Towards a New Physics of Software Systems: A Finite Element and Spectral Framework for Distributed Architectures

Anand Sunder

Independent Researcher
Email: *anand.sunder[at]syf.com*

Abstract: ***Modern distributed software systems exhibit complex, non-linear, and emergent behaviors under load, failure, and dynamic scaling. Traditional models based on graph theory, queuing theory, and discrete event simulation fail to fully capture the continuous propagation of stress, failure, and performance degradation across interconnected services. This paper introduces a novel framework termed the Physics of Software Systems, which models distributed architectures using principles derived from continuum mechanics, finite element analysis (FEA), and spectral theory. We define software analogues of physical constructs such as force, stiffness, deformation, and stress, and formulate a global system equation governing system behavior. The proposed approach enables predictive analysis of failure propagation, bottleneck identification, and system resilience through eigenvalue-based spectral metrics.***

Keywords: distributed software behavior, failure propagation modeling, system stress analysis, continuum inspired computing, spectral resilience metrics

1. Introduction

Distributed software systems, particularly microservice-based architectures, have reached a level of complexity where traditional analytical tools are insufficient.

Failures propagate non-locally, performance degrades non-linearly, and system behavior often resembles physical systems under stress.

This paper proposes a foundational shift: treating software systems as continuous, deformable systems governed by physical laws.

Fundamental Definitions

- Software System Domain: Ω = {S1, S2, ..., Sn}
- Software Force: fi = incoming request rate
- Software Displacement: xi = latency or degradation
- Software Stiffness: kij = interaction strength between services

Finite Element Formulation

- Element stiffness matrix: [[kii, -kij], [-kji, kjj]]
- Global stiffness matrix: K = Σ ke
- Governing equation: Kx = f

Stress and Failure Propagation

- Software stress: σi = Σ kij (xi - xj)
- High stress regions indicate bottlenecks and failure-prone services

Spectral Analysis

- Eigenvalue equation: Kφ = λφ
- Spectral Resiliency Index (SRI) = λmin / λmax
- Lower SRI indicates higher fragility

Applications

- Failure prediction using stress distribution
- System optimization via stiffness tuning
- Dynamic auto-scaling based on system state

2. Conclusion

This work introduces a new paradigm: Physics of Software Systems.

It enables predictive, analytical, and physically interpretable modeling of distributed architectures.

Paper ID: MR26323100538 DOI: https://dx.doi.org/10.21275/MR26323100538