점도 가지고 있다. 이러한 문제점은 최초의 시스템 개발에 국한된 부분이며, 반복과 지속적인 개발에 있어서는 소프트웨어 생산성 향상에 크게 기여할 수 있다.

참고문헌

[1] R. Lipsett. E. Marschner, "VHDL-The Language", IEEE Design Test, 1986.

[2] A. S. Gilman, "VHDL-The Design Environment", IEEE Design and Test, 1986.

[3] Scott Henninger, "Supporting the onstruction and Evolution of Component Repositories", 1998 International Workshop on Component-Based Software Engineering, ICSE, 1998.

[4] 김행곤 외, 영역별 분류 방법, 최종연구보고서, 한국전자통 신연구원, 1999, 11.

[5] 김수동, "컴포넌트 기반 소프트웨어 공학의 개요 및 요소 기술," 한국정보처리학회 소프트웨어 공학연구회지, 제2권 1호, p.16, 1999. 3.

[6] 한국게임개발원, KGDI 연구보고서 02-004, 한국게임산업개 발원, 2002.

[7] 송의철, "Game Architecture and Design", 도서출판 모두원, 2003.

[8] http://JTgame.com

[9] Scott Henninger, "Supporting the onstruction and Evolution of Component Repositories", 1998 International Workshop on Component-Based Software Engineering, ICSE, 1998.

[10] Robert C. Seacord, "Software Engineering Component Repository", Proceedings of 1999 International Workshop on CBSE, Los Angeles, at www.sei.cmu.edu/cbs/

송 의 철

경남대학교 컴퓨터공학과 석사
경남대학교 컴퓨터공학과 박사수료
1994년 - 현재 세경대학 멀티미디어과 부교수
관심분야 소프트웨어공학, 게임엔진, 가상현실