가상환경 내의 개체 행위 설계*

서진석 김정현
포항공과대학교 전자계산학과 가상현실감 연구실

Behavior Modeling of Entities in a Virtual Environment

Jinseok Seo G. Jounghyun Kim
VR Lab, Dept. of Computer Science & Engineering, POSTECH

요 약

가상현실 시스템을 구현하고자 할 때, 그 환경에 존재하는 개체(entity)들의 복잡한 행위(behavior)를 설계 및 기술하는 것이 쉽지 않을 뿐만 아니라, 사용자와의 상호작용까지 고려한다면 매우 복잡하게 된다. 게다가, 이러한 행위와 상호작용의 설계 및 기술에 대한 정형화된 도구조차 찾기 어려운 것이 사실이다. 이 논문에서는 이러한 행위를 설계할 때, 소프트웨어 공학에서 이용되고 있는 도구 중 하나인 Statecharts와 DFD(Data Flow Diagram)을 기반으로 하는 ASADAL/SIM을 이용하여 설계하고 구현한 예를 들어 프로그래밍이 아닌 다른 도구를 통한 가상환경 내의 개체들의 행위 설계와가 가능함을 보이고, 그로 인해 얻을 수 있는 여러 장점에 관하여 소개한다.

1. 서 론

최근 세계적으로 많은 곳에서 가상현실 시스템에 관한 연구가 활발하게 이루어지고 있다. 그 중에서도 특히 인간과 컴퓨터의 상호작용(Human-Computer Interface)에 그 관심이 많이 모아지고 있는데, 실시간 렌더링에(rendering)에 관한 연구와 사용자에게 몰입감을 줄 수 있는 입출력 에 관한 연구가 주된 것이다.

그리고, 가상환경 내에 존재하는 개체들의 행위 및 사용자와의 상호 작용에 대한 연구가 있다. 즉, 인간의 시각, 청각 및 촉각에 충실한 시스템이 중요한 것은 사실이지만, 어떤 개체의 행위와 사용자, 개체간의 상호작용 또한 사용자에게 현실감과 몰입감을 줄 수 있는 중요한 요소가 될 수 있다는 근거에서 나온 연구이다

이러한 개체들의 행위 및 상호작용을 설계하고 기술할 때, 많은 경우 프로그래밍 언어나 스크립트 언어에 의존하여 왔다. 그러나, 이러한 언어들의 특성상 실세계에 존재하는 개체들의 동적인 행위를 기술하기엔 많은 노력과 어려움이 따르고, 그 행위를 검증하고 개량(refinement)할 필요가 있을 경우에는 더욱 많은 노력이 필요하다.

본 논문은 프로그래밍 언어나 스크립트 언어에 의존하지 않는, 다른 접근방법으로 개체의 행위를 설계하고자 했던 시도의 과정과 결과를 소개한다.

구성은 두 번째 절에서 본 연구와 관련된 연구들을 소개, 세 번째 절에서 ASADAL/SIM[1]을 이용하여 실제 설계에 응용한 예, 마지막으로 네 번째 절에서 결론을 맺는다

2. 관련 연구

가상환경 내의 개체의 행위나 상호작용에 관한 많은 연구 중 대부분은 프로그래밍 라이브러리나 툴킷(toolkit)을 이용한 설계나 VRML을 확장한 스크립트를 이용한 설계이다.

그중, James F. Cremer는 가상환경의 시나리오를 저작하는 데에 유한상태기(Finite State Machine)를 운전 시뮬레이터에 적용하였다[2]. 그러나, 표준 상태전이도(State Transition Diagram)는 큰 시스템을 설계할 경우, 상태의 수가 많아지기 때문에 도식이 비대해지고 복잡해져 그 설계를 변경 및 수정하기가 어려울 뿐만 아니라, 추상화기능 및 concurrency를 제공하기 어렵다.

이에 유한상태기에 계층(hierarchy)적 구조와 concurrency기능을 추가하여 운전 시뮬레이터에 존재하는 개체들의 행위를 설계하였다.

그러나, 가상환경에 존재하는 개체들과 같이 복잡한 시스템을 설계

* 본 연구는 한국과학기술연구원(KIST)의 G7 감성공학 위탁과제로 수행되었음.

하기에 유한상태기와 상태전이도는 근본적으로 부적합하기 때문에 위와 같이 계층과 동시발생기능을 추가했다고 해도 해결하지 못하는 부분이 있다[4]. 특히, 각 유한상태기 사이의 통신이 부자연스럽고 concurrent한 유한상태기간의 관계도 명확하지 않다.

다음절에서 말하는 Statecharts를 기반으로 하는 ASADAL/SIM은 이러한 부분을 자연스럽게 해결할 수 있다.

3. ASADAL을 이용한 개체 행위 설계

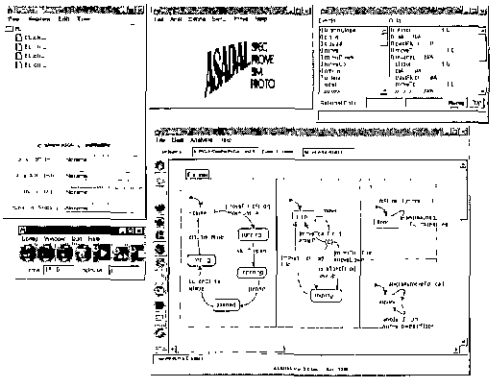
본 연구의 목적은 포항공과대학교 소프트웨어공학 연구실에서 개발한 ASADAL을 이용하여 가상환경 내에 존재하는 개체의 행위를 설계하고 기술하는 것이다 본래 ASADAL은 실시간 시스템을 위한 환경으로써, 실시간 시스템의 명세와 분석에 대한 방법 및 이를 뒷받침해줄 수 있는 CASE(Computer-Aided Software Engineering) 도구를 포함한다 그 구성요소로는 여러 가지가 있는데 본 연구에서는 그 중 시뮬레이션과 분석을 위한 도구를 제공하는 ASADAL/SIM을 이용하였다.

3.1. Statecharts

Harel의 Statecharts는 표준적인 상태전이도에 Depth, Orthogonality, Broadcast Communication을 더한 것이라고 볼 수 있다[3].

Depth는 상태를 나타내는 Blob들의 포함관계로 표현한다. 이는 계층적 구조를 자연스럽게 추가하여 추상화 및 캡슐화기능을 제공하여준다. Orthogonality는 상태들의 집합을 Cartesian Product로 표현할 수 있는 집합들로 분할(partition)하여 concurrency를 표현할 수 있다 Broadcast Communication은 모든 상태들이 받아들일 수 있는 출력을 가능케 함으로써 공통의 이벤트나 조건(condition)으로만 가능했던 concurrent한 상태들의 동기화 및 통신기능을 확장시켜 준다.

3.2. ASADAL/SIM



(그림 1) ASADAL/SIM의 실행화면

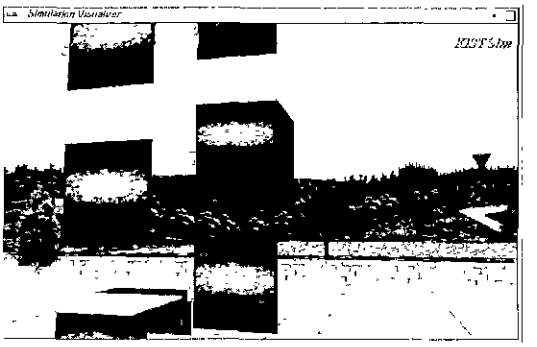
ASADAL/SIM(그림 1)은 Statecharts를 확장한 TES(Time-Enhanced Statecharts)와 DFD를 기반으로 명세된 실시간 시스템을 시뮬레이션 및 분석할 수 있는 도구이다.

그림에서 볼 수 있듯이 여러 개체에 해당하는 TES를 동시에 시뮬레이션할 수 있으며 concurrent한 여러 상태들의 전이 상황을 잘 보여준다. 사용자는 필요에 따라 시뮬레이션을 잠시 멈추거나, 한 스텝씩 진행하게 할 수도 있으며, 임의의 이벤트나 데이터의 변화를 발생시킬 수 있다 또한, 시스템의 부하가 허락하는 한 여러 개의 개체에 해당하는 TES나 같은 개체라도 여러 개의 인스턴스(instance)를 동시에 시뮬레이션할 수 있다

DFD로는 프로세스간 데이터의 처리나 흐름을 설계하면, TES와 연동하여 복잡한 계산이나 처리가 필요한 시뮬레이션도 가능하다.

3.3. 엘리베이터 구현

실제 가상환경에 적용하기 위하여 우선 엘리베이터라는 특정 개체를 선택하였다. 엘리베이터는 가상현실 시스템 중에서도 특히 건물등의 항해시스템(Navigation System)에 흔히 등장하는 개체로 엘리베이터 자체의 행위가 생각보다 매우 복잡할 뿐만 아니라 사용자와의 상호작용을 설계하기란 그리 쉬운 일이 아니다.



(그림 2) 항해시스템(Navigation System)중 엘리베이터*

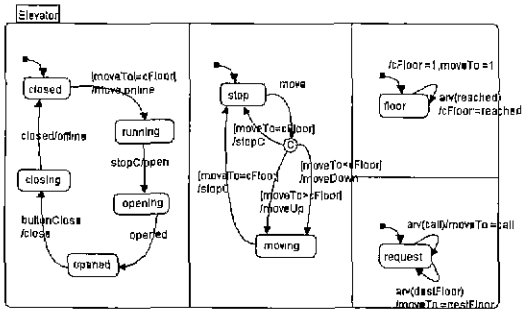
항해시스템에는 이미 모델링 되어 있는 엘리베이터의 데이타와 이 엘리베이터를 움직일 수 있는 기본적인 함수(MoveUp, MoveDown, OpenDoor, CloseDoor등)들이 구현되어(그림 2) 있고, ASADAL에는 이러한 기본함수들을 바탕으로 엘리베이터의 행위를 설계한 TES가 존재한다 실제로는 같은 엘리베이터 두 개를 동시에 시뮬레이션하였다.

통신 부분은 TCP 및 Datagram 소켓(Socket)으로 구현하였다 클라이언트/서버 시나리오 상으로는 ASADAL측이 서버가 되지만, ASADAL이 없더라도 항해가 가능한 시스템이기 때문에, 초기엔 ASADAL이 Datagram 소켓으로 항해시스템에 접속을 시작하여 서버의 존재를 알려주고 TCP 서버 소켓을 생성시키면, 다시 항해시스템이 ASADAL에 접속을 하는 형식의 프로토콜을 이용하였다.

다음(그림 3)은 엘리베이터의 행위를 설계한 TES의 예이다. 전체 blob을 네 부분으로 나눈 신들은 Statecharts의 concurrency를 표현하는 것이다 좌측의 blob은 엘리베이터의 문이 열리고 닫히고 움직이는 상태들의 동작 cycle을 설계한 것이고, 중앙의 blob은 엘리베이터의 현재 층과 움직여야 할 층을 비교하여 층 사이로 움직이고 있는 상태와 멈춰 있는 상태를 전이하는 과정을 설계한 것이다 우측 하

* 항해시스템 및 엘리베이터의 동작함수들은 KIST에서 구현하였음

단의 blob은 외부로부터의 어떤 요청을 처리하는 blob이다. 여기에서 요청은 흥해 시스템으로부터 발생하는 여러 이벤트에 해당한다고 볼 수 있다.

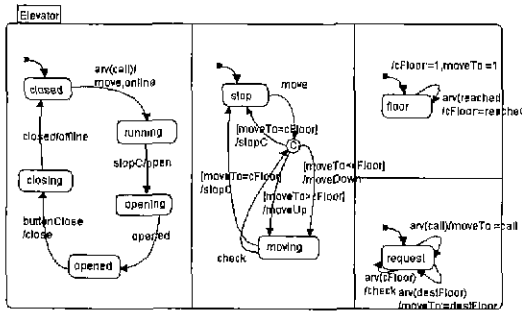


(그림 3) 엘리베이터에 대한 TES

실행 결과 엘리베이터는 두 가지 오류를 제외하고는 사용자와의 상호작용에 의해 잘 작동하였다. 그 첫 번째 오류는 설계자가 초기 엘리베이터의 행위를 분석할 때 착오로 일으킨 것으로 위 TES(그림 3)의 좌측 blob에서 "closed"상태에서 "running"상태로의 전이를 자세히 살펴보면 알 수 있다. 즉, 현재 층과 움직여야 할 층이 다를 경우에만 "running"상태로 갈 수 있고, "running"상태로 가야만 문이 열리는 "opening"상태로 갈 수 있는 것이다 이는 실제 엘리베이터 상자는 사용자와 같은 층에 있지만 아직 사용중이지 않을 경우, 사용자가 아무리 "call"버튼을 눌러도 엘리베이터가 작동하지 않는 상황을 초래할 수 있다

두 번째 오류는 엘리베이터의 행위를 설계한 설계자와 항해시스템의 엘리베이터의 기본동작에 관한 함수를 구현한 프로그래머간의 오류라고 할 수 있다. 엘리베이터의 동작에 관한 함수를 구현한 프로그래머는 "MoveUp"또는 "MoveDown"과 같은 이벤트를 한층 만 올라가거나 내려가는 것으로 해석하여 구현하였으나, 위 TES(그림 3)을 보면, 이를 작성한 설계자는 목표 층으로 도달할 때까지 계속 동작하는 것으로 설계한 것을 알 수 있다.

위와 같은 오류들은 실시간 시스템의 행위 설계뿐만이 아니라 가상환경 내에 존재하는 개체들의 행위를 설계할 때 흔히 발생할 수 있는 것이다. 다음(그림 4) 위 오류를 수정한 TES의 한 예이다. 첫 번째 오류를 고치기 위해, 사용자로부터의 "call"요청이 오면 엘리베이터가 움직이기 시작하여 사용자가 있는 층으로 움직일 수 있도록 수정하였고, 두 번째 오류를 고치기 위해, 프로그래머의 의도대로 "MoveUp"과 "MoveDown"을 한 층씩만 움직이는 것으로 해석하여 현재 층에 대한 데이터가 올 때마다 목표 층과 비교하여 "MoveUp"과 "MoveDown"이벤트를 발생시키는 것으로 설계하였다.



(그림 4) 오류가 수정된 엘리베이터의 TES

오류를 수정할 수 있는 방법은 설계자에 따라 수없이 많이 존재한다 두 시스템간의 인터페이스(interface)나 통신 프로토콜(protocol)의 변화를 주지 않은 한도 내에서는 재 컴파일이나 링크의 과정 없이, 단지 도식을 마우스로 수정함으로써 설계의 자유로운 변경이 가능하다

만일 프로그래밍이나 스크립트 언어를 이용하였다면, 설계 자체를 이해하기가 어려울 뿐만 아니라 이의 검증 및 수정은 더욱 복잡하였을 것이다

4. 결 론

본 논문에서는 가상환경 내 개체들의 행위 및 사용자와의 상호작용을 Statecharts와 DFD를 기반으로 하는 ASADAL/SIM을 이용하여 설계, 기술하는 것에 대하여 소개하였다.

설계자는 표현의 한계성과 어려움이 있는 프로그래밍 언어나 스크립트 언어가 아닌, 직관적으로 이해하기 쉽고 표현이 자유로운 도식을 이용하여 복잡한 개체의 행위와 상호작용을 설계할 수 있다. 그리고, 이미 설계되어 있는 TES를 보고도 오류를 쉽게 발견하고 수정할 수 있음은 위 예로부터도 알 수 있다 물론, 단순한 설계로부터 복잡한 설계로까지 개량(refinement)해가면서 시뮬레이션하는 점진적 설계도 가능하다.

본 연구에서는 아직 시도해보지 않은 것으로써, 복잡한 항해시스템에서의 경우 시뮬레이션에 의한 시스템의 부하를 줄이기 위해 Deborah[5]가 말하는 "Simulation Levels of Detail"을 적용시켜 볼 수 있지만, 각 레벨간의 교환(switching)시 해결해야 할 것들이 남아있다. 또한, JAVA언어로 구현되어 있는 ASADAL을 CORBA를 이용, 서로 상이한 여러 시스템에 분산시키면 매우 큰 시스템이라도 쉽게 동적으로 시뮬레이션할 수 있을 것으로 본다.

참고 서적

[1] Kyo C. Kang, ASADAL, *http://rao.postech.ac.kr/~realtime/*

[3] James F. Cremer and Joseph K. Keaney. Scenario authoring for virtual environments *Proceedings of the IMAGE VI Conference*, June 1994.

[3] David Harel. Statecharts· A visual formalism for complex system. *Science of Computer Programming*, 8(3):231-274, June 1987

[4] David Harel On Visual Formalism. *Communicatoins of the ACM*:514-530, May 1988

[5] Deborah A. Carlson and Jessica K Hodgins, Simulation Levles of Detail for Real-Time Animation, *Proceedings of Graphics Interface '97*, 1997