# Abstract

Current approaches to software development suffer from some major limitations. Demand for scarce expertise and long development time lead to increased cost, and the impact of such a costly project would be detrimental. Difficulties in managing system evolution also contribute to these problems. Comparing software development process with biological systems formation, we identified lack of automation as one of the root causes for these limitations. With the aim of providing solutions to these limitations, we have developed a novel approach for software development based on a theoretical framework which was derived from biological systems.

Biological systems are living beings which can be described as a unity of structure (anatomy) and functionality (physiology), on to which biological requirements can be mapped. With regard to the formation of biological systems, we have identified four principles: hierarchical organization, Genes storing bioengineering knowledge, evolution and automated formation. A Software system can be compared with biological systems with regard to its mapping of user requirements into system architecture and system functionality. These four principles have been adopted to lay down a theoretical foundation for software system formation.

In applying the first principle, a hierarchy is defined and its nodes are referred to as slots. A slot is a reference point to six categories of slot entities which are a set of functional user Requirement Constructs, a set of non-functional requirements, a Design Template, a set of Design Constructs, a set of software genes and a set of System Constructs. With regard to the hierarchy, two types of relationships are defined, namely, intra-slot and inter-slot, between slot entities. Intra-slot relations describe functional mappings

between the three constructs, which are Requirement Constructs, Design Constructs and System Constructs during process of system formation, which can be thought as two sequential transformations. Firstly, requirement specification is transformed into design specification using Design Templates. Secondly, the design specification is transformed into the system using software genes. Inter-slot relations describe how hierarchies of requirement specification, design specification and system itself are formed by means of combining child constructs to form parent constructs iteratively.

In adopting the second principle, repositories of Design Templates and software genes have been developed. Design templates are derived for families for user requirements by generalization its members. Given a member, its Design Template is capable of transforming it into a Design Construct. Genes are developed by storing general knowledge required to build a System Constructs for members of a requirement family. It includes program instructions which are customizable according to the specific nature of a particular member and also the instructions that are required to generate the parent System Construct by combining its child System Constructs in its hierarchy. Third principle has been adopted to achieve software system evolution in a comparable way to biological systems by changing the composition of genes as the functional and non-functional requirements change. Lastly the adaptation of the fourth principle achieves the automated system generation. This involves identifying and organizing user requirements in a hierarchical form based on Design Templates and transforming it into design specification. Then genes are selected according to non-functional requirements and the design specification is transformed into the system. This theory has been extended to build the novel software process model named as Biological Process Model, and its algorithms have been incorporated into a new CASE tool, GeneSys.

Biological Process Model has two major phases, Knowledge Engineering and System Generation. Knowledge Engineering Phase involves three stages namely Defining System Architecture, Requirement Engineering and Gene Engineering. The Gene Repository and the Design Templates repository are the main outcome of this phase and they will be used during the System Generation phase. Given the condition that required Genes and Design Templates are available, System Generation phase involves generating various software systems. It involves four phases, which are Requirement Analysis, Gene Selection, Automated System Generation and Evaluation.

BPM was evaluated by comparing it against the Rational Unified Process which is considered to be one of the widely accepted development processes. This involved developing a medium size business application using both approaches and estimating the effort using the cost estimation tool CoStar which is based on the COCOMO cost estimation model. Based on the results, we reached the conclusion that BPM requires a much less effort by reusing design and construction knowledge.

While some programming languages would be incompatible with BPM if they do not support hierarchical structures, making a paradigm shift in software development with the new concepts would also be challenging. Despite these limitations, the potentials of the BPM can be increased by further research into areas such as expanding Gene Repository and Design Template repository, multiple programming language support, introducing umbrella activities and enhancing the system evolution.