SEDRIS STF를 이용한 데이터 변환

이 광 형†

요 약

멀티미디어 분야에서는 오늘날의 시스템의 요구사항을 만족할 뿐만 아니라 미래의 데이터 공유 필요성을 만족할 수 있도록 확장될 수 있는 환경 데이터 표현 및 교환 메카니즘을 필요로 한다. 이러한 메카니즘은 표준화된 방법으로 데이터 표현 및 액세스를 할 수 있도록 해야 한다. SEDRIS의 STF(SEDRIS Transmittal Format)는 이러한 목적을 가능케하는 표준으로써, 환경 데이터 사용자 및 생성자에게 명료하게 정의된 교환 명세를 제공한다. 본 논문에서는 SEDRIS의 표준 교환 포맷을 이용하여 상용 포맷(3DS MAX)을 정보 내용의 의미 손실없이 교환 포맷으로, 교환 포맷을 상용 포맷으로 변환하기 위한 컨버터를 개발하였다.

키워드 : SEDRIS, DRM, 컨버팅, 교환 포맷, 객체 모델링

Data Conversion using SEDRIS STF

Lee, Kwang-Hyung

ABSTRACT

The multimedia community needs an environmental data representation and interchange mechanism which not only satisfies the requirements of today's systems, but can be extended to meet future data sharing needs. This mechanism must allow for the standard representation of, and access to data. The SEDRIS STF(SEDRIS Transmittal Format)provides environment data users and producers with a clearly defined interchange specification. In this paper I develop data converter commercial data(3DS MAX) format to standard interchange format and vice verse without losing semantic of information content using SEDRIS standard interchange format.

key word : SEDRIS, DRM, converting, exchange format, object modeling

1. 서론

한 시스템의 데이터를 다른 형식으로 변환하는 것은 M&S(Modeling and Simulation) 시스템의 소스(source)와 대상(destination) 모두에 대해 엄격하게 정의되어 있는 데이터베이스 형식 명세에 기초한다. 서로 다른 데이터베이스 형식 때문에 각각의 변환은 일회성에 의한 자료 변환 소프트웨어 애플리케이션의 개발을 요구한다. 이러한 일대일 솔루션은 비싸고, 시간 소모적이며, 종종 신뢰성이 떨어진다. 대상 시스템의 특수한 수행 요구를 맞추기 위해, 변환된 데이터베이스는 사용할 수 있는 실시간 형식을 얻기 전에 통상적으로 추가적인 변환을 수행한다. 또한 각각의 변환은 데이터 손실이

† 정 회 원: 고려대학교 강사(교신저자)
논문접수: 2004년 5월25일, 심사완료: 2004년7월17일

나 오류에 대한 위험이 높으며, 변환 소프트웨어 모듈의 개발과 유지 관리는 매우 어려워진다. 이러한 근거로 가상적으로 구현된 환경 데이터베이스의 효과적인 교환을 지원하기 위한 새로운 솔루션이 요구되었다. SEDRIS 프로젝트는 이러한 문제들을 해결하기 위해 시작되었다[1,3,6].

SEDRIS의 주된 목적들 중 하나는 모호하지 않은 환경 데이터 교환에 대한 메카니즘을 제공하는 것이다. 이 목표를 수행하기 위해서 SEDRIS는 모든 시뮬레이션 응용에 사용된 종합 환경의 모든 데이터 요구사항을 둘러싸고 있는 데이터 표현 모델(DRM)을 개발하였다.

SEDRIS 개발 이전에, 환경 데이터 상호 교환 메카니즘을 제공하기 위해 시도하는 프로젝트들은 상호교환 매개물의 개발에 중점을 두었다. 그러나 형식에 집중된 접근은 애매한 데이터를 이끌어 냈는데, 그 이유는 데이터의 기초 의미와 관계는 단지 데이터 형식의 묘사만으로 얻을 수 없기 때문이다. SEDRIS DRM은 데이터의 명백한 묘사를 제공할 뿐만 아니라 사용자가 해석을 제대로 할 수 있기 위해 중요한 데이터간의 관계도 정의한다. DRM의 중요한 이점은 모든 데이터 형태는 시스템내의 다른 데이터 형태와 가지는 관계뿐만 아니라 속성의 완전하고 모호하지 않은 정의를 포함함을 보장할 방법을 제공한다[2].

본 논문에서는 국제 표준으로 규격화된 SEDRIS 데이터 모델인 DRM과 교환 포맷인 STF를 이용한 중간 교환 포맷을 정의하고, 상용 그래픽 도구인 3DS MAX 데이터 포맷의 물리적인 구조뿐만 아니라 논리적인 구조를 중간 교환 포맷으로 변환하고 또한 중간 교환 포맷을 3DS MAX 데이터 포맷으로 변환하기 위한 컨버터를 개발한다.

2. SEDRIS

미국방성은 모델링 및 시뮬레이션이 점차적으로 성숙됨에 따라 상호 조작과 재사용을 강조하여 왔다. DIS, ALSP, HLA 등은 상호 조작과 재사용의 촉진을 겨냥한 미 국방성 초기의 소수의 예들이다. 군사작전 모델은 자연 환경의 표현과의 상호 작용에 의존한다. 결과적으로 환경의 공통적인 표현은 이질적인 시뮬레이션의 상호 조작을 위해 없어서는 안될 필수 조건인 것이다. 이것은 지형, 해양, 대기 및 우주공간에 대한 균일하고 믿을만한 3차원 표현을 요구하게 되었으나 미 국방성은 M&S(Modeling and Simulation) 애플리케이션들 중에는 환경 데이터의 표현, 재사용 및 상호 교환을 위한 표준 메카니즘이 존재하지 않는다는 것을 인식하였다. SEDRIS 프로그램의 목적은 실제 환경의 완벽한(지형, 해양, 대기, 및 우주 공간) 데이터 모델, 데이터 모델에 접근하기 위한 방법과 관련된 상호 교환 양식을 획득하여 제공하고자 하는데 있다.

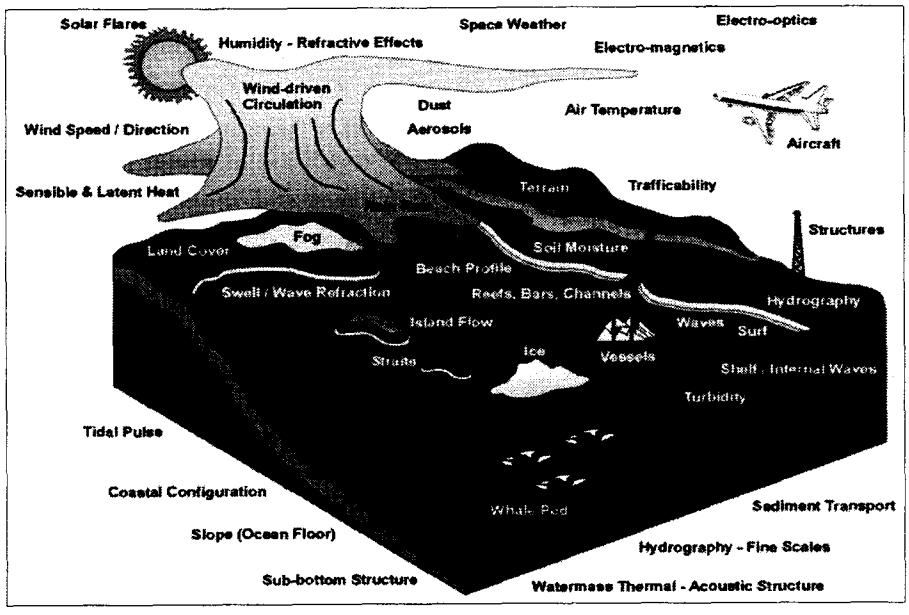
환경 데이터의 재사용 및 공동 사용의 진행에서 SEDRIS의 목적은 다음과 같다.

- ○ 완벽하고 명백한 데이터 표현
  SEDRIS는 환경 데이터에 대한 명시적인 정보 표현 방법/기능을 갖는다. 이는 데이터 모델링이라고 하는 과정을 통하여 실행된다. 이 메타 데이터는 통합 환경의 성분들을 묘사하기 위하여 공통적이고 표준적인 접근을 가능케 한다.
- ○ 보편적, 손실 없는 데이터 교환
  SEDRIS는 처음 형식을 무시하고 데이터의 손실 없는 교환을 보장하는 강인한 인터페이스 메카니즘을 구현한다. 보편적 표준이 인터페이스되고 데이터의 다항식적 표현은 호환적이고 분산적인 시스템이 통합환경을 묘사하는데 공여하게 되는 것을 보장한다.
- ○ 공통 액세스 인터페이스
  SEDRIS 실행 명령을 쉽게 하기 위하여 도구와 소프트웨어 공통 집합이 재사용을 위해 제공되어 진다. SEDRIS 응용 프로그래머의 인터페이스(API : Application Programmer's Interface)는 통합 환경 데이터에 접근토록 제작된 소프트웨어와 SEDRIS 실행 명령 매체 사이의 인터페이스를 제공한다.

SEDRIS에서 표현 가능한 환경 영역에는 지상, 해상 등과 같은 공간적 영역뿐만 아니라 바람의 방향 및 속도, 날씨의 변화 등과 같은 대기의 변화도 표현이 가능토록 구성되어 있으며 하여 비시각적 다양성을 구현하기 위한 요소 또한 포함하고 있다. <그림 1>은 SEDRIS가 표현 가능한 환경 영역을 보여주고 있다.

3. SEDRIS 기술적 특성

SEDRIS의 기술적 요소는 살펴보면 환경 데이터



< 그림 1 > 환경 영역

를 표현하기 위한 문법 및 구조적 시맨틱 모델이 표현하고 있는 환경적 객체를 명확하게 정(semantic)을 제공하는 DRM(Data Representation Model), 정확하고 효율적인 소프트웨어 구현을 위한 좌표계를 일관성 있게 정의한 SRM(Spatial Reference Model), 각종 환경적 사물(Thing)을 분류와 속성 등과 같은 것을 통해 정의한 EDCS(Environmental Data Coding Specification), 환경 데이터의 독립적인 저장 및 전송 플랫폼인 STF(SEDRIS Transmittal Format)과 응용 프로그래밍 인터페이스(API)로 구성되어 있다.

데이터 표현 모델(DRM)이란 각기 속성을 가진 데이터 요소 및 그 데이터 양식들 간의 논리적인 관계에 대한 설명이다. DRM은 UML를 이용하여 표현되어 있으며 총 327개의 클래스로 구성되어 있다.

공간 데이터의 상호운용성은 공통 공간 참조 모델의 사용을 통해 확보될 수 있다. 공간 정보의 상호운용성을 확보하기 위해서는 공간 참조 프레임과 ORM(Object Reference Model), ERM(Earth Reference Model)은 좌표계가 일관되게 위치를 기술하도록 정의되야 하며 상이한 공간 참조 프레임 간 변환 및 전송을 할 수 있는 메카니즘을 가지고 있어야 한다. 현재 SEDRIS는 151개의 공간 참조 프레임을 지원하고 있다.

환경 데이터 코딩 규약(EDCS)은 특정한 데이터의하여 혼란을 야기하지 않기 위한 메카니즘을 제공하고 있으며 다음과 같이 총 9개의 EDCS 사전들로 구성된다.

- EDCS Classification (EC) Dictionary
- EDCS Attribute (EA) Dictionary
- EDCS Attribute Value Metadata (EM) Dictionary
- EDCS Attribute Enumerant (EE) Dictionary
- EDCS Unit (EU) Dictionary
- EDCS Unit Scale (ES) Dictionary
- EDCS Unit Equivalence Class (EQ) Dictionary
- EDCS Organizational Schema (EO) Dictionary
- EDCS Group (EG) Dictionary

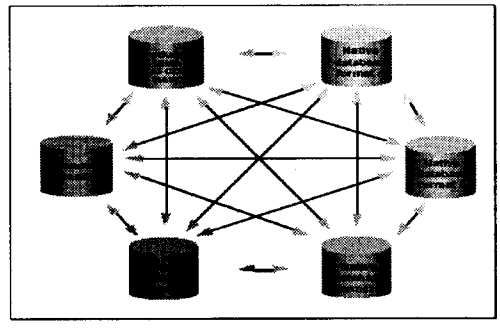
4. 기존 컨버터 문제 분석

상용 3D 애니메이션 또는 그래픽 도구에 있어서, "계산(computation)구조"는 애플리케이션이 작동하기 위한 매우 중요한 부분이며, 각각의 도구에 있어서의 경쟁력과 우수성을 나타내는 차별성이 상당히 많이 반영하게 되는 부분이기도 하다. 이러한 "계산구조의 반영은, 애플리케이션마다의

scene-graph 구조의 특징적 요소나 소프트웨어 (plug-in) 인터페이스를 통해 현실적으로 나타나는데, 여기서 나타나는 연동 및 호환성의 문제가 컨버팅(converting)에 상당한 난점들로 작용하게 된다.

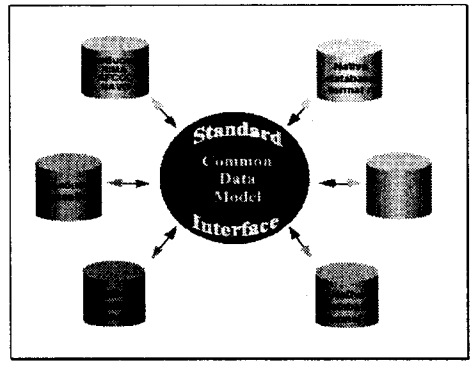
가장 이상적인 컨버팅은, 가장 정확한 "3D-semantic"에 접근한다는 의미에서 소스와 대상의 메모리 scene-graph에 동시에 접근하는 것이 좋지만, 이 방법은 연동상의 이유로 불가능에 가깝다. Maya[5]의 경우, 다른 애플리케이션에 Maya의 'scene-graph' 개념을 적용할 수 있도록 라이브러리를 초기화 할 수 있어 문제가 없지만, 이는 예외적인 경우에 불과하다. 또한, Max와 같은 경우 다른 애플리케이션에서 Max의 'scene-graph'를 사용하기 위해서는, Max 애플리케이션을 따로 작동하고, 이를 'COM 통신'을 통해 접근해야 하는데, 이러한 연동자체가 어렵고, COM 통신을 통한 접근이 본래의 scene-graph 접근이라는 취지에 맞을 정도로 충분하게 지원되는지의 여부도 불투명하다. Max[4]나 Maya 외에 다른 .도구의 경우에는 다른 애플리케이션에서 해당 scene-graph로의 접근이 불가능한 경우도 있을 수 있다. 이는 요즘의 3D-애니메이션 도구들이 라이브러리 기반이 아닌, 플러그인(plug-in) 기반으로 구성되는 추세와 무관하지 않은데, 이럴 경우 플러그인을 통해 scene-graph에 접근하는 것은 필수라고 해도, 별도의 애플리케이션을 통해 scene-graph를 조작하는 것을 지원하는 것은 선택사항일 뿐이기 때문에 "양쪽의 scene-graph에 모두 접근하는 컨버팅 플러그인"의 경우 양쪽 모두의 플러그인(예를 들어, Maya에서도 동작하고, Max에서도 동작하는)이 되어야 한다는 결론밖에는 되지 않는다. 따라서, 소스와 대상의 메모리 scene-graph를 직접 연결시키는 방식은 플러그인 아키텍처 간에는 현실적으로 이루어지기 어렵다.

그러나 어느 한쪽의 scene-graph만을 작동시키고 (플러그인을 통해서건, 아니면 별도 애플리케이션을 통해서건), 대응되는 상대편은 파일 형태의 데이터로 import/export하는 차선책은 가능하다. 이 방법은 연동의 문제를 상당부분 피할 수 있기 때문에 많이 쓰이고 있다. 이 방법은 '데이터 중심적인 접근'이라는 면에서는 위의 방법보다는 현실적으로 상당히 타당한 시사점을 보여주고, 실제 Max나 Maya에서는 몇가지 다른 포맷(vrml, dwg, igs, ...)의 importer/exporter를 플러그인으로 제공하며, 개발자나 소프트웨어 업체에서 별도로 개발할 수 있는 플러그인 API를 제공한다. 하지만, 이 경우의 문제는 컨버팅을 위해 너무 많은 노력이 드는 단점이 있다. 즉, n개의 서로 다른 도구나 포맷간에는 n * (n-1) / 2의 관계가 발생하는데, n개의 다른 포맷간의 컨버팅을 위해 $O(n^2)$개의 export/import 플러그인이 필요하고, 어느 한 도구 입장에서만 보아도, O(n)개정도의 export/import를 만들어야 한다. 이는, 노력의 분산으로 인해 "컨버팅 결과"를 현격하게 떨어뜨릴 우려가 있다.



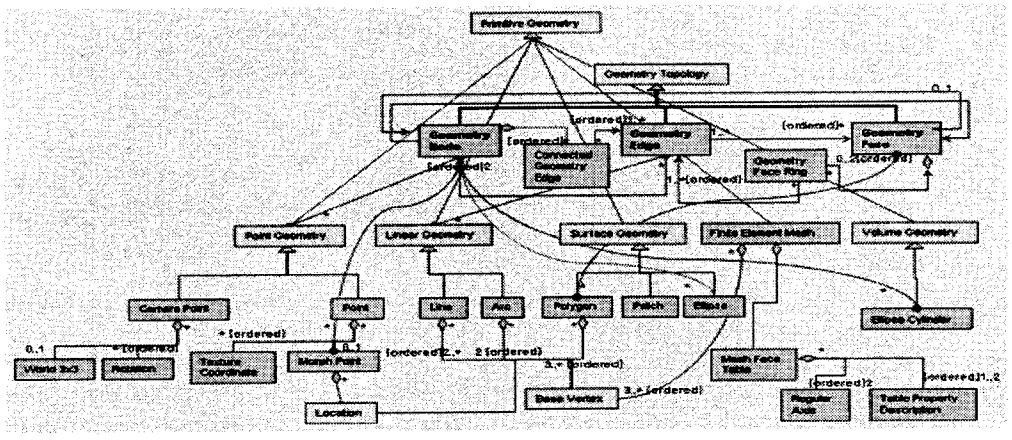
<그림 2> 교환 포맷이 없는 경우의 컨버팅

이러한 문제는 중간에 공통의 "교환(중간)-포맷"이 존재할 경우, 상당히 개선될 수 있다. 왜냐하면, n개의 서로 다른 포맷간의 컨버팅이 O(n)개의 export/import 플러그인으로 해결될 수 있고, 어느 한 도구의 입장에서는 공통의 호환 포맷으로 1개의 export/import 플러그인만을 지원하면 되기 때문이다.



<그림 3>교환 포맷을 이용한 컨버팅

직접 연결 방식을 취하고 있는데, 이것은 치명적인



<그림 4> Mesh 데이터 표현 관련 SDRM

5. 그래픽 객체 구조 및 연관관계

5.1 3DS MAX의 내부 객체 구조 및 연관관계

3차원 그래픽 모델 및 애니메이션을 구조적으로 표현하고 이를 포맷으로 구현하기 위해서, 3DS MAX[4], MAYA[5] 등 많은 그래픽 에디팅 및 렌더링 프로그램들은, 자체적으로 고유의 그래픽 객체를 표현하기 위한 객체 표현 방식 및 객체간의 연관관계를 구현하고 있다. 3차원 그래픽 객체를 가장 효율적으로 표현하기 위해 사용되는 구조적 관계성에서 가장 중요한 것은, 객체들 간의 연관관계를 표현하기 위한 관계 모델이다.

3DS MAX의 객체 계층구조 표현 모델의 특징은 널 노드(Dummy Object)가 실제 그래픽 객체에 연결 되기도 하고, 실제 객체가 널 노드(Null Node)에 직접 연결 되기도 한다. 널 노드는 비쥬얼 객체가 아닌 객체의 연관성, 객체에 영향을 미치는 환경 객체의 존재를 나타내기 위해서 또는 객체에 직접 부여되기 보다는 주변 환경의 조건을 표현하기 위해서 필요한 비 그래픽적 객체를 나타낼 때 주로 사용된다. 또한 이들 널 노드는 객체들간의 피봇 포인트(pivot point)를 나타내기 위해서도 사용되어진다. 그러나 3DS MAX에서의 이러한 피봇 포인트 구조와 다른 여타 그래픽 프로그램들(Maya, Lightwave 등)은 전혀 다른 방식을 취하고 있다.

3DS MAX는 이러한 다양한 종류의 객체들의 관계와 조건 등을 표현하기 위해서 그래픽 객체간의 단점을 가지고 있는 그래픽 표현 모델이다. 그 이유는 객체들간의 계층구조를 나타내기 위한 일종의 스켈레톤(skeleton) 타입에 대한 구조가 문제가 되기 때문이다. 즉 3DS MAX에서와 같이, 기하(geometry) 객체가 직접 다른 기하 객체에 직접 연결 되는 방식을 취하게 되면, 기하 객체를 나타내는 객체를 포함하여 하단의 모든 자식 객체들도 모두 각각의 좌표을 나타내기 위하여 4x4의 변환 행렬를 가져야 하며, 이는 객체들간의 상대적인 좌표를 계산하기 위해서 객체들간의 계층구조에 변화가 생길 때 마다 매번 변환 행렬를 재 생성하거나 피봇 포인트를 다시 설정해야 하는 등 많은 수정을 요하는 어려움을 만들게 된다.

5.2 SEDRIS 내부 객체 구조 및 연관관계

SEDRIS의 DRM 즉, SDRM은 현상적으로 존재하는 모든 환경 데이터를 표현하기 위해 만들어진 모델과 모델링 방법에 대한 규칙이다. 각각의 DRM 클래스는 추상적 혹은 실제적인 하나의 객체를 표현하기 위한 객체로서 정의할 수 있으며, 각각의 객체들은 서로간의 연관관계를 표현할 수 있는 다양한 연관 방법을 제시하고 있다. 또한 각 객체에는 객체만이 가지는 고유의 속성(Attribute)들을 표현할 수 있다.

SEDRIS의 DRM 클래스중에서 3D 그래픽 데이터의 변환을 위해 사용될 수 있는 가장 중요한 DRM 클래스는 <GEOMETRY>이다. 이 클래스와 하위의 모든 aggregate 클래스들은 그래픽 기하

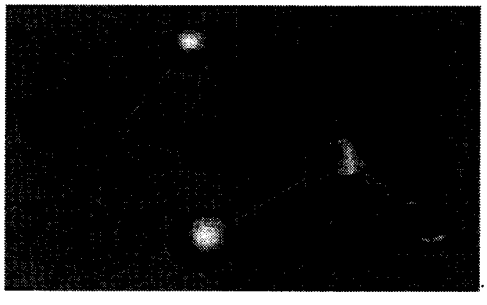
객체를 표현하기 위해서 바로 사용될 수 있는 유용한 DRM 클래스들이 존재한다(<그림 4> 참조).

MAX와 Maya등 3D 그래픽 에디터의 데이터 변환에는 이들 클래스중에서, <Union of Geometry>, <Union of Geometry Hierarchy>, <Union of Primitive Geometry> 등이 주요한 클래스들로서 사용된다. 또한 SEDRIS DRM의 가장 큰 특징중의 하나인 <Model> 클래스는 자체의 지역 좌표계(Local Coordinate)와 변환 정보를 가진 하나의 객체로서, 주어진 환경 영역에서 한번이상 존재할 수 있다. 즉 <Model>은 언제나 재사용이 가능하다. 또한 <Model> 내에서도 다른 하위 <Model>을 컴포넌트 개념으로서 가지고 있을 수 있어, 다중 복합 <Model>이 존재할 수 있다. 이것은 3D MAX나 Maya에서 사용되는 하나의 캐릭터의 복합적인 구성 요소들을 표현할 수 있다. 즉, 인간 모델의 경우, 몸통에 붙은 회전이 가능한 팔과 다시 팔에 연결된 일정한 회전각을 가지는 팔목과 손가락 그리고 다시 움직임이 지정된 발과 관절의 표현 등이 그대로 SEDRIS DRM으로 표현이 가능하다.

5.3 3DS MAX와 SEDRIS DRM의 변환

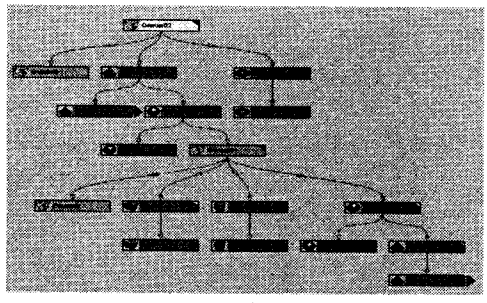
3D 그래픽 데이터간의 데이터 변환을 위해서는 무엇보다도 그래픽 데이터 내에서 표현되고 있는 그래픽 객체간의 연관관계를 나타내는 객체 계층구조의 변환이 가장 핵심이 된다.

다음 <그림 5>는 3DS MAX에서의 객체간의 그룹 및 링크 관계를 나타내고 있는 실제 화면이다



<그림 5>객체 연관관계 예제

<그림 6>은 <그림 5>에 대한 객체 계층구조를 트리구조로 나타낸 것이다.



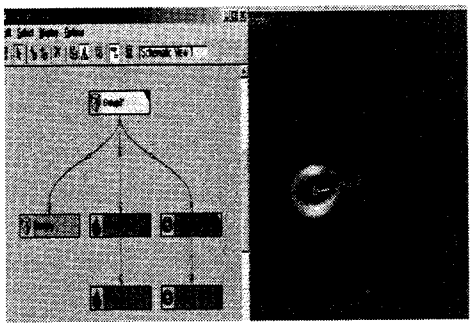
<그림 6> <그림 5>에 대한 트리 구조

<그림 6>에서 보는 바와 같이 최상위에, torus와 cone이 그룹되어 있으며, 하나의 sphere가 다시 cone을 링크하고 있다. cylinder와 box가 그룹을 이루고 있으며, 이 그룹이 상위의 sphere를 링크하고 있으며, 또 다른 sphere가 이 두 번째 그룹을 링크하고 있고, 마지막으로 또 다른 Cone object가 이 sphere를 링크하고 있다. 3D MAX viewport상에서, 최상위의 그룹인 torus와 cone을 회전시키거나 이동시키면 그 하위의 모든 객체는 같이 이동하거나 회전하게 된다. 이동하거나 회전의 중심 축인 피봇 포인트는 최상위 그룹에 상대적으로 반영되게 된다. 이러한 객체 계층구조를 다른 그래픽 프로그램으로 변환하기 위해서는 이를 적절히 보존할 수 있는 중간 객체 계층구조(intermediate object hierarchy) 시스템이 필요하다.

앞에서 설명한 바와 같이, 3DS MAX가 채택하고 있는 객체 계층구조 시스템은 단점이 존재한다. 이것은 Maya나 기타 다른 3D 그래픽 에디팅 프로그램이 채택하고 있는 좀더 효율적이고 개념적 일치성을 보장하는 DAG 시스템보다 뒤떨어진 방식이다. SEDRIS DRM(SDRM)은 3D 그래픽 데이터들의 객체 계층구조를 정의하기 위한 근본적인 객체 표현을 충분히 지원하는 클래스들과 연관관계 규칙을 이미 제공하고 있다. 이에 3D MAX의 객체 계층구조를 SDRM으로 표현하기 위한 SDRM 클래스들과의 대응관계는 다음과 같다.

o Group/Link의 SDRM으로의 변환 개념

먼저 그룹(<그림7> 참조)에 대하여, 3DS MAX는 Dummy 객체라 불리는 루트 객체를 그룹된 여러 개의 객체들이 링크를 하고 있는 방식으로 구현하고 있다. 따라서 DRM으로 다음과 같은 구조로 표현할 수 있다

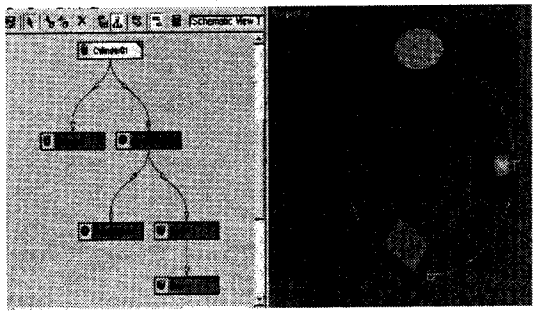


< 그림 7 > 그룹의 예

```
<Union of Geometry Hierarchy> // Scene
   <Union of Geometry Hierarchy>
      // 3DS MAX의 Dummy Object
   <LSR Transformation>
      <Geometry Model Instance>
            // Model Torus-01
      <LSR Transformation>
      <Geometry>Model Instance>
            // Model cone-01
      <LSR Transformation>
```

<Union of Geometry Hierarchy>가 한 레벨 존재해야 것은, 그룹되지 않은 개별 모델과의 분별을 위함이다, 즉 <Union of Geometry Hierarchy> 가 하나의 레벨만 나오는 경우(그룹이 아닌, 단지 여러 개의 모델 인스턴스가 존재하는 경우)와의 구별을 위함이다. 두번째 <Union of Geometry Hierarchy> 아래의 <LSR Transformation>이 바로 널 노드의 LSR 변환을 나타내며, 이것은 그룹의 기하 피봇 포인트를 나타낸다.

링크(그림 8> 참조)의 경우는, 3DS MAX가 객체에서 객체로 직접 연결하는 방식을 취하고 있는



< 그림 8 > 링크의 예

데, 이것의 단점을 보완하고 다른 포맷(Maya 등)으로 변환의 일반성을 구현하기 위해서 SDRM의 <Union Of Geometry Hierarchy>를 DAG 시스템의 널 노드 개념으로 정의하여 사용하면 된다

```
<Union of Geometry Hierarchy> // Scene
    <Geometry Model Instance>
            // Model cylinder-01
    <LSR Transformation>
        <Geometry Model Instance>
            // Model Sphere-01
        <LSR Transformation>
        <Geometry Model Instance>
            // Model box-01
        <LSR Transformation>
```

따라서 앞서 예로 든 3DS MAX의 예제를 SEDRIS의 DRM으로 표현하면 <표 1>과 같다.

주의 할 점은, (1) 이라고 표시된 부분에서 정의된 <Union of Geometry Hierarchy> : shpere-01 links Group-01 부분의 역할이다. <Union of Geometry Hierarchy>는 여기서 그룹을 표시하기 위한 부분이 아니라 상위 그룹에 대한 개별 객체(여기서는 sphere-01)의 링크 정보를 나타내기 위함이다. 즉 <Union of Geometry Hierarchy> 아래의 또 하나의 <Union of Geometry Hierarchy>는 그 하위의 그룹핑을 의미할 수도 있지만 여기서처럼 상위 그룹에 대한 링크를 의미할 수도 있다. 이것의 구분은 <Union of Geometry Hierarchy>에 <LSR Transformation>의 정보가 있느냐 없느냐로 구분된다. <LSR Transformation>이 존재한다면 그것은 그룹의 Pivot Point를 나타내주는 것이므로 그룹을 정의하는 것으로 볼 수 있고, 만일 존재하지 않는다면, 단순한 상위 그룹이나 객체에 대한 링크를 의미한다고 정의할 수 있다. 이것은 STF를 해석하는데 있어서도 자연스러운 것이며, 원래의 DRM의 계층구조를 해석하는데 있어서도 자연스럽다.

6. 구현

본 프로젝트의 결과물인 "3DS MAX To STF Exporter"는 3DS MAX plug-in의 규약

```
<Union of Geometry Hierarchy> // Scene
    <Union of Geometry Hierarchy> // Group-02
    <LSR Transformation>
        <Geometry Model Instance> // Model torus-01
        <LSR Transformation>
        <Geometry Model Instance> // Model cone-01
        <LSR Transformation>
            <Geometry Model Instance> // Model sphere-02
            <LSR Transformation>
                <Union of Geometry Hierarchy> // Group-01
                <LSR Transformation>
                    <Geometry Model Instance> // Model box-01
                    <LSR Transformation>
                    <Geometry Model Instance> // Model cylinder-01
                    <LSR Transformation>
            (1)-----<Union of Geometry Hierarchy> // shpere-01 links Group-01
                        <Geometry Model Instance> // Model shpere-01
                        <LSR Transformation>
                            <Geometry Model Instance> // Model cone-02
                            <LSR Transformation>
```

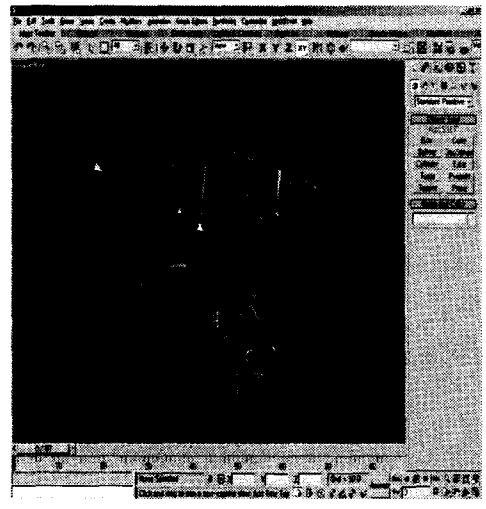
< 표 1 > 그룹과 링크에 대한 DRM 구조

에 의하여 작성했으며, 기능은 3DS MAX의 scene에 있는 그래픽 객체중 기하 객체들의 내용과 부모/자식 계층구조의 의미(semantic)의 변환에 중점을 두었으며, 외관(material)에 관해서는 기본적인 기능만 구현했다. 3DS MAX나 MAYA 등 모든 3D 그래픽 저작도구들의 외관 부분은 화려한 랜더링을 위하여 상당히 자기들만의 노하우가 담긴 부분으로 모든 외관의 완벽한 변환은 사실상 불가능하며 크게 의미도 없는 일이기 때문이다. 따라서 이번 Exporter에서 외관은 객체의 와이어-프레임 칼라와 비트맵 형식(JPEG)의 텍스춰 맵만 export하였다.

다음의 수행 실례는 3DS MAX 4.2에 3DS MAX To STF exporter 플러그인"을 설치하고 화면에 그린 그림을 실제로 STF 파일로 exporting하는 예제이다. 다음과 같은 순서로 하면 STF로의 export를 할 수 있다.

(1) 3DS MAX 4.2 로 그림 그리기

이 단계에서는 디자이너가 3DS MAX의 기능을 이용하여 <그림 9>와 같이 3DS MAX의 캔버스상에서 그림을 직접 그리거나, 기존의 max 파일을 화면으로 로드한다.

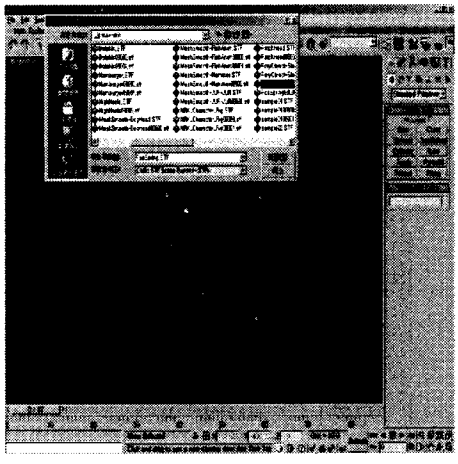


< 그림 9 > 3DS MAX의 canvas에 그림을 그린 화면

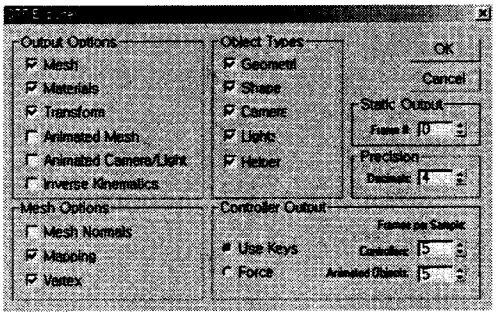
(2) STF 형태로 Export 수행

디자이너가 MAX의 캔버스상에 그림을 다. 그

린 후 MAX의 메뉴에서 "File/Export"를 선택하면 <그림 10>과 같이 파일형식과 저장할 STF 파일의 이름을 선택하고 입력하는 "Select File to Export" 대화상자가 나타난다. 여기에서 파일형식은 "STF Scene Export(*.STF)"를 선택하고, 저장위치에서 결과 STF 파일이 저장될 디렉토리를 선택한 후, 파일이름 입력란에는 원하는 파일명을 넣어주고, "저장(S)" 버튼을 눌러주면 <그림 11>과 같이 "STF Exporter" 대화상자가 나타나는데 여기서 "OK" 버튼을 눌러주면, STF 형태로의 exporting이 시작되고 지정한 디렉토리에 지정한 STF 파일이 생성된다.



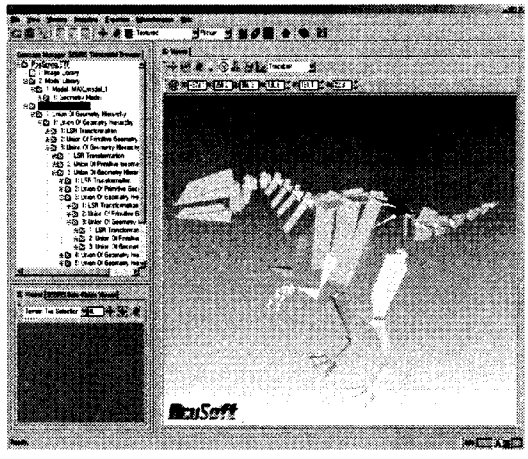
< 그림 10 > STF Export 파일타입과 파일명 설정 화면



< 그림 11 > STF Export 대화상자 화면

(3) Export되어 생성된 STF 파일 확인

export된 결과는 사용자가 지정한 디렉토리와 파일명에 따라 STF 형태로 저장이 되어있는데, 이것을 Side-By-Side 브라우저로 로드하여 확인한다.(<그림 12> 참조)



< 그림 12 > Side-By-Side browser로 STF로 Export된 결과 확인하기.

7. 결론

SEDRIS는 국방 분야뿐만 아니라 각종 멀티미디어, 지리정보시스템, 그래픽 등 산업분야 등에서의 적용을 적극적으로 시도하고 있다. 특히 시스템간의 데이터 교환을 위한 상호 운용성을 제공하기 위한 근본적인 요소를 제공한다.

본 논문에서는 SEDRIS의 표준 공통 교환 포맷으로의 특징을 이용하여 상용 포맷을 교환 포맷으로, 교환 포맷을 상용 포맷으로 변환하기 위한 컨버터를 개발하였다.

상용 포맷인 3DS MAX 자료를 교환 포맷으로 변환할 때 가장 중요한 점은 포맷 내용의 물리적□□논리적 의미 즉 semantic을 손실없이 보존하는 것이다. 따라서 본 연구 결과인 "3DS MAX To STF Exporter"에서도 이러한 기하 객체들간의 "semantic hierarchy"를 유지하는 것을 가장 중요한 핵심으로 삼았다

참고 문헌

[1] Defense Modeling and Simulation Office, "Synthetic Environment Data Represention and Interchange Specification Overview,", Mar. 1998.

[2] Michael R. Welch, "SEDRIS As An Interchange Medium," Orion Development Group, Inc. Mar. 1998.

[3] John E. Carswell,

"Fundamentally SEDRIS: The Technology Components," Aug. 2002.
[4] Boris Kulagin,
"3ds max 4 : from objects to animation", Charles River Media, 1998
[5] Mark Adams, Erick Miller, Max Sims
"Inside Maya 5" , New Riders, 2001
[6] ISO/IEC JTC1/SC24,
"SEDRIS Environmental Data Representation and Interchange Specification (SEDRIS)-
Part 1: Functional specification." Jun. 2002.

이 광형

1995 고려대학교 전산과학과 졸업(이학박사)
1996.2 ~ 1997.1 한국표준협회 선임연구원
1997.3~ 2000.2 동서대학교 컴퓨터공학과 전임강사
2000.3 ~ 2002. 3 (주)ECO 시스템개발실 부장
2002.4~ 003. 6 (주)코딕커뮤니케이션즈 책임연구원
현재 : 고려대학교 강사
관심분야: 컴퓨터 그래픽스, CBD, 병렬/분산시스템