1

모델링 시뮬레이션 공학

한국과학기술원 ▌김 탁 곤*

1. 서 론

최근 들어 우리나라에서도 시스템 모델링 시뮬레이션(M&S: Modeling and Simulation)에 대한 관심이 높아지고 있다. 이것은 우연이 아닌 필연적인 것으로 국민소득이 2만 불 시대에서 나아가 선진국 진입을 위한 산업 구조의 변화와 맥을 같이 한다. 이러한 산업 구조는 기존 시스템의 복사/개조 기술에서 새로운 시스템 설계/개발 기술로 진화가 일어나게 되며 이러한 진화를 위한 핵심 기술이 시스템 M&S 기술이다. M&S 기술은 시스템을 구성하는 부 시스템들인 하드웨어(전자, 기계, 장치 등), 소프트웨어(알고리즘, 프로세스 등), 휴먼웨어(운용 개념, 에이전트 등) 및 바이오웨어(환경, 생명 과학 등) 등에 각각 독립적으로 적용될 뿐 아니라 이들 부 시스템을 결합한 전체 시스템의 설계/개발/분석/획득 등에도 적용되어야 할 것이다.

시스템 설계/개발/분석/획득 시 M&S 적용은 시스템 공학 이론에 기반 한 표준화된 프로세스 내에서 이루어져야 한다. 이러한 관점에서 M&S는 독자적인 학문 영역으로 분류할 수 있으며 본 저자는 이러한 학문 영역을 모델링 시뮬레이션 공학(M&S Engineering)이라 부르고 있으며 약어로 M&S 공학으로 명명하고 있다. M&S 공학은 학술적으로 명확히 정의된 바는 없지만 본 저자는 그림 1에 제시한 M&S 전 프로세스의 각 단계별 업무(Activity)에서 필요한 이론, 방법론 및 응용을 다루는 다 학제적(Multi-disciplinary) 학문을 M&S 공학으로 정의 하고자 한다. 따라서 M&S 공학 발전을 위해서는 M&S 공학 자체에 대한 다 학제적 이론 연구와 아울러 응용 분야에서 M&S 기술을 적용하는 응용 연구가 균형 있게 이루어져야 할 것이다. 아울러, M&S 공학 발전에 무엇보다 중요한 것은 M&S 공학의 필요성에 대한 깊은 인식과 저변 확대 및 M&S 공학 적용에 대한 제도적 뒷받침 등이 보다 적극적으로 이루어져야 할 것이다. 그러나 M&S 기술을 도구로 이용하고 있는 많은 공학/과학 분야에서 M&S 공학 자체를 독자적 학문 분야로 이해하지 못하고 있는 것은 M&S 공학 발전을 위해서 매우 안타까운 일이다.

본 논문은 M&S 공학의 정의, 모델링 이론, 시뮬레이션 방법론/환경 및 응용 등에 대하여 간략히 기술한다. 2장에서는 M&S 공학 정의 및 시스템 모델 분류를 소개하고, 3장에서는 M&S 공학의 핵심이 되는 모델링 형식론 및 시뮬레이션 방법론을 소개한다. 4장에서 M&S 공학 연구 분야를 소개하고 5장에서 결론을 맺기로 한다.

2. M&S 공학 및 시스템 모델 분류

M&S 공학은 그림 1에 제시한 M&S 전체 프로세스의 각 단계에서 필요한 이론을 연구하는 다 학제적 학문이라고 정의하였다. 본 절에서는 M&S 공학 세부 분야에 대한 개요 및 M&S 대상 시스템의 분류에 대하여 기술한다.

2.1 M&S 공학

M&S 공학은 모델링/시뮬레이션을 이용한 문제 해결의 전 과정과 이들 전 과정에서 일어나는 각 단계별 주제를 연구하는 학문이다. 그림 1은 M&S 전 과정과 관련 이론 및 기술을 나타내었고 아울러 각 과정에서 필요한 전문 지식/전문가를 나타내었다. 첫 번째 관련 분야(Related discipline)인 요구공학(Requirement engineering)은 대상 시스템에 대한 M&S의 목적을 정립하고 이로부터 모델링 요구 사항을 도출해 내는 과정에서 필요한 학문이다[2]. M&S 목적 정립은 M&S 전 과정에서 가장 중요한 과정이다. 왜냐하면 모델링은 대상 시스템 전체를 100% 표현하는 것이 아니라 미리 설정된 M&S의 목적에 맞도록 시스템을 추상화(Abstraction)하는 과정이다. 따라서 똑같은 대상 시스템에 대하여 M&S의 목적이 달라지면 시스템 모델이 달라질 수밖에 없다. 예를 들면, 컴퓨터 시스템을 모델링하는 경우, 계산 능력 예측이 목적인 경우와 동작 가능한 온도 범위 예측이 목적인 모델은 완전히 다른 모델이 될 것이다. 즉, 한 개의 시스템에 대하여 M&S의 목적에 따라 수많은 모델이 존재하게 된다. M&S

* 종신회원

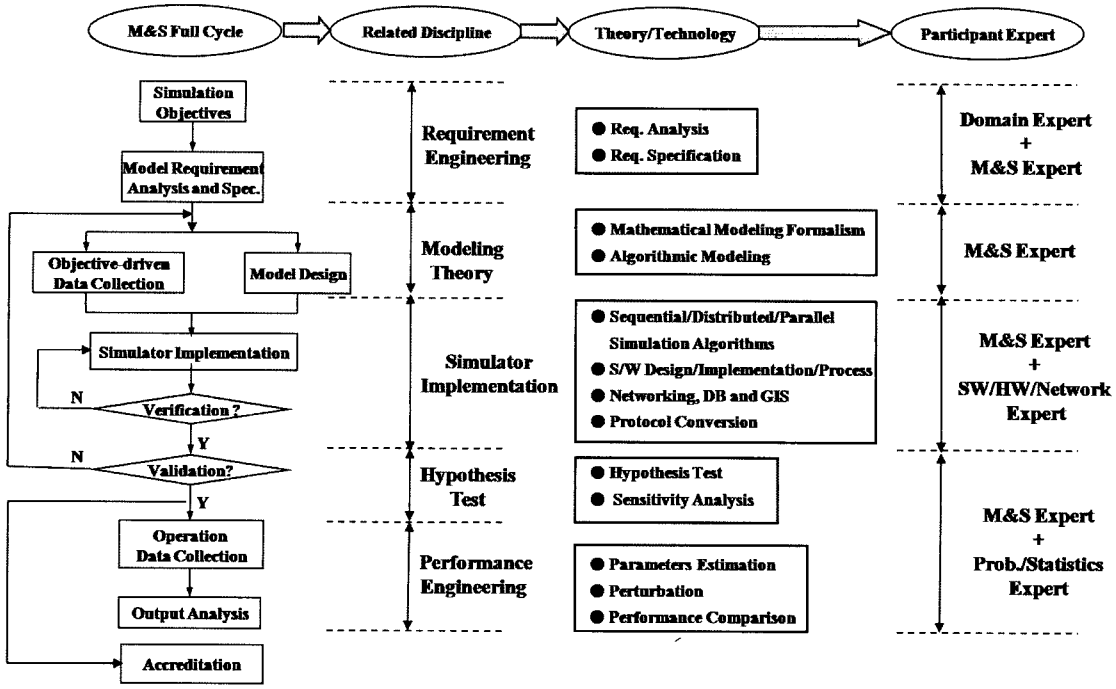


그림 1 M&S 공학 범위 및 관련 전공 분야[1]

목적에 맞는 모델링 과정을 모델링 이론에서는 목적 지향적 모델링(Objective-driven modeling)이라고 부르며 객체지향적 모델링인 경우 OPI(Objective-Performance Index) 매트릭스를 이용한 목적 지향적 모델 개발 방법론이 제안된 바 있다[3]. 이러한 관점에서 M&S 공학을 위한 요구 공학의 역할은 목적지향적 모델 요구 사항 분석법 및 요구 사항 명세 방법론 제공이 될 것이다.

두 번째 관련 분야인 모델링 이론(Modeling Theory)은 대상 시스템의 행위를 모델로 표현하는 기법에 대한 이론이다. 수학적 형식론(Mathematical formalism)은 대상 시스템의 행위를 표현할 수 있는 수학적 의미론(Semantics)에 기반한 모델링 틀(Framework)을 제공 한다. 수학적 모델링은 대상 시스템의 전체 행위 중 목적에 맞는 행위만을 추상화하여 형식론이 제공하는 틀에 집어넣는 것이다. 대표적인 모델링 틀로서는 연속시스템의 모델링 틀인 미분방정식(Differential Equation)과 이산사건 시스템의 모델링 틀인 DEVS(Discrete Event Systems Specification) 형식론이 있다[4]. 아울러 이산사건 시스템 모델링의 경우 수학적 형식론에 기반하지 않고 알고리즘으로 시스템 행위를 추상화 시키는 모델링 기법도 가능하다. 대표적인 알고리즘적 모델링 기법은 사건중심 모델링(event-oriented modeling)으로 이산사건 시스템의 행위를 사건 중심의 알고리즘으로 추상화 시킨 것으로 SIMSCRIPT와 같은 시뮬레이션 언어에서 사용하는 방법이다[5].

세 번째 관련 분야인 시뮬레이터 구현(Simulator Implementation)은 모델링 이론으로부터 얻어진 모델들을 효율적으로 시뮬레이션 하는 알고리즘과 모델 및 이들 알고리즘의 프로그램 구현에 관한 것이다. 연속시스템의 미분방정식 모델의 경우 미분방정식의 수치해석 알고리즘들이 연구 대상이며 이산사건 시스템의 DEVS 모델인 경우 DEVS 모델의 효율적인 시뮬레이션 알고리즘이 연구 대상이다. 이러한 시뮬레이션 알고리즘은 컴퓨팅 환경이나 가용 자원에 따라 순차적/분석/병렬 알고리즘 형태로 개발되어 진다[35]. 시뮬레이션 알고리즘이 구해지면 이들은 모델과 함께 프로그램으로 구현되어야 시뮬레이터가 만들어진다. 따라서 시뮬레이터 구현을 위해서는 소프트웨어 기술과 더불어 네트워킹, 데이터베이스(DB), 지형정보 시스템(GIS) 등에 대한 기술이 필요하다. 아울러 여러 개의 다른 기종 시뮬레이터 간의 연동을 위해서는 프로토콜 변환 기술이 필요하게 된다[6]. 구현된 시뮬레이터가 모델 명세를 잘 반영하는지를 확인하는 과정인 모델 검증(Model Verification)은 프로그램 디버깅과 동일한 과정으로 모델 명세서로부터 테스트 입력과 대응 출력을 구하여 시뮬레이터를 테스팅하면 된다[7].

네 번째 관련 분야인 가설 시험(Hypothesis Test)은 통계학에서 사용되는 가설 시험 기법을 M&S 분야에 적용하여 모델/시뮬레이터 실증(Model Validation)에 사용하는 것이다. 즉, 모델/시뮬레이터 실증은 실제 시스템에서 수집한 데이터와 이에 대응하는 시뮬레이터 생성 데이터가 같은 지를 통계학적으로 비교하는 과

정이다[8]. 그런데, 문제가 되는 것은 실제 시스템의 어떤 데이터와 시뮬레이터에서 생성된 데이터를 비교할 것인가이다. 여기서 우리는 앞서 설명한 목적 지향적 모델링 이론을 상기할 필요가 있다. 즉, 데이터 수집을 위해서는 시뮬레이션 목적 수립이 선행되어야 하고 시뮬레이션 목적에 맞는 데이터를 실제 시스템에서 수집 하여야 한다. 아울러 이러한 데이터를 발생시킨 실제 시스템의 입력 시나리오를 시뮬레이터에 입력하여 대응하는 출력 데이터를 얻어야 한다. 위와 같은 통계적 모델 실증 방법 이외에도 도메인 전문가와 협조하여 민감도 분석(Sensitivity Analysis)을 통한 모델 실증도 사용되고 있다. 민감도 분석은 모델 파라미터의 변화에 따른 시뮬레이터 출력 값의 변화를 관찰하는 기법으로 이론적으로 완전한 실증은 불가능 하다. 예를 들면, 대기 행렬 시스템(Queuing System) 모델에서 고객의 도착율(Arrival Rate)이 일정한 경우 시스템 내부 처리 속도(Service Rate)가 낮을수록 고객의 평균 대기 시간은 길어지게 된다. 따라서 시뮬레이션 결과 일정한 도착율 하에서 내부 처리 속도를 점차 낮추어 가면서 평균 대기 시간을 관찰한 결과 평균 대기 시간이 점점 짧아졌다면 모델 내부에 오류가 있다는 것을 의미한다.

마지막으로 M&S와 관련된 분야는 성능공학(Performance Engineering) 분야이다[9]. 시뮬레이션 모델을 이용한 성능 측정 과정을 통계적 관점에서 보면, 한 번의 시뮬레이션 결과는 통계적 분석을 위하여 형성된 모 집단에서 한 개의 샘플을 추출하는 것과 같은 의미이다. 신뢰도 높은 성능 지수를 얻기 위해서는 통계적 분석에서 많은 샘플을 추출하는 것과 마찬가지로 M&S에서는 시뮬레이션 횟수를 증가시켜야 한다. 성능 지수는 이렇게 반복 시뮬레이션을 통해서 수집된 데이터 군을 통계적 대표 값으로 요약한 평균값과 이 값의 신뢰도를 나타내는 신뢰 구간으로 표현된다. 섭동 해석(Perturbation Analysis)은 시스템 파라미터의 불규칙적인 변화에 따른 성능 지수를 예측하는 해석법으로 성능 지수의 통계적 대표 값을 여러 번의 시뮬레이션 과정을 거치지 않으면서 구할 수 있는 대안이다[36]. 아울러, 여러 개의 시스템 성능을 비교하는 경우는 상대적 우월성을 비교하거나 이들의 우수성 순위(Ranking)를 결정하는 것도 필요하다[10].

2.2 시스템 모델의 분류

모델링은 대상 시스템을 모델링 목적에 부합되는 추상화 수준에 맞게 각기 다르게 표현할 수 있다. 시스템(혹은 체계)이란 용어는 여러 가지 의미에서 일반적으로 사용되고 있지만 학술적인 정의는 아래와 같다.

[정의] 시스템(System)

각기 다른 기능을 갖는 한 개 이상의 부 시스템(컴포넌트)를 결합하여 한 개의 컴포넌트로 단독으로는 수행할 수 없는 주어진 기능을 수행하는 부 시스템의 결합체.

시스템 컴포넌트의 종류에는 하드웨어, 소프트웨어, 사람(운영자), 바이오웨어(바이오 시스템) 등이 존재한다. 실제 시스템은 같은 종류의 컴포넌트들이 결합된 경우도 있지만 일반적으로는 여러 종류의 컴포넌트들이 복합적으로 결합되어 있다. 예로서, 전자회로 시스템은 하드웨어로(아날로그 및 디지털 회로)만 구성되어 있고 컴퓨터 운영체제는 소프트웨어로만 구성되어 있다. 또한, 경제 시스템은 하드웨어(물건, 돈 등), 사람(경제 주체) 및 소프트웨어(경제 정책, 프로세스 등)로 구성된 복합 시스템이다.

동적 시스템(Dynamic System) 모델은 시스템의 입력, 출력, 상태 변수를 정의 한 후 상태 변수가 시간에 따라 변하는 규칙을 기술한 동작 명세서(Behavioral Specification)이다. 시스템 이론에서 상태 변수를 정의하는 이유는 아래의 정의와 같이 현재의 상태 변수의 값이 시스템의 미래 출력을 결정하기 때문이다.

[정의] 시스템 상태(State)

시각 t에서의 시스템 상태는 t'>t에서의 시스템 출력을 유일하게 결정할 수 있는 모든 정보.

시스템의 상태는 시스템 내에서 출력에 영향을 미치는 변수 값들로 이루어지며 현재의 상태 변수 값은 미래의 출력을 위해서 기억(메모리) 될 수 있어야 한다. 예를 들면, 비행 물체에서 상태 변수는 비행체의 "속도", 아날로그 전자 회로에서 상태 변수는 콘덴서 양단의 "전압", 통신 라우터에서 상태 변수는 버퍼에 저장된 "메시지 수", 그리고 컴퓨터 운영 체제에서 상태 변수는 "CPU 상태" 등이 될 수 있다. 상태 변수의 값은 모든 값(실수 값)을 갖는 경우와 이산적인 값을 갖는 두 가지 경우로 표현 된다. 예를 들어, 아날로그 전자회로에서 정의된 상태 변수인 콘덴서 양단의 "전압"은 V = { v | v : 실수 && 0 <= v <= 10 }으로 표현되며 0-10[V] 사이의 모든 실수 값을 가질 수 있음을 의미한다. 반면, 운영 체제에서 정의된 상태 변수 "CPU 상태"는 CPU_Status = { BUSY, FREE }의 두 가지 값으로 표현되며 어떤 순간 CPU 상태는 정의된 두 가지 값 중 한 개의 값을 가지게 된다. 즉, CPU가 작업을 하고 있는 순간은 "CPU_Status = BUSY"로 작업을 하지 않고 대기하고 있는 순간은 "CPU_Status = FREE"

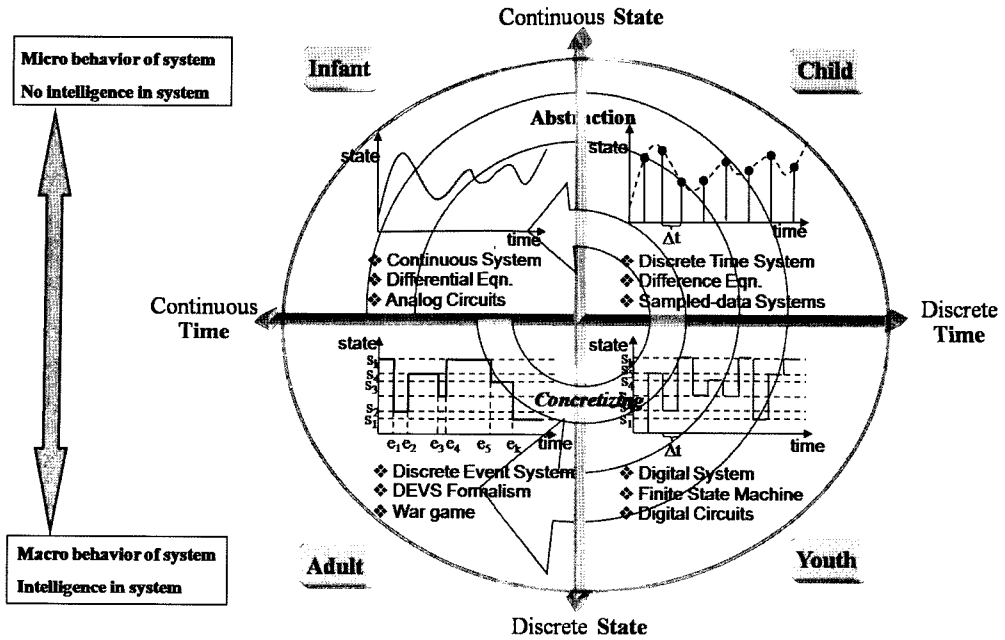


그림 2 시스템 이론적 시스템 분류[1]

로 된다. 마찬가지로, 비행 물체의 상태 변수인 "속도"는 10.45 등의 실수 값이 되며 라우터의 상태 변수인 "메시지 수"는 0, 1, 2 ... 등의 정수 값이 된다.

시스템의 상태는 시간이 경과함에 따라 정해진 규칙에 따라 동적으로 변하게 되며 이런 과정을 상태 천이(State Transition)라고 하며 상태 천이 규칙을 상태 천이 방정식(State Transition Equation) 혹은 간단히 줄여서 상태 방정식(State Equation)이라고 부른다. 따라서 "시스템 모델 = 상태 방정식"으로 귀결되며 모델링은 상태 방정식을 구하는 문제이다. 상태 방정식이 구해지면 원하는 시각에서 시스템 상태를 알 수 있고 따라서 시스템 출력이 유일하게 결정된다. 상태 천이는 시간 함수이다. 시스템 이론에서는 상태 변수와 시간이 가질 수 있는 값의 범위를 각각 연속적인 값(실수 값) 혹은 이산 값으로 나누고 이에 따라 동적 시스템을 분류 한다. 따라서 시스템 종류는 상태 변수 및 시간 값의 표현 방법을 조합한 4가지가 존재하게 되며 그림 2는 이들의 학술 명, 수학적 모델링 형식론 및 예를 나타내었다.

그림 2에 제시한 4가지 시스템 종류는 전공 분야에 따라 다양한 형태로 나타나게 된다. 연속 시스템은 전자공학/기계공학/물리학 전공에서 다루는 전자회로/열역학/유체역학 등이 포함되며 이들은 모두 미분 방정식으로 모델링 할 수 있다. 이산사건 시스템(Discrete Event system)은 산업공학/OR/전산학 전공에서 다루는 생산시스템/물류시스템/소프트웨어 등이 포함되며 이들은 모두 DEVS(Discrete Event Systems Specification) 형식론으로 모델링 할 수 있다[4]. 이산시간 시스템은 연속 시스템의 시간 간격을 일정 간격으로 샘플링 하여 추상화한 모델로서 샘플링 된 두 시간 사이의 상태 변수 값을 표현하지 않는다. 유한 상태 기계(Finite State Machine) 모델은 DEVS 모델에서 시간 정보를 명세하지 않은 특수한 모델 형태로서 시간 관련 정보를 분석하지 못하고 상태 변수의 궤적만을 분석할 수 있다. 그림 2의 시스템 분류는 시스템을 관찰하는 추상화 수준에 따른 분류이다. 따라서 동일한 시스템이라고 하더라도 다른 추상화 수준에서 관찰하게 되면 다른 종류의 시스템이 된다. 현실 세계의 대부분의 시스템은 여러 개의 추상화 레벨을 갖는 복합적인 시스템이 계층적 구조로 구성되어 있다. 그림 3에 예시한 로봇 시스템은 3개의 시스템이 계층적으로 이루어져 있으며 이러한 시스템을 하이브리드 시스템(Hybrid System)이라고 부른다.

3. 모델링 형식론 및 시뮬레이션 방법론

모델링 형식론은 모델을 수학적 방정식 형태로 기술하는 틀(Framework)이며 시뮬레이션은 모델 방정식을 푸는 과정이다. 따라서 주어진 모델(방정식)에 대한 시뮬레이션(방정식 푸는 과정) 방법은 여러 가지가 존재할 수 있으며, 모델링과 시뮬레이션 방법은 완전히 별개의 문제이다. 본 논문에서는 앞서 분류한 4종류의 시스템 중 연속 시스템 및 이산사건 시스템 모델링 및 시뮬레이션 방법론을 간략히 다루기로 한다.

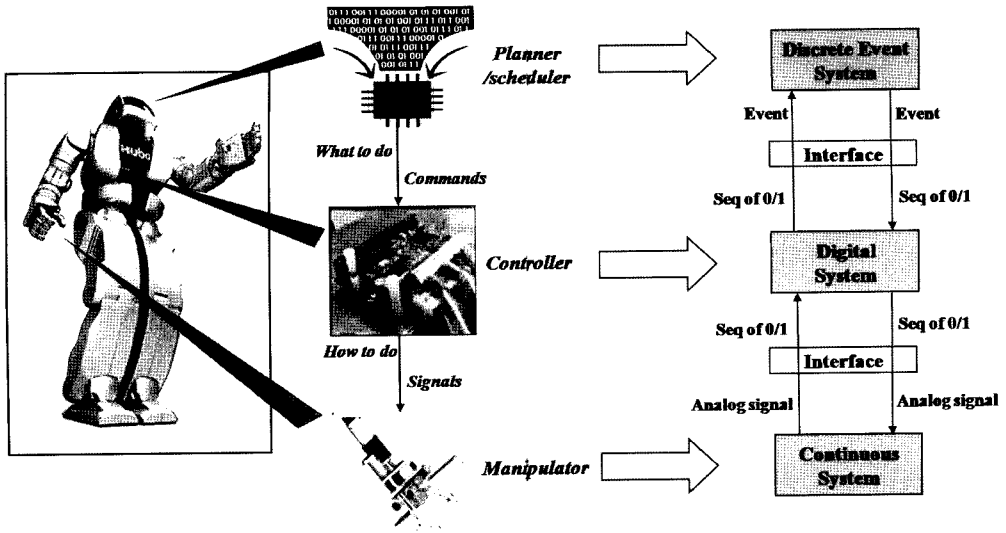


그림 3 하이브리드 시스템 예: 로봇 시스템[1]

3.1 모델링 형식론 = 모델링 공식 = 모델링 틀(Frame-work)

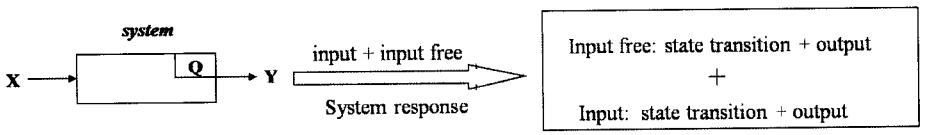
시스템은 외부 입력에 의한 외부 상태 천이와 더불어 입력이 없을 경우라고 할지라도 시스템 내부 조건에 의한 내부 상태 천이가 일어난다. 따라서 상태 방정식은 입력 및 무입력 시 시스템의 외부 및 내부 상태 천이를 나타내어야 한다. 또한, 시스템 이론 기반 상태 방정식은 연속 시스템뿐만 아니라 아래 정의된 이산사건 시스템에도 통일적인 형식(Unified Form)으로 표현되어야 한다.

[정의] 이산사건(Discrete event) 시스템

시스템 내부 혹은 외부 요인으로 인하여 불규칙적인 시간 간격으로 일어나는 이산 사건 발생 시점에서만 상태를 천이하는 시스템.

이산 사건 시스템에서 이산 사건 발생 시점은 외부에서 들어오는 입력 사건과 아울러 입력이 없는 경우 내부적 조건이 만족될 경우 발생하는 내부 사건이 있다. 예를 들어, 프로토콜 시스템의 경우 입력 사건은 "메시지 도착", 내부 사건은 "Time out" 그리고 출력 사건은 "메시지 전송" 등이 될 수 있다. 또한 이산 사건 시스템의 상태 변수는 유한개의 이산 값을 가지게 된다. 예를 들어, 프로토콜의 경우 상태 변수 값은 S = {SEND, WAIT} 등으로 표현된다. 여기서 "SEND"는 메시지를 보내고 있는 상태를 의미하며 "WAIT"는 메시지를 기다리고 있는 상태를 의미한다.

이산사건 시스템의 상태 방정식은 이산 값을 갖는 상태 변수 천이 규칙으로 연속시스템의 상태 방정식인 미분 방정식처럼 닫힌 형태(Closed form)로 표현



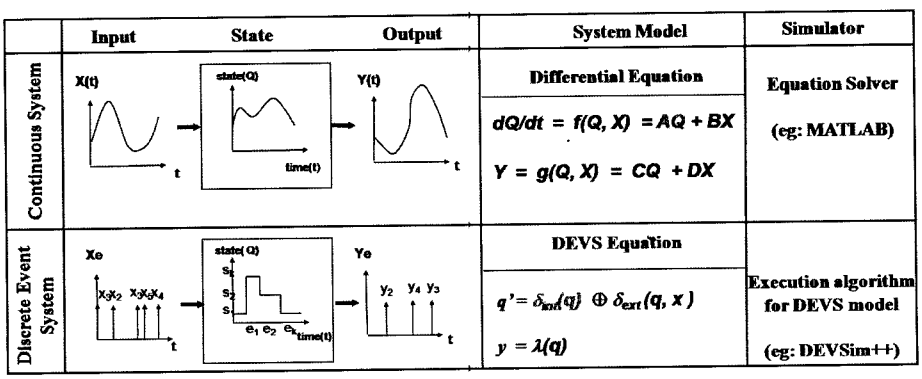


그림 4 연속 시스템과 이산사건 시스템의 상태 방정식[1]

할 수 없다. 이산 값을 갖는 상태 변수의 천이 규칙은 이산 수학(Discrete Mathematics)에 기초를 둔 집합 이론을 이용하면 표현이 용이하다[11]. 이산사건 시스템의 상태 방정식을 집합론에 기반 하여 표현하는 수학적 틀이 DEVS 형식론이다. 그림 4는 연속시스템과 이산사건 시스템의 상태 방정식을 통일적인 형태로 나타내고 있으며 이들은 각각 미분방정식 및 DEVS 방정식으로 그림 4와 같이 표현 된다.

미분 방정식: $dQ/dt = AQ + BX$ (1)

DEVS 방정식: $q' = \delta_{int}(q) \oplus \delta_{ext}(q, x)$ (2)

방정식 (1)에서 Q(t)는 상태 변수의 행렬(벡터)로서 각각의 상태 변수는 실수 값을 가진다. 반면, 방정식 (2)의 q와 q'은 각각 현재 상태 및 다음 사건 발생 시의 미래 상태를 나타낸다. 이산사건 시스템의 상태 변수는 q = (s, r)로 표현되며 s는 이산 상태 변수를 r은 이산 상태 s에서 경과한 시간을 나타내는 실수 값이다. 예를 들면, CPU 상태를 나타내는 상태 변수 q = (CPU_Busy, 10.3)는 CPU가 Busy 상태에서 10.3 time unit 머물고 있다는 것을 의미한다. 방정식 (1)에서 A와 B는 계수 행렬이며 X는 입력 배열(벡터)이다. 방정식 (1)의 dQ/dt = AQ는 무 입력 시 상태 천이를 나타내고 dQ/dt = BX는 입력 시 상태 천이를 나타낸다. 선형 시스템인 경우 상태 천이는 이들 두 상태 천이의 대수적 합, 즉 dQ/dt = AQ + BX로 표현 된다. 마찬가지로, 방정식 (2)에서 $q' = \delta_{int}(q)$는 무 입력 시 상태 천이를 나타내며 $q' = \delta_{ext}(q, x)$는 입력(x) 사건 발생 시 상태 천이를 나타낸다. 방정식 (2)에서 사용된 $\oplus$ 기호는 두 개의 함수 $\delta_{int}(q)$와 $\delta_{ext}(q, x)$를 모두 적용하여 상태 천이를 수행하되 이들을 동시에 수행해야 할 경우가 발생하면 미리 정해진 우선순위에 따라 순차적으로 수행함을 의미한다. 우선순위가 필요한 이유는 $\delta_{int}(q)$를 먼저 수행한 상태 $q' = \delta_{ext}(\delta_{int}(q), x)$와 $\delta_{ext}(q, x)$를 먼저 수행한 상태 $q' = \delta_{int}(\delta_{ext}(q, x))$가 서로 다른 결과값이 될 수 있기 때문이다. 두 함수 사이의 우선순위 결정은 대상 시스템의 도메인 지식에 의해서 결정되어야 한다. 방정식 (1)과 방정식 (2)는 모두 시스템 이론적 모델 표현으로 미분 방정식 (1)의 AQ와 BX는 각각 방정식 (2)의 $\delta_{int}(q)$와 $\delta_{ext}(q, x)$에 정확히 일치 한다.

그림 5는 DEVS 방정식 (2)로 표현되는 DEVS 형식론을 나타내며 아래와 같은 특징을 지니고 있다[4].

- 시스템 이론적 (입력, 출력, 상태 및 상태 천이) 모델링 틀 제공
- 집합론에 기반하여 함수, 관계(Relation) 등을 이용한 모델 표현
- 계층적(Hierarchical)으로 모듈화(Modular)된 모델 표현
- 객체지향적(Object-oriented)으로 컴포넌트화된 모델 표현

DEVS 형식론은 시스템을 계층적으로 모듈화 하여 표현한다. 이를 위하여 DEVS 형식론은 2개의 모델 클래스 즉, 원자 모델(Atomic Model)과 결합 모델(Coupled Model)을 정의한다. 원자 모델은 더 이상 분해 할 수 없는 컴포넌트로 식 (2)와 같은 상태 방정식으로 표현되며 결합 모델은 부 모델들의 결합체로서 이들 부 모델들은 각각 원자 모델 혹은 결합 모델이다. 따라서 시스템 모델은 결합 모델이 또 다른 결합 모델을 부 모델로 포함한 나무(Tree) 형태의 계층적 구조를 갖게

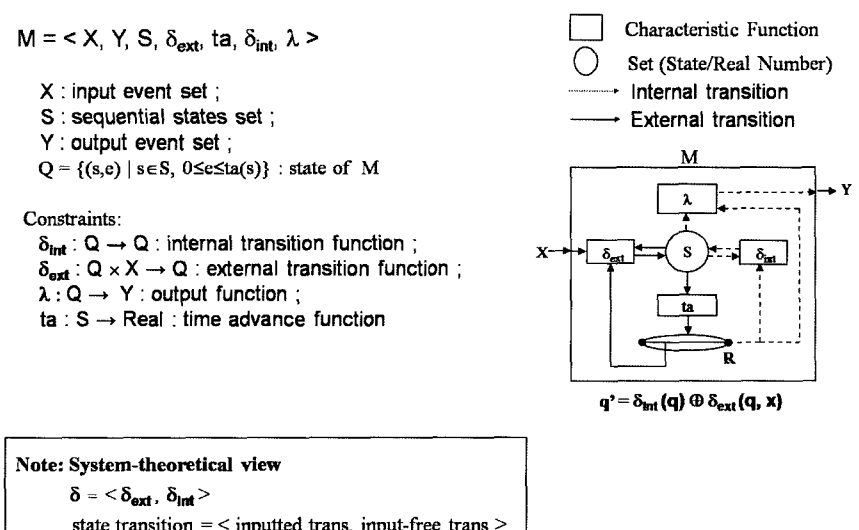


그림 5 원자 DEVS 모델 형식론[1]

된다. DEVS 형식론을 이용한 모델링 이론 및 응용은 [4]를 참조하기 바란다.

DEVS 형식론의 핵심은 이산사건 시스템 모델을 상태 방정식으로 표현할 수 있는 수학적 틀이다. 따라서 DEVS 모델 방정식은 대상 이산사건 시스템의 논리 검증 및 성능 해석에 통합적으로 적용할 수 있다. 시스템 이론적 상태 방정식 형태는 아니지만 이산사건 시스템의 논리 검증 및 성능 해석을 위한 수학적 모델링 기법들이 다수 존재하며 이들을 그림 6에 요약하였다. 그림에서 알 수 있듯이 이산사건 시스템 모델링을 위하여 여러 가지의 수학적 형식론이 고안 되어 사용되고 있다. 따라서 정확히 모델링을 위해서는 이들 형식론의 의미론에 대한 정확한 이해와 아울러 대상 시스템에 대한 도메인 지식이 필수적이다. 중요한 것은 모델링 형식론은 모델링을 위한 틀을 제공하는 것에 지나지 않으며 틀에 도메인 지식을 채워 넣는 과정인 모델링은 사용자의 몫이다. 따라서 모델링 형식론의 선택 기준은 형식론이 대상 이산사건 시스템을 얼마나 직관적이며 간편하게 표현 할 수 있으며 프로그램 언어로 구현하기 용이 한가 이다. 이런 관점에서 DEVS 형식론의 모델링 간편성 및 객체지향적 프로그래밍 구현 용이성이 가장 큰 장점이다. 아울러 DEVS 형식론의 모델 표현 일반성은 수학적 틀을 사용하지 않고 관찰자 관점(World View)에서 모델링 할 수 있는 모든 시뮬레이션 언어들의 의미론을 제공할 수 있다. 참고 문헌 [22]는 사건 중심(Event-oriented), 프로세스 중심(Process-oriented) 등의 관찰자 관점(World View)에 기반한 시뮬레이션 언어들은 모두 DEVS 형식론으로 기술할 수 있음을 보여준다.

3.2 시뮬레이션 방법론

앞서 기술한 대로, 시뮬레이션은 시스템 모델인 상태 방정식을 푸는 과정으로 주어진 상태 방정식을 푸는 방법 즉, 시뮬레이션 방법은 여러 가지가 존재할

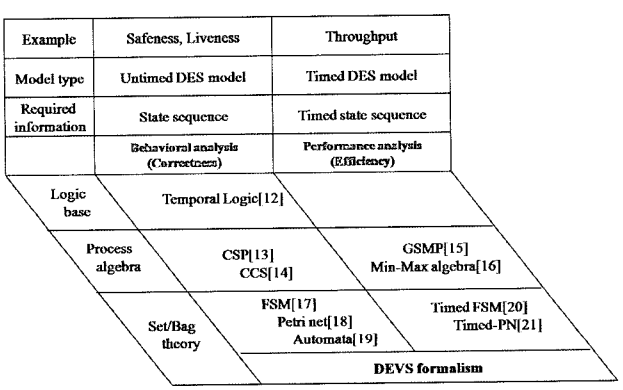


그림 6 이산사건 시스템의 모델링 형식론[1]

수 있다. 시뮬레이션 방법을 구현한 시뮬레이션 프로그램(혹은 시뮬레이터)의 성능은 상태 방정식을 푸는 데 사용되는 메모리 공간(Space)과 소요되는 시간(Time)으로 평가되지만 주로 소요되는 시간(시뮬레이션 속도)에 관심을 가지고 있다. 따라서 주어진 상태 방정식을 최대한 빨리 풀어내는 시뮬레이션 알고리즘 개발이 연구 대상이 된다.

연속 시스템의 상태 방정식을 푸는 수치 해석법은 수학 분야에서 잘 연구되어 있으며 이들 수치 해석법을 프로그램 언어를 사용하여 효율적으로 구현하는 것은 어려운 일이 아니다. 수치 해석 알고리즘의 핵심은 수치 적분을 이용하여 미분 방정식을 푸는 것으로 대부분의 수치 해석 프로그램은 [23] 등의 참고문헌에서 쉽게 접할 수 있으며 이들을 Fortran, C/C++ 혹은 Java 등으로 구현한 공개 소프트웨어들도 풍부하다. 수치 해석 알고리즘 성능의 핵심은 상태 방정식을 푸는 적분 시간 및 적분 오차가 최소화되는 적분 시간 간격을 찾아내는 것이다. 이를 위해서 상태 방정식을 푸는 과정에서 상태 변수의 변화율이 낮은 경우는 적분 시간 간격을 크게 하고 변화율이 큰 경우는 적분 시간 간격을 작게 하는 가변 적분 시간 간격이 사용 된다.

이산사건 시스템의 상태 방정식인 DEVS 방정식을 푸는 시뮬레이션 알고리즘은 미분 방정식의 수치 해석에 비해 훨씬 복잡한 알고리즘으로 표현된다. 이산사건 시스템은 여러 개의 원자 모델로 구성되며 원자 모델 각각은 DEVS 방정식으로 주어지게 되므로 시스템 상태 방정식은 여러 개의 DEVS 상태 방정식들의 연립 방정식 형태로 표현된다. 연립 방정식 해석은 계층적 스케쥴링 알고리즘으로 제안되어 있다. 제안된 알고리즘의 핵심은 모든 원자 모델 중 내부 사건 발생 시간이 최소인 원자 모델을 선택하고 선택된 원자 모델의 내부 상태 천이 함수를 수행시킨다. 내부 상태 천이가 발생한 원자 모델은 출력 사건을 발생시키고 이 출력 사건이 다른 원자 모델의 입력 사건을 발생시켜 그 모델의 외부 상태 천이를 일으키게 된다. 이 과정을 거친 후 다시 모든 원자 모델 중 내부사건 발생 시간이 최소인 원자 모델이 선택되어 내부 상태 천이를 하는 위의 과정을 되풀이 한다. 자세한 알고리즘은 참고문헌 [4]을 참조하기 바란다. 그림 6에 제시한 형식론으로 모델링된 이산사건 모델은 각 모델 고유의 시뮬레이션 알고리즘이 존재하게 된다. 그러나 모든 모델에 적용되는 시뮬레이션 알고리즘의 공통적인 기본 원리는 시스템에서 일어나는 사건 발생 시간 순서에 따라 시스템 정보(상태 변수)를 갱신(상태 천이)하는 것이다.

4. M&S 공학 연구 분야

M&S 공학은 다 학제적 융합 학문이다. M&S 공학의 연구는 M&S 전체 프로세서에서 일어나는 각 과정에 대한 이론 연구와 M&S 기반 문제 해결을 목적으로 하는 응용 연구로 나눌 수 있다. 본 절에서는 이들 각각을 간단히 소개 한다.

4.1 이론 연구 분야

M&S 공학 이론 연구는 시스템 공학, 수학, 전산(공)학 등을 포함한 학문들의 다 학제적 협동교류로 이루어져야 한다. 따라서 이러한 학문에서의 연구 결과를 M&S 프로세서의 각 과정에서 M&S 의미에 맞게 손질하거나 확장하여 사용하게 된다. 반대로, M&S 공학의 연구 결과가 타 학문에서 계승/발전되어 사용되는 경우도 있다. 예를 들면 M&S 공학의 시뮬레이터 구현 연구는 전산(공)학의 프로그래밍 언어, 소프트웨어 공학, 하드웨어, 네트워크 등의 플랫폼 기술과 함께 진화해 왔다. 최근 플랫폼 기술을 사용한 시뮬레이터 개발은 20년 전의 그것과는 개발 환경, 구현 방법, 사용자 인터페이스 및 성능 등이 확연히 달라져 있다.

반대의 예로서, M&S 공학 연구 결과가 전산(공)학 분야 연구에 사용된 경우도 있다. 대표적인 예로서 소프트웨어/하드웨어 시스템의 수학적 모델링인 정형적 명세(Formal Specification) 및 모델의 정형적 검증(Formal Verification) 기법이다. 뿐만 아니라, 객체지향 프로그램 언어로 널리 사용되고 있는 C++ 언어는 처음에는 이산사건 시뮬레이션 언어로 개발되어 현재는 범용 언어로 사용되고 있다. AT&T에서 C++를 최초로 개발한 Stroupstrup은 그의 저서[24] 머리말에서 C++는 이산사건 시뮬레이션용으로 만들어졌고 최초의 객체지향형 시뮬레이션 언어 SIMULA[25]을 C로 흉내낸 것으로 초창기의 이름은 "C with Class"라고 명시하고 있다.

M&S 공학 연구의 핵심은 모델링 형식론 및 시뮬레이션 방법론 고안에 대한 것으로 중요한 연구 주제는 모델링 형식론, 모델 개발 방법론, 정형적 검증법(Formal Verification), 시뮬레이션 방법론 및 환경, 시뮬레이터 연동 등이 포함된다. 본 절에서는 이들 각각의 문제 정의 수준만을 간략히 언급한다.

모델링 형식론 연구는 효율적인 시스템 해석(시뮬레이션)을 위한 모델링 기법 고안과 모델 표현력의 자유도가 높은 모델링 기법 고안에 대한 연구 들이다. 모델 기반 시스템 해석(시뮬레이션)은 주어진 모델의 의미론에 따라 모델의 행위를 추적하는 것이다. 따라서 대상 시스템의 효율적인 해석(시뮬레이션)을 위해서는 모델 표현력에 제약을 가해야 하며 이러한 제약은 대상 시스템이 달라질 경우 모델 표현이 불가능해질 수 있다. 반대로, 표현력에 자유도가 높은 모델 의미론은 모델 해석(시뮬레이션) 과정이 복잡하게 되어 비효율적이다. 예를 들면, CCS 모델은 통신 시스템을 모델링하여 분석하는 데는 효율적이지만 생산시스템을 모델링, 시뮬레이션 및 분석하는 데는 부적합하다. 마찬가지로, DEVS 모델은 모든 이산사건 시스템을 모델링할 수 있는 표현력의 자유도가 있지만 Petri Nets 모델에 비해 시스템의 논리적 특성 분석에는 비효율적이다. 모델링 의미론 연구는 논리학, 대수학, 해석학 등 응용 수학 분야의 이론적 기반을 필요로 한다.

모델 개발 방법론은 대상 시스템과 모델링 형식론이 주어진 경우 모델 개발 절차에 대한 연구이다. 개발 절차(프로세스)는 구현된 모델의 신뢰도와 직접 관련이 있으며 모델의 검증, 실증 및 인증(VV&A: Verification, Validation and Accreditation)[26]을 위한 표준화된 프로세스를 따르는 것이 필요하다. 표준화된 프로세스는 모델링 전체 절차와 각 단계에서 수행할 업무가 무엇인지(즉, 모델 설계, 모델 구현 등)를 정의하고 있지만 이 업무를 어떻게 수행할 지는 채택된 모델링 형식론에 따라 달라지게된다. 예를 들어 컴포넌트 기반 개발(CBD: Component Based Development) 절차를 준수하여 이산사건 모델을 개발하는 경우 모델 설계는 DEVS 형식론으로 모델 구현은 DEVSim++로 할 수 있다[33].

정형적 검증법은 주어진 두 개의 명세서(Specification)가 완전히 일치하는 지를 검증하는 것이다. 예를 들면, 시스템 설계에서 상위 레벨의 개략적인 설계 명세서(HS)와 하위 레벨의 상세 설계 명세서(LS)가 작성된 경우 이들이 완전히 일치하는지를 검증하는 문제이다. 연구의 핵심은 HS와 LS를 각각 수학적 모델(형식론적 모델 혹은 정형적 모델) HS-M과 LS-M으로 명세하고 이들의 행위가 완전히 일치하는지를 확인하는 효율적인 방법을 찾아내는 것이다. 여기서 "완전히"라는 용어는 HS-M의 모든 행위에 대하여 LS-M의 대응하는 행위가 같고 반대로, LS-M의 모든 행위에 대하여 HS-M의 행위가 같다는 것을 증명해야 한다. 이러한 검증은 일반적인 시뮬레이션으로는 불가능 하다. 왜냐하면 일반적으로 시뮬레이션은 모델에 기술된 모든 행위를 수행할 수 없으며 단지 테스트 시나리오에 기술된 부분만을 수행하기 때문이다. HS-M과 LS-M을 위한 모델링 형식론은 검증 대상 분야에 따라 달

라지며 동일한 형식론을 사용하는 경우와 다른 형식론을 사용하는 경우로 나누어진다. 두 개의 정형적 모델 HS-M과 LS-M의 완전 일치를 검증 방법은 Modeling Checking[27], System Morphism[28], Bi-Simulation 기법[29] 등이 있으며 이들 방법을 소프트웨어로 구현하면 자동 검증(Automatic Verification)[30] 시스템이 된다.

시뮬레이션 방법론 및 환경에 대한 연구는 주어진 모델을 효율적으로 시뮬레이션 하기 위한 알고리즘과 개발 환경에 대한 것이다. 앞서 언급한 바, 주어진 모델(방정식)에 대한 시뮬레이션 알고리즘(방정식 푸는 방법)은 다수가 존재하므로 주어진 연구는 이들 중 최적의 알고리즘을 고안하는 문제이다. 한 개의 프로세서에 모델을 구현하여 시뮬레이션을 수행하는 순차적 시뮬레이션 방법 연구와 모델을 여러 개의 프로세서에 분산시켜 구현한 후 시뮬레이션을 수행하는 분산/병렬 시뮬레이션 방법에 대한 연구들이 있다[35]. 분산/병렬 시뮬레이션의 목적은 컴퓨팅 자원 활용 및 시뮬레이션 속도 개선이다. 이 분야 연구는 알고리즘, 운영체제, 프로그램 언어, 네트워크 등과 관련되어 전산(공)학 및 플랫폼 기술과 함께 융합적으로 이루어져야 한다.

시뮬레이터 연동은 분산 시뮬레이션의 한 특수한 형태로서 국방 M&S 분야에서 필요성이 제기되어 연구가 시작된 분야이다. 국방 M&S에서는 기존에 개발된 독립적인 시뮬레이션 프로그램(혹은 시뮬레이터)을 컴포넌트 화하여 연동함으로서 한 개의 컴포넌트 수행만으로는 불가능한 임무를 연동 시뮬레이션을 통하여 수행하고자 한다. 예를 들면, 워 게임 시뮬레이션인 경우 육·해·공군 에서 각각 개발한 워 게임 시뮬레이터를 연동하여 합동 시뮬레이션을 하는 경우이다. 연동 시뮬레이션은 참여하는 컴포넌트 시뮬레이터의 종류와 개발 환경이 다양한 관계로 이들의 정보(데이타 및 시뮬레이션 시각) 교환을 위한 인터페이스를 표준화 할 필요성이 있다. 이러한 인터페이스의 국제 표준이 HLA(High Level Architecture) 이고 HLA 를 구현한 소프트웨어를 RTI(Run Time Infrastructure)라 한다. HLA/RTI는 1999년 미 국방성(DoD) 표준으로 지정되었고 2000년 국제 전기전자학회(IEEE) 표준으로 제정되었다[31]. 이 분야의 연구는 HLA/RTI를 사용하여 다른 종류의 시뮬레이터를 효과적으로 연동하는 방법론과 도구 개발이 주가 되고 있다[6,34].

4.2 응용 분야

응용 연구는 M&S를 문제 해결 도구로 사용하는 모든 학문 분야에서 이루어지고 있다. 과학, 공학, 의학, 생태학, 경영학, 정치학, 군사학 등을 포함한 거의 모든 학문 분야에서 M&S 기술을 활용하여 시스템을 모델링하여 여러 가지 가상 실험(시뮬레이션)을 하게 된다. 아울러, M&S 기술은 시스템/장비 등을 가상적으로 운용하는 훈련 시뮬레이터 분야와 시뮬레이션 기반 획득(SBA: Simulation Based Acquisition) 분야에서 강력한 의사 결정 수단으로 활용된다[32]. 이러한 응용 연구를 위해서는 적용하고자 하는 모델링 의미론의 완전한 이해와 더불어 대상 시스템의 도메인 지식과 시스템 동작 법칙에 대한 정확한 이해가 필수적이다. 응용 연구에서 핵심이 되는 것은 주어진 문제 해결을 위하여 어떤 모델링/시뮬레이션 방법론이 가능한 지를 찾아내고 가능한 방법론이 다수 개가 존재할 경우 가장 효과적인 방법론을 적용하는 것이다.

M&S 분야 연구 사례를 보이기 위해서 2007년 7월 16일-7월 19일 기간 동안 미국 San Diego에서 개최된 국제 시뮬레이션 학회(SCS: Society for Modeling and Simulation International)가 주최한 2007년 하계 시뮬레이션 종합 학술대회(2007 Summer Simulation Multi-Conference)에서 발표된 이론 및 분야별 응용 논문들을 표 1에 요약하였다.

표 1 SCSC'2007 발표 논문 현황

	Track 명	논문 수
이론 분야	Model-Based Specification & Simulation-Based Design and Procurement	42
	DEVS Workshop	10
	Methodology, Tools and Software Applications	13
	Simulation Tools Interfaces	3
응용 분야	Computational Modeling and Simulation of Embedded Systems	14
	Applications in Business Management, Planning & Forecasting	12
	Workshop on the Design, Analysis and Simulation of Distributed systems	12
	Bioinformatics/Biology	11
	Environment, Agriculture and Ecology	11
	Agent-Directed Simulation	12
	At Man's Step	10
	Military Application & Simulation	8
	Education and Body of Knowledge	3
	3D Simulation and Visualization	5
	Emergency Simulation	6
	Inventory Control and Production Planning	3
	Symposium of Performance Evaluation of Computer Telecom Systems	75

5. 결 론

산업 경쟁력은 최소 비용으로, 최고의 제품을 최단 시간 내에 생산하는 것을 요구하며 이를 충족하기 위한 핵심 기술이 M&S 공학이다. M&S 공학은 시스템의 설계/개발/생산 전 과정을 가상적으로 수행하고 검증할 수 있는 유일한 도구로서 사회적 수요가 점차 높아지고 있다. 그러나, 현재 까지는 고등 교육 기관(대학교 학부 이상)에서 M&S 공학을 독립적 전공 분야로 분류하고 체계적인 교과 과정을 개발하여 전공자를 배출하지 못하였다. 그나마도, 각 전공 분야에서 M&S 기술을 그 분야를 연구하는 가장 경제적이고 실용적인 도구로 인식하여 적극적으로 활용하고 있는 것은 다행스러운 점이다.

M&S 공학을 독립적 전공 분야로 분류하여 대학에서 전공 학과를 설치한 경우는 유럽의 일부 대학을 제외하고는 국·내외 적으로 그 사례를 찾아보기 어렵다. 그러나 미국과 유럽의 경우 대학원 과정에서 M&S 전공 학위(석·박사) 과정 프로그램은 오래전부터 시행해 왔다. 또한 미국의 경우 M&S 전공자의 자격을 공인기관(NTSA: National Training and Simulation Association)에서 인증하는 M&S 기술사(Modeling and Simulation Professional) 자격증 제도가 있다. 이러한 배경에서 볼 때, 국내에서도 M&S 공학 전공자의 체계적인 양성을 고려할 시점이라고 생각한다. 이를 위하여 학술단체 등을 주축으로하여 M&S 공학을 학술적으로 정의하고 국가적 차원에서 M&S 전공자의 수요 조사가 선행되어야 할 것이다. 이러한 선행 연구를 바탕으로 고등 교육 기관에서 정규 과정을 통한 전문가 양성의 틀(교과 과정, 운영 방안 등)이 제안되어야 한다. 아울러, 양성된 M&S 전문가를 공인하고 관리하는 기사/기술사 자격증 제도의 도입도 이루어져야 할 것이다.

참고문헌

[1] T.G. Kim, http://sim.kaist.ac.kr/Course/EE612/, Lecture Note on EE612: Discrete Event Systems Modeling and Simulation, Department of EECS, KAIST, 2007.

[2] Richard H. Thayer and Merlin Dorfman, System and Software Requirements Engineering, IEEE Computer Society Press, 1990.

[3] T. G. Kim and C. H. Sung, "Objective-driven DEVS Modeling Using OPI Matrix for Performance Evaluation of Discrete Event Systems," Proc. of 2007 Summer Computer Simulation Conference(SCSC' 2007), pp.305-311, San Diego, California, USA, 2007.

[4] B. P. Zeigler, H. Praehofer and T. G. Kim, Theory of Modeling and Simulation(2nd ed.), Academic Press, 2000.

[5] Edward C. Russell, Building Simulation Models with SIMSCRIPT Ⅱ.5, CACI, 1983.

[6] T. G. Kim and J. H. Kim, "DEVS Framework and Toolkits for Simulators Interoperation Using HLA/RTI," Proc. of Asia Simulation Conference/the 6th International conference on System Simulation and Scientific Computing, pp.16-21, China, October 2005.

[7] K. J. Hong and T. G. Kim, "Timed I/O Test Sequences for Discrete Event Model Verification," LNAI 3397, pp.275-284, 2005.

[8] R. G. Sargent, "Verification and validation of simulation models," Proc. of the 2003 Winter Simulation Conference, vol. 1, pp. 27-48, December 2003.

[9] R. Dumke, C. Rautenstrauch, A. Schmietendorf and A. Scholz(Eds.), Performance Engineering: State of the Art and Current Trends, Springer, 2001.

[10] Jerry Banks and John S. Carson, II, Discrete-Event System Simulation, Prentice Hall, 1984.

[11] Donald F. Stanat and David F. McAllister, Discrete Mathematics in Computer Science, Prentice Hall, 1977.

[12] Zohar Manna and Amir Pnueli, The Temporal Logic of Reactive and Concurrent Systems, Springer-Verlag, 1992.

[13] C. A. R. Hoare, Communicating Sequential Processes, Prentice Hall, 1985.

[14] Robin Milner, Communication and Concurrency, Prentice Hall, 1989.

[15] Peter W. Glynn, "Special issue: Generalized semi-Markov processes," Discrete Event Dynamic Systems: Theory and Application 6, 1996.

[16] Raymond Cuninghame-Green, Minimax Algebra, Springer-Verlag, 1979.

[17] Arthur. Gill, Introduction to the Theory of Finite-State Machines, Mc-Graw Hill, 1962.

[18] James Lyle Peterson, Petri Net Theory and the Modeling of Systems, Prentice Hall, June 1981.

[19] Zvi Kohavi, Switching and Finite Automata Theory 2nd Edition, McGraw-Hill, 1978.

[20] G. Noubir, Daniel R. Stephenes and Prasad Raja, "Specification of timed finite state machine in Z for distributed real-time systems," Proc. of the

IEEE 4th Workshop on Future Trends in Distributed Computing Systems, pp.319-325, Lisbon, Portugal, 1993.

[21] Jiacun Wang, Timed Petri Nets: Theory and Application(The international Series on Discrete Event Dynamic Systems), Springer, October 1998.

[22] B. P. Zeigler, Theory of Modeling and Simulation, John Wiley & Sons, 1976.

[23] William H. Press, Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery, Numerical Recipes 3rd Edition: The Art of Scientific Computing, Cambridge University Press, 2007.

[24] Bjarne Stroustrup, The C++ Programming Language, Addison Wesley, 1986.

[25] O. Dahl and K. Nygaard, "SIMULA - An ALGOL-based Simulation Language," Communications of ACM, vol. 9., no. 9, pp. 671-678, 1966.

[26] Paul K. Davis, Generalizing concepts and methods of verification, validation and accreditation(VV&A) for military simulations, Rand, 1992.

[27] K. G. Larsen and A. Skou, Computer Aided Verification, Springer-Verlag, 1991.

[28] Michael K. Sain, Introduction to Algebraic System Theory, Academic Press, 1981.

[29] Kim G. Larsen and Yi Wang, "Time Abstracted Bisimulation: Implicit Specifications and Decidability," Proc. 9th International Conference on the Mathematical Foundations of Programming Semantics, New Orleans, LA, USA, 1993.

[30] Krzysztof R. Apt and Ernst-Rüdiger Olderog, Verification of Sequential and Concurrent Programs, Springer-Verlag, 1991.

[31] IEEE std. 1516-2000, "IEEE Standard for Modeling and Simulation(M&S) High Level Architecture(HLA)," IEEE Computer Society, September 2000.

[32] R. Frost, "Simulation based acquisition - an ongoing look," Proc. of 1999 INCOSE, 1999.

[33] J.H. Kim, T.G. Kim and J. Jeong, "Embedding DEVS Methodology in CBD Process for Development of War Game Simulators," Poster Session, Proc. of 2007 Summer Sim. Conference, San Diego, USA, 2007.

[34] Y.J. Kim, J.H. Kim and T.G. Kim, "Heterogeneous Simulation Framework Using DEVS BUS, Simulation," vol. 79, no. 1, pp. 3-18, 2003.

[35] Alex C. Chow and B. P. Zeigler, "Parallel DEVS: A parallel, hierarchical, modular modeling formalism," Proc. of 1994 Winter Simulation Conference, Orlando, Florida, USA, 1994.

[36] C. Cassandras, Discrete Event Systems: Modeling and Performance Analysis, IRWIN, 1993.

김 탁 곤

1975 부산대학교 전자공학과(학사)
1980 경북대학교 전자공학과(석사)
1988 Univ. of Arizona, 전기및컴퓨터공학과(박사)
1980~1983 부경대학교, 통신공학과, 전임강사
1987~1989 (미)아리조나환경연구소, 연구엔지니어
1989~1991 Univ. of Kansas, 전기및컴퓨터공학과, 조교수
1991~현재 KAIST 전자전산학과, 교수
한국시뮬레이션 학회 회장 역임
SIMULATION(국제시뮬레이션 학회(SCS) 논문지) 편집위원장 역임
국제 학회: SCS Fellow, IEEE Senior Member, ACM Member
자격증: 모델링 시뮬레이션 기술사(미국)
국방 M&S 자문위원: 국방부, 합참, KIDA, ADD 등
관심분야: 모델링/시뮬레이션 이론, 방법론 및 환경개발, 시뮬레이터 연동
연구실: 시스템 모델링 시뮬레이션 연구실(SMS Lab) (http://sim.kaist.ac.kr)
E-mail: tkim@ee.kaist.ac.kr