시스템 오브 시스템즈 메타모델의 시나리오 기반 분석 및 통계적 검증 활용 가능성 분석

신용준○, 박수민, 백영민, 배두환
한국과학기술원
{yjshin, smpark, ymbaek, bae}@se.kaist.ac.kr

Scenario-based Analysis of System-of-Systems Meta-model and Applicability Analysis for Statistical Verification

Yong-Jun Shin○, Sumin Park, Young-Min Baek, Doo-Hwan Bae
School of Computing, Korea Advanced Institute of Science and Technology (KAIST)

요 약

단일 시스템으로 해결할 수 없는 복잡한 문제를 해결하기 위해 독립성과 자율성을 가지는 다수의 구성 시스템(Constituent System, CS)이 통합된 시스템인 시스템 오브 시스템즈(System-of-Systems, SoS) 개념이 대두되었다. SoS 공학에서 핵심적인 문제 중 하나는 SoS 수준의 목표 달성 여부를 검증하는 것이며, 이를 효율적으로 수행하기 위해 시뮬레이션 기반의 통계적 모델 체킹을 활용할 수 있다. 통계적 모델 체킹을 하기 위해서 시뮬레이션 가능한 SoS 모델이 필요한데, 기존 많은 연구에서는 시뮬레이션과 통계적 검증을 위한 SoS 모델링에 대한 지침을 제공해주지 못하고 있다. 이전 연구에서 SoS 모델을 생성하기 위한 온톨로지 기반의 메타모델을 정의하였는데, 본 연구에서는 M2SoS의 SoS 사례 시나리오 표현력과 검증 및 시뮬레이션에 대한 활용성을 분석하고자 한다.

1. 서 론

정보 통신 및 지능형 기술이 발달함에 따라 사용자에게 제공할 수 있는 서비스의 다양성과 질이 높아지고 있으며 이에 따라 시스템의 복잡도가 증가하고 있다. 또한 사회의 복잡도가 증가함에 따라 해결해야 할 문제의 크기가 증가하여 단일 시스템으로 해결할 수 없는 문제들이 발생한다. 이를 해결하기 위해 다수의 독립 시스템들을 하나의 거대한 시스템으로 묶는 시스템 오브 시스템즈(System-of-Systems, SoS) 개념이 대두되었다 [1]. SoS를 구성하는 구성 시스템(Constituent System, CS)은 각각 독립성과 자율성을 가지며 서로 간의 상호작용과 협력을 통해 공통의 목표를 달성하고자 한다. 예를 들어 대형 재난 대응 SoS는 인명 구조 최대화라는 SoS 수준의 목표를 위해 소방 시스템, 이송 시스템, 의료 시스템들이 협력하는 SoS라고 할 수 있다.

CS들이 협력하는 목적인 SoS 수준의 목표의 달성 여부를 검증하는 것은 SoS 공학에서 핵심적인 문제 중 하나다. 그러나 SoS의 거대한 규모와 복잡도 때문에 전통적인 정형 검증을 이용하는 것은 어렵고 높은 비용이 요구된다 [2]. SoS를 구성하는 CS들의 상호작용에 따라 목표 달성 여부가 달라질 수 있고, 독립성과 자율성을 갖는 CS들의 행동을 정확하게 예측하기 어렵다. 또한 CS는 독립적인 시스템이기 때문에 SoS의 관점에서 CS의 행동과 의사 결정 과정을 정확하게 알 수 없을 수 있으며, 불확정성을 나타낼 수도 있다.

전통적인 방법을 이용한 SoS 수준의 목표 달성 여부 검증은 SoS의 복잡성과 규모로 인한 상태 폭발 문제 등으로 인해 어렵다. 이를 해결하기 위해 통계적 모델 체킹(Statistical Model Checking, SMC)을 사용하여 정량적 SoS 목표 달성을 검증할 수 있다 [3]. SMC를 이용해 SoS를 검증하기 위해서는 시뮬레이션 가능한 SoS 모델을 만드는 작업이 필요하다. 그러나 독립적인 CS들로 구성된 SoS의 모델을 만드는 것은 큰 비용을 초래한다. 또한 SoS 모델링을 위한 전형적인 SoS 메타모델이 제안되지 않아 SoS의 목표 달성 검증을 위한 일반적인 모델 기반 프레임워크가 제안되기 어렵다. COMPASS 그룹에서 SoS 모델링 언어를 만들었지만, 구체적인 SoS 모델을 만들기 위한 기준인 SoS 메타모델은 제안된 바가 없다 [5]. 따라서 SoS 모델링을 위한 SoS 메타모델의 정의가 필요하다. SoS 메타모델은 SoS 엔지니어가 SoS를 모델링 할 때 따라야 할 지침이 될 수 있다. 또한 SoS 메타모델은 통계적 검증이 가능해야 하며 이를 위해 시뮬레이션 가능한 SoS 모델을 만들 수 있어야 한다.

따라서 본 논문은 이전 연구에서 정의한 SoS 메타모델(Meta-model for SoS, M2SoS)을 소개한다. 이후 SoS 사례 시나리오를 기반으로 M2SoS의 표현력과 SoS의 통계적 검증 활용 가능성을 확인한다. 그리고 SoS의 통계적 검증을 위한 요구사항을 도출한

* 이 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원(No. R0126-18-1101, (SW 스타랩) 모델 기반의 초대형 복잡 시스템 분석 및 검증 SW 개발)과 2017년도 정부(과학기술정보통신부)의 재원으로 한국연구재단-차세대정보・컴퓨팅기술개발사업의 지원을 받아 수행된 연구임 (2017M3C4A7066212)

다. SoS 사례 시나리오는 대형 재난 대응 시나리오와 연구팀 SoS 시나리오를 사용한다. 또한 도출된 요구사항을 반영해 메타모델 기반 시뮬레이션 및 통계적 검증 도구의 아키텍쳐를 보인다.

본 논문의 구성은 다음과 같다. 2장에서 본 연구의 배경 지식과 관련 연구를 살펴본다. 3장에서는 정의한 SoS 메타모델인 M2SoS에 대해 소개한다. 4장에서는 시나리오 기반 분석에 사용할 SoS 사례 시나리오를 보인다. 5장에서는 시나리오 기반으로 M2SoS의 SoS 표현력과 통계적 검증 활용 가능성을 분석하고 SoS 검증을 위한 요구사항 도출과 이에 따른 통계적 검증 도구의 아키텍쳐를 보인다. 6장에서는 본 연구의 한계점과 향후 연구 방향을 논의하고 7장에서 결론을 보인다.

2. 배경 지식 및 관련 연구

2.1 시스템 오브 시스템즈

단일 시스템으로 해결할 수 없는 복잡한 문제를 다수의 구성 시스템으로 해결하는 시스템을 시스템 오브 시스템즈(System-of-Systems, SoS)라고 부른다. SoS는 단일 시스템으로 해결하기 어려운 서비스, 변화 등에 의해 발생하는 문제들을 더 효과적으로 해결할 수 있는 시스템이라 할 수 있다.

SoS는 Boardman과 Sauser의 정의에 의해 5가지의 특징을 가지고 있다 [11]: 자율성(autonomy), 소속성(belonging), 연결성(connectivity), 다형성(diversity), 창발성(emergence). 자율성은 각 구성 시스템은 운영 목적에 따라 자유롭고 독립적임을 의미한다. 소속성은 더 높은 목적은 위해 시스템들이 협력함을 의미한다. 연결성은 구성 시스템들이 동적 분산 네트워크에 의한 시너지 효과를 낸다는 것을 의미한다. 다형성은 구성 시스템은 이질적인 자급자족 시스템이며, 자율적으로 발전함을 의미한다. 마지막으로 창발성은 시스템 간의 상호작용으로 SoS에 기여하는 새로운 행동이 생겨남을 의미한다.

Nielsen은 여러 문헌으로부터 SoS의 8가지 차원(dimension)에 대하여 정의했다 [1]: 자율성(autonomy), 독립성(independence), 분산(distribution), 진화(evolution), 동적 재구성(dynamic reconfiguration), 행동의 창발성(emergence of behavior), 상호의존성(interdependence), 상호운용성(interoperability). 8가지 차원에 대한 설명은 표 1에 나타난다. 본 논문은 앞서 소개한 여러 연구 중 Nielsen이 정의한 모델 기반 시스템 오브 시스템즈 공학(System of Systems Engineering, SoSE)의 8가지 차원에 따라 분석을 수행한다.

2.2 모델 기반 SoS 공학 기법

2.2.1 SoS 모델링 및 아키텍쳐

SoS는 규모가 비교적 크고 시스템 구성이 매우 복잡하기 때문에 SoS의 전반적인 시스템을 이해하고 분석하기 위해 다양한 모델링 및 아키텍쳐 기법이 연구되어 왔다. 특히, SoS가 갖는 주요 특성을 모델링 언어에 반영하고자 하는 연구가 많이 이뤄졌다. 2009년 Baldwin과 Sauser의 연구에서는 SoS의 다섯 가지 특성

표 1. Nielsen의 연구[1]에서 정의한 SoS의 8가지 차원(dimension)

차원(Dimension)	내용
자율성 (Autonomy)	CS의 행동은 SoS 수준의 규칙이 아닌 CS 자신의 규칙에 의해 통제됨
독립성 (Independence)	CS가 SoS로부터 분리되어도 CS는 운영될 수 있는 능력을 의미함
분산 (Distribution)	직접적, 간접적 연결을 통해 통신 또는 정보 공유를 가능하게 함
진화 (Evolution)	SoS는 CS의 구조 및 구성에 관계 없이 오래 지속되며 변경 가능함
동적 재구성 (Dynamic Reconfiguration)	계획되지 않은 개입에 의해 SoS의 구조 및 구성이 변화할 수 있음
행동의 창발성 (Emergence of Behavior)	CS들의 협력을 통해 생겨난 시너지 효과로 인해 발생하는 행동을 의미함
상호의존성 (Interdependence)	SoS 공동 목표를 달성하기 위해 CS가 서로 의존해야 함
상호운용성 (Interoperability)	SoS가 다양한 이질성의 CS들을 통합할 수 있는 능력을 의미함

을 이론적인 방정식 형태의 모델로 표현하였으며 이를 간단한 시뮬레이션을 통한 실험을 진행했다 [10]. SoS의 적응성 및 진화에 대한 설계를 연구한 DANSE (Designing for Adaptability and Evolution in System of Systems Engineering)의 문헌에서 역시 SoS를 모델링하기 위한 정형적 시맨틱(formal semantic)을 정의하고 계약 기반의(contract-based) 의 접근 방법을 활용한다 [17]. 그 외에도, 국방 SoS를 위한 DoDAF (DoD Architecture Framework)는 SoS의 관점 기반(viewpoint-based) 분석과 설계를 위한 아키텍쳐를 정의하였고 [20], AMADEOS 그룹 역시 관점 기반의 분석을 수행하고 시뮬레이션이 가능한 형태의 모델링을 위한 SysML 기반의 아키텍쳐 및 메타모델을 제안했다 [21].

COMPASS 그룹에서 소개한 모델 기반 SoS 연구의 로드맵에 따르면 총 11가지의 모델 기반 기법 요구사항을 정리하였다. 그 중 가장 중요한 요구사항은 SoS의 모델링과 아키텍쳐가 이종의(heterogeneous) 시스템들이 상호 연결되어 통합될 수 있도록 모델링 기법 및 아키텍쳐를 지원해야 한다는 것이다 [22]. 또한 앞서 소개한 Nielsen의 연구에서는 다양한 모델링 기법과 아키텍쳐에 대한 문헌 분석을 수행하였으며, SoS의 효과적인 모델링과 아키텍쳐를 위해서는 8가지 차원에 대해 충분하고 철저히 반영될 수 있어야 하고 SoS의 특성을 반영한 유연한 아키텍쳐 제공이 필요하다고 설명한다 [1].

2.2.2 모델 기반 SoS 통계적 검증 및 시뮬레이션

SoS는 분석과 모델링뿐만 아니라, 효과적으로 CS들을 통합하여 목표를 달성할 수 있는지에 대해 검증할 필요성이 있다. 특히 SoS 전반의 모든 구조와 행동을 검증하거나 테스팅하기 어렵기 때문에 많은 연구에서 모델로 표현된 SoS를 기반으로 모델 기반의 테스팅, 시뮬레이션, 검증 등을 수행한다. 그러나 SoS는 몇 가

지 특성으로 인해 기존에 수행하던 일반적인 모델 기반 검증 기법을 적용하는 데에 한계가 따른다. 첫번째 이유는 SoS의 상태 집합(set of states)이 매우 크기 때문에 전통적인 모델 체킹을 적용한 검증 수행이 어렵다는 점이 있고, 둘째로 SoS를 이루는 여러 CS들의 자율성과 독립성으로 인해 매우 비결정적이고 불확실한 행위를 보일 수 있기 때문이다.

위와 같은 문제를 해결하기 위해 SoS와 같은 대규모 복잡 시스템에는 통계적 모델 체킹(statistical model checking) 검증 기법이 자주 활용된다. 이는 가설 검증에 기반한 기법으로써, 시스템을 확률 모델(probabilistic model)이나 비결정적 행동을 포함한 모델로 표현하고 반복적인 시뮬레이션 수행을 통해 검증 속성이 만족되는 지에 대한 정량적 결과를 도출한다 [4]. 통계적 검증을 사용한 저자의 이전 연구는 PRISM을 활용해 네 가지 타입의 SoS를 확률 모델로 표현하고 목표 달성률을 검증한 연구가 있다 [23]. 또한 CS들의 자율적인 의사 결정 매커니즘을 행동-이익-비용(action-benefit-cost) 모델로 정의하고, 자율 로봇 시나리오를 적용한 통계적 모델 체킹 연구도 진행된 바 있다 [3]. 또한 이를 SoS 수준 목표 달성 검증 도구로 구현한 연구도 진행되었다 [12].

2.3 사례 시나리오 기반 분석

시나리오 기반 분석(scenario-based analysis)이란 특정 시나리오를 구체적인 사례로 이용하여 풀고자 하는 문제에 대한 통찰이나 구체적인 정보 및 해답을 얻는 방법이다. 시나리오 기반 분석은 문제에 대한 완벽한 해답을 내기 어렵거나 구체적인 정보를 필요로 할 때, 문제를 대표하거나 그 특성을 잘 보이는 시나리오를 이용해 문제를 분석한다. 시나리오 기반 분석은 소프트웨어 아키텍처, 모델링, 요구사항 공학 등에서 사용되었다.

소프트웨어 아키텍처 연구에서는 시나리오 기반으로 아키텍처의 요구 사항 만족 여부를 분석하거나 평가하는 연구가 진행되었다 [6]. 또한 시나리오 기반 모델링 방법과 도구에 대한 연구가 진행되었다 [8]. 요구사항 공학에서는 시나리오를 이용한 구사항 명세 및 비 기능 요구사항 분석에 대한 연구가 진행되었다 [7]. 또한 레거시 시스템에서의 관심 특징 추출 등에도 시나리오 기반 분석 방법이 적용되었다 [18]. 본 논문은 구체적인 SoS 사례 시나리오 기반 분석을 이용해 SoS 메타모델의 통계적 검증 활용 가능성을 분석한다.

3. M2SoS: SoS 메타모델

본 연구에 앞서 개발된 SoS 메타모델은 M2SoS (Meta-model for System-of-Systems)이며, 온톨로지 형식으로 개발 대상 SoS를 포괄적으로 표현하고 모델링할 수 있도록 지원한다. M2SoS를 개발하기 위해 본 연구는 SoS 모델 기반 공학 기법이 갖춰야 할 네 가지 요구사항을 만족시킬 수 있도록 설계하였다. 첫째, 운영 및 관리 독립성을 갖는 CS를 통합하여 구성된 SoS를 모델링하기 위해서는 SoS 수준 요소(SoS-level entities)와 CS 수준 요소(CS-level entities)가 명시적으로 구분되어야 한다. 둘째, SoS의 효과적인 모델링 및 검증 수행이 가능하도록 SoS 수준 목표, 요구사항, 서비스, 행동을 명시적으로 정의하고 상호 간의 관계를 표현할 수 있도록 메타모델이 지원할 수 있어야 한다. 셋째, SoS을 철저하게 분석하기 위해서는 많은 불확정성(uncertainty)을 포함하는 SoS 수준 환경 요소를 메타모델에서 명시적으로 정의하고 분류할 수 있어야 한다. 넷째, SoS 수준과 CS 수준의 시스템/소프트웨어 공학자의 충분한 이해에 기반한 의사소통 및 의사 결정을 할 수 있도록 메타모델은 도메인 지식을 표현할 수 있는 온톨로지 형식을 갖추고 있어야 한다.

앞서 정의한 네 가지 요구사항을 바탕으로 아래에서는 수준별 요소(entity)를 단계별로 정의하고자 한다. M2SoS는 기본적으로 UML의 클래스 다이어그램을 이용하여 메타모델링 되었으며, 온톨로지 형식으로 각 요소와 관계를 정의하였다. M2SoS는 SoS 수준 요소와 CS 수준 요소를 명시적으로 구분하여 정의하고, SoS 수준 목표, 요구사항, 서비스, 역할 등을 명시적으로 표현함으로써 각 요소들을 명세하는 데에 활용할 수 있다. 또한 SoS 수준 환경 요소를 정의하고 세 가지 유형으로 분류한다. 마지막으로, M2SoS는 그림 1과 같이 온톨로지 형식의 메타모델로 구성되며, M2SoS를 기반으로 특정 시나리오 모델링하기 위한 개체와 관계를 충분히 정의할 수 있는 가이드라인으로 활용할 수 있다. (전체적인 M2SoS 그림은 부록에 첨부).

[1단계] SoS 주요 구성 요소 정의

M2SoS에서는 SoS를 이루는 네 가지 주요 구성 요소를 아래와 같이 정의하며, 각 주요 구성 요소는 세부적으로 다양한 하위 구성 요소를 포함한다.

- **SoS-level Target Problem**: 개발될(된) SoS가 CS들의 역량을 바탕으로 해결하고자 하는 상위 수준의 문제를 가리키며, SoS 개발 및 운영의 목표(Goal)로 연결되는 요소
- **SoS-level Infrastructure**: SoS가 SoS-level Target Problem을 해결하고 상위 수준의 목표를 달성하기 위해 개발된 기반 시설이며, 다수의 CS를 통합(integrate)하고 조정 및 지휘(orchestrate)하는 역할을 수행하는 요소. SoS의 경계(boundary)를 결정하는 역할을 하며, CS들의 상호작용을 위한 네트워크 등의 시설을 제공하는 요소
- **Constituent System (CS Organization)**: SoS를 이루는 컴포넌트 시스템의 기본 단위이며, SoS 수준 서비스를 제공하기 위한 부분적인 역할을 수행하는 요소. CS는 새로 개발될 시스템과 이미 존재하는 레거시 시스템(legacy system)을 모두 포함하며, 관리 독립성(managerial independence)과 운영 독립성(operational independence)을 가지는 요소
- **SoS-level Environment**: SoS-level Infrastructure 외부에서 SoS와 상호작용하는 물리적이거나 가상으로 존재하는 요소

[2단계] SoS 수준 요소(SoS-level Entities) 정의

M2SoS에서는 SoS를 구성하는 상위 수준 요소를 SoS 수준 요소(SoS-level Entity)라고 정의하며, SoS 수준 요소는 다수의 CS

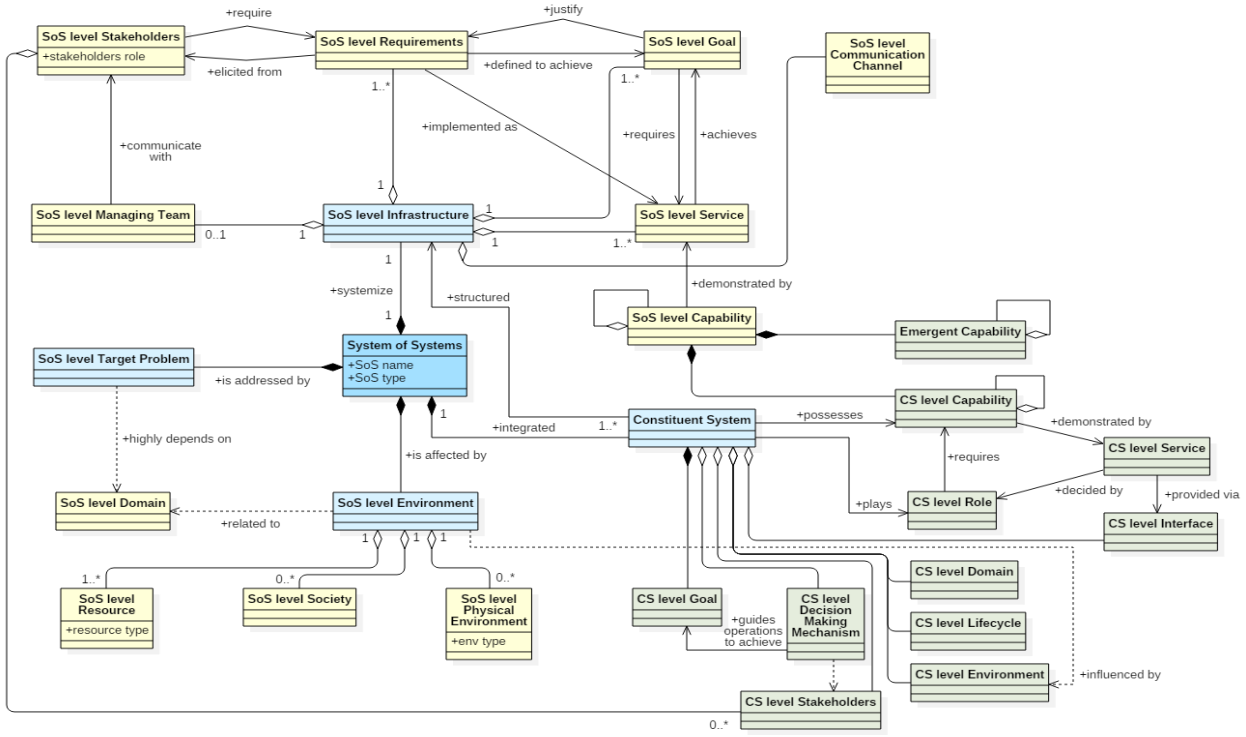

그림 1. SoS 수준 요소(SoS-level Entities)와 CS 수준 요소(CS-level Entities)를 포함하여 간략하게 표현된 M2SoS

들을 통합함으로써 상위 수준에서 정의하거나 분석해야 하는 요소들을 포함한다.

- SoS-level Goal: 다수의 CS들의 협업을 통해 달성하고자 하는 SoS 수준의 공통 목표이며, 일반적으로 독립적인 CS가 달성하기 어려운 수준의 목표로 설정
- SoS-level Requirements: SoS 수준 목표(SoS-level Goal)를 달성하기 위해 정의되는 요구사항이며, 운영 요구사항(operational requirements), 개발 요구사항(requirements), 설계 요구사항(design requirements)으로 구분하여 정의 가능
- SoS-level Service: CS들의 협업을 통해 제공되는 상위 수준의 서비스를 의미하며, 순서적(sequential), 결과적(resultant), 창발적(emergent) 행동이 서비스로 정의 가능
- SoS-level Capability: CS들의 협력을 통해 수행 가능한 SoS-level Service로부터 획득할 수 있는 SoS 수준 역량
- SoS-level Communication Channel: 여러 CS들 간, CS와 SoS 관리자 간 통신을 수행하는 데에 사용되는 통신 채널
- SoS-level Managing Team: 전체적인 SoS를 관리하고, SoS 운영에 있어 공학적 이슈를 해결하기 위한 관리팀 및 공학팀
- SoS-level Domain: 개발될 SoS가 속해있는 대상 분야(도메인)이며, 도메인 모델 등을 활용한 도메인 지식 표현을 위한 요소
- SoS-level Stakeholders: SoS 수준 서비스를 통한 목표 달성에 영향을 받는 모든 개인, 그룹 및 조직

[3단계] CS 수준 요소(CS-level Entities) 정의

M2SoS에서 정의한 CS 수준 요소(CS-level Entity)는 CS가 갖는 관리 독립성(managerial independence)과 운영 독립성(operational independence)을 표현하기 위해 SoS 수준 요소와 구분하여 정의가 필요한 요소이다. 여기서 CS 수준 요소는 CS 수준 공학자에 의해 분석 및 설계되는 요소일 수 있으며, SoS 관리자 및 공학자의 직접적인 접근 및 조작이 불가할 수 있다.

- CS-level Goal: CS가 독립적으로 갖는 기존의 시스템 목표를 의미하며, CS 수준 목표는 SoS 수준 목표와 다르거나 일부 상충할 수 있음
- CS-level Role: CS가 SoS 조직에 통합되면서 SoS 수준 목표 달성을 위해 부여된 역할
- CS-level Capability: CS가 수행 가능한 기능과 서비스로부터 도출 가능한 역량
- CS-level Service: CS의 기능적 요구사항 개발을 통해 수행 가능한 서비스로, CS 수준 목표를 달성하기 위해 구현된 요소
- CS-level Interface: CS가 다른 시스템 및 외부 환경과 통신하기 위해 필요한 인터페이스로 데이터 등을 전달하기 위해 사용되는 요소
- CS-level Decision Making Mechanism: CS가 SoS 운영 중 상위 수준의 서비스를 제공하는 동안 독립적으로 내리는 의사 결정에 필요한 매커니즘. 대표적으로 BDI (Belief-Desire-Intention) 모델이나 CBA(Cost Benefit Analysis) 등을 활용 가능
- CS-level Stakeholders: 기존의 CS 운영에 있어 CS 수준 목표를 달성하기 위한 요구사항을 정의하고, CS의 운영에 영향을 받는 모든 개인, 그룹 및 조직
- CS-level Domain: CS가 속해있는 도메인을 의미하며, 이는

SoS 수준 도메인과 다를 수 있음

- CS-level Lifecycle: CS 수준의 독립적인 운영에서 갖는 생명주기로, 항시 동일한 서비스 제공 상태를 갖지 않을 수 있는 부분을 표현하기 위한 요소
- CS-level Environment: CS가 각자의 서비스를 제공하는 데에 있어 영향을 받는 모든 외부 요소

[4단계] SoS 수준 환경(SoS-level Environment) 정의

다수의 CS가 통합되면서 구성된 SoS는 상위 수준에서 독립적인 환경 요소를 가지므로, M2SoS에서는 이를 SoS 수준 환경(SoS-level Environment)이라고 명시적으로 정의한다. 이는 SoS 수준 인프라와 상호작용 할 수 있는 모든 SoS 외부의 요소를 가리키며, 자원, 사회, 물리적 환경 요소 등을 정의할 수 있다.

- SoS-level Resource: SoS 수준 환경 요소 중 SoS의 운영에 활용될 수 있는 다양한 자원 요소로써, 인적 자원(Human Resource), 시스템 자원(System Resource), 금전적 자원(Monetary Resource) 등을 정의하기 위해 명세하는 요소
  - Resource State: 사용 가능 자원(Available Resource)과 잠재적 자원(Potential Resource)으로 구분하여 표현
- SoS-level Society: SoS 수준 환경 중 사람 혹은 외부 조직 및 시스템을 정의하기 위해 명세하는 요소
- SoS-level Physical Environment: SoS 수준 환경 요소 중 물리적으로 SoS에 영향을 주거나 SoS가 정보를 획득해야 하는 측정 가능한(measurable) 요소.

4. 메타모델의 검증 가능성 분석을 위한 시나리오

4.1 다중 손상 사고 대응 SoS (MCI Response SoS)

4.1.1 사례 시나리오 개요 및 구성

다중 손상 사고(Mass Casualty Incident, 이하 MCI)는 일반적으로 많은 수의 사상자와 피해자가 발생하여, 의료 및 구조 지원이 부족한 사고를 의미한다 [13]. MCI가 발생하는 대표적인 원인으로는 대형 사고, 자연 재해 등이 있다. MCI 대응 시나리오는 대형 사고 또는 자연 재해가 발생한 경우 상위 수준 시스템을 구성하여 부족한 의료 및 구조 자원을 지원하여 문제를 해결하는 시나리오다.

다중 손상 사고 대응 SoS(MCI Response SoS, MCIRSoS)를 SoS 레벨로, 여러 대응 시스템들을 CS로 생각한다. CS들은 독립성과 자율성을 지니고, 서로 다른 목표, 능력 그리고 서비스 등을 제공한다. MCIRSoS는 CS들과 함께 인지하는 공통의 목표가 존재하며, 목표를 달성하기 위한 능력과 역할을 가진다. 또한 환경, 인프라 그리고 자원을 가지고 있다.

MCIRSoS 시나리오에서 MCIRSoS는 구조 파견 시스템, 소방 시스템, 응급 의료 시스템 및 이송 시스템, 의료 시스템으로 구성된다. 구조 파견 시스템의 목표는 사고 정보를 통해 환자 관리를 지시하고, 소방 시스템은 환자 구조 및 화재를 진압한다. 응급 의료 시스템 및 운송 시스템은 환자 응급 처치 및 이송을 담당하고, 의료 시스템은 환자 치료를 통해 환자의 생명 유지를 목표로 한다.

MCIRSoS는 환자 95% 이상 구조, 80% 이상의 생존율을 목표로 한다. 또한 금전적 자원, 인적 자원, 도로 환경, 날씨 환경 에 영향을 받는다. MCIRSoS에는 대응을 지휘하는 관리팀이 존재하며, 정보 공유를 위한 커뮤니케이션 채널 등을 가지고 있다. MCIRSoS 사례 시나리오의 적합함을 보이기 위해 SoS의 8가지 차원으로 분석하여 표 2에 정리하였다.

4.1.2 다중 손상 사고 대응 SoS 시나리오 예시

A시에서 고층 건물에 화재가 발생하였다. 다중 손상 사고 대응 SoS(MCI Response SoS, MCIRSoS)의 목표는 환자 95% 이상을 구조하고, 80% 이상 생존할 수 있는 구조 및 치료를 제공하는 것이다. MCIRSoS의 지휘 센터는 구성 시스템에 업무를 분배한다. 현재 MCIRSoS는 1개의 구조 파견 시스템, 5개의 소방 시스 템, 2개의 응급 의료 시스템 및 이송 시스템, 1개의 의료 시스템으로 구성된다.

구조 파견 시스템은 전화 제보를 통해 화재의 위치, 크기 그리고 심각성 등을 파악한다. 또한 주변의 환경 및 지원 가능한 자원들을 파악 후 소방, 응급 의료 및 운송 시스템을 파견하고, 의료 시스템의 수용 능력을 조사한다.

소방 시스템의 소방 대원은 환자를 안전한 위치로 구조를 하는 동시에 소방차를 이용해 화재를 진압한다. 5개의 소방 시스템이 존재하므로 3명의 소방 대원은 환자를 구조하고 2명의 소방 대원은 소방차를 이용해 화재를 진압할 수 있다.

응급 의료 시스템 및 이송 시스템의 의사는 응급 처치를 제공하고, 구급차와 헬리콥터는 의료 시스템으로 환자의 이송을 담당한다. 2개의 응급 의료 시스템 및 이송 시스템이 존재하므로 각각 1대씩의 소방차와 헬리콥터를 파견한다. 소방 시스템으로부터 구조된 환자들은 구급차를 통하여 응급 처치를 진행하며 이송한다. 지상에서 구조가 어려운 경우, 헬리콥터를 이용하여 환자를 구조하여 응급 처치를 진행하며 이송한다. 이송 시스템은 의료 시스템과 통신하여 의료 시스템의 수용 능력을 판단 후 환자를 이송한다. 가까운 거리의 의료 시스템의 수용 능력이 부족할 경우 수용이 가능한 의료 시스템으로 이송한다. 또한 지휘 센터에서 이송 시스템이 부족하다고 판단하면 이송 시스템이 추가될 수 있다.

의료 시스템은 수용 능력 내에서 환자를 수용하며, 수용한 환자에 대하여 환자의 상태를 분석하여 적절한 치료를 제공한다. 현재 1개의 의료 시스템이 존재하므로, 해당 의료 시스템의 수용 능력을 초과할 경우 환자를 수용할 수 없다. 이럴 경우 지휘 센터에서는 의료 시스템의 공간과 인적 자원을 고려하여 더 많은 의료 시스템을 공급하여야 한다.

4.2 연구팀 SoS (Research Team SoS)

4.2.1 사례 시나리오 개요 및 구성

연구팀 시나리오(Research Team SoS, RTSoS)는 논문 출판을 위해 연구원들이 협업하는 과정을 나타낸 시나리오이다. RTSoS

표 2. Nielsen이 정의한 8가지 차원(dimension)에 따른 SoS 사례 시나리오 분석

차원(Dimension)	MCI 대응 SoS (MCIRSoS)	연구팀 SoS (RTSoS)
자율성 (Autonomy)	각각의 CS들은 고유한 목표를 가지고 해당 목표를 이루기 위해 각자의 규칙에 따라 행동이 통제되므로 자율성을 가짐	연구원들은 자신이 연구에 어느정도 참여할지 스스로 결정하며 결정 메커니즘을 소유하여 자율성을 가짐
독립성 (Independence)	각각의 CS들은 목표, 능력, 규칙 등에 의해 결정되어 운영하므로 독립성을 가짐	연구원은 연구팀에 소속되지 않아도 스스로 연구하고 논문을 쓸 수 있는 능력 소유하여 독립성을 가진다고 할 수 있음
분산 (Distribution)	지리적으로 다른 위치에 존재하는 CS들은 SoS 수준의 인프라와 통신을 하여 상호작용을 이루므로 분산되어 있음	연구원들은 지리적으로 분산되어 있을 수 있고, 분산된 상황에도 협업이 가능함
진화 (Evolution)	MCI의 상황이 변함에 따라 CS들의 목표, 역량, 행동 등은 변화하므로 진화함	새로운 기능, 도메인의 연구원 등 연구팀 구성 변화와 환경의 변화로 인해 진화함
동적 재구성 (Dynamic Reconfiguration)	환경의 변화로 인해 관리 팀이 재구성 될 수 있으며, 상황에 따라서는 MCIRSoS의 지휘관이 바뀔 수 있으므로 동적 재구성을 가짐	목표 논문지가 달라지거나 마감 기한이 달라지는 등 변화하는 환경과 상황에 따라 연구 팀 내부 구성을 동적으로 재구성함
창발성 (Emergence)	서로 다른 CS들 간에 협업을 하고 협업을 통해 시너지 효과를 창출하여 더 나은 목표를 달성 가능하므로 창발성을 가짐	새로운 능력이나 도메인을 가진 연구원 고용으로 연구팀의 행동 변화, 새로운 기능이 발현되어 창발성을 가짐
상호 의존성 (Interdependence)	SoS 수준의 목표 달성을 위해 각각의 CS들은 분업을 하여 행동을 하는 것으로 보아 서로 상호 의존성을 가짐	연구원들의 공통의 목표를 달성하기 위한 상호작용, 협력 및 장점 극대화함으로 상호 의존성을 가짐
상호운용성 (Interoperability)	SoS 수준의 관리 팀은 여러 개 존재할 수 있으며 관리 팀들은 서로 협력을 통해 SoS 수준의 목표를 달성할 수 있는 것으로 보아 상호운용성을 가짐	연구팀은 능력과 목표 등이 서로 다른 연구원들을 연구팀에 포함하여 상호운용성을 가짐

는 기존의 연구에서 SoS의 사례 시나리오로 사용된 바 있으며 본 연구는 이를 구체화하였다 [9]. 연구팀은 SoS 레벨로, 여러 개인 연구원들을 CS로 생각한다. CS인 개별 연구자는 독립성과 자율성을 가지는 독립 시스템으로 볼 수 있다. 따라서 연구자들은 개인의 각자 다른 능력, 목표 등을 가지며 다른 연구자들과 상호작용과 협력한다. 이 때 RTSoS는 연구자들이 함께 인지하는 공통의 목표가 있으며, 개별 연구자들이 공통의 목표를 달성하기 위해 가지는 능력, 역할을 가진다. 또한 이들 주위의 환경과 자원을 가진다.

RTSoS 시나리오의 구성은 다음과 같다. 연구팀은 개별 연구자들로 구성된다. 연구원은 교수, 박사과정, 석사과정, 등으로 나뉠 수 있다. 연구팀의 목표는 논문 및 특허 출원 실적을 달성하는 것이고, 특정 연구 분야를 가지고 있다. 연구팀은 운영 자금, 물적 자원, 연구 환경을 갖추고 있으며, 공통 연구 분야를 가진 연구 커뮤니티에 속해있다. 또한 연구팀의 기반 시설로 연구팀 관리팀, 커뮤니케이션 채널 등을 가지고 있다. SoS 사례 시나리오로 사용하기 적합함을 보이기 위해 RTSoS를 SoS의 8가지 차원으로 분석하여 표 2에 보였다.

4.2.2 연구팀 SoS 시나리오 예시

연구팀 A의 공통의 목표는 연간 국외 논문 5편 이상을 포함한 논문 10편 이상 작성이라는 실적을 달성하는 것이다. 또한 특허 출원이라는 추가 실적이 존재한다. 연구팀에는 지도 교수 1명, 박사과정 학생 3명, 석사과정 학생 4명이 있다. 지도교수는 학생의 연구를 지도하고, 박사과정은 연구 및 프로젝트 팀을 이끌고, 석사과정은 연구 및 연구 보조 등의 역할을 가진다. 이렇게 각각의 개인들은 개별의 능력과 역할이 정해진다. 연구팀은 정기적인 모임, 세미나, 면담이나 이메일을 통해 소통과 상호작용한다. 그리고 연구팀은 컴퓨터와 같은 물적 자원과 자금을 가지고 있고, 학교나 연구실 건물의 환경에 영향을 받는다. 또한 소프트웨어 공학 연구 커뮤니티와 학교의 학과에 속해있다.

지도교수 또는 지도교수와 박사과정으로 이뤄진 연구팀 관리팀은 목적 달성을 위해 연구팀 내부 소그룹을 운영한다. 연구팀 A는 현재 소그룹 3개를 운영 중이며 각각 박사과정 한명이 한 개의 소그룹을 이끈다. 박사과정 한 명이 졸업으로 연구팀을 떠나게 되었을 때, 목적 달성에 차질이 없도록 다른 소그룹의 박사과정과 석사과정 2명을 해당 소그룹에 투입하여 구조를 변경한다. 또한 연구팀 관리팀은 추가 실적 달성을 위해 변리사를 투입함으로 특허 출원 실적을 달성하도록 할 수 있다. 따라서 새로운 도메인의 전문가를 통해 연구팀은 새로운 기능을 갖게 된다.

5. M2SoS의 검증 활용성 분석

5.1 M2SoS의 사례 시나리오 표현력

M2SoS는 여러 도메인의 SoS의 요소를 모두 표현할 수 있는 능력이 있어야 하는데, 이를 위해 M2SoS가 사례 시나리오의 주요 구성 요소를 표현 가능한지 분석하였다. 분석 결과에 따라 두 가지 SoS 사례 시나리오의 구성 요소들을 M2SoS에서 정의한 요소들과 대응시킬 수 있었으며 이를 표 3에 정리하였다. 표를 통해 M2SoS에서 SoS의 주요 구성 요소를 포괄적으로 모델링하기 위해 정의했음을 알 수 있었고, SoS 메타모델 개발을 위해 필요한 요구사항도 만족됨을 확인할 수 있었다. 특히, 기존 메타모델 연구에서 중점적으로 다루지 않았던 상위 수준(SoS 수준)과 하위 수준(CS 수준) 요소 간 구분이 명확하게 이루어졌고, SoS 수준 인프라를 통해 CS를 조직하고 연결하기 위한 기반 요소들의 명세가 가능하다고 분석할 수 있다.

표 3. M2SoS의 SoS 시나리오 표현력 분석

		MCI 대응 SoS (MCIRSoS)	연구팀 SoS (RTSoS)
SoS-level Target Problem		다중 손상 사고 상황의 환자 발생	학과와 프로젝트에서 요구하는 연간 실적
SoS-level Domain		다중 손상 사고 재난 대응, 대형 사고 대응 계획	공학 연구, 소프트웨어 공학, 비즈니스 프로세스 관리
SoS-level Infrasturcture	SoS Level Requirements	● 소방 시스템과 의료 시스템 은 행동 지침에 따라야 함 ● 이송 시스템은 의료시스템과 통신을 통하여 환자를 최대한 빠르게 이송할 수 있어야 함 ● 환자 구조를 위한 구조 행동의 순서를 따라야 함	● 연구원은 연구팀이 공식 일정에 참석해야 함 ● 연구실 운영에 필요한 최소한의 연구원이 필요하고, 학사 졸업 이상의 능력이 요구됨 ● 연구원은 한국어나 영어로 의사소통이 가능해야 함
	SoS Level Goal	MCI 상황에서 발생한 환자를 95% 이상 구조하고 80%이상 생존할 수 있는 구조 및 치료	연간 논문 10편 출판, 국외 논문을 5편 이상 작성 연간 2개 이상의 프로젝트를 진행
	SoS Level Service	구조 파견 시스템과 소방 시스템의 협력을 통한 인력 분배 및 구조 시간 단축 소방 시스템과 이송 시스템의 협력으로 환자 구조 및 이송	국외 및 국내 논문 작성 프로젝트 연간 보고서 및 발표 자료 작성 연구실 내부 워크샵 진행
	SoS Level Communication Channel	문자, 재난 문자, 전화, 무전과 같은 데이터 전송을 위한 directed channel	정기적, 비정기적인 모임과 면담, 이메일 등을 통한 상시적인 소통, 정기적인 세미나
	SoS Level Stakeholders	정부, MCI 지역의 주민, 후원금을 기부한 사람 등 MCI상황에 관련된 모든 이해 관계자들	교수, 학생 등 연구실 구성 인원, 소속 학교, 연구사업 프로젝트 관계자(국가 또는 기업)
	SoS Level Managing Team	다중 손상 사고의 사건 지휘관 및 지휘 센터 (Incident Commander (IC), Command Center 등)	지도 교수 또는 지도 교수와 박사과정 학생으로 구성된 연구실 관리팀
SoS-level Environment	SoS Level Resource	버스나 택시와 같은 잠재적 이동 수단 정부 지원금, 후원금 등을 통해 사용할 수 있는 금전적 자원	연구실 운영을 위한 자금 컴퓨터, 프린터, 사무 용품 등의 연구실의 물적 자원
	SoS Level Society	다중 손상 사고 재난 대책 본부, 재난 대응 팀, 재난 구조 팀 등 다중 손상 사고에 대응할 수 있는 다양한 그룹	소프트웨어 공학 연구 커뮤니티(타 연구 그룹 등), 소속 학교의 컴퓨터 공학 학과
	SoS Level Physical Environment	지리적 환경으로 도로 상태, 신호등 등이 있으며, 기후와 날씨 환경 및 관련되는 모든 물리적인 환경 요소	소속 학교 내 연구실 건물과 연구실, 교내 학과 건물
Constituent Systems (CS)		**구조 파견 시스템:** 사고 정보를 통해 적절한 CS에 환자 관리 지시 ● 보고된 정보를 통해 사고 분석 수행 역량 필요 ● 다중 손상 사고의 규모 및 심각도를 파악하여 CS들에 업무 분담	**교수:** 연구실 운영 및 학생 지도 ● 본인 연구와 연구실의 지도 교수로서 학생의 연구 방향을 지도 ● 논문 작성 방법 지도를 통해 학생의 논문의 질을 향상시키고, 연구실이 운영될 수 있도록 프로젝트 수주
		소방 시스템: 환자 구조 및 화재 진압 ● 다중 손상 사고의 현장 상황에 적절한 장비를 사용하여 환자 구조 및 화재 진압 능력 필요 ● 환자의 위치 정보를 통해 환자에게 접근하여 환자 구조 및 안전 장소로 이송, 그리고 화재 진압	**박사 과정:** 연구 및 해외 저널 논문 출판, 석사 과정 지도 ● 연구 팀의 팀장과 프로젝트의 리더를 수행함 ● 실적 달성을 위해 연구 및 논문을 작성하고, 석사 과정 학생 면담과 회의 및 프로젝트를 진행
		응급 의료 시스템 및 이송 시스템: 최단 시간에 환자 응급 처치 및 의료 시스템으로 이송 ● 응급 처지 또는 수술을 제공 능력 및 빠른 시간 안에 의료 시스템으로 이송 가능한 능력 필요 ● 환자의 위치 정보를 통해 환자에게 접근하여 응급처치 제공 및 의료 시스템으로 이송	**석사 과정:** 연구 및 국내 저널 논문 출판, 연구 보조 ● 연구 팀의 팀원으로서 연구실에서 진행 중인 프로젝트에 참여하며 행정을 보조함 ● 실적 달성을 위해 연구 및 논문을 작성하고 프로젝트 보고서 작성 보조
		의료 시스템: 환자 치료를 통한 환자의 생명 유지 ● 병원은 환자를 수용할 수 있는 능력 필요 ● 전달 받은 환자의 상태를 분석 후 적절한 치료 제공	**변리사:** 프로젝트 실적을 위한 특허 출원 진행을 보좌 ● 연구실 실적에 필요한 특허 출원과 관련된 정보를 자문 ● 특허 출원에 문제가 발생하지 않도록 특허권 검색

다시 말해, 표 3에 정리된 내용은 M2SoS를 기반으로 개발하고자 하는 SoS를 이루고 있는 물리적이거나 가상의 구성 요소를 분석한 것이라고 할 수 있다. 이러한 분석을 통해 개발하고자 하는 SoS의 요소를 정의하고 SoS의 도메인을 표현했다면, 이를 다양한 모델링 기법을 통해 모델링 할 수 있다. M2SoS는 사용되는 모델링 기법에 중립적인 기술-중립적인(technique-neutral) 메타모델이기 때문에 각 요소에 대한 구체적인 모델링 기법의 개발은 본 논문에서 다루지 않는다.

5.2 모델 기반의 SoS 통계적 검증

5.2.1 검증을 위한 M2SoS의 개선 요구사항

M2SoS는 MCIRSoS와 RTSoS 시나리오에 대한 표현력을 갖추었으므로, M2SoS를 따라 SoS를 모델링하면 검증 대상을 충분히 표현할 수 있는 모델을 만들 수 있다. 그러나 SoS의 표현력을 갖춘 모델만으로 통계적 검증을 수행할 수는 없다. SMC를 이용한 통계적 검증을 위해서는 시나리오를 시뮬레이션 및 검증 가능해야 하며 이를 위해 필요한 요소가 정의되어야 한다. 검증을 위한 M2SoS 개선 요구사항을 도출하기 위해 AMADEOS에서 정의한 SoS 분석의 7가지 관점 중 통계적 검증과 연관된 역동성, 진화, 시간, 창발성의 관점으로 분석하였다 [14]. 다음은 이를 통해 도출된 M2SoS의 개선 요구사항이다.

시뮬레이션 실행 가능한 모델 정의: M2SoS는 시나리오를 정적으로 표현하는 능력만 가지고 있어 SoS의 역동성, 진화, 시간, 창발성의 관점은 보이지 못한다. 현재의 M2SoS를 따르는 SoS 모델이라 하더라도 통계적 검증을 위한 시뮬레이션이 불가능할 수 있다. 따라서 실행 가능한 SoS 모델로의 정의가 필요하다.

시뮬레이션 시간 정의: 시뮬레이션에서 시간 요소는 필수적이나 M2SoS는 SoS의 시간을 표현하지 못하고 있다. 시뮬레이션에서 시간을 표현하는 방식은 크게 이산 시간과 연속 시간으로 나눌 수 있다. 또한 시뮬레이션 시작 및 종료 시간에 대해서도 명시해야한다.

시나리오의 이벤트 정의: M2SoS는 검증하고자하는 SoS를 표현 가능하나 SoS의 행동과 구조의 변화에 대한 표현이 어렵다. 즉 SoS의 역동성, 진화, 창발성 관점을 표현하지 못한다. 따라서 검증 대상 모델에서 발생하는 일련의 이벤트들이 명시되어야 한다. SoS 시뮬레이션과 검증을 위해서는 SoS의 행동, 구조 변화 등을 표현할 수 있는 시나리오의 이벤트 정의가 필요하다.

검증 속성 및 모델 체커: SoS 시나리오를 검증하기 위해서는 검증 속성과 모델 체커가 필요하다. 모델 체커는 주어진 모델의 검증 속성 만족 여부를 확인한다. 그러나 M2SoS는 검증 속성과의 관계를 보이지 않아 통계적 검증에 직접 이용하기 어렵다. 따라서 검증 속성과 이를 검증 할 체커를 명시 해야한다.

5.2.2 모델 기반 통계적 검증을 위한 아키텍쳐 설계

위 절에서 도출한 SoS 검증을 위한 요구사항 요소를 반영해 M2SoS를 따르는 SoS 모델을 통계적 검증하기 위한 아키텍쳐를 설계했다. 그림 2는 SoS 메타모델 기반의 시뮬레이션 및 통계적 검증 도구의 아키텍쳐다.

시뮬레이션 실행 가능한 모델(Simulation Executable SoS Model)은 M2SoS를 따르는 모델(SoS Model)을 Java와 같은 프로그래밍 언어로 변환하여 만들어진다. 따라서 M2SoS의 표현력과 현실성은 검증의 대상이 되는 실행 가능한 모델에 직접적으로 반영되고 영향을 준다. 시뮬레이션 시간(Time)은 실행 가능한 SoS 모델의 CS들의 생명 주기, 서비스 수행 시간 등 동적인 요소에 표현된다. 또한 시나리오의 각 이벤트에 발생 시간, 종료 시간 등이 함께 명시된다. 이런 시뮬레이션 시간 요소는 통계적 검증 수행 시간에 영향을 준다.

시나리오(SoS Scenario)는 검증 대상인 SoS에서 나타날 수 있는 이벤트들로 구성되고 SoS 시뮬레이션 엔진(Simulation Engine)에 의해 실행된다. 또한 SoS 시나리오는 불확정적인 요소나 동적인 요소를 가질 수 있다. 검증 속성(Verification Property)은 SoS 목표를 정형적으로 표현한 것이다. 모델 체커는 정형적으로 표현된 검증 속성으로 시뮬레이션의 결과를 검증한다.

M2SoS에 기반한 SoS 모델과 검증을 위해 요구사항으로부터 정의된 요소들은 시뮬레이션 기반 SoS 검증(Simulation-based Verification for SoS)에 사용된다. 이는 크게 SoS 시뮬레이터(SoS Simulator)와 통계적 검증기(Statistical Verifier)로 구성된다. 실행 가능한 모델과 시나리오는 각각 시뮬레이션의 모델과 시나리오 데이터로 시뮬레이션 엔진의 입력이 된다. 시뮬레이션 엔진은 입력들을 실행하고 결과물인 트레이스를 통계적 검증기의 입력으로 전달한다. 검증 체커는 시뮬레이션 트레이스와 SoS 목표를 표현한 검증 속성을 입력으로 받아 검증 결과를 반환하고 이를 분석한다. 본 아키텍쳐는 SoS의 8가지 차원을 표현할 수 있는 M2SoS에 기반하여, M2SoS가 가지는 표현력을 보존하며 통계적 검증을 수행할 수 있다. 또한 이를 기반으로 SoS 모델링과 검증을 통합하는 연구로 이어질 수 있을 것이다.

6. 연구 한계점 분석 및 향후 연구 방향

시나리오 기반 분석에 사용한 SoS 시나리오: 본 연구의 시나리오는 메타모델 분석을 위한 용도로 저자들에 의해 작성되었다. 즉, 분석 대상인 M2SoS에 기반하여 기술된 시나리오라고 보일 수 있다. 그러나 MCIRSoS의 경우 다중 손상 사고 대응에 대한 전문 문헌을 30건 이상 참고하여 현실을 나타낸 시나리오이고, RTSoS는 선행 연구에서 모델링과 시뮬레이션이 수행되었던 시나리오에 구체적인 스토리텔링을 추가하여 기술한 것이다 [10]. 그리고 기존의 SoS 연구들에서는 본 논문의 시나리오와 같이 구체적으로 기술된 공통된 시나리오가 없었다는 문제가 있다. 따라서 본 연구를 기반으로 SoS 연구에 공통적으로 사용될 수 있는 구체적인 시나리오를 확립해나가는 연구가 필요할 것이다.

전통적인 검증 기법 고려의 부재: 본 연구는 전통적인 검증 기법을 고려하지 않으며 통계적 모델 체킹에만 집중하고 있다. 실제로 전통적인 검증 방법을 사용해 SoS를 검증하는 연구가 존재

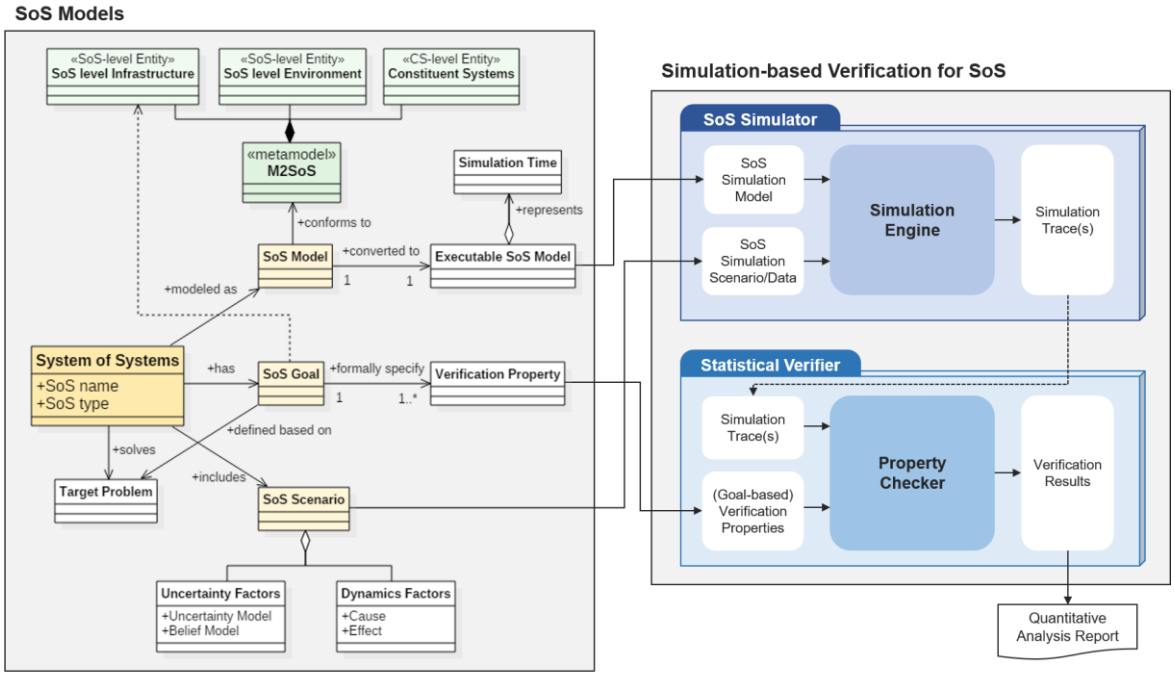

그림 2. SoS 메타모델 기반 시뮬레이션 및 통계적 검증 도구의 아키텍쳐

한다 [15]. 그러나 전통적인 검증 기법을 이용한 SoS 연구의 경우, 제한된 행동을 하는 디바이스간의 프로토콜 수준의 사례에만 적용하여 그 크기와 복잡도가 한정된다. 하지만 MCIRSoS와 같은 초대형 SoS의 경우 SoS가 가지는 상태의 수가 무한에 가까워 모델 체킹과 같은 전통적인 검증 기법에 의해 다뤄지기 어렵고, SMC를 이용한 검증이 효과적임을 저자의 이전 연구를 통해 확인했다 [23]. 또한 SoS 통계적 검증 도구를 개발 중에 있어, 통계적 검증에 기반한 SoS 연구를 지속적으로 이어나갈 수 있을 것이다.

메타모델에 기반한 모델링 결과의 부재: 본 연구는 정의한 M2SoS를 시나리오 기반으로 요소를 도출하여 두 가지 SoS 도메인에서의 표현력을 분석한 후, 최종적으로 통계적 검증에의 활용 가능성을 분석하였다. 그러나 본 논문에서는 M2SoS를 기반으로 요소를 도출하여 비교 분석을 수행했지만, 두 가지 사례 SoS의 실제 모델링 결과를 통한 표현력 분석은 이루어지지 않았다. 그 이유는 현재의 M2SoS는 개발하고자 하는 여러 도메인의 SoS를 표현할 수 있도록 도메인 일반적인(domain-generic) 표현을 지원하는 데에 초점을 맞추고 있기 때문이다. 또한, 본 연구는 다중 에이전트 시스템을 대상으로 한 Gaia 방법론과 같이 기술 중립적(technique-neutral)인 방법론의 기반 메타모델로 활용하는 데에 목적이 있다 [16].

7. 결 론

본 연구는 이전 연구에서 개발한 SoS 메타모델인 M2SoS를 소개하고, SoS 표현력과 통계적 검증 가능성을 시나리오 기반으로 분석했다. 두 가지 도메인의 SoS 사례 시나리오를 8가지 차원에 기반해 분석한 결과, M2SoS가 SoS의 모델링 및 도메인 지식을 표현하기 위한 가이드라인으로 활용될 수 있음을 보였다. 하지만 현재 개발된 메타모델을 시뮬레이션 기반의 통계적 검증에 활용하기 위해서 M2SoS의 개선을 위한 요구사항 정의와 통계적 검증 기법 및 도구의 아키텍쳐 설계가 필요했다. 이를 위해 본 연구에서는 M2SoS가 통계적 검증에 활용되기 위해 만족해야 할 네 가지 개선 요구사항(시뮬레이션 모델, 시간 요소, 시나리오와 이벤트, 검증 속성 및 모델 체커)을 정의하였으며, M2SoS 기반 통계적 검증 도구의 아키텍처를 제안했다. 향후 연구로는, 본 연구에서 제안한 SoS 통계적 검증 도구의 아키텍처를 활용해 SoS의 시뮬레이션 기반 통계적 검증 도구(Simulation-based Statistical Verification Framework for SoS)를 개발하고자 한다. 또한 본 논문에서 소개한 두 가지 SoS 사례 시나리오를 보다 구체화하여 시뮬레이션 및 검증에 실제 적용할 예정이다.

참 고 문 헌

[1] Nielsen, Claus Ballegaard, et al. "Systems of systems engineering: basic concepts, model-based techniques, and research directions." ACM Computing Surveys (CSUR) 48.2: 18, 2015.

[2] Kim, Youngjoo, et al. "Statistical model checking for safety critical hybrid systems: An empirical evaluation." Haifa Verification Conference. Springer Berlin Heidelberg, 2012.

[3] 김준호, 신동환, 배두환 (2017). 시스템 오브 시스템즈 수준의 목표 달성 검증을 위한 행동-이익-비용 모델과 통계적 모델 체킹 적용 연구. 정보과학회 컴퓨팅의 실제 논문지, 23(4), 256-261.

[4] Legay, A., Delahaye, B., & Bensalem, S. (2010). Statistical Model Checking: An Overview. RV, 10, 122-135.

[5] Woodcock, J., Cavalcanti, A., Fitzgerald, J., Larsen, P., Miyazawa, A., & Perry, S., "Features of CML: A formal modelling language for systems of systems." 2012 7th International Conference on IEEE System of Systems Engineering (SoSE), pp. 1-6, Jul. 2012. [5] Kazman, R., Abowd, G., Bass, L., & Clements, P. (1996). Scenario-based analysis of software architecture. IEEE software, 13(6), 47-55.

[6] Kazman, R., Abowd, G., Bass, L., & Clements, P. (1996). Scenario-based analysis of software architecture. IEEE software, 13(6), 47-55.

[7] Sutcliffe, A. (1998). Scenario-based requirements analysis. Requirements engineering, 3(1), 48-65.

[8] Greenyer, J., Gritzner, D., Gutjahr, T., König, F., Glade, N., Marron, A., & Katz, G. (2017). ScenarioTools-A tool suite for the scenario-based modeling and analysis of reactive systems. Science of Computer Programming, 149, 15-27.

[9] Baldwin, W. C., Sauser, B., & Cloutier, R. (2015). Simulation approaches for system of systems: Events-based versus agent based modeling. Procedia Computer Science, 44, 363-372.

[10] Baldwin, W. C., & Sauser, B. (2009, May). Modeling the characteristics of system of systems. In System of Systems Engineering, 2009. SoSE 2009. IEEE International Conference on(pp. 1-6). IEEE.

[11] Boardman, John and Brian Sauser. 2006. System of systems - the meaning of Of. In the Proceedings of the 2006 IEEE/SMC International Conference on System ©2009 IEEE of Systems Engineering, 118-123, Los Angeles, CA. April 24-26.

[12] 진민규, 신동환, 김준호, 배두환. (2017). 시스템 오브 시스템즈 수준의 목표 달성 검증 도구. 한국정보과학회 학술발표논문집, 552-554.

[13] CHONG, Shao Foong. Mass Casualty Incident. In: Disaster Medicine. Springer London, 2013. p. 165-177.

[14] Mori, M., Ceccarelli, A., Lollini, P., Fromel, B., Brancati, F., Bondavalli, A., Andrea, "Systems-of-systems modeling using a comprehensive viewpoint-based SysML profile," (2017), Journal of Software: Evolution and Process, Article in Press

[15] Faldik, O., Payne, R., Fitzgerald, J., & Buhnova, B. (2017). Modelling System of Systems Interface Contract Behaviour. arXiv preprint arXiv:1703.07037.

[16] L, Cernuzzi, T. Juan, L. Sterling, and F. Zambonelli, "The Gaia Methodology: Basic Concepts and Extensions," Methodologies and Software Engineering for Agent Systems, pp. 69-88.

[17] A. Arnold, B. Boyer, and A. Legay, "Contracts and Behavioral Patterns for SoS: The EU IP DANSE Approach," In Proceedings AiSoS 2013, arXiv:1311.3195 (https://arxiv.org/abs/1311.3631)

[18] Kim, J. A., Lee, H., Jung, R., & Kim, S. (2014). Goal and Scenario-based Feature Identification Techniques from Legacy System. International Journal of Software Engineering and Its Applications, 8(2), 145-150.

[19] Song, Jiyoung & Baek, Young-Min & Jin, Mingyu & Jee, Eunkyoung & Bae, Doo-Hwan. (2017). SoS GaP slicer: SoS Goal and PRISM Models for Change-Responsive Verification of SoS.

[20] M. Hause, "The Unified Profile for DoDAF/MODAF (UPDM) Enabling Systems of Systems on Many Levels," 4th Annual IEEE Systems Conference, 2010.

[21] M. Mori, et al., "Systems-of-Systems Modeling Using a Comprehensive Viewpoint-Based SysML Profile," Journal of Software: Evolution and Process, Special Issue - HASE 2016, 2017.

[22] COMPASS Group, "Roadmap for Research in Model-Based SoS Engineering," Comprehensive Modelling for Advanced Systems of Systems (COMPASS) - Public Document (Grant Agreement: 2872829), 2014.

[23] D. Seo, D. Shin, Y. M. Baek, J. Song, W. Yun, J. Kim, E. Jee, and D. H. Bae, "Modeling and Verification for Different Types of System of Systems using PRISM," Workshop of Software Engineering for System of Systems (SESoS), 2016.

부 록

부록 A. M2SoS: Meta-model for System-of-Systems