한국시뮬레이션학회 논문지
Vol. 26, No. 2, pp. 9-17 (2017. 6)
http://doi.org/10.9709/JKSS.2017.26.2.009
ISSN 1225-5904

이산사건 시뮬레이션 시스템을 위한 웹 기반 분산 실험 틀

정인호 · 최재웅 · 최창범 †

Web-based Distributed Experimental Frame for Discrete Event Simulation System

Inho Jung · Jaewoong Choi · Changbeom Choi†

ABSTRACT

The problem of social phenomenon is getting more complicated than past decades, and the simulation engineers need more computation power to solve the problem. Therefore, the needs of the computational resources of the modeling and simulation environment are increasing. In the perspective of the simulation, it is necessary to allocate computational resources flexibly so that the simulation can be performed per the available budget range. As an alternative to the simulation environment to accommodate these requirements, cloud service has emerged as an environment in which computing resources can be used flexibly. This paper proposes a web-based simulation framework which consists of a front-end that reconstructs the simulation model using the web, and a back-end that executes the discrete event simulation. This paper also carried out a case study which shows web-based simulation framework has better overall runtime than standalone simulation framework.

Key words : SES/MB(System Entity Structure / Model Base), DEVS(Discrete Event System Specification) Formalism, Operating-system-level Virtualization, Single-page Application, Cloud Computing, Web Services

요 약

현대사회의 다양한 분야에서는 복잡한 사회현상에 대하여 문제를 정의하고 문제해결책의 결과를 분석하는 과정으로 모델링 및 시뮬레이션을 활용하고 있다. 나날이 사회현상의 문제가 복잡해짐에 따라 요구되는 모델링 및 시뮬레이션 환경의 계산 자원의 요구사항도 높아지고 있다. 이러한 요구사항을 수용하기 위해서 클라우드 서비스 등과 같이 컴퓨팅 자원을 유동적으로 사용할 수 있는 환경이 대두되었다. 본 연구에서는 이러한 컴퓨팅 자원 활용 시스템을 보다 효율적으로 활용할 수 있는 웹 기반 재구성이 가능한 시뮬레이션 실험 틀을 제안한다. 제안하는 시뮬레이션 실험 틀은 다양한 분산 컴퓨팅 환경을 지원할 수 있도록 프론트엔드(Front-end)에서 웹을 활용하여 시뮬레이션 모델의 재구성 시스템을 구축하고 백엔드(Back-end) 이산사건 시뮬레이션의 실행을 담당하는 이산사건 시뮬레이션 실행단으로 구성된다. 본 연구는 사례연구를 통해 분산형 시뮬레이션 환경이 단일 시뮬레이션 환경보다 시간적 효율이 더 높음을 확인하였다.

주요어 : SES/MB 형식론, 이산사건 시스템 형식론, 운영체제 레벨 가상화, 단일 페이지 응용프로그램, 클라우드 컴퓨팅, 웹 서비스

1. 서론

현대사회는 다양한 분야에서 복잡한 사회현상에 대하여 문제를 정의하고 문제해결책의 결과를 분석하는 과정으로 모델링 및 시뮬레이션을 활용하고 있다. 새로운 제품을 개발하고, 서비스를 도출하기 위해서는 실물 개발과

Received: 17 February 2017, **Revised**: 24 April 2017, **Accepted**: 24 April 2017

† **Corresponding Author**: Changbeom Choi
E-mail: cbchoi@handong.edu
Dept. of Global Entrepreneurship and ICT, Handong Global University, Pohang, Gyeongsangbuk-do, Korea

정에 진입하기 전에 해당 제품과 서비스에 대한 수요를 예측하고, 적격성 여부 검증을 수행하여 실제 제품 개발 비용을 감축하는 과정이 필수적이며 이 과정들은 모델링 및 시뮬레이션을 통해서 달성할 수 있다.

컴퓨터를 통한 모델링 및 시뮬레이션 과정이 도입되기 이전의 근현대 사회에서는 제품을 실제로 생산하기에 앞서 제품의 형상과 기능을 확인하기 위해 실제의 제품과 동일한 형상으로 소량의 시제품을 제작하고 각종 검증과정을 시행했다. 하지만 컴퓨터를 통한 모델링 및 시뮬레이션 분야의 대두 이후, 소프트웨어를 사용해 문제에 대한 가상의 시뮬레이션 환경을 구축하고, 해당 문제를 모델링하여 각종 검증 및 테스트를 수행할 수 있게 되었다(Jain, 1990). 특히, 로켓공학 분야, 군사무기 분야 등 실제 제품의 개발비용이 막대한 산업분야에서 개발과정의 시행착오를 최소한으로 줄이기 위해 컴퓨터를 통한 모델링 및 시뮬레이션을 적극적으로 활용하고 있으며 이에 따라 모델링 및 시뮬레이션 학문도 빠르게 발전 및 분야를 확장하고 있다.

기존의 모델링 및 시뮬레이션은 하나의 거대한 서버 클러스터 환경에서 실행되어 왔다. 실제제품을 사용해 검증한다면 수개월, 간혹 수년이 걸리는 검증 및 테스트 과정을 수행해야 하므로 서버 클러스터의 컴퓨팅 능력이 곧 시뮬레이션 과정에 필요한 시간을 결정하게 된다. 하지만 해당 환경이 감당할 수 있는 한계치 이상의 연산자원이 필요한 시뮬레이션이 필요한 경우 추가적으로 하드웨어 자원을 구입해 새롭게 증설해야 하는 제약이 존재한다(Voorsluys et al., 2011).

또한, 기존의 모델링 및 시뮬레이션은 시뮬레이션 모델을 구성하고 배포하는 사용자단 시스템과, 실제 시뮬레이션이 실행되는 서버 환경이 로컬 네트워크로 연결되어 있었으며, 해당 연결은 통상적으로 외부에서 접근이 불가능한 폐쇄적인 구조였다. 만약 외부에서 해당 환경에 접속할 필요성이 생기거나, 사용자단 소프트웨어가 설치되지 않은 시스템에서 해당 환경에 접속해야 할 경우 많은 시간적, 금전적 비용이 발생한다는 단점 또한 존재한다.

이러한 문제를 해결하고 시뮬레이션 환경을 유연하게 구성하기 위해 클라우드 컴퓨팅(Cloud Computing) 환경을 활용할 수 있다. 클라우드 컴퓨팅이란 인터넷 기반의 컴퓨팅 기술을 의미한다. 인터넷 상의 데이터 서버에 응용프로그램을 두고 필요할 때마다 사용자 컴퓨터에 불러와서 사용하는 웹에 기반을 둔 환경이다. 이 용어는 사용자가 웹에서 필요한 작업을 다 할 수 있지만 작업에 사용되는 연산자원의 실제위치와 정보는 명확하게 알 수 없는, 마치 구름처럼 퍼져있다는 점에서 클라우드 컴퓨팅이라고 명명되었다 (Yang and Liu, 2012).

본 연구에서는 클라우드 컴퓨팅 환경의 이점을 살려서 유연하게 시뮬레이션 모델을 구성하고 시뮬레이션을 실행하는 웹 기반 실험 틀(Experimental Frame)에 대해서 소개한다. 또한 사례연구로써, Conway의 라이프 게임(Game of Life) 세포 자동자 중 특정 모델을 시뮬레이션 모델로 사용해 분산형 시뮬레이션 환경에서 시뮬레이션을 반복 실행하여 집계된 수치를 분석하였고, 이를 통해 분산형 시뮬레이션 환경이 기존의 단일 시뮬레이션 환경보다 2~14배까지의 성능 향상이 있음을 분석하였다.

2. 배경지식 및 관련연구

본 장에서는 시뮬레이션 모델을 구성하여 시뮬레이션을 수행하기 위한 배경지식을 소개하고 본 연구와 관련된 연구를 본 연구와 비교분석한다.

2.1 배경지식

2.1.1 System Entity Structure / Model Base

System Entity Structure(SES) 형식론은 하나의 복합체계를 구조적으로 다수의 부체계(Subsystem)로 조합으로 표현하고, 각 부체계가 가질 수 있는 대안을 표현함으로써 다양한 복합체계의 대안을 모의할 수 있는 형식론이다(Kim et al., 1990). 따라서 다양한 분석을 위하여 체계를 기술하고, 체계 내의 구조변경을 통하여 기존에 개발된 모델들을 재사용하기에 용이하다는 장점이 있어서 대안분석에 활용하기에 용이하다(Lee, 1994).

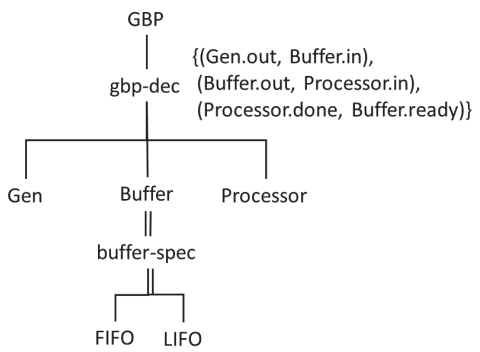

Fig. 1. A simple figure of SES

이렇게 기술된 SES 형식론을 바탕으로 시뮬레이터를

합성하기 위해서는 시뮬레이션 모델을 저장하고 관리할 수 있는 데이터베이스로서 모델 베이스(Model base) 개념이 필요하다. 다양한 대안분석을 위해서 시뮬레이션 모델의 정보와 요구사항을 담고 있으며 SES 형식론으로 구성된 체계구성에 따라 저장되어 있는 시뮬레이션 모델을 가져오고, 이를 합성하는 개념이다.

이와 같이 SES/MB는 시뮬레이션을 수행하기 위하여 시뮬레이션 모델을 선택하고 시뮬레이션을 합성하는 과정과, 시뮬레이션 모델을 저장소에 저장하고 관리하면서 시뮬레이션의 요구사항에 따라 시뮬레이션을 수행해야 하는 상황에서 시뮬레이션 모델을 합성하고 시뮬레이션을 수행한다는 점에서 웹 기반 서비스와 유사하다.

웹 기반 서비스는 사용자와 접점이 되는 웹 서버와 데이터를 관리하는 데이터베이스가 존재하여 웹 서버에서 발생하는 사용자의 입력에 따라 데이터베이스에서 데이터를 받아 이를 처리한다. 따라서 본 연구에서는 시뮬레이션을 수행하기 위하여 시뮬레이션에 대한 요구사항 및 입력 데이터를 웹 기반으로 처리하고 시뮬레이션 모델을 웹 서비스의 데이터베이스를 활용한 관리체계를 구축함으로써 웹 기반의 실험 틀을 구성한다.

2.1.2 이산사건 시스템 형식론

이산사건 시스템 명세(Discrete Event System Specification, DEVS)란, 불규칙적인 시간간격으로 일어나는 시스템 내부 혹은 외부에서의 이산사건 발생시점에서만 상태를 변경하는 시스템의 명세이다(Zeigler et al., 2000).

이 시스템의 사건(Event)은 외부사건과 내부사건 두 가지로 정의된다. 즉, 이산사건 시스템에서의 이산사건 발생시점은 외부에서 들어오는 입력사건과, 입력이 없는 경우 내부적 조건이 만족될 경우 발생하는 내부사건으로 나누어 생각할 수 있다. 이와 같은 이산사건 시스템의 상태방정식은 연속시스템의 상태방정식인 미분방정식처럼 닫힌 형태(Closed form)로 표현할 수 없고, 이산수학(Discrete mathematics)에 기초한 집합이론을 이용하여 천이(Transition) 규칙을 기술한다. 이산집합론에 근거한 DEVS 형식론은 이산사건 시스템을 모듈 별로 나누고 이를 계층적인 연결로 모델링 할 수 있는 수학적인 기반을 제공한다.

이러한 DEVS 형식론을 기반으로 구현된 DEVS 모델은 시뮬레이션을 수행하기 위하여 다른 시뮬레이션 모델에서 메시지를 전달 받으면 이들 메시지를 통하여 이벤트와 상태천이를 수행한다.

본 연구에서는 시뮬레이션 모델을 모델링할 때 DEVS 형식론에 근거하여 모델링하고 이를 SES/MB 형식론을 따르는 구조로 합성(Synthesis)하여 시뮬레이션을 진행한다.

2.1.3 컨테이너 가상화

가상화기술은 클라우드 컴퓨팅에 있어서 핵심 기술이며 일반적으로 가상화기술은 컨테이너 기반의 시스템 가상화기술을 가리킨다(Hogg, 2014).

컨테이너 가상화기술은 응용프로그램을 구동할 수 있는 환경을 가상화하는 기술이다. 컨테이너는 시스템에서 실행되는 응용프로그램 구동 환경을 가상화한다. 따라서 해당 응용프로그램이 동작할 수 있도록 CPU와 메모리 영역만 가상화되며 운영체제와 라이브러리는 시스템 전반에서 공유된다.

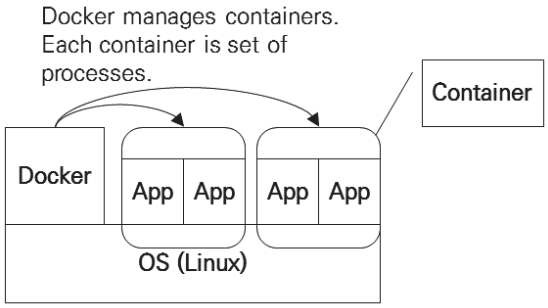

Fig. 2. Architecture of docker

도커(Docker)는 컨테이너 응용프로그램의 배포를 자동화하는 오픈소스 프로젝트이다. Figure 2와 같이 도커는 응용프로그램이 한 컴퓨팅 환경에서 다른 컴퓨팅 환경으로 이동할 때 안정적으로 운영될 수 있도록 해당 응용프로그램과 이를 구동하는데 필요한 모든 의존성, 라이브러리, 기타 바이너리와 구성파일 등을 패키지로 묶어 하나의 런타임 환경으로 제공한다(Merkel, 2014).

본 연구에서는 도커를 중계자로써 사용자 시스템에서 클라우드 컴퓨팅 환경으로 시뮬레이션 모델을 배포하는 관리자로 활용한다.

2.1.4 단일 시뮬레이션 환경과 분산형 시뮬레이션 환경

단일 시뮬레이션(Standalone Simulation Environment, SSE)은 시뮬레이션 연산을 수행하는 프로세스가 하나의 시스템에 집중되어져 있는 환경이다. 이러한 환경에서 시뮬레이션 모델들은 시스템 저장장치에서 로드되어 강력한 성능을 가진 단일 CPU에서 연산된다. 하지만 시뮬레

이션은 일반적으로 시스템의 모든 부분에서 변수를 광범위하게 공유하므로 순차적으로 연산된다. 따라서 SSE에서 대량의 시뮬레이션 연산을 수행할 때, CPU의 연산 능력이 뛰어나다 하더라도 동시에 연산을 수행할 수 있는 수치의 한계점이 존재한다(Chandy, 1979).

분산형 시뮬레이션 환경(Distributed Simulation Environment, DSE)은 단일 시스템 단일 CPU 연산 능력의 한계점을 개선하고자 고안된 환경이다. 수행하고자하는 시뮬레이션에 요구되는 최소한의 사양만을 가진 가상머신(Virtual Machine, VM) 대량으로 생성해 시뮬레이션을 병렬적으로 빠르게 처리할 수 있는 구조를 가진다. 시뮬레이션에 사용되는 VM은 각각 외부 시스템과 격리되어 있어 외부 시스템의 방해를 받지 않고 시뮬레이션을 수행할 수 있고, 컨테이너 가상화기술을 통해 연산처리속도를 네이티브 시스템 대비 96%까지 유지할 수 있기 때문에 더욱 효율적으로 병렬연산을 수행할 수 있다는 장점이 있다(Felter et al., 2015).

2.2 관련연구

본 절에서는 연구주제와 관련된 기존의 클라우드 기반 시뮬레이션 연구를 비교분석한다.

2.2.1 IoTSim

Zeng et al.(2016)은 IoT 응용프로그램에 대한 시뮬레이션을 실행할 수 있는 맵리듀스(MapReduce) 기반 빅데이터 처리 시스템과 이 시스템을 대한 다양한 토폴로지 구성을 적용할 수 있는 시뮬레이트된 클라우드 환경을 제안하였다.

Calheiros et al.(2011)의 CloudSim을 기반으로 클라우드 환경을 시뮬레이션하고 그 위에 IoT 시뮬레이터를 배포하여 IoT 응용프로그램의 효용성을 검증함으로써 일종의 IoT 환경 에뮬레이터의 역할을 기대할 수 있다.

또한 Zeng의 연구에서 다양한 IoT 토폴로지를 수용해서 시뮬레이션 모델을 변경 및 조절할 수 있다는 점은 SES/MB를 사용해 시뮬레이션 모델을 조합하고 합성할 수 있는 본 연구와 유사하다. 하지만 클라우드 환경 자체를 시뮬레이션하고 그 환경 위에 IoT 시뮬레이션을 수행하는 점에서 시뮬레이션 퍼포먼스에 대해서는 고려치 않은 점이 본 연구와 다른 점이다.

2.2.2 Cloud-based Simulation

Liu et al. (2012)은 병렬분산 시뮬레이션(Parallel and Distributed Simulation, PADS)의 성능에 영향을 주지 않으면서 클라우드 기반 실행이 가능하도록 지원하는 클라우드 기반 시뮬레이션(Cloud-based Simulation, CSim) 환경을 제안하였다. 기본적으로 CSim은 상용화된 클라우드 환경에서 여러 개의 프로세스로 구성된 분산 시뮬레이션 모델이 실행될 수 있도록 다수의 시뮬레이션 모델을 클라우드 컴퓨팅 환경에 배포하여 분산 시뮬레이션이 수행될 수 있도록 하며, 시뮬레이션 흐름에 따라 시뮬레이션 모델이 사용하는 연산자원의 양이 달라질 수 있기 때문에 프로세스의 우선순위를 조정할 수 있는 스케줄링 알고리즘들을 제안하였다.

이와 같은 CSim은 상용 클라우드 환경에 하나의 시뮬레이션 목적에 부합하는 다수의 분산 시뮬레이션 모델을 배포하고 시뮬레이션의 진행을 제어할 수 있도록 웹 인터페이스를 사용한다는 측면에서 본 연구와 유사하지만 이미 구현이 완료된 시뮬레이션 소프트웨어들을 조합하여 분산 시뮬레이션을 수행한다는 측면에서 SES로 명세된 시뮬레이션 틀을 활용하여 단일 시뮬레이션 소프트웨어를 합성하고 이를 클라우드 환경에 배포하는 웹 기반의 유연한 시뮬레이션 환경인 본 연구와 차이점이 있다.

3. 웹 기반 유연한 실험 틀

기존 시뮬레이션 환경은 폐쇄적인 인트라넷 네트워크 환경에서 제한된 하드웨어 자원(CPU, 메모리, 저장장치 등 연산 자원)을 활용해 시뮬레이션하는 환경이었다. 이러한 환경 아래 더 효율적이고 빠른 시뮬레이션을 위해서는 많은 비용을 추가 지불하고 하드웨어 자원을 구매해야 하는 제약이 있었다. 만약 시뮬레이션 환경에 웹 응용프로그램 유저 인터페이스(User interface, UI)를 접목하고 시뮬레이션에 필요한 하드웨어 자원을 필요한 만큼만 사용할 수 있다면 웹의 특성으로 인해 시뮬레이션 환경의 접근성이 매우 높아질 것이고, 클라우드 컴퓨팅의 특성으로 인해 하드웨어 자원에 지출되는 비용을 효율적으로 절감할 수 있을 것이다. 본문에서는 웹과 클라우드 컴퓨팅의 장점을 크게 살려 기존 시뮬레이션 환경이 가지는 제약을 없애고, 유연하면서도 비용절감효과가 높은 웹 기반의 유연한 실험 틀을 제안한다. 또한, 이 실험 틀은 클라우드 컴퓨팅 환경 사용이 불가한 경우 기존 구축된 하드웨어 자원으로도 유연하게 시뮬레이션 실행 및 결과 보고를 받을 수 있는 환경도 포함한다.

3.1 웹 서비스의 구성 요소

이 실험 틀은 Figure 3과 같은 구성을 가진다. 사용자

는 기존 설치형 시뮬레이터에서 사용하던 시뮬레이션 구성과 설정을 웹 기반 UI를 통해서 언제 어디서나 접근 가능하고 사용 가능하며, 사용자가 구성한 시뮬레이터는 시뮬레이션 관리자가 모델 저장소(Repository)로부터 모델 바이너리를 전송 받아 시뮬레이터 바이너리로 합성(Synthesis)한다. 합성된 시뮬레이터는 도커 컨테이너 가상화기술을 통해 클라우드 기반의 시뮬레이션 컨테이너 또는 기존의 로컬 기반의 프라이빗 시뮬레이션 서버로 전송된다.

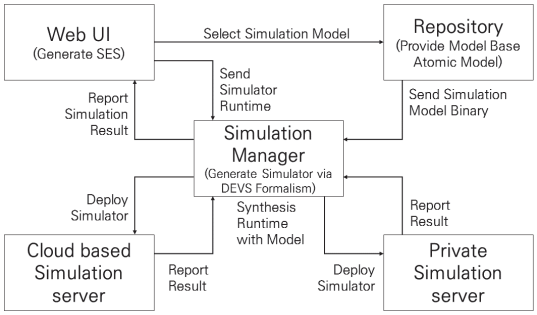

Fig. 3. Architecture of web-based experimental frame

사용자가 웹 기반 UI에서 구성한 시뮬레이션의 구성은 SES 형식론을 따르는 스크립트로 표현되며 해당 스크립트의 예제와 이를 도식화한 그림은 각각 Figure 4, Figure 5를 참고한다.

```
[Model Synthesizer Version 0.01]
[BEGIN SCRIPT]
        [BEGIN SYSTEM ROOT]
                Root : ABC
        [END SYSTEM ROOT]
        [BEGIN STRUCTURE]
                ABC consist_of AB, C
                AB consist_of A, B
        [END STRUCTURE]
        [BEGIN COUPLING]
                ABC:(ABC, in, AB, in)
                AB: (AB, in, A, in)
                AB: (A, out, B, in)
                AB: (B, out, AB, out)
                ABC: (AB, out, C, in)
                ABC:(C, out, ABC, out)
        [END COUPLING]
        [BEGIN ATTRIBUTE]
        [END ATTRIBUTE]
[END SCRIPT]
```

Fig. 4. SES scripts created by the experimental frame

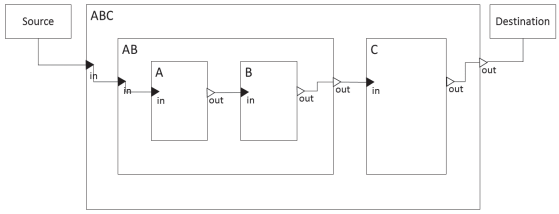

Fig. 5. SES script example

시뮬레이션 관리자는 클라우드 컴퓨팅 서비스 또는 사전 구성된 로컬 시뮬레이션 서버에 연결되어 있으며 하드웨어 자원을 관리하는 역할을 한다. 이 시뮬레이터는 클라우드 컴퓨팅 환경 또는 사전 구성된 로컬 시뮬레이션 서버 환경에 전송되어 시뮬레이션 과정을 수행한 뒤 결과를 다시 시뮬레이션 관리자로 전송한다. 시뮬레이션 관리자는 결과를 다시 웹 기반 UI로 전송해 사용자에게 표시한다.

3.2 실험 틀의 서버 아키텍처

Figure 6에서 이 실험 틀의 서버 아키텍처를 설명한다. 웹 기반 UI는 SPA 기술을 사용하여 빠르고 간결하게 구현이 가능하다. 시뮬레이션 관리자는 모델 저장소에 질의(Query)하여 사용 가능한 모델의 정보를 획득하고, 이를 웹 기반 UI에 전송해 사용자에게 표시한다. 이 때 통신 과정은 시뮬레이션 매니저 서버를 통해 Representational state transfer(REST) API로 이루어진다(Richardson, 2008).

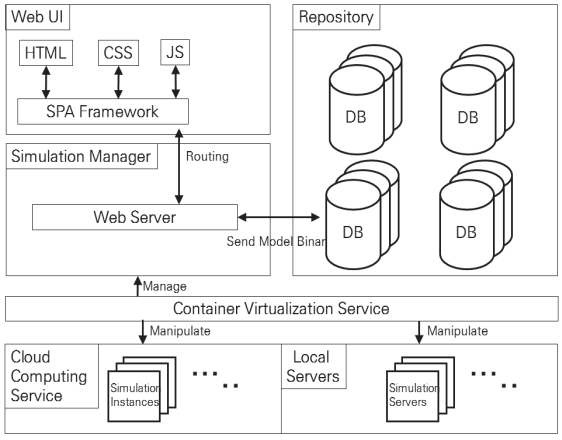

Fig. 6. Server architecture of the experimental frame

Figure 7과 같이 사용자가 모델과 시뮬레이션 설정을 선택하면 웹 기반 UI는 이를 시뮬레이션 관리자로 보내

고 관리자는 수신한 모델 목록을 다시 저장소에 질의해 해당 모델의 바이너리를 전송 받는다. 시뮬레이션 관리자는 DEVS 형식론에 의거해 모델 바이너리와 시뮬레이션 설정을 합성해 시뮬레이터를 생성한다.

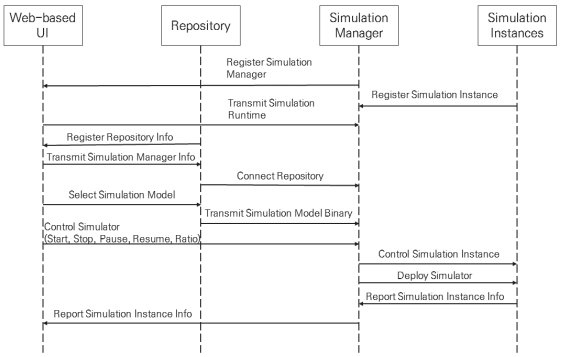

Fig. 7. Flowchart of the experimental frame

생성된 시뮬레이터를 도커 등의 컨테이너 가상화 서비스를 사용해 클라우드 컴퓨팅 서비스 또는 로컬 시뮬레이션 서버로 전송해 시뮬레이션을 실행하고 결과를 다시 반환 받는다. 관리자는 실행결과를 웹 기반 UI에 전송하고 사용자는 실행결과를 확인한다. 이와 같이 사용자는 별도의 프로그램 설치나 구성없이도 웹페이지에서 모든 작업을 완료할 수 있다.

이 실험 틀을 통해 사용자는 웹 기반 UI에서 사용자가 필요로 하는 시뮬레이션 모델을 조합하여 이를 클라우드 컴퓨팅 서비스에 바로 배포할 수 있다. 통상적으로 하나의 시뮬레이션 모델에 대해 다양한 상태 및 인자(Parameter)를 부여하면서 여러 번 반복 시뮬레이션하는 모델링 및 시뮬레이션의 특성 상, 간단한 시뮬레이션이라도 상당한 연산 자원을 필요로 한다. 만약 사용자가 소유한 하드웨어 자원이 필요한 자원보다 적다면, 시뮬레이션을 완료하기까지 걸리는 시간비용은 배수적으로 증가할 것이다. 이 실험 틀을 사용하면 클라우드 컴퓨팅 서비스의 연산 자원을 필요한 만큼 사용할 수 있기 때문에 하나의 시뮬레이션 모델에 대해서 수십, 수백 번의 시뮬레이션을 동시에 진행할 수 있다. 클라우드 컴퓨팅 서비스는 사용한 만큼만 비용을 지불하는 비용정책을 가지고 있으므로 사용자는 시뮬레이션을 위해 별도로 비용을 지불하고 하드웨어 자원을 구입할 필요가 없다.

다음 장의 사례연구에서는 단일 시뮬레이션 모델을 실험 틀을 통해 클라우드 컴퓨팅 서비스에 배포하여 동시에 대량으로 반복 실행하였을 경우 기존 단일 시뮬레이션 서버 대비 시간적 효율이 얼마나 증가하는지에 대한 연구 결과를 기술한다.

3.3 사례연구

본 장에서는 이산 사건 시스템 형식론을 적용한 분산형 시뮬레이션 환경을 구축하는 과정에서, 분산형 시뮬레이션 환경의 성능척도를 가늠하기 위해 시스템 성능을 분석하였다. Conway의 라이프 게임(Game of Life) 세포 자동자 중 특정 실행시간을 가지는 시뮬레이션 모델을 상정하여, 이를 분산형 시뮬레이션 환경과 기존의 단일 시뮬레이션 환경에서 시뮬레이션을 반복 실행하여 집계된 수치로 결과를 도출하였다(Gardener, 1970).

라이프 게임 시뮬레이션 모델은 주어진 임의의 seed 값을 기반으로 무작위로 진행되는 세포 자동자 시뮬레이션 모델이다. 주어진 값에 따라 바로 시뮬레이션이 종료될 수도, 영원히 진행될 수도 있기 때문에 다음 Table 1과 같이 최소 실행시간과 최대 실행시간을 가지도록 제약조건을 설정했다.

Table 1. Characteristics of the simulation model

Min Execution Time	100 ms
Max Execution Time	7000 ms

해당 시뮬레이션 모델을 실험 틀에 배포하여 대량 반복 시뮬레이션 했을 때의 총소요시간과 기존의 단일 시뮬레이션 환경에서 시뮬레이션 했을 때의 총소요시간을 비교하여 백분율로 총 얼마만큼의 시간이 단축되었는지 확인하는 것이 사례연구의 목표이다.

해당 모델의 시뮬레이터 개수를 각 1, 2, 3, 4, 5, 6, 8, 10, 20, 30, 40, 60, 120으로 지정하고 총 반복횟수는 120번으로 지정하여 분산형 시스템에 배포하였다. 도커를 사용하여 시뮬레이터 개수와 총 반복횟수, 그리고 시뮬레이터 바이너리를 명령줄 인자로 지정해 도커가 시뮬레이터 개수만큼 가상머신 인스턴스를 생성해 바이너리를 바로 실행하는 구조로 구성하였다. 총 반복횟수는 가상으로 상정한 이 시뮬레이션을 통해 유의미한 결론을 얻기 위해 실행해야 하는 반복횟수이다. 따라서 총 반복횟수에 시뮬레이터 개수를 나눈 값이 각 시뮬레이터의 실행횟수가 된다.

각 시뮬레이션 세션을 실행한 결과는 Table 2와 같다. 각 인스턴스의 시뮬레이션 실행횟수는 총 실행횟수 / 인스턴스 개수이다. Max Time은 해당 세션에서 단일 시뮬레이션의 최대실행시간이며 Avg Time은 해당 세션의 모

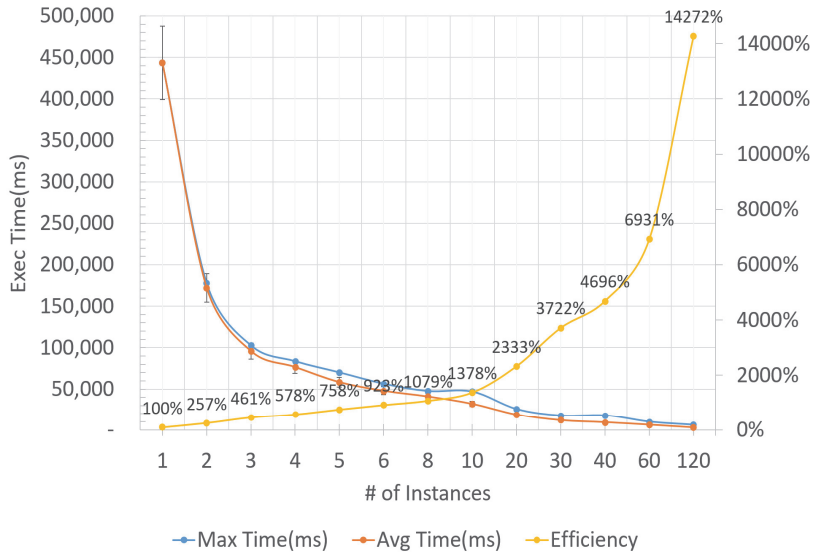

Fig. 8. Visualization of the simulation execution time, maximum execution time and efficiency versus the number of instances

든 시뮬레이션 실행시간의 평균값이다. 인스턴스 개수가 1일 때는 단일 시뮬레이션 환경과 같은 결과이므로 이를 기준치로 각 세션에서 단축된 시간을 Efficiency 열에 백분율로 표시하였다.

Table 2. Simulation result

No.	# of Instances	Individual Iteration	Max Time (ms)	Avg Time (ms)	Efficiency
1	1	120	443,292	443,292	100%
2	2	60	177,746	172,276	257%
3	3	40	102,797	96,065	461%
4	4	30	83,752	76,731	578%
5	5	24	70,246	58,492	758%
6	6	20	56,543	48,036	923%
7	8	15	47,717	41,076	1,079%
8	10	12	47,050	32,166	1,378%
9	20	6	25,832	19,000	2,333%
10	30	4	17,877	11,910	3,722%
11	40	3	17,974	9,439	4,696%
12	60	2	10,046	6,396	6,931%
13	120	1	6,268	3,106	14,272%

Table 2를 바탕으로 인스턴스 수 대비 시뮬레이션 평균 실행시간(Avg time), 최대 실행시간(Max time)과 Efficiency를 시각화한 그래프는 Figure 8과 같다(오차율 ±10%).

동시에 실행하는 인스턴스의 개수가 많아질수록 전체 실행시간이 감소하는 추세가 나타났다. 인스턴스가 10개일 때까지는 전체 실행시간이 큰 폭으로 감소하는 것을 확인할 수 있다. 하지만 10개 이후부터는 인스턴스 개수는 큰 폭으로 증가했지만 이에 따른 실행 시간은 큰 감소를 보이지 않은 것을 확인할 수 있다. 따라서 인스턴스 개수 증가에 따른 실행 시간 감소폭은 선형적이 아닌 지수적 특성을 가지고 있음을 알 수 있다. 차후 각 시뮬레이션 모델의 특성에 따른 최적의 인스턴스 개수를 계산하는 알고리즘 정립이 필요하다.

4. 결론

웹 기반의 실험 틀은 클라우드 컴퓨팅 환경에서 사용자가 언제 어디서든 시뮬레이션 모델을 구성하고 시뮬레이션을 실행하여 그 결과를 받아 볼 수 있다. 특히 클라우드 컴퓨팅의 장점을 최대한 활용한 기법은 다양한 상황과 경우에 따라 무한한 확장성을 가진다.

사례연구를 통해 단위시간당 시뮬레이션 인스턴스 수가 증가할수록 전체 실행시간이 크게 감소하는 결과를 획득했다. 각 시뮬레이션 모델의 특성을 고려해 최적의 시뮬레이션 인스턴스 수를 검출하는 알고리즘과, 이에 따른 비용 및 시간 절감효과를 도출하는 알고리즘이 필요하며, 향후 도커를 활용하여 다중 시뮬레이션 분석환경을 구성하고 이 결과를 분석할 수 있는 환경에 대한 연구가 필요하다.

References

Calheiros, R.N. et al. (2011) “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms”, *Software: Practice and experience*, 41(1), 23-50.

Chandy, K.M., J. Misra, (1979) “Distributed simulation: A case study in design and verification of distributed programs”, *IEEE Transactions on software engineering*, 440-452.

Felter, W. et al., (2015) “An updated performance comparison of virtual machines and linux containers”, In *Performance Analysis of Systems and Software (ISPASS)*, 2015 IEEE International Symposium, 171-172.

Gardener, M (1970) “Mathematical Games – The fantastic combinations of John Conway's new solitaire game ‘life’ ”, *Scientific American*, 223, 120-123.

Hogg, S. (2014) *Software Containers: Used More Frequently than Most Realize*, Network World, Inc.

Jain, R. (1990) *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*, John Wiley & Sons.

Kim, T.G. et al. (1990) “System entity structuring and model base management”, *IEEE Transactions on Systems*, Man and Cybernetics, 27(3), 1013-1024.

Lee, W.B. (1994) “Development of the multifaceted system modelling / simulation environment”, Master Thesis, Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea.

Liu, X. et al. (2012) “Cloud-based simulation: The state-of-the-art computer simulation paradigm”, In *Proceedings of the 2012 ACM/IEEE/SCS 26th Workshop on Principles of Advanced and Distributed Simulation*, 71-74.

Merkel, D. (2014) “Docker: lightweight linux containers for consistent development and deployment” *Linux Journal*, 239(2).

Richardson, L., S. Ruby (2008) *RESTful Web Services*, O'Reilly Media, Inc.

Voorsluys, W., J. Broberg, and R. Buyya (2011) *Introduction to cloud computing. Cloud computing: Principles and paradigms*, Wiley, 1-41.

Yang, H., X. Liu (2012) “Software Reuse in the Emerging Cloud Computing Era”, Hershey, *PA: Information Science Reference*, 204-227.

Zeigler, B.P., H. Praehofer and T.G. Kim (2000) *Theory of Modeling and Simulation*, Academic Press.

Zeng, X. et al. (2016) “IOTSim: a Cloud based Simulator for Analysing IoT Applications”, *arXiv preprint arXiv:1602.06488*.

정 인 호 (wbqdinno@gmail.com)

2016 한동대학교 컴퓨터공학 학사
2016~ 현재 한동대학교 정보통신공학과 석사과정

관심분야 : 웹 기반 시뮬레이션, 시뮬레이션 환경, 분산 시뮬레이션 환경

최 재 웅 (21631005@handong.edu)

2016 한동대학교 컴퓨터공학, 전자공학 학사
2016~ 현재 한동대학교 정보통신공학과 석사과정

관심분야 : 모델링 형식론, 모델 검증, 시뮬레이터 검증

최 창 범 (cbchoi@handong.edu)

2005 경희대학교 컴퓨터공학 학사
2007 KAIST 전산학 석사
2014 KAIST 전자공학 박사
2014~ 현재 한동대학교 ICT창업학부 조교수

관심분야 : DEVS 형식론, 소프트웨어 품질 보증, VV/A