1. Introduction

오늘날의 컴퓨터 시스템은 점점 더 크고 복잡해지면서 대부분 **분산 시스템(distributed systems)** 구조를 기반으로 운영되고 있다. 분산 시스템은 여러 대의 컴퓨터나 프로세서가 협력하여 동작하는 구조를 가지며, 그 특성상 다루기가 단일 스레드 순차 프로그램보다 훨씬 어렵다. 이는 각 구성 요소가 언제든 독립적으로 동작할 수 있고, 메시지 전달 여부와 시점을 예측하기 어렵기 때문이다. 또한 다른 구성 요소의 동작 역시 불확실하기 때문에 전체 시스템의 정확성을 보장하는 일이 쉽지 않다. 대표적인 예로, 잘 알려진 **Needham–Schroeder 공개키 프로토콜(NSPK)**은 단 세 줄의 명세로 이루어져 있음에도 불구하고 오랫동안 안전한 것으로 여겨졌으나, 이후 공격 가능성이 발견되면서 그 한계가 드러났다. 이 사례는 단순한 분산 프로그램 조차 올바르게 설계하기가 매우 어렵다는 점을 잘 보여준다.

이러한 이유로 분산 시스템의 설계와 구현 과정에서는 **모델링(modeling)** 단계가 필수적이다. 과거의 소규모 프로그램은 집을 바로 짓는 것처럼 즉각적으로 작성할 수 있었지만, 오늘날의 복잡한 시스템은 마치 고층 건축물과 같이 정교한 모델을 선행하여 설계 검증을 수행해야 한다. 모델은 불필요한 세부사항을 추상화(abstraction)하면서도 핵심 속성은 유지해야 하며, 이를 통해 시스템의 안전성, 교착 상태 회피 가능성, 보안 취약성 여부 등을 사전에 분석할 수 있다. 따라서 분산 시스템 모델은 명확하고 정밀한 수학적 규칙을 갖춘 **형식적 모델(formal model)**로 표현되어야 하며, 시스템 동작을 정의하는 **시스템 명세(system specification)**와 반드시 충족해야 할 속성을 정의하는 **요구사항 명세(requirement specification)**를 구분하여야 한다. 궁극적으로는 시스템 모델의 모든 가능한 동작이 요구사항을 만족함을 **증명**하는 것이 목표이다.

모델은 단순한 추상적 그림에 머무르지 않고 실제 시스템 구현으로 이어져야 한다. 실제로 구현 단계에서 발생하는 결함보다 설계 단계에서 발생하는 결함이 훨씬 많으며, 이를 조기에 검증하는 것이 비용과 위험을 크게 줄인다. NASA 보이저와 갈릴레오 우주선 사례에서도 치명적인 결함의 대부분은 코딩 오류가 아닌 설계상의 문제에서 비롯된 것으로 보고된 바 있다. 이는 초기 설계 검증의 중요성을 단적으로 보여준다.

이러한 맥락에서 **Maude**는 분산 시스템을 모델링하고 분석하기 위한 강력한 도구로 주목받는다. Maude는 **재작성 논리(rewriting logic)**를 기반으로 하여, 데이터 타입을 방정식(equations)으로 정의하고 시스템의 동적 동작을 재작성 규칙(rewrite rules)으로 정의한다. 이를 통해 사용자는 개별 실행 경로를 **시뮬레이션(rewriting)** 하거나, 특정 상태의 도달 가능성을 **탐색(search)**하고, 나아가 선형 시간 논리(LTL)를 활용하여 요구사항 충족 여부를 **모델 검증(model checking)** 할 수 있다. Maude는 직관적이면서도 표현력이 뛰어나 복잡한 분산 시스템을 다룰 수 있으며, 보안 프로토콜, 네트워크 통신, 임베디드 소프트웨어, 생물학적 시스템 모델링 등 다양한 분야에서 실제 연구와 응용이 활발히 이루어지고 있다.