이기종 분산 시스템을 위한 통합 관리 시스템 아키텍처 설계

국승학[01], 최훈[1], 김현수[1], 정인상[2], 김점수[3]
[1]충남대학교 전기정보통신공학부 컴퓨터전공, [2]한성대학교 컴퓨터공학과, [3]국방과학연구소
triple888@cnu.ac.kr, hc@cnu.ac.kr, hskim401@cnu.ac.kr, insang@hansung.ac.kr, chskim@add.re.kr

An Architecture of the Integrated Management System for Heterogeneous Distributed System

Seunghak Kuk[01], Hoon Choi[1], Hyeon Soo Kim[1], In-Sang Chung[2], Chum-Su Kim[3]
[1]Dept. of Computer Science & Engineering, Chungnam National University, [2]Dept. of Computer Engineering, Hansung University, [3]Agency for Defense Development

요 약

본 논문에서는 다양한 이기종 분산 시스템을 위한 통합 관리 시스템의 아키텍처를 제시한다. 이는 규모가 큰 다양한 분산 시스템들의 통합 관리 방법의 필요성이 증가함에 따라 이기종 환경에 영향을 받지 않고 시스템을 관리 및 모니터링하는 방법이 요구된다. 이를 위해 본 논문에서는 관리 대상 시스템과 독립적인 추상화된 정보 모델을 이용한다. 또한 관리 대상 시스템의 의존성과 상호작용을 고려하여 어플리케이션 모델을 구축하고 이를 통해 다양한 시스템의 조합과 이를 통제 감시하는 방법이 가능하다.

1. 서 론

컴퓨터 및 정보 통신 기술의 발달로 시스템의 규모가 커지고 복잡해짐에 따라 이러한 요구사항을 만족하기 위한 다수의 분산 컴퓨팅 환경으로 전환되고 있다[1]. 공장의 대형 플랜트, 함정 및 전투관리 시스템 등 규모가 큰 시스템일수록 전체 시스템을 구성하는 다양한 하위 시스템으로 구성된다. 이러한 큰 규모의 분산 시스템 환경에서는 다양한 이기종 시스템이 사용되고 있으며, 그 기능을 확장함에 있어 새로운 이기종 시스템이 지속적으로 도입되고 있다. 예를 들어 그림 1의 제품의 생산 공정 시스템 역시 다양한 생산 관련 시스템의 통합을 통해 전체 공정이 이루어진다[2]. 제품 생산 시뮬레이터 시스템, 생산 관리 시스템, 하위의 생산 시스템 등 다양한 하위 시스템으로 전체 시스템이 구축된다. 이 경우 각각의 시스템은 독립적으로 개발되고 각각을 통합하기 위한 추가적인 노력이 요구된다. 또한 이러한 시스템들이 이기종 환경에서 개발될 경우 전체 시스템에 대한 관리 및 모니터링에 대한 시스템 역시 필요하다.
또 다른 예로 함정 전투 시스템을 간단하게 보면 센서의 입력을 통해 적을 탐지하고, 무기를 통제하고, 교전하는 프로세스를 갖는다. 이때 탐지, 통제, 교전은 각각의 다른 시스템으로 구축되고, 이를 통합하여 전체 기능을 완성한다[3].

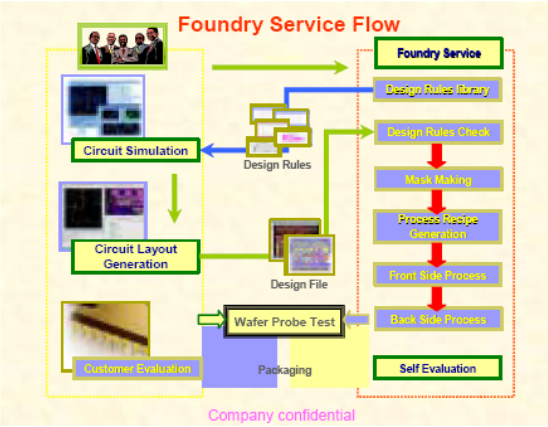


그림 1 재품 생산 과정 및 시스템 구성

그러나 각각의 시스템은 서로 다른 업체에서 독립적으로 개발될 수 있으며 이에 대한 관리 및 모니터링 방법은 구현에 의존적일 수밖에 없다. 만약 이러한 시스템에 새로운 시스템이 추가되는 것을 가정하면 추가되는 시스템의 특성에 따라 관리 및 모니터링 방법이 달라진다. 단일 환경의 시스템의 경우 이를 통합하거나 관리하는 것이 쉽게 이루어질 수 있지만 하위 시스템의 환경이 다양해짐에 따라 통합하여 관리하는 것이 쉽지 않다. 또한 하위의 시스템의 구성이 복잡해지면 이를 전체적으로 관리하고 모니터링 하는 것이 쉽지 않다. 따라서 이러한 다양한 이기종 환경에서 시스템의 제어 및 감시 시스템의 필요하다.

본 논문에서는 이러한 이기종 환경의 분산 시스템을 위한 통합 관리 시스템의 아키텍처를 제안한다. 본 논문에서는 관리 대상 시스템의 추상화된 모델을 이용하여 새로운 환경의 시스템이 도입되더라도 쉽게 추가될 수 있으며, 관련된 전체 시스템에 대한 관리 및 모니터링이 가능하다.

2. 관련연구

다양한 이기종 분산 시스템의 관리 및 모니터링 방법에 대한 기존 연구들이 존재한다[1]. 기존의 분산 시스템 관리 방법은 크게 네트워크 기반 관리 시스템과 시스템 기반 관리 시스템 두 가지로 구분될 수 있다. 네트워크 기반 관리 시스템의 경우 네트워크상의 하드웨어 시스템에 대한 감시 기능을 이용하여 관련된 하드웨어의 상태를 원격지에서 감시한다. 반면 시스템 관리 방법은 하드웨어에 대한 감시와 더불어 사용자 응용에 대한 관리 및 사용자 관리와 관련된 다양한 기능을 중앙에서 효율적으로 처리할 수 있다[1].

[1]의 경우 본 논문이 제시하는 분산 시스템 통합 관리 및 모니터링 방법과 가장 유사한 연구이다. 하드웨어 시스템 및 관련 어플리케이션을 관리함에 있어 원격의 관리 시스템을 이용하여 각각의 관련된 시스템을 모니터링하고 제어할 수 있는 방법을 제시한다. 그러나 이 논문의 경우 이기종 분산 환경에 대한 충분한 고려를 하지 않았다. 관리 시스템과 관리 대상 시스템들이 직접 연결되어 동작하기 때문에 이기종 환경에는 적합하지 않다. 본 논문에서는 이러한 문제를 해결하기 위해 관리 대상 시스템과는 독립적인 정보모델을 구축하고 이를 기반으로 시스템을 관리하며 모니터링한다. 또 다른 연구인 [4]의 경우 P2P 기반의 분산 시스템에 대한 모니터링 방법을 제시하고 있다. 이 논문에서는 P2PMonitor라는 시스템을 구축하고 각각의 시스템에 Alerters라는 모듈을 추가하여 각 시스템이 발생시키는 이벤트를 P2PMonitor에 전달하는 방식으로 모니터링을 수행한다. 그러나 이 논문의 경우 실제 호스트에 존재하는 P2P 어플리케이션만을 대상으로 모니터링을 수행한다. 시스템의 규모가 커지고, 하나의 호스트에 다양한 관리 대상 어플리케이션이 존재할 경우 Alerters의 확장이 필요하다. 또한 P2P 시스템 이외의 이기종 환경에서의 어플리케이션이 추가될 경우 이를 관리하거나 모니터링 할 수 있는 방법을 제시하지는 않는다. 본 논문에서는 어떠한 어플리케이션이든 이에 대한 추상화된 모델을 이용하기 때문에 다양한 시스템에 대한 관리 및 모니터링이 가능하다. [5]의 경우 이기종 분산 시스템의 통합 방법과 제어 메커니즘을 제시하였다. 이 논문에서는 다양한 이기종 분산 시스템들 조합하기 위한 정책을 제시하고, Law-Governed Interaction이라는 제어 전달 방법을 제시하였다. 그러나 본 논문에서는 이기종 시스템을 전체적으로 모니터링하고 제어할 수 있는 시스템의 아키텍처를 제시한다. [5] 논문에서의 시스템 통합 관점이 아닌 시스템의 관리 관점에 초점을 두고 있다. [6]의 경우 [5]의 논문과 같이 이기종 분산 시스템에서의 데이터 및 제어 방법에 대한 논문이다. 이기종 분산 시스템에서의 병렬적 특성과 작업의 제어를 위해 각각의 시스템들의 커뮤니케이션 패턴을 분석하고, 이에 따른 스케줄링 방법을 적용한 시스템을 제시하였다. 그러나 앞선 다른 연구들과 마찬가지로 이 연구 역시 이기종 시스템 사이의 통합 및 커뮤니케이션 방법에 대해 초점을 맞춘 연구이다. 본 논문에서는 이기종 분산 시스템의 관리 및 모니터링을 위한 시스템의 아키텍처를 제시한다.

3. 통합 관리 시스템 아키텍처 설계

본 논문의 통합 관리 시스템은 기존의 시스템들의 소프트웨어 및 하드웨어에 대한 정보 모델을 추출하여 관리하고, 이를 기반으로 시스템을 모니터링하고 관리할 수 있는 구조로 설계하였다. 이를 통해 기존의 시스템들의 독립성을 유지하면서 관리에 필요한 기능을 쉽게 추가, 제거할 수 있다. 또한 독립적인 정보모델을 유지함으로써 새로운 관리 대상 시스템이 추가되거나, 기존의 시스템이 제거될 때 이를 쉽게 반영할 수 있는 구조로 설계하였다.

3.1 전체 아키텍처

다음 그림은 본 논문에서 제시하는 통합 관리 시스템의 전체 아키텍처의 계층적인 구조를 보여준다.

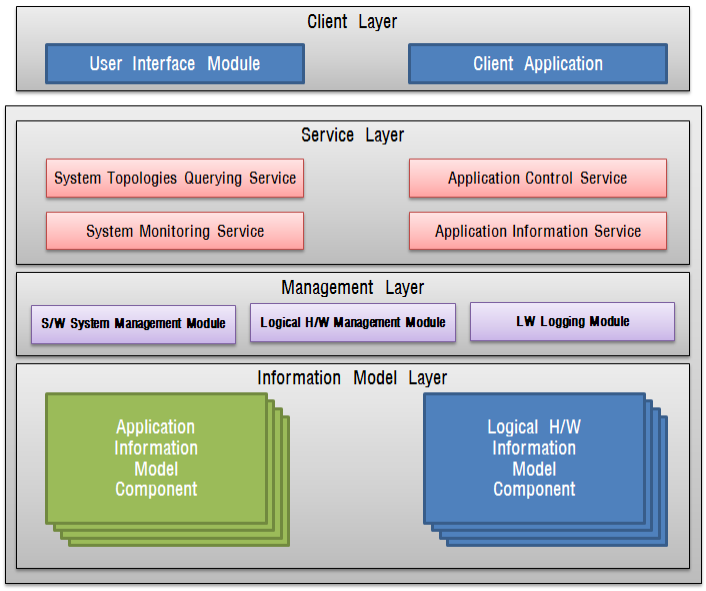


그림 2 시스템 아키텍처

본 논문의 시스템은 크게 클라이언트 계층, 서비스 계층, 관리 계층, 정보 모델 계층으로 구성된다. 다음은 각 계층에 대한 설명이다.

(1)클라이언트 계층(Client Layer)

클라이언트 계층은 이 통합 시스템을 사용하는 사용자에게 다양한 정보를 제공할 수 있는 계층이다. 일반적으로 시스템 자체의 사용자 인터페이스를 제공함으로써 기본적인 제어 기능 및 모니터링 정보를 제공할 수 있다. 또

한 기존의 다른 어플리케이션이나 새로운 기능을 추가할 수 있도록 클라이언트 어플리케이션과 통합할 수 있는 인터페이스를 제공함으로써 통합 시스템을 확장할 수 있는 기능을 제공한다.

(2)서비스 계층(Service Layer)

서비스 계층은 관리 대상이 되는 다양한 소프트웨어 및 하드웨어에 대한 여러 가지 정보를 서비스 형태로 제공한다. 이는 각 관리 대상에 대한 정보 요청 서비스, 각 관리 대상에 대한 제어 서비스, 관리 대상에 대한 모니터링 서비스, 시스템의 구성 정보 서비스로 구성된다. 이러한 서비스들은 클라이언트 계층에 필요한 정보를 제공할 수 있는 인터페이스 역할을 수행하며, 클라이언트 계층의 요청을 관리 계층의 기능을 조합해 수행된다.

- "System Topology Querying Service"는 시스템 토폴로지(System Topology)에 대한 질의 처리 기능을 제공한다. 여기에서 시스템 토폴로지는 관리 대상이 되는 소프트웨어 및 하드웨어의 배치 정보를 나타낸다.
- "Application Control Service"는 어플리케이션을 제어하는 모듈이다. 이는 관리 대상 어플리케이션의 실행, 중지, 배치 등의 생명주기를 제어한다.
- "System Monitoring Service"는 시스템을 감시하는 모듈이다. 관리 대상 소프트웨어 및 하드웨어 시스템의 상태를 주기적으로 모니터링하거나 상태변화가 일어날 경우에 알리는 기능을 수행한다.
- "Application Information Service"는 관리 대상 어플리케이션 동작에 필요한 관련 정보나 어플리케이션 자체의 정보를 관리하고 제공하는 모듈이다.

(3)관리 계층(Management Layer)

이 계층은 관리 대상이 되는 소프트웨어 및 하드웨어 시스템을 실질적으로 관리하는 계층이다. 이러한 관리 계층은 크게 소프트웨어 시스템을 관리하는 부분과 하드웨어 시스템을 관리하는 부분으로 나뉘어진다.

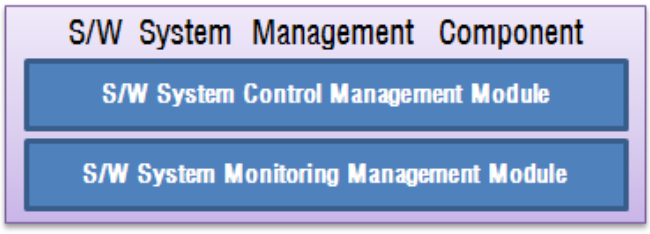


그림 3 소프트웨어 시스템 관리 컴포넌트

소프트웨어 시스템 관리 컴포넌트는 관리 대상이 되는 소프트웨어 시스템을 실질적으로 제어하거나, 모니터링하는 기능을 제공한다. 내부적으로는 관리 모듈과 모니터링 모듈로 구성된다.

- “S/W System Control Management” 모듈은 소프트웨어 시스템의 생명주기를 관리하기 위한 기능을 담당한다. 예를 들어 특정 소프트웨어를 시작시키거나 종료시키는 기능을 담당하며, 각각의 소프트웨어의 의존성에 따라 제어하는 기능을 담당한다.
- “S/W System Monitoring Management” 모듈은 관리 대상 소프트웨어 시스템의 상태를 감시하는 기능을 수행한다. 각각의 소프트웨어 시스템은 논리적인 상태를 갖는다. 이러한 상태는 크게 작업을 진행 중인 Running 상태, 작업이 종료된 Stop 상태, 작업 도중 오류가 발생한 Error 상태이다. 이 모듈에서는 실제 소프트웨어 시스템이 동작 중인 호스트에 주기적인 모니터링 정보를 요청하여 각각의 소프트웨어 시스템에 대한 상태정보를 유지한다.

다음 그림은 하드웨어 시스템 관리 컴포넌트의 내부 구조를 보여준다. 이는 관리 대상이 되는 하드웨어 시스템에 대한 실질적인 관리 기능을 제공한다. 하드웨어 시스템의 경우 원격의 관리 시스템에서 제어하는 것이 쉽지 않다. 예를 들어 특정 호스트를 시작시키거나 종료시키는 것이 기반 환경이 제공되지 않는다면 불가능하다. 따라서 이 컴포넌트에서는 주로 각각의 하드웨어 시스템에 대한 모니터링 기능을 수행한다. 그러나 단지 모니터링 기능만을 수행하는 것이 아니라 논리적인 단위의 하드웨어 그룹을 관리함으로써 사용자가 원하는 시점에 원하는 대상에 대한 모니터링을 수행할 수 있는 기능을 제공한다.

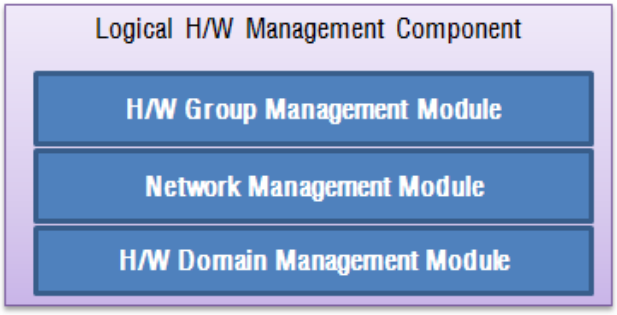


그림 4 논리적 하드웨어 시스템 관리 컴포넌트

- “H/W Group Management" 모듈에서는 컴퓨터 시스템들의 그룹화를 표현하고 각 시스템의 이름, 상태 등의 정보를 관리한다. 하드웨어 그룹의 관련된 모든 하드웨어 시스템의 통합된 상태에 대한 정보를 관리한다. 이는 하드웨어 그룹 내의 각 하드웨어 아이템들의 주기적인 상태를 받아오거나, 특정 하드웨어 아이템들의 상태 변화 시 정보를 알린다.
- "Network Management" 모듈에서는 네트워크로 연결된 각종 컴퓨터 시스템들에 대한 관리를 한다. 라우터, 프린터, 호스트, 스위치 등이 그 대상이 될 수 있으며, 네트워크 연결에 대한 상태 정보도 관리한다.
- "H/W Domain Management" 모듈에서는 단일 관리 도메인에서의 컴퓨터들의 집합에 대한 모델을 제공한다. 하드웨어 도메인을 관리하고 시스템에서 컴퓨터들의 노드를 나타낸다.

본 논문에서는 실질적인 관리 기능을 수행하는 관리 계층과 서비스 계층을 분리하였다. 이는 관리 대상의 하드

웨어 및 소프트웨어 시스템에 대한 관리 기능을 조합하여 새로운 서비스를 쉽게 추가 제거할 수 있는 구조를 갖기 위함이다.

(4)정보 모델 계층(Information Model Layer)

이 계층에서는 실질적인 관리 대상이 되는 하드웨어 및 소프트웨어 시스템에 대한 정보 모델을 나타낸다. 본 논문에서는 관리 대상이 되는 시스템과 실제 관리 시스템에서의 정보모델을 분리하여 관리한다. 이를 통해 관리 시스템과 실제 대상 시스템의 독립성을 보장한다. 따라서 새로운 시스템의 추가나 기존 시스템의 제거, 각 시스템의 의존성 관리와 같은 기능을 손쉽게 추가, 제거할 수 있는 구조로 설계하였다. 또한 이기종 환경의 소프트웨어들의 추상적인 모델을 이용하기 때문에 각각의 환경에 영향을 받지 않고 소프트웨어나 하드웨어 시스템의 관리 및 모니터링이 가능하다.

- 어플리케이션 정보 모델 컴포넌트(Application Information Model Component)

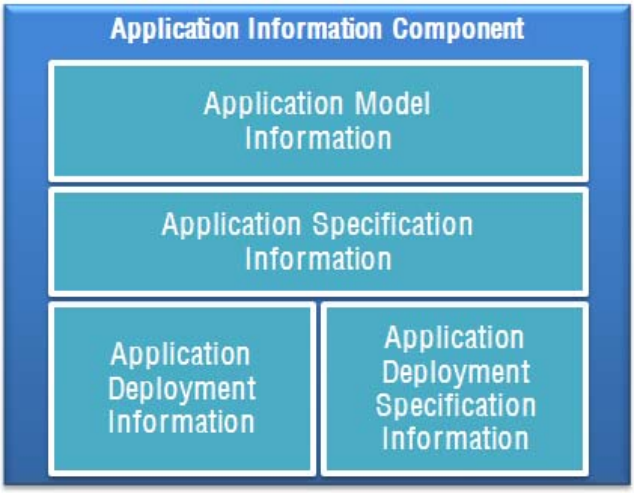


그림 5 어플리케이션 정보 모델 컴포넌트

관리 대상 소프트웨어 시스템은 어플리케이션 정보 모델 컴포넌트(Application Information Model Component)들로 나타낼 수 있다. 이 컴포넌트는 어플리케이션 정보 관리를 위해 구성되는 컴포넌트로 어플리케이션의 동작과 관련 정보에 대해서 다루게 된다. 이 컴포넌트 내부 모듈의 역할은 다음과 같다.

- "Application Model Information" 모듈은 어플리케이션을 실행하는 동안 관리하고 모니터링 하는데 필요한 정보 모델로 이루어진다. 어플리케이션의 실행 시작, 중지 등의 기본적인 동작을 비롯하여 상태 정보 등을 얻어 오는데 사용된다.
- "Application Specification Information" 모듈은 어플리케이션을 모델링하는데 필요한 정보 모델로 이루어지며 소프트웨어 시스템이 배치되기 이전의 정보를 유지 및 관리한다.
- "Application Deployment Information" 모듈은 어플리케이션을 실행하기 위한 배치 환경을 기술하는데 필요한 정보 모델이다. 배치의 실행, 중지를 비롯한 환경 설정 등을 제공한다.
- "Application Deployment Specification Information" 모듈은 배치 환경설정을 나타내고 그 결과로써 배치됨을 보이는 데 필요한 정보 모델로 이루어진다. 호스트와 소프트웨어 항목 간의 연결에 대해서 명세하거나 배치 환경 설정에 대한 정의를 제공하는 모듈이다.

- 논리적 하드웨어 정보 모델(Logical H/W Information Model Component)

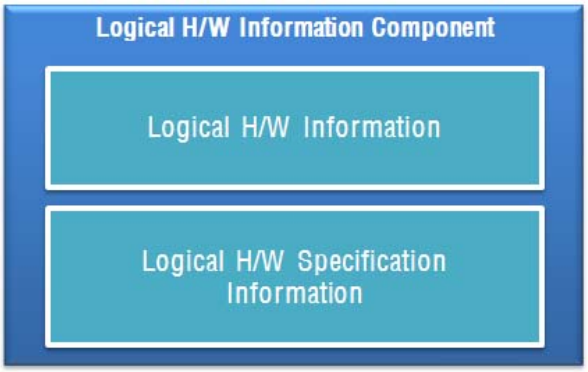


그림 6 논리적 하드웨어 정보 컴포넌트

하드웨어 시스템은 논리적 하드웨어 정보 모델 컴포넌트(Logical H/W Information Model Component)들로 나타낼 수 있다. 이 컴포넌트에서는 하드웨어 엘리먼트들의 집합으로써의 컴퓨터, 네트워크 등의 정보를 제공한다. 이 컴포넌트 내부 모듈의 역할은 다음과 같다.

- "Logical Hardware Information" 모듈은 효과적인 하드웨어 토폴로지를 기술한 정보모델이다.
- "Logical Hardware Specification Information" 모듈은 개별 하드웨어를 위한 환경 설정 명세를 기술하는 정보모델로 구성된다.

3.2 어플리케이션 모델

본 논문에서 제시하는 통합 관리 시스템은 전체 시스템을 구성하는 다양한 하드웨어와 소프트웨어를 관리 및 모니터링하는 시스템이다. 그러나 관리 대상 소프트웨어들은 단순히 독립적으로 동작하는 것이 아니라 상호간에 영향을 미칠 수 있다. 따라서 각각의 상호작용 및 의존성에 대한 정보가 충분히 반영되어 있어야 사용자가 의도하는 관리가 가능하다. 본 논문에서는 이러한 의존성과 상호작용을 고려한 상위 개념의 어플리케이션 모델을 이용하여 각 소프트웨어 시스템의 관리 및 모니터링 기능을 설계하였다. 어플리케이션 모델을 구성하는 가장 기본이 되는 단위는 소프트웨어 시스템의 정보 모델이다. 아래 그림은 본 논문의 관리 시스템에서의 정보 모델과 실제 소프트웨어와의 관계를 나타낸다. 동작 가능한 프로그램에 관한 정보는 어플리케이션 스펙 모델을 통해 관리하며, 해당 프로그램의 프로세스는 어플리케이션 정보 모델에서 관리한다. 또한 해당 프로그램이 위치

하는 호스트에 대한 정보는 논리적인 하드웨어 정보 모델에서 관리하며, 프로그램과 호스트와의 관계정보는 어플리케이션 배치 모델에서 관리한다.

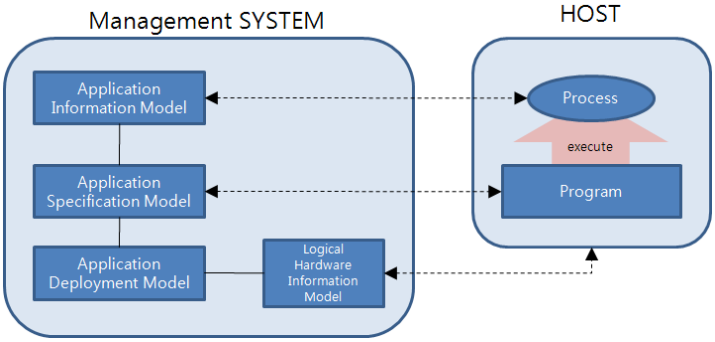
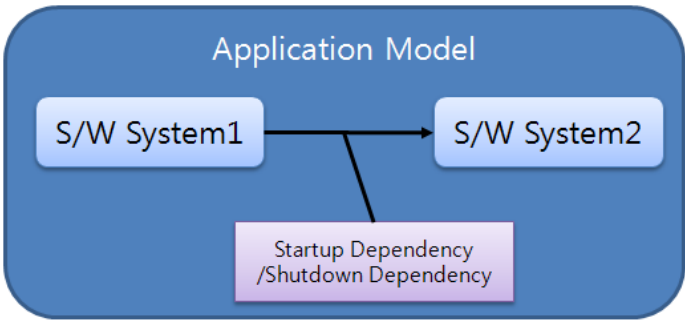


그림 7 소프트웨어 모델

이러한 소프트웨어 시스템들 사이의 의존성과 상호작용은 크게 Startup Dependency와 Shutdown Dependency로 나눌 수 있다. 이는 특정 소프트웨어 시스템이 시작된 이후에 다음 시스템이 시작되거나, 특정 소프트웨어 시스템이 종료되면 관련된 시스템들이 함께 종료되어야 함을 나타낸다. 따라서 어플리케이션 모델이서는 이러한 의존성 관계를 정보 모델로 나타내고, 소프트웨어 시스템을 자동으로 제어할 수 있도록 한다. 다음 그림은 의존성과 상호작용을 고려한 어플리케이션 모델을 나타낸다.




```
<Application Model>
  <SoftwareSystem ref='S/WSystem1'>
    <ShutdownDependency type="post" target="S/WSystem2" delay="10"/>
  </SoftwareSystem>
  <SoftwareSystem ref='S/WSystem2'>
    <StartupDependency type="pre" target="S/WSystem1" delay="100"/>
  </SoftwareSystem>
</Application Model>
```

그림 8 어플리케이션 모델

3.3 기존 시스템과의 통합 방법

본 논문에서 제시하는 통합 관리 시스템은 기존의 관리 대상 시스템과 독립적인 시스템으로 원격의 시스템을 관리한다. 따라서 원격의 시스템을 제어하거나 모니터링 하는 기능을 제공해야 한다. 이를 위해 각각의 관리 대상 호스트에 제어 및 모니터링 관련 모듈을 추가한다. 본 논문에서는 이를 호스트 측 관리 모듈(Host Side Management Module)이라한다. 이 모듈은 실제 통합 관리 시스템과 네트워크로 연결되어 있고, 필요한 제어 정보나 모니터링 정보를 송/수신하는 기능을 담당한다. 다음 그림은 관리 대상 시스템과 통합 관리 시스템 사이의 연동 방법을 나타낸다.

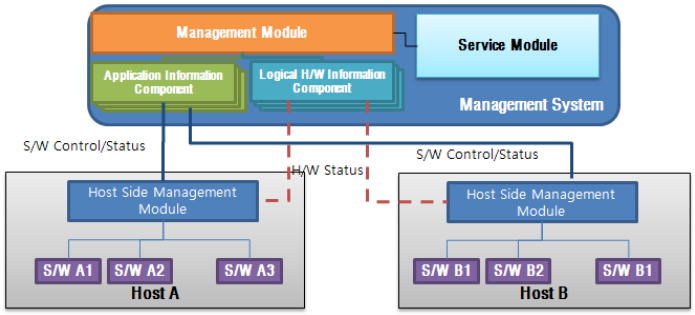


그림 9 기존 시스템과의 통합

위와 같이 관리 대상 시스템과 통합 관리 시스템을 분리함으로써 새로운 관리 대상이 추가되더라도 관리 대상 시스템에 영향을 주지 않는다. 예를 들어 일반적인 소프트웨어 시스템을 관리하는 시스템에 CORBA 기반의 어플리케이션이 추가되는 경우, 대상 관리 시스템의 호스트 측 관리 모듈을 추가하거나 수정하여 그 영향 범위를 최소화 할 수 있다.

3.5 통합 관리 시스템 동작 시나리오

본 논문이 제시하는 관리 시스템의 동작 시나리오는 다음 그림과 같다. 아래 그림에서는 대상 소프트웨어 시스템을 시작 시키고 상태 정보를 모니터링하는 과정을 나타낸다.

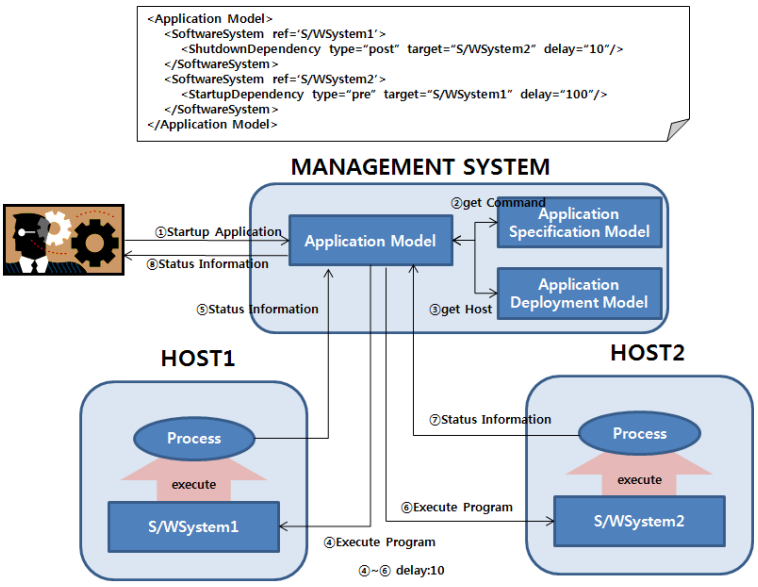


그림 10 동작 시나리오

우선 관리 대상이 되는 소프트웨어 및 하드웨어 시스템에 대한 정보 모델이 그림 8과 같이 기술되어 있음을 가정한다. 사용자는 관리 시스템을 통해 관리 대상 어플리케이션 모델을 시작시킨다. 이때 해당 소프트웨어를 실

행 시키는 명령어 정보와 어떤 호스트 상에서 동작하는지에 대한 정보는 어플리케이션 스펙 모델과 어플리케이션 배치 모델에 정의되어 있다. 관리 시스템은 이러한 정보를 이용하여 원격의 호스트에 제어 정보를 전달한다. 어플리케이션 모델에서는 S/WSystem1과 S/WSystem2의 실행 순서가 명시되어 있다. S/WSystem1이 먼저 수행되어야 하며, delay 10이후에 S/WSystem2가 실행되어야 한다. 원격의 Host1은 S/WSystem1에 대한 실행 요청을 받아 해당 프로그램을 시작시키고, 실행 후 상태 정보를 관리 시스템에 전달한다. delay 10 이후에 시스템은 Host2에 S/WSystem2의 실행을 요청하고, 실행 후 상태정보를 받는다. 관리 시스템은 이러한 상태정보를 사용자에게 전달하고, 이후에 주기적인 모니터링을 수행하여 소프트웨어의 동작 상태 정보를 사용자에게 알린다. 이러한 어플리케이션 모델을 통해 각각 다른 환경에서 개발된 소프트웨어 시스템들의 관계 정보를 명시하고, 이를 통해 각각의 소프트웨어 시스템들의 제어가 가능하다. 또한 소프트웨어 시스템의 수행 이후 주기적인 모니터링을 수행하고, 문제가 발생 시 관련된 소프트웨어 시스템을 종료시킨다. 예를 들어 위 어플리케이션 모델에서는 S/WSystem1이 종료되면 delay 100 이후에 S/WSystem2를 종료시킨다. 향후 이러한 어플리케이션 모델과 관리 시스템의 확장을 통해 특정 오류가 발생하는 경우 이를 재시작하거나 복구 하는 것도 가능하다.

3.5 제시하는 통합 관리 시스템의 장점

본 논문에서 제시하는 통합 관리 시스템은 다음과 같은 장점을 갖는다.

- 관리 대상 소프트웨어 및 하드웨어와 독립적인 정보 모델을 이용하여 관리 및 모니터링 기능을 제공: 새로운 관리 대상 시스템을 추가하거나, 기존의 시스템을 제거 또는 변경하는 것이 쉽다.
- 서비스 계층과 실제 관리 계층을 분리: 관리 기능을 이용한 새로운 서비스를 추가하는 것이 가능하다. 이러한 서비스는 클라이언트 계층을 통해 사용자에게 제공된다. 이때 단순히 사용자 인터페이스를 통해 관련 정보를 제공하는 것뿐만 아니라 서비스 인터페이스 모듈을 통해 다른 시스템과의 연동이 용이하다.
- 소프트웨어 시스템의 의존성 및 상호작용 : 단순히 개별 소프트웨어 시스템을 관리하거나 모니터링하는 것이 아니라 각각의 소프트웨어 시스템의 의존성 및 상호작용에 대해 고려하고, 이를 상위 개념인 어플리케이션 모델로 적용함으로써 자동화된 관리 기능을 제공할 수 있는 구조이다.

4. 결 론

최근 시스템의 규모가 점점 커져감에 따라 다양한 시스템의 통합 관리에 대한 요구사항이 증가하고 있다. 이러한 대규모 시스템은 일반적으로 단일 시스템으로 구성되는 것이 아니라 다양한 이기종 시스템이 통합되는 구조를 갖는다. 이러한 환경에서 이기종 시스템에 대한 관리 및 모니터링을 위한 시스템이 필요하다. 이에 본 논문에서는 이기종 시스템을 위한 통합 관리 시스템을 위한 아키텍처를 설계하였다. 본 논문의 시스템은 실제 관리 대상이 되는 하드웨어와 소프트웨어 시스템을 위한 추상적인 정보모델을 이용하여 실제 시스템과 독립적인 방법으로 관리 및 모니터링 하는 것이 가능하다. 이를 통해 이기종 환경으로 인한 의존성을 해결하였다. 또한 관리 및 모니터링 기능과 실제 사용자에게 제공하는 서비스를 분리하여 사용자가 원하는 다양한 조합의 기능을 제공하는 것이 가능하다. 현재 본 논문의 시스템의 프로토타입을 구현하고 있으며, 향후 이를 다양한 시스템에 적용하여 그 타당성을 검증할 예정이다.

참고문헌

[1]윤영현, 안순신, “ ISMM:분산 응용 환경을 위한 통합 김시 및 관리 시스템의 설계 및 구현”, 정보과학회 논문지, vol. 4, pp 265-279, 1997

[2]이승우, 박지훈, 박재우, 이화기, “화합물 반도체 공장의 통합생산 시스템 설계에 관한 연구”, 산업경영시스템학회지, vol. 32, pp258-261, 2003

[3]김영길, “전투체계 개념연구”, 해군해양과학기술심포지움, 2003

[4]Serge Abiteboul, Bogdan Marinoiu, Pierre Bourhis, "Distributed Monitoring of Peer-to-Peer Systems" International Conference on Data Engineering, 2008

[5] NAFTALY H. MINSKY and VICTORIA UNGUREANU, "Law-Governed Interaction: A Coordination and Control Mechanism for Heterogeneous Distributed Systems", ACM Transactions on Software Engineering and Methodology, vol. 9, pp 273-305, 2000

[6]Prashanth B. Bhat, C. S. Raghavendra, and Viktor K. Prasanna, “Efficient collective communication in distributed heterogeneous systems”, Journal of Parallel and Distributed Computing, vol. 63, pp 251-263, 2003