Why Componentology is Vital Foundation for real-CBSE

Even elementary school children can be taught to draw pictures of elephants. However, even the most talented artists cannot accurately depict an elephant if they do not know what an elephant is—whether it is a tree, a bird, a fish, a flower, a mammal such as canine or feline, or a landmark building.

Similarly, inventing real software components or discovering the essential methods and mechanisms to assemble them into ideal Component-Based Products (CBPs) is impossible without first acquiring scientific knowledge and an objective understanding of:

- The nature, essential function, and intrinsic properties of physical components,
- The anatomy, structure, design, and construction of physical CBPs, and
- The principles and mechanisms underpinning real Component-Based Engineering (CBE).

Such foundational understanding can only be developed and accumulated through the systematic study of Componentology.

Almost everything currently accepted about so-called software components, CBPs, and CBE is rooted in pseudoscientific dogma. These dogmatic beliefs are based on flawed descriptions, misleading explanations, and unscientific concepts about fictitious/abstract constructs erroneously or deceptively labelled as "components." This widespread dogma is the root cause of the infamous software crisis—manifested in pervasive complexity, unreliability, and the epidemic of spaghetti code.

If someone has no idea what an elephant is, yet wishes to draw one, common-sense dictates that they must first understand what an elephant is. Otherwise, the result will inevitably be a misrepresentation. Analogously, the software research community has long attempted to define, construct, or create real software components, assemble CBPs, and engage in CBE—without first establishing a clear, scientifically grounded understanding of what these concepts truly entail.

In the absence of such foundational knowledge, any attempt to define, construct, or apply them is not only misguided but also perpetuates entrenched misconceptions, as evidenced by the persistent failures and limitations of the prevailing software engineering paradigm.

Consequently, many researchers have inadvertently misled both themselves and the broader field into believing they are working with real components, CBPs, or authentic CBE—when, in fact, they are not. Rather than critically engaging with the empirical evidence and conceptual clarity offered by *Componentology* to address these foundational errors, much of the community continues to ignore, suppress, or even undermine this inconvenient scientific truths/reality.

Today, no one in the software world can answer even simple questions such as (i) what the distinct or striking difference is between parts that are most certainly ideal components and parts that are most certainly not components, and (ii) what the distinct or striking difference is between products that are most certainly ideal CBPs and products that are most certainly not component-based.