Name : Isiram OUKAKI
Name of your level 1 : Morgane Gaspar
Tittle : One the evolution of Lehman's Laws
Keyword specific : software evolution, engineering, laws, industrial experiences, agile development, cloud-based services, run-time environments, software systems, development, emergent uses.
In this brief paper, we acknowledge the contributions of the late Prof. Manny Lehman to software evolution studies. We delve into his engineering background, which influenced his perspectives on software systems and development. Then, we explore the laws of software evolution he formulated from his industrial insights. Finally, we analyze the significant changes occurring in software systems and their development, contemplating their implications for future evolutionary studies.

I. LEHMAN'S INTELLECTUAL JOURNEY

Meir "Manny" Lehman's unconventional journey into academia started during WWII repairing civilian radios. Despite being relegated to maintenance, he emphasized the value of thinking in software maintenance. Lehman's focus shifted to investigating programming practices during his time at IBM. His project on IBM S/360 and S/370 highlighted a trend among programmers prioritizing productivity over product quality. Partnering with Laszlo Belady, he challenged prevailing software maintenance assumptions. Joining Imperial College London in 1972, Lehman continued his research on software evolution, leveraging insights gained from IBM to understand real-world software development challenges.

II. LEHMAN'S LAWS OF EVOLUTION

Lehman emphasized that once a program is installed, its environment changes, making it impossible to fully specify it. As the program interacts with its environment, both the program and its surroundings evolve in unpredictable ways. This creates a feedback loop where the program must continually adapt to meet changing requirements. Lehman stressed that software evolution is inherent and essential in real-world systems, with requirements evolving over time. Programmers play a crucial role in evolving software systems, but Lehman acknowledged that it's a challenging task.

III. SOME OBSERVATIONS ON MODERN SOFTWARE EVOLUTION AND LEHMAN'S LAWS

Lehman formulated his observations on software evolution, later termed his "laws," during the 1960s and 1970s while working at IBM and developing large systems like OS/360. He continued to refine these laws into the 1990s. However, his experiences were primarily with large, systems-oriented applications developed using traditional management styles, programming languages, and tools. Given the significant changes in the software landscape, including agile development, cloud-based services, and advanced run-time environments, it's essential to consider how these trends may impact Lehman's laws. This prompts a discussion on recent software development trends and how Lehman's laws may need to adapt to them.

A. The Emergence of Software Architecture

Developing a workable software architecture for large systems is crucial but challenging. As developers gain understanding of the problem space, internal boundaries and interfaces within the system begin to solidify. Well-designed interfaces can hide complexity and facilitate component interaction. However, the complexity of a software system cannot be solely judged by component size; rather, it depends on the details necessary to interact with subcomponents.

B. The De-monolithization of Software Systems

Modern software systems are no longer monoliths but are embedded within a network of peer components, including libraries, frameworks, and services. Developing such systems involves understanding available services, evaluating security concerns, and specifying deployment details. This complexity doesn't always reflect in traditional software metrics.

C. Open Source Development and Agile Processes

Industrial software development has shifted towards open-source components and agile processes, diverging from traditional models. Developers contribute to open-source projects, creating specialized adaptations. This raises questions about evaluating developer contributions and measuring product characteristics in such environments.

D. Emergent Uses of Software

Software systems are embedded in evolving environments, including changing development teams and user communities. The internet has accelerated the feedback loop between users and developers, enabling emergent uses of software systems beyond their original design goals. This represents a fundamental shift in software evolution.