(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0089145
(43) 공개일자 2022년06월28일

(51) 국제특허분류(Int. Cl.)
G06F 30/18 (2020.01) *G06F 111/20* (2020.01)
G06F 30/12 (2020.01) *G06F 30/20* (2020.01)
(52) CPC특허분류
G06F 30/18 (2020.01)
G06F 30/12 (2020.01)
(21) 출원번호 10-2020-0179505
(22) 출원일자 2020년12월21일
심사청구일자 2020년12월21일

(71) 출원인
한국과학기술원
대전광역시 유성구 대학로 291(구성동)
(72) 발명자
배두환
대전광역시 유성구 대학로 291 (구성동, 한국과학기술원)
백영민
대전광역시 유성구 대학로 291 (구성동, 한국과학기술원)
(뒷면에 계속)
(74) 대리인
양성보

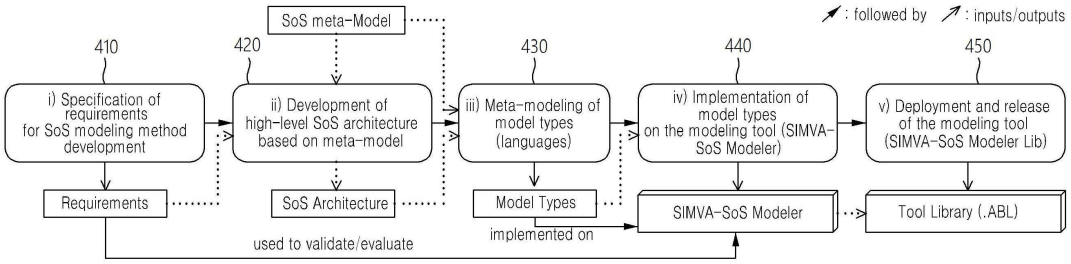
전체 청구항 수 : 총 15 항

(54) 발명의 명칭 **시스템 오브 시스템즈 시각적 모델링 시스템 오브 시스템즈 시각적 모델링 방법 및 장치**

(57) 요 약

시스템 오브 시스템즈 시각적 모델링 방법 및 장치가 제시된다. 일 실시예에 따른 컴퓨터 장치를 통해 구현되는 시스템 오브 시스템즈(System-of-Systems, SoS) 시각적 모델링 방법은, 분석, 설계, 시뮬레이션 및 검증 중 적어도 어느 하나 이상의 관점에서 SoS 모델링 방법을 제공하기 위한 요구사항을 수집하는 단계; 수집된 상기 요구사항에 기초하여 적어도 하나 이상의 요구되는 모델 유형을 식별하는 단계; 및 식별된 상기 요구되는 모델 유형을 메타 모델링하여 메타 모델 기반 모델 유형을 생성하는 단계를 포함하여 이루어질 수 있다.

대 표 도



(52) CPC특허분류
G06F 30/20 (2020.01)
G06F 2111/20 (2020.01)
(72) 발명자
젤라렘 빌레이
대전광역시 유성구 대학로 291 (구성동, 한국과학기술원)
신용준
대전광역시 유성구 대학로 291 (구성동, 한국과학기술원)

이 발명을 지원한 국가연구개발사업
과제고유번호 1711120092
과제번호 20200017950011001
부처명 과학기술정보통신부
과제관리(전문)기관명 정보통신기획평가원
연구사업명 정보통신.방송 연구개발사업
연구과제명 (N01200948)(통합EZ)엣지 클라우드에서 고신뢰 고사용성 빅데이터 플랫폼 및 분석 예측 서비스 기술 개발(2020년도)
기 여 율 1/2
과제수행기관명 한국과학기술원
연구기간 2020.07.01 ~ 2020.12.31
이 발명을 지원한 국가연구개발사업
과제고유번호 1711103148
과제번호 20150002500061001
부처명 과학기술정보통신부
과제관리(전문)기관명 정보통신기획평가원
연구사업명 정보통신.방송 연구개발사업
연구과제명 (통합EZ)(SW 스타랩) 모델 기반의 초대형 복잡 시스템 분석 및 검증 SW 개발(2020)
기 여 율 1/2
과제수행기관명 한국과학기술원
연구기간 2020.01.01 ~ 2020.12.31

명 세 서

청구범위

청구항 1

컴퓨터 장치를 통해 구현되는 시스템 오브 시스템즈(System-of-Systems, SoS) 시각적 모델링 방법에 있어서,

분석, 설계, 시뮬레이션 및 검증 중 적어도 어느 하나 이상의 관점에서 SoS 모델링 방법을 제공하기 위한 요구사항을 수집하는 단계;

수집된 상기 요구사항에 기초하여 적어도 하나 이상의 요구되는 모델 유형을 식별하는 단계; 및

식별된 상기 요구되는 모델 유형을 메타 모델링하여 메타 모델 기반 모델 유형을 생성하는 단계

를 포함하는, 시스템 오브 시스템즈 시각적 모델링 방법.

청구항 2

제1항에 있어서,

상기 요구사항을 수집하는 단계는,

생성된 모델이 충족되어야 하는 상기 요구사항의 집합을 수집하여 출력하는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 방법.

청구항 3

제1항에 있어서,

상기 요구되는 모델 유형을 식별하는 단계는,

상기 요구되는 모델 유형의 체계적인 식별을 위해 SoS 메타 모델을 입력으로 사용하여 상기 상위 레벨 SoS 아키텍처의 레벨 및 레이어(layer)를 정의하는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 방법.

청구항 4

제1항에 있어서,

상기 요구되는 모델 유형을 식별하는 단계는,

상기 요구되는 모델 유형은 소프트웨어와 하드웨어 구성요소, 에이전트, 서비스, 인적, 조직 요소, 데이터 중 적어도 어느 하나 이상을 포함하는 SoS 엔티티를 다루고, 상기 요구사항을 충족하기 위해 복수 개의 모델 유형으로 분류되는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 방법.

청구항 5

제4항에 있어서,

상기 메타 모델 기반 모델 유형을 생성하는 단계는,

상기 분류에 따라 상기 복수 개의 모델 유형의 메타 모델을 생성하는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 방법.

청구항 6

제1항에 있어서,

상기 메타 모델 기반 모델 유형은 모델링 툴(modeling tool)에서 구현되는 단계

를 더 포함하는, 시스템 오브 시스템즈 시각적 모델링 방법.

청구항 7

제6항에 있어서,

상기 메타 모델 기반 모델 유형은 모델링 툴에서 구현되는 단계는,

상기 메타 모델 기반 모델 유형은 SIMVA-SoS Modeler라는 모델링 툴에서 모델링 언어로 구현되는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 방법.

청구항 8

제6항에 있어서,

상기 모델링 툴을 배포 또는 공개하는 단계

를 더 포함하는, 시스템 오브 시스템즈 시각적 모델링 방법.

청구항 9

제8항에 있어서,

상기 모델링 툴을 배포 또는 공개하는 단계는,

다른 도메인 방법론자(domain methodologist)에 의해 확장될 수 있는 도구 라이브러리(tool library)로 개방형 모델링 커뮤니티(open-modeling community)에 배포 또는 공개되는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 방법.

청구항 10

시스템 오브 시스템즈(System-of-Systems, SoS) 시각적 모델링 장치에 있어서,

분석, 설계, 시뮬레이션 및 검증 중 적어도 어느 하나 이상의 관점에서 SoS 모델링 방법을 제공하기 위한 요구사항을 수집하는 요구사항 수집부;

수집된 상기 요구사항에 기초하여 적어도 하나 이상의 요구되는 모델 유형을 식별하는 모델 유형 식별부; 및

식별된 상기 요구되는 모델 유형을 메타 모델링하여 메타 모델 기반 모델 유형을 생성하는 메타 모델링부

를 포함하는, 시스템 오브 시스템즈 시각적 모델링 장치.

청구항 11

제10항에 있어서,

상기 모델 유형 식별부는,

상기 요구되는 모델 유형의 체계적인 식별을 위해 SoS 메타 모델을 입력으로 사용하여 상기 상위 레벨 SoS 아키텍처의 레벨 및 레이어(layer)를 정의하는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 장치.

청구항 12

제10항에 있어서,

상기 모델 유형 식별부는,

상기 요구되는 모델 유형은 소프트웨어와 하드웨어 구성요소, 에이전트, 서비스, 인적, 조직 요소, 데이터 중 적어도 어느 하나 이상을 포함하는 SoS 엔티티를 다루고, 상기 요구사항을 충족하기 위해 복수 개의 모델 유형으로 분류되는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 장치.

청구항 13

제10항에 있어서,

상기 메타 모델 기반 모델 유형이 구현되는 모델링 툴(modeling tool)

을 더 포함하는, 시스템 오브 시스템즈 시각적 모델링 장치.

청구항 14

제13항에 있어서,

상기 모델링 툴에서,

상기 메타 모델 기반 모델 유형은 SIMVA-SoS Modeler라는 모델링 툴에서 모델링 언어로 구현되는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 장치.

청구항 15

제13항에 있어서,

상기 모델링 툴을 배포 또는 공개하는 모델링 툴 배포부

를 더 포함하고,

상기 모델링 툴 배포부는,

다른 도메인 방법론자(domain methodologist)에 의해 확장될 수 있는 도구 라이브러리(tool library)로 개방형 모델링 커뮤니티(open-modeling community)에 배포 또는 공개되는 것

을 특징으로 하는, 시스템 오브 시스템즈 시각적 모델링 장치.

발명의 설명

기 술 분 야

[0001] 아래의 본 발명의 실시예들은 시스템 오브 시스템즈 시각적 모델링 방법 및 장치에 관한 것으로, 더욱 상세하게는 모델 기반 SoS 엔지니어링(Model-Based SoS Engineering, MBSoSE)을 위한 범용 모델링 방법 및 장치에 관한 것이다.

배 경 기 술

[0002] 최근 몇 년 동안, 많은 소프트웨어 집약적 시스템 도메인은 새로운 행동과 목표 지향적 조정을 기반으로 여러 소프트웨어의 집합적 기능을 활용한다. 예를 들어, 많은 네트워크 시스템과 분산 시스템은 사물인터넷(Internet-of-Things, IoT), 사이버 물리 시스템(Cyber-Physical Systems, CPS), 멀티 에이전트 시스템(Multi-Agent Systems, MAS)과 같이 다른 전문화된 시스템 도메인으로 확장되었다. 이들은 공유되고 상위 레벨(high-level)의 공통 목표를 달성하는 데 기여할 수 있는 잠재적 기능을 가진 소프트웨어 집약적인 구성요소를 채택하여 이점을 취하고자 한다.

[0003] 한편, 소프트웨어 집약적인 통합 시스템의 품질은 향상되었고 그 크기와 복잡성은 점점 커지고 있다. 이러한 변화와 함께, 시스템 오브 시스템즈(System-of-Systems, SoS)라는 보다 복잡한 시스템 도메인이 등장했다. ISO/IEC/IEEE 15288 부록 G는 SoS를 어떤 구성 요소도 스스로 달성할 수 없는 목표 기반 과제에 대한 구성 요소 집합을 취합하는 시스템 유형으로 정의한다.

[0004] 기존의 소프트웨어/시스템에 비해 SoS는 엔지니어링을 더욱 까다롭게 만드는 독특한 특성 조합을 가지고 있다. 시스템 특성에 대한 가장 뚜렷한 차이는 구성 요소의 어두움이다. 시스템 엔지니어에게 있어 개별 수행자에 대한 포괄적인 이해와 통제는 거의 불가능하다. 또한, 구성원들은 자신의 행동을 결정하고, 독점적으로 자원을 관리할 뿐만 아니라, SoS가 있든 없든 존재하며 활동할 수 있다. 기존의 많은 분산형 및 네트워크형 시스템은 필요한 기능을 수행할 수 있는 다중 구성요소 시스템으로 구성되지만, 상호간에, 그리고 상위 레벨 시스템으로

부터 모두 높은 독립성을 지니지는 못한다. 반대로, SoS의 구성 시스템(CS)은 운영과 관리 측면에서 매우 자율적이고 독립적이다. 이러한 특성은 복수의 CS 간에 불확실한 상호작용을 야기한다. 이는 SoS가 목표 달성을 위해 활용할 수 있는 다양한 돌발 행동으로 이어질 수 있다.

[0005] SoS 엔지니어링(SoSE)의 목표는 불확실성 속에서 복수의 CS의 돌발 행동을 올바른 방향으로 추진하는 것이다. 그러나 기존 연구(비특허문헌 1)에 따르면 SoS의 규모와 복잡성 때문에 가능한 모든 출현을 예측하고 엔지니어링하는 것은 거의 불가능하다. 또한, SoSE에 기존의 시스템/소프트웨어 엔지니어링 기법을 적용하는 것은 많은 경우 구성요소를 블랙박스 요소로 간주해야 하기 때문에 많은 어려움이 있다. 따라서 모델 기반 소프트웨어/시스템 엔지니어링은 체계적인 SoS 엔지니어링을 위해 정제, 개선 및 전문화되어야 한다.

선행기술문헌

비특허문헌

[0006] (비특허문헌 0001) I. Sommerville, D. Cliff, R. Calinescu, J. Keen, T. Kelly, M. Kwiatkowska, J. Mcdermid, and R. Paige, "Large-scale complex it systems," Communications of the ACM, vol. 55, no. 7, pp. 71-77, 2012.

발명의 내용

해결하려는 과제

[0007] 본 발명의 실시예들은 시스템 오브 시스템즈 시각적 모델링 방법 및 장치에 관하여 기술하며, 보다 구체적으로 상위의 목표를 달성하기 위해 단일 시스템들이 하나의 대규모 복잡 시스템을 이루는 시스템 오브 시스템즈(System-of-Systems, 이하 SoS)의 시각적 모델링을 위한 방법 및 장치를 제공한다.

[0008] 본 발명의 실시예들은 모델 기반 SoS 엔지니어링(Model-Based SoS Engineering, MBSoSE)에 특화된 범용 모델링 방법을 제공하며, 확장 가능한 형태의 SoS 모델(메타 모델)을 제공하는 시스템 오브 시스템즈 시각적 모델링 방법 및 장치를 제공하는데 있다.

과제의 해결 수단

[0009] 일 실시예에 따른 컴퓨터 장치를 통해 구현되는 시스템 오브 시스템즈(System-of-Systems, SoS) 시각적 모델링 방법은, 분석, 설계, 시뮬레이션 및 검증 중 적어도 어느 하나 이상의 관점에서 SoS 모델링 방법을 제공하기 위한 요구사항을 수집하는 단계; 수집된 상기 요구사항에 기초하여 적어도 하나 이상의 요구되는 모델 유형을 식별하는 단계; 및 식별된 상기 요구되는 모델 유형을 메타 모델링하여 메타 모델 기반 모델 유형을 생성하는 단계를 포함하여 이루어질 수 있다.

[0010] 상기 요구사항을 수집하는 단계는, 생성된 모델이 충족되어야 하는 상기 요구사항의 집합을 수집하여 출력할 수 있다.

[0011] 상기 요구되는 모델 유형을 식별하는 단계는, 상기 요구되는 모델 유형의 체계적인 식별을 위해 SoS 메타 모델을 입력으로 사용하여 상기 상위 레벨 SoS 아키텍처의 레벨 및 레이어(layer)를 정의할 수 있다.

[0012] 상기 요구되는 모델 유형을 식별하는 단계는, 상기 요구되는 모델 유형은 소프트웨어와 하드웨어 구성요소, 에이전트, 서비스, 인적, 조직 요소, 데이터 중 적어도 어느 하나 이상을 포함하는 SoS 엔티티를 다루고, 상기 요구사항을 충족하기 위해 복수 개의 모델 유형으로 분류될 수 있다.

[0013] 상기 메타 모델 기반 모델 유형을 생성하는 단계는, 상기 분류에 따라 상기 복수 개의 모델 유형의 메타 모델을 생성할 수 있다.

[0014] 상기 메타 모델 기반 모델 유형은 모델링 툴(modeling tool)에서 구현되는 단계를 더 포함할 수 있다.

[0015] 상기 메타 모델 기반 모델 유형은 모델링 툴에서 구현되는 단계는, 상기 메타 모델 기반 모델 유형은 SIMVA-SoS Modeler라는 모델링 툴에서 모델링 언어로 구현될 수 있다.

[0016] 상기 모델링 툴을 배포 또는 공개하는 단계를 더 포함할 수 있다.

[0017] 상기 모델링 툴을 배포 또는 공개하는 단계는, 다른 도메인 방법론자(domain methodologist)에 의해 확장될 수 있는 도구 라이브러리(tool library)로 개방형 모델링 커뮤니티(open-modeling community)에 배포 또는 공개될 수 있다.

[0018] 다른 실시예에 따른 시스템 오브 시스템즈(System-of-Systems, SoS) 시각적 모델링 장치는, 분석, 설계, 시뮬레이션 및 검증 중 적어도 어느 하나 이상의 관점에서 SoS 모델링 방법을 제공하기 위한 요구사항을 수집하는 요구사항 수집부; 수집된 상기 요구사항에 기초하여 적어도 하나 이상의 요구되는 모델 유형을 식별하는 모델 유형 식별부; 및 식별된 상기 요구되는 모델 유형을 메타 모델링하여 메타 모델 기반 모델 유형을 생성하는 메타 모델링부를 포함하여 이루어질 수 있다.

[0019] 상기 모델 유형 식별부는, 상기 요구되는 모델 유형의 체계적인 식별을 위해 SoS 메타 모델을 입력으로 사용하여 상기 상위 레벨 SoS 아키텍처의 레벨 및 레이어(layer)를 정의할 수 있다.

[0020] 상기 모델 유형 식별부는, 상기 요구되는 모델 유형은 소프트웨어와 하드웨어 구성요소, 에이전트, 서비스, 인적, 조직 요소, 데이터 중 적어도 어느 하나 이상을 포함하는 SoS 엔티티를 다루고, 상기 요구사항을 충족하기 위해 복수 개의 모델 유형으로 분류될 수 있다.

[0021] 상기 메타 모델 기반 모델 유형이 구현되는 모델링 툴(modeling tool)을 더 포함할 수 있다.

[0022] 상기 모델링 툴에서, 상기 메타 모델 기반 모델 유형은 SIMVA-SoS Modeler라는 모델링 툴에서 모델링 언어로 구현될 수 있다.

[0023] 상기 모델링 툴을 배포 또는 공개하는 모델링 툴 배포부를 더 포함하고, 상기 모델링 툴 배포부는, 다른 도메인 방법론자(domain methodologist)에 의해 확장될 수 있는 도구 라이브러리(tool library)로 개방형 모델링 커뮤니티(open-modeling community)에 배포 또는 공개될 수 있다.

발명의 효과

[0024] 본 발명의 실시예들에 따르면 모델 기반 SoS 엔지니어링(Model-Based SoS Engineering, MBSoSE)에 특화된 범용 모델링 방법을 제공하며, 확장 가능한 형태의 SoS 모델(메타 모델)을 제공하는 시스템 오브 시스템즈 시각적 모델링 방법 및 장치를 제공할 수 있다.

도면의 간단한 설명

[0025] 도 1은 일 실시예에 따른 모델링 방법 및 아키텍처의 구성 요소를 나타내는 도면이다.

도 2는 일 실시예에 따른 레벨 및 레이어를 개념화하는 상위 레벨 SoS 아키텍처를 나타내는 도면이다.

도 3은 일 실시예에 따른 SoS 엔지니어링용 메타 모델(M2SoS)을 단순화하여 나타내는 도면이다.

도 4는 일 실시예에 따른 시스템 오브 시스템즈 시각적 모델링 방법을 나타내는 도면이다.

도 5는 일 실시예에 따른 시스템 오브 시스템즈 시각적 모델링 장치를 나타내는 블록도이다.

도 6은 일 실시예에 따른 모델 유형의 분류를 설명하기 위한 도면이다.

도 7은 일 실시예에 따른 SIMVA-SoS Modeler의 사용자 인터페이스의 예시를 나타내는 도면이다.

발명을 실시하기 위한 구체적인 내용

[0026] 이하, 첨부된 도면을 참조하여 실시예들을 설명한다. 그러나, 기술되는 실시예들은 여러 가지 다른 형태로 변형될 수 있으며, 본 발명의 범위가 이하 설명되는 실시예들에 의하여 한정되는 것은 아니다. 또한, 여러 실시예들은 당해 기술분야에서 평균적인 지식을 가진 자에게 본 발명을 더욱 완전하게 설명하기 위해서 제공되는 것이다. 도면에서 요소들의 형상 및 크기 등은 보다 명확한 설명을 위해 과장될 수 있다.

[0028] 최근에는 집합적이고 협업적인 시스템 기능을 활용해야 하는 필요성 때문에 시스템 오브 시스템즈(System-of-Systems, SoS) 도메인이 등장하고 있다. SoS 엔지니어링(SoSE)에 대한 관심이 커짐에 따라 본 발명의 실시예에서는 SoS의 모델 기반 분석과 설계에 초점을 맞추고 있으며, 모델 기반 SoS 엔지니어링(Model-Based SoS Engineering, MBSoSE)을 위한 범용 모델링 방법을 제안한다.

[0029] 실시예들은 MBSoSE 접근방식의 모델링 방법이 충족해야 하는 요건에 기초하여 필요한 모델 유형을 식별하고(12개 모델 및 3개 사양) 다른 모델링 목적(3개 모델 카테고리)에 따라 분류한다. 모델 유형은 ADOxx Metaomodeling Platform을 이용하여 변형되며, SIMVASoS Modeler라고 불리는 도구에서 모델링 언어로 구현된다. 제안된 모델링 도구를 사용하여 시뮬레이션과 검증 툴의 입력으로 활용할 수 있는 두 가지 다른 SoS 사례와 시나리오를 설계했다. 사례 연구를 통해 MBSoSE에 대한 모델링 방법의 전반적인 적용 가능성을 평가하고 구체적인 모델링 결과를 기준 참조 모델로 제공한다.

[0030] 아래에서 시스템 오브 시스템즈 시각적 모델링 방법 및 장치에 대해 보다 상세히 설명한다.

[0032] 본 발명의 실시예들은 SoS 특성으로 인한 규모와 복잡성 문제를 해결하기 위해 모델 기반 SoS 엔지니어링(MBSoSE) 접근법에 초점을 맞추고 있다. 다양한 MBSoSE 접근법 중에서 본 발명의 실시예들은 SoSE의 분석 및 설계 단계에서 엔지니어링 문제를 다루기 위해 구상되는 SoS의 모델 기반 분석 및 설계를 위한 모델링 방법을 제안한다. 이 모델링 방법은 방법론학자 및 모델링 엔지니어(즉, 모델러, 시스템 설계자)에게 이중 목적을 제공할 수 있다. 도메인 방법론자는 이 방법을 기본 방법으로 확장함으로써 메타 레벨에서 도메인별 설계 방법을 개발할 수 있으며, SoS를 설계하는 실무자는 방법론자들이 개발한 모델링 도구에 대해 모델링 활동을 수행할 수 있다. 모델링 방법은 툴 SIMVA-SoS Modeler(SoS의 시뮬레이션 기반 검증 및 분석을 위한 모델링 툴)로 개발되었으며, 이 툴은 다양한 SoS 영역에 대한 범용 모델링 기법과 절차를 지원한다.

[0033] 본 발명의 실시예들에 따른 방법은 기술적인 접근법으로 개발되었을 뿐만 아니라, SoS 아키텍처와 SoS 메타 모델을 준수하는 오픈 소스 모델링 툴로도 구현된다. 본 발명의 주요 기여는 다음과 같이 요약할 수 있다.

[0034] - 본 발명은 모델 기반 SoS 엔지니어링(MBSoSE)에 특화된 범용 모델링 방법을 제안한다. 이 방법으로 개발된 12개 모델과 3개 사양을 확장함으로써 도메인별 SoS 모델링 방법을 개발할 수 있는 확장을 지원한다.

[0035] - 본 발명의 모델링 방법은 SIMVA-SoS Modeler라고 불리는 모델링 도구로서 구현된다. 이 도구는 일관되고 체계적인 방법으로 시뮬레이션 모델과 시뮬레이션 및 검증 도구의 사양에 대한 입력을 생성하는 수단으로 사용될 수 있다.

[0036] - 개방형 모델 실험실(Open Model Laboratory, OMiLab) 커뮤니티가 지원하는 확장 가능한 라이브러리를 갖춘 오픈소싱 모델링 툴킷으로서 모델링 방법을 구현했다. 따라서 방법론자는 이 라이브러리를 활용하여 자신만의 도메인별 모델링 언어(Domain-Specific Modeling Languages, DSML)나 방법을 체계적으로 개발할 수 있다.

[0038] 여기에서는 SoS 엔지니어링에 활용할 수 있는 모델 기반 엔지니어링 접근방식을 조사한다. Nielsen은 모델 기반 SoS 엔지니어링(MBSoSE) 연구에 대한 체계적인 문헌 검토를 실시하고, SoS 엔지니어링 문제와 해당 접근법을 배치하기 위한 8차원의 관점에서 접근법을 조사했다. 이러한 종합적인 분석을 바탕으로, 대표적인 모델링 방법 및 분석 방법을 제공할 수 있다.

[0039] MBSoSE를 위한 모델링 방법 및 아키텍처 프레임워크

[0040] UML(Unified Modeling Language)은 소프트웨어 엔지니어링 분야에서 표준화된 범용 모델링 언어이다. 정보시스템 개발 과정에서 생성된 아티팩트(artifact)의 설계, 사양, 문서화에 사용된다. UML에서 확장된 SysML(시스템 모델링 언어)은 시스템 엔지니어링 애플리케이션을 위한 객체 지향의 범용 아키텍처 모델링 언어이다. SysML은 사양, 설계 및 분석 이외에도 시스템 속성의 확인과 검증을 가능하게 한다. 이러한 목적을 위해 파라메트릭 다이어그램과 같은 추가 도표를 SysML에 도입한다. UML과 SysML은 모두 MDD(모델 구동 개발)를 위한 활성화 기술로 간주된다.

[0041] 전통적 시스템에 대한 기존 아키텍처는 긴밀하게 결합된 시스템 구성요소를 가정하여 확립되며, 정적 또는 동적 아키텍처의 사양(그러나 설계 시 알려진 예상 재구성에 국한됨)에만 대처하는 공식적인 기초를 가지고 있다. 이는 SoS 엔지니어가 진화적 개발 등 고유한 SoS 특성을 다룰 수 있는 모델과 툴이 부족한 데 큰 기여를 하고 있다. SoS는 돌발 행동 때문에 예측할 수 없이 역동적일 수 있다. 또 다른 도전과제는 SoS의 크기와 복잡성에 기인한다. SoS와 그 행동을 완전히 표현하기 위해서는 모델링 언어를 사용한 기존의 산업 툴과 실행이 강화되어야 한다. 이는 통합 중에 차선의 설계와 값비싼 재작업으로 이어진다.

[0042] 다음의 3가지 아키텍처와 모델링 프레임워크는 SoS와 높은 관련성이 있거나 SoS 고유의 관심사를 다루기 위해

직접 개발되었다. 기업 아키텍처인 TOGAF는 비즈니스, 데이터, 애플리케이션 및 기술의 네 가지 관점을 가지고 있다. 기업 요구사항에 기반한 기업 IT 시스템의 라이프 사이클 전체에 걸쳐 아키텍처와 관리 방법을 제공한다. 시스템 모델링 프레임워크인 DoDAF는 기능, 데이터, 운영, 프로젝트, 서비스, 표준 및 시스템과 같은 몇 가지 관점을 가지고 있다. 시스템, 미션, 이해관계자, 부서의 경계를 넘어 조직적이고 일관된 정보 공유를 통해 효과적인 의사결정을 촉진하는 데 활용할 수 있다. AMADEOS 프레임워크는 개념, 논리적 및 구현 수준 모두에서 SoS 모델링을 지원한다. 모델은 구조, 동적성, 진화성, 신뢰성 및 보안성, 시간, 다중임계성, 출현 등 다양한 관점으로 분석된다.

[0043] 모델 기반 SoS 시뮬레이션 및 검증

[0044] 이전 연구 중 하나는 확률론적 모델을 사용하여 다양한 유형의 SoS를 모델링했으며, 통계적 모델 검색 프로그램을 사용하여 이를 검증했다. 연구는 확률론적 모델이 블랙박스 요소의 비결정론적 행동을 나타내기 위해 필수적이라는 것을 발견했다. 또한 기존 모델 점검 방식은 복잡한 일련의 행동을 점검하는 데 어려움이 있어 통계 모델 점검(SMC)이 실제 SoS 사례에 보다 실현가능하고 실용적인 것으로 나타났다. 보다 최근에, S. Park은 SIMVA-SoS라고 불리는 SoS에 대한 시뮬레이션 기반의 검증과 분석을 위한 툴을 개발했다. 본 발명의 실시예들은 순차 확률비 테스트(SPRT)에 기반한 SMC 기법을 개발했으며, 툴은 입력으로 확률론적 시뮬레이션 모델과 확률론적 시나리오를 지원한다.

[0046] 도 1은 일 실시예에 따른 모델링 방법 및 아키텍처의 구성 요소를 나타내는 도면이다.

[0047] 모델링 방법을 새로 설계하기 위해서는 방법론자가 어떤 요소를 개발해야 하는지 알아야 모델링 기능을 충분히 활용할 수 있다. 도 1에 도시된 바와 같이, 이해관계자는 시스템에 대한 시스템과 관리/엔지니어링에 대한 우려에 관심이 있다. 더 나은 엔지니어링과 관리를 위해, 아키텍처와 아키텍처 설명을 기술함으로써 처리되는 하나 이상의 관점에 의거하여 관심사가 틀이 잡힌다.

[0048] 특정한 관점에 기초하여 모델링 언어와 모델링 절차로 구성된 모델링 기법을 개발함으로써 모델링 방법을 개발할 수 있다. 모델링 언어는 표기법, 구문(문법), 의미론의 세 가지 필수 구조를 포함한다. 표기법은 모델링 언어의 다양한 개념의 표현과 관련이 있다. 모델 기반 시스템/소프트웨어 엔지니어링 접근법에서는 대부분의 경우 그래픽 기호(즉, 아이콘)를 시각적 표현을 목적으로 사용한다. 구문(및 그 구문법)은 표기(모델링 언어의 상징)가 유효한 언어 형태에 맞게 올바르게 구성되었는지 여부를 정의하고, 모델링 구성요소를 어떻게 조합하는지를 결정한다. 반면에 의미론은 전체 모델, 구성요소 및 관계와 연관되거나 의도된 의미에 관한 것이다. 정의한 표기, 구문 및 의미론에 기초하여 의미 매핑은 모델링 언어의 구문에 포함된 의미를 의미 영역과 연관시킨다. 모델러 관점에서 모델링 방법은 모델링 엔지니어(즉, 설계자)가 의도한 목표에 모델링 언어의 기능을 최적으로 활용할 수 있도록 효과적인 모델링 절차를 제공해야 한다. 또한, 모델링 방법은 시뮬레이션, 시각화, 변환, 평가 등 다양한 목적을 위한 메커니즘과 알고리즘을 제공한다.

[0049] MBSoSE에 대한 모델링 방법을 개발하기 위해 그라운드 모델 두 개를 사용한다. 즉, (1) 상위 레벨 SoS 아키텍처와 (2) SoS 메타 모델이다.

[0051] 상위 레벨 SoS 아키텍처(High-level SoS Architecture)

[0052] ISO/IEC/IEEE 42010에 따르면, 아키텍처 설명은 시스템의 아키텍처를 표현해야 하며, 그 설명은 이해관계자와 이해관계자의 관심사를 식별하는 데 사용된다. 다중 아키텍처 관점을 도출하고 이를 유지함으로써 일관성을 확보할 수 있고 아키텍처 근거를 확립할 수 있다.

[0053] 일반적인 소프트웨어 아키텍처와 비교하여, SoS의 아키텍처를 대표하는 다른 과제가 있다. 첫째, SoS는 매우 불확실하고 결정적이지 않은 복잡한 사이버 부분과 물리적인 부분 모두를 포함한다. 또한, SoS 엔지니어들은 기술적인 부분뿐만 아니라 사회적 또는 사회 기술적 측면도 고려해야 하기 때문에 인적 요인은 아키텍처를 더욱 어렵게 만든다. 대부분의 SoS는 두 번째로 고정된 정적 시스템 경계를 가지고 있지 않다. 경계는 엔지니어링 관심사, 런타임 상황 또는 SoS의 진화적 특성에 따라 동적으로 정의할 수 있다. 이 특성은 적응력을 결정하는데, SoS는 내부 및 외부 상황의 변화에 대응하여 스스로 조정할 수 있다. 셋째, 모든 구성 요소와 구성요소 성분이 서로 어떻게 상호 작용하는지 어느 정도이며, SoS 엔지니어들은 공유된 목표나 임무를 수행하기 위해 이들을 조정해야 한다. 구성 요소들을 효과적으로 통합하기 위해서, 구조적인 설명을 사용하여 전체적인 관점이 제

공되어야 한다. 이러한 이유로, 상위 레벨의 추상적 아키텍처를 정의하는 것은 하위 레벨의 특정 접근방식(예: 도메인별, 기술/기술 특정 접근법)을 사용하는 것보다 더 실현 가능하고 효과적인 해결책이 될 수 있다.

[0054] 도 2는 일 실시예에 따른 레벨 및 레이어를 개념화하는 상위 레벨 SoS 아키텍처를 나타내는 도면이다.

[0055] 다양한 SoS에 대해 범용적 아키텍처 설명을 제공하기 위해, 아키텍처는 도 2에 도시된 바와 같이 추상적 레벨, 계층, 추상적 인터페이스를 포함함으로써 종합적인 뷰를 제공한다. 레벨은 관리 및 운영 상황을 고려하며, 적절한 전략에 의해 실현된다. 각 레벨에 대해, 관련 SoS 이해관계자 및 그 엔지니어링 관심사에 따라 1개 이상의 계층이 정의되며, 이 계층은 서로 다른 모델 유형에 의해 분석 및 설계된다. 각 모델 유형에는 특정 엔지니어링 규율이 있으므로, 단일 레벨이 여러 가지 규율을 구성할 수 있다는 것을 의미한다.

[0056] 실시예에 따른 아키텍처에서는, 각각 교차 컨텍스트와 cross disciplinary 두 가지 유형의 추상 인터페이스가 식별된다. 교차 컨텍스트는 서로 다른 레벨 사이의 인터페이스로 정의될 수 있으며, 다른 계층들 사이의 인터페이스에서 cross disciplinary 측면을 고려할 수 있다. SoS의 "상단 개방" 및 "하단 개방" 특성을 반영하기 위해, 아키텍처는 특정 최상위 애플리케이션이나 고정된 하위 레벨 요소를 포함하지 않는다.

[0058] SoS 엔지니어링을 위한 메타 모델(Meta-Model for SoS Engineering)

[0059] 두 번째 기본 모델은 M2SoS라는 이름을 가진 메타 모델이며, M2SoS는 시스템 오브 시스템즈(엔지니어링)용 메타 모델을 의미한다.

[0060] 도 3은 일 실시예에 따른 SoS 엔지니어링용 메타 모델(M2SoS)을 단순화하여 나타내는 도면이다.

[0061] 도 3에 도시된 바와 같이, M2SoS는 SoS 아키텍처를 기반으로 SoS에 대한 전체적인 상위 레벨의 뷰를 제공한다. M2SoS를 사용하면 MBSoSE 전체 단계에서 SoS 엔지니어와 관리자 모두를 안내할 수 있으며, 분석 및 설계 결과의 완전성 또는 정확성(예: 모델, 아티팩트)을 평가할 수도 있다. M2SoS의 또 다른 기능은 SoS 엔지니어가 다양한 SoS 도메인에 대해 온톨로지 분석을 수행할 수 있다는 것이다. 이를 통해 SoS 이해당사자(즉, SoS 레벨 이해당사자와 구성 레벨 이해당사자 모두)가 대상 SoS의 공통 지식 기반을 구축할 수 있다. M2SoS를 기반으로 온톨로지를 개발하여 SoS의 공유 프로젝트 저장소에 저장하면 일반적으로 사용되는 어휘에 접속하여 통신 등 다양한 용도로 활용할 수 있다.

[0062] M2SoS에 기초하여, SoS의 필수 클래스를 설계할 수 있고 클래스 정의에 부합하는 모델링 언어를 개발할 수 있다. 클래스를 정의하는 동안 범용 메타 모델링에는 몇 가지 요구사항이 있었다. 첫째로, 모델링 엔티티(entity)는 소유물과 소속 면에서 구별되어야 하며 SoS 아키텍처에 정의된 레벨과 레이어(계층)을 고려해야 한다. 둘째로, 대부분의 SoS 메타 모델링은 명시적이거나 암묵적으로 상위 레벨의 공통 목표를 가지고 있기 때문에 SoS의 변형은 목표 지향 엔지니어링(예: GORE)을 고려해야 한다. 세 번째 요구사항은 주로 SoS에 내재된 불확실성에 초점을 맞춘다. 불확실성과 예측불가능성의 대부분을 산출하는 환경요인은 1차 엔티티 중 하나로 간주해야 한다.

[0064] 도 4는 일 실시예에 따른 시스템 오브 시스템즈 시각적 모델링 방법을 나타내는 도면이다.

[0065] 도 4를 참조하면, 본 발명의 실시예들은 모델링 방법을 개발하고 그 방법은 툴에 의해 지원된다. 그 방법을 개발하기 위한 전반적인 과정의 예시가 도시되어 있다.

[0066] 일 실시예에 따른 컴퓨터 장치를 통해 구현되는 시스템 오브 시스템즈(System-of-Systems, SoS) 시각적 모델링 방법은, 분석, 설계, 시뮬레이션 및 검증 중 적어도 어느 하나 이상의 관점에서 SoS 모델링 방법을 제공하기 위한 요구사항을 수집하는 단계(410), 수집된 요구사항에 기초하여 적어도 하나 이상의 요구되는 모델 유형을 식별하는 단계(420), 및 식별된 요구되는 모델 유형을 메타 모델링하여 메타 모델 기반 모델 유형을 생성하는 단계(430)를 포함하여 이루어질 수 있다.

[0067] 또한, 메타 모델 기반 모델 유형은 모델링 툴(modeling tool)에서 구현되는 단계(440)를 더 포함할 수 있다.

[0068] 또한, 모델링 툴을 배포 또는 공개하는 단계(450)를 더 포함할 수 있다.

[0069] 일 실시예에 따른 시스템 오브 시스템즈 시각적 모델링 방법은 시스템 오브 시스템즈 시각적 모델링 장치를 통해 수행될 수 있다.

[0071] 도 5는 일 실시예에 따른 시스템 오브 시스템즈 시각적 모델링 장치를 나타내는 블록도이다.

[0072] 도 5를 참조하면, 일 실시예에 따른 시스템 오브 시스템즈 시각적 모델링 장치(500)는 요구사항 수집부(510), 모델 유형 식별부(520) 및 메타 모델링부(530)를 포함하여 이루어질 수 있다. 실시예에 따라 시스템 오브 시스템즈 시각적 모델링 장치(500)는 모델링 툴(540) 및 모델링 툴 배포부(550)를 더 포함하여 이루어질 수 있다.

[0073] 아래에서는 시스템 오브 시스템즈 시각적 모델링 장치(500)를 예를 들어 시스템 오브 시스템즈 시각적 모델링 방법의 각 단계를 설명한다.

[0075] 단계(410)에서, 요구사항 수집부(510)는 분석, 설계, 시뮬레이션 및 검증 중 적어도 어느 하나 이상의 관점에서 SoS 모델링 방법을 제공하기 위한 요구사항을 수집할 수 있다. 보다 구체적으로, 요구사항 수집부(510)는 분석, 설계, 시뮬레이션 및 검증의 관점에서 SoS 모델링 방법을 개발하기 위한 요구사항을 명시할 수 있다. 이 단계는 각 엔지니어링 단계에 대해 생성된 모델에서 충족해야 하는 요구사항을 포괄적으로 분석한다. 이 때, 요구사항 수집부(510)는 생성된 모델이 충족되어야 하는 요구사항의 집합을 수집하여 출력할 수 있으며, 이 요건은 이후에 방법을 검증하는 데 사용될 것이다.

[0076] 단계(420)에서, 모델 유형 식별부(520)는 수집된 요구사항에 기초하여 적어도 하나 이상의 요구되는 모델 유형을 식별할 수 있다. 모델 유형 식별부(520)는 요구되는 모델 유형의 체계적인 식별을 위해 SoS 메타 모델을 입력으로 사용하여 상위 레벨 SoS 아키텍처의 레벨 및 레이어(layer)를 정의할 수 있다. 또한, 모델 유형 식별부(520)는 요구되는 모델 유형은 소프트웨어와 하드웨어 구성요소, 에이전트, 서비스, 인적, 조직 요소, 데이터 중 적어도 어느 하나 이상을 포함하는 SoS 엔티티를 다루고, 요구사항을 충족하기 위해 복수 개의 모델 유형으로 분류될 수 있다.

[0077] 예를 들어, 모델 유형 식별부(520)는 체계적 식별을 위해 상위 레벨의 SoS 아키텍처와 SoS 메타 모델을 입력으로 활용할 수 있다. 또한 모델 유형은 일반적으로 SoS 엔티티를 다루기 위해 세 가지 모델 유형으로 분류되며 요구사항을 충족한다.

[0078] 단계(430)에서, 메타 모델링부(530)는 식별된 요구되는 모델 유형을 메타 모델링하여 메타 모델 기반 모델 유형을 생성할 수 있다. 이 때, 메타 모델링부(530)는 분류에 따라 복수 개의 모델 유형의 메타 모델을 생성할 수 있다.

[0079] 이와 같이, 메타 모델링부(530)는 실제로 이전 단계에서 식별된 모든 모델 유형을 메타 모델링하여 모델링 방법을 개발한다. 이 단계는 단계(420)의 분류에 따라 여러 가지 모델 유형(즉, 메타 모델)을 생성하며, 모델 유형은 SIMVA-SoS Modeler라고 하는 툴로서 단계(440)의 모델링 툴에 구현된다. 앞에서 설명한 바와 같이, 도구는 다중 모델링 언어(다이어그램)와 모델링 기법을 지원하는 절차를 포함한다.

[0080] 단계(440)에서, 모델링 툴(540)은 메타 모델 기반 모델 유형은 모델링 툴(modeling tool)에서 구현될 수 있다. 모델링 툴(540)에서, 메타 모델 기반 모델 유형은 SIMVA-SoS Modeler라는 모델링 툴에서 모델링 언어로 구현될 수 있다.

[0081] 단계(450)에서, 모델링 툴 배포부(550)는 모델링 툴을 배포 또는 공개할 수 있다. 보다 구체적으로, 모델링 툴 배포부(550)는 다른 도메인 방법론자(domain methodologist)에 의해 확장될 수 있는 도구 라이브러리(tool library)로 개방형 모델링 커뮤니티(open-modeling community)에 배포 또는 공개될 수 있다.

[0083] 아래에서는 모델링 방법 개발을 위한 요구사항에 대해 설명한다.

[0084] SoS 분석 및 설계 요구사항

[0085] a) SoS 엔티티 및 경계 식별: 아키텍처(및 아키텍처 설명)를 사용하는 주요 목적 중 하나는 시스템 엔티티와 기업들 간의 관계를 식별하여 특정 관점을 설정하는 것이다. SoS의 엔티티는 소프트웨어와 하드웨어 구성요소, 에이전트, 서비스, 인적 및 조직 요소, 데이터 등을 포함한다. 또한 SoS의 경계는 효과적인 MBSoSE를 위해 어떻게든 결정되어야 한다. 그러나 SoS의 경계는 정적이지 않지만 모호하고 유동적이며 협상할 수 있다. 또한,

단일 SoS의 경계는 시간적, 지리적(공간적), 개념을 가지며 엔지니어링 또는 목표 달성의 수행 전체에 걸쳐 변화할 수 있다. 경계가 모호할 수 있지만, 실시예들에 따른 모델링 방법은 모델 유형에 유한한 클래스 집합을 제공함으로써 SoS에 포함되고 제외되는 엔티티에 대한 기준을 제공할 수 있어야 한다.

[0086] b) 목표 지향적 분석 및 설계 지원: 요구사항 엔지니어링의 한 분야로서, 기존의 시스템/소프트웨어 엔지니어링은 우려와 목표를 획득하고, 그에 상응하는 요구사항을 도출하며, 대안 및 갈등을 포착하기 위해 목표 지향 요구사항 엔지니어링(GORE)을 수행했다. SoS는 암묵적이든 명시적이든 공통된 목표를 공유해야 하며, SoSE의 가장 중요한 목표는 목표를 달성하기 위해 새로운 행동을 추진하는 것이다. 따라서 목표 기반 분석과 설계는 적절한 모델링 방법에 의해 뒷받침되어야 하며, 그 방법은 잘 수용되고 확립되어야 한다. SoS의 공통 목표에 기초하여 엔지니어는 전체적인 엔지니어링 접근방식을 선택할 수 있다. 즉, 하향식 또는 상향식일 수 있다. 기존의 획일적 시스템이나 소프트웨어와 달리 SoS는 상위 레벨 시스템(즉, SoS)에 속하는 것을 고려하지 않고 이미 설계한 레거시(legacy) 시스템으로 구성될 수 있기 때문에 경우에 따라 상향식 접근방식이 불가피할 수 있다. 상향식 접근방식은 성분들의 능력 기반 분석을 통해 실현될 수 있으므로, 다중 성분의 집합적 능력 분석은 실시예들에 따른 모델링 방법에 의해 뒷받침되어야 한다.

[0087] c) 모델 간 또는 모델링된 객체 간 추적성 지원: 모델링 도구를 사용하는 주된 이유 중 하나는 만들어진 모델의 완전성과 그들 사이의 추적성을 보장하기 위함이다. 모델 추적성은 엔지니어와 이해관계자가 아티팩트 및 엔티티 간의 연관성과 의존성을 이해하는 데 도움이 된다. 특히, SoS는 서로 상호 연결되고 상호의존적인 훨씬 더 많은 엔티티와 아티팩트를 가지고 있으며, 추적가능성의 지원은 복잡성 관리와 통신에 대한 다른 어떤 문제보다 더 중요하다.

[0088] d) 도메인별 확장 지원: 실시예들에 따른 목표는 실제 도메인 엔지니어 및 이해관계자에 의해 도메인별 SoS 모델링 방법으로 확장될 수 있는 범용(즉, 도메인-일반) 모델링 방법과 도구를 개발하는 것이다. 하나의 범용 모델링 방법으로는 모든 도메인별 시스템을 다룰 수 없기 때문에, 실시예들에 따른 모델링 방법은 도메인별 확장을 적절히 고려해야 한다. 모든 클래스가 모든 도메인에 대해 사용(및 인스턴스화)되지 않더라도, 실시예들에 따른 모델링 방법의 모델 유형은 도메인별 엔지니어들에 의해 확장될 수 있어야 한다.

[0089] e) 온톨로지(존재론적) 분석 지원: 온톨로지는 지식의 공식적이고 노골적인 표현으로, 그 범주, 속성 및 다른 개념과의 관계를 가진 개념을 설명한다. 온톨로지 구축은 공통의 어휘와 공통의 이해가 확립될 수 있기 때문에 상당한 가치가 있을 수 있다. 이를 통해 시스템 이해당사자들 간의 말단적/개념적 불일치를 줄임으로써 시스템 이해당사자들 간의 정밀한 커뮤니케이션을 가능하게 한다. 이외에도 온톨로지는 구조화된 용어와 분류법으로 시스템의 복잡성을 포착하기 위해 사용되며, 이를 위한 방법을 온톨로지 분석이라고 한다. 분석된 결과는 시스템 설계 및 개발에 대한 결정을 뒷받침할 수 있다. 앞에서 SoS 엔지니어의 온톨로지 분석을 지원하기 위해 M2SoS가 개발되었다고 소개했다.

[0091] SoS 시뮬레이션 및 검증 요구사항

[0092] a) 객체 지향 및 에이전트 지향 구현에 대한 고려사항: 객체지향 프로그래밍(OOP)은 컴퓨터 시스템이 서로 통신할 수 있는 객체로 구성되는 패러다임이다. OOP의 목적과 유사하게 에이전트 지향 프로그래밍(AOP)에서 에이전트는 상위 시스템의 구성요소다. OOP와 AOP 패러다임을 기반으로 MBSoSE에 특화된 설계 및 프로그래밍 패러다임도 기존 시스템 개발 패러다임의 연장이 될 수 있다. 에이전트 기반 시스템(예: 다중 에이전트 시스템(MAS))의 에이전트는 SoS의 구성 요소와 많은 유사점을 공유한다. 두 시스템 모두 에이전트와 구성요소라 불리는 자율적 요소 시스템의 집합적 능력에 의해 달성되는 더 상위 레벨의 공통 목표를 가지고 있다. 그러나 구성 시스템은 일반 객체나 에이전트로부터 훨씬 더 독립적일 수 있으며 SoS 엔지니어는 구성 요소에 접근하거나 제어할 수 있는 완전한 권한을 갖지 못할 수 있다. 이러한 이유로, MBSoSE의 모델링 방법은 OOP와 AOP로부터 기본 개념과 아이디어를 빌려서 SoS의 구성요소에 더 적합하도록 확장해야 한다.

[0093] b) 시간적 및 지리적 속성 표현: 보다 현실적인 시뮬레이션 기반 분석을 위해, 실시예들에 따른 모델링 방법에 의해 생성된 시뮬레이션 모델은 시간적 및 지리적인 정보를 나타낼 수 있어야 한다. 시간적 정보는 주로 시간, 행동 지속시간 및 사건의 시간적 특성을 포함한다. 지리적 정보에는 논리적으로 설계된 지도에 물체를 위치시키는 모든 시뮬레이션 객체의 지리적 위치가 포함된다. 시간적, 지리적 특성의 시뮬레이션 방법은 각 시뮬레이션 엔진의 메커니즘에 따라 다를 수 있다. 다양한 시뮬레이션 모델 중에서 실시예들에 따른 모델링 방법은 시뮬레이션을 이산 이벤트/시간 시뮬레이션(DES)으로 간주하며, 이 시뮬레이션은 시스템의 동작을 로직 시간에 이

산 이벤트 시퀀스로 모델링한다.

[0094] c) 불확실성의 표현: 모델 기반 시뮬레이션은 주어진 모델에 대해 가능한 일련의 행동을 관찰하기 위해 수행되며, 관찰은 실제 문제를 안전하게 사전에 해결하는 데 사용된다. 실제 문제를 일으킬 가능성이 있는 가장 중요한 것은 내재적이고 산재된 불확실성으로, SoS 목표를 달성하는 위험일 수도 있다.

[0095] d) 동적 재구성 지원: SoS와 그 구성원들은 작업을 수행하는 동안 그들의 행동과 구조를 동적으로 변화시킨다. SoS의 이러한 특성을 동적 재구성 또는 (자체)적응이라고 하며, 모델 기반의 SoS 엔지니어링 방법을 배치하고 개발하는 것이 주요한 차원 중 하나이다. 이 기능은 SoS의 동작, 구조(연결) 및 구성에 대한 내부 및 외부 변경을 모두 수행하는 것이다. 재구성 기능을 실현하고 시뮬레이션하기 위해 입력 모델은 시뮬레이션 중에 재구성할 수 있어야 한다. 재구성 가능한 모델은 확률론적 행동과 구조 표현을 허용함으로써 개발될 수 있다. 또한, 규칙(또는 정책)을 모델링하여 시뮬레이션 하에서 모델의 런타임 동작과 구조를 동적으로 조작할 수 있다.

[0096] e) 돌발 행동을 포착하고 관찰하는 것: 행동의 출현이란 단일 구성원이 전달할 수 없는 시너지 행동을 말하며, 그 출현을 포착하는 것이 시뮬레이션 기반 분석의 일차적 원인이다. 시뮬레이션 모델이 돌발 행동을 수행할 수 있도록 입력 구성 모델의 자율성을 적절하게 표현하고 시뮬레이션 할 필요가 있다. 구성원의 자율성에 따라 시뮬레이션 엔진은 모델 전체가 스스로 동작하도록 하여 출현을 나타내도록 해야 한다. 시뮬레이션 중에는 목표 달성, 과제 달성 또는 사고/실패 발생이 될 수 있는 긴급 행동을 관찰할 수 있다. 이들을 포착하고 분석하기 위해, 모델링 방법은 시나리오와 속성의 사양을 지원하고, 시뮬레이션을 안내하고, 출현을 발견/조사해야 한다.

[0098] 아래에서는 다면적 모델 유형을 정의한다.

[0099] 모델 유형의 분류

[0100] UML, SysML과 같은 많은 범용 소프트웨어/시스템 모델링 방법에서 모델 유형은 일반적으로 (a) 구조 모델, (b) 행동 모델, (c) 상호작용 모델의 세 가지 모델로 분류된다. 그러나 SoS 엔지니어링의 경우 상당수의 내부 정보를 완전히 사용할 수 없다. 뿐만 아니라, 조직, 구성자, 인프라 및 이들의 컨텍스트와 환경은 서로 더 복잡한 상호연계를 가질 수 있다.

[0101] 표 1은 모델 유형의 분류의 예시를 나타낸다.

[0102] [표 1]

Category	Included Model Types
Architectural Model Types (AMT)	SoS Integration Model (SoSIM) SoS Organization Model (SoSOrgM) SoS Infrastructure Model (SoSInfraM) SoS Environment Model (SoSEnvM) SoS Map Model (SoSMapM)
Reference Model Types (RMT)	
RMT-Goal-based/-oriented Model Types (RMT-GMT)	Goal Decomposition Model (GDM) Requirement Specification Model (RQSM) Rule Specification Model (RLSM) Risk Analysis Model (RAM)
RMT-System Model Types (RMT-SMT)	System Capability Model (SCM) Service Description Model (SDM) Interface Description Model (IDM)
RMT-Operational Model Types (RMT-OMT)	Task Process Model (TPM) State Machine Model (SMM)
RMT-Domain Model Types (RMT-DMT)	Environmental Factor Model (EFM)
Verification & Validation Model Types (VVMT)	Simulation Specification (SimSPEC) Simulation Scenario Specification (SimScnSPEC) Verification Specification (VerifSPEC) Verification Property Specification (VerifPptySPEC)

[0103]

[0104] 도 6은 일 실시예에 따른 모델 유형의 분류를 설명하기 위한 도면이다.

[0105] 도 6에 도시된 바와 같이, 실시예들에 따른 모델링 방법은 모델 유형(model type, MT)(640)을 세 가지 범주로 분류한다. 다시 말하면, 모델 유형(640)은 아키텍처 MT(610), 참조 MT(620), 사양 MT(630)의 세 가지 범주로 분류되며, 모델링 언어(650)로 표현될 수 있다. 여기에서는 공간 부족으로 인해 각 다이어그램의 정의와 메타 모델에 대한 상세한 설명은 생략하고 웹(SIMVA-SoS Modeler, 2020 (accessed July 24, 2020), https://sites.google.com/view/sos-modeler.)에 업로드하였다.

[0106] 아키텍처 모델 유형

[0107] 아키텍처 모델 유형은 SoS의 아키텍처(즉, 구조) 관점을 나타낸다. 이 모델 유형의 주요 목적은 모델 기반 방식으로 구성 요소와 구성요소 엔티티를 SoS로 통합하고 조정하는 것이다. SoS 구조를 설명하기 위해 조직, 인프라, 환경을 정의하고, SoS 통합 모델을 구성하여 SoS 경계를 결정한다.

[0108] SoS의 아키텍처 측면을 나타내는 SoS 통합 모델, SoS 조직 모델, SoS 인프라구조 모델, SoS 환경 모델의 4가지 주요 모델이 있다. 전체적인 상호접속은 기술되고 구성요소는 SoS 통합 모델에 의해 통합되며, 다른 아키텍처 모델(조직, 인프라구조 및 환경)에 대한 다중 상호 참조를 가지고 있다.

[0109] 참조 모델 유형

[0110] 참조 모델 유형은 기본적으로 SoS의 종합적이고 보충적인 정보를 모델링하기 위해 아키텍처 모델 유형에 의해 참조된다. 참조 모델 유형을 확장함으로써 다중 및 교차 도메인 SoS 엔지니어가 도메인 특정 모델링을 협업적으로 수행할 수 있다.

[0111] a) 시스템 참조 모델 유형: 이 모델 유형은 SoS 레벨의 목표 달성을 위해 활용/전개되는 시스템과 그 능력(즉, 시스템의 능력 및 용량)을 기술하는 모델을 정의하는 데 사용된다. 이를 시스템 기능 모델이라 표현할 수 있다.

[0112] b) 동작 기준 모델 유형: 이 모델 유형은 업무(및 비즈니스) 프로세스, 행동, 행동 및 운영과 같은 SoS 실체의

행동 측면을 나타내는 모델을 정의하는 데 사용된다. SoS의 행동은 두 가지 카테고리로 분류될 수 있다. 하나는 (조직 수준) 과제 프로세스 모델로, 여러 구성원에 의한 집단 행동(즉, 협업, 협력)이고, 다른 하나는 (개인 수준) 운영 모델로, 단일 구성원에 의해 수행되는 개별 행동이다. 집단 행동은 업무 프로세스 모델에서 설명되며, 개별행동은 SoS 엔지니어가 정보를 얻을 수 있는 경우에만 운영 모델을 사용하여 모델링한다.

[0113] c) 목표(기반) 참조 모델 유형: 이 모델 유형은 SoS 레벨 공통 목표 및 목표와 관련된 모델을 정의하는 데 사용된다. SoS 목표를 분해하고 개선함으로써 요구사항, 규칙, 정책 및 기타 요소를 도출하고 체계적으로 지정할 수 있다. 이는 목표 분해 모형, 요구사항 사양 모델, 규칙/정책 사양 모델, 및 결함/고장/위험 모델로 표현될 수 있다.

[0114] d) 도메인 참조 모델 유형: 이 모델 유형은 환경 요인, 영역 및 컨텍스트 정보를 기술하는 모델을 정의하는 데 사용된다. 이는 환경 요인 모델로 표현될 수 있다.

[0115] 사양 모델 유형

[0116] 사양 모델 유형은 실행시간에 대한 입력 아티팩트를 지정하는데, 주로 시뮬레이션, 시험, 검증(모델확인) 등 V&V 단계에 초점을 맞춘다. 이러한 규격 모델 유형은 SIMVA-SoS라고 하는 시뮬레이션 기반 통계 모델 점검 도구의 입력을 생산하기 위해 설계된다. SIMVA-SoS를 사용한 분석의 경우, 의도된 시뮬레이션 실행을 위해 시뮬레이션 시나리오 사양이 필요하며, 시뮬레이션 모델의 (통계적) 모델 확인을 위해 검증 속성 사양이 필요하다. 다음은 가능한 사양 모델 유형이다. 이는 시뮬레이션 시나리오 사양, 검증 속성 사양, 및 테스트 사양으로 표현될 수 있다.

[0118] 아래에서는 모델링 툴 개발에 대해 설명한다.

[0119] 모델링 방법의 구현

[0120] a) ADOxx 메타 모델링 프레임워크 및 개방형 모델링 실험실(OMiLab): ADOxx 메타 모델링 플랫폼은 모델링 방법(또는 방법론)의 전체적인 개발을 제공하고 지원하며, 이는 모델링 도구를 자동으로 구축함으로써 효율적인 개발을 가능하게 한다. ADOxx 플랫폼은 주로 메타 모델링 접근방식을 지원하며, 여기에서는 클래스와 그 속성 및 운영을 정의함으로써 모델링 언어를 쉽게 개발할 수 있다.

[0121] ADOxx 프레임워크에서는 출력용으로 ABL(ADOxx 라이브러리 언어)과 ADL(ADOxx 모델 언어) 두 가지 유형의 언어 형식이 지원된다. ADL은 모델링된 결과(예: 다이어그램, 객체, 값 등)를 기술하고 저장하는 언어여서 ADL 파일은 주로 모델의 보내기/받기에 사용된다. 반면 ABL은 모델링 방법의 모든 메타 레벨 정보를 저장하는 데 사용되는 라이브러리이다. 이 정보에는 모델 유형(언어), 메커니즘, 알고리즘 및 기타 툴 지원 기능의 정의가 포함된다.

[0122] 실시예들에 따른 모델링 툴은 시스템 엔지니어링 방법의 시스템 고유의 문제를 해결하고 구현하기 위해 ADOxx 메타 모델링 플랫폼에서 개발되었다. 정의된 모델 유형의 그래픽 표현은 ADOxx 라이브러리에서 기본 구성요소를 상속하여 ADOxx 개발 툴에 의해 설계된다. ADOxx는 원래 비엔나 대학의 OMILAB에 의해 개발되어 발매되었다. 그것은 모델링 방법을 모델링하는 가장 혁신적인 메타 모델링 도구 중 하나로 알려져 있다. 총 42개의 개방형 모델이 ADOxx에서 개발되어 비영리 애플리케이션에 공개된다. 오픈 모델 계획의 아이디어는 기업 참조 모델을 협업적으로 개발할 수 있도록 하고, 누구나 개방적이고 공개적인 프로세스로 복사, 사용, 수정 및 (재배포)할 수 있는 범위 내에서 개발된 소프트웨어 제품의 공유를 촉진하는 것이다.

[0123] b) SIMVA-SoS Modeler: SIMVA-SoS는 주로 분석, 설계, 검증 및 검증을 위해 모델 기반 SoS 엔지니어링을 지원하는 SoS에 대한 시뮬레이션 기반 검증 및 분석을 의미한다. SIMVA-SoS는 일반적으로 에이전트 기반 시뮬레이션을 여러 번 수행하여 통계적 모델 점검 도구로 사용할 수 있다. SoS를 시뮬레이션하기 위해 SIMVA-SoS는 시뮬레이션 모델, 구성 및 선택적 시나리오가 입력으로 필요하다. SIMVA-SoS Modeler는 이에 의해 SoS 엔지니어가 앞에서 정의한 모델 유형을 활용하여 체계적으로 SoS의 시뮬레이션 모델을 구축하는 데 사용된다.

[0124] SIMVA-SoS Modeler는 SoS 엔지니어가 모델링 작업을 수행할 수 있도록 몇 가지 기능을 지원한다. 현재 버전의 툴은 7가지 주요 모델 유형의 모델링을 지원하며, 사용자(즉, SoS 모델링 엔지니어)는 선택된 접근방식(하향식 또는 상향식)의 모델링 절차를 따라 모델 유형을 선택할 수 있다.

[0125] 도 7은 일 실시예에 따른 SIMVA-SoS Modeler의 사용자 인터페이스의 예시를 나타내는 도면이다.

[0126] 도 7에 도시된 바와 같이, 사용자는 모델링 인터페이스(예: 툴박스, 캔버스, 대화 상자)를 이용하여 그래픽 모델링을 수행할 수 있으며, 툴은 모델링 규칙이나 카디널리티(cardinality)를 위반했는지 여부를 확인한다. 모델이 생성된 후에는 XML 또는 ADL 형식의 파일로 내보낼 수 있으며 외부 모델 파일도 실시예들에 따른 툴로 가져올 수 있다.

[0128] 실시예들은 위에서 설명한 SoS 모델들에 대해 시각적 모델링 환경을 지원한다. 중요한 특징은 위에서 설명한 실시예들 고유의 모델들을 시각적으로 표현하기 위한 이미지를 제공한다는 것이다. 각각의 모델을 구성하는 요소들은 개별 아이콘을 가지고 있으며, 요소들간 관계도 그림으로 표현된다. 구조적 모델 유형의 모델은 상위의 요소가 하위의 요소를 포함하는 관계가 계층적인 이미지로 나타난다. 참조 모델 유형의 모델들은 구조적 모델들에 의해 참조관계를 갖는다. SoS 구조 모델을 더블 클릭하면 해당 모델을 더 깊게 설명할 수 있는 참조 모델로 이동할 수 있으며, 이러한 기능을 통해 복잡한 SoS의 모델을 쉽게 관리할 수 있도록 하고, 모델 간의 관계를 동적으로 보여준다. 모든 모델링 과정은 모델 요소 생성을 위한 아이콘 클릭, 모델 요소 이동을 위한 드래그 앤 드롭, 세부 내용 작성을 위한 키보드 입력으로 진행되며, 이 모든 기능을 통합한 시각적 모델링 환경을 제공한다.

[0129] 실시예들은 도메인 특정적(domain-specific) 모델링 도구로의 확장 기능을 제공한다는 특징을 가지고 있다. 기존의 모델링 도구는 대부분 모델링 엔지니어가 도구의 제한적 사용만 허용한다는 점에서 닫힌(closed) 도구의 성격을 갖는다. 반면, 실시예들은 모델링 엔지니어가 자유롭게 확장할 수 있는 열린(open) 도구이다. 위에서 설명한 모든 SoS 모델들은 메타 모델(meta-model)의 형태로 도구에 내재되어 있다. 메타 모델이란 모델이 따라야 할 문법(syntax)와 의미(semantic)을 명세한 모델이다. 즉, 모델이 올바른 모델이 되기 위한 규칙을 모델링해 놓은 것이라 할 수 있다. 실시예들의 모델 유형은 내재적으로 메타 모델 형태로 제공되어 있기 때문에, 이 메타 모델을 SoS 엔지니어가 확장해서 사용할 수 있으며, 이러한 메타 모델 확장 기능이 제공된다. 따라서 실시예들은 도메인 일반적인(domain-general) 도구이며, 어떠한 SoS의 모델링에도 사용될 수도 있고, 이와 동시에 이를 확장하여 “재난 대응 SoS”, “스마트 도시 SoS” 등과 같은 도메인 특정적인(domain-specific) SoS를 위한 모델링 도구로 확장해서 사용할 수 있다.

[0130] 실시예들은 다양한 SoS 모델링 전반에 사용될 수 있다. SoS는 다중의 시스템으로 구성된 대형 복잡 시스템이며, 이러한 시스템의 예시는 실생활에서 많이 찾을 수 있다. 어떠한 시스템이 독립성(구성 시스템이 독립적임), 연대성(구성 시스템들의 협력이 존재함), 연결성(구성 시스템들이 연결되어 있음), 다형성(구성 시스템의 능력 등이 다양함), 창발성(단일 시스템으로 불가능한 능력이 발현됨)의 성격을 갖는다면 SoS라고 분류할 수 있으며, 그러한 모든 시스템에 실시예들이 활용될 수 있다. 아래에서 실시예들을 사용할 수 있는 SoS 예시를 나타낸다.

[0131] 일례로, 스마트 도시는 에너지 시스템, 교통 관리 시스템, 치안 담당 시스템, 화재 관리 시스템 등 다양한 시스템이 함께 협력하여 도시의 효율적인 자원 배분, 안전 관리 등을 보장하고자 하는 대규모 시스템이다. 각 시스템은 고유의 목적을 가지고 있지만, 스마트 도시의 관리 측면에서 상위의 목표를 위해 서로 협력하고 정보를 공유할 필요가 있다. 이러한 측면에서 스마트 도시는 SoS의 한 예로 볼 수 있으며, 이러한 대규모 시스템을 설계할 때는 체계적인 모델링 도구의 지원이 필요하다. 또한 도구가 지원하는 모델들을 통해 스마트 도시 구축에 참여하는 서로 다른 참여자들이 일관되게 소통할 수 있다.

[0132] 다른 예로, 대규모 화재 혹은 선박 사고와 같이 대형 재난이 발생한 경우에는, 재난 전반을 관리하는 컨트롤 타워, 인명 구조를 담당하는 소방 시스템, 구조된 환자를 치료하고 관리하는 의료 시스템 등이 동적으로 긴밀하게 협력해야 한다. 이 때, 각 시스템들은 서로 다른 개별 목표와 정책, 프로토콜 등이 존재할 수 있다. 이러한 개별 시스템들의 효율적인 협력을 준비하기 위해서는 각 시스템들을 하나의 SoS로 통합하여 설계해두는 것이 필요하며, 이때 실시예들을 사용해 효율적인 통합 재난 대응 관리 시스템을 모델링 할 수 있다.

[0133] 또 다른 예로, 자율 주행 및 주행 보조 시스템들이 고도화됨에 따라, 다양한 자동차들이 협력하여 도로의 교통을 줄이거나, 군집 주행을 통해 사용 에너지 효율을 높이고 환경 오염을 줄이고자 하는 등의 연구들이 진행되고 있다. 이 또한 개별 목표를 가진 다수의 자동차들을 통한 협력이라 볼 수 있고, 따라서 이러한 자율 주행 자동차 시스템들의 집합을 하나의 SoS라 할 수 있다. 실시예들을 통해 이러한 군집 주행 SoS를 모델링하고 분석할 수 있다.

[0134] SoS의 활용분야는 단일 시스템으로 목적을 달성하기 어려운 대규모 시스템으로 무궁하게 확장할 수 있으며, 그

중 SoS의 구축이 어렵지만, SoS의 실패가 치명적인 안전 시스템(원자로, 교통 관리, 스마트 에너지, 스마트 시티, 스마트 팩토리)에 체계적인 SoS 모델링 도구가 활용될 수 있다. 반면, 현재 흔히 사용되고 있는 모델링 도구들은 대부분 단일 시스템을 설계하기 위한 목적이 주되며, SoS 모델링에는 제약이 있다. 따라서 SoS 모델링 도구의 시장이 다양한 시스템 개발 분야에 걸쳐 있으며 확장 가능성이 크다고 볼 수 있다.

[0135] 실시예들은 복잡한 SoS의 효율적인 모델링을 시각적으로 지원하기 때문에 SoS 설계에 소모되는 비용을 크게 줄일 수 있다. SoS 설계에 소모되는 비용을 크게 요구사항 명세 비용, 디자인 비용, 디자인 검증 비용으로 나눌 수 있다. 실시예들은 명세된 SoS의 요구사항을 목표 모델로 시각적으로 표현하고, 이를 계층적으로 관리할 수 있기 때문에 요구사항 명세 비용을 줄이는데 이바지할 수 있다. 그리고 시각적인 모델링과 관련 모델들간의 참조를 지원하기 대문에 디자인 자체와 다지인에 드는 이권자들간 소통을 효율적으로 만들어 그 비용을 줄일 수 있다. 또한, 본 모델링 도구는 SoS 모델의 시뮬레이션 시나리오, 검증 속성, 테스트 명세 기능도 지원하기 때문에 모델의 분석 비용도 낮추는데 이바지한다. 따라서 실시예들은 복잡하고 거대한 SoS의 모델링을 효율적으로 함으로써 SoS 설계 비용 절감, 모델의 체계적인 관리, 의사소통 비용 절감 등을 기대할 수 있다.

[0137] 이상에서 설명된 장치는 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성요소는, 예를 들어, 프로세서, 컨트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPA(field programmable array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 컨트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.

[0138] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치에 구체화(embody)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.

[0139] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다.

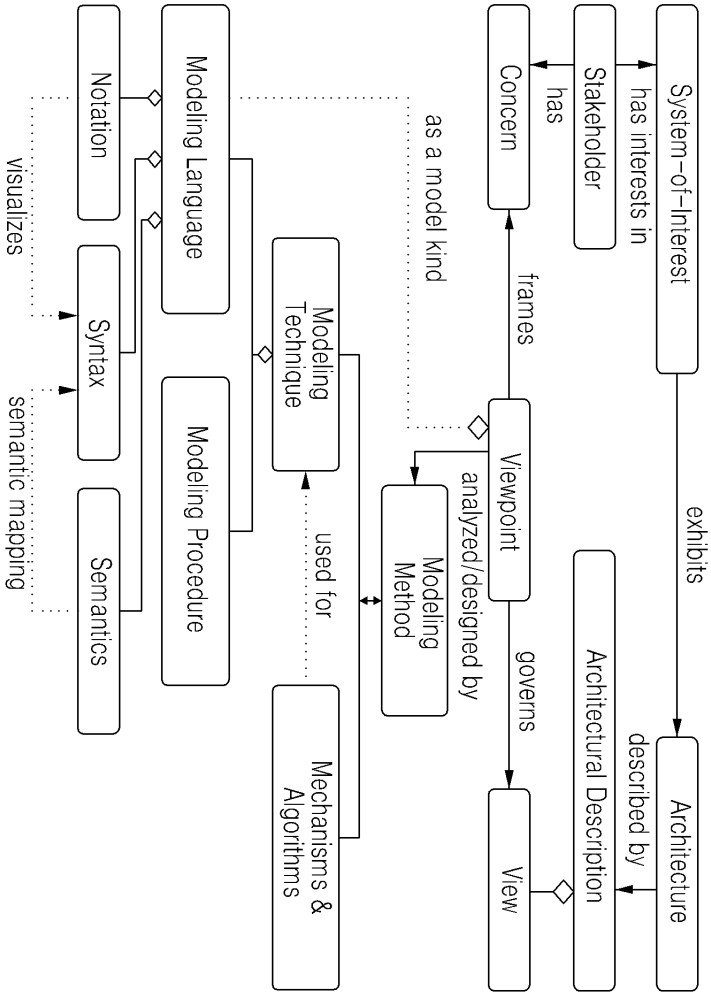
[0140] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

[0141] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한

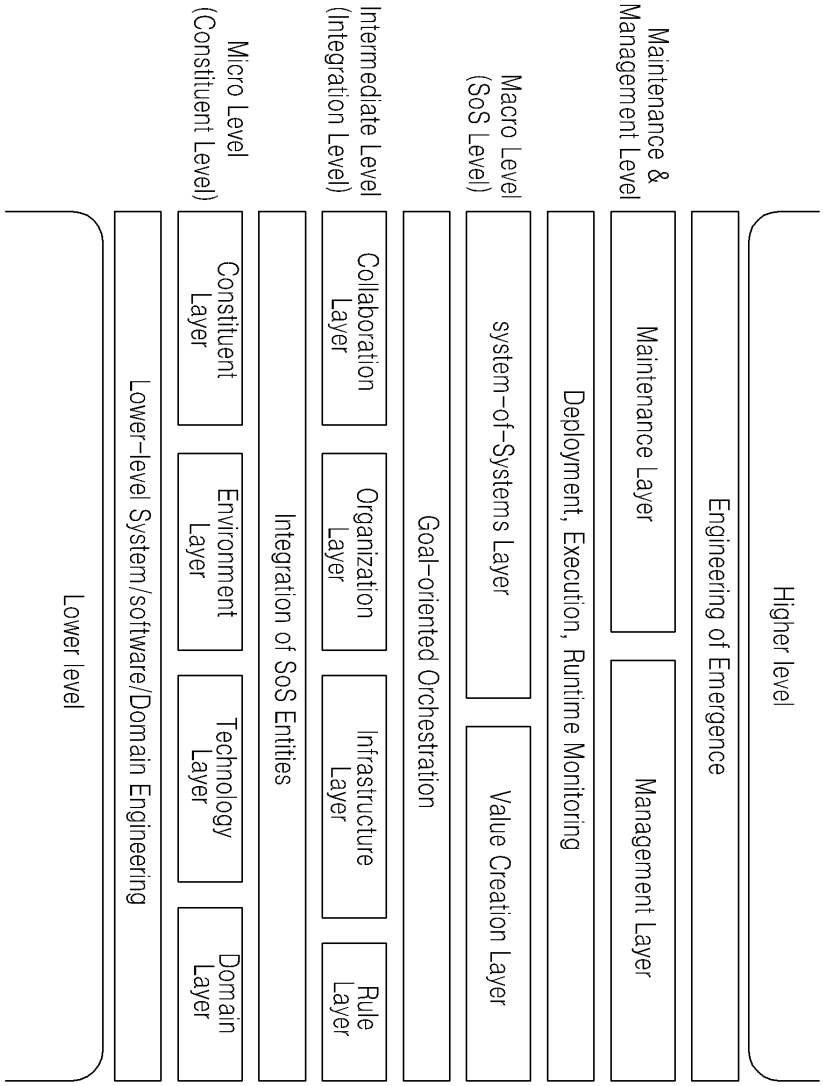
다.

도면

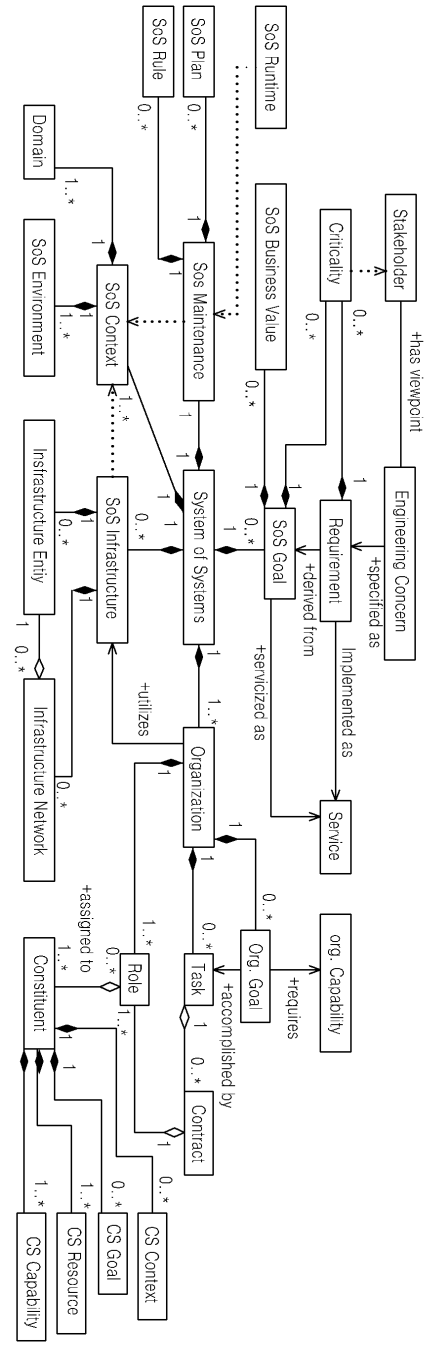
도면1


System-of-Interest
has interests in
Stakeholder
has
Concern
frames
Viewpoint
analyzed/designed by
Modeling Method
as a model kind
Modeling Technique
used for
Mechanisms & Algorithms
governs
View
exhibits
Architecture
described by
Architectural Description
Modeling Language
Modeling Procedure
Notation
Syntax
Semantics
visualizes
semantic mapping

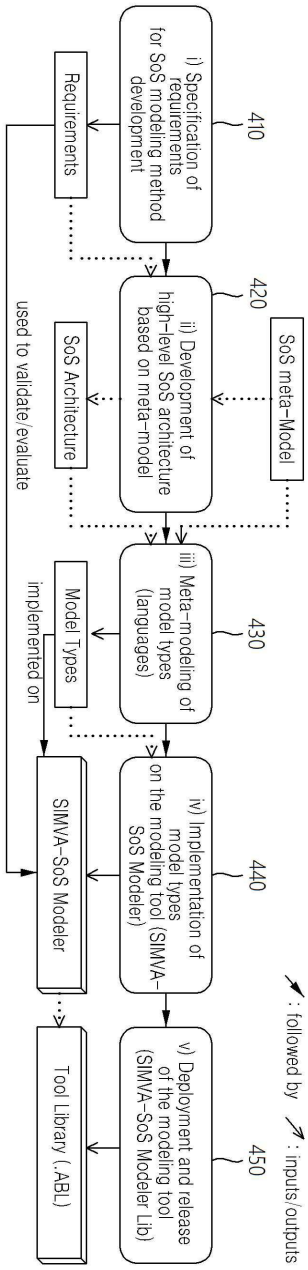
도면2


Higher level
Engineering of Emergence
Maintenance & Management Level
Maintenance Layer
Management Layer
Deployment, Execution, Runtime Monitoring
Macro Level (SoS Level)
system-of-Systems Layer
Value Creation Layer
Goal-oriented Orchestration
Intermediate Level (Integration Level)
Collaboration Layer
Organization Layer
Infrastructure Layer
Rule Layer
Integration of SoS Entities
Micro Level (Constituent Level)
Constituent Layer
Environment Layer
Technology Layer
Domain Layer
Lower-level System/software/Domain Engineering
Lower level

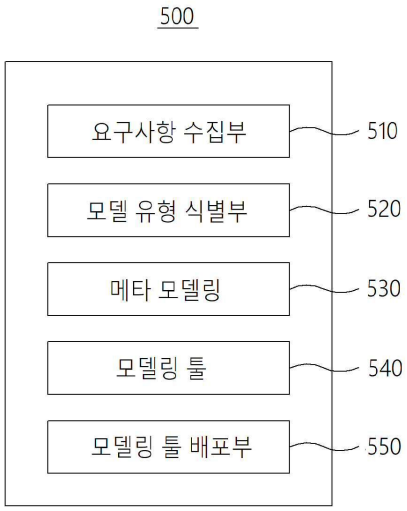
도면3


Stakeholder
+has viewpoint
Engineering Concern
Criticality
SoS Runtime
SoS Business Value
SoS Plan
SoS Rule
Domain
SoS Environment
SoS Context
Sos Maintenance
System of Systems
SoS Goal
Requirement
+specified as
+derived from
Implemented as
Service
+serviceized as
SoS Infrastructure
Infrastructure Entity
Infrastructure Network
+utilizes
Organization
Org. Goal
org. Capability
+requires
Role
Task
+accomplished by
Contract
+assigned to
Constituent
CS Capability
CS Resource
CS Goal
CS Context

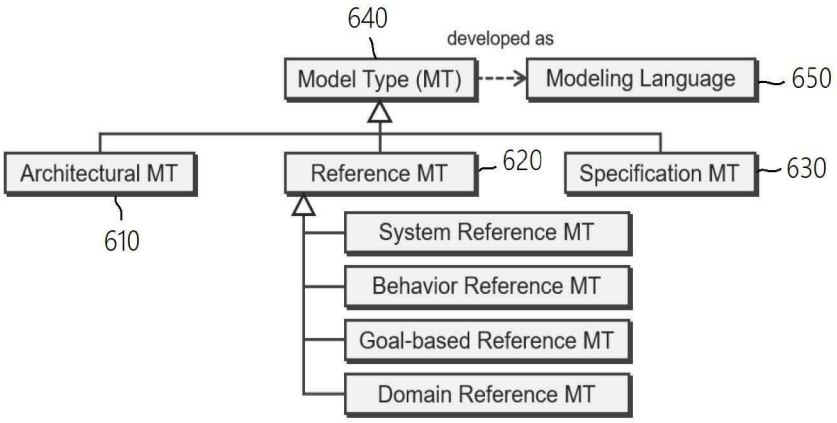
도면4


i) Specification of requirements for SoS modeling method development
410
Requirements
ii) Development of high-level SoS architecture based on meta-model
420
SoS meta-Model
SoS Architecture
used to validate/evaluate
iii) Meta-modeling of model types (languages)
430
Model Types
implemented on
iv) Implementation of model types on the modeling tool (SIMVA-SoS Modeler)
440
SIMVA-SoS Modeler
v) Deployment and release of the modeling tool (SIMVA-SoS Modeler Lib)
450
Tool Library (.ABL)
: followed by
: inputs/outputs

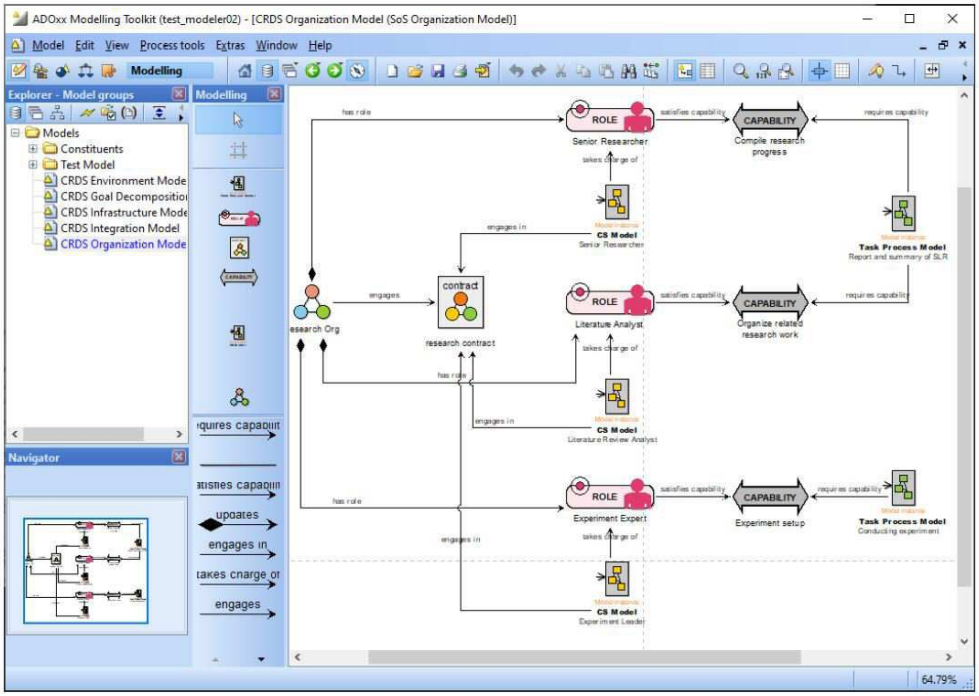
도면5


500
요구사항 수집부
510
모델 유형 식별부
520
메타 모델링
530
모델링 툴
540
모델링 툴 배포부
550

도면6


640
developed as
Model Type (MT)
Modeling Language
650
Architectural MT
610
Reference MT
620
Specification MT
630
System Reference MT
Behavior Reference MT
Goal-based Reference MT
Domain Reference MT

도면7


ADOxx Modelling Toolkit (test_modeler02) - [CRDS Organization Model (SoS Organization Model)]
Model Edit View Process tools Extras Window Help
Modelling
Explorer - Model groups
Models
Constituents
Test Model
CRDS Environment Mode
CRDS Goal Decomposition
CRDS Infrastructure Mode
CRDS Integration Model
CRDS Organization Mode
Navigator
Modelling
updates
engages in
takes charge of
engages
has role
ROLE
Senior Researcher
satisfies capability
CAPABILITY
Compile research progress
requires capability
takes charge of
CS Model
Senior Researcher
Task Process Model
Report and summary of SLR
engages in
engages
contract
research contract
esearch Org
ROLE
Literature Analyst
CAPABILITY
Organize related research work
CS Model
Literature Review Analyst
ROLE
Experiment Expert
CAPABILITY
Experiment setup
Task Process Model
Conducting experiment
CS Model
Experiment Leader
64.79%