A SYSTEMATIC APPROACH TO BIO-INSPIRED CONCEPTUAL DESIGN

A Dissertation
Presented to
The Academic Faculty

By

Jamal Omari Wilson

In Partial Fulfillment
Of the Requirements for the Degree
Doctor of Philosophy in Mechanical Engineering

Georgia Institute of Technology

December, 2008

A SYTEMATIC APPROACH TO BIO-INSPIRED CONCEPTUAL DESIGN

Approved by:

Dr. David W. Rosen, Advisor
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Janet K. Allen
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Bert Bras
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. David N. Ku
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Meisha L. Shofner
School of Polymer, Textile, and Fiber Engineering
Georgia Institute of Technology

Dr. Jeannette Yen
School of Biology
Georgia Institute of Technology

Date Approved:

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my Lord and Savior Jesus Christ for not only giving me the ability to achieve this accomplishment. My faith has truly given me calm throughout my storms. To my wife, Lyn Wilson, my parents, Deborah and Roger Wilson, and my brother, Rashad Wilson, I truly thank you for your support and encouragement through my educational pursuits.

I would like to thank my advisor, David Rosen, for his guidance throughout my time as a graduate student. He truly gave me the latitude to explore topics that I found value in and supported me in these pursuits. I would also like to thank my dissertation reading committee, Janet Allen, Bert Bras, Meisha Shofner, Jeannette Yen, and David Ku, for their support and guidance in finishing my dissertation.

To my SRL Family, current and extended, I extend a large hand of gratitude for their intellectual stimulation and camaradarie inside and outside of the lab. I owe a special thanks to several SRL members. To Jeff Olson, I thank you for your collaborative work and his eager ear. I thank John Reap for his research reviews and constructive criticism. To Sungshik Yim and Patrick Chang, I thank you for your help in developing the Strategy Repository. To Chris Williams and Benay Sager, I also thank you for your mentorship and guidance during this process. I also owe a special thanks to Brent Nelson and the Center for Bio-Inspired Design (CBID) for assistance with the psychological studies and furthering my thoughts on the power of bio-inspired design.

Last but not least, this thesis could not have been completed without the financial support of the NASA Harriett G. Jenkins Predoctoral Fellowship Program, the David and Lucille Packard Foundation Fellowship, and the Office of Naval Research Future Engineering Faculty Fellowship.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

GLOSSARY

Analogical reasoning – the cognitive process of transferring knowledge from a source domain to a target domain

Behavior - intrinsic change of state of the attributes of a system

Causal behavioral description - a behavior-based system representation linking structural, behavioral, and functional information.

Function - the effect of a component on the system around it

Knowledge Representation – a cognitive model of reality

Novelty - the degree to which a given design concept is unique relative to other concepts

Ontology – a highly structured system of concepts and relationships between concepts.

Repository – a tool used to capture, store, and retrieve domain knowledge.

Semantic retrieval – an information retrieval strategy based on semantically-rich representations of information

State - the value of the system attributes at a given instant of time.

Strategy - the means by which the behavior of the system is performed

Structure - the entities of interest and the interactions or relations between these entities

Subsumption - A mechanism by which two concepts are compared to determine whether one concept is a more general expression of the other

System - The set of physical components and interactions between these components.

Taxonomy – hierarchical classification of concepts utilizing parent-child relationships between concepts

Variety - the degree to which design concepts can be distinguished from one another.

SUMMARY

In Conceptual Design, the designer is tasked with searching for new and innovative design solutions. This search process, referred to as the designer's design space, has been shown to dictate the quality and effectiveness of the final design solution[1]. According to the theory of bounded rationality, this design space is bounded by the limited cognitive abilities of the designer [2], amongst other factors. To overcome the limitations imposed by bounded rationality, designers often employ several techniques to aid in idea generation, including building upon analogous solutions in the current domain of application. However, designers commonly fail to take to advantage of solutions and practices of other sciences and technologies and/or fail to even recognize the similarities between their technical problems and solutions to similar problems from otherwise alien domains [3]. To address this problem, the following research question is proposed, "*How can we aid the designer in more effective ideation in Conceptual Design?"* In this research, biological strategies are used to aid the designer in Conceptual Design. Through millions of years of research and development, nature has developed efficient and economical solutions to the problems it faces. It is believed that harnessing design strategies from nature will lead to more effective solutions to the engineering problems currently faced by designers. Specifically, the primary hypothesis of this research is as follows:

"Building upon a rich behavioral model of biological systems and a strategy repository, the proposed approaches to Bio-inspired Conceptual Design can be used to aid the designer in (1) identifying relevant biological systems and (2) using biological strategies in Conceptual Design to produce 2a) a larger variety of design ideas (2b) design ideas of greater novelty and (2c) higher quality design ideas".

The fundamental claims of this hypothesis include that of biological system representation, behavioral decomposition, efficient retrieval, and assessing the impact of biological strategies on Conceptual Design.

In this research, it is believed that representations play a key role in bridging the gap between the biological and engineering domains. In the first part of this research, a rich, causal behavioral model was developed for representing the behavior of biological systems. For this purpose, the hierarchical Petri net representation was developed. This representation has the advantage of representing both behavioral refinement and abstraction. The purpose of this behavioral model is to aid the designer in systematically extracting design strategies from biological systems. To ensure consistency in the behavioral model, a systematic method for decomposing and representing the behavior of biological systems, the Method for Reverse Engineering Biological Systems, was developed. This method was found to preserve the fundamental properties of the behavioral systems across hierarchical levels of the representation.

The identification of relevant biological strategies in Conceptual Design is also a key issue in bio-inspired design. Although current approaches are useful in storing and providing access to biological information, the current retrieval strategies often result in either providing too many and/or irrelevant results. In this research, an engineering ontology was developed based on concepts from the hierarchical Petri net representation. This ontology was then encoded into a strategy repository using Description Logic (DL), knowledge representation formalism used for representing domain knowledge and reasoning about it. Subsumption, an inference algorithm in DL for determining if one concept is a member of another concept, was shown to enable both consistent and precise retrieval of biological strategies. When compared to current approaches to representing and retrieving biological strategies, subsumption-based retrieval was found to be more effective.

Next, the constructs of the Method for Reverse Engineering Biological Systems and the strategy repository were synthesized into two distinct approaches to Bio-inspired Conceptual Design: problem-based and solution-driven. In the problem-based approach, the design begins with an engineering system and searches for solutions through engineering design. In this approach, the strategy repository is used to identify relevant biological strategies and stimulate idea generation. To validate the problem-based approach to Conceptual Design, cognitive studies of Mechanical Engineering students and a comprehensive example of the design of hybrid, bullet resistant armor were used.

In this approach, biological strategies were found to increase the novelty of design ideas generated, while also preserving the variety of design ideas generated.

In the solution-driven approach to Bio-Inspired Conceptual Design, the designer begins with a biological solution and attempts to mimic the behavior of this system in the engineering domain. In this approach, the Method for Reverse Engineering Biological Systems is used to systematically decompose the behavior of biological systems and extract behavioral strategies. These strategies are then used as the foundation for the generation of new ideas. To validate the solution-driven approach, historical case studies of bio-inspired systems and a comprehensive example of the development of a novel, renal replacement therapy system were used. In this approach, bio-inspired systems possessing a deeper level of behavioral similarity to their analogous systems were found to perform better than those with less similarity.

CHAPTER 1 INTRODUCTION

Due to increasing globalization, traditional market differentiators such as cost and quality are becoming increasingly irrelevant in today's market. Because of this, corporations must compete on the basis of innovation. A competitive advantage can be achieved by developing products that are technologically distinct from its competitors; one way to accomplish this differentiation is the development of products with original features along meaningful dimensions [4]. Since innovation is the differentiator of the future, engineers and designers alike are being forced into new thought patterns and processes to create novel and innovative solutions.

In the Conceptual Design phase of the engineering design process [5], the designer is tasked with searching for these novel and innovative solutions. However, humans are imperfect search engines [6] and tend to focus on a narrow part of the design space and overlook many valuable solutions [4]. According to the theory of bounded rationality [7-9], an individual is limited by unconscious skills, habits, and reflexes, by values and conceptions of purpose, and by the extent of his/her knowledge of information. Therefore, the space that can be searched is bounded by the limited cognitive abilities of the designer [2].

To overcome the limitations imposed by bounded rationality, designers often employ several techniques to aid in idea generation (i.e, see Section 2.1). According to Pertulla [4], if we assume that creativity is not a static quality, then we also accept the notion that creativity can be learned and that structured techniques can be employed to support the creative process. These techniques aid the designer in expanding and exploring her/his design space more efficiently.

One technique often employed to aid in manipulating the designer's design space is that of analogies. Analogy in design is used to serve as a means of problem solving, whereby designers use solutions from other domains to solve problems in his/her current domain. In using analogies in engineering design, designers often rely upon existing

solutions in the current domain of application, while failing to leverage solutions from outside this domain. Vincent and Mann [6] comment, "When we innovate, we commonly fail to take advantage of the solutions and practices of other sciences and technologies or to recognize the similarities between our technical problems and the solutions to similar problems in otherwise alien technologies". In this research, we explore nature as a possible source of analogies, whereby strategies from the biological domain are used to solve problems in the engineering domain.

1.1 NATURE AND BIO-INSPIRED DESIGN

1.1.1 Why Nature?

Nature's design process, evolution, is enacted by natural selection. Natural selection can be summarized as the process by which favorable characteristics become more prevalent in successive generations of an organism. In essence, "design in nature is a process of generating variability and then selecting the variants that are favorable[10]". At the fundamental level, nature is driven by this survival of the fittest, where energy efficiency is key. It should also be noted that development in nature is done so in the context of its environment, meaning that systems evolve differently in different environmental contexts.

Through natural selection, nature has undergone millions of years of research and development. "Through evolution, nature has experimented with principles of physics, chemistry, mechanical engineering, material science, mobility, control, sensors, etc."[11] Through this, nature has created effective solutions to many problems faced by both natural and engineering systems. Table 1.1 summarizes many differences between solutions found in natural and engineering systems.

Table 1.1 Common Characteristics of Natural and Engineering System[10]

Natural systems are typically:	Engineering systems are typically:
• Made from **fewer components** whose properties **vary internally**	• Made from individual **homogenous** components
• Concerned with **strength**, making their materials **tougher**	• Concerned with **stiffness** in engineering materials, making them more **brittle**.
• Built from **fibrous** composites	• Built from **metals and alloys**
• **Adaptive**, meaning they adapt to varying inputs such as loads and environmental changes over different time scales	• **Maladaptive** and **overdesigned**
• **Multifunctional**, dedicating multiple functions to a single component (integrated)	• **One-to-one** functional mapping
• Arranged **hierarchically**, having many size scales and levels of organization	• **Non-hierarchical**, design confined to one size scale

As shown above, when compared to typical engineering systems, natural systems show many distinct differences. This is largely due to the differing developmental context and constraints on these systems. However, there are limitations to this evolutionary change, including growth (change in shape) constraints, reproduction, information shortages, resource limitations, incremental change, and evolutionary history [10].

Why nature as a source for analogs? Although natural systems have a different developmental context and constraints, the duality between these systems and engineering systems still exists in the search for economical solutions to problems. Since nature has spent millions of years developing these economical solutions, it seems rational that engineers should take a look to nature for energy-efficient answers to similar problems raised by their technologies [12]. "By adapting mechanisms and capabilities from nature, scientific approaches have helped humans understand related phenomena and associated principles in order to engineer novel devices and improve their capability"[11]. This concept of borrowing ideas from nature was originally coined as bionics by Jack Steele of the US Air Force in 1960 at a meeting at Wright-Patterson Air

Force Base in Dayton, OH [13]. Otto Schmitt [14] later used the term Biomimetics to describe the field. Biomimetics can be seen as the transfer of these natural technologies to other domains, such as engineering, chemistry, materials, etc. Other synonymous terms include biomimesis, biomimicry, biognosis, and bio-inspired design. In this research, the term bio-inspired design is utilized. Bio-inspired design is defined in this research as the transfer of design strategies used in the natural domain to the engineering domain.

1.1.2 Motivating Examples

There have been several successful examples of designers using natural systems to inspire engineered systems. Figure 1.1 displays several bio-inspired products that serve as motivating examples for this research.

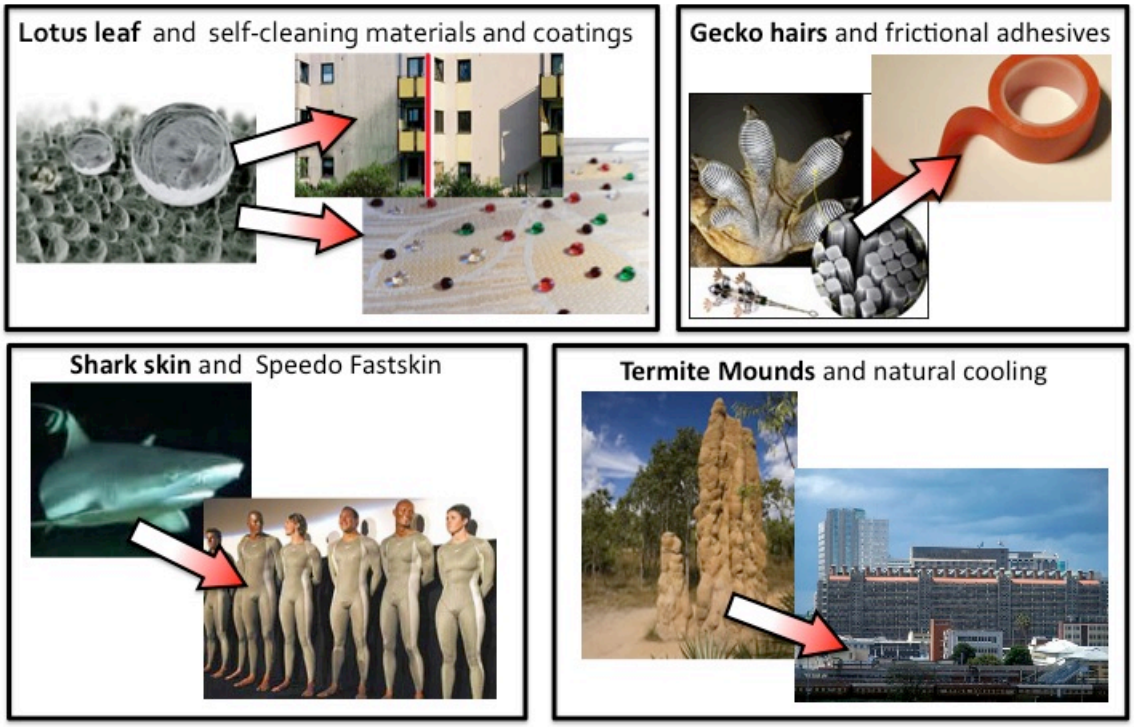


Figure 1.1 Motivating examples of Bio-Inspired Design

The motivating examples from Figure 1.1 are reviewed in the following discussion.

Lotus leaf – Lotusan by Sto Corporation® is a self-cleaning exterior coating that was inspired by the Lotus effect exhibited by the lotus leaf. Despite growing in muddy and dirty conditions, the surface of the lotus leaf makes water bead up and runoff, taking dirt with it. These coating have been shown to reduce maintenance costs of building painted with the self-cleaning coating.

Gecko tape – Dry adhesive tape inspired by the adhesive mechanism of gecko feet. The gecko uses millions of microscopic hairs on the pad of their feet, with each hair providing a Van der Waals attractive force. These hairs allow the gecko feet to bond to just about any surface. The advantage of this type of dry adhesive tape is reusability.

Olympic swimwear – Speedo® created full body swimwear for the 2004 Olympic Games, named Fastskin, inspired by the surface ridges on shark skin. These ridges reduce passive drag by up to 4 percent more than the next best swimsuit. Swimmers wearing Speedo Fastskin won 80% of the swimming medals in the 2004 Olympic Games and broke 13 out of 15 world records.

Natural Cooling – Inside the termite mound in Zimbabwe, termites farm a fungus as their only food. The fungus must be kept at exactly 87 degrees, while temperatures on the outside of the mound range from 35 degrees at night to 104 degrees during the day. The Eastgate Shopping Centre in Harare, Zimbabwe, designed on the principle of natural cooling based on the termite mounds, uses 90% less energy than a conventional building of equal size [15].

As can be seen in the above examples, leveraging biological technologies in the engineering domain can lead to many technological innovations and novel products. In the section, motivation for bio-inspired design was put forth. In the following section, we review current approaches to bio-inspired design.

1.2 CURRENT RESEARCH ON BIO-INSPIRED DESIGN

1.2.1 Key areas of bio-inspired design research

Bio-inspired design research can be divided into 4 key areas: biological/engineering research, representation, analogical translation, and design utilization. These key areas, displayed in Figure 1.2, span the spectrum of research from research strictly on understanding biological phenomena to development of engineering systems.

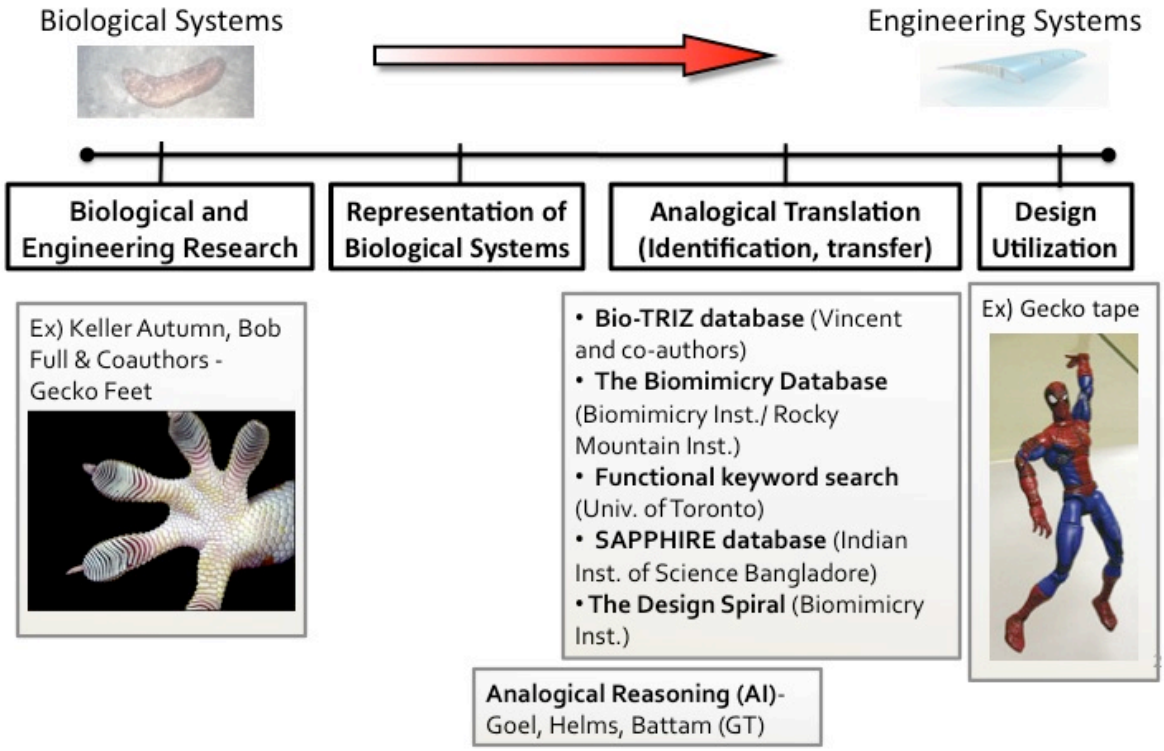


Figure 1.2 Current Approaches to Bio-Inspired Design

Biological and engineering research involves research being performed on biological systems by biologists and engineers. This research is typical of that done in the biological domain, where biological systems and their associated behaviors are studied and characterized. An example of this type of research includes research on gecko feet (Figure 1.1) where the physical phenomena driving gecko adhesion is studied.

Representation of biological systems involves creating models of biological systems that aid in transferring principles from the biological system to the engineering

domain. This research includes the SBF models utilized in research by Goel and coauthors [16-20]

Analogical translation involves research on identification and systematizing the transfer of biological principles to the engineering domain. These approaches include the Bio-TRIZ [12, 3, 13], the Biomimicry Database [21], the Functional keyword search [22, 23], the SAPPhIRE database [24], The Design Spiral [25], and cognitive research by Goel and coauthors [20].

Design utilization involves developmental research on products that utilize principles from the biological domain. Examples of design utilization include the examples displayed in Figure 1.1, where bio-inspired products were developed.

Representation and analogical translation are directly relevant to this research and are given a more thorough review in Section 1.2.2.

1.2.2 Research Problem and Current Approaches

Due to its inherent difficulties, bio-inspired design has thus far followed an ad hoc path. These difficulties include (1) the large analogical distance, (2) lack of cross-domain knowledge, and (3) identification of relevant strategies. Analogical transfer in bio-inspired design is difficult due to the large analogical distance between the natural and engineering domain. The large distance often makes it difficult to draw parallels between the solutions found in nature and the problems of the engineering domain. The lack of cross-domain knowledge of engineers and biologists also makes bio-inspired design difficult. Currently, those studying the novel phenomena found in nature (biologists) know very little about the implications of these phenomena in the engineering domain. Vice versa, engineers and designers know very little in studying and understanding novel biological phenomena. These difficulties are compounded by the difficulty found in accurately identifying relevant biological solutions from the vast number of solutions found in nature.

Several researchers are developing systematic approaches for leveraging biological examples in engineering design. These approaches include the IDEA-

INSPIRE Database [24], the Functional Database [26], the Functional Keyword Search [22, 23], Bio-TRIZ [12, 3, 13], the Design Spiral [25], the Biomimicry Database [21], and the cognitive research by Vattam and coauthors [20].

IDEA-INSPIRE Database - Researchers at the Indian Institute of Science at Bangalore [24] are developing a searchable database containing natural and artificial systems. These researchers present a generic causal behavioral model, called SAPPhIRE, for representing the behavior of these systems and implement the model using software. The behavioral model is represented using natural language format using nouns, verbs, and adjectives. The database, as well as the retrieval process, is implemented in a software package called IDEA-INSPIRE. The SAPPhIRE causal model is used to enter information into the database in both natural language and computer-interpretable form. Design problems are described as the action required to be fulfilled using a verb(V)-noun(N)-adjective/adverb (A) triplet. To retrieve solutions, the user can both browse entries in the database for random stimulation, as well as systematically search through the entries. When the design problems are given in the V-N-A form, the software matches them with the selected variables from the computer-interpretable form of the entries. The matched entries are then sorted in descending order of importance by the degree of matching found between the variables. The degree of matching is determined using weights of 32, 16, and 8 for verb, noun, and adjective/adverb matches, respectively. Synonyms are given weights of 4, 2, and 1, respectively. The entries are then sorted based on the match score received.

Functional Database - Bruck and co-authors [26] sought to develop a repository to provide students with easy access to bio-inspired design concepts and products. The research approach includes 3 key components: (1) functional description templates, (2) repository of bio-inspired concepts and products, and (3) search tools. Bio-inspired designs and concepts are entered into the repository using functional description templates. The functional description template provides a predetermined sentence structure as a template to record more complete functional descriptions from the user. These templates utilize both freeform text and menu selection agents to aid in entry. Functions are described using actions (intended behaviors of the biological concepts or

products), entities, and properties (characteristics of the behavior and structure). Retrieval of these entries is performed using three search tools: (1) keyword search, (2) category filters, and (3) function search. In the keyword search, retrieval is performed using a string matching algorithm combining both equivalence and similarity methods. The category filter is then used to filter the results on the basis of product type, biological type, and development stage. The function search utilizes the keyword search algorithms to search for functions.

Functional Keyword Search - As opposed to using user-populated repositories, researchers at the University of Toronto utilize keyword searches through biological resources already in natural language format (ie. biological texts, journals, etc.) to locate novel biological phenomena. To aid retrieval, the authors utilize a language framework called Wordnet® [27]. Wordnet is an electronic lexical database that organizes word entries based on relations to other words, as opposed to alphabetical order. To use this specialized search procedure, the user first identifies the applicable functional keywords to search the database. These functions are represented as verbs. Wordnet is then used to identify bridge words, which are troponyms and hypernyms used as alternative keywords, to broaden the search. These keywords are then used to search through biological text for related biological phenomena. The search program then quantitatively determines dominant biological phenomena through examining the frequency of collocated words within a 50 word window around the keyword. The top 5% of biological phenomena are then identified and retrieved in the form of passages from the text. The user then sorts through these results manually for relevancy.

Bio-TRIZ - Using TRIZ [28], a systematic method for inventive problem solving developed by Russian researcher Genrich Altschuller, Vincent and Mann [3] introduce a systematic means for drawing functional parallels between natural and engineering systems. Using TRIZ, design problems are characterized as a pair of conflicting characteristics. These characteristics are assigned based on 39 contradiction features defined in TRIZ. The contradictions of the design problem are then matched to solution strategies from previously solved problems; these previously solved problems come from examination of more than 3 million patents. The TRIZ Contradiction Matrix is used to

match the contradictions of the design problem to innovative solution strategies. The researchers [12, 3, 13] seek to integrate knowledge from nature into the existing TRIZ database, aiding access to biological solutions as well as engineering solutions. In the newly-termed Bio-TRIZ database, users are allowed to search for biological phenomena based on the determined contradictions or the function of the design problem. The project has since been discontinued and the database has been taken offline.

Biomimicry Database - The Biomimicry Database [21] is a joint project between the Rocky Mountain Institute and the Biomimicry Guild. The Biomimicry Database utilizes a searchable database of biological information to identify biological analogs. The database contains six types of searchable information, including: challenges (problems that need to be solved), strategies (potential solutions to the challenges), organisms (associated biological system), people (person/user records), citations, and products (associated bio-inspired products). The user uses keywords to search across all database records. This project was in alpha testing, but the database is no longer supported on the Biomimicry Guild website.

Design Spiral - The Design Spiral [25] is a design process for bio-inspired design that includes 6 steps (Identify, Translate, Observe, Abstract, Apply, and Evaluate) performed in an iterative fashion. In the Identify step, the designer develops a design brief of the human need and defines the specifics of the problem. In the Translate step, the designer biologizes the question, asking "How does nature do this function?" The third step of the Design Spiral is the Observe step, where the designer is asked to look for champions in nature who can answer the question. The Observe step is followed by the Abstract step, where the designer finds repeating patterns and processes within nature and abstracts the strategy. In the fifth step, Apply, the designer develops ideas and solutions based on the natural models found in the previous step. Lastly, in the Evaluate step, the designer compares the ideas with successful principles found in nature. If needed, the designer repeats the process until a final solution is devised.

Cognitive modeling and SBF modeling in BID – The goal of the work by Goel and coauthors [20] is the cognitive modeling of the bio-inspired design process and representation of biological behavior. Specifically, the authors observed sessions in a

Bio-Inspired Design course as well as conducted cognitive studies. From the study, the authors found that students arrived at bio-inspired design solutions using two distinct approaches: a problem-driven and a solution-driven approach. They also found that once a biological solution was selected, the remainder of the design process was constrained due to fixation. The authors are continuing this work in an attempt to understand the cognitive basis for bio-inspired design and develop tools to support the process.

In this section, relevant research in representation and analogical translation in bio-inspired was reviewed. In Section 1.2.3, research opportunities in bio-inspired design are identified.

1.2.3 Research Opportunities

Several opportunities exist to further the present state of research in bio-inspired design. These opportunities include biological representations to bridge the gap between biological and engineering domains, systematic method to guide decomposition and strategy extraction, efficient retrieval of relevant solutions to aid in identifying relevant solutions, and providing empirical evidence to support bio-inspired design.

1.2.3.1 Biological Representations

In the reviewed approaches, a significant gap exists between the biological and engineering research being performed by biologists in their respective fields and the analogical translation that aims to transfer knowledge from biology to engineering. In this research, it is believed that this gap can be filled with systematic representation of biological systems. Representations are essential to aiding people in understanding phenomena, especially when these phenomena cannot be experienced directly. These representations are used to aid the engineer in understanding relevant biological systems, then transferring this knowledge to engineering. Current approaches to biological system representation to aid in transferring knowledge from nature to engineering include function structures and SBF models. Vakili and co-authors [29] use function structures to represent biological phenomena, whereas Goel and coauthors [20] use SBF models to represent biological phenomena. However, these models lack the theoretical rigor needed for consistent representation of biological systems.

1.2.3.2 Systematic method

Biological phenomena in bio-inspired design are vulnerable to misunderstanding and misinterpretation, especially by novices to biology [29]. The extraction of the correct strategy from biological phenomena is a difficult task and misinterpretation can lead to the extraction of incorrect and incomplete strategies [29]. After reviewing the current approaches, a research opportunity can be identified in the area of biological behavior decompostion and strategy extraction. To aid in extraction of these strategies, a systematic method for decomposing the biological system behavior is needed. Along with biological system representation, this systematic method can be used in bridging the gap between the biological and engineering domains. Vakili and co-authors [29] give some consideration for strategy extraction using function structures. However, the authors fail to define a systematic and consistent method for defining these structures and extracting strategies.

1.2.3.3 Retrieval of relevant solutions

Efficient identification of relevant biological strategies to use in Conceptual Design is key to harnessing biological technologies in engineering, however, this identification is one of the most difficult tasks in bio-inspired design. Current approaches to identifying relevant biological strategies include using searchable databases of biological solutions [12, 3, 24, 26, 13] and a functional keyword search [22, 23] through biological literature. Although these approaches offer access to these biological solutions, the generic keyword-based retrieval mechanisms utilized by these approaches often suffer from providing either too many and/or irrelevant results [30].

1.2.3.4 Lack of empirical evidence to support bio-inspired design

Although significant advantages of bio-inspired have been theorized by examining scattered examples of successful cases, there has been a lack of research on how the use of biological strategies in engineering impacts the designer and the products that follow. With respect to the current approaches for systematizing bio-inspired design, there is a definite lack of empirical evidence on the value of these methods, and bio-inspired design as a whole.

In this section, research opportunities in bio-inspired design were identified. In the following section, Section 1.3, a plan of action for addressing these gaps is put forth.

1.3 RESEARCH OVERVIEW

1.3.1 Research Approach

This research is motivated by the notion of "effective" idea generation. The goal of this research is to aid the designer in generating ideas in Conceptual Design through biological strategies. Idea generation is bounded by the limited cognitive abilities of the designer. However, in this research, it is believed that these limitations can be overcome through the use of biological strategies. Biological strategies are viewed as refinements of behavior, where specific physical phenomena driving a particular behavior (and function) are identified. For example, take one view of the behavior of the human kidney, which is "filtration of the blood". The strategy gives the physical phenomena that drive this behavior. In this case, the strategy can be stated concisely as "glomerular diffusion and convection of substances from the blood followed by reabsorption and secretion of substances across the membrane and ending with the excretion of unwanted substances through the collecting duct".

In this research, there are two contexts in Conceptual Design in which bio-inspired design is utilized [20]. In the first context, problem-based Conceptual Design, the designer begins with an engineering problem and seeks to develop a solution to this problem. In this context, it is advantageous for the designer to navigate as broad a design space as possible, as this increases the likelihood of a "winning" solution. In this context, bio-inspired design is used in ideation as a means of aiding in the expansion and exploration of design space.

In the second context, solution-driven Conceptual Design, the designer takes a reverse-engineering approach to solving an engineering design problem. The main purpose of reverse engineering is to (1) identify the biological system's components and relationships between those components and (2) represent the system in another form or

higher level of abstraction [31]. In the solution-driven context, the general approach of reverse engineering is applied to biological systems.

1.3.2 Primary Research Question and Hypothesis

Several researchers have developed approaches for identifying and transferring biological strategies to the engineering domain. However, several shortcomings have been identified, including the lack of research on representing biological systems so that strategies can be easily accessed and comprehended, lack of a systematic method for extracting biological strategies, inefficient identification of these strategies, and a lack of empirical evidence on the advantage of these biological strategies in Conceptual Design. To address these gaps, in this research, the following question is put forth:

Primary Research Question: How can we aid the designer in more "effective" idea generation in Conceptual Design?

To answer this question, the following hypothesis is proposed:

Primary Research Hypothesis:
Building upon a rich behavioral model of biological systems and a strategy repository, the proposed approaches to Bio-inspired Conceptual Design can be used to aid the designer in (1) identifying relevant biological strategies and (2) using biological strategies in Conceptual Design to produce 2a) a larger variety of design ideas (2b) design ideas of greater novelty and (2c) higher quality design ideas.

The fundamental claims of this hypothesis include that of biological system representation, behavior decomposition, efficient retrieval, and assessing the impact of biological strategies on Conceptual Design. To validate these claims, several sub-research questions are proposed in the following discussion.

Biological Representation

The backbone of the proposed method is that of representation of biological systems. When designers are unable to experience phenomena directly, representations play a key role in understanding physical phenomena. Representations help to filter out

unimportant information and present the designer with information relevant to the given task. Developing accurate representations is of critical importance, and must be presented in a manner that helps the user reason about the system[32], especially in the field of bio-inspired design. Representing and understanding biological phenomena is especially difficult for several reasons, including:

- Biological systems are complex, interactive systems ranging in scales [33]
- Most biological systems are embedded invisibly throughout the body and their functions hidden from view [33]
- Lack of cross-domain knowledge by biologists and engineers.
- Differing motivations. For example, biologists seeks to understand nature while engineers seek to generate designs for new problems [20].
- Differing methodologies and representations

In this research, it is believed that representation of biological systems is key to systematizing bio-inspired design. In essence, representations are used to bridge the gap between the biological and engineering domains, as displayed in Figure 1.3.

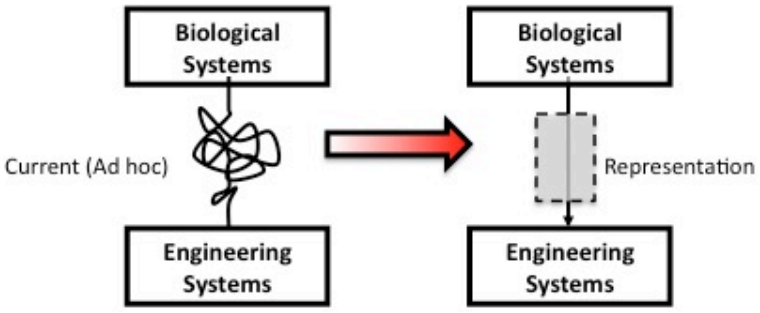


Figure 1.3 Representation Gap

As seen in the figure, representation is used to bridge the gap in the analogical translation between biological and engineering systems. In general, generic representations have three key components: form, behavior, and function. To support conceptual design and ideation, "it is now a consensus that design information should include not only the physical structure of a design, but also its required function and implementing physical behavior"[34]. Representations are reviewed in Section 2.2.

This motivation leads to the following question:

***Question 1*:** *"What type of representation can be used to model the behavior of biological systems?"*

To answer this question, it is hypothesized that:

Hypothesis 1: A representation based on (1) a causal behavioral description and (2) hierarchical Petri nets can be used to model the behavior of biological systems

Systematic Decomposition

The purpose of a behavioral model is to aid the designer in extracting the biological strategy from the system. With that, the system should be represented in a manner that the designer can visualize the behavior of lower entities (strategy) and how it affects the overall behavior of the system. The representation should also allow the behavioral relationships across different levels of abstraction of the biological system to be modeled. Therefore, a systematic method for decomposing biological system behavior is sought to ensure that the behavior is consistent across these levels of hierarchy. To ensure consistency, three fundamental properties of Petri nets are considered: reachability, liveness, and boundedness. These properties are reviewed in Section 2.3. This leads to the following question:

Question 2: *How can the behavior of biological systems be hierarchically represented using Petri nets, while preserving the fundamental properties at each hierarchical level?*

In answering this question, the following is hypothesized:

Hypothesis 2: Using the systematic method for Reverse Engineering Biological Systems will ensure that the fundamental properties of boundedness, reachability, and liveness will be preserved across hierarchical levels.

Efficient Retrieval

Identifying relevant biological strategies in Conceptual Design is a key issue in bio-inspired design. Current approaches are useful for storing and providing access to biological information, however, the generic keyword-based retrieval process utilized by these approaches often suffers from providing too many and/or irrelevant results [30]. It

is believed that by structuring biological information using ontologies, biological strategies can be more accurately and efficiently retrieved. Engineering ontologies and retrieval algorithms are reviewed in Section 2.4. With respect to retrieval of biological strategies, the following question is asked:

Question 3: *How can hierarchical Petri net representations of biological systems be structured to aid retrieval of relevant strategies from a knowledge repository?*

To answer this question, it is hypothesized that

Hypothesis 3: An ontology of concepts from hierarchical Petri net representations of biological systems can be represented using Description Logics. Subsumption in Description Logics will enable consistent and precise retrieval of relevant biological strategies from a knowledge repository.

Assessing the impact of biological strategies on Conceptual Design

Although there have been several approaches researched for systematizing bio-inspired design, there has been very little research in quantifying its impact on the designer in Conceptual Design. In the problem-driven Conceptual Design approach, the designer's design space is of primary importance. Design space can be viewed as the space of all possibilities for a given problem. Shah et al. [35] put forth metrics for evaluating the design space of a designer in idea generation using outcome-based methods (See Section 2.4 for further discussion). These metrics can be used to evaluate idea generation using biological strategies. In the solution-driven approach, extraction of design strategies is of primary importance. This research aims to show the value of rich behavioral representations of biological systems, such as the hierarchical Petri net representation. To do so, the performance of bio-inspired designs with deep similarity can be compared to those possessing only superficial similarities. In this case, positive correlation between performance and level of similarity will show value in richer representations of biological system behavior in Conceptual Design. This leads to the following question:

Question 4: *What is the impact of biological strategies in the Conceptual Design process?*

In answering this question, the following is hypothesized:

Hypothesis 4: *4(a)* Exposure to biological strategies will increase the novelty of design ideas generated and *4(b)* will increase the variety of design ideas generated. Additionally, *4(c*) bio-inspired engineering systems possessing a deeper level of biological system behavior will perform better than those possessing superficial behavioral similarities.

In this section, the research questions driving this research were presented. In Section 1.3.3, an overview of how this research is organized within this dissertation is presented.

1.3.3 Research Brief and Dissertation Roadmap

The goal of this research is to aid the designer in generating ideas in Conceptual Design through the use of biological strategies. First, a hierarchical representation of biological systems and a method extracting behavioral strategies from these representations, termed the Method for Reverse Engineering Biological Systems, is developed. Next, using this representation, an ontology is developed to aid in retrieving relevant biological strategies from a strategy repository. The representation and ontology are then structured into two approaches for Bio-inspired Concept Generation, the problem-based and solution-driven approaches. These approaches are then validated using example problems. Specifically, the problem-based approach is validated using cognitive studies and an illustrative example of the development of hybrid bullet-resistant armor. The solution-driven approach is validated using case studies and an illustrative example of a wearable artificial kidney.

The overall research plan and its relation to the organization of this dissertation are displayed in Figure 1.4.

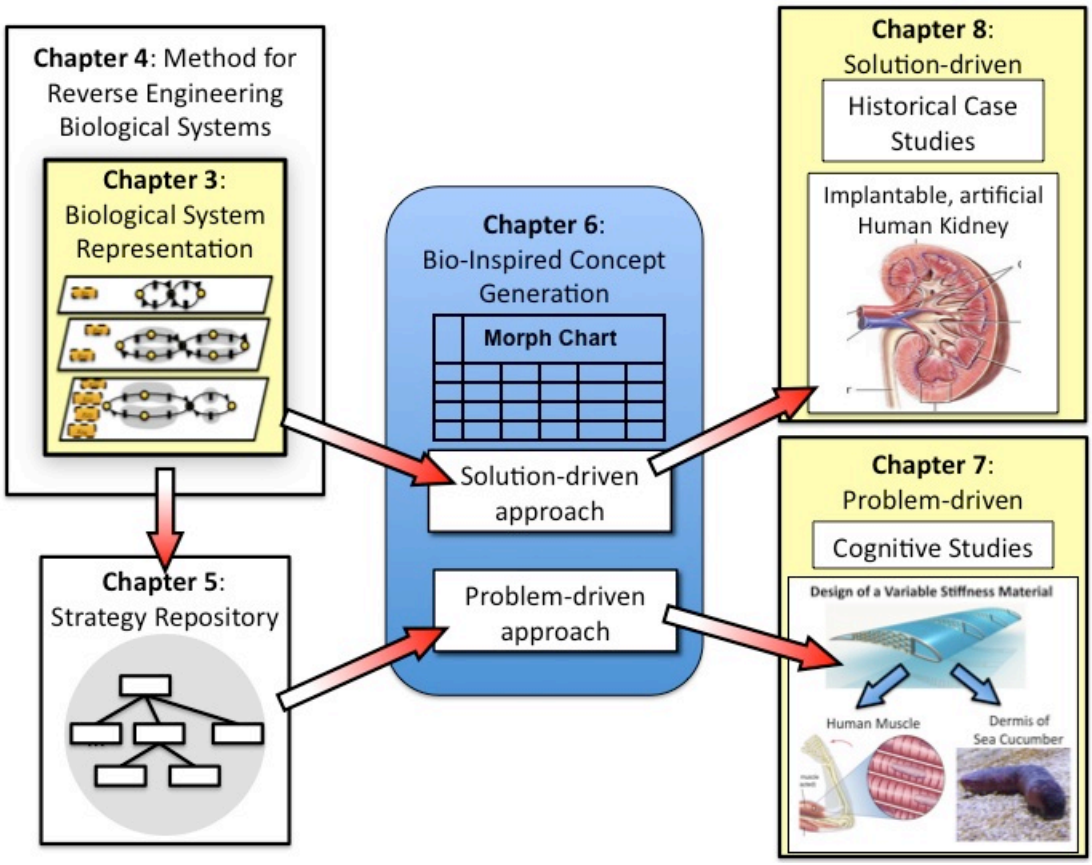


Figure 1.4 Research Organization

The organization of this dissertation is described in the following.

In Chapter 2, the theoretical foundation upon which this research is built is presented. This foundation includes that of systematic design and idea generation techniques (Section 2.1), representations in engineering design (Section 2.2), design repositories and semantic retrieval (Section 2.3), and metrics for evaluating idea generation (Section 2.4).

In Chapter 3, the backbone of the method for Reverse Engineering Biological Systems, the hierarchical representation for biological systems, is developed. This includes elaborating requirements for the representation and selecting the most suitable representation.

The Method for Reverse Engineering Biological Systems is presented in Chapter 4. In Section 4.1, the method for generating the hierarchical representation is presented, followed by systematic steps in Section 4.2. Examples of the proposed method are presented in Section 4.4.

In Chapter 5, the strategy repository is developed. In Sections 5.1 and 5.2, the ontology for representing biological information is developed. In Section 5.3, the software implementation of the strategy repository is described. The retrieval mechanism used to efficiently retrieve strategies from the repository is presented in Section 5.4, followed by validation of the retrieval strategy in Section 5.5.

Chapter 6 presents the proposed approaches for bio-inspired Conceptual Design. The problem-based approach is presented in Section 6.2, followed by the solution-driven approach in Section 6.3.

Chapters 7 and 8 deal with validation of the proposed approaches to bio-inspired concept generation. Specifically, validation for the solution-driven approach is presented in Chapter 7 with case studies (Section 7.1) and an illustrative example on the design of a hybrid, bullet resistant armor (Section 7.2). In Chapter 8, validation for the problem-based approach is presented with cognitive studies (Section 8.1) and an illustrative example on the design of a wearable, artificial kidney (Section 8.2).

In the final chapter, Chapter 9, the research questions and their respective hypothesis are revisited. The specific contributions to the body of knowledge on bio-inspired design are also reviewed in this chapter.

1.4 VALIDATION PLAN

The validation and verification strategy in this thesis is two fold. The first strategy addresses the verification of the hypotheses proposed to answer the secondary research questions proposed in Section 1.3. The second strategy involves the validation of the proposed method for Reverse Engineering Biological Systems.

1.4.1 Validation Plan for Hypotheses

In this dissertation, four hypotheses are proposed to address the secondary research questions presented in Section 1.3.2. These hypotheses are presented with the validation tests in

Table 1.2.

Table 1.2 Hypothesis Validation Strategy

Research Question	***Hypothesis***	***Tests***
RQ1: *What type of representation can be used to model the behavior of biological systems?*	**Hyp1:** A representation based on (1) a causal behavioral description and (2) hierarchical Petri nets can be used to model the behavior of biological systems	- Qualitative evaluation with respect to requirements put forth for representation of biological systems
RQ2*: How can the behavior of biological systems be hierarchically represented using Petri nets, while preserving the fundamental properties at each hierarchical level?*	**Hyp2**: Using the systematic method for Reverse Engineering Biological Systems will insure that the fundamental properties of boundedness, reachability, and liveness will be preserved across hierarchical levels.	Find mathematical evidence of boundedness, reachibility, and liveness for hiearchical Petri net representation
RQ3*: How can hierarchical Petri net representations of biological systems be structured to aid retrieval of relevant strategies from a knowledge repository?*	**Hyp3:** An ontology of concepts from hierarchical Petri net representations of biological systems can be represented using Description Logics. Subsumption in Description Logics will enable consistent and precise retrieval of relevant biological strategies from a knowledge repository.	- Find mathematical evidence of consistency through subsumption - Evaluate retrieval precision in various scenarios using test queries
RQ4*: What is the impact of biological strategies in the Conceptual Design process?*	**Hyp4(a,b):** Exposure to biological strategies will increase the novelty of design ideas generated and will increase the variety of design ideas generated	- Cognitive studies on mechanical engineering students - Problem-based Conceptual Design Example (Design of Hybrid Bullet-Resistant Armor)
RQ4*: What is the impact of biological strategies in the Conceptual Design process?*	**Hyp4(c):** Bio-inspired engineering systems possessing a deeper level of biological system behavior will perform better than those possessing superficial behavioral similarities.	- Historical case studies on bio-inspired design - Solution-driven Conceptual Design Example (Design of a wearable, artificial kidney)

Since the focus of this dissertation is the proposed method for Reverse Engineering Biological systems and its accompanying hierarchical Petri net representation, validation of the method as a whole is performed using the Validation Square[36]. This validation strategy is presented in Section 1.4.2.

1.4.2 Intro to Validation Square

A significant part of this work lies in the validation strategy for the method for reverse engineering biological systems. For validation, the Validation Square proposed by Seepersad, et al. [36] is used. In this work, the authors believe that validation in engineering design, because it is based largely on designers' subjectivity, cannot be pursued in formal, rigorous, quantitative verification [37]. Instead, "knowledge validation becomes a process of building confidence in its usefulness with respect to a purpose". The usefulness of a method is associated with whether the method provides design solutions 'correctly' (structural validity) and whether it provides 'correct' design solutions (performance validity). Structural validity involves a qualitative assessment and performance validity involves a qualitative assessment of the proposed method. The Validation Square is displayed in Figure 1.5.

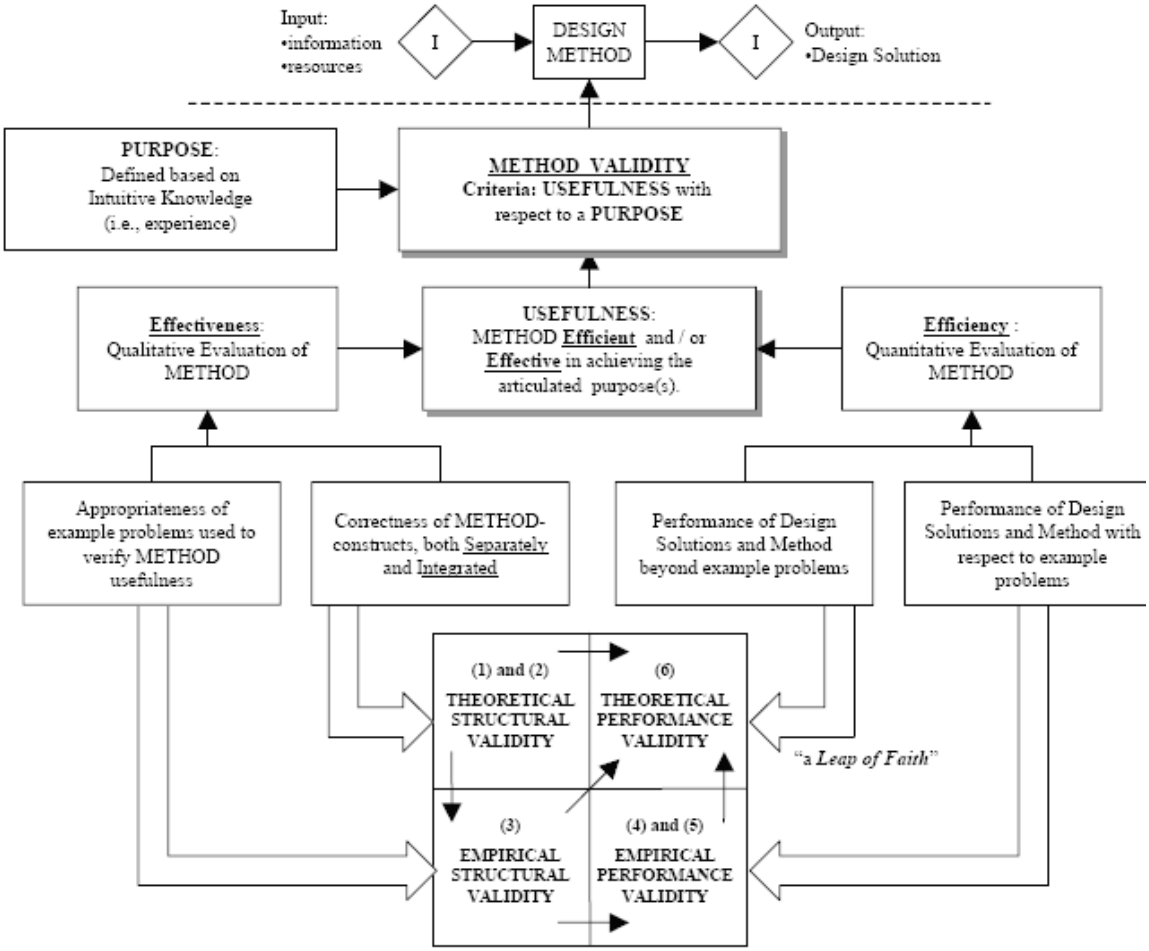


Figure 1.5 Validation Square [36]

As seen in Figure 1.5, there are four aspects to the Validation Square: Theoretical Structural Validation, Empirical Structural Validation, Empirical Performance Validation, and Theoretical Performance Validation. **Theoretical Structural Validation** involves checking the individual constructs and assumptions upon which the method is built, as well as checking the internal consistency of the method when combining the individual constructs. **Empirical Structural Validation (ESV)** includes building confidence in the appropriateness of the example problems used for verifying the usefulness of the method. **Empirical Performance Validation (EPV)** includes building confidence in the 'usefulness' of the proposed method with respect to the example problems. **Theoretical Performance Validation (TPV)** involves building confidence in

the ability to extend the proposed method beyond the scope of the example problem to a general class of problems.

Theoretical Structural Validity:

The first step in validating the method for Reverse Engineering Biological Systems is evaluating the theoretical structural validity. In Chapters 2 and 3, relevant literature on systematic design and knowledge representations is reviewed. The specific construct of the method, hierarchical Petri nets, is reviewed and tested for consistency in Chapter 3. The constructs of the strategy repository, engineering ontologies and Description Logics, are reviewed in Chapter 2.

Theoretical Structural Validity also involves examining the consistency of the method. Consistency is checked to ensure that sufficient information is available to execute the steps. This consistency is checked through describing the tasks needed for each step, as well as the inputs needed and outputs generated. In this research, a flowchart of the systematic steps of the proposed method will be used to check internal consistency. The method consistency is addressed in Chapter 4.

Empirical Structural Validity:

Empirical Structural Validity includes accepting the appropriateness of the example problems that are used to verify the method performance. In this research, the method for Reverse Engineering is used in both a problem-driven and the solution-based conceptual design context (Chapter 6). An example problem is developed for each context. The example problems used are (1) the development of a hybrid bullet resistant armor (problem-based) and (2) the design of a wearable, artificial kidney (solution-driven). These examples, presented in Chapters 7 and 8, respectively, fall within the scope of use of the method.

Cognitive studies of designers in ideation and historical case studies are also used to validate the method. In the context of problem-based Conceptual Design, cognitive studies (Chapter 7) on designers are used to evaluate the novelty and variety of design ideas generated in idea generation using biological strategies. Historical case studies are

used to evaluate the impact of rich behavioral models (ie. hierarchical Petri nets) in the solution-driven Conceptual Design context. These case studies are presented in Chapter 8.

In Chapter 5, the strategy repository test-bed is structured and tested. This test-bed is used to test the precision of subsumption-based retrieval method. This test-bed mimics the envisioned repository, and is deemed appropriate for testing the retrieval method.

Empirical Performance Validation

Empirical Performance Validation involves accepting the usefulness of the method for some representative example problems. In essence, we are evaluating whether or not the method is doing what it has set out to do. In this research, we wish to aid the designer in generating ideas in Conceptual Design using biological strategies. Specifically, we wish to aid the designer in generating a large variety of novel, quality concepts. This is indeed useful as the success of final products can be directly linked to this idea generation process. The usefulness of the method will be assessed in both the problem-based and solution-driven context using the example problem.

For the problem-based context, the proposed method is tested using cognitive studies and an example problem of the design of hybrid, bullet resistant armor. In the cognitive studies, the results are evaluated on the basis of the novelty and variety of design ideas generated. The results from participants receiving biological strategies are compared to those of students not receiving any stimuli and those receiving engineering strategies. In the example problem, the designs generated using the proposed method are compared to the armor currently found in the market. This work is presented in Chapter 7.

For the solution-driven context, the proposed method is tested using historical case studies and an example of the design of a wearable, artificial kidney. In the case studies, the correlation between the performance of bio-inspired designs and the level of similarity with the biological system is evaluated. The results from the example problem

are compared to existing renal replacement therapies found in the market. This work is presented in Chapter 8.

With respect to the strategy repository, the precision of subsumption-based retrieval is tested using test queries. These queries are used as representative retrieval scenarios. The results will then be compared to the performance of current retrieval methods used in bio-inspired designs. The strategy repository is evaluated in Chapter 5.

Theoretical Performance Validation

Theoretical Performance Validation involves building confidence in the generality of the method, in its usefulness beyond the example problems. Success in the previous validation steps helps to build a case for this generality. Although we can make a case for generality, every validation strategy relies on ultimately on a "leap of faith"[36]. Theoretical Performance Validity is addressed in Chapter 9.

The validation strategy proposed in this dissertation is summarized in Figure 1.6.

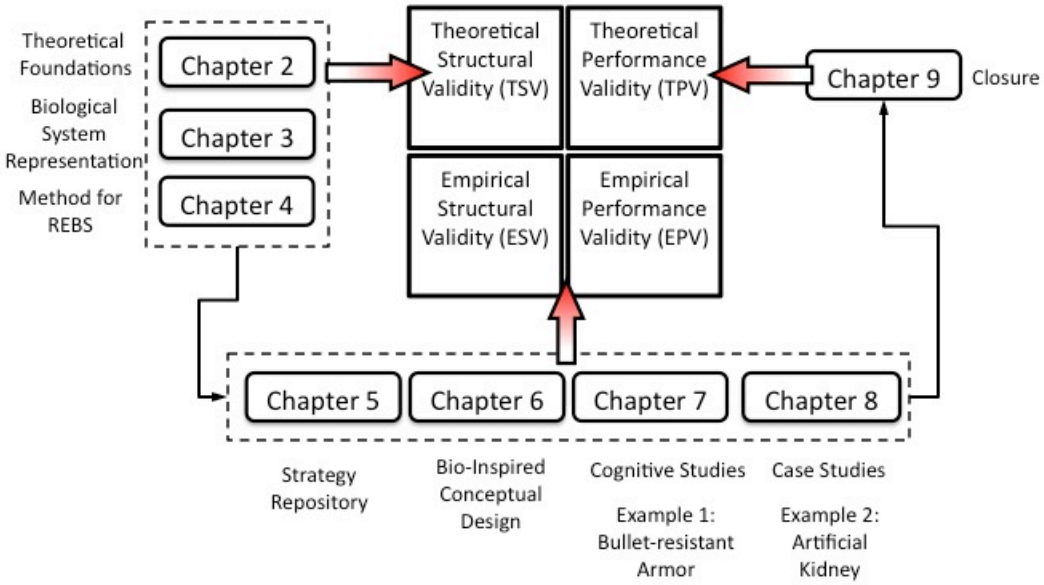


Figure 1.6 Validation Strategy

1.5 CHAPTER SUMMARY

In Chapter 1, the motivation of this research (Section 1.1), current research on bio-inspired design (Section 1.2), the research approach undertaken in this research (Section 1.3) , and the validation strategy (Section 1.4) was presented.

In the next chapter, Chapter 2, the theoretical foundation of this work is laid. This includes systematic design, engineering representations, engineering ontologies, and evaluation techniques.

CHAPTER 2 THEORETICAL FOUNDATIONS

In Chapter 1, current methods for systematizing bio-inspired design were reviewed and several research gaps were identified. To address these gaps, a method for Reverse Engineering Biological Systems and an engineering ontology to support of retrieval biological strategies is proposed. The objective of this chapter is to introduce and review the theoretical foundation upon which this work is built. Specifically, systematic design and idea generation, representations in engineering design, and engineering ontologies are reviewed. Additionally, metrics used to empirically evaluate idea generation techniques are reviewed. The relationship between these concepts and the proposed approach for Bio-Inspired Concept Generation is displayed in Figure 2.1.

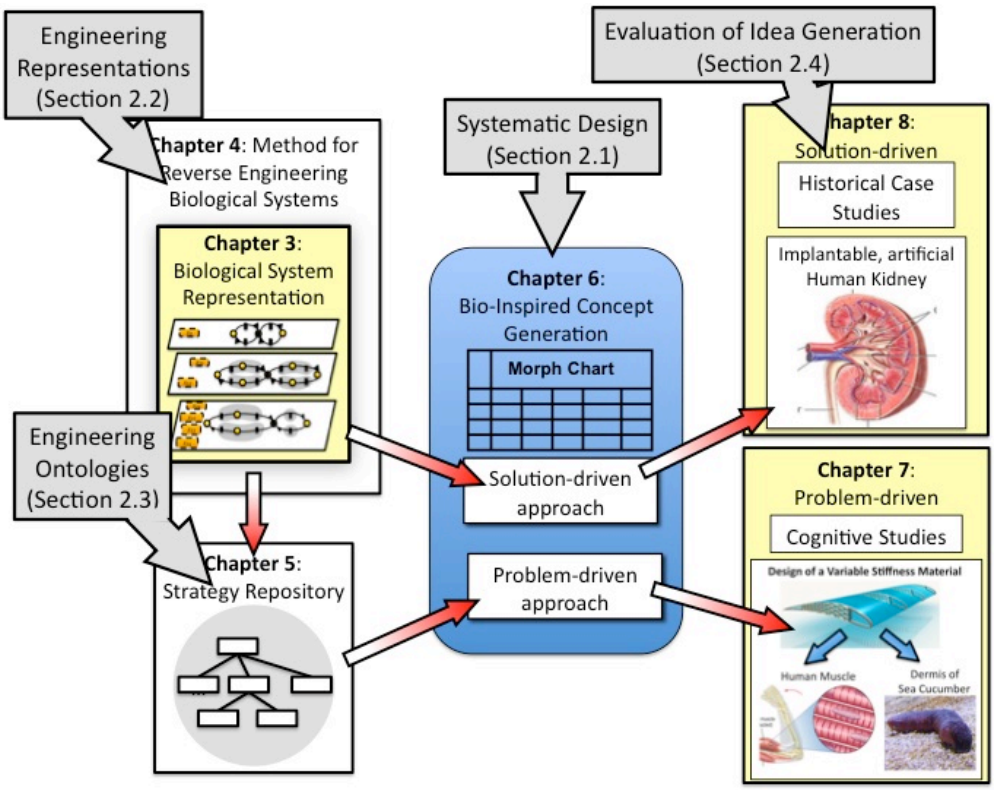


Figure 2.1 Relationships between concepts reviewed in this chapter and proposed method

One of the goals of this work is to systematize the use of biological strategies in engineering design. In Section 2.1, the context in which the method is used, the systematic design method, is used. Existing methods for idea generation, including analogies, are also reviewed in Section 2.1. In the proposed method, hierarchical Petri nets are used to represent biological systems and extract functional strategies from these systems. In Section 2.2, the value of representations as mental models and common representations used to represent engineering systems are reviewed. To aid access to these strategies, the hierarchical Petri net representation is used to structure a repository of biological and engineering strategies. Foundations for design repositories and semantic retrieval are reviewed in Section 2.3. In Section 2.4, metrics for evaluating idea generation techniques are reviewed. These metrics are used to empirically evaluate the value of proposed method.

2.1 SYSTEMATIC DESIGN AND THE SEARCH FOR SOLUTIONS

2.1.1 The Design Process (Pahl and Beitz overview)

The research put forth in this dissertation is set in the context of the systematic design methodology, developed by Gerhard Pahl and Wolfgang Beitz. A design methodology is a "concrete course of action for the design of technical systems that derives its knowledge from design science and cognitive psychology, and from practical experience in different domains."[5] The Pahl and Beitz systematic design methodology is divided into four main phases: Planning and Clarification of Task, Conceptual Design, Embodiment Design, and Detail Design. These phases are detailed in the following discussion.

The four phases of the Pahl and Beitz systematic design method are as follows:
Planning and Clarification of Task – The first phase, Planning and Clarifying the Task, begins with the designer identifying and analyzing the market for a potential product. From this analysis, product ideas are generated and a product proposal is developed. The task is then clarified by collecting information about the requirements and specific

constraints on the future product. This phase ends with the articulation of a Requirements List based on the identified requirements and constraints.

Conceptual Design – In Conceptual Design, the designer takes the requirements developed in the Planning and Clarification of Task phase and develops concepts bases on them. The designer first abstracts the requirements to identify a solution neutral problem statement, then establishes a function structure for the future product based on the essential problem identified. A function structure is a representation of the functions and sub-functions of a given system along with the relationships (flows of energy, material, and signal) between these functions. A function structure for a potato harvesting machine is displayed in Figure 2.2.

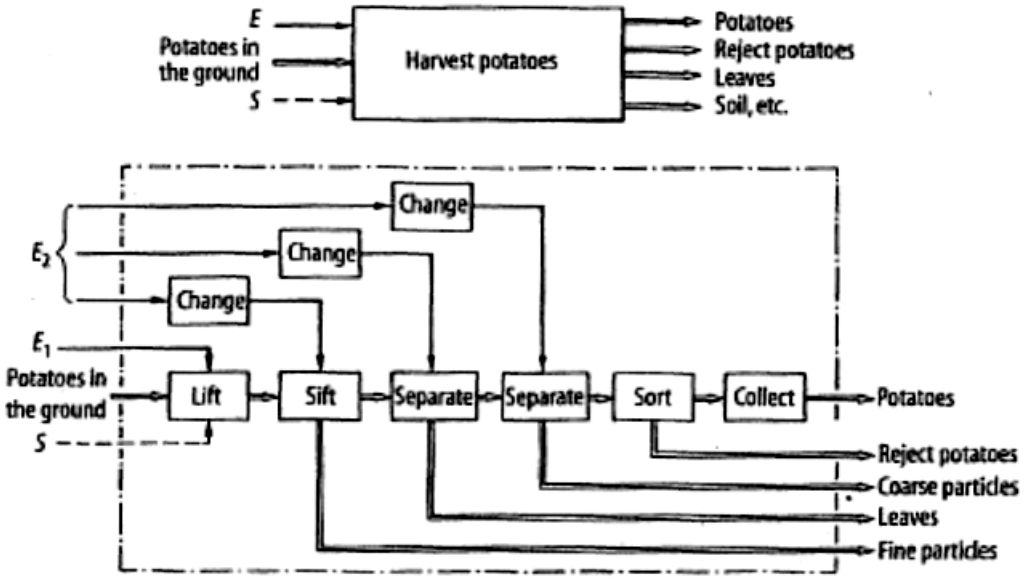


Figure 2.2 Function Structure of potato harvesting machine [5]

In Figure 2.2, the overall function of "harvest potatoes" is further decomposed into its sub-functions of "lift", "sift", etc. The flows of energy, material and signal are also displayed in Figure 2.2. After defining the function structure, the designer then searches for working principles (functional solutions) for the identified functions and sub-functions. A morphological matrix is commonly used to facilitate the search for

working principles, or behaviors to fulfill the function of interest. A morphological matrix for the previous example of the potato-harvesting machine is displayed in Figure 2.3.

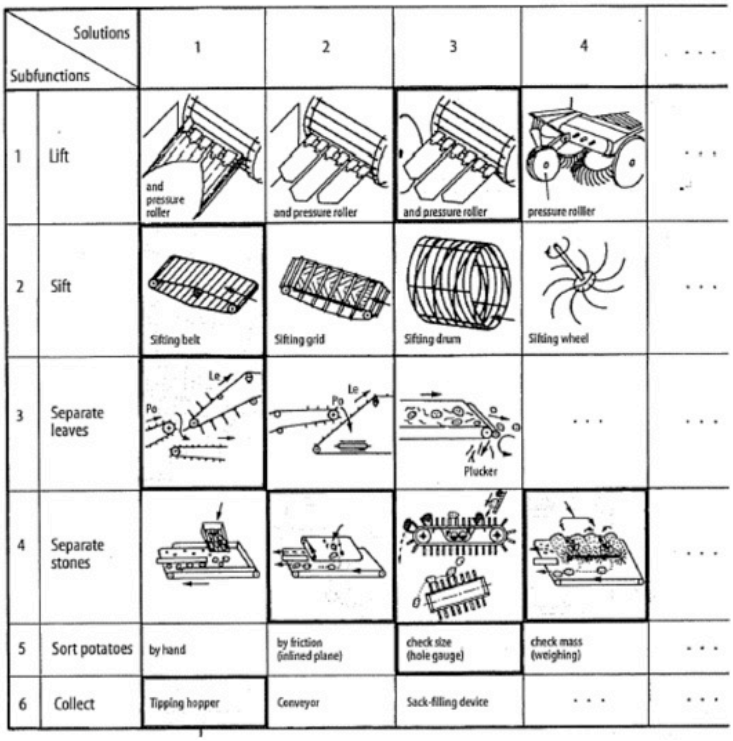


Figure 2.3 Morphological matrix for potato harvesting machine [5]

In Figure 2.3, the "sifting belt", "sifting grid", "sifting drum", and "sifting wheel" working principles were elaborated to fulfill the "sift" sub-function of the potato harvesting machine. Next, working structures (preliminary concepts) are developed by combining suitable working principles while ensuring physical and geometric compatibility. These working structures are then firmed up into solution variants and the designer evaluates these variants against technical and economic criteria. The Conceptual Design phase ends with the specification of a principal concept.

Embodiment Design – During this phase, the designer takes the concept selected in Conceptual Design and develops a preliminary layout for it. In most cases, several layouts are developed. These layouts are evaluated based on concrete technical and

economic criteria and the best layout is selected. This preliminary layout is then refined into a definitive layout by identifying and eliminating it shortcomings.

Detail Design – During Detail Design, the designer finalizes the arrangement, forms, dimensions, and surface properties of the product. This phase also includes material specification, cost estimation, and the development of all production documents needed.

A flowchart depicting the phases of the Pahl and Beitz systematic design method is displayed in Figure 2.4. The flowchart displays the inputs and outputs of the four phases, as well as the working steps of the methodology.

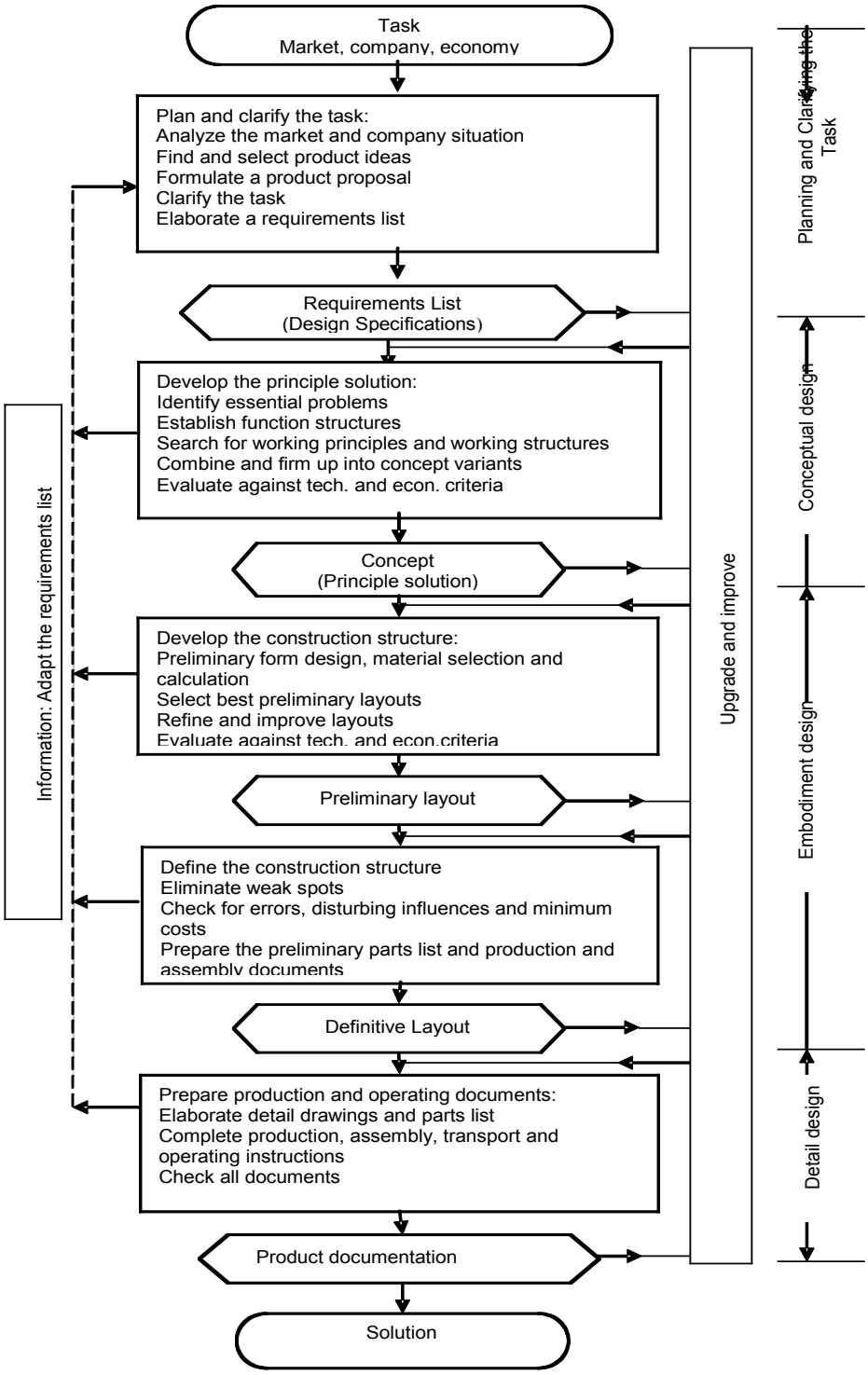


Figure 2.4 Pahl and Beitz Systematic Design Methodology [5]

2.1.2 Idea Generation Techniques

The phase of particular interest in this research is the Conceptual Design phase, where the designer is tasked with the search for working principles to fulfill functions. In this step, the designer searches for specific solutions to the sub-functions defined in the function structure. In this search, the designer may use several methods for identifying solutions for each sub-function, including conventional methods, intuitive methods, and discursive methods. These search methods are explained as follows:

Conventional methods - Conventional methods for searching for solutions include:

- *Information gathering* – Information gathering includes conducting a survey through various forms of related literature, trade publications, catalogues, patent websites, etc. Some popular online sources for information gathering are Google Patents (www.patents.google.com) and the United States Patent and Trademark Office (www.uspto.gov).
- *Analysis* – Analysis involves the reverse engineering existing systems "aimed at the discovery of functional and physical features and their respective relations". Using this method, the knowledge extracted from existing technical systems, as well as biological systems, is used to stimulate the design of new systems.
- *Analogies* – In the search for solutions, it is often useful to substitute an analogous problem for the one under consideration. Solutions from the analogous system are then used to solve problems from the current problem domain.

Other conventional methods include synthesis, measurements, and model tests. A brief review of these methods can be found in Messer [38] and Pahl and Beitz [5]

Intuitive methods – Intuitive methods rely mostly on the designer's intuition in the search for solutions. Some methods include:

- *Brainstorming* - Brainstorming is a method by which a group of open-minded individuals from diverse backgrounds generate ideas in an open forum. This

method 'relies strongly on stimulation of the memory and on the association of ideas that have never been considered in the current context." [5]

- *Method 635* – Method 635 is a brainstorming method where groups of 6 participants are formed. Each participant writes down three keywords, which are then passed to his/her neighbor. The neighbor records three further solutions. Ideas are passed a total of five times, hence, the Method 6 (participants) – 3(ideas) – 5(ideas passed).
- *Gallery Method* - In the Gallery method, a group of participants sketch solutions to an ideation task for 15 minutes. After this initial generation period, the group reviews the ideas of the individual participants. The participants then refine and further develop the ideas, followed by a group selection of the most promising solutions.
- *Synectics* - Synectics is similar to brainstorming, but uses analogies to stimulate ideas from the participants. The two guiding principles of synectics are "making the strange familiar" and "making the familiar strange".
- *Delphi Method* - The Delphi method relies on the opinions of a panel of independent experts. In this method, experts are given anonymous surveys. After submitting their responses, the responses of all participants are collated and sent back out to the group. Participants are encouraged to revise their original answers until the group converges and a consensus drawn. This method is usually used for long-term studies in design, such as forecasting.

Other intuitive methods include collaborative sketching, the input/output technique, lateral thinking, visual thinking, attribute listing, forced relationship technique, blockbusting, and the parameter analysis. A brief review of these methods can be found in Messer [38] and Pahl and Beitz [5]

Discursive methods – Discursive methods seek to deliberate a step-by-step approach to solution searching. In discursive methods, problems are decomposed into manageable parts and systematically analyzed. Founded in intution, the additional use of systematic procedures serves to increase the output and inventiveness of designers [38]

- *The Method of Forward Steps (Method of Divergent Thought)* - In this method, the designer starts with an initial solution. From this initial solutions, the designer follows as many divergent paths as possible away from this initial solution, yielding more solutions.
- *The Method of Factorization* - The method of Factorization involves decomposing complex problems in to smaller, manageable and definable sub-problems (factors). Each of these sub-problems is then solved independently.
- *Design Catalogs* - Design catalogs are "collections of known and proven solutions to design problems." [5] In this method, the user searches through these design catalogs for possible solutions.

Other discursive methods include the method of persistent questions, checklisting, morphological thinking, method of negation (systematic doubting), method of systematic variation, systematic study of physical processes, and systematic search with the help of classification. Brief reviews of these methods can be found in Messer [38] and Pahl and Beitz [5].

It should be noted that it is beneficial to use multiple methods for identifying solutions so as to broaden the search through the design space as much as possible. The focus of this research is on using biological systems as analogies to stimulate concepts. Analogical reasoning is reviewed in Section 2.1.3.

2.1.3 Analogical Reasoning

Analogy refers to a similarity of relations between two different situations, or A:B::C:D, where A is related to B like C is related to D [39]. This relationship implies that there is a higher order abstraction that holds in both cases [39]. Analogy is often used to transfer knowledge, through analogical mapping, from a source domain containing the analogous phenomena to a target domain containing the problem to be solved by analogy [40].

In engineering design, analogy functions in many different ways, mainly for explanation, problem solving, and problem identification. Explanation involves using

analogies as a means of communicating novel ideas, while avoiding misunderstanding. Analogy is also used as a means of problem solving, whereby designers use solutions from a source, or base, domain to solve problems in the target domain. Problem identification is also considered an important function of analogy, whereas analogies are used to identify potential problems and evaluate novel concepts [41].

The source-target transfer in analogical reasoning occurs in the following stages: (a) accessing the source domain, (b) mapping elements of the target onto source, (c) transferring knowledge from the source to the target domain, and (d) inducing a schema. The access stage involves activating the user's mental representation of the source domain. The goal of the mapping stage is aligning the source and target domain in a manner that the knowledge from the source domain can transfer to the target. In the next stage, knowledge from the source domain is transferred to the target domain. Inherent in this transfer is the "belief that domains known to be similar in certain respects are likely to be similar in others". In the last stage, a more abstract knowledge structure is created. This abstract knowledge structure is often used as a source in future analogical reasoning situations [42].

Analogies can be classified by their similarity, or conceptual distance, between the source and target domain. Local analogies are analogies where the source domain is similar to that of the target, where surface-level attributes and relations between these attributes can be mapped [43]. Surface-level attributes are easily retrievable aspects of representation, such as color and shape [44]. In distant analogies, where the source and target domains are very different, few surface-level attributes can be mapped and relational similarity must be relied upon [43]. For example, take the example of a target analog of an air conditioning system for a hotel. A local analog would be other commercial air conditioning systems from other companies, where a distant analogy would be self-cooling termite mounds.

Because distant analogies involve vastly different domains of knowledge, it is usually more difficult to transfer knowledge or solutions [45]. These analogies are more difficult to access because the nature of the similarity is at a more abstract, relational level, thus causing an increase in cognitive effort [43].

2.1.4 Relation to Bio-Inspired Concept Generation

In this research, biological strategies are used in idea generation in the search for solutions to engineering problems. Specifically, biological systems are used as analogies; solutions to problems in the biological domain are leveraged to solve problems in the engineering domain. The question now becomes, *Why biological systems as a source for potential analogies?*

Because of the large conceptual distance between the biological and engineering domain, biological systems are considered distant analogies. Analogical theorists have indicated that the number of distant analogies used during design is positively related to the originality (novelty) of the resulting design [43] and these analogies are considered the main drivers of truly innovative thought [46]. Similarly, Benami and Jin [47] found that ambiguous ideas stimulated more ideas than non-ambiguous entities, which tend to be fixating. Local analogies represent smaller conceptual distances, and researchers [48, 43, 41] have found that these analogies constrain the creativity process by providing paths-of-least resistance for analogizing, resulting in less original, incremental ideas. These ideas typically show deviation from the source and more attributes are preserved[46]. Thus, because of their large conceptual distance for engineers, biological analogies should lead to more innovative design ideas.

In Section 2.1, systematic design and idea generation techniques were reviewed and a case was made for the used of biological systems as inspiration in the idea generation process. In Section 2.2, representations used in engineering design are reviewed.

2.2 REPRESENTATIONS

In Section 2.2, representation in engineering design is reviewed. This section begins by reviewing general models of cognitive processes in idea generation in Section 2.2.1. In Section 2.2.2, mental models used in creative cognition are reviewed. In Section 2.2.3, common representations used in engineering design are reviewed.

2.2.1 Creative Cognition

A general model of cognitive processes involved in creative thought is that of the Geneplore model, developed by Finke, Ward, and Smith [49] and displayed in Figure 2.5.

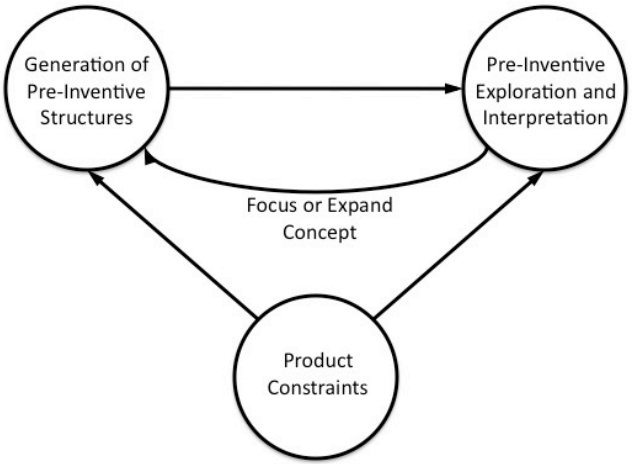


Figure 2.5 Geneplore Model [49]

With the Geneplore model, creative cognition is divided into two phases: a generative phase, followed by an exploratory phase. In the generative phase, mental representations, called pre-inventive structures, are constructed. The properties of these structures are then exploited for creative purposes in the exploratory phase. If the pre-inventive exploration leads to resolution of the task, the pre-inventive structures may lead to a creative product [49]. On the other hand, if exploration does not lead to resolution, one then returns to the generation phase by either focusing the emergent structure on specific problems or expanding the structure to explore more general conceptual possibilities. Constraints on the products are also considered in both generative and exploratory processes. Examples of cognitive processes, structures, properties, and constraints of the Geneplore model are displayed in Table 2.1.

Table 2.1 Examples of processes, structures, properties, and constraints in the Geneplore Model [49]

Generative Processes	Pre-inventive Structures	Pre-inventive Properties	Exploratory Processes	Product Constraints
Retrieval	Visual Patterns	Novelty	Attribute finding	Product type
Association	Object forms	Ambiguity	Conceptual interpretation	Category
Synthesis	Mental blends	Meaningfulness	Functional interference	Features
Transformation	Category exemplars	Emergence	Contextual shifting	Functions
Analogical transfer	Mental models	Incongruity	Hypothesis testing	Components
Categorical reduction	Verbal combinations	Divergence	Searching for limitations	Resources

A much more specific model of creative cognitive processes has been developed by Benami and Jin [47]. Benami and Jin [47] build upon the Geneplore model in building a cognitive model for Conceptual Design. In this model, the pre-inventive structures can include the functional (F), behavioral (b), and/or structural (f) elements that make up a design entity. As new elements are generated and explored, the pre-inventive entities become knowledge entities as their relationships with other functions, structures, and behaviors are fully interpreted. The model is displayed in Figure 2.6.

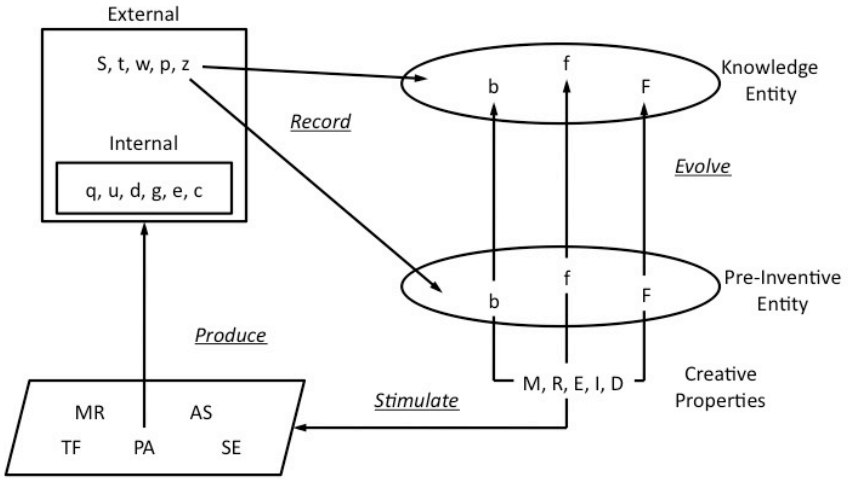


Figure 2.6 Cognitive Model of Conceptual Design [47]

The first step of the model addresses the stimulation process, whereby designers are stimulated to generate and explore ideas after viewing existing design entities in catalogues or other documentation. Stimulating properties include *Meaningfulness* (M), *Relevance* (R), *Emergence* (E), *Incongruity* (I), and *Divergence* (D). In the second step, internal design operations are produced, followed by the production of external design operations in step 3. Internal cognitive processes include *Suggest* (g), *Compute* (c), *Question* (q), *Declare* (d), *Suppose* (u), and *Explain* (e) and external cognitive processes include *Talk* (t), *Write* (w), *Sketch* (s), *Point* (p), *and Stimulate (*z). In cognitive studies on the stimulation phase, Benami and Jin found that (1) short-distance analogies resulted in a larger quantity of ideas, (2) long-distance analogies resulted in more original ideas, and (3) neither resulted in a larger variety of ideas. They also found that behaviors stimulated twice as many ideas as functions.

The Benami and Jin Cognitive Model of Conceptual Design is a specialization of the Geneplore model put forth by Finke, Ward, and Smith. Specifically, the Benami and Jin model takes the two-phase approach of generation and exploration to show how ideas are stimulated and transformed into knowledge entities through exploration. The most critical phase of this model of Conceptual Design is the stimulation phase. In this research, biological analogies are used to stimulate these new knowledge entities. In their work, criteria for meaningfulness, relevance, emergence, incongruity, and divergence are put forth to characterize the properties needed in order for an idea or analogy to be stimulating. Biological analogies can be seen as meaningful, as nature is bound by the same physics as designed systems. Incongruity comes in the distance between the engineering and biological domains. These analogies can be seen as divergent since biological solutions do not offer ready-made solutions for engineering problems and some idea generation is still needed in applying the strategies from the solution. Since biological systems are hierarchically arranged, many emergent features of the system come about in decomposition of the system. Therefore, according to this model, biological analogies should be stimulating, assuming that the analogies are

relevant. Retreiving relevant biological analogies is addressed in Chapter 5 of this dissertation.

2.2.2 Representation and Mental Models

In both models, pre-inventive structures play a critical role in creative cognition. Pre-inventive structures take several representation formats, as displayed in Table 2.1. Markman [50] broadly defines representation as having four components:

- Represented world – the domain that the representations are about
- Representing world - the domain that contains the representation
- Representing rules – rules that map elements of the represented world to elements in the representing world.
- Representation process – a process that uses the representation.

In essence, the representing world is used to represent knowledge in the represented world. Representations in the representing world are bound by representing rules that relate them. The process by which the representation is created usually results in an loss of information between the represented and the representing worlds.

Mental models have been explored as key representations for physical systems. Mental models are commonly defined in terms of a set of autonomous objects and relationships between these objects. Autonomous objects are mental objects with an explicit representation of state, an explicit representation of topological connections to other objects, and a set of internal parameters [51]. As the name suggests, mental models can be observed, manipulated, and reasoned by the mind [52]. Williams and co-authors [51] found that mental models play an important role in human reasoning; they allow the user to reason about the effects of changes in a system using qualitative relationships. White and Frederiksen [53] also argue that people reason about physical systems using qualitative reasoning, primarily by zero-order (presence or absence) and first-order (incremental changes) models. The authors also argue that quantitative reasoning only comes after the system is understood qualitatively.

2.2.3 Representations in Engineering Design

In Section 2.2.2, representations, in general, used in creative cognition were discussed. In this section, specific representations used in engineering design are reviewed. In Section 2.2.3.1, common definitions for function and behavior are reviewed. This discussion is followed by a review of representations for function, structure, and behavior in Section 2.2.3.2. Lastly, current methods for synthesizing functional, behavioral, and structural knowledge into a complete system model are reviewed in Section 2.2.3.3.

2.2.3.1 Function and Behavior

Defining Function

In defining function, researchers [54, 55, 32, 56] have defined multiple distinct types of functions, including that of purposive and operational (action) functions. Action functions are defined as " a physical interaction between two objects of interest, each of which may be a component of a design or the design itself and its environment". Purpose functions can be defined as " a description of the designer's intention or the purpose of a design."[56] Purpose functions can be seen as a higher level design abstraction and allow for a much wider variety of design solutions to be found. On the other hand, action functions are much more specific to a given design or behavior and relate directly to the physical principles of the device.

In this research, the functions are considered action functions. This is mainly due to the fact that biological systems have already been designed, thus any specifications or predictions of purpose are highly subjective.

Defining Behavior

Behavior is commonly defined as the change of state of a particular system. Chittaro and Kumar [32]define behavior as describing "how components work and interact in terms of quantities which characterize their state (variables and parameters) and the laws that govern their behavior". Along these same lines, Deng [56] characterizes behavior as a chain or network of physical state change.

2.2.3.2 Representations of function and behavior

In the previous section, common definitions of function and behavior were put forth. In this section, common representations for function and structure are reviewed.

Function

There are several common representations of function, including the input-output flow transformation, transformation of input-output states, and informal representations.

Input-output (I/O) flow transformations

In the I/O transformation representation [5], function is represented in terms of its input and output flows of energy, material, and signal. For example, the function of a lever can be represented with an input force and a multiplied output force [56]. Function structures, utilized in Pahl and Beitz, use the I/O flow functional representation. An example of a function structure is displayed in Figure 2.2.

Transformations between input-output states.

In this representation, function is represented in terms of its input and output states. For example, the functional of a lever can be represented by the changes in the angle or height of the end of the lever.

Informal representations

In the informal representation of function, functions are expressed using either that of a verb-noun pair or a natural language sentence representations. In the verb-noun representation, function is expressed as a verb-noun pair. For example, the function of a lever is expressed as "to magnify force". In the natural language representation, restrictions on expression are dropped and sentences are used to describe function.

It should be noted that combinations of these representation types are also used to represent functions in design. For example, function structures utilize both I/O flow and verb-noun representations.

The view adopted of function in this work is that of a mapping of the behavior of the system to that of its supersystem[54]. In this view, the I/O flow transformation representation of function is used to represent function. Flows are used to map the behavior of the system to other systems around it, which forms the supersystem. Because

flows are not used, the transformation between I/O state representation is limited in its representation of this mapping. In addition, informal representations, such as the verb-noun and natural language, lack of rigor and uniqueness [56] in the representation of this mapping. They lack rigor in the sense that the meanings of the words are open for interpretation. These informal representations also lack uniqueness, as they can be described using multiple synonyms for the same words.

Behavior

Common representations for behavior include formal methods such as mathematical representations, textual descriptions, bond graphs, and Petri nets.

Formal mathematical representations

Formal representations of behavior include the use of mathematical transformations expressed using equations and mathematical relations to represent the behavior of an object.

Textual Description

In the textual description, the behavior is expressed using a textual and natural language format. For example, the SAPPhIRE causal behavioral description by Chakrabarti [24] is expressed in textual format. In this representation, the content is divided into a list of actions, state, physical phenomena, physical effects, inputs, organs, and parts. An example of the causal behavioral description for a Venus flytrap is displayed in Figure 2.7.

Action

- Feed on insects by trapping them between leaves

State

- Insect, which is the Venus flytrap's prospective prey, is freely moving outside the trap.
- Cells in the underlying layer are compressed, creating tension in the plant tissue and holding the trap open.
- …….

Physical phenomenon

- Emit a scent by secreting chemicals from glands on the inside of the open leaf/trap.
- Attract insects toward the trap with the help of the scent.
- …….

Physical effects

- Stimulus-response effect of the glands that produce scent-emitting chemicals.
- Stimulus-response effect of the insect's nostrils.
- …….

Input

- Electrical signals to the gland that produces chemicals responsible for the scent.
- Chemical stimulation in the form of the scent to the nostrils of the insect.
- ……..

Organ

- The ability of the scent gland to produce appropriate chemicals that emit the scent.
- The composition of the scent, which is responsible for stimulating the sense of smell in insects.
- ……..

Parts

- Nectar glands present on the inside of the leaf.
- Scent, which is made up of chemicals that stimulate insects in particular.
- …….

Figure 2.7 Textual representation of Venus Flytrap example [24]

Bond Graphs

Bond graphs, developed by Henry M. Paynter at Massachusetts Institute of Technology in 1959, are considered static graphical representations of dynamic physical systems. Bond graphs automatically conserve energy, and depict flows of energy into and out of the system through energy ports (or the intersection of the boundary of the system and the environment) of a system. For example, consider the following example [57] (Figure 2.8) of a motor connected to a battery. A dashed, gray circle denotes the boundary of the system and energy crosses this boundary at energy ports. The energy ports of the motor are the shafts and electrical wires connecting the battery to the motor.

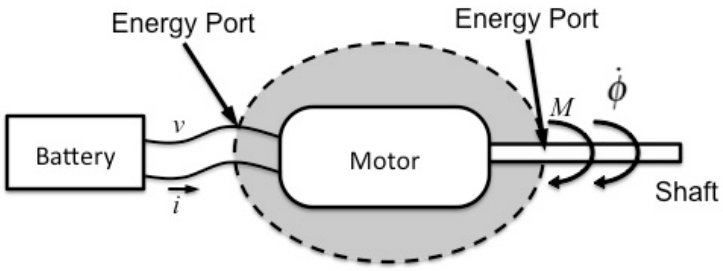


Figure 2.8 Electric Motor Example

Bond graphs have been used to model many types of physical systems, including pneumatic, hydraulic, mechanical, electrical, and combinations of these types of systems. Bond graphs are modelled using the flow of power between systems. Power flow is denoted as the product of a generalized forces (efforts) and a generalized velocities (flows) [57]. In Figure 2.8, power flow is denoted by the flow of electrical voltage (e) and current (i) to the motor, and the power flow of torque (M) and angular velocity($\dot{\phi}$) from the motor, or as the following equation:

$$P = e\dot{q}$$

where e is the effort and $\dot{q}$ is the flow. Using a bond graph of the motor, the system is graphically depicted in Figure 2.9.

Figure 2.9 Bond graph of motor

In Figure 2.9, the single-sided arrow denotes the sign convention of the power flow variables. In analysis, engineering systems are modeled as ideal systems, meaning that energy is not stored, generated, or dissipated[57]. If the motor in the figure above was modeled as ideal, then the graph reduces to the following equation,

$$e \cdot i = M \cdot \dot{\phi}$$

Causality in bond graphs is depicted using what is termed as a causal stroke. The causal stroke denotes which component generates the effort. In Figure 2.9, the single bar on the end (or start) of the single-sided arrow denotes the causal stroke. In this case, the battery generates the voltage that goes to the motor, and the motor generates the torque (moment) effort.

In bond graph representation, physical elements are replaced with bond graph elements, including compliance energy storage elements (C), resistor elements (R), inertance storage elements (I), junctions, transformers, and gyrators. Once the bond graphs are generated for the system, constitutive equations for each type of element are used to derive differential and algebraic equations to analyze the behavior of the systems.

Petri nets

Petri nets can also be used to represent the causal behavioral description. Petri nets are a graphical and mathematical tool used for modeling, formal analysis, and design of discrete-event systems[58]. As a graphical tool, Petri nets provide the means to represent the behavior of dynamic systems. As a mathematical tool, they allow for the formal analysis of the behavioral properties of a system.

An ordinary Petri net is a 4-tuple, where $PN = (P, T, F, M_0)$. Petri nets contain two types of nodes, termed places *(P)* and transitions *(T)*. Places can be defined as states of a discrete-event system and transitions can be defined as changes between those states, where arcs *(F)* define the relationship either from a transition to a place or from a place to a transition. The initial marking, M_0, is considered the initial state of the graph. This marking is denoted graphically by a token distribution amongst the places, whereby a token denotes "truth" of a given place.

The formal definition of an ordinary Petri Net is defined as follows[59]:

An ordinary Petri net is a 4-tuple, $PN = (P, T, F, M_0)$ where:

$P = \{p_1, p_2, \ldots\ldots, p_m\}$ is a finite set of places,

$T = \{t_1, t_2, \ldots\ldots, t_m\}$ is a finite set of transitions,

$F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs, or flow relations

$M_0 : P \to \{0,1,2,.....\}$ is the initial marking

$P \cap T = \varnothing$ and $P \cup T \neq \varnothing$

A Petri net structure $N = (P, T, F)$ without any specific initial marking is denoted by N. A Petri net with the given initial marking is denoted by (N, M_0).

The graphical representation of a Petri net is displayed in Figure 2.10.

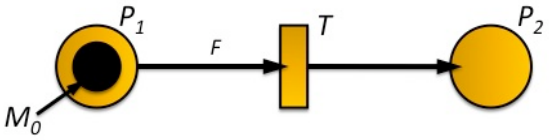


Figure 2.10 Graphical representation of a Petri net

The behavior of discrete event systems is described by the change in the marking of the system. The marking is changed through the firing of transitions, which transfer tokens from their input places to their output places. A transition is said to be enabled (able to be fired), if each of its input places is marked with a token.

Koga and Aoyama [60] used the Petri net representation to generate product behavior and structure based on step-by-step decomposition. In this representation, the authors use the Petri net representation to model the structural and behavioral hierarchy of a stapler design. The authors specifically use this model as a means of improving product quality and dependability through behavior generation. The first two levels of the stapler design hierarchy are displayed in Figure 2.11.

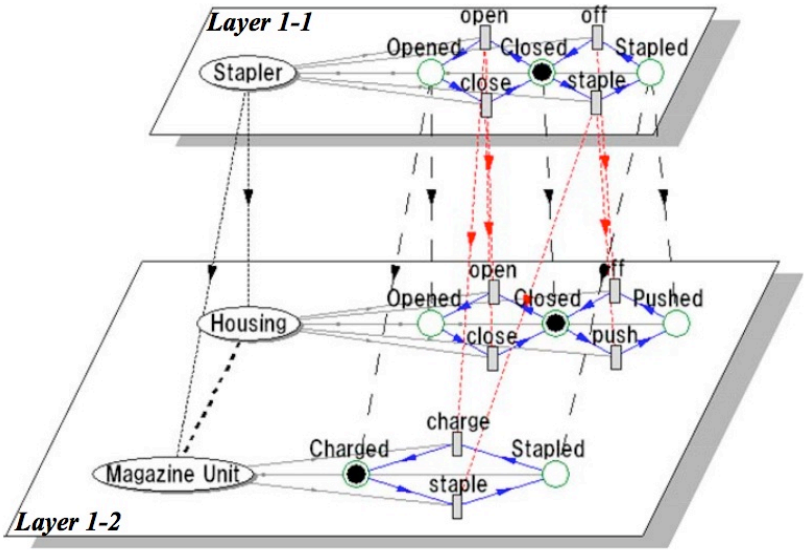


Figure 2.11 Hierarchical Petri net representation for a stapler design [60]

In the representation proposed in this work, information about what the system is (structure), what the system does (behavior), and "why" it does what it does (function) is needed. It is commonly known that a complete system representation includes structural, behavioral, and functional views of the system, as well as the relationships and mapping between the different views. The completeness is measured by its accounting of these views of the system

2.2.3.3 Knowledge Representations

The separate representations for function and behavior were discussed in Section 2.2.3.2. In Chang et al. [61], the authors found individual views of function, behavior, and structure to be inadequate for representing complex systems. Because of this, many multi-view representations have been developed. These multi-view representations, termed knowledge representations, include that of the Structure-Behavior-Function model, Function Behavior-State model, Functional Rationale, Function-Behavior-Structure, Function-Environment-Behavior-Structure, and the Causal Behavioral Model. These representations are discussed in more detail below.

1. *Structure-Behavior-Function model*: Goel and co-authors [16-20] developed a theory of modeling complex systems termed a Structure-Behavior-Function model, or SBF. These models explicitly represent a device's structure (configuration of components and relationships), behavior (internal causal process represented by states and transitions between them), and function (output behaviors). SBF models consider behavior as the causal link between structure and subjectively-defined functions. SBF models are organized in a F➔B➔F➔B➔…➔F➔S hierarchy, decomposing function and behavior in a coupled manner. The lowest level functions are then associated with structure.
2. *Function-Behavior-State model:* Umeda, Tomiyama and co-authors [62-64] developed the Function-Behavior-State (FBSt) representation, whereby subjective function is distinguished from the objective parts of design, behavior and state. In FBS, state is defined as entities, attributes of entities, and relations among entities, between entities and attributes, and among attributes. Behavior is defined as a sequence of states of time, and function is defined as a "description of behavior

recognized by a human through abstraction in order to utilize it". With FBSt, designs are modeled hierarchically with respect to function, behavior, and state. In this representation, the objective parts of the design (behavior, state) are modeled as a set called the "aspect".

3. *Functional Representation:* Chandrasekaran and co-authors [65] present a function-oriented causal representation scheme for design. Functional Representation (FR) takes a top-down approach to representing a device by first describing the overall function. The behavior of each component is then described in the context of the overall function. In Chandrasekaran and Josephson [66], structure, behavior, and function are defined and explored from both device-centric and environment-centric views.
4. *Function-Behavior-Structure* – Gero and colleagues [67-69] developed a framework for modeling the function, behavior, and structure of a design object using an FBS representation, and present a design process in the context of transformations between function, behavior, and structure. In this research, function, behavior and structure are considered as classes of properties of a design object, where function properties dictate the object's intended purpose or teleology, structure properties represent the physical components and their relationships, and the behavior properties describe what the object does to achieves its function.
5. *Function-Environment-Behavior-Structure* – Deng and co-authors [34] present a representation where four key aspects of the design are represented, including function, behavior, structure, and working environment. In this work, function characterizes the general purpose of the device, and is hierarchically decomposed to a set of sub-functions. Behavior is represented as a flow-of-action, input-output relationship, as opposed to a flow-of-object relationship, which is typically used. The structure consists of the physical components being represented, and the working environment consists of the environmental elements that contribute to the device's function. The causal behavioral process based on the flow-of-action representation is key to this work , as it bridges the four aspects of the design and

provides a means for relating them. This behavioral process is represented by a directed graph.

6. *Causal Behavioral Process* – Chakrabarti and co-authors [24] have developed a representation that links function, structure, and behavior with a generic causal behavioral model. This generic model, termed SAPPhIRE, links the seven constructs of State, Action, Part, physical Phenomena, Input, oRgan, and Effect, and is implemented using software. The SAPPhIRE model is represented in natural language format using nouns, verbs, and adjectives and used to represent the function, structure, and behavior of biological and artificial systems

2.2.4 Research Opportunity

Representations play a key role in cognition when physical phenomena cannot be experienced directly. In this section, a case was made for the value of qualitative mental models in creative cognition. In the proposed method, we develop a representation of biological systems that can aid engineers in understanding and leveraging biological phenomena in idea generation. In engineering design, functional, behavioral, and structural representations of systems serve as models that can be manipulated throughout the design process to create engineering artifacts. In the proposed method for Reverse Engineering Biological Systems, a multi-viewed representation of biological systems is leveraged. This will not only aid in comprehension of the behavioral strategy utilized by the system, but also aid in reusing and cataloguing these models for future use in ideation.

A review of representations in engineering design was presented in Section 1.2.3. In Chapter 3, these engineering representations will be reviewed with respect to representing biological systems.

2.3 ONTOLOGY DEVELOPMENT

2.3.1 Design repositories

A design repository is an intelligent knowledge-based design artifact modeling system used to facilitate the representation, capture, sharing, and reuse of design knowledge [70]. In a review of design repositories, Szykman and co-authors distinguish repositories from traditional design databases in several ways, including [70]:

- Traditional design databases are typically more data-centric than knowledge centric; design repositories attempt to capture more comprehensive information such as characterization of function, behavior, design rules, simulation models, etc.
- Design repositories tend to be more heterogeneous in the types of information they contain, whereas databases tend to be homogenous.
- Design repositories allow not only the storage of complex information, but also support the retrieval and reuse of design knowledge through sophisticated methods.

In the engineering domain, there have been several research efforts with the purpose of developing design repositories to aid in the storage and retrieval of complex information. Particularly, case-based reasoning approaches (CBR) have been heavily used (See Refs. [71-74] for a review of CBR approaches). Although CBR approaches have shown value in aiding the storage and retrieval of complex information, there are limitations in the current research. One major limitation is that of extensibility [75]. Current CBR approaches require representing the domain knowledge, indexing cases, and detecting similarities; however, these procedures are performed in an ad-hoc fashion [76]. Yim [75] comments that there is no formalism for representing and reasoning cases, which makes extending previously built repositories using CBR extremely difficult, especially in a distributed environment.

To overcome the limitations of traditional CBR, several researchers [70, 77-81, 30] are developing repositories following an ontological approach. In the ontological approach, domain knowledge is formally and explicitly represented, while retrieval is performed using semantic inference using rule-based logics or ontological matching [75]. An ontology is a highly structured system of concepts covering the processes, objects, and attributes of a domain along with the relationships between these concepts [30]. Noy and McGuinness [82] list several advantages to developing ontologies, including that of (1) sharing a common understanding of the strucure of information among people or software agents and (2) enabling reuse and extension of domain knowledge.

Because of the many advantages, several researchers have worked on developing ontologies for the engineering domain. Kim and coauthors developed of a method for storing and retrieving electromechanical components [77] using the knowledge representation environment LOOM. Ramani and co-authors developed an approach for building a design repository using ontologies and natural language processing [83]. Kopena and coauthors [84] developed a method for retrieving mechanical devices in Concepual Design using description logics. Li and coauthors [78, 79, 83, 81, 30] developed an engineering ontology for information retrieval of unstructured engineering documents. Yim [75] demonstrated utilization of description logics to represent and retrieve design for additive manufacturing problems to support a new process planning. Udoyen [85] demonstrated usage of description logics to represent and retrieve finite element analysis models for electronics package to support a new finite analysis model.

2.3.2 Semantic Retrieval

Of critical importance to the field of design repositories is that of efficient retrieval of engineering knowledge from the repository [85, 75, 86]. Semantic retrieval, also termed content-based retrieval, is founded in the use of semantically-rich representations and associated algorithms to facilitate retrieval [85]. As stated by Udoyen, to ensure precise queries, retrieval must be based on a definition of relevance that reflects the user's conceptualization and intended data use. To overcome the challenges with retrieval of relevant information from large, complex information

repositories, research into different semantic retrieval methods has been prevalent in the field of repository structuring and development.

Semantic retrieval methods can be classified into three main categories [87], including: distance-based method, indexing methods, and hybrids. As reviewed by Udoyen [85], distance-based methods compute a semantic distance between concepts by measuring the distance between the attributes of the concepts. The semantic distance can be calculated using ad hoc routines or operations using mathematical routines such as a feature vector. On the other hand, indexing methods are based on the creation of indexing structures that represent and organize the information to be retrieved. These methods also support reasoning about the structures.

In reviewing semantic retrieval methods, Udoyen [85] concludes that distance-based methods are overall unsuitable for retrieval where extensibility is needed. Vector-based methods are not easily extensible and are most useful for small, stable vocabularies. These methods rely on comparisons of vectors of the same size, and expansion of the vocabulary entails updating the vector length for every term in the vocabulary. The change in length of the feature vector makes recomputation of semantic distances between defined concepts intractable for large vocabularies. Ad hoc computational methods, such as those used in semantic nets, are also limited with respect to extensibility due to the high cost of computing semantic distance when adding large numbers of concepts.

Indexing methods, on the other hand, rely on symbolic representations of information. These representations can be manually created or automatically extracted from documents. These methods preclude the use of simple mathematical operations to determine relevance and shifts the emphasis in retrieval to the efficient reduction of the number of options accessed, while retrieving the most relevant [85]. These classification-based searches are efficient, as long as the classification hierarchies can be easily and consistently expanded.

Description Logics (DLs) is a formal and well-understood indexing-based approach to semantic retrieval. In the following section, referenced from Yim [75], is a brief introduction to Description Logics.

2.3.3 Description Logics

Description logics are knowledge representation formalisms that represent domain specific concepts and their relationships by first defining the relevant concepts of the domain (its terminology), and then using these concepts to specify the properties of objects and individuals occurring in the domain. The description logics can be viewed as formal languages for representing knowledge and reasoning about it. Among the many things that description logic provides, description language and inference algorithms are relevant to this research. The description language is used to define and manage concepts and their relationships. The inference algorithms are used to determine the relationships between concept descriptions. The basics of DL are described in the following section.

2.3.3.1 Basics

In description language, elementary descriptions are atomic concepts and atomic roles. Complex descriptions can be built inductively from these by using concept constructors. Description logics provide the attributive language ($\mathcal{AL}$) and other languages of this family are extensions of $\mathcal{AL}$. Concept descriptions in $\mathcal{AL}$ are formed according to the following syntax rules:

$C, D \rightarrow A$	(atomic concept)
$\top$	(universal concept)
$\bot$	(bottom concepts)
$\neg A$	(atomic negation)
$C \sqcap D$	(intersection)
$\forall R.C$	(value restriction)
$\exists R.\top$	(limited existential quantification)

where A denotes atomic concepts, R denotes atomic roles, and C, D denotes concept descriptions. The expressive power can be further enhanced by the following constructors:

$\mathcal{U} \rightarrow C \sqcup D$	(union of atomic concepts)
$\mathcal{E} \rightarrow \exists R.C$	(full existential quantification)
$\mathcal{N} \rightarrow \geq n\mathrm{R}, \leq n\mathrm{R}$	(number restriction)
$C \rightarrow \neg\mathrm{C}$	(negative for arbitrary concepts, "complement")

Extending $\mathcal{AL}$ by any subset of the above constructors yields a particular $\mathcal{AL}$ language [88]. Their names are $\mathcal{AL}[\mathcal{U}][\mathcal{E}][\mathcal{N}][C]$. The concept descriptions using description logics are constructed by determining base symbols for atomic concepts and roles first. Then, the set theory constructors are used with atomic concepts and roles to describe more specific and complex concepts. In this research, to balance the trade-off between expressive power and computational complexity, the attribute language with full existential quantification ($\mathcal{ALE}$) is used. The inference algorithms that are relevant to this research are satisfiability and subsumption.

Satisfiability algorithm determines the logical soundness of concepts with respect to terminologies. When the domain specific concepts are modeled, terminology is constructed by defining new concepts, possibly in terms of other concepts that have been defined before. During this process, a newly defined concept is checked to determine whether the concept makes sense or whether it contradicts existing concepts. The satisfiability algorithm tests the newly defined concept by determining whether there is some interpretation that satisfies the axioms of the terminology such that the newly defined concept denotes a nonempty set in that interpretation.

Subsumption is an algorithm that determines whether one concept or role is more general expression of another concept or role. For example, a concept C subsumes concept D if every member of concept D is also a member of C [89].

2.3.3.2 Utilization example

Figure 2.12 displays a simple example of description logics representation of concepts woman and mother[88]. Also, it presents subsumption reasoning procedures that determine their subsumption relations.

Atomic concepts: Person, Female

Atomic roles: hasChild

Concept description of woman and mother:

Woman ≡ Person.Female

Mother ≡ Woman ⊓ ∃hasChild.Person

Subsumption reasoning:

Query: Mother ⊏ Woman ?

Proof:

Mother ⊏ Woman → Mother ⊓ ¬Woman = ∅

Substituting definitions for Mother from concept descriptions, above becomes

(Woman ⊓ ∃hasChild.Person) ⊓ ¬Woman = ∅

(Woman ⊓ ¬Woman) ⊓ (∃hasChild.Person ⊓ ¬Woman) = ∅

Due to Woman ⊓ ¬Woman = ∅, above equation is true

Therefore, Mother ⊏ Woman is true

Figure 2.12 Description Logic representation example [75]

In Figure 2.12, the atomic concepts and roles are defined. For this example, Person and Female are chosen for atomic concepts. Also, hasChild is selected as atomic role. Then, mother and woman are defined using atomic concepts, role and set theory operators including full existential operator (∃) and intersection operator (⊓). For example, Woman is defined as something that is a Person and Female (ie. intersects Person and Female). Also, Mother is defined as something that is a Woman and something that has a person as its child. Then, the subsumption reasoning is presented. The set of presented procedures is called a tableau algorithm. It reduces subsumption to satisfiability. For example, the statement "Mother subsumed by Woman" is reduced to a statement "Mother intersect with not Woman is null". Using the tableau algorithm, the subsumption relation between Mother and Woman is determined as Mother ⊏ Woman, or a mother is a woman.

2.3.4 Research Opportunity

One of the key difficulties in the use of biological analogies in engineering design is that of identification of relevant biological design solutions. To overcome these difficulties, in this research, a strategy repository is used to capture biological and engineering design solutions and allow retrieval of these solutions in the conceptual design process. To enable efficient retrieval of these solutions from the repository, an ontology is structured and encoded using Description Logics. The foundations for ontologies and Description Logics was discussed in this section. The development of this repository is discussed in Chapter 5.

2.4 EVALUATION TECHNIQUES

Evaluation of idea generation techniques can be broadly grouped into two categories: process-based and outcome-based. Process-based approaches seek to evaluate idea generation by the occurrence of cognitive processes inherent to creative thought. Protocol studies are commonly used in process-based approaches. One such study, using the "think aloud" protocol, asks the designers to think aloud in idea generation, while being videotaped. It should be noted that there are no commonly agreed upon techniques to conduct and analyzed the data from these protocol studies. Due to the inherent complexity and subjectivity in using process-based approaches, outcome-based evaluation approaches been used [35]. Outcome-based approaches seek to evaluate the ideation process on the designs (outcomes) produced by the designers during ideation exercises. The premise of outcome-based approaches is that an idea generation technique is considered effective if its use results in 'good' ideas, with specific metrics being used to relate goodness of design ideas to the performance of the idea generation technique [35].

There have been several metrics used to evaluate the performance of idea generation techniques, including the total number of design ideas generated, the total number of categories of design ideas generated, the uniqueness or novelty of design ideas, and the practicality of design ideas. However, the most comprehensive set of

metrics proposed for evaluating idea generation were put forth by Shah et al.,[35] in which the authors identify four key metrics for evaluating the exploration and expansion of design space by a given designer: novelty, variety, quality, and quantity. Design space can be thought of as a hypothetical space encompassing all possible solutions to a given problem [90]. Novelty was defined as the degree to which a given design concept was unusual relative to other ideas, including those from other individuals. Variety was defined as the degree to which the concepts from a single designer were dissimilar from one another. Quantity was simply the number of different concepts generated by a designer. Higher scores for novelty, variety, and quantity implied greater exploration of the design space during ideation exercises. Quality was a somewhat subjective measure of the degree to which a concept was feasible and met design specifications. These will be discussed in the following section.

2.4.1 Evaluation Metrics for Idea Generation

2.4.1.1 Novelty

To assess novelty, the design problem is first decomposed into its key functions or characteristics. Next, each design idea is categorized on the basis of the solution method, or principle, used to address the key functions and characteristics of the design problem. Finally, a count of the number of instances of each solution method is taken and the overall novelty for each idea calculated. Using this method, the lower the count of instances of a solution method used in an idea, the higher the novelty score for that idea. Overall novelty for each idea can be calculated from the following equations:

$$N = \sum_{j=1}^{m} f_j \sum_{k=1}^{n} S_{1\,jk} P_k \qquad \text{Equation 2.1}$$

where N is the overall novelty score for an idea with m functions or attributes and n stages; f_j is a weight assigned according to the importance of each function or

characteristic; p_k is the weight assigned to stage k, where stage k is the stage at which the function is addressed; S_{1jk} is given by the following equation:

$$S_{1jk} = \frac{T_{jk} - C_{jk}}{T_{jk}} \times 10 \qquad \text{Equation 2.2}$$

where T_{jk} is the total number of ideas produced for function j and stage k; C_{jk} is the count (number of ideas) of the current solution for that function.

The novelty scores for each idea are then averaged to compute a novelty score for each participant. Such a measure of novelty by frequency of occurrence was shown to be similar to subjective novelty scores assigned by external judges [35].

2.4.1.2 Quality

The quality metric is used to assess the technical feasibility and performance of a set of design ideas. These ideas are evaluated using both analytical and experiential knowledge. Shah and his colleagues recommend using domain specific means to determine key characteristics for performance, then evaluating the design ideas based on these characteristics. The quality scores for all the alternatives are then summed for all design ideas to get a total score for the set of ideas. The quality score is calculated using the following equation

$$Ql = \sum_{j=1}^{m} f_j \sum_{k=1}^{2} S_{jk} p_k \Big/ n * \sum_{j=1}^{m} f_j \qquad \text{Equation 2.3}$$

where Ql is the overall novelty score for an idea with m functions or attributes and n stages; S_{jk} is the quality score for function j at stage k; f_j is a weight assigned according to the importance of each function j; p_k is the weight assigned to stage k; the denominator normalizes the score to a scale of 10.

2.4.1.3 Variety

Shah and his colleagues developed the variety metric to characterize the degree of difference within a set of designs generated by a designer, giving a score between zero and ten [35]. Measuring the variety requires first creating a genealogy tree of the solution approaches for each function being executed by the designed device. Solutions are first

differentiated among the hierarchical branches of the tree by the *physical principle* used to achieve the function. The next level of division occurred based on the *working principle* of the solution, followed by the *embodiment* of the solution, and then the *details* of the solution. After generating the tree, the number of ideas in each differentiated category is tabulated. Differentiation at higher hierarchical levels implies greater variety within the design set and is given higher point totals than differentiation at lower levels of the hierarchy. The total variety score is given by the equation

$$V = \sum_{j-1}^{m} f_j \sum_{k=1}^{4} S_k b_k / n \qquad \text{Equation 2.4}$$

where V is the variety score, m is the total number of required functions solved by the design, f_j is a weighting factor for the relative importance of each function, S_k is the score for hierarchical level k (Shah et al. suggest scores of 10, 6, 3, and 1 for the four levels, respectively[35]), b_k is the number of branches at hierarchical level k, and n is the total number of ideas in the set.

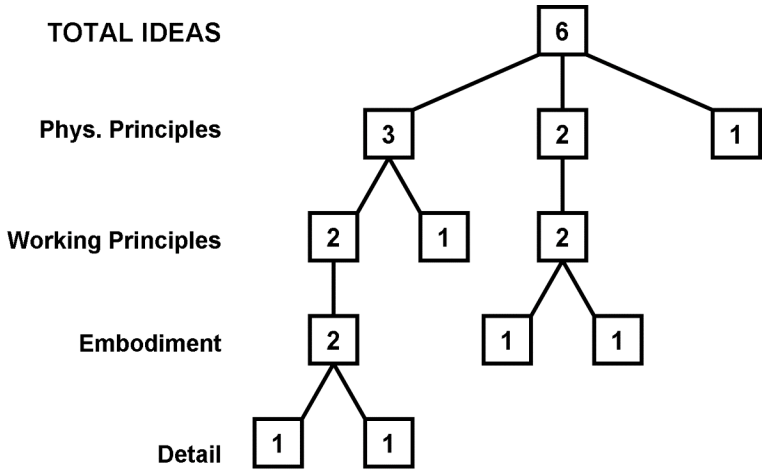


Figure 2.13 Example design genealogy tree for a set of 6 designs

As an example calculation, Figure 2.13 shows a sample genealogy tree for a set of 6 designs for a single function. The set of designs utilizes three separate physical principles to achieve the required function, resulting in 3 physical principle branches.

The set also includes 3 working principle branches, 3 embodiment branches, and 2 detail branches. Using Equation 1.4, variety would be calculated as

$$V = \frac{(10*3)+(6*3)+(3*3)+(1*2)}{6} = 9.83$$

Equation 2.5

2.4.1.4 Quantity

Quantity is the total number of ideas generated by a participant over the course of the study. Quantity is simply a count of the ideas.

2.4.2 Revised Metrics for Variety

In reviewing the metrics for variety in practice, several key shortcomings were found. These shortcomings are discussed in the following sections.

2.4.2.1 Lower Scores for Higher Variety

Figure 2.14 shows hypothetical genealogy trees for 2 sets of designs, with 3 designs in each set. In Genealogy A, the 3 designs utilize only 2 physical principles, with the third differentiation occurring at the working principle level. In Genealogy B, the 3 designs each utilize separate physical principles, which should be the maximum possible variety. However, applying Equation 2.4 yields a score of $V = 10.67$ for Genealogy A and $V = 10$ for Genealogy B. Thus, the higher variety of Genealogy B resulted in a lower overall variety score. Additionally, the metric was intended to scale from 0-10, and using Equation 2.4 to calculate the variety gives a score outside this range to Genealogy A.

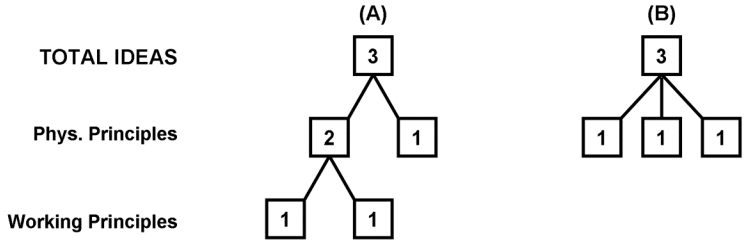


Figure 2.14. Higher variety can result in lower score

The shortcoming in Equation 2.4 comes from essentially double-counting design ideas. Note that in Equation 2.5 points are given $3+3+3+2=11$ times, yet the set consists of only 6 designs. The flaw can be resolved by counting the number of differentiations in design principles rather than counting the number of branches at each level. For example, 2 physical principle branches only correspond to a single differentiation between physical principles, and 3 physical principle branches corresponds to 2 differentiations, and so on. Thus the number of differentiations is always one less than the number of branches at a given hierarchical level of a given branch. No differentiations occur when a single branch emanates from a node.

Assigning points at nodes where differentiation occurs rather than counting the number of branches readily resolves the double-counting flaw. This modifies the variety metric to

$$V=\sum_{j-1}^{m} f_j\left(S_1(b_1-1)+\sum_{k=2}^{4} S_k \sum_{l=1}^{b_{k-1}} d_l/(N-1)\right) \qquad \text{Equation 2.6}$$

where the first term inside the parenthesis is the score for differentiation at the physical principle level, d_l is the number of differentiations at node l (one less than the number of branches emanating from node l), and 1 is subtracted from N to preserve the normalization from 0-10 since the maximum number of differentiations is one less than the number of designs. Points are given only when branches differentiate, and Equation 1.6 calculates the average level at which differentiation between ideas occurs.

Applying Equation 2.6 to the genealogies shown in Figure 2.14 yields $V=\frac{(10*1)+(6*1)}{2}=8$ for Genealogy A and $V=\frac{(10*2)}{2}=10$ for Genealogy B. As a more complex example, applying Equation 1.6 to the genealogy in Figure 2.13 yields a variety score of $V=\frac{(10*2)+(6*1)+(3*1)+(1*1)}{5}=6$, which is a better indication of the displayed variety. Using Equation 2.4 yielded a variety score close to 10, which should be reserved only for genealogies with the majority of the design differentiation occurring at the physical principle level. Note that in the described calculation, nodes

without differentiation are simply ignored, as they make no contribution to the variety score.

2.4.2.2 Normalizing a Group Score

Variety can only be calculated for a set of multiple design ideas, unlike novelty, which can be calculated for a single design. An average novelty score for a set of designs is therefore a relevant metric for a set of designs, whereas an average variety score per design is not, as the variety score only applies to the set itself. Figure 2.15 demonstrates the flaw encountered by normalizing the variety score by the number of designs. Genealogy D can be viewed as an expansion of Geneology C since since Genealogy C could be a subset of Genealogy D. However, using Equation 2.6, the variety scores for Genealogies C and D are 10 and 8, respectively. Genealogy D added more physical principles, working principles, and designs to Genealogy C, thus demonstrating greater exploration of the design space yet receiving a lower variety score. A non-normalized variety score would measure actual design space exploration, applying to the entire set of ideas rather than averaged per idea. Variety would then be calculated as

$$V = \sum_{j-1}^{m} f_j \left(S_1 (b_1 - 1) + \sum_{k=2}^{4} S_k \sum_{l=1}^{b_{k-1}} d_l \right) \qquad \text{Equation 2.7}$$

Changing the values of S_k to 10, 5, 2, and 1 assures that at least two ideas at one hierarchical level must be added to equal the variety gain by adding a single idea at the next higher hierarchical level. Using Equation 2.7 and the new values for S_k, the variety scores for Genealogies C and D become 10 and 30, respectively, giving a more accurate representation of their relative degrees of design space exploration. Not normalizing the variety in Equation 2.7 also eliminates the need for the quantity metric, as Equation 2.7 incorporates the quantity of designs by not normalizing.

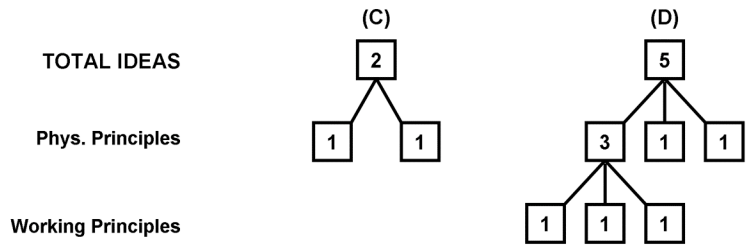


Figure 2.15. Normalized variety score can penalize greater actual variety.

2.5 CLOSURE AND VALIDATION

The role of Chapter 2 was to lay the theoretical foundation to support the method for Reverse Engineering Biological Systems and the engineering ontology developed in this work. The four foundational constructs of systematic design and idea generation (Section 2.1), engineering representations (Section 2.2), ontology development (Section 2.3), and evaluation metrics (Section 2.4) were presented. The literature supporting these constructs was also presented and reviewed.

In Section 2.1, systematic design and idea generation was reviewed. The Pahl and Beitz systematic method was reviewed in Section 2.1.1, followed by a review of idea generation techniques in engineering design in Section 2.1.2. Analogical reasoning, the idea generation technique used in this research, was also reviewed in Section 2.1.3. It was concluded that due to their large analogical distance from the engineering domain, biological systems provide a good source of analogies for innovative design.

In Section 2.2, representations in engineering design are reviewed. In Section 2.2.1, general models of cognitive processing in creativity were reviewed, followed by a review of the role of representations in creative cognition in Section 2.2.2. In Section 2.2.3, specific representations used in engineering design were reviewed. The backbone of the proposed method for Reverse Engineering Biological Systems is the hierarchical Petri net representation. Biological strategies are extracted from these representations and used to inspire new and innovative design solutions in Conceptual Design. In engineering design, representations are used throughout the design process to create design artifacts. In this research, complete representations (including functional, behavioral and structural information) of systems are needed to aid in manipulating and

understanding the related biological behavior and strategy. The hierarchical Petri net representation is developed in Chapter 3.

In Section 2.3, engineering ontology development is reviewed. Specifically, design repositories are reviewed in Section 2.3.1, followed by semantic retrieval and Description Logics in Sections 2.3.2 and 2.3.3, respectively. In this work, ontologies are used to build a repository of biological and engineering strategies. Semantic retrieval strategies were reviewed and indexing-based approaches were found to be more efficient and accurate at retrieving strategies from these ontologies. One such method, Description Logics, was reviewed and used in this work to identify relevant biological strategies.

In this work, the value of bio-inspired design is evaluated empirically through cognitive studies. Evaluation metrics for idea generation techniques, such as the use of biological strategies proposed in this work, were reviewed in Section 2.4. These metrics, reviewed in Section 2.4.1, include novelty, variety, quality, and quantity metrics. The variety metric is refined in Section 2.4.2.

Validation Strategy: Theoretical Structural Validity

The validation strategy for this dissertation is presented in Figure 2.16.

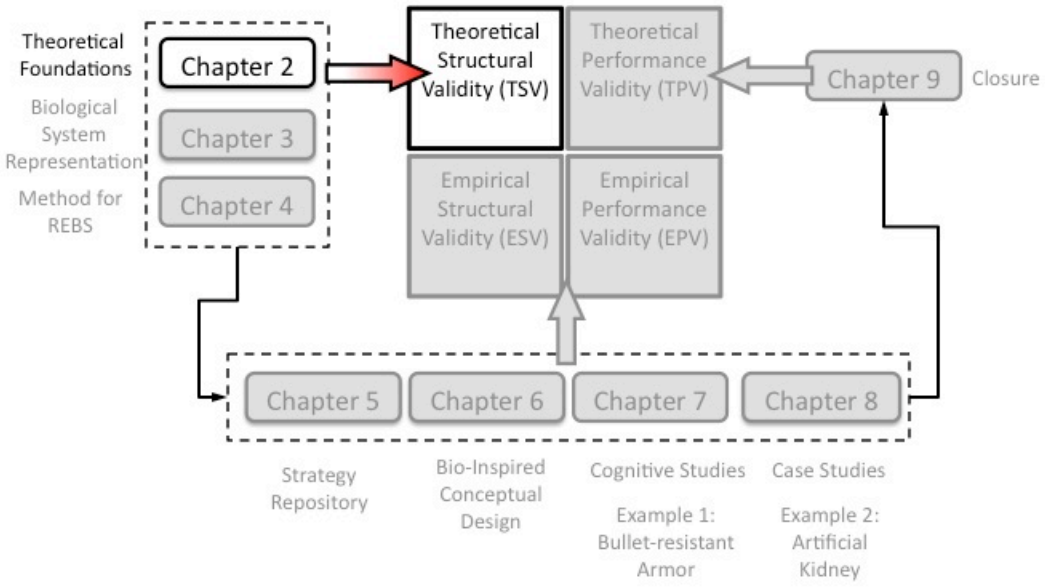


Figure 2.16 Validation for Chapter 2

As presented in Section 1.4, Theoretical Structural Validation (TSV) involves checking the individual constructs and assumptions upon which the method is built, as well as checking the internal consistency of the method when combining the individual constructs. In this chapter, the theoretical foundations of the method for Reverse Engineering Biological Systems and the engineering ontology were validated through review of the relevant literature. In Chapter 3, the specific representation used in the proposed method, the hierarchical Petri net representation is validated. In Chapter 4, the latter part of TSV will be considered, where the internal consistency of the method when combining the individual constructs will be evaluated.

CHAPTER 3 BIOLOGICAL SYSTEM REPRESENTATION

In Chapter 2, the theoretical foundations of the method for Reverse Engineering Biological Systems were presented. In this chapter, the backbone of the proposed method, biological system representation, is presented. The role of this chapter to the dissertation as a whole is presented in Figure 3.1.

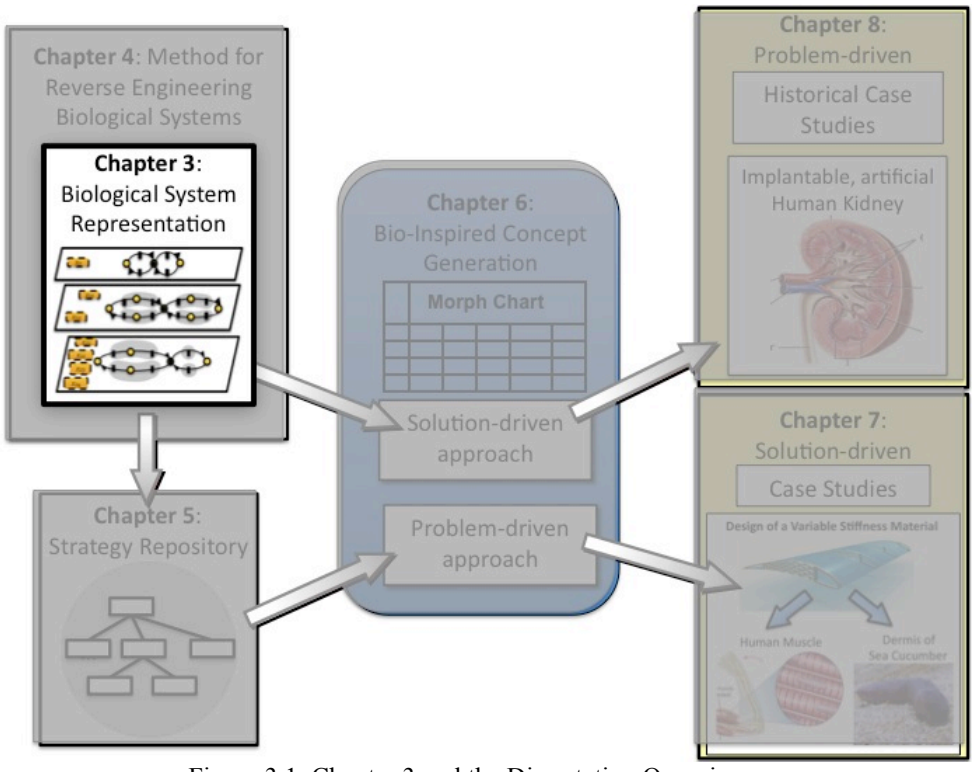


Figure 3.1 Chapter 3 and the Dissertation Overview

The goal of this research is to aid the designer in the ideation process through the use of biological strategies. Given this aim, the designer must first be able to systematically extract correct strategies from the biological systems of interest. To aid in extracting this strategy, representations can be used to (1) filter out unimportant information and present the designer with information relevant to the given task and (2) aid the designer in reasoning about the system. In this chapter, the hierarchical

representation used to represent biological systems is developed as a means for extracting behavioral strategies. In this research, biological strategies are viewed as refinements of behavior, where specific physical phenomena driving a particular behavior (and function) are identified as the underlying behavior used to accomplish the function of the system of interest.

For example, in the context of an engineering system, consider the function of a garage door opener. The function can be defined as "send control signal to the garage door unit". By examining the garage door opener as a system, the behavior can be simply defined as "based on a hand pressing a button, the remote produces an IR signal, which controls the garage door". At this level of abstraction of behavior, there is not much knowledge about strategy that can be extracted from the system. By further decomposing the behavior, the underlying mechanism that actually converts the hand input to the IR signal can be viewed. This strategy can be extracted by viewing the system at multiple levels of abstraction, examining the behavior of subsystems and components of the system. For instance, the garage door opener includes a controller chip, DIP switch, power source, transmitter, etc. By viewing the behavior of these subsystems, and how they impact the behavior of the top level system, a much richer description of system behavior can be extracted.

In this research, behavioral strategies are systematically extracted using a hierarchical representation of the biological system, allowing the system to be viewed at multiple levels of abstraction. Specifically, in this chapter, the following research question is considered:

RQ1: *"What type of representation can be used to model the behavior of biological systems?"*

To answer this question, Hypothesis 1 proposed in Chapter 1 is as follows:

Hypothesis 1: A representation based on (1) a causal behavioral description and (2) hierarchical Petri nets can be used to model the behavior of biological systems

This hypothesis is validated using the following procedure:

1. Identify several defining characteristics of biological systems (Section 3.1.1).
2. Define requirements for representation of biological systems (Section 3.1.2)
3. Review several knowledge representation formalisms for modeling engineering systems (Section 3.1.3), and evaluate them against the biological representation requirements (Section 3.1.4)
4. Develop a knowledge representation framework (Section 3.2)
5. Define requirements for expressing the biological representation (3.3.1)
6. Review and evaluate several representation expressions versus the requirements (Sections 3.3.2 and 3.3.3)

Lastly, develop a hierarchical Petri net representation for biological systems and evaluate versus representation criteria

3.1 REPRESENTING BIOLOGICAL SYSTEMS

In this section, we characterize biological systems and evaluate traditional representation methods in the context of representing biological systems.

3.1.1 Biological System Characterization

There are several characteristics of biological systems that make them extremely difficult to represent, as opposed to traditional engineering systems, using traditional methods of representation, including:

- ***Complexity and Hierarchical arrangement*** – Biological systems are arranged and organized hierarchically, meaning that systems contain subsystems that contribute to their overall behavior. This hierarchical arrangement is used to cope with the large complexity inherent to biological systems [91]. Jagers op Akkerhuis [92] comments, "the organization of nature is profoundly hierarchical, because from its beginning, interactions between simple elements have continuously created more complex systems, that themselves served as the basis for still more complex systems".

Consider the example of human muscle displayed in Figure 3.2. Human muscle contains groups of muscle fiber bundles. These muscle fiber bundles consist of many muscle fibers, whose basic unit is that of the myofibril. The myofibril is made of groups of actin and myosin myofilaments.

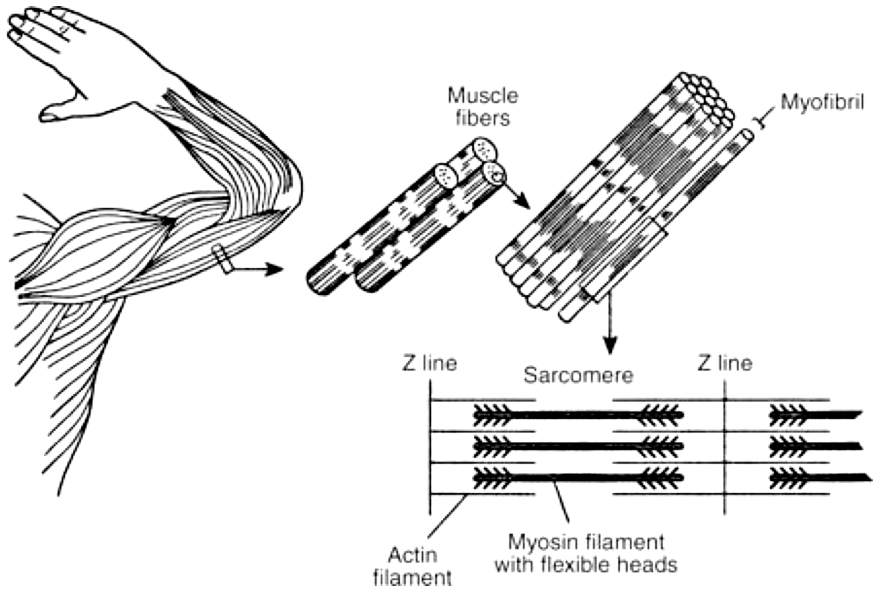


Figure 3.2 Diagram of Human Muscle[93]

This hierarchical arrangement helps assure system robustness, meaning that lower levels of the system hierarchy cope with the changing environment, while keeping the external properties of the system the same.

- *Dynamic (living)* – Biological systems are dynamic, meaning the properties of these systems change with respect to time. The change of state is usually a result to a change in an external environmental condition. For instance, human bone has the ability to adapt to varying mechanical loading conditions by either locally adding or removing mass and by changing shape. Another example is that of tropic movement in many plants found in nature. Many plants exhibit tropic movement in response to environmental stimuli, including sunlight, chemical, touch, etc. A change in internal pressure allows this movement.
- *Multifunctional* – Many biological systems can be characterized as being multifunctional, meaning that multiple functions are carried out by one system.

Since resources in nature are limited, by sharing resources, multifunctionality helps to reduce the resources needed for a given function. In the words of McShea [94], some overlap is expected as it allows an economical use of parts, and therefore favorable in natural selection. An example of multifunctionality can be seen in the human nose, which functions simultaneously as a sensory and respiratory feature.

- *Integrated Architecture* - With respect to integration, many-to-one mapping also plays a significant role in biological systems. As stated in our previous discussion on hierarchical organization, integration, or the coupling of many systems to accomplish a given function, has large implications in reducing biological complexity.

3.1.2 Representation Requirements

When designers cannot experience phenomena directly, representations can play a crucial role in helping them understand and reason about the phenomena [95]. In essence, a representation is sought that can accurately cope with biological systems and their inherent properties of hierarchy, dynamic behavior, and multifunctionality/integration. In this research, this representation will be used to aid designers and engineers in understanding and extracting biological design strategies. These design strategies will then be used to stimulate the generation of new and novel design ideas for the engineering domain.

Based on these characteristics, the following requirements were derived and are presented as follows:

- *Hierarchical Representation* – Because of their ability to simplify and systematize complexity, hierarchical representations have been commonly used by biologists to represent complex, biological systems [96]. Hierarchical representations have the advantage of "implicitly incorporating abstraction and refinement[97], thus making the systems easier to study. Because of the inherent complexity of biological systems, the representation must explicitly allow for hierarchical arrangement. Jagers op Akkerhuis [92] comments that scientists have attempted to

capture the essence of this complexity in easy to understand hierarchies, including those levels defined by Miller [98] as cell, organ, organism, group, organization, community, society, and supranational system. This requirement implicitly allows for representation of integrated biological systems.

- *Dynamic System Representation* – Since biological systems are "living", our representation must allow for a dynamic, causal behavioral model to be explicitly represented. This causal behavioral model establishes the flow of causality throughout the system. This requirement also specifies an explicit representation of the state, or change of state, of the system of interest.
- *Explicit Representation of Working Environment* – The working environment is defined as other surrounding systems external to the system boundary. Because of their dynamic nature, biological systems react to and buffer environmental inputs. Given the high level of environmental interaction of these biological systems, the representation must explicitly represent external relations (inputs and outputs) to and from its working environment.
- *Behavior-centric approach* – The purpose of reverse-engineering is to map the structure of a system to a function of that system. We use behavior as a means of extracting this functional information from the structure of the system. Therefore, our representation must utilize a behavior-centric approach, allowing for analysis of the system in an objective fashion. This behavior-centric approach allows a representation independent of the so-called "purpose" of the system and allows us to better separate the objective and subjective views of the system. The objective views of a system include views of the behavior and structure of the system, without ascribing a specific purpose to this behavior and structure. The subjective view, on the other hand, include some forms of function in which a purpose is ascribed to the system.
- *Completeness of representation* – In our representation, we wish to include information about what the system is (structure), what the system does (behavior), and "why" it does what it does (function). A complete system representation includes structural, behavioral, and functional views of the system, as well as the

relationships and mapping between the different views [61]. The completeness is measured by its accounting of these views of the system.

- *Uniqueness of Representation* – Uniqueness implies the ability to represent a system in a "single" way [61]. To achieve uniqueness, the system must be represented objectively, and minimize subjective treatments of function and behavior.

3.1.3 Existing Knowledge Representations in Design

In this section, several existing knowledge representation frameworks for engineering systems (reviewed in Section 2.2.3.3) currently employed in the design and artificial intelligence research communities are highlighted. These include that of the Structure-Behavior-Function (SBF) model [16-20], Function Behavior-State (FBSt) model [62-64], Functional Rationale (FR) [65], Function-Behavior-Structure (FBS) model [67-69], Function-Environment-Behavior-Structure (FEBS) model [34], and the Causal Behavioral Model (SAPPhIRE) [24].

In the Section 3.1.2, requirements for adequately representing biological systems was set forth. In the next section, the current knowledge representations for engineering systems are evaluated for feasibility in representing biological systems.

3.1.4 Comparison of Representations against Requirements

In this section, the current knowledge representation frameworks are evaluated against the criteria for representing biological systems.

1. *Hierarchical Representation* – With respect to hierarchical representation, none of the knowledge representations explicitly represent hierarchy, allowing for view of the system and multiple levels of abstraction.
2. *Dynamic Representation* - With respect to dynamic representation, all the representations except that of FBS and FEBS utilized some sort of causal behavioral process and state change to represent the behavior of the system. The FBS representation lacks an explicit representation of causality in the

frameworks, while the FEBS representation lacks an explicit representation of the state of the system.

3. *Environmental Representation* – With respect to an explicit representation of the environment, the FEBS and SAPPhIRE representations were the only knowledge representations to explicitly represent environmental inputs and outputs. FEBS represents the environment as driving inputs and functional outputs to the system of interest, while SAPPhIRE implicitly represents the working environment in the form of inputs to the causal behavioral model.
4. *Behavior-centric approach* - As for behavior-centric approaches, the SBF, FEBS, and SAPPhIRE employ a behavior-centric representation, allowing for objective representation of the system for analysis. The remaining representations employ a more subjective, function-based approach.
5. *Completeness* - All the knowledge representations were complete in the sense that they allowed for representation of structure, function, and behavior, and the relations between, however, SAPPhIRE implicitly represents these views.
6. *Uniqueness* – FBSt, FR, and FBS lacked unique representations, as they all focused on a subjective representation of function, allowing multiple interpretations of a given device or system.

The evaluation of these systems is summarized in Table 3.1 below.

Table 3.1 Evaluation of existing knowledge representation frameworks

Requirements / Knowledge Representation Frameworks	Structure-Behavior-Function (SBF)	Function-Behavior-State (FBSt)	Functional Rationale (FR)	Function-Behavior-Structure (FBS)	Function-Environment-Behavior-Structure (FEBS)	SAPPhIRE
Hierarchical Representation	✗	✗	✗	✗	✗	✗
Dynamic Representation	✓	✓	✓	✗	✗	✓
Environmental Representation	✗	✗	✗	✗	✓	✓
Behavior-centric approach	✓	✗	✗	✗	✓	✓
Completeness of Representation	✓	✓	✓	✓	✓	✗/✓
Uniqueness of Representation	✓	✗	✗	✗	✓	✓

In Section 3.1.2, the requirements for the representation of complex, biological systems were developed. As can be seen in Table 3.1, none of the existing knowledge representations meet all the requirements for representation of these systems. They specifically lack in the representation of the hierarchy of the biological systems. The causal behavioral model, SAPPhIRE did however meet most of the requirements for representation of biological systems, but lacked in the explicit representation of the working environment as well as the completeness of the representation. In this work, we utilize a causal behavioral process model similar to that of SAPPhIRE to represent our system of interest but improve on many of the shortcomings of this type of representation.

3.2 HIERARCHICAL REPRESENTATION DEVELOPMENT

In the previous section, current knowledge representation frameworks were evaluated against the requirements put forth for representing biological systems and several shortcomings were identified. In this section, we improve on the shortcomings of these representations and develop a hierarchical representation suitable for representing biological systems. We believe this hierarchical systems view aids in exploring the complexity of biological systems, especially with respect to behavior. We also believe this representation can aid in the systematic extraction of functional strategies of biological systems.

We begin this section by defining our view of a "system". In Section 3.2.2, we present our causal behavioral description, which is used as the foundation for the proposed hierarchical representation.

3.2.1 What is a 'System'?

In this research, we view a system similarly in terms of (1) the system itself, (2) its supersystem, and (3) its subsystems. The system itself is the specific level of interest of the identified behavior or function. We identify this system by a boundary. Everything external to this boundary is considered the system's working environment. This includes all other systems that the system of interest interacts with. The system and its environment combine in forming the system's supersystem. The system itself can be decomposed into its subsystems. We define subsystems as lower-level systems and components, within the system boundary, that contribute to the function and behavior of the system. These sub-systems can also be decomposed into their sub-systems in an iterative fashion.

We now turn to the definition of the system and its super- and subsystems. In this research, we define systems following the view of McShea [94, 91, 99], where object parts are used to define the hierarchy of parts in simple biological systems. These systems function in an integrated fashion with little interaction (relatively) with surrounding systems. We define these systems by a high level of internal interaction and low level of external interaction (see Figure 3.3) [91, 99]. In Figure 3.3, the small circles represent

parts and the arrows represent interactions between them. The thickness of the arrows denote the strength of interaction.

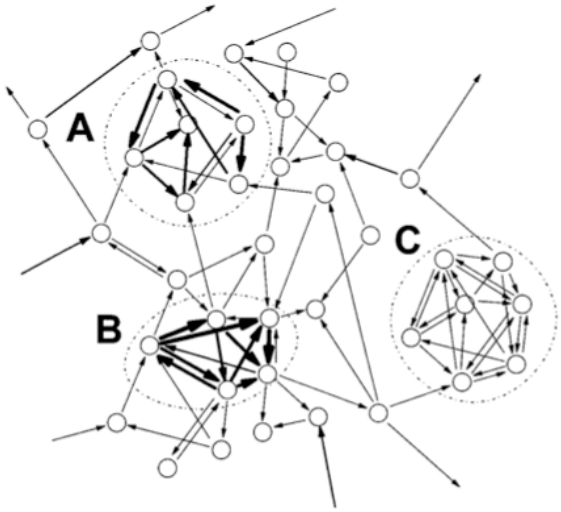


Figure 3.3 Definition of a 'system' [99]

In Figure 3.3, the dashed lines are definitions of what we consider a system based on tight integration and isolation. System A is defined as a system because although it has weak internal interactions, they are many compared to its external interactions. System B has few internal interactions, but the strength of these interactions is high relative to that of its external interactions. System C is defined as a system because of its large number of internal interactions compared to that external to the system [99].

This systems view is also similar to that of modules by Wagner [100]. Modules also function in an integrated fashion (large internal interaction), with little external interaction with surrounding systems (isolated). The difference between object parts and modules is that parts are units in the operation of the organism[99], whereas modules relate to the evolution or development of the organism.

3.2.2 Causal Behavioral Description

Based on this systems view in Section 3.2.1, our causal behavioral description (CBD) can now be defined. This CBD is used as the foundation of our biological system representation. In our causal behavioral description, we wish to link structure, behavior,

and function in a representation where each can be easily extracted in a systematic fashion. In this research, we define structure as the entities of interest and the interactions or relations between these entities. Each entity is defined by a set of properties, or attributes. Behavior is then defined as the intrinsic change of state of these attributes. We define function as a mapping of the behavior of a system to the behavior of its supersystem [101], as well as the mapping of a subsystem to that of its system. In other words, we view function as the effect of a component on its working environment (ie. other components), which makes up the supersystem. Based on this view of structure, behavior, and function, we define the causal behavioral description. To do so, we define the following six individual constructs:

1. *System:* The set of physical components and interactions between these components. These interactions can be either (1) flows of energy, material, and signal or (2) physical interactions between components.
2. *Working Environment:* The working environment is defined by the boundary of the specific entity or system as all entities or systems external to the defined boundary. It includes "environmental elements that contribute to the product's functions" [34], such as temperature, force, etc.
3. *Driving inputs:* the driving input is considered the flow of energy, material, or signal needed to activate the physical phenomenon that causes the change of state of the system of interest. Driving inputs originate from the working environment. [34]
4. *Functional output:* Functional output is the output flow of energy, material, or signal from the system resulting from the change of state. Functional outputs affect the system's target enviornment. [34]
5. *State:* the value of the system attributes (or specific characterics) at a given instant of time.
6. *Physical phenomena:* The physical phenomenon is considered the action governing the change of state of the system.

The relationship between these constructs is displayed in Figure 3.4.

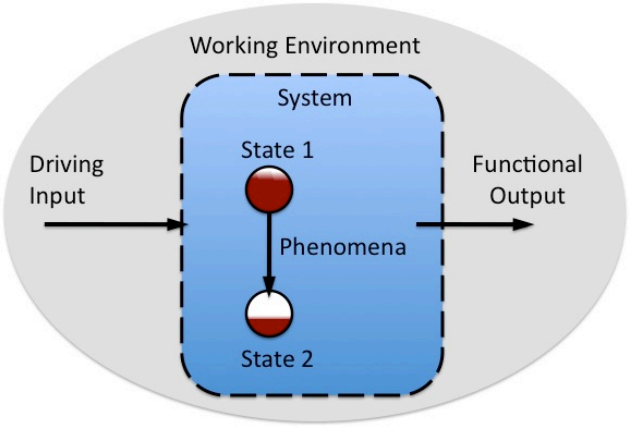


Figure 3.4 Causal Behavioral Description

As seen in Figure 3.4, *systems* operate in a *working environment*. *Driving inputs* from the source environment activate *physical phenomena*. *Physical phenomena* create a change of *state* in the *entity*. The change of *state* of the system creates *functional outputs* to the target environment.

Using the definitions put forth earlier in our discussion, structure, function, and behavior can easily be extracted from the CBD. Structure is defined as our system of interest and its working environment. Behavior is defined as the states of the system and the physical phenomena causing the change of the state. Function is defined as a mapping of the behavior of the system to that of its supersystem [54]. We define the function in the CBD framework as a set containing the driving input and the functional output of the system. From this, we see how the representation is complete with respect to explicitly representing function, behavior, and structure. For clarity, consider the following piston-cylinder assembly example in Figure 3.5.

Piston-Cylinder Example

The system in question is the gas inside the piston-cylinder assembly, whereby the boundary is denoted by the dashed line. Heat (Q_{in}) is being added to the system, causing the gas inside the system to expand. The expansion of the gas forces the piston upward, thus causing work (W_{out}) to be transferred to the environment. In the automotive context, the piston-cylinder (*system*) operates in an internal combustion engine (*working environment*). Heat (*driving input*) causes the gas to expand (*change state*) inside the

piston cylinder assembly following the First Law of Thermodynamics and the Ideal Gas Law (*physical phenomena*). This expansion causes work (*functional output*) to be done on the connecting rod, which turns the engine crankshaft.

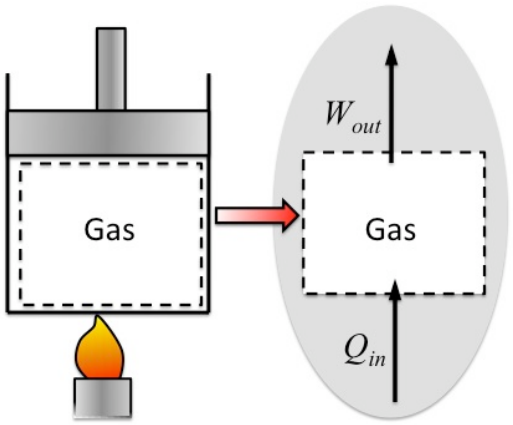


Figure 3.5 Piston-Cylinder Assembly

Based on the causal behavioral model of the piston-cylinder assembly, the function, behavior, and structure of the system can also be defined. Specifically, the structure is defined as the system of interest, or the piston-cylinder assembly. The behavior is defined as the change of the state of the system, or the expansion of the gas from State 1 (T,P,v) and State 2 (T, P, v). The function of the system is defined as the driving input and functional output of the system, [Thermal Energy, Mechanical Energy].

3.2.3 Hierarchical System Representation

Using the hierarchical view of systems discussed in Section 3.2.1, we can also construct a hierarchical representation based on the causal behavioral model. In this hierarchical systems view, the super-system is composed of the system and its working environment and the sub-systems compose the system of interest. By defining a causal behavioral model, it allows us to view a system at multiple levels of abstraction. In Figure 3.6, the behavior of the super-system can be viewed at the system level, whereby the system interacts with its working environment. The behavior of the system itself can also be viewed at the sub-system level. For each component in the system hierarchy, a causal behavioral model is defined. Multiple systems are linked by their respective

driving inputs and functional outputs. It should be noted that the functional outputs of one system are in fact the driving inputs of other systems on the same hierarchical level.

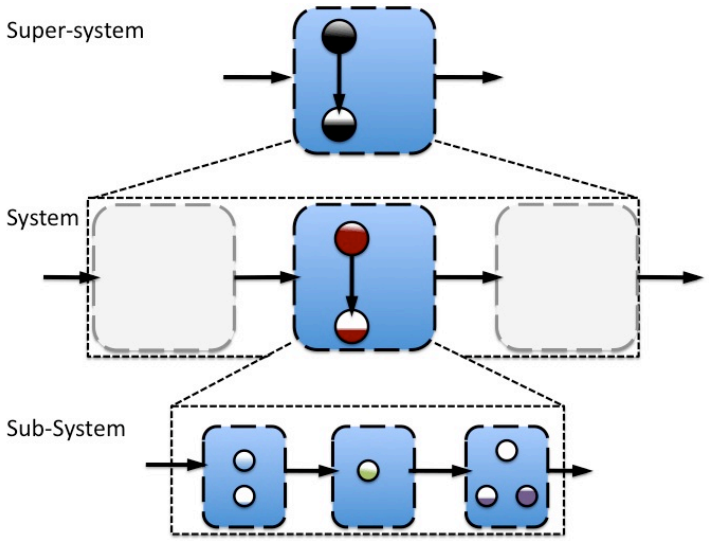


Figure 3.6 Hierarchical Causal Behavioral Model

Given the piston-cylinder assembly example in Figure 3.5, let us view the assembly as a system composed of the heat source, the gas inside the assembly, and the piston itself. Therefore, the causal behavioral model of the system can be viewed at the subsystem level as the behavior of the heat source, gas, and piston subsystems. In this case, the functional output of the heat source, heat, will also be the driving input for the gas, causing the expansion of the gas. The functional output of the gas, expansion or work, will be the driving input of the piston, causing it to change its relative position in the assembly. The system can also be viewed in similar fashion at the super-system level of the internal combustion engine.

3.2.4 Evaluation of Causal Behavioral Description

In this section, the CBD is evaluated against the criteria put forth earlier for the representation of complex, biological systems. This evaluation is displayed in Table 3.2.

Table 3.2 *Evaluation of CBD*

Hierarchical Representation	In Section 3.2.3, we define a hierarchical system representation that allows us to view the behavior of the system at multiple levels of abstraction
Dynamic Representation	The CBD allows for explicit representation of causality throughout the system, as well as explicitly reperesents states of the system.
Environmental Representation	CBD explicity represents functional inputs to and driving inputs from the working environment of the biological system
Behavior-centric approach	CBD is inherently behavior-centric
Completeness	As discussed in Section 3.2.2, function, behavior, and structure are explicitly represented in the CBD. Specifically, structure is defined as our system of interest and its working environment. Behavior is defined as the states of the system and the physical phenomena causing the change of the state. Function is defined as a set containing the functional output and driving input
Uniqueness	Structure, behavior, and function are objectively defined based on the behavior of the system. This objective view helps to assure uniqueness of representation.

As displayed in the table, the CBD meets all the requirements for a representation of a biological system, including that of hierarchical representation, explicit dynamic representation, explicit representation of the environment, behavior-centric approach, and completeness and uniqueness of representation. In Section 3.3, we define a representation, or an expression, of the CBD.

3.3 REPRESENTATION OF THE CAUSAL BEHAVIORAL DESCRIPTION

Now that the causal behavioral description has been put forth, we now turn to how this representation will be expressed in a manner to aid in the extraction of behavioral strategy from the system. In simpler terms, we define how this representation will look, or its expression. To do so, we define specific requirements for the expression (Section 3.3.1), as well as evaluate several commonly used representation expressions against these requirements (Section 3.3.2).

3.3.1 Expression Requirements

To aid in strategy extraction, we examine several different requirements for the expression of our causal behavioral model, including: cognitive offloading, inference, validity, consistent reasoning, isomorphism, model complexity, and model verification. These requirements are explored in further detail as follows:

1. *Computational offloading* - Computational offloading [102] is the extent to which a representation reduces the amount of cognitive effort required to solve a problem. Computation offloading is directly related to the amount and type of information that is presented explicitly in a representation.
2. *Inference* - Inference refers to the extent to which a representation allows the user to infer new knowledge, based on the existing information presented. Inference is also linked to graphical constraining, which refers to the way graphical information is able to constrain the types of inference that can be made about the represented world [102].
3. *Validity* – Validity refers to the extent at which a representation is based in theory and rigorously defined. This includes the extent to which the theoretical foundations of the representation have been researched and validated.
4. *Consistent Reasoning across abstraction levels [103]* - Most design representations operate at a single layer of abstraction. In order to model complex systems, hierarchical representations are needed to reduce and order the complexity. In the case of hierarchical system representation, the representation must allow for consistent reasoning across different levels of abstraction of the system, and its Causal Behavioral Description.
5. *Isomorphism* – Isomorphism can be described as the direct structural mapping relationship between two objects. For example, a wristwatch and a wall clock can be said to be isomorphic because there exists a direct mapping of time between the two objects. With this requirement, we require that there exists a direct mapping between the Causal Behavioral Description and that of the representation expression.

6. *Model Complexity* – Model complexity refers to the ability of a representation to model complex relationships between subsystems and components, such as precedence of action, synchronization, mutual exclusion of resources, and concurrency. [104]
7. *Model Verification* – One of the key issues in modeling systems is the ability to check the model for correctness. Model verification refers to the ability of a representation to be verified that it is indeed doing what it is put forth to do. This usually involves some form of qualitative simulation.

3.3.2 Current Expressions of Representations

In this work, we wish to extract the behavioral strategy directly from the causal behavioral model of the biological system. To aid in this process, we wish to choose a suitable expression for the causal behavioral model for strategy extraction. The manner in which a representation is visualized, or expressed, is extremely important to how designers can access and process the information contained within it. [105]. There are three general types of representation expressions, including: sentential (textual), mathematical, and diagrammatic. Sentential, or textual, expressions are written in text format and may be structured either in natural language format or in list format [105]. Mathematical expressions use equations and rules that describe knowledge about a system. Diagrammatic expressions use iconic or pictorial representations of knowledge, and preserve explicitly topological and geometric information [106]. When examining biological systems, many of the quantitative relationships that are the foundation of mathematical expressions are not known. Therefore, exclusive mathematical expressions are excluded from further study. For clarity, consider the following Piston-cylinder assembly example. The behavior of a piston-cylinder being heated is expressed sententially in Figure 3.7.

(i) A piston-cylinder assembly contains an ideal gas.
(ii) The piston-cylinder assembly is heated by an external flame
(iii) Heat is transferred to the gas, causing expansion.
(iv) The expansion of the gas causes the piston to do work on the environment

Figure 3.7 Sentential representation of the piston-cylinder example

The piston-cylinder assembly is expressed diagrammatically in Figure 3.8.

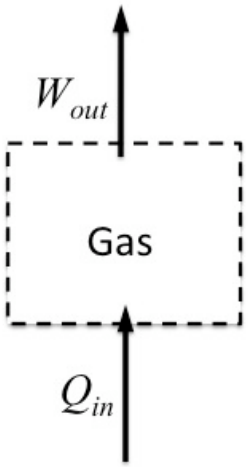


Figure 3.8 Diagrammatic representation of the piston-cylinder example

For expression of the causal behavioral description, three different representations are explored, including the textual expression as used by Chakrabarti [24], a static diagrammatic expression as seen in bond graphs, and a Petri net model as used by [60]. These representations are reviewed in Section 2.2.3.2.2.

3.3.3 Evaluation of Current Expressions

In this section, the three expressions (textual, static diagrammatic, and dynamic diagrammatic) are compared to the requirements put forth earlier in this section.

1. *Computational Offloading* - With respect to computational offloading, diagrammatic representations have the advantage in representing more information explicitly in the expression. This is largely due to the fact that textual descriptions typically are implicit and need to be mentally formulated, requiring greater computational effort. The general conclusion from the body of literature in cognitive offloading is the need

to maximize the load on the external representation so as to minimize the cognitive load needed to reason about the representation[102]. In other words, an increase in information presented explicitly in a given representation (computational offloading) leads to a reduction in the cognitive load on the reasoner of the representation. For example, consider the example of the transitivity relations between sets A, B, and C in the following textual (Figure 3.13) and diagrammatic (Figure 3.14) figures [107]. The transitivity relation is sententially expressed in Figure 3.9.

(i) All A are B
(ii) All B are C
(ii) (therefore) All A are C

Figure 3.9 Textual description of the transitivity relation

The diagrammatic representation of the transitive relation is displayed in Figure 3.10.

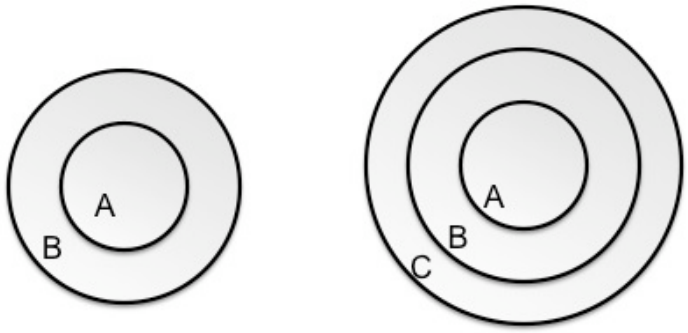


Figure 3.10 Transitivity in Euler's circle (modified from [107]

As can be seen in the Figure 3.9 and Figure 3.10, the conclusion of "*All A are C"* is more direct and straightforward in the diagram, as opposed to the sentential representation. The conclusion appears "for free", whereas in the case of the sentential representation, some logical inference is needed [107].

With respect to computational offloading, many of the same conclusions as above can be applied to the static versus animated diagrammatic representation case. Specifically, general statements of visual explicitness and directness of representation apply. Studies by Kaiser [108] and Jones, Scaife [109] conclude that dynamic representations include more explicit information about the state and dynamics of the

system, thus allowing more cognitive offloading by the reasoner. However, as Jones, Scaiffe [109] points out, this increased amount of information can also lead to user overconfidence and added complexity. They also point out that these negative effects can be countered by enabling user control of the dynamic representation.

2. *Inference* - With respect to the ability to infer new information from a representation, diagrammatic representations are preferred over textual representations. In diagrammatic representations, information and relationships are more direct, and offer clearer path to that of the represented system. Because of this, interpretation and inference from these representations is more obvious and more immediate [107] than that of textual representations. Diagrammatic representations also have the advantage of inference in that these representations can restrict (or enforce) kinds of interpretations or inferences that can be made about the represented system [110]. By restricting the possible interpretation of the representation, the representation can guide the reasoner in making the correct assumptions about the system and in the case of the CBD, guide them towards extracting the correct strategy.
3. *Validity* - Representation validity denotes the extent and rigor by which a representation is defined. Beyond common grammar rules, the textual representation offers the least rigor of definition. The advantage of textual representation is the freedom by which it can be expressed. Due to their basis in engineering and systems analysis, bond graph representations have been rigorously defined and extended over the years. The advantages of bond graphs are the direct expression of causality and the conservation of energy principle across the systems. The graphical representation can also be easily converted to differential equations and behavior analyzed through traditional means.

    Petri nets have the most extensive body of work in the literature, as well as many extensions and applications to varying fields. Although expressed graphically, Petri nets are also used as a mathematical tool for formal analysis of the behavior of systems. Petri nets also have a strong mathematical definition and rigorous rules for application.

4. *Consistent Reasoning across abstraction levels* - With respect to consistency in reasoning across abstraction levels, the textual description typically does not explicitly represent hierarchy. In the textual description, multiple levels of abstraction are typically described within the same view. Due to its flexibility, the natural language representation lacks a consistent reasoning structure. The diagrammatic representations discussed in this section both offer a consistent reasoning structure. The port-based approach employed by the bond graph representation offers very consistent reasoning in the transformation of power across the system. This same view holds for the system, as well as its subsystems. The state-based approach employed by the Petri net framework works equally as well, in that the reasoning structure is unchanged for the system.
5. *Isomorphism* - With respect to directness of representation, although the textual description lacks a structured representation, the freedom of expression allows a direct isomorphic mapping to the causal behavioral model being used. As seen in the textual description used for the SAPPhIRE representation [24], each construct of causal behavioral model can be directly described and mapped.

   The bond graph representation does not allow for a direct mapping from the causal behavioral model. Specifically, the physical phenomena and the states of the system are not explicitly represented. The Petri net representation allows for mapping of the physical phenomena (transitions) and states (places) of the system explicitly, but there is no direct representation of driving input and functional output in the representation.
6. *Model Complexity* – Model complexity refers to the ability of a representation to model complex relationships between subsystems and components, such as precedence of action, synchronization, mutual exclusion of resources, and concurrency [104]. With respect to model complexity, the textual description, due to its freedom of expression, can represent these complex relationships. It should be noted that although a textual description can represent complexity, it fails in representing complexity efficiently. On the other hand, Bond graphs, with the causal strokes, can represent precedence of action and synchronization. However, based on

its flow-based representation, bond graphs fail in the ability to represent concurrency and also lack to represent resources in the model. Due to the token-based approach of Petri nets, precedence, synchronization, resources, and concurrency can all be explicitly represented.

7. *Behavior Verification* – One of the key issues in modeling systems is the ability to check the behavioral model for correctness. Model verification refers to the ability of a representation to be verified that it is indeed doing what it is put forth to do. This usually involves some form of qualitative simulation.

   With respect to behavior verification, textual descriptions allow no means to formally check the properties of the behavior, aside from mental simulation. Bond graphs allow for behavior and power flows to be checked by causal analysis at the ports of the system. One of the strengths of the Petri net framework is that it allows analysis of many behavioral properties, including *reachability, boundedness*, and *liveness* [59]. A system is said to be *reachable* if there exists a set of transitions that can transform M_0 to M_n. Analysis of reachability of a system allows verification of reachable states of the system. A Petri net is said to be *bounded* "if the number of tokens in each place does not exceed a finite number k for any marking reachable from M_0"[59]. Analysis of boundedness prevents an overload of tokens at a state. A Petri net is considered *live* "if, no matter what marking has been reached from M_0, it is possible to ultimately fire any transition of the net by progressing through some further firing sequence"[59]. Analysis of liveness assures that there are no overflows in the system and that the system is deadlock-free.

The above analysis is summarized in Table 3.3 below. In the table, ✓ means fulfillment of a specific requirement, ✗ means that the requirement is not fulfilled, ✓✓ means a higher degree of fulfillment of a specific requirement, and ✓/✗ means in some cases the requirement is fulfilled and in others it is not fulfilled.

Table 3.3 Summary of Expression Analysis

	Textual	Static Diagrammatic (Bond Graphs)	Dynamic Diagrammatic (Petri nets)
Computational Offloading	✗	✓	✓✓
Inference	✗	✓	✓
Validity	✗	✓	✓✓
Consistency	✗	✓	✓
Isomorphism	✗	✓	✓✓
Model Complexity	✓	✓/✗	✓
Behavior Verification	✗	✓	✓✓

Based on the table above, Petri nets offer a very flexible framework for modeling the behavior of biological systems. Based on the analysis, the Petri net modeling framework should be used as a baseline for our representation. The Petri net framework is flexible in the sense that it can be used to model many different types of dynamic systems. In the next section, we extend the Petri net framework to be used with biological systems.

3.4 REPRESENTATION DEVELOPMENT

In this section, the Petri net formalism is extended for use in hierarchically representing biological systems. Peleg et al. [111] comment that one of the primary advantages of using Petri nets to represent biological systems is that the system behavior can be represented even when the biological mechanism is not fully understood. Additionally, they allow qualitative simulation, allowing a process to be described at variable levels of granularity [111]. Petri nets have been extensively used in modeling communications and workflow systems in manufacturing, software development, safety-critical control systems, etc. Petri nets have also been used to some extent in the biological domain. However, use of the Petri nets in this domain has been limited to modeling biological process, such as metabolic and biochemical pathways and networks [112-117, 111]. In Section 3.4.1, the hierarchical Petri net representation utilized in this research is presented, followed by a review of current approaches for creating these representations in Section 3.4.2.

3.4.1 From Causal Behavioral Description to Petri Nets

The foundations of the Petri net modeling framework are presented in Section 2.4. Peterson[118] comments that a valuable feature of Petri net modeling framework is the ability to model a system hierarchically. In doing so, entire nets can be replaced by single places and transitions at more abstract levels [118]. In this research, similar to that of Koga and Aoyama [60], a hierarchical system model, G^{system}, is defined using the Petri net framework for hierarchical behavioral representation. Specifically, a system is composed of components, $G^{structure}$, and the behavior of the systems, $G^{behavior}$. This system is defined at different levels, i, of abstraction, denoted simply as G_i, for i=1,2,…,n. This coupled structure-behavior system model allows us to decompose the behavioral hierarchy alongside that of the structural hierarchy. This allows us to view the system, and its associated structure and behavior, at different levels of refinement and abstraction.

The behavior of the system is described using the Petri net framework in the form of hierarchical Petri nets. A simple Petri net is defined as a 4-tuple, $PN = (P, T, F, M_0)$, whereby $P = \{p_1, p_2, \ldots, p_m\}$ is a finite set of places, $T = \{t_1, t_2, \ldots, t_m\}$ is a finite set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs, and $M_0 : P \rightarrow \{0,1,2,.....\}$ is the initial marking. A hierarchical Petri net is further defined as follows:

1. A system on the ith level (i=0,….,L) is defined as $G_i^{system} = (G_i^{structure}, G_i^{behavior})$
2. $G_i^{structure} = (S_i, I_i)$, where S_i are the components of the system on level i and I_i are the flows of material, signal, and energy between components on that level.
3. $G_i^{behavior} = (PN_i) = (P_i, T_i, F_i)$ where: $P_i = \{p_1, p_2, \ldots, p_m\}$ is a finite set of places on level i, $T = \{t_1, t_2, \ldots, t_m\}$ is a finite set of transitions on level i, and $F_i \subseteq (P_i \times T_i) \cup (T_i \times P_i)$ is a set of arcs or flow relations.
4. Arcs F_i can be further divided into internal arcs, F_{IN} , and external arcs, F_{EXT}. Internal arcs denote the internal behavior of the system and external arcs denote interactions with other components of the system at level i. External arcs define relationships, such as precedence and synchronous. In this model, interfaces I_i of $G^{structure}$ map to the external arcs of $G^{behavior}$.

5. A transition t_i in this model may be associated with another lower level system net $PN_{i+1}=(P_{i+1}, T_{i+1}, F_{i+1})$. This net is called a "subnet" of transition t_i. The transition t_i with an associated subnet is termed a macrotransition. The term macro is used to denote an associated micro-graph, or subnet.

For clarity, consider the following generic Petri net model, displayed in Figure 3.11.

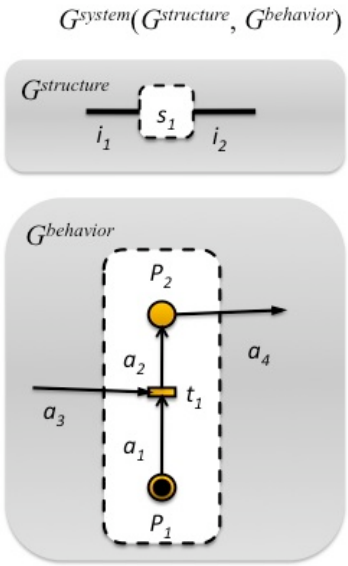


Figure 3.11 Petri Net System Model

In Figure 3.11, the $G^{structure}=(s_1)$ and $G^{behavior}=(PN)=(P, T, F)$ where: $P = (p_1, p_2)$, $T = (t_1)$, $F_{IN}=(a_1, a_2)$ and $F_{EXT}=(a_3, a_4)$. Based on this Petri net system model, a hierarchical Petri net representation is displayed in Figure 3.12.

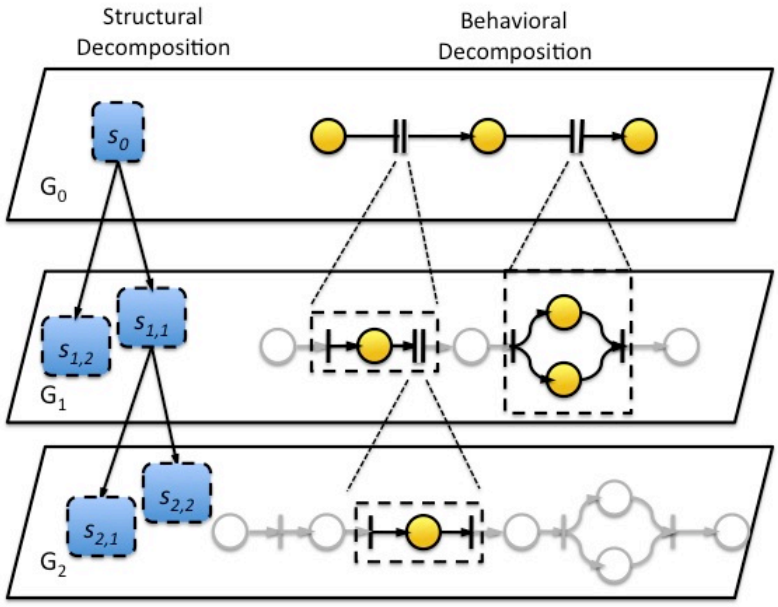


Figure 3.12 Hierarchical Petri Net model

As seen is Figure 3.12, the most abstract level of the system is denoted by G_0. This system is viewed as its associated structure, containing component s_0, and an associated behavior, PN_0. In this model, PN_0 is considered the combined behavior of the components on level 1. PN_0 has 3 states (represented as circles) associated with the system and two transitions. These transitions are termed macro-transitions since they each have a subnet associated with them. In the hierarchical representation, macro-transitions are modeled as a double-bar. On level G_1, the physical structure of the system is decomposed into components $s_{1,1}$ and $s_{1,2}$, as well as its behavior PN_1. PN_1 represents the combined behavior of components $s_{1,1}$ and $s_{1,2}$. As seen in the figure, the transitions on level G_0 are associated with subnets on level G_1. When viewed at this lower level of abstraction, we can view how the behavior of components $s_{1,1}$ and $s_{1,2}$ contribute to that of s_0. One can easily see how continued decomposition leads to a more complete view of the behavior of the system, and how components and subsystems contribute to the behavior of that system.

The hierarchical Petri net representation presented in Figure 3.12 has advantages in representing many types of biological system behaviors. The primary type of behavior that the representation is intended is that of biological systems whose structure is

decomposable and whose behavior can be discretized. To use this representation, the physical structure of the biological system must be decomposable, meaning that the target system must have subsystems associated with it. Also, behavior can be defined by the change of state of the system. To describe the behavior of the system using hierarchical Petri nets, the physical characteristics of interest of the system (ie. size, shape, color, orientation) must be discretizable into distinct states.

3.4.2 Creating Hierarchical Petri Nets

There have been several approaches introduced for generating hierarchical Petri nets [119-123], with all having their respective merit. The approach utilized in this work is that of Lee and Favrel [119]. Lee and Favrel [119] propose a step-by-step reduction method for creating hierarchical Petri nets. The approach introduces the concepts of macronets, which are composed of macrotransitions and macroplaces, the degree of a subnet, and the reducible subnet (RSN). In each step of the method, the lowest RSN is reduced into a macroplace or macrotransition. These macroplaces and macrotransitions correspond to the subnets. The sequence of reduction defined by the authors allows the study of complex systems in a step-by-step fashion. Lee and Favrel [119] define four different classes of RSNs: RSN-1, RSN-2, RSN-3P, and RSN-4T. These classes are displayed in Figure 3.13 and described below.

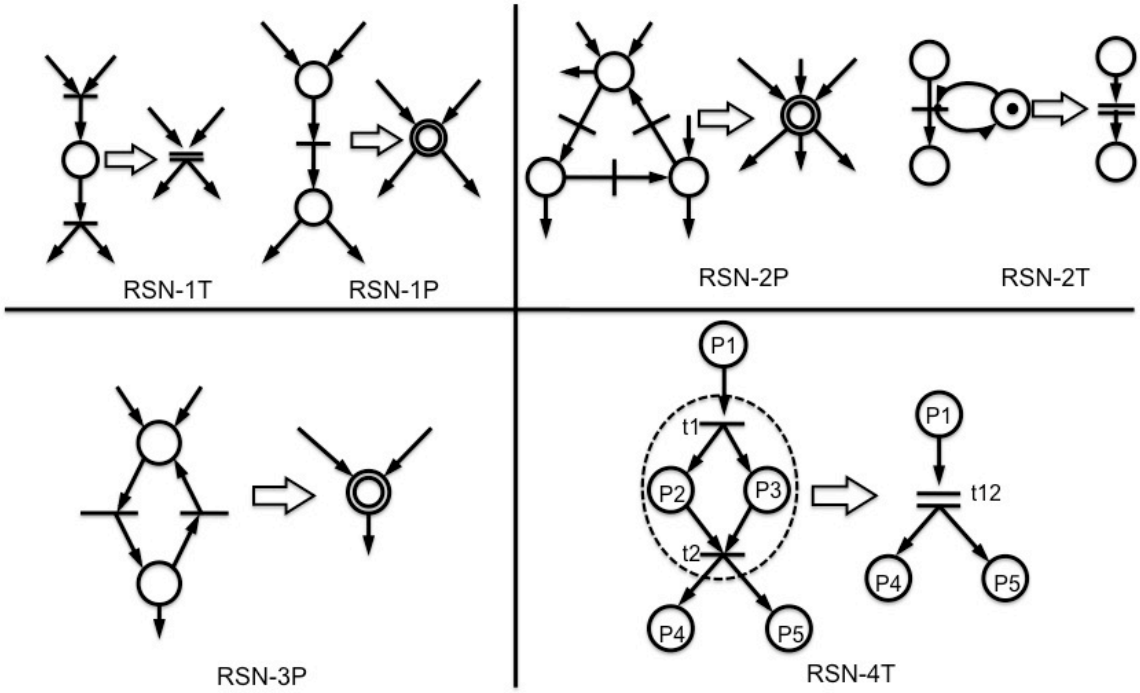


Figure 3.13 Reducible Subnets (RSN), adapted from Lee and Favrel [119]

RSN-1 is divided into two subclasses, RSN-1T, which denotes a subnet of transition, and RSN-1P, which denotes a subnet of place. RSN-1T (or P) forms (1) a directed path having only one input and one output door and (2) where the output and input degree of the doors is 1. Input and output doors are the nodes connected to the incoming and outgoing arcs of the subnet, respectively, and the degree of a node refers to the number of incoming and outgoing arcs from a node. The RSN-1T (or P) can then be replaced by a macrotransition (place). RSN-2T (or P) constructs a directed circuit containing at least one token, and can be replaced by a macrotransition (or place). RSN-3P, described as a unidirected cycle containing only one input door and one output door, can be replaced by a macroplace. RSN-4T describes a subnet that (1) contains all the paths from its input door to output door and (2) where the places within these paths contain only places with a degree of 2.

Lee and Favrel [119] then define an algorithm for creating the hierarchical Petri net which includes two steps in each iteration (*i)*: (1) the determination step of the RSN (A_i) to be reduced and (2) and the replacement of the RSN with a macronode (a_i). For synchronization of macronodes and subnets, the author uses the concepts of keys and

doors. Keys refer to the input and output nodes of the subnet, whereas doors refer to the input and output nodes of the macrotransition. This approach allows the behavior of the subnet to be synchronized to that of the macrotransition or macroplace of interest.

In this approach, the RSNs preserve many of the fundamental properties of the original net. The properties of most interest in this research are that of liveness, boundedness, and reachibility. These properties give us a means for checking the 'correctness' of our models. A biological system at any moment in time can only have one state, meaning that only one token is allowed in the net. By checking boundedness, specifically making sure the net is 1-bounded, we can assure the net only has one token. Biological systems are also free of behavioral deadlocks, and thus, the model must also be free of deadlocks, or live. Reachability allows us to assure that the model represents the behavior of the biological system correctly. In modeling a biological system, we wish to represent how the system gets from one state to another. Reachability assures that the model can indeed reach that state.

When building a hierarchical model based on the Petri nets, we must be able to assure that the properties of liveness, boundedness, and reachability are preserved among hierarchical levels. Lee and Favrel [119] put forth the following theorems and proofs regarding the preservation of liveness and boundedness, displayed below.

Theorem 1: A Petri net is live iff its subnets and macronet are live.

Proof: The liveness of a Petri net is defined by the firing sequence of transitions. If a reduction does not change any firing sequence, the liveness is not changed. A reduction of RSN by a macrotransition means a replacement of the subfiring sequence by a macrotransition. Because RSN-1T, RSN-2T, and RSN-4T construct subfiring sequences and can be fired by these subsequences, the replacement of these RSN's does not change the original firing sequence and hence does not change liveness. A reduction of RSN by a macroplace means a deletion of a subfiring sequence in the original sequence. Because all RSNs can fire by subfiring sequences and the input and output keys of a macronode are the same as those of the corresponding RSN, then RSN-1P, RSN-2P, and RSN-3P do not change the liveness.

Theorem 2: A Petri net is bounded iff its subnets and macronet are bounded.

Proof: The boundedness can be studied with a difference between the number of input keys and output keys of an RSN. Because the input keys and output keys of a macronode are the same as those of the corresponding RSN, the reduction does not change the boundedness of the Petri net.

3.5 CLOSURE AND VALIDATION

At the beginning of this chapter, the following research question was posed:

(RQ1) "What type of representation can be used to model the behavior of biological systems?"

To answer this question, it was hypothesized in *Hypothesis 1* that a representation based on (1) a causal behavioral description and (2) hierarchical Petri nets can be used to model the behavior of biological systems. In validating this hypothesis, a qualitative evaluation of the proposed representation versus several representation requirements was performed. In Section 3.1.2, the following requirements for representation of biological systems were presented: hierarchical representation, explicit dynamic representation, explicit representation of the environment, behavior-centric approach, and completeness and uniqueness of representation. In Section 3.2.4, the proposed hierarchical causal behavioral description of biological system behavior was found to meet these requirements. With respect to expression of the causal behavioral description, the following requirements were developed in Section 3.3.1: computational offloading, inference, validity, consistency, isomorphism, model complexity, and behavior verification. After evaluation, the Petri net representation was found to meet these requirements. After qualitative evaluation, a representation based on a causal behavioral description and hierarchical Petri nets was found to meet the requirements for modeling the behavior of biological systems.

The overall goal of this research is to leverage biological strategies in the conceptual design of engineering systems. The hierarchical Petri net representation not only affords representation of the behavior of these systems at multiple levels of abstraction, but also allows functional and structural information to be represented. The multi-layered view of behavior proposed in this chapter gives a much richer description

of behavior needed to aid in the systematic extraction of these strategies from biological systems.

Validation Strategy – Theoretical Structural Validity

With respect to our validation strategy for the Method for Reverse Engineering Biological Systems, the key theoretical construct of the proposed method, the hierarchical Petri net representation, was reviewed. The validation found in this chapter is presented in context of the validation strategy for this dissertation in Figure 3.14. Specifically, Theoretical Structural Validity was addressed.

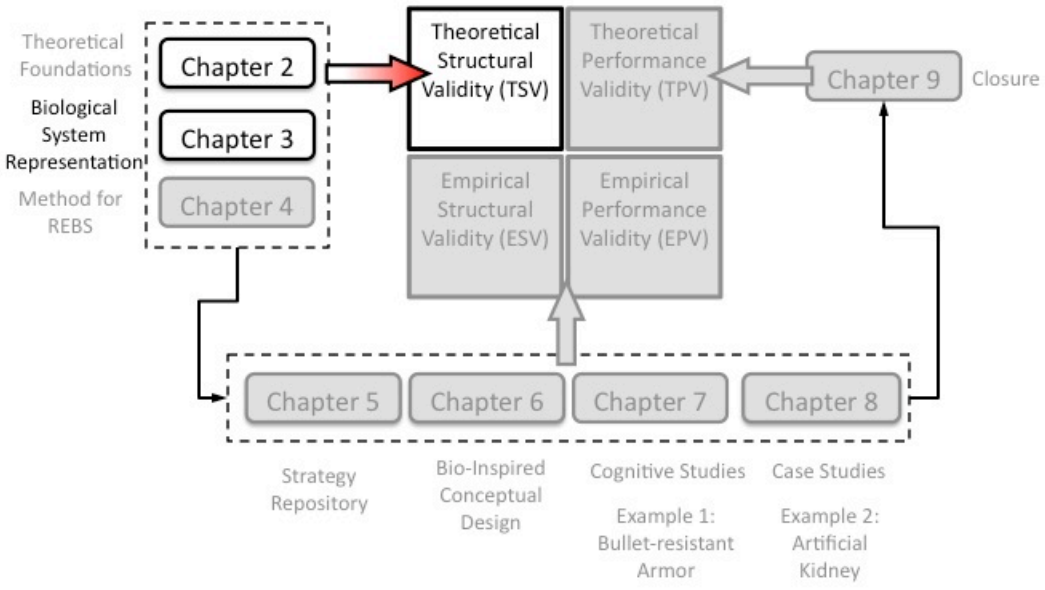


Figure 3.14 Validation in Chapter 3

Theoretical Structural Validity involves accepting the validity of the individual constructs that constitute the method. In this chapter, a rigorous assessment of both engineering representations and representation expression against several key requirements for representing biological systems was presented. In our evaluation, Petri nets presented the best foundation to represent the behavior of biological systems. Building on this foundation, the hierarchical Petri nets were developed to represent the behavior of biological systems. Through this assessment, along with the assessment performed in Chapter 2, the theoretical structure of the proposed method has been validated.

In Chapter 3, the backbone of the proposed method for Reverse Engineering Biological Systems, the hierarchical Petri net representation, was presented and evaluated. In Chapter 4, building on this representation, the proposed method for Reverse Engineering Biological Systems is developed.

CHAPTER 4 METHOD FOR REVERSE ENGINEERING

BIOLOGICAL SOLUTIONS

In Chapter 3, the backbone of the proposed method, hierarchical Petri net representation, was put forth. In this chapter, the proposed method for Reverse Engineering Biological Systems is presented. The dissertation outline is displayed in Figure 4.1.

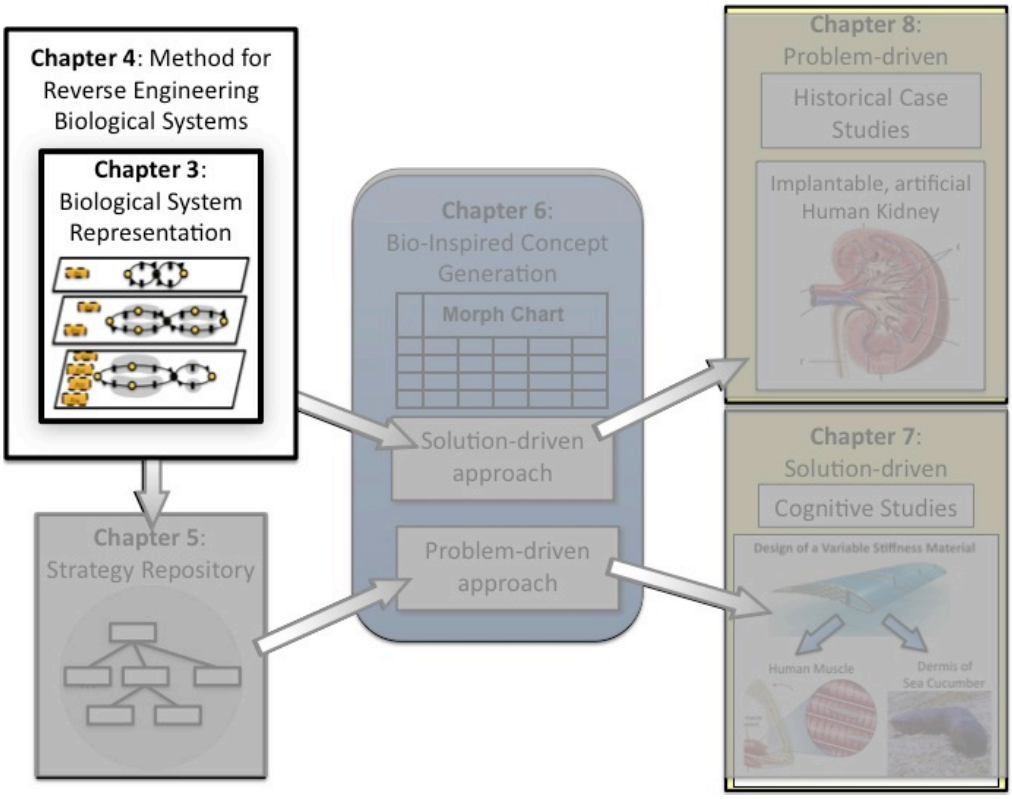


Figure 4.1 Chapter 4 and the Dissertation Overview

The overall goal for this dissertation is to put forth an approach for aiding the designer in generating ideas in Conceptual Design through the use of biological strategies. It is believed that leveraging biological strategies in Conceptual Design will lead to a more effective ideation process. In this work, a method for extracting behavioral

strategies from biological systems, the method for Reverse Engineering Biological Systems, is developed. As the name suggests, the proposed method aids the designer in reverse engineering biological systems and extracting behavioral strategies. Reverse engineering can be defined as the "process of developing a set of specifications for a complex hardware system by an orderly examination of specimens of that system"[124]. This process is conducted "without the benefit of any of the original drawings" [124]. The main purpose of reverse engineering is to (1) identify the system's components and relationships between those components and (2) represent the system in another form or higher level of abstraction [31].

The backbone of the proposed method is the hierarchical Petri net representation of biological systems. Using hierarchy, this representation allows the designer to visualize the behavior of lower subsystems (strategy) and how it affects the overall behavior of the system. To extract the correct strategy, behavior must be consistent across these levels of hierarchy. To ensure consistency, three fundamental properties of Petri nets, reachability, liveness, and boundedness, are considered.

Specifically, in this chapter, the following question is addressed:

(RQ2) *How can the behavior of biological systems be hierarchically represented using Petri nets, while preserving the fundamental properties at each hierarchal level?*

In answering this question, Hypothesis 2 from Chapter 1 is as follows:

Hypothesis 2: Using the systematic method for Reverse Engineering Biological Systems will ensure that the fundamental properties of boundedness, reachability, and liveness will be preserved across hierarchical levels.

To validate this hypothesis, the following approach is followed:

1) Develop the general phases of the proposed method (Section 4.1)
2) Present specific steps for the method for Reverse Engineering Biological Systems. (Section 4.2)
3) Evaluate the preservation of fundamental properties (Section 4.3)

4) Put forth several example problems in which the proposed method is used (Sections 4.4)

4.1 METHOD DEVELOPMENT

In Chapter 3, the hierarchical Petri net representation for biological systems was presented. In this chapter, the two key phases of hierarchical Petri net generation, System Decomposition and Behavioral Mapping, are defined. These phases, displayed in Figure 4.2, are discussed in further detail in Section 4.1.1 and 4.1.2 below. In Section 4.1.3, a method for extracting behavioral strategies from these representations is presented.

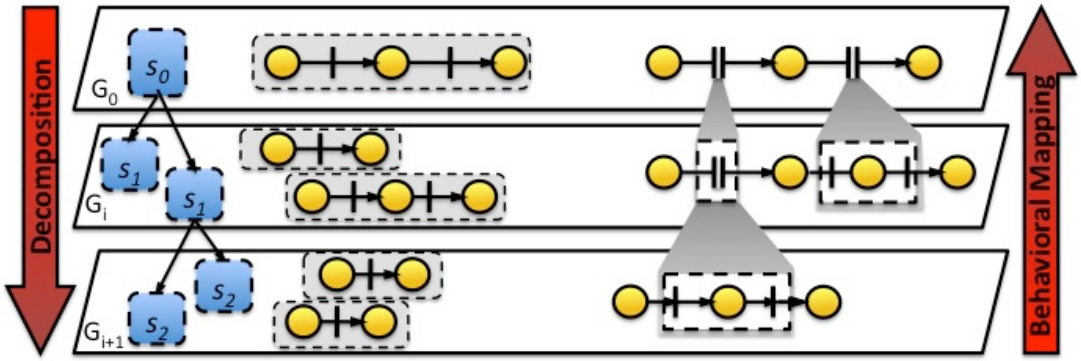


Figure 4.2 System Decomposition and Behavioral Mapping Phases

4.1.1 System Decomposition

In the System Decomposition phase of hierarchical net generation, the system, G^{system}, is decomposed into its two parts, $G^{structure}$ and $G^{behavior}$. Following a top-down approach, the physical structure and individual behaviors are decomposed in a coupled fashion. This coupling helps assure only relevant behaviors and systems are represented. It should be noted that the accessibility of these systems will vary, thus, reliance on biological literature on the subject system will also vary. The two key steps of System Decomposition, Structural Decomposition and individual Behavior Generation, are discussed in further detail in Sections 4.1.1.1 and 4.1.1.2.

4.1.1.1 Structural Decomposition

In the Structural Decomposition step, the physical structure, $G^{structure}$, of the system is decomposed. The purpose of the decomposition phase is to decompose the system into its subsystems and components, and create hierarchical relationships between these systems. Due to the inherent complexity of biological systems, these hierarchical relationships are not easily defined. However, biological systems perform specific functions, and through natural selection, these systems are produced and localized within parts to some extent [125]. The reason is that to achieve the coordination of internal activity that function demands, these systems must be integrated internally and isolated externally to limit interference from other functions [94]. Using the assumption of tight integration and isolation, the structural decomposition of these biological systems becomes more manageable.

To aid in defining the hierarchical relationships between biological systems and subsystems, we suggest using a structured decomposition framework such as the decomposition protocols put forth by McShea [99]. According to McShea's [99] protocols for classifying biological parts, "partness" is evaluated based on two criteria: (1) enclosure (physically isolated) and (2) contiguity with a difference in composition. According to McShea, the appearance of an object is usually a consequence of a relatively tight integration and the object's boundary corresponds to isolation from its surroundings. McShea developed these protocols, displayed in Figure 4.3, for somewhat structurally simple organisms, but also includes the notion that they may be extendable to the more complex biological systems that are used in this work.

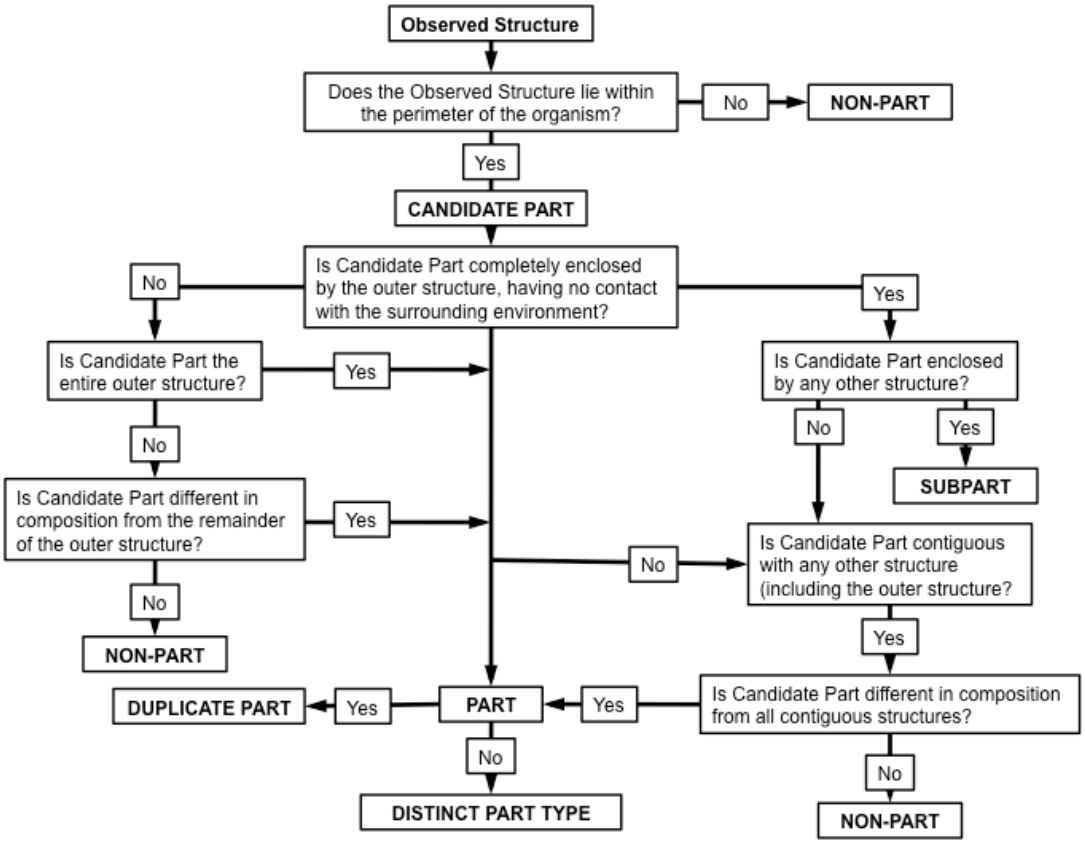


Figure 4.3 McShea's protocols for identifying hierarchy of parts [99]

McShea's protocols, displayed in Figure 4.3, are a means for generating hierarchical relationships in the structural decomposition process. Specifically, the criteria for enclosure and contiguity with a difference in composition are especially useful when identifying the level at which systems and subsystems fit into the hierarchy. It must also be noted that in many cases, it is infeasible to use systematic protocols such as those presented by McShea. In these cases, we rely on biological literature to guide the decomposition process.

4.1.1.2 Behavior Generation

In the second step of decomposition, the individual behaviors, $G^{behavior}$, of the system are generated using the Petri net modeling framework. Specifically, system attributes by which to define behavior are first identified. Discrete states of these attributes are then identified and labeled as places. Next, the physical phenomena driving the change of state of the system are labeled as transitions. The behavior of the system is

then characterized by a firing sequence transforming from one state of the system, M_0, to another state, M_n. A firing sequence is denoted by $\sigma(M_0,M_n) = M_0,t_1,M_1,t_2,\cdots t_n,M_n$ or simply as $\sigma(M_0,M_n) = t_1,t_2,\cdots t_n$.

Next, the behaviors of the individual subsystems are joined using precedence and synchronous relations [60]. Synchronous arcs are events (physical phenomena) that occur concurrently and Precedence arcs set order to events. For clarity, consider the arbitrary system in Figure 4.4. In the figure, system s_0 is decomposed into subsystems $s_{1,1}$ and $s_{1,2}$. The individual behaviors of two components, $s_{1,1}$ and $s_{1,2}$, are modeled. Precedence relations (denoted by a dashed arrow) are also defined between the two components from place B_2 and transition t_6 and from place B_3 and t_5. The precedence relations mean that places B_2 and B_3 must have tokens before transitions t_6 and t_5 can fire, respectively.

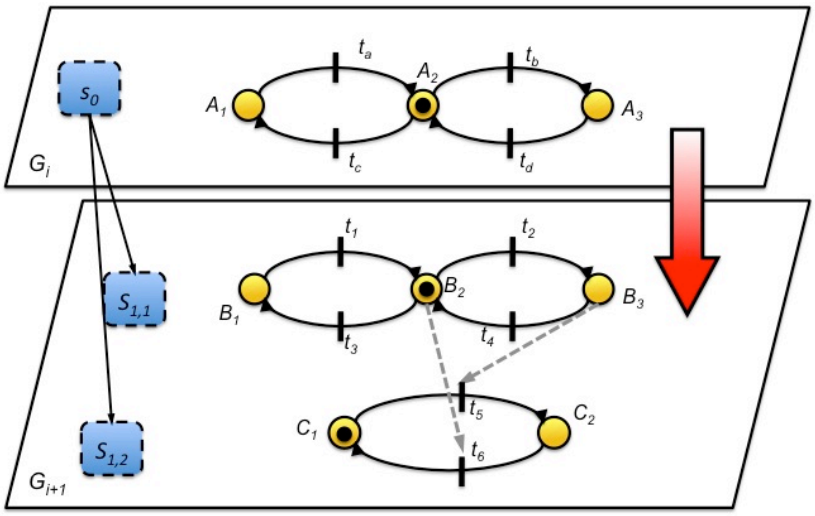


Figure 4.4 Decomposition

In the decomposition step, a mapping is created between the states of the higher level system, s_0, and the states of the lower level subsystems, $s_{1,1}$ and $s_{1,2}$. For example, consider the arbitrary system in Figure 4.4. When the system s_0 is in state A_1, subsystem $s_{1,1}$ is in state B_1 and subsystem $s_{1,2}$ is in state C_1, thus a mapping can be created between state A_1 of s_0 and state B_1 of $s_{1,1}$ and C_1 of $s_{1,2}$. This mapping is denoted as $(B_1, C_1) \subseteq A_1$. Similarly, when the system s_0 is in state A_2, subsystem $s_{1,1}$ is in state B_2 and subsystem $s_{1,2}$ is in state C_1, thus $(B_2, C_1) \subseteq A_2$. Also, when the system s_0 is in state A_3, subsystem $s_{1,1}$ is

in state B_3 and subsystem $s_{1,2}$ is in state C_2. It follows that $(B_3, C_2) \subseteq A_3$. These mappings will be used later to map the subnets of the lower level subsystems to the higher level system.

4.1.2 Behavioral Mapping

The next phase of the hierarchical net generation is Behavioral mapping. In this phase, following a bottom-up approach, the behaviors of the lower level systems are mapped to that of the higher level system. This phase includes two steps, Combined Behavior Generation and Inheritance. These steps are discussed in detail in Sections 4.1.2.1 and 4.1.2.2.

4.1.2.1 Combined Behavior Generation

The first step of Combined Behavior Generation is behavioral mapping, the behaviors of the individual subsystems are combined to form a joint behavioral net. In phase 1, the individual behaviors of each system in the hierarchy were generated. In this step, a combined behavioral graph is generated using the precedence and synchronous relationships between the subsystems determined in Section 4.1.1.2.

The behavior of the individual subsystems is combined using a reachability graph, which is a graph of all possible markings in a given Petri net. In a reachability graph, the nodes represent markings of the system and the arcs represent the transition firing sequences, transforming one marking of the system to another (murata, 89). For clarity, consider the Petri net displayed in Figure 4.5.

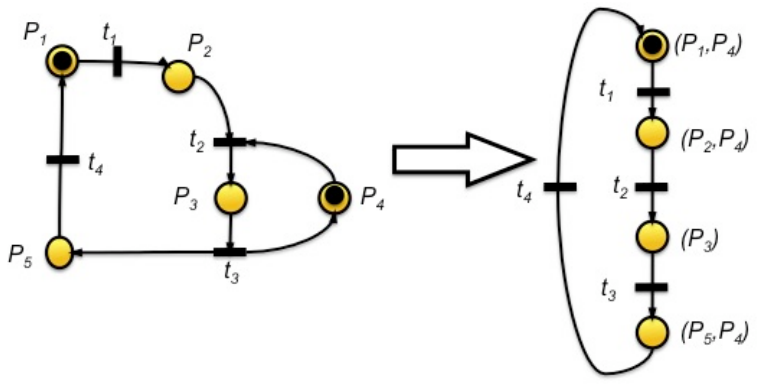


Figure 4.5 Reachability graph example: (a) Petri net model and (b) Reachability graph

In Figure 4.5, an example of a Petri net containing places P_1- P_5 and transitions t_1-t_4 is displayed. In the initial marking of the net, tokens are located at places P_1 and P_4. A reachability graph is then generated for the net containing all the possible markings. The reachability graph begins by defining a node for the initial marking (P_1, P_4). When transition t_1 is fired in the net, the marking becomes (P_2, P_4), meaning that a token has moved from P_1 to P_2. In the reachablity graph, this new state is denoted by place (P_2, P_4). When t_2 is fired, transition P_3 receives a token from both P_2 and P_4, denoted by place (P_3) in the reachability graph. Following this procedure, the reachability graph is generated for all possible states of the net. In the graph, a token is used to denote the current state, or marking, of the net. This also transforms the reachablity graph into a Petri net itself, able to represent the movement of both tokens in the original net in a concise fashion. For the initial marking of the system, a token is placed in place (P_1, P_4) of the reachability graph.

In this work, similar to that of Koga and Aoyama [60], the reachability graph is used to combine the behaviors of individual components of a system. For clarity, consider the arbitrary system from Figure 4.4. The procedure for generating the combined behavior graph is displayed in Figure 4.6.

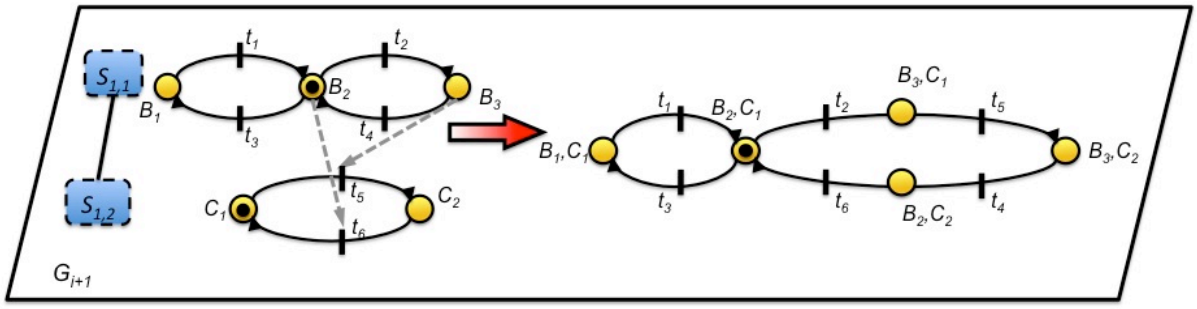


Figure 4.6 Combined Behavior Generation

In Figure 4.6, a reachability graph is generated for the combined behavior of subsystems $s_{1,1}$ and $s_{1,2}$. In the decomposition phase, precedence and synchronous relations between the individual subsystem behaviors were identified. Using these relations, the reachability graph is generated for both subsystems $s_{1,1}$ and $s_{1,2}$. First the

initial marking of the subsystems is identified as state B_2 of $s_{1,1}$ and C_1 of $s_{1,2}$. We denote this as place (B_2,C_1) in the reachability graph. From this state, transition t_2 and t_3 can be fired. We begin by firing t_2, which moves tokens to state B_3 of $s_{1,1}$. This is denoted by place (B_3,C_1) in the reachability graph. Next, based on the precedence relation, t_5 can now be fired and moves a token from place C_1 to C_2 of $s_{1,2}$. This is denoted by place (B_3,C_2). This procedure is followed until all reachable markings of the systems are represented. Using the reachability graph, the behaviors of subsystems $s_{1,1}$ and $s_{1,2}$ are combined into one Petri net model with the behavior of both systems represented by the path of a single token.

4.1.2.2 Inheritance

The final step of the Behavioral Mapping phase is Inheritance, where the hierarchical Petri net model of the biological system is generated. Similar to Lee and Favrel [119], model generation involves two key steps: (1) subnet identification and (2) subnet replacement. However, several refinements are made to the method proposed to Lee and Favrel. In the case of the reachability graph (combined behavioral model), this research only deals with reducible subnets (RSN) of the RSN-1T variety. Also, a much more targeted identification of subnets is performed, as this research attempts to directly link the behavior of the lower level components to that of the system. With respect to step 2 (replacement), instead of replacing the RSN with a macronode on the same level of hierarchy G_{i+1}, this subnet is inherited by a macrotransition on level G_i. This process is displayed in Figure 4.7.

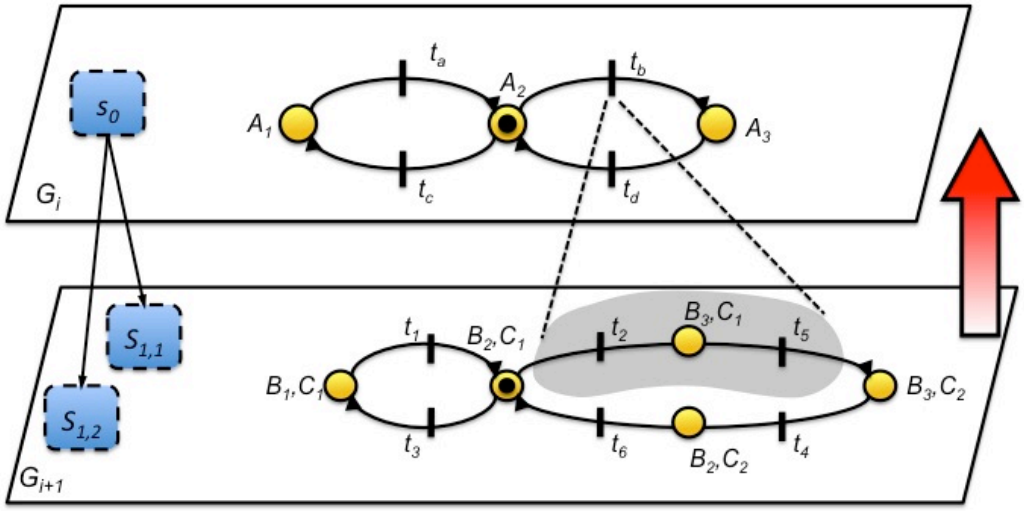


Figure 4.7 Subnet Inheritance

Figure 4.7 displays two levels of decomposition of an arbitrary system, G_i and G_{i+1}. In step 1 of Inheritance, the subnets are identified. The following is defined to aid in identifying subnets.

Let:

$\bullet t = \{p | (p,t) \in F\}$ is the set of input places (p) of transition (t) through an arc in F, and

$t\bullet = \{p | (t,p) \in F\}$ is the set of output places (p) of transition (t) through an arc in F,

where $\bullet t = t\bullet = \varnothing$ and F is the set of arcs.

This notation can also be extended to a net of transition S^t, meaning that its input and output doors are transitions. In this case, let:

$S^t\bullet = \{p | (p,t) \in F\}$ is the set of input places (p) of transition (t) through an arc in F

$\bullet S^t = \{p | (p,t) \in F\}$ is the set of output places (p) of transition (t) through an arc in F

where $\bullet S^t = S^t\bullet = \varnothing$ and F is the set of arcs.

Subnet definition

$S^t \subseteq t$ (ie. net of transition S^t is a subnet of transition t) *if and only if* $\bullet S^t \subseteq \bullet t$ and $S^t\bullet \subseteq t\bullet$

For example, consider the arbitrary system in Figure 4.7. Let $S^t = (t_2,(B_3,C_1),t_5)$ *and* $t = t_b$. If $B_2,C_1 \subseteq A_2$ and $B_3,C_2 \subseteq A_3$, then $(t_2,(B_3,C_1),t_5) \subseteq t_b$. In other words, net $(t_2,(B_3,C_1),t_5)$ is a subnet of transition, t_b, meaning that subnet $(t_2,(B_3,C_1),t_5)$ refines the behavior of t_b.

In step 2 of Subnet Inheritance, the subnet, S^t, is inherited to transition t, thus making t a macro-transition for the subnet.

4.1.3 Strategy extraction

In this work, biological strategies are viewed as refinements of behavior, where specific physical phenomena driving a particular behavior (and function) are identified. The hierarchical Petri net representation is key as it gives us a means to view the multiple levels of system behavior needed to extract a strategy, as well as a causal path of behavior across levels of abstraction of the system. To systematically extract strategy from the biological systems, the hierarchical Petri net model generated in Section 4.1 is utilized.

In Section 4.1, the method for identifying subnets was defined. These subnets are considered refinements of macro-transitions t. In the Petri net modeling framework, behavior is defined as a path, σ, where $\sigma(M_0,M_n) = t_1,t_2,\cdots\cdots t_n$, transforming an initial state of a system, M_0, to another state, M_n. Based on the definition of strategy put forth in this work, subnets are considered the strategies by which the behavior t is performed. Therefore, the strategy (S') of behavior (t) is denoted by the behavioral path ($\sigma(\bullet S^t, S^t \bullet)$) of its subnet S^t, or $S'(t) = \sigma(\bullet S^t, S^t \bullet)$.

4.2 METHOD FOR REVERSE ENGINEERING BIOLOGICAL SYSTEMS

In Section 4.1, the three phases of the method for Reverse Engineering Biological Systems were presented. In this section, systematic steps for the proposed method are prescribed. A flowchart for the proposed method is displayed in Figure 4.8.

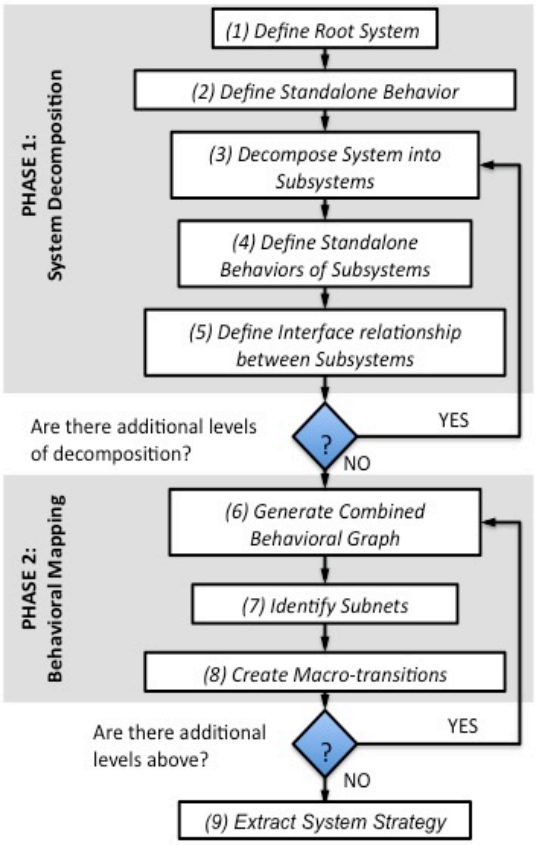


Figure 4.8 Flowchart for Method for Reverse Engineering Biological Systems

The individual steps for generating the hierarchical Petri net representation are detailed as follows:

System Decomposition

1) *Define root system:*

In this step, the designer defines the root biological system of interest, G^{system}. This system is defined by a boundary around the system. Each component, $G^{structure}$, is then defined by its attributes of interest, or properties, by which we describe the behavior, $G^{behavior}$, of the system. Once the system is defined, the working environment is also defined, including the system's interactions with the environment. These interactions are either (1) flows of material, signal, energy or (2) physical interactions between the entity and its environment. These interactions also have attributes. For instance, an attribute of energy flow may be mechanical energy at one level of abstraction, and force at another.

2) *Define standalone behavior:*

In step 2, the designer defines the behavior of the root system using the Petri Net modeling formalism. In this step, the behavior is defined by a change of state of a given system. These states are defined by different levels of the attributes defined in step 1. If the behavior is considered continuous, then it is discretized into discrete states. Using the PN formalism, these states are defined as places and the physical phenomena driving the state change are defined by transitions.

3) *Decompose system and sub-systems*

Once behavior of the root entity is defined, if the entity has subsystems, these subsystems are identified using McShea's protocols for partness (because bio systems are highly integrated and modular). It should be noted that only the entities that directly affect the behavior of the root entity are modeled. Attributes for these subcomponents are also identified. Interactions between these subcomponents are also defined.

4) *Define standalone behaviors of sub-systems*

Following the procedure from Step 2, the standalone behavior of the subcomponents are identified. In this step, the states of the root system are mapped to the states of the lower entities. The subsystem interactions are inherited from the subsystems into the behavioral model as external arcs. These external arcs are used to describe the interaction between the subsystems in the behavioral model. In this step, the states of the root system to that of the subsystems are also mapped.

5) *Define interface relationships between subsystems*

External arcs are used to define the interface relationships between subsystems. These external arcs are defined as either (1) synchronous or (2) precedence [60]. Synchronous relationships are denoted using a dashed, double bar, and precedence relationships using a dashed arrow.

Behavioral Mapping

6) *Generate combined behavioral model*

Using the external arcs defined between the subsystems in Step 4, a reachability graph is generated beginning with the initial marking (M_0) of the system. With this, a causal behavioral model for the combined subsystems is generated.

7) *Identify subnets*

Using the subnet definition from Section 4.1.2.2 and the state mappings from step 4, subnets from the combined behavioral model are identified and mapped to transitions in the upper behavioral model.

8) *Create Macro-transitions*

In this step, the subnets identified in Step 7 are inherited to their corresponding macro-transitions. A hierarchical model of behavior consisting of macro-transitions and subnets describing the behavior at increasing levels of detail has now been created.

Strategy Extraction

9) *Extract System Strategy*

Using the hierarchical Petri net representation, strategy is systematically extracted from the subnet representing the behavior of the lower level systems. Behavioral strategies (*S'*) are defined with respect to macro-transitions, such that *S'(t)* = $\sigma(\bullet S^t, S^t \bullet)$. Strategy can also be denoted between two states as *S'(•t,t•)* = $\sigma(\bullet S^t, S^t \bullet)$.

4.3 LIVENESS, BOUNDEDNESS, AND REACHABILITY

In Section 4.2, the systematic steps for the Method for Reverse Engineering Biological Systems were presented. Consistency in behavior across hierarchical levels is key to the extraction of correct strategies using the proposed method. To examine consistency across hierarchical levels, three fundamental properties are considered: boundedness, liveness, and reachability. Specifically, in this research, it was hypothesized that:

Hypothesis 2: Using the systematic method for Reverse Engineering Biological Systems will insure that the fundamental properties of boundedness, reachability, and liveness will be preserved across hierarchical levels.

In the following section, mathematical evidence is presented to validate this hypothesis.

Boundedness

A system is considered bounded if the number of tokens in each place does not exceed 1, or $M(p) \leq 1$. In the Combined Behavior Generation phase, the reachability graph is used to combine the behavior of the individual subsystems. This combined behavior graph is then inherited by its upper level behavioral graph in the Inheritance phase.

The reachability graph is defined in this work such that $|\bullet t| = |t \bullet| = 1$ (each transition has exactly one input and output place) and $\sum M(p) = 1$ (a system can only have 1 live marking at a given instant). Therefore, $M(p) \leq 1$, or the system is fully bounded.

By subnet definition $S^t \subseteq t$, $M(\bullet S^t)=M(\bullet t)$ and $M(S^t \bullet)=M(t \bullet)$ (ie. the marking of the input and output place of subnet is equal to that of its macro-transition). Therefore, boundedness is preserved across hierarchical levels.

Liveness

The liveness is defined by the firing order of transitions. By definition of a reachability graph, a transition *t* appears only if *t* is live for marking *M* in $R(M_0)$, where *R* is the reachability. Therefore, S^t is live.

By defining $S^t \subseteq t$, then $\sigma(\bullet S^t, S^t \bullet) \subseteq \sigma(\bullet t, t \bullet)$, meaning that the firing order is not changed. Since subnet inheritance does not change the firing order, the liveness is preserved.

Reachability

A marking M_n is reachable from M_0 if there exists a sequence of transition firings transforming M_0 to M_n. By definition, a reachability graph, $R(M_0)$, generates all reachable states, M, from state M_0. Thus, all the states in S^t are reachable.

Since $S^t \subseteq t$, $M(\bullet S^t)=M(\bullet t)$ and $M(S^t \bullet)=M(t \bullet)$. Therefore, reachability is preserved.

4.4 EXAMPLES

In Section 4.2, the systematic steps for the method for Reverse Engineering Biological Systems were presented. In this section, illustrative examples of the proposed method are presented. In Section 4.4.1, strategy is extracted from the Mutable Connective Tissue of the Echinoderm. In Section 4.4.2, strategy is extracted from Human Muscle in Isometric Contraction.

4.4.1 Strategy Extraction from the Mutable Connective Tissue of Echinoderms

1) Define root system:

In this first step, the root system is identified as the dermis of the sea cucumber. The boundary of the system is drawn just around the mutable connective tissue of the dermis. The interactions with the environment, displayed in Figure 4.9, include neural control signal (chemical) coming into the system and mechanical energy (reaction force) output to the environment as the dermis changes stiffness. The attribute of interest is defined as the stiffness of the dermis.

Figure 4.9 Root system (Dermis)

2) Define standalone behavior:

In step 2, the behavior of the root system, the dermis, is modeled using the Petri net modeling formalism (displayed in Figure 4.10). The states of the system are defined as Flexible (*Fl*), Natural (*Nat*), and Rigid (*Rgd*). The transitions are defined in Table 4.1

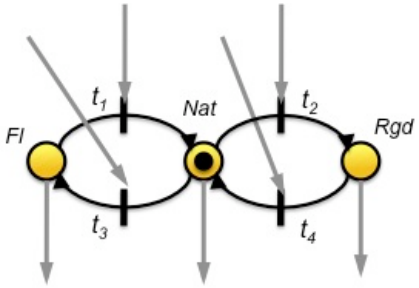


Figure 4.10 PN model of Dermis

Table 4.1 Terms for PN model of Dermis

t_1	Increase stiffness
t_2	Increase stiffness
t_3	Decrease stiffness
t_4	Decrease stiffness

3) *Decompose system and sub-systems*

In step 3, the system is decomposed into its subsystems. Through review of literature [126-133] on the dermis of the sea cucumber and use of McShea's protocols for partness[99] , the physical structure, displayed in Figure 4.11, was decomposed. The dermis consists of collagen fibril bundles, neurosecretory cells, and an extracellular matrix. The collagen fibril bundles are groupings of parallel collagen fibrils associated by a microfibrillar network. These fibrils are held in close association by a protein named 'Stiparin'. Under inputs from the nervous system, the neurosecretory cells release a protein called 'stiffener', which binds the collagen fibrils to one another. The neurosectory cells also release a protein called a 'stiparin-inhibitor', which inhibits the close association of fibrils caused by the 'stiparin' protein. The extracellular matrix contains many other soluble proteins, glycans, and proteoglycans dissolved in the liquid phase of the matrix.

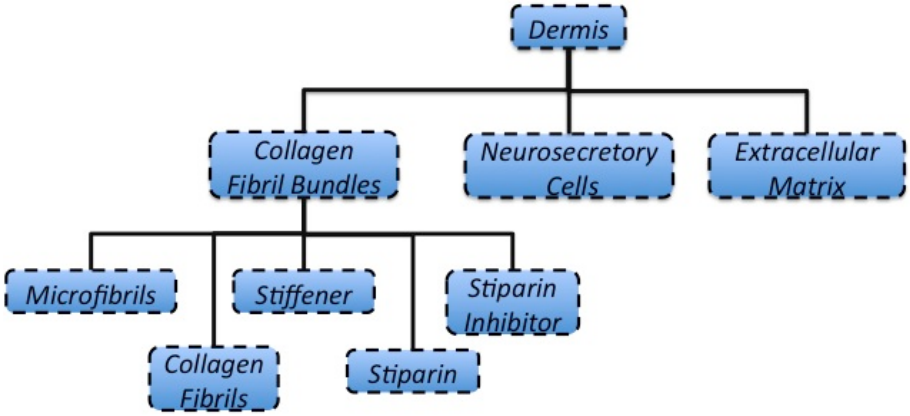


Figure 4.11 Structural Decomposition of Dermis

Next, the interactions between the components are modelled and displayed in Figure 4.12 and Figure 4.13. In reaction to a chemical control signal (S_{chem}), the neurosecretory cells release a stiffener or a stiparin inhibitor into the extracellular matrix. The chemical energy (E_{chem}) of these proteins cause either a binding of the collagen fibril bundles or association between the fibrils to be broken, respectively. Based on these chemical bonds between the fibrils, the system's stiffness is changed and a change in the reaction force (E_{mech}) output to the environment follows.

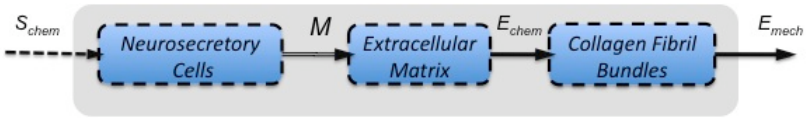


Figure 4.12 Interactions for first layer of Dermis decomposition

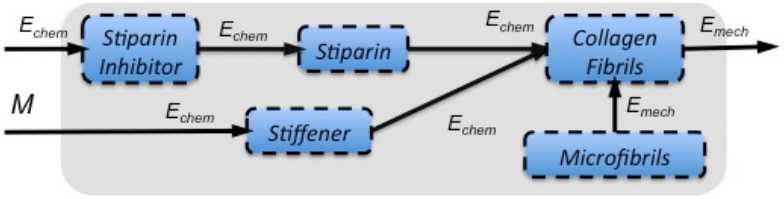


Figure 4.13 Interactions in Collagen Fibril Bundle decomposition

4) <u>Define standalone behaviors of sub-systems</u>

In step 4, the standalone behaviors of each of the subsystems are modeled using the Petri net model (displayed in Figure 4.14). The states and transitions of the behavioral models are displayed in Table 4.2.

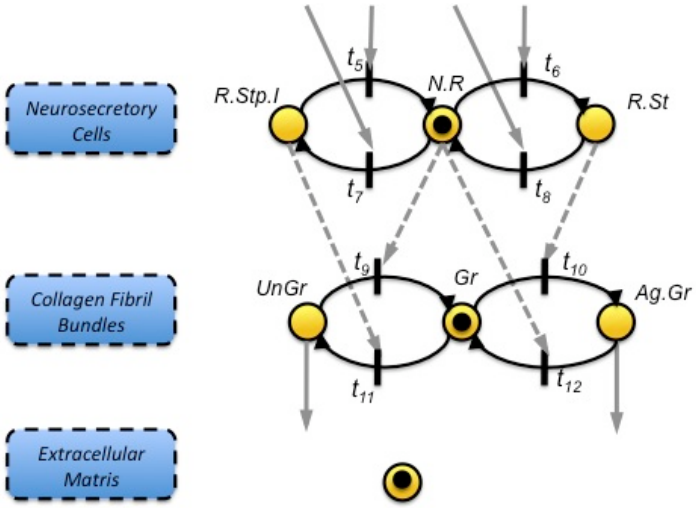


Figure 4.14 Standalone behaviors of the subsystems of the dermis

Table 4.2 Terms for Figure 4.14

R.Stp.I	Releasing Stiparin-Inhibitor	*UnGr*	Ungrouped
N. R.	No Release	*Gr*	Grouped
R.St	Releasing Stiffener	*Ag.Gr*	Aggregated and Grouped
t_5	Stop release of Stiparin-Inhibitor	t_9	Group fibrils
t_6	Release Stiffener	t_{10}	Aggregate fibrils
t_7	Release Stiparin-Inhibitor	t_{11}	Ungroup fibrils
t_8	Stop release of stiffener	t_{12}	Un-aggregated fibrils
Nat	Natural		

In this step, the states of the subsystems are mapped to that of the root system. In this case, the states of the collagen fibril bundles (CFBs), neurosecretory cells (NCs), and extracellular matrix (EM) are mapped to the states of the dermis. This mapping is displayed in Table 4.3.

Table 4.3 State Mappings for Sea Cucumber Dermis

Dermis (*Fl*)	CFBs (*R.Stp.I*), NC (*UnGr*), EM (*Nat*)
Dermis (*Nat*)	CFBs (*N.R.*), NC (*Gr*), EM (*Nat*)
Dermis (*Rgd*)	CFBs (*R.St*), NC(*Ag.Gr*), EM (*Nat*)

As displayed in Table 4.3 when the Dermis is in the Flexible (*Fl*) state, the collagen fibrils bundles, neurosecretory cells, and extracellular matrix are in the Releasing Stiparin Inhibitor (*R.Stp.*I), Ungrouped (*UnGr*), and Natural (*Nat*) states, respectively. The state mappings for the Dermis in the Natural (*Nat*) and Rigid (*Rgd*) state are similarly defined in Table 4.3. After the standalone behaviors and state mappings for the subsystems of the dermis are determined, the behaviors for the subsystems of the Collagen Fibril Bundles are defined (displayed in Figure 4.15 and Table 4.4). The subsystems of the Collagen Fibril Bundles are Stiparin Inhibitor (*Stip.Inh*), Stiparin (*Stip*), Collagen Fibrils (*CFs*), Stiffener (*Stiff*), and Microfibrils (*MFs*).

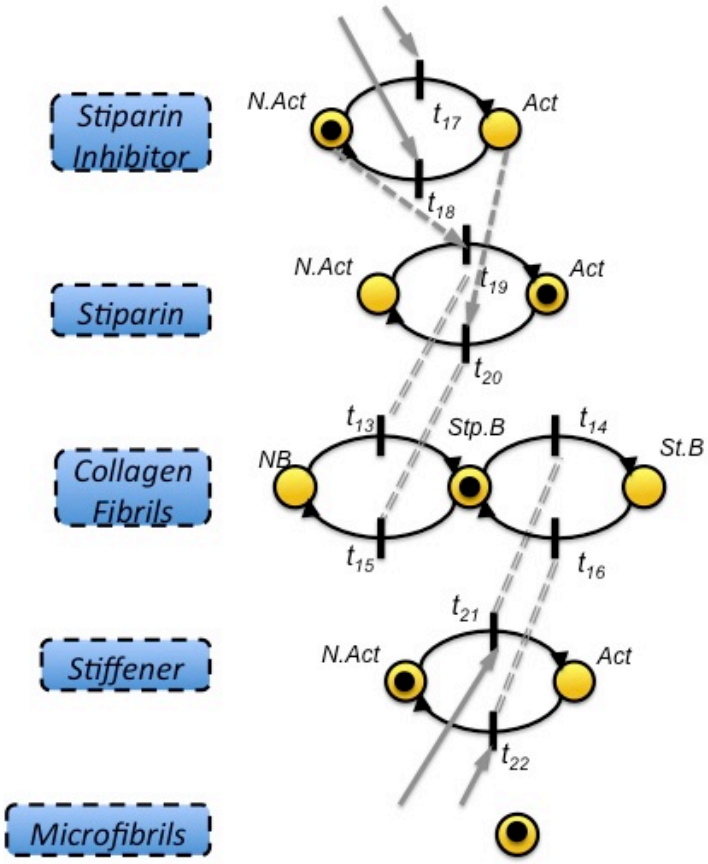


Figure 4.15 Standalone behaviours for subsystems of Collagen Fibril Bundles

Table 4.4 Terms for Figure 4.15

N.Act	Not Active	t_{17}, t_{19}, t_{21}	Activate
Act	Active	t_{18}, t_{20} ,t_{22}	De-activate
NB	No binding	t_{13}	Associate fibrils
Stp.B	Stiparin-bound	t_{14}	Bind fibrils
St.B	Stiffener-bound	t_{15}	De-associate fibrils
		t_{16}	Un-bind fibrils

Next, the states of the collagen fibril bundles are mapped to that of its subsystems. The mappings are displayed in Table 4.5.

Table 4.5 State Mappings for Collagen Fibril Bundles

CFBs(*UnGr*)	Stip.Inh(*Act*), Stip (*N.Act*), CFs (*NB*), Stiff (*N.Act*), MFs (*nat*)
CFBs (*Gr*)	Stip.Inh(*N.Act),* Stip (*Act*), CFs (*Stp.B*), Stiff (*N.Act*), MFs (*nat*)
CFBs (*Ag.Gr*)	Stip.Inh(*N.Act*), Stip (*Act*), CFs (*Stp.B*), Stiff (*Act*), MFs (*nat*)

5) Define interface relationships between subsystems

The interface relationships are defined in Figure 4.14 and Figure 4.15 above. Precedence arcs are denoted by dashed arrows and synchronous arcs are denoted using the double, dashed lines.

6 - 8) Generate combined behavioral model, Identify Subnets, and Create Macrotransitions

Since steps 6-8 are iterative, the discussion below is combined.

Next, the reachability graph of the lowest decomposition level is generated. The reachability graph begins with the initial state (or marking) of the combined system. Next, using the firing sequences and the external arcs of the system, the combined behavioral model is generated. The combined behavioral model for the subsystems of the collagen fibril bundles is displayed in Figure 4.16. As displayed in the figure, the initial marking of the system is [Stip.Inh(*N.Act*), Stip (*Act*), CFs (*Stp.B*), Stiff (*N.Act*), MFs (*nat*)]. When transition t_{21} and t_{14} (linked by a synchronous relationship) are fired, the marking of the system becomes [Stip.Inh (*N.Act*), Stip (*Act*), CFs (*Stp.B*), Stiff (*Act*), MFs (*nat*)], denoting an aggregated/grouped state of the collagen fibril bundles.

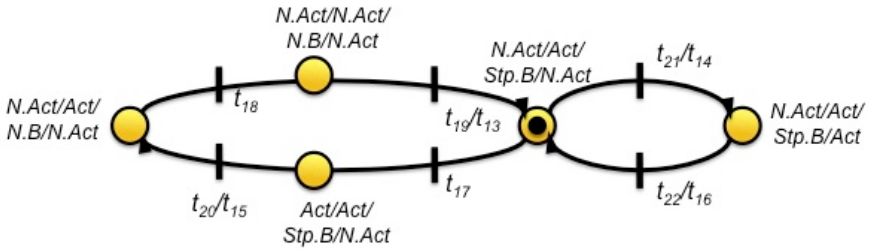


Figure 4.16 Combined Behavioral Model for subsystems of the Collagen Fibril Bundles

Next, the subnets are identified using the isomorphic state mappings in Table 4.5 above. Using the inheritence procedure, the subnets are inherited to the macrotransitions of the collagen fibril bundle system. For instance, following the definition put forth in Section 4.1.2.2, subnet (t_{18},*(N.Act/N.Act/N.B/N.Act),* t_{19}/t_{13}) is inherited by macrotransition t_5 of the collagen fibril bundle behavior graph. This process is displayed in Figure 4.17. In Figure 4.17, subnets are denoted by a gray shadow with its associated

macrotransition listed in a white box. Macrotransitions are denoted using a double black bar. Similarly, subnets (t_{17},*(Act/Act/Stp.B/N.Act)*,t_{20}/t_{15}), (t_{21}/t_{14}), and (t_{22}/t_{16}) are inherited by macrotransitions t_{11}, t_{10}, and t_{12} respectively. Using this notation, we can view how the behavior of the subsystems contributes to that of the higher level system. In this case, we can view how the subsystems of the collagen fibril bundles contribute to that of the system.

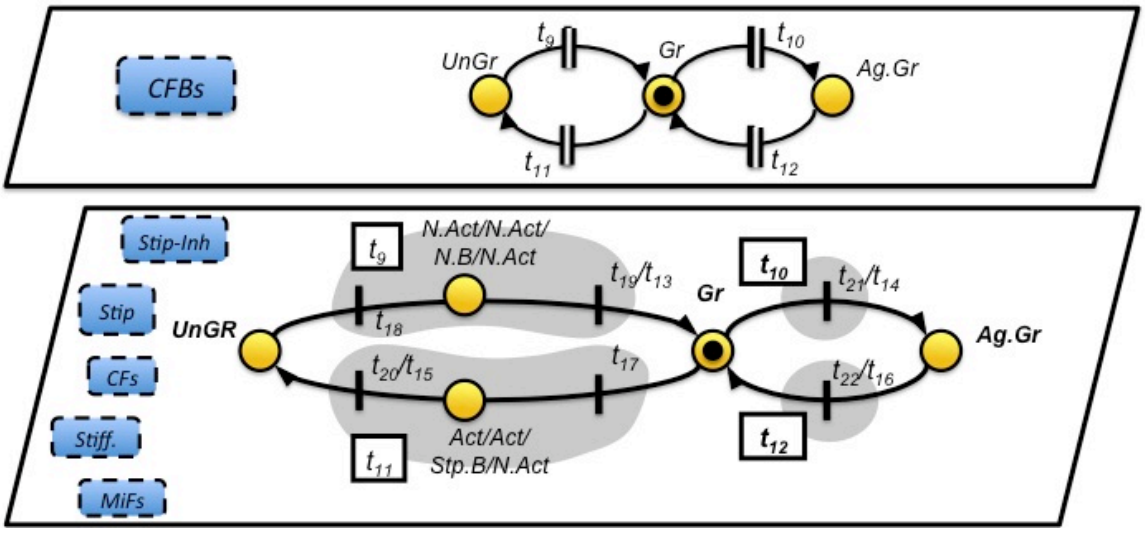


Figure 4.17 Identification of subnets for Collagen Fibril Bundles

Next, the combined behavioral graph for the subsystems of the dermis (Collagen fibril bundles, neurosecretory cells, and extracellular matrix) is generated using the reachability graph. This graph is displayed in Figure 4.18.

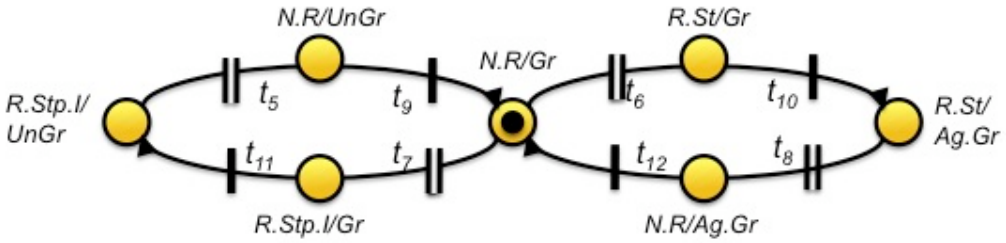


Figure 4.18 Combined Behavior Graph for subsystems of the Dermis

In the figure, notice that the subnets of the lower systems are included in the combined behavioral graphs as macrotransitions t_5, t_6, t_7, t_8. Using this combined

behavioral graph, the subnets are identified and inherited into the behavior graph of the dermis. The overall hierarchical model is displayed in Figure 4.19. The overall hierarchical model displays three levels of abstraction of the Dermis system. As can be seen in Figure 4.19, hierarchical Petri net modeling allows us to map the behaviors of the lower level subsystems to that of the Dermis. The hierarchical relationships are expressed in the form of macrotransitions and subnets.

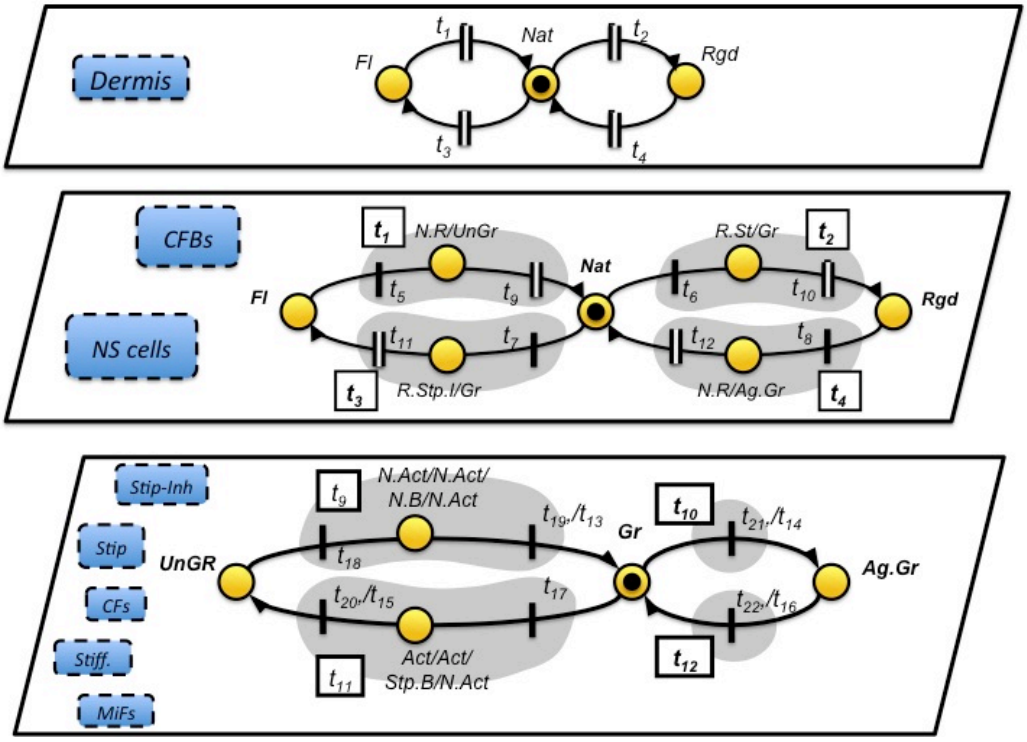


Figure 4.19 Overall Hiearchical PN Model for Sea Cucumber Dermis

Based on this hierarchical model displayed in Figure 4.19, the behavior of the dermis can be viewed at multiple levels of abstraction. This model combines the individual subsystem behaviors found in Figures 4.10, 4.14 and 4.15. This multi-layered view allows us to systematically extract the functional strategy from the system.

9) Extract system strategy

Based on the hierarchical net in Figure 4.19, the strategy can be extracted as follows:

Strategy (*Nat, Rgd*) = (t_6, t_{10})=(NCs(*Release stiffener*),CFBs(*aggregate fibrils*)). We can also expand t_{10} to reveal another layer of behavior as Strategy (*Nat, Rgd*) = (t_6 ,t_{21}/t_{14})= (NCs(*Release stiffener*),Stiffener(*activated*), CFs(*bind fibrils*)). Using natural language, the strategy is as follows:

"An increase of stiffness from the natural state of the dermis to the rigid state is caused by release of stiffener by the neurosecretory cells, which causes the collagen fibril bundles to aggregate. The stiffener released by the neurosecretory cells becomes active and binds the individual fibrils together, causing fibril aggregation."

Following the same procedure, Strategy (*Nat, Fl)* = (t_7, t_{11})=(NCs(*release stiparin-inhibitor*), CFBs(*ungroup fibrils*)). By expanding t_{11}, Strategy (*Fl, Nat*) = (t_7, (t_{17}, t_{20}/t_{15})) = (NCs(*release stiparin-inhibitor*), ((Stiparin-Inhibitor (*activate stiparin-inhibitor*), Stiparin (*de-activate stiparin*)/CFs(*de-associate fibrils)*). Using natural language, the strategy is as follows:

"*A decrease in stiffness from the natural state of the dermis to the flexible state is caused by the release of stiparin-inhibitor by the neurosecretory cells, causing the collagen fibril bundles to ungroup. The stiparin-inhibitor released by the neurosecretory cells de-activates the stiparin, causing de-association of the individual collagen fibrils. This causes the collagen fibril bundles to ungroup."*

By considering the strategies used for changes of state of the entire system, the overall strategy can be abstracted and stated in simpler terms as:

"Stiffness in the dermis is changed by controlling the association of the collagen fibril bundles"

In Section 4.4.2, we use the method for Reverse Engineering Biological Systems to extract the behavioral strategy from Muscle Fiber.

4.4.2 Strategy Extraction from Muscle Fiber in Isometric Contraction

For this example, behavioral strategy is extracted from the human muscle using the method for Reverse Engineering Biological Systesm. The human muscle is displayed in Figure 4.20.

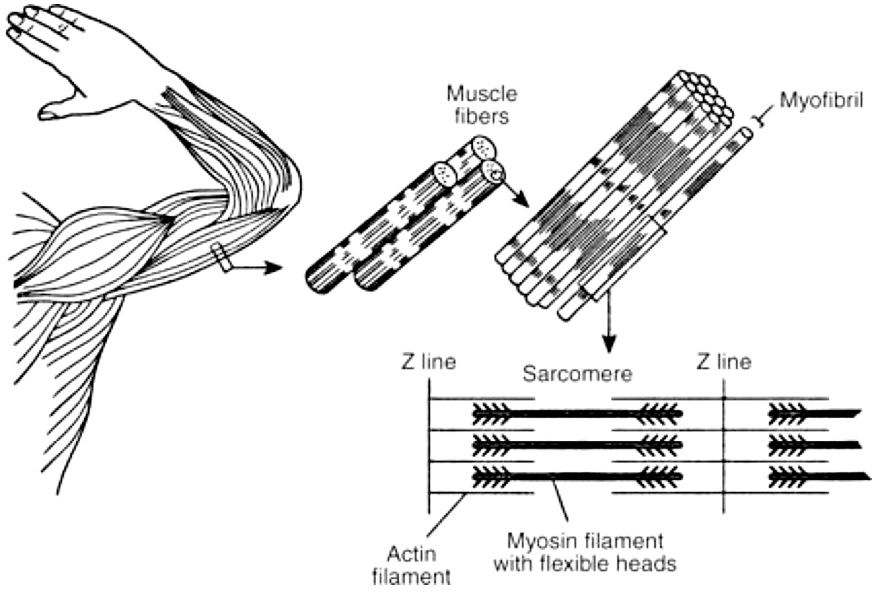


Figure 4.20 Human Muscle (Figure 3.2)

1) Define root system:

In this step, the root system is identified as the muscle fiber subunit of human muscle. The boundary of the system is drawn just around the muscle fiber. The interactions with the environment include an action potential (electrical) from the motor end plate coming into the system, controlling the mechanical energy (force) output to the environment as the muscle fiber changes stiffness. As in the case of the sea cucumber dermis, the attribute of interest is defined as the stiffness of the muscle fiber. The muscle fibers and its interactions with the environment are displayed in Figure 4.21.

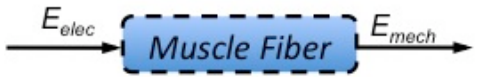


Figure 4.21 Root system (Muscle Fiber)

2) Define standalone behavior:

The behavior of the root system, the muscle fiber, is modeled using the Petri net modeling formalism. This model is displayed in Figure 4.22. The states of the system are defined as Flexible (*Fl*) and Rigid (*Rgd*). The transitions are defined in Table 4.6.

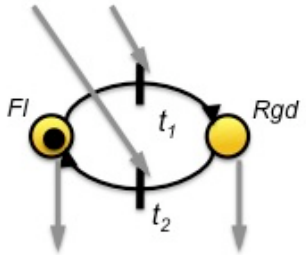


Figure 4.22 PN model of Muscle Fiber

Table 4.6 Terms for PN model of Dermis

t_1	Increase stiffness
t_2	Decrease stiffness

3) Decompose system and sub-systems

The system is decomposed into its subsystems in Step 3. Through review of literature [134, 135] on the muscle fiber, the physical structure was decomposed. This decomposition is displayed in Figure 4.23.

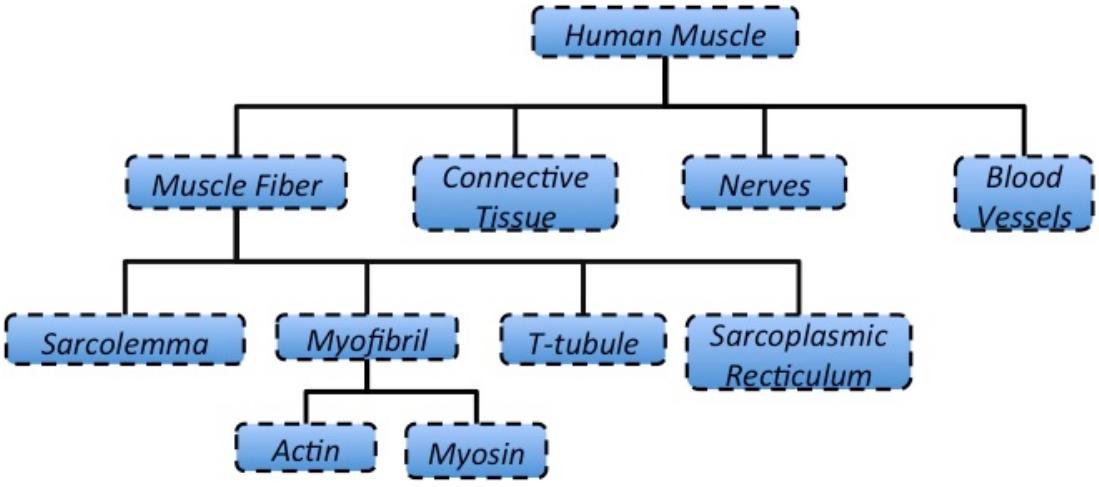


Figure 4.23 Structural Decomposition of the Human Muscle

The human muscle is composed of the muscle fiber, connective tissue connecting the fibers, nerves, and blood vessels. The muscle fiber can be further decomposed into the sarcolemma, myofibrils, transverse tubules (T-tubules), and the sarcoplasmic reticulum. The sarcolemma acts as the muscle fiber's plasma membrane. The myofibrils are the fiber's "contractile machinery". Each myofibril is a bundle of overlapping thick and thin filaments. The thick filaments are composed of the protein myosin and thin filaments composed of the protein actin. The transverse tubules and sarcoplasmic reticulum both play a key role in the activation of the muscle fiber. An action potential (E_{elec}), from the axon, propagates through the sarcolemma and down the T-tubules to the interior of the cell. This action potential triggers Ca^{2+} release (E_{chem}) in the sarcoplasmic reticulum, which causes the thick and thin filaments of the myofibril to bridge. Specifically, the Ca^{2+} release (E_{chem}) in the sarcoplasmic reticulum exposes the myosin-binding sites on the actin filament, which triggers (S_{chem}) the myosin filaments to bridge (E_{mech}) with the actin filaments. These interactions are displayed in Figure 4.24 and Figure 4.25.

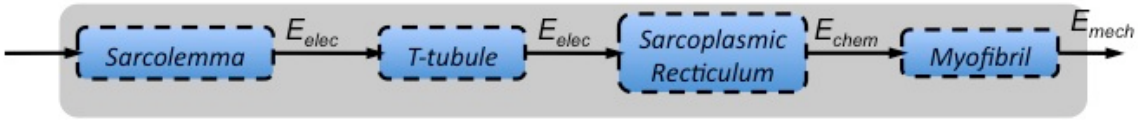


Figure 4.24 Interactions for first layer of the Muscle Fiber decomposition

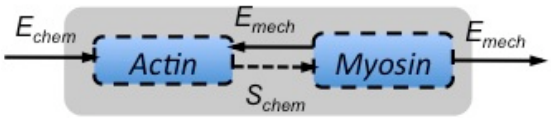


Figure 4.25 Interactions in Myofibril decomposition

4) Define standalone behaviors of sub-systems

In step 4, the standalone behaviors of each of the subsystems are modeled using the Petri net model (displayed in Figure 4.26). The states and transitions of the behavioral models are displayed in Table 4.7.

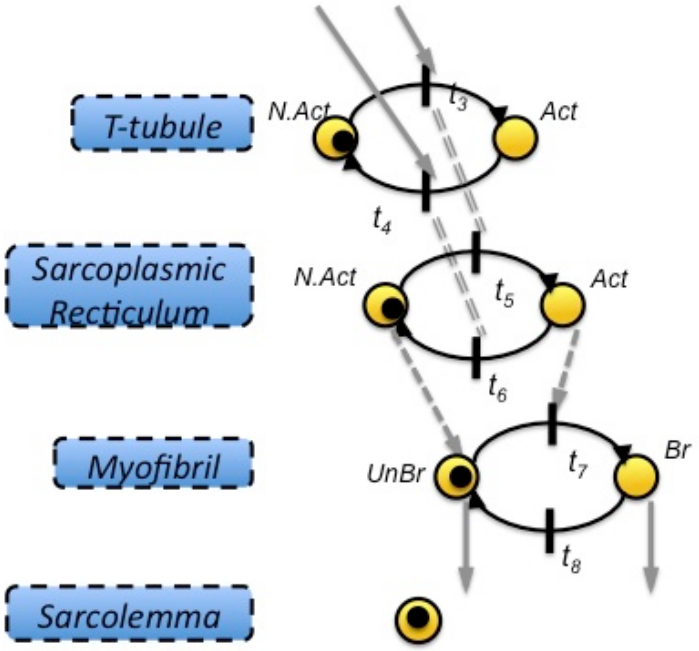


Figure 4.26 Standalone behaviors of the subsystems of the Muscle Fiber

Table 4.7 Terms for Figure 4.26

N.Act (T-tubule)	Action potential not active	t_3	Transmit action potential
Act (T-tubule)	Action potential active	t_4	Stop action potential
N.Act (Sarc. Reticulum)	No Ca^{2+} release	t_5	Activate release of Ca^{2+}
Act (Sarc. Reticulum)	Ca^{2+} release active	t_6	Stop release of Ca^{2+}
UnBr	Unbridged	t_7	Bridge myofibrils
Br	Bridged	t_8	Unbridge myofibrils

In this step, the states of the subsystems are mapped to that of the root system. In the case of the muscle fiber, the states of the T-tubules (T-ts), Sarcoplasmic Reticulum (SR), Myofibril (Myo) and the Sarcolemma (S) are mapped to the states of the muscle fiber. This mapping is displayed in Table 4.8. The state of the Sarcolemma does not change, thus it is in its Natural state (Nat).

Table 4.8 State Mappings for the Muscle Fiber

Muscle Fiber (*Fl*)	T-ts (*N.Act*), SR (*N.Act*), Myo (*UnBr*), S (*Nat*)
Muscle Fiber (*Rgd*)	T-ts (*Act*), SR (*Act*), Myo (*Br*), S (*Nat*)

As displayed in Table 4.8, when the Muscle Fiber is in the flexible (*Fl*) state, the T-tubules and Sarcoplasmic Reticulum are not active (*N.Act*) and the myofibril is in its unbridged (*UnBr*) state. When the Muscle Fiber is in its rigid (*Rgd*) state, the T-tubules are transmitting the action potential (Act), the Sarcoplasmic Reticulum is releasing Ca^{2+} (*Act*), and the Myofibrils are in the bridged (*Br*) state.

The subsystems of the Myofibril, the actin and myosin filaments, can now be modeled. The standalone behaviors and state mappings are displayed in Figure 4.27 and Table 4.9.

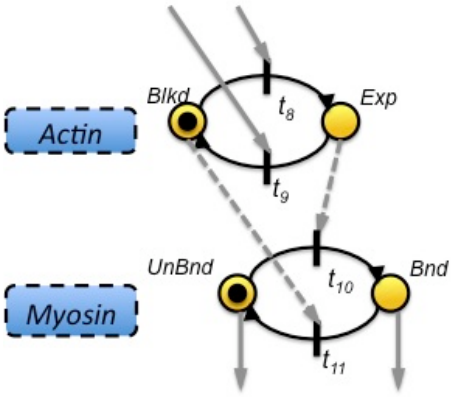


Figure 4.27 Standalone behaviors for subsystems of Myofibril

Table 4.9 Terms for Figure 4.27

Blkd	Myosin-binding sites blocked	t_9	Expose myosin-binding sites
Exp	Myosin-binding sites exposed	t_{10}	Block myosin-binding sites
UnBnd	Myosin unbound to actin	t_{11}	Fire myosin head
Bnd	Myosin bound to actin	t_{12}	Release myosin head

Next, the states of the collagen fibril bundles are mapped to that of their subsystems. The mappings are displayed in Table 4.10.

Table 4.10 State Mappings for Myofibril

Myo (*UnBr*)	Actin(*Blkd*), Myosin (*UnBnd*)
Myo (*Br*)	Actin(*Exp*), Myosin (*Bnd*)

5) Define interface relationships between subsystems

The interface relationships are defined in Figure 4.26 and Figure 4.27. Precedence arcs are denoted by dashed arrows and synchronous arcs are denoted using the double, dashed lines.

6-8) Generate combined behavioral model, Identify Subnets, and Create Macrotransitions

Since steps 6-8 are iterative, the discussion below is combined.

The reachability graph of the lowest decomposition level, the actin and myosin filaments are now generated. The combined behavioral model for the actin and myosin filaments is displayed in Figure 4.28. As displayed in the figure, the initial marking of the system is [Actin(*Blkd*), Myosin (*UnBnd*)]. When transition t_9 fires, the marking of the system becomes [Actin(*Exp*), Myosin (*UnBnd*)], denoting that the myosin-binding site of the actin filament is exposed.

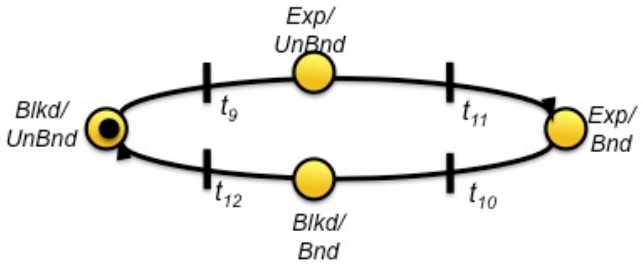


Figure 4.28 Combined Behavioral Model for subsystems of the Myofibril

Next, using the isomorphic state mappings in Table 4.10 above, the subnets are identified. Following the inheritance procedure, the subnets are inherited to the macrotransitions of the Myofibril. The subnets and macrotransitions of the Myofibril are displayed in Figure 4.29.

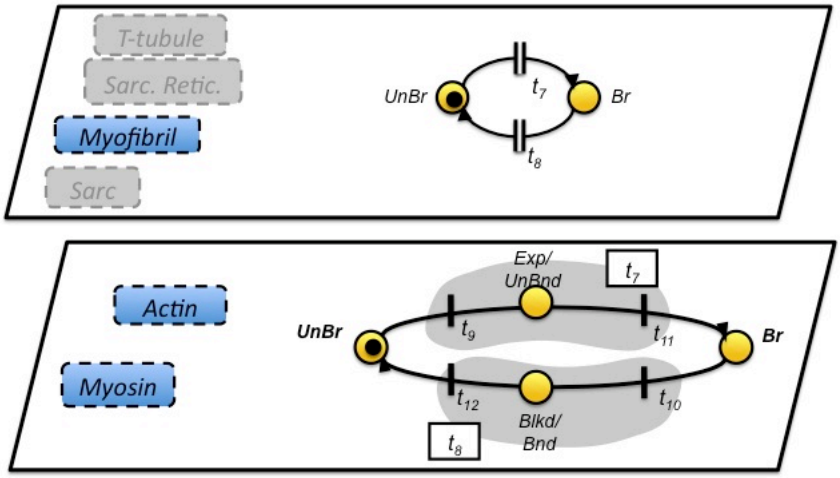


Figure 4.29 Identification of subnets for the Myofibril

As displayed in Figure 4.29, subnet (t_9,(*Exp/UnBnd*),t_{11}) is inherited to macrotransition t_7 of the Myofibril. Next, the combined behavioral graph for the subsystems of the Muscle Fiber (Sarcolemma, Myofibril, Sarcoplasmic Reticulum, and T-tubule) is generated using the reachability graph. This graph is displayed in Figure 4.30.

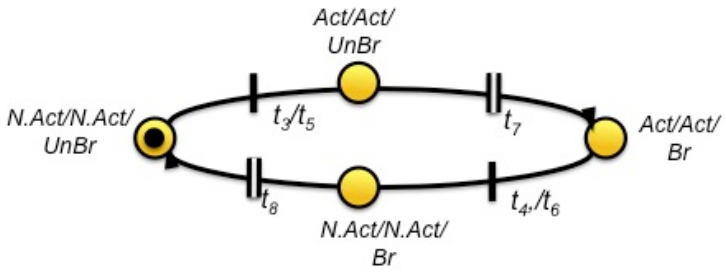


Figure 4.30 Combined Behavior Graph for subsystems of the Dermis

Using this combined behavioral graph, the subnets are identified and inherited into the behavior graph of the Muscle Fiber. The overall hierarchical model of the behavior of the Muscle Fiber is displayed in Figure 4.31.

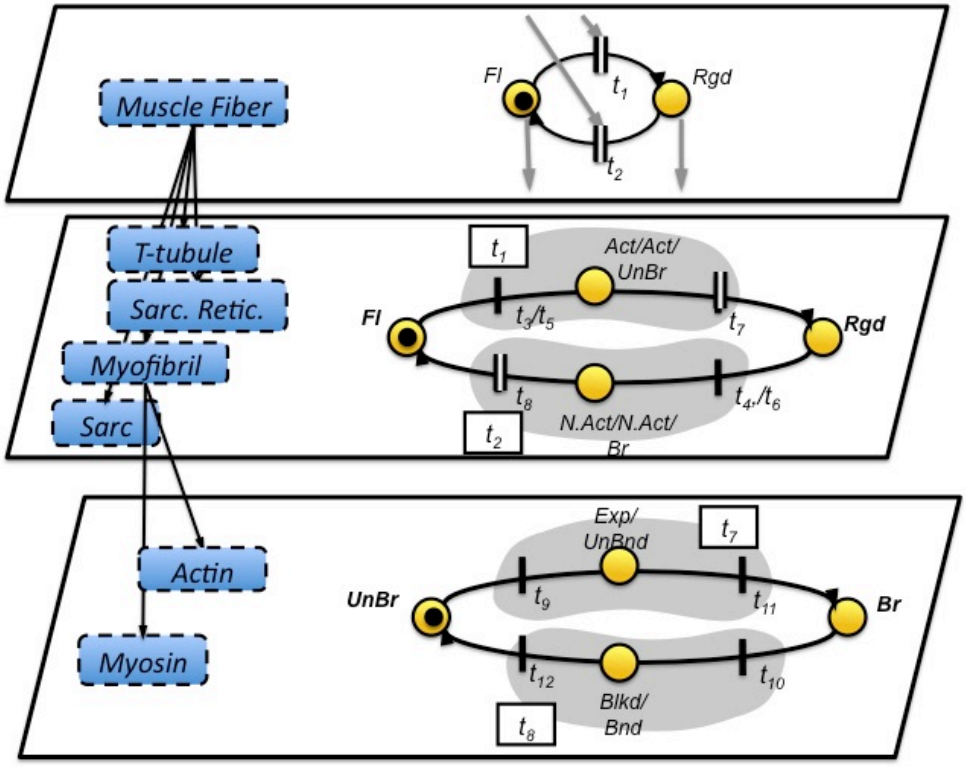


Figure 4.31 Overall Hierarchical PN Model for Sea Cucumber Dermis

Based on this hierarchical model displayed in Figure 4.31, the behavior of the dermis can be viewed at multiple levels of abstraction. This multi-layered view allows us to systematically extract the functional strategy from the system.

9) Extract system strategy

Based on the hierarchical net in Figure 4.31, the strategy can be extracted as follows:

Strategy (*Fl, Rgd*) = (t_3/t_5, t_7)=(T-ts(*transmit action potential*)/Sarc. Reticulum (*Activate release of* Ca^{2+}), Myofibril (*Bridge myofibrils*)). We can also expand t_7 to reveal another layer of behavior as Strategy (*Nat, Rgd*) = (t_3/t_5,t_9,t_{11}) = (T-ts(*transmit action potential*)/Sarc. Reticulum (*Activate release of* Ca^{2+}), Actin (*Expose myosin-binding sites*), Myosin(*Fire myosin head*)). Using natural language, the strategy is as follows:

"An increase of stiffness from the flexible state of the Muscle Fiber to the rigid state is caused by the following process: an action potential of the T-tubules causes the Sarcoplasmic Reticulum to release Ca^{2+}, which then causes the Myofibrils to become bridged. The Myofibrils become bridged because the Ca^{2+}causes myosin-binding site of the actin filament to be exposed, which then triggers the myosin filament to fire and bind to actin."

Following the same procedure, Strategy (*Rgd, Fl)* = (t_4/t_6, t_8)=(T-ts(*stop action potential*)/Sarc. Reticulum (*stop release of* Ca^{2+}), Myofibril (*Unbridge myofibrils*)). By expanding t_8, Strategy (*Rgd, Fl*) = (t_4/t_6, t_{10} ,t_{12}) = T-ts(*stop action potential*)/Sarc. Reticulum (*stop release of* Ca^{2+}), Actin (*Block myosin-binding sites*), Myosin(*release myosin head*)). Using natural language, the strategy is as follows:

"A decrease in stiffness from the rigid state of the Muscle Fiber to the flexible state is caused by an unbridging of the myofibrils, which unbridge in the absence of Ca^{2+} *release by the Sarcoplasmic Reticulum. The absence of* Ca^{2+} *causes the myosin-binding site of the actin filament to become blocked again, which causes the head of the myosin filament to retract."*

By considering the strategies used for changes of state of the entire system, the overall strategy can be abstracted and stated in simpler terms as:

"Stiffness in the Muscle Fiber is changed by controlling the bridging of the actin and myosin filaments of Myosin"

4.5 CLOSURE AND VALIDATION

At the onset of this chapter, the following question was proposed:

(RQ2) *How can the behavior of biological systems be hierarchically represented using Petri nets, while preserving the fundamental properties at each hierarchal level?*

It was hypothesized in Hypothesis 2 that using the systematic method for Reverse Engineering Biological Systems will ensure that the fundamental properties of boundedness, reachability, and liveness will be preserved across hierarchical levels. To

validate this hypothesis, mathematical proofs of boundedness, liveness, and reachability across hierarchical levels were presented in Section 4.3. Through this analysis, it was concluded that the proposed method does indeed ensure that the fundamental properties are preserved. This was largely due to the use of the reachability graph to generate the combined behavioral graph and the subnet inheritance definition. Example problems of the use of the proposed method in representing biological system behavior and extracting behavioral strategies were also presented in Section 4.4.

Theoretical Structural Validity

With respect to our validation strategy presented in Figure 4.32, theoretical structural validity was addressed.

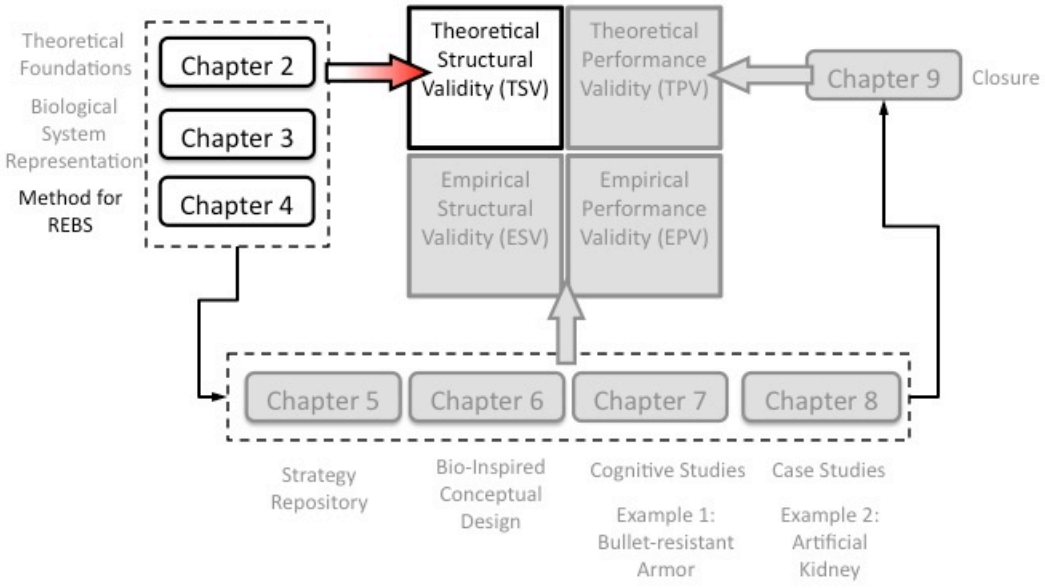


Figure 4.32 Validation Summary for Chapter 4

Theoretical Structural Validation involves checking the individual constructs and assumptions upon which the method is built, as well as checking the internal consistency of the method when combining the individual constructs. In Chapters 2 and 3, the theoretical constructs of the method were addressed. In this chapter, the internal consistency of the method is checked using a flowchart and systematic steps presented in Section 4.2.

In this chapter, the method for Reverse Engineering Biological Systems was presented. In Chapter 5, identification of relevant biological solutions and strategies is addressed. Specifically, a repository structure for efficiently storing and retrieving biological strategies is presented.

CHAPTER 5 STRATEGY REPOSITORY DEVELOPMENT

The overarching aim of this research is that of aiding the designer in the ideation process through the use of biological strategies. We believe that leveraging biological strategies in the design process will lead to a more thorough navigation of the designer's design space. In Chapters 3 and 4, the hierarchical Petri net (hPN) representation for biological systems, as well as a method to extract behavioral strategies from these representations, was presented. In this chapter, a repository is developed to capture biological (and engineering) strategies and allow speedy access to these strategies in the Conceptual Design process. The hPN representation is used to structure this repository. The outline of this dissertation is displayed in Figure 5.1.

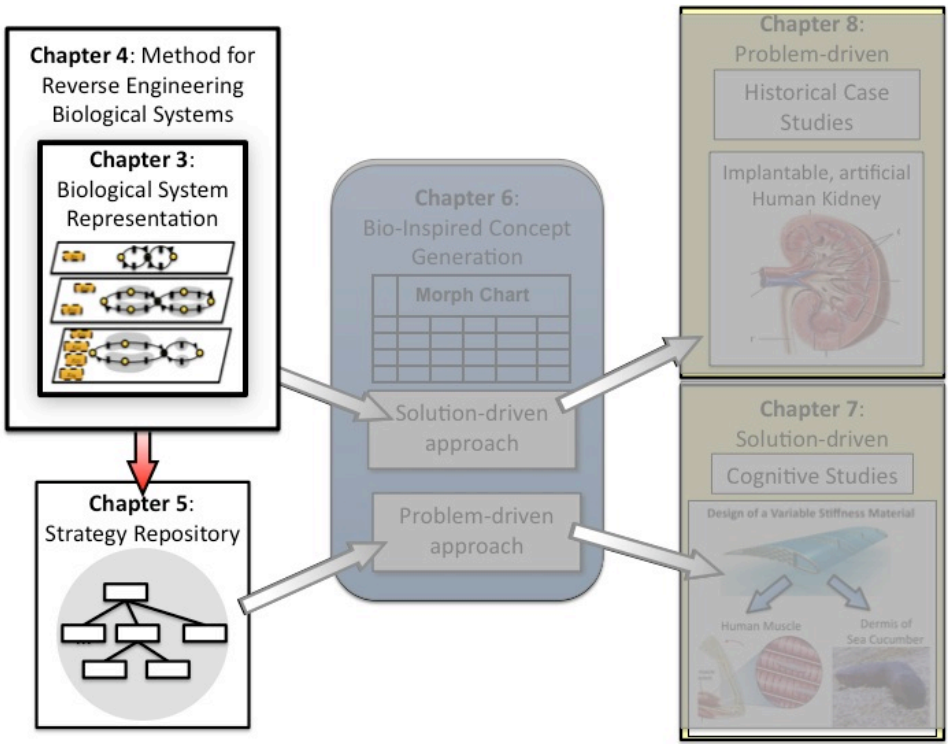


Figure 5.1 Chapter 5 and the Dissertation Outline

Several researchers have attempted to systematize the bio-inspired design process. Vincent and coauthors [3, 13] seek to integrate knowledge from nature into TRIZ, a systematic method for inventive problem solving developed by Russian researchers.

Researchers from the University of Toronto [136, 22, 23] proposed using a functional keyword search through biological literature to identify potential analogies. Researchers from the Rocky Mountain Institute/Biomimicry Guild and the University of Maryland [26] have developed searchable databases of biological systems. Other researchers [24] have developed a searchable database containing both natural and artificial systems. Refer to Section 2.2 for a lengthier presentation of the current approaches.

Although the current approaches are useful in storing and providing access to biological information in design, the generic keyword-based retrieval process often suffers by either providing too many and/or irrelevant design results [30]. By structuring biological information using ontologies, biological strategies can be more efficiently retrieved from a knowledge base. Specifically, in this chapter, we ask the following research question:

" How can hierarchical Petri net representations of biological systems be structured to aid retrieval of relevant strategies from a knowledge repository?"

To answer this question, it was hypothesized in Chapter 1 that:

Hypothesis 3: An ontology of concepts from hierarchical Petri net representations of biological systems can be represented using Description Logics. Subsumption in Description Logics will enable consistent and precise retrieval of relevant biological strategies from a knowledge repository.

To test this hypothesis, an ontology is developed for representing the strategy, as well as the functional, behavioral, and structural information, of biological and engineering systems (Section 5.1). Next, Description Logics (DL) are used to encode this ontology into the repository (Section 5.2). Once encoded, the subsumption inference mechanisms in DL are utilized to ensure consistent and precise retrieval of biological and engineering strategies (Section 5.3). Lastly, a testbed repository of biological and engineering strategies is developed to empirically evaluate the retrieval process afforded by subsumption in DL (Section 5.4)

5.1 ONTOLOGY DEVELOPMENT

In this research, biological systems are represented using a hierarchical Petri net representation (Chapters 3 and 4) and behavioral strategies are extracted from these systems using this representation. The goals of the repository are to aid the designer in identifying relevant biological systems and allowing access to their respective strategies. To aid in retrieval of these strategies, the information contained in the hPN representation is structured using an ontology. An overview of the Petri net representation is displayed in Figure 3.12, and presented here for convenience.

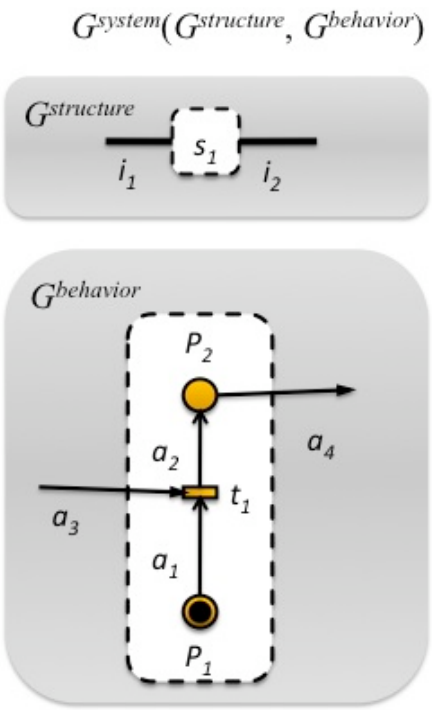


Figure 5.2 Petri net representation (Figure 3.11)

To develop the ontology, this research follows the steps outlined by [30] for ontology development. First the overall schema, or scope, of the ontology is identified and primary concepts identified. Next, taxonomies under these individual concepts are constructed. Inter-relationships are then formed between concepts across taxonomies. Lastly, the ontology is structured using the concepts and inter-relationships between these concepts.

5.1.1 Schema definition

The first step in developing an ontology is to define the overall schema of the ontology [30]. This schema indicates the relevant set of concepts that will be represented in the ontology. The following concepts, or themes, of the hPN model are important for retrieval in Conceptual Design, and must be explicitly represented in the ontology.

Functional information - Functional information is needed to aid in the retrieval of these strategies from the repository. In the Conceptual Design phase of P&B [5], one of the primary tasks of the designer is the search for working principles to fulfill the function of interest. Because of this, systems and their associated behavioral strategies are represented by the functions they achieve. In this research, function is viewed as a mapping of the behavior of a system to the behavior of its supersystem [54]. In the hPN representation (see Figure 5.2), function is defined as a tuple of the driving inputs (a_3, in Figure 5.2), and the functional outputs (a_4) of the system. Therefore, in the ontology, the driving inputs and functional outputs of the system should be explicitly represented. For example, in the search for a system that produces force as a response to an electrical input, biological and engineering strategies can be supplied that fulfill this function, such as that of piezoelectric effect.

Behavioral information – Behavioral information includes information on how a particular system achieves its function. Behavioral information needs to be described for situations where designers are looking for novel strategies for performing a particular behavior. In the hPN, behavior is defined as the intrinsic change of state of the system. Specifically, in the hPN representation, behavior is defined by the states of specific attributes of the system (P) and the actions governing the change of state (t). In this ontology, the attributes by which the system is defined as well as the actions governing a change of state of these attributes should be explicitly defined. For example, if the designer is searching for a particular strategy for increasing the stiffness of a particular system, strategies for electrorheological fluids can be supplied.

System Strategy- The aim of the repository is allow access to and retrieval of relevant biological and engineering strategies. Therefore, this information must be explicitly represented in the ontology. In this research, the strategy of a system is defined as the means by which a system achieves a behavior. In the hPN representation, the strategy is considered the behavior of the lower level subsystems that contribute to the behavior of the system, and defined using the subnets of behaviors (*t*). The strategy is extracted from the hPN representation through simulation of the net. For example, the behavior of a magnetorheological fluid may be defined as "increase stiffness", whereby the strategy can be stated shortly as the *"stiffness in the MR fluid is changed by controlling the bonding between magnetic particles in a carrier fluid with a magnetic field"*. Strategy gives a much richer view of behavior, describing how the individual components of the system contribute to the system behavior of increasing stiffness.

Structure – Structural information allows the user to identify the strategies of specific systems. The structure of the system is the identifier for the component of the system. In this research, the terms system and structure are used synonymously. In the hPN, the system is defined by its boundary.

Domain – Domain allows the designer to do a specialized search for a system or strategy within a particular domain, such as a biological or engineering domain.

The schema can now be described as including functional, behavioral, structural, strategy, and domain information. The specific concepts from the hPN representation that will be defined are as follows: function by its driving inputs and functional outputs, behavior by its attributes and actions governing change of these attributes, system type, and environment of operation. The ontology model is displayed in Figure 5.3.

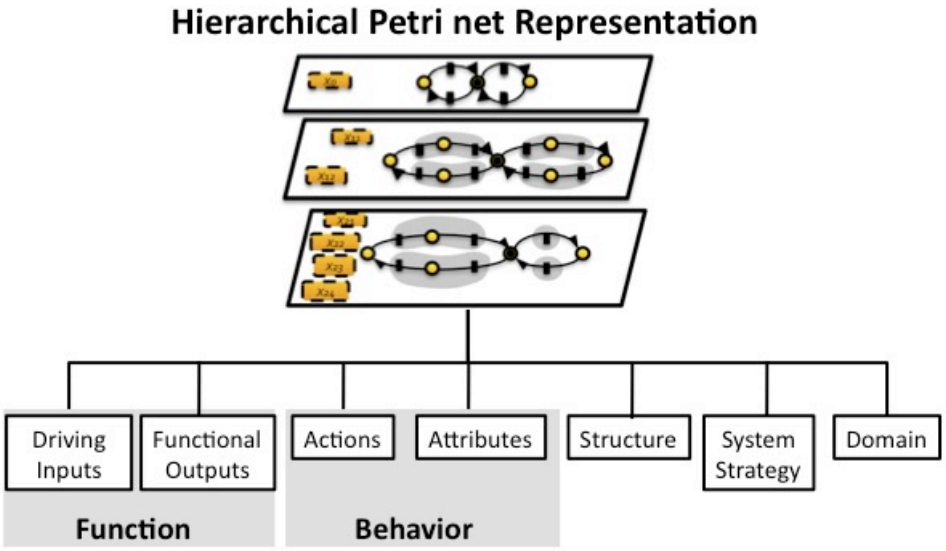


Figure 5.3 Overall schema for strategy ontology

5.1.2 Taxonomy development

Now that the concepts of the ontology have been defined, taxonomies of these concepts are constructed. "Taxonomies are hierarchical classifications of concepts within a subdomain" [30]. Taxonomies utilize 'is_a' relationships, which denote parent-child relationships between concepts. Taxonomies for the concepts defined in Figure 5.3 are as follows:

Flow (Driving Input and Functional Output) Taxonomy

The driving inputs and functional outputs of the systems are flows of energy, material, and signals into and out of the system. To define these taxonomies, we leverage the functional basis [30], a classification of functions (verbs) and flows (nouns) used to formally describe the function of a system. The functional basis integrates research efforts from the National Institute of Standards and Technology (NIST) and two US universities and their industry partners into a single classification system. Specifically, we leverage the flow classification scheme from the functional basis to define the flow taxonomy. A sample of the flow taxonomy is displayed in Figure 5.4.

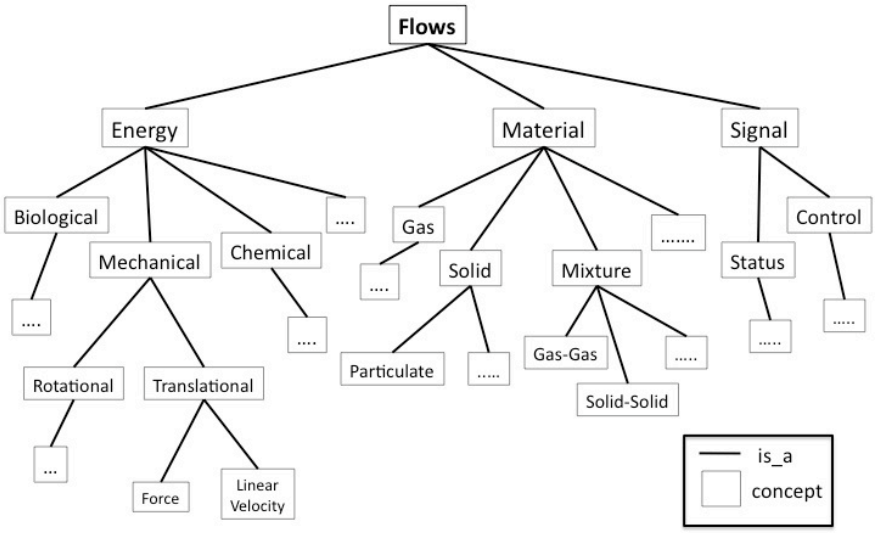


Figure 5.4 Sample of Flow Taxonomy

Action Taxonomy

Given a driving input, the actions of the system govern the change of state of the system. These actions are described using verbs, such as increase, control, stop, etc. The action taxonomy also leverages the work done in the functional basis. In the functional basis, a verb classification scheme is developed to describe the action of the function. In this case, we use these verbs to describe the actions of the behavior of the system. In evaluating the functional basis, Ahmed and Wallace [137] found that the functional basis could be used to describe 94% of the verbs used by engineers in describing their designs.

A sample of the action taxonomy is displayed in Figure 5.5.

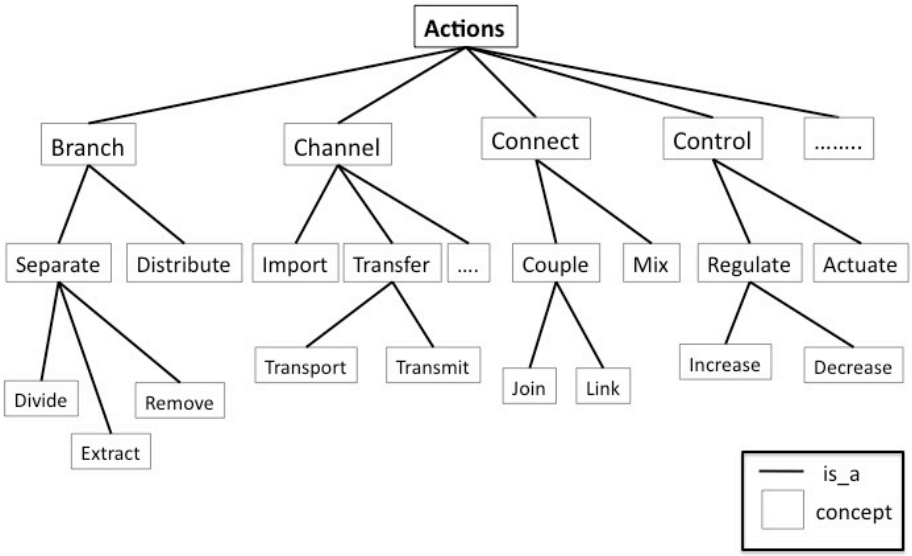


Figure 5.5 Sample of the Action Taxonomy

Attribute taxonomy

The attributes of the system are used to define the context by which the states of the system are defined. Attributes are defined using properties of the system. To define the attribute taxonomy, a comprehensive survey of mechanical engineering textbooks and reference books was performed. Common properties were classified and the taxonomy was structured. A sample of the property taxonomy is displayed in Figure 5.6.

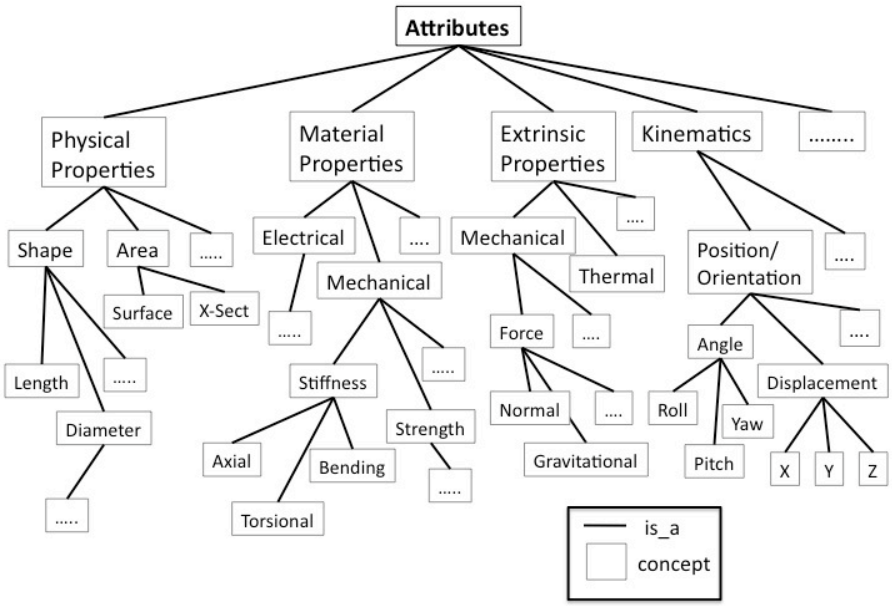


Figure 5.6 Sample of the Attributes Taxonomy

System strategy taxonomy

The system strategy is the means by which the behavior of the system is performed. The strategy taxonomy is composed of defined concepts, meaning that not only is the taxonomy defined by is_a relationships, but is also defined by relationships with other concepts. These inter-relationships are defined in Section 5.1.3. A sample of the strategies entered into the ontology is displayed in Figure 5.7. In Figure 5.7, these strategies are only structured using is_a relationships.

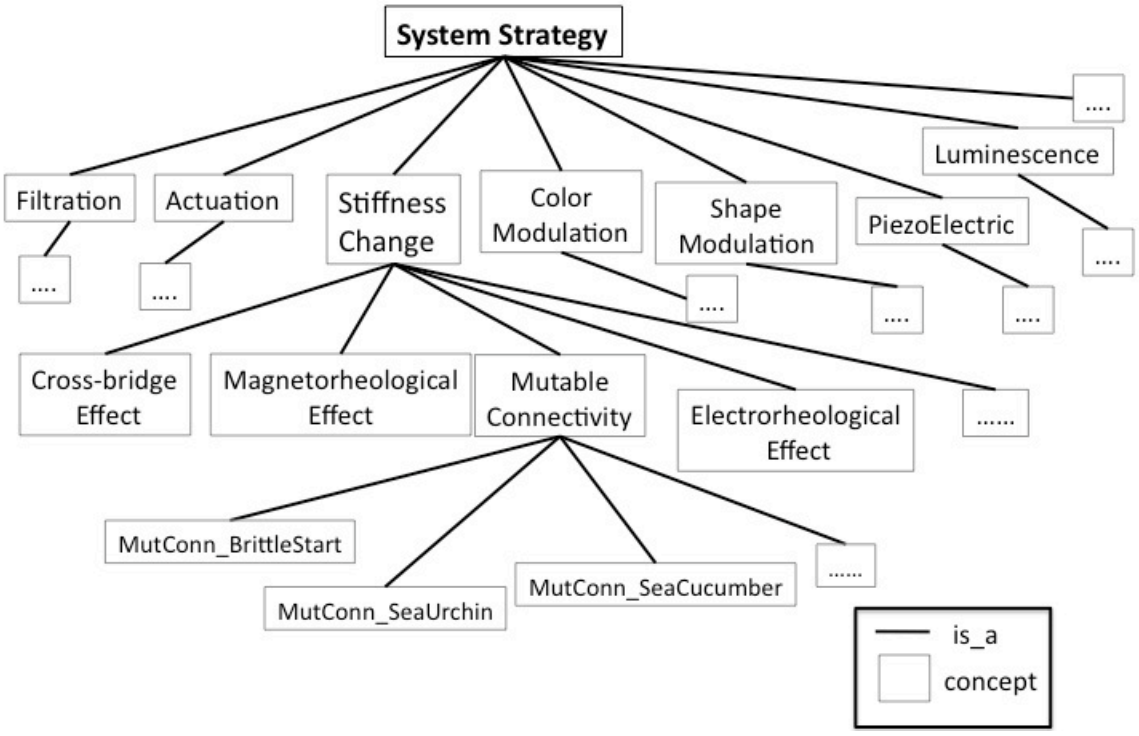


Figure 5.7 Sample of the System Strategy Taxonomy

Structure Taxonomy

The structure taxonomy is composed of the systems in which the strategy is performed. In the hierarchical Petri net representation, a strategy is attached to the top-level system of interest, as opposed to its subsystems. Therefore, the scope of the structure taxonomy only includes representation of this top-level system. For instance, consider the Magnertorheological Effect strategy. This strategy is enacted by Magnetorheological fluids, not by the components of the fluid, such as the carrier fluid and particles. A

structure can have multiple strategies enacted within it (ie. the human muscle (structure) can have a strategy for contraction and for stiffness change). A sample of the structure taxonomy used in this work is displayed in Figure 5.8.

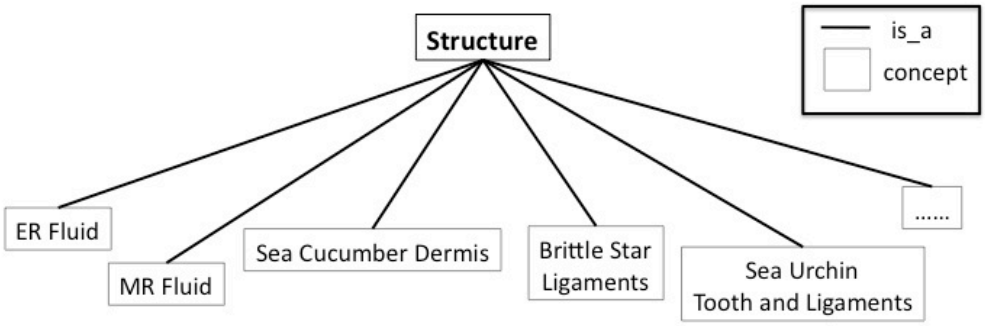


Figure 5.8 Sample of the Structure Taxonomy

Domain taxonomy

The domain taxonomy is used to distinguish the domain from which the system originates. In this research, we focus on two domains, engineering and biological. These domains can also be divided into smaller sub-domains, but this was excluded as part of this work.

5.1.3 Ontology Structuring

In the next step in the development of the ontology, inter-relationships between concepts across different taxonomis are formed. The relationships and their definitions are summarized in Table 5.1.

Table 5.1 Definitions of the Relationships

Relationship	**Concept**	**Filler**	**Definition of the relationship**
satisfiesFunction	System strategy	has_input, has_output	Describes the function that the strategy fulfills
hasInput	System strategy	Flow concept	A nested role of satisfies_function representing the relationship between a system strategy and the driving input of the system
hasOutput	System strategy	Flow conept	A nested role of satisfies_function representing the relationship between a system strategy and the functional output of the system
refinesBehavior	System strategy	has_property, has_action	Describes the behavior that the strategy refines.
hasAttribute	System strategy	Attribute concept	A nested role of has_behavior representing the relationship between a system strategy and the attribute of the system
hasAction	System strategy	Action concept	A nested role of has_function representing the relationship between a system strategy and the action of the system
hasSystem	System strategy	Structure concept	Describes the structure that the strategy is performed in.
hasStrategy	Structure Concept	System Strategy	Inverse relationship of hasSystem. Describes the strategy that the structure uses to fulfill its respective function.
fromDomain	System strategy	System type concept	Describes the domain that the strategy originates.

The last step in developing the ontology is structuring the ontology using the concepts and relationships defined in Table 5.1. The basic structure of the ontology is displayed in Figure 5.9.

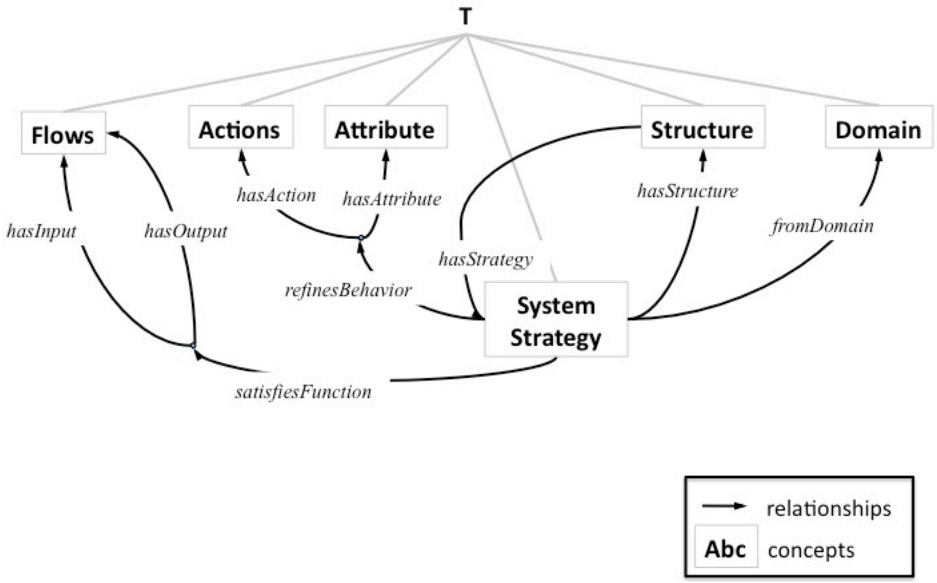


Figure 5.9 Ontology Structure

As seen in the figure, the ontology is structured by relationships between the system strategy and its functional, behavioral, structural, and domain concepts.

5.2 DESCRIPTION LOGICS

Description logics [138] are formalisms used to represent domain-specific concepts and relationships between them. DLs, reviewed in Section 2.3, provide a formal syntax and semantics for describing knowledge within a domain in terms of *concepts* and *properties* that specific *individuals* must satisfy. In this ontology, the concepts in Figure 5.9 are formally defined using Description Logics as follows:

Flows $\sqsubseteq \top$

Actions $\sqsubseteq \top$

Attributes $\sqsubseteq \top$

Domain $\sqsubseteq \top$

Structure $\sqsubseteq \top \sqcap \exists hasStrategy.\text{SystemStrategy}$

SystemStrategy $\sqsubseteq \top \sqcap \exists satisfiesFunction.[\exists hasInput.\text{Flow} \sqcap \exists hasOutput.\text{Flow}] \sqcap \exists refinesBehavior.[\exists hasAction.\text{Action} \sqcap \exists hasAttribute.\text{attribute}] \sqcap \exists hasStructure.\text{Structure} \sqcap \exists fromDomain.\text{Domain}$

The focus of this ontology is the retrieval of relevant strategies. In this work, strategies are extracted using the Method for Reverse Engineering Biological Systems (see Chapter 3). Strategies are viewed as refinements of system behavior, or specific physical phenomena driving a particular behavior. Strategies (and behaviors) are used to satisfy a particular function of a system. Strategies are also utilized by a particular system or structure. A domain specification is used for strategy to distinguish between strategies from the engineering and biological domains. Therefore, as seen above, system strategy is defined as something that (1) satisfies a function, (2) refines a behavior, (3) has a structure, and (4) from a particular domain of application.

The schema of the proposed ontology is illustrated using the example of the mutable connective tissue of the sea cucumber (Section 4.4.1). The strategy extracted using the hierarchical Petri net representation is as follows:

"Stiffness in the dermis is changed by controlling the association of the collagen fibril bundles"

This strategy is short-termed 'mutable connectivity', and listed in the ontology as 'MutConn-SeaCucumber'. Using the relationships defined in Table 5.1, the strategy is represented in the ontology as follows:

MutConn-SeaCucumber ≡ SystemStrategy ⊓ ∃*satisfiesFunction*.[∃*hasInput*.Affinity ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*.SeaCucumberDermis ⊓ ∃*fromDomain*.Biological

This representation describes the strategy 'Mutconn-SeaCucumber' as :

- something that satisfies a function having an input of chemical affinity and an output of force
- something that refines a behavior of 'increase stiffness'
- something performs with the dermis of the Sea Cucumber
- something that is from the biological domain.

5.3 IMPLEMENTATION

5.3.1 Software Implementation

The ontology and its description logic implemention are encoded using an ontology editor software (Protégé), a DL reasoner software (RacerPro), and a DIG interface between them. These components are displayed in Figure 5.10 [139].

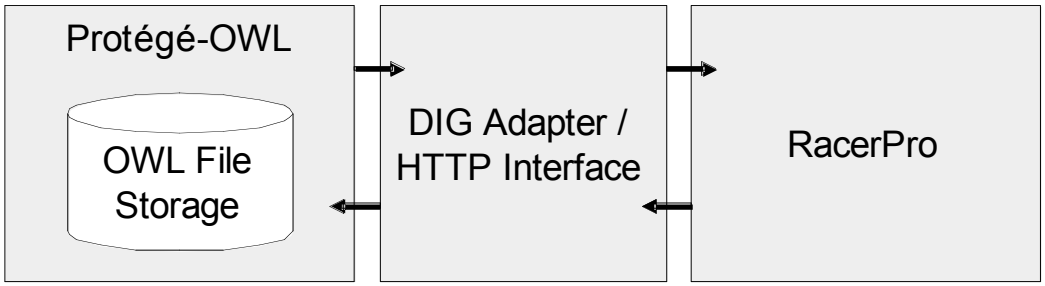


Figure 5.10 Ontology Software Implementation Environments

The individual components and their descriptions, referenced from [75], are as follows:

Protégé-OWL

Protégé-OWL editor is an extension of Protégé that supports developing ontologies using the Web Ontology Language (OWL) [140-142]. Protégé-OWL is an open-source ontology development environment with functionality for editing OWL based ontologies[65].

RacerPro

RacerPro is a knowledge representation system that implements a highly optimized tableau calculus for various DLs. RacerPro is the back-end reasoner used within Protégé-OWL and implements the HTTP interface called DIG for connecting with Protégé-OWL. This reasoner was initially developed at the University of Hamburg, Germany. RacerPro is actively supported and future releases are developed at Concordia University in Montreal, Canada, and at the University of Applied Sciences in Wedel near Hamburg, Germany.

OWL DL File Storage

OWL DL is a standard XML-based language that is used for explicitly representing the meaning of terms in vocabularies and the relationships between those terms. OWL DL provides support for developing ontologies using DLs representations. OWL is a standard ontology language by W3C. OWL is the markup language used to store DL ontologies [66].

DIG Interface

The DIG Interface is a standardized interface based on XML for DLs systems. The DIG interface is developed by the DL Implementation Group (DIG). The DIG interface is an emerging standard for providing access to description-logic reasoning via an HTTP-based interface to a separate reasoning process [143].

Using the above environment, concepts (termed classes in Protégé) and representations (termed roles within Protégé) are implemented, as shown in Figure 5.11 and Figure 5.12, respectively.

Figure 5.11 Concept (class) Taxonomies implemented in Protégé

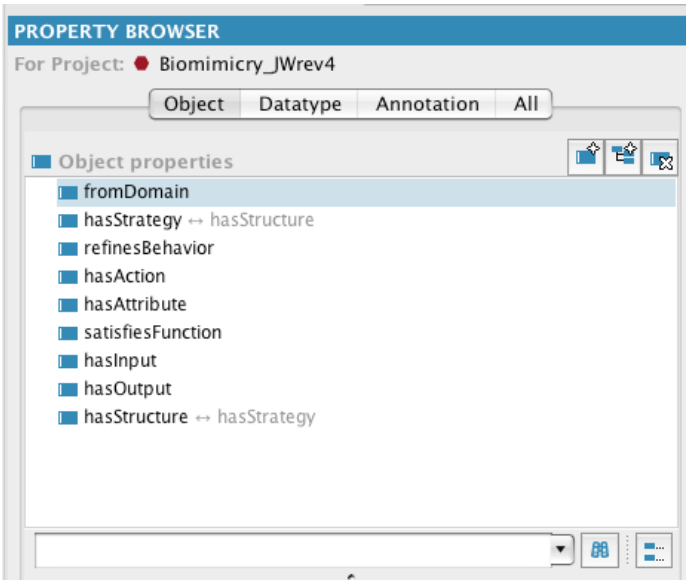


Figure 5.12 Relationship (role) implementation in Protégé

Figure 5.11 displays the basic 'is_a' relationships used to structure the concept taxonomies and Figure 5.12 shows the more complex relationships used to link concepts from different taxonomies. Strategy concepts are linked to other concepts using these inter-relationships. The Protégé condition window for the mutable connectivity example from Section 5.2 is displayed in Figure 5.13.

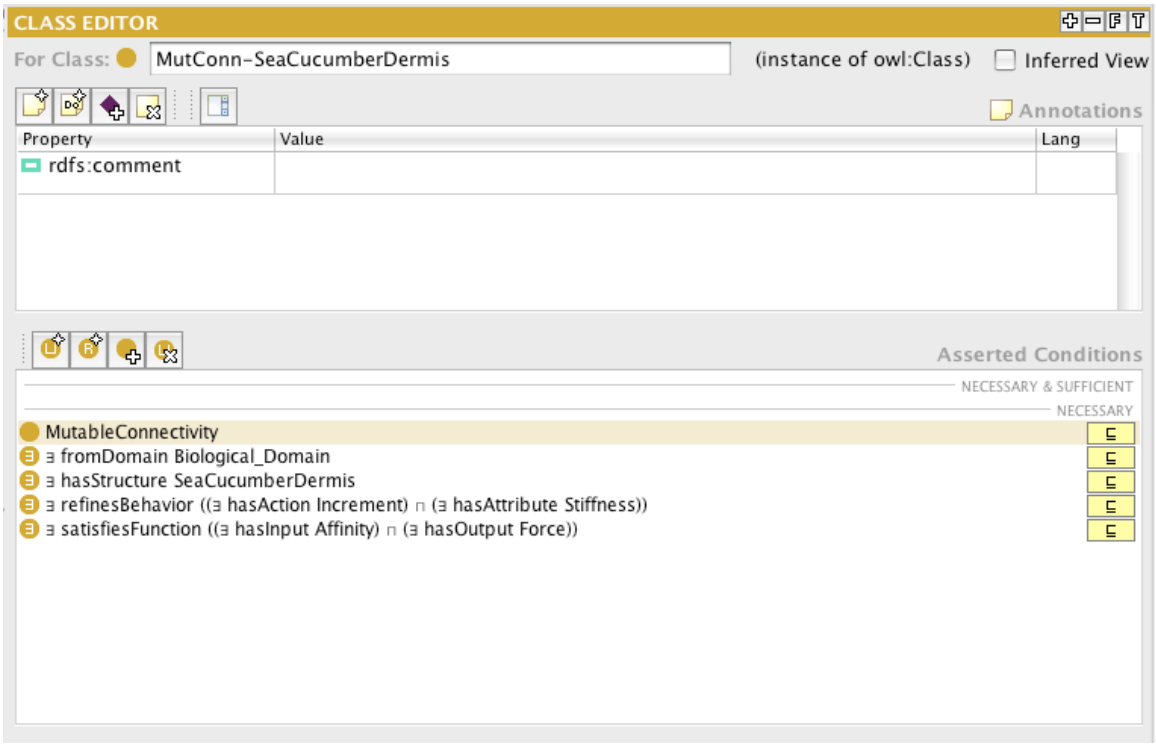


Figure 5.13 Protégé Implementation of Sea Cucumber Dermis

5.3.2 Repository Implementation

The software implementation of the repository was considered in Section 5.3.1. In this section, the different roles that are needed to build the repository are considered. There are two distinct roles needed in building the repository: the information gatherer and repository encoder. The primary role of the information gatherer is to retrieve biological system information from the biological domain. This includes decomposing the biological system into its functional, behavioral, and structural parts (using the

method for Reverse Engineering Biological Systems). Someone familiar with the biological domain should perform this role.

The role of the encoder includes encoding the information from the biological domain into the repository. Specifically, this role includes encoding the functional, behavioral, and structural information into the repository using Description Logics. A simple software interface could be used to reduce the load on the encoder. This interface, such as that found in Section 5.5.2.1, can be used to shield the DL representation process from the repository encoding process. The interface could then allow the information gatherer the tools to encode the biological information into the repository.

5.4 SUBSUMPTION IN DESCRIPTION LOGICS AND RETRIEVAL

In this research, we utilize inference mechanisms from Description Logics to aid in retrieval of relevant strategies from the repository. The two mechanisms of primary interest in this research are subsumption and satisfiability. Satisfiability determines the logical soundness of concept descriptions with respect to a terminology, whereas subsumption tests whether a concept or a role is a more general expression of another role [85]. As described in Section 2.3, using the tableau methods, subsumption can be reduced to the satisfiability of concept descriptions expressed as follows:

$$C \sqsubseteq D \text{ iff } C \sqcap \neg D \rightarrow \varnothing \qquad \text{Equation 5.1}$$

In other words, C is subsumed by D if and only if the intersection of C and the negation of D is null.

In this research, two types of subsumption hierarchies are identified. A type I subsumption is formed by the modification of expressions without using the taxonomic structure of the defined vocabularies. A type II subsumption is formed using the taxonomic structure of the vocabularies. In a simplified example, let us define three general concepts as follows:

- Concept A $\equiv \exists$ *satisfiesFunction*.[$\exists$*hasInput*.Current $\sqcap$ $\exists$*hasOutput*.Force]

- Concept B ≡ ∃ *satisfiesFunction*.[∃*hasInput*.Electrical_Energy ⊓ ∃*hasOutput*.Mechanical Energy]
- Concept C ≡ ∃*satisfiesFunction*.[∃*hasInput*.Current ⊓ ∃*hasOutput*.Force] ⊓ ∃*fromDomain*.Engineering

From this example, Concept A ⊑ Concept B by type II subsumption. Type II subsumption is formed because Electrical_Energy and Mechanical Energy subsume Current and Force, respectively, in the Flow taxonomy (defined in Section 5.1). With respect to querying the repository, this means that a strategy satisfying a function with an input of current and output of force (Concept A) will also satisfy a more general query requesting a strategy for a function with an input of electrical energy and output of mechanical energy (Concept B). Also, due to the addition of the Engineering domain restriction in Concept C, Concept C ⊑ Concept A by type I subsumption. In this case, a strategy from the engineering domain satisfying a function with an input of current and output of force (Strategy C) will satisfy the more general function without the domain restriction (Straegy A).

5.5 CONSISTENCY AND PRECISION IN RETRIEVAL

In this research, subsumption is used to retrieve relevant strategies from the strategy repository. For retrieval through subsumption, query nodes are utilized. These queries into the repository are viewed as concept descriptions with necessary and sufficient conditions for objects to satisfy. Therefore, query nodes are entered into the repository as a search mechanism. Through subsumption, only the relevant strategies that satisfy the query should be returned. Specifically, it is hypothesized that

Hypothesis 2: Subsumption in Description Logics will enable consistent and precise retrieval of relevant biological strategies from a knowledge repository.

The two key claims in this hypothesis are that of (1) consistency and (2) precision of subsumption in DLs. Consistency in retrieval is important because it shows that the order in which we build the repository, as well as the order in which we query the repository, have no impact on the retrieval process. Precision is important in retrieval as it shows that all the results returned will be directly relevant to the query. These claims will be evaluated in the following sections.

5.5.1 Consistency in retrieval

Yim [75]and Udoyen [85] have shown subsumption in DL to produce unique and consistent hierarchies. Hierarchies created using subsumption in DL can be shown to be consistent and correct for acyclic terminologies by proving that subsumption in DL imposes an order relation, or partial order on entities when the subsumption relation is computed. Wille and Ganter [144] list conditions for asserting that a binary relation R on a set M is a partial order relation. They state that for all elements $x, y, z \in M$,

- the relation is reflexive, i.e., xRx
- the relation is antisymmetric, i.e., xRy and $x \neq y \Rightarrow$ not yRx
- the relation is transitive, i.e., xRy and $yRz \Rightarrow xRz$

These conditions can be shown to hold for subsumption by evaluating the condition for logical subsumption, which is a binary relation. The three conditions can be shown to hold by asserting their truth value when the condition for logical subsumption expressed in Equation 5.1 is true. Equation 5.1 represents the tableau algorithm of subsumption in DL. Udoyen presents the mathematical proof for showing that Equation 5.1 satisfies the above three conditions[145]. This proof is generic such that it is applicable to subsumption for any description logics. Subsumption in DL has been shown to be consistent and correct, therefore retrieval through subsumption should also be consistent and correct.

5.5.2 Precision in retrieval

Now that subsumption in DL has been shown to be consistent, we now empirically evaluate the precision in retrieval of biological and engineering strategies through subsumption. Precision is defined [85] as follows:

$$\text{Precision} = \frac{\text{Number of retrieved relevant documents}}{\text{Number of retrieved documents}} \qquad \text{Equation 5.2}$$

As seen in Equation 5.2, a precision of 1 means that 100 percent of the documents retrieved were relevant, and thus the search was fully precise. A precision score of less than 1 means that some (or many) irrelevant results were returned and some secondary filtering process is needed to identify the relevant results.

5.5.2.1 Test-bed development

A testbed was developed to aid in querying the ontology and receiving the retrieval results. The repository testbed can be divided into three major components: the user interface, the reasoner, and the processing modules. A graphical illustration of this testbed is displayed in Figure 5.14.

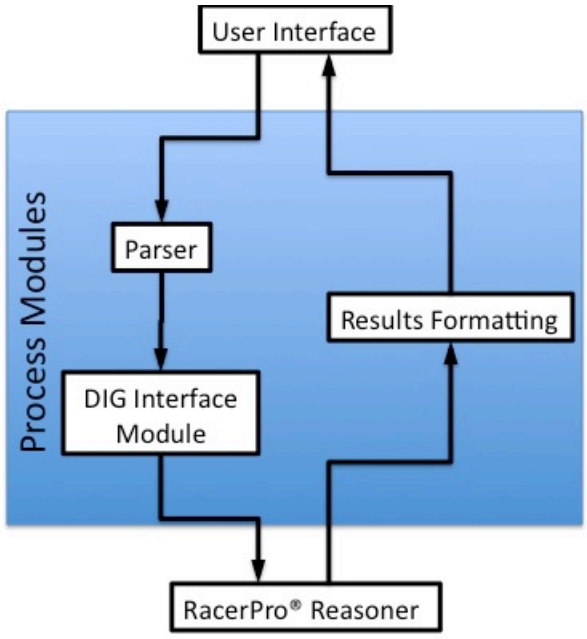


Figure 5.14 Illustration of the Repository Testbed

User Interface

Communication with repository begins with the user interface. The user interface is the only component that directly interacts with the user. The interface has two major functions: (1) to take information from the user and pass it to the process modules and (2) to display search results. No major computations or calculations occur within the user interface—it simply facilitates the searching of the repository.

Process Modules

After the user interface has collected the data, the information is passed to the processing modules. The processing modules can be viewed as 'translators"—the processes that convert information into languages recognizable by the user interface or the reasoner. First, the information is parsed into specific categories, such as "Input" and "Verb," using a parser. After the categories are parsed, they are then encoded into information modules using the DIG module. The DIG module utilizes the DIG interface to encode these modules. The DIG interface is a standardized XML (extensible markup language) developed by the Description Logic Implementation group specifically to

interface with description logic systems. Once the information has been processed using the DIG interface, the information modules will be ready to interface with the reasoner. In addition to these modules, another process module is used singularly to convert query results from the reasoner to a form that is easily understandable by the user.

Reasoner

The third and final component, the reasoner, functions as the repository, storing and retrieving all biological and engineering design models. The actual information models are implemented in a specialized language, OWL, (web ontology language) by using the program, Protégé (see Section 5.3). These models are then converted into the DIG interface in Protégé and are subsequently stored in the reasoner. For the purposes of this research, the open source reasoner, RacerPro, has been utilized (see Section 5.3). The information modules created by the process modules component are used to query the reasoner. The reasoner searches through the design hierarchy and returns the relevant search results. These results are passed back to the processing modules, converted, and eventually back to the user interface. The user interface and process modules were developed using Microsoft Visual C# 2008 Express Edition, XML DOM (Document Object Model), and XSLT (Extensible Stylesheet Language).

The user interface is displayed in Figure 5.15, Figure 5.16, and Figure 5.17. The code for the user interface is presented in Appendix A.

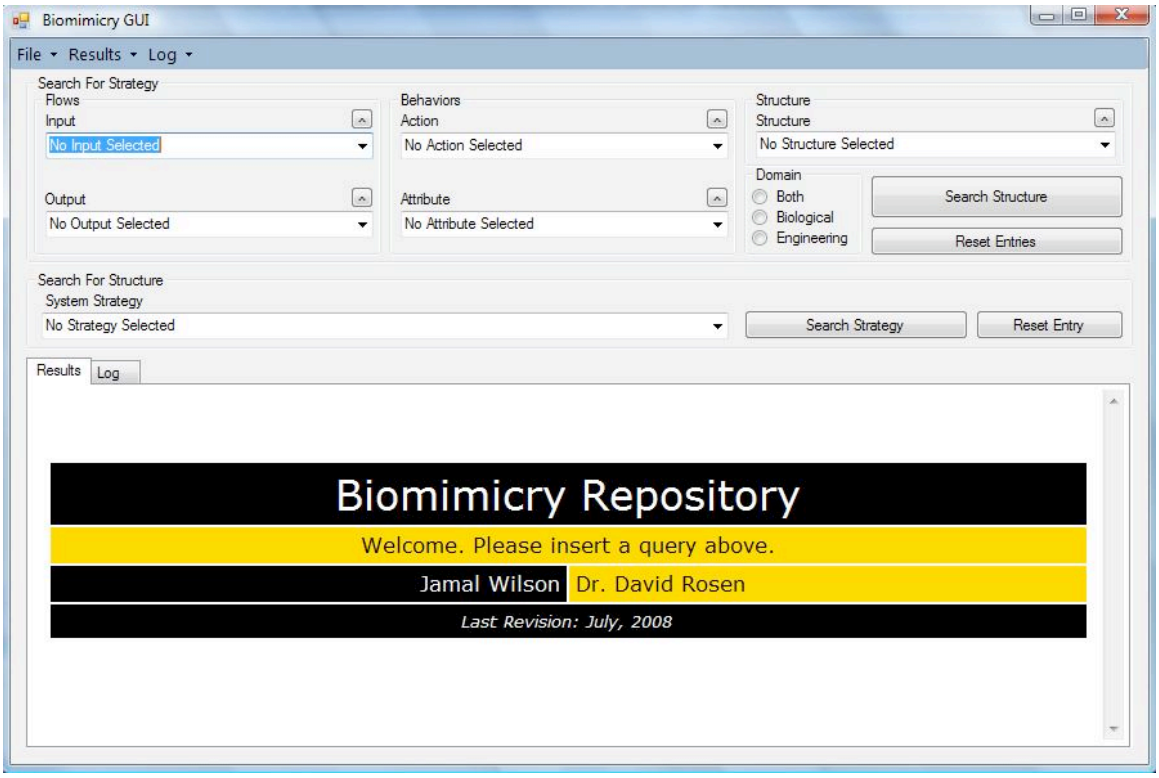


Figure 5.15 User Interface

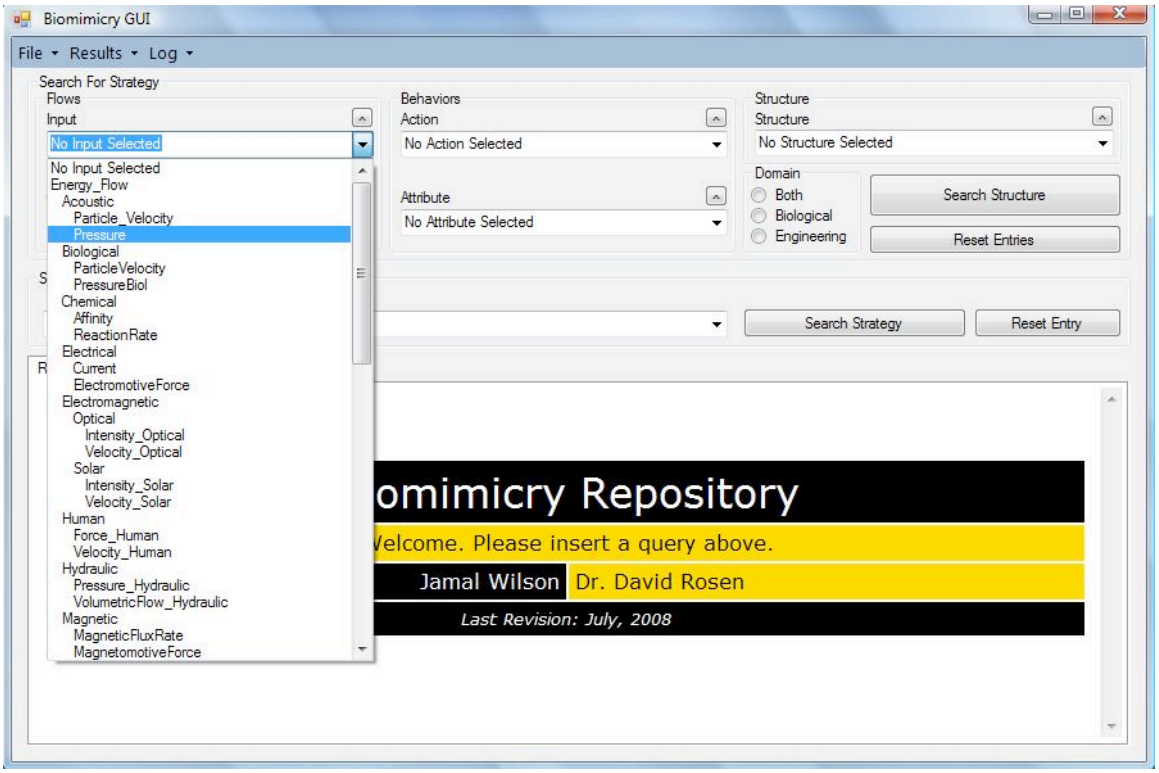


Figure 5.16 User Interface - Drop-down menu

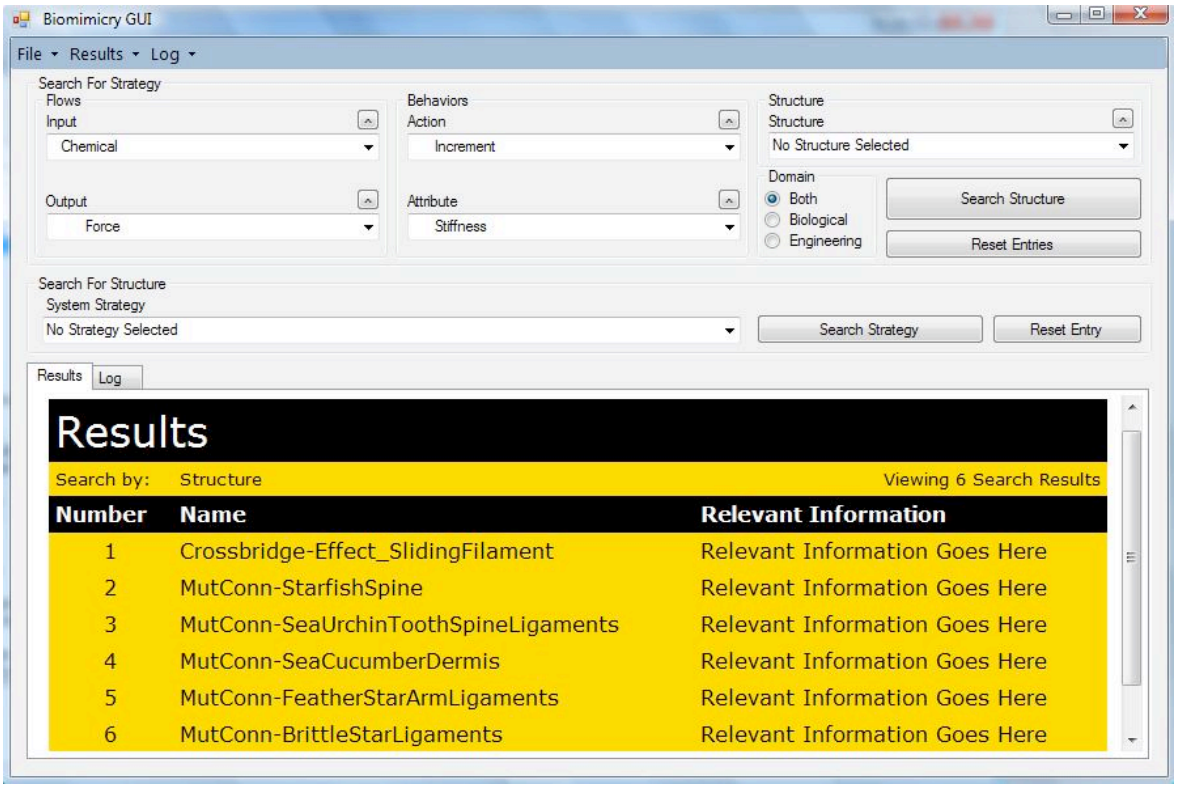


Figure 5.17 Formatted results of the User Interface

In Figure 5.15, the user interface for querying the repository is displayed. The interface includes inputs for function, behavior, structure, domain, and strategy. The user inputs a function-based strategy search using 'Input' flow and 'Output' flow menu boxes. Flow concepts are hierarchically arranged in drop-down menus for user selection (see Figure 5.16). A behavior-based strategy search is input using the verb-noun pairing of 'Action' and 'Attribute' concepts. These concepts are also arranged using drop-down menus. Users can further refine the function and behavior-based searches using 'Structure' and 'Domain' concepts. A strategy-based structure search can also be used to query the repository using the "Strategy" drop-down menu. This will retrieve the strategies related to a specific structure. Once all the desired criteria are entered into the user interface, the query is formed using the 'Search' button. The retrieval results are then formatted and returned to the user, as displayed in Figure 5.17. In the results tab, the name of the retrieved concepts is displayed. In the future iteration of this user interface,

other relevant information, such as the extended strategy description, will also be displayed.

One of the advantages of ontologies is concept abstraction, where a query can be further abstracted up the hierarchy to give a broader range of results. Arrow buttons have been included in the User Interface beside each input to allow the results from the parent of an existing concept to be retrieved.

The top menu bar of the User Interface has 3 tabs: 'File', 'Results', and'Log'. The 'File' tab allows the user the close the application. The results from the query can be saved to a text file or launched in a web browser using the 'Results' tab. The 'Log' tabs allows the user to clear the log.

In this section, the repository testbed was presented. In the next section, this repository is populated with biological and engineering strategies.

5.5.2.2 Repository population

To test the hypothesis, a set of 38 biological and engineering systems was selected and encoded into the repository. Nine of these systems with variable-stiffness properties are described below; the remaining systems are displayed in Appendix B:

- **Human muscle in isometric contraction** (Human_Muscle-IsomContraction) with a strategy of "Stiffness in the Muscle Fiber is controlled by controlling the bridging of the actin and myosin filaments of Myosin" (Crossbridge effect)
- **Electrorheological fluids** (ER_Fluid) with a strategy of "Stiffness in the ER Fluid is changed by controlling the alignment of dielectric particles with an electrical field" (Electrorheological Effect)
- **Shear-thickening fluids** (Shear-Thickening_Fluid) with a strategy of "Stiffness in the Shear-thickening fluids is changed by the hydro-clustering of particles in a carrier fluid as a result of high velocity force" (Hydro-clustering)
- **Magnetorheological fluids** (MR_Fluid) with a strategy of "Stiffness in the MR fluid is changed by controlling the bonding between magnetic particles in a carrier fluid with a magnetic field" (Magnetorheological_Effect)

- **Dermis of the Sea Cucumber** (SeaCucumberDermis) with a strategy of "Stiffness in the dermis is changed by controlling the association of the collagen fibril bundles" (MC-SeaCucumber) for mutable connectivity.
- **Invertebral Ligaments of the Brittle Star** (BrittleStar-InvertebralLigaments) with a strategy of "Stiffness in the invertebral ligaments is changed by controlling the association of the collagen fibril bundles" (MC-BrittleStar)
- **Arm Ligaments of the Feather Star** (FeatherStarArmLigaments) with strategy of "Stiffness in the arm ligaments is changed by controlling the association of the collagen fibril bundles" (MC-FeatherStar)
- **Tooth and spine ligaments of the Sea Urchin** (SeaUrchinToothSpineLigaments) with a strategy of "Stiffness in the tooth and spine ligaments is changed by controlling the association of the collagen fibril bundles" (MC-Sea Urchin)
- **Spine of the Starfish** (StarfishSpine) with strategy of "Stiffness in the starfish spine is changed by controlling the association of the collagen fibril bundles" (MC-Starfish)

Next, these strategies were described using Description Logics. The DL descriptions of the above strategies are displayed in Table 5.2**.**

Table 5.2 Description Logic descriptions

Strategy	DL Description
Crossbridge-Effect_SlidingFilament	∃ *satisfiesFunction*.[∃*hasInput*.Affinity ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*.Human_Musle-IsomContraction ⊓ ∃*fromDomain*.Biological_Domain
Electrorheological_Effect	∃ *satisfiesFunction*.[∃*hasInput*.Current ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*.ER_Fluid ⊓ ∃*fromDomain*.Engineering_Domain
Hydro_clustering	∃ *satisfiesFunction*.[∃*hasInput*.Force ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*.Shear-Thickening_Fluid ⊓ ∃*fromDomain*.Engineering_Domain
Magnetorheological_Effect	∃ *satisfiesFunction*.[∃*hasInput*.MagneticFluxRate ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*.MR_Fluid ⊓ ∃*fromDomain*.Engineering_Domain

Table 5.2 Continued

MutConn-BrittleStarLigaments	∃ *satisfiesFunction*.[∃*hasInput*.Affinity ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*. BrittleStar-InvertebralLigaments ⊓ ∃*fromDomain*.Biological_Domain
MutConn-FeatherStarArmLigaments	∃ *satisfiesFunction*.[∃*hasInput*.Affinity ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*. FeatherStarArmLigaments ⊓ ∃*fromDomain*.Biological_Domain
MutConn-SeaCucumberDermis	∃ *satisfiesFunction*.[∃*hasInput*.Affinity ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*.SeaCucumberDermis ⊓ ∃*fromDomain*.Biological_Domain
MutConn-SeaUrchinTooth	∃ *satisfiesFunction*.[∃*hasInput*.Affinity ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*. SeaUrchinToothSpineLigaments ⊓ ∃*fromDomain*.Biological_Domain
MutConn-StarfishSpine	∃ *satisfiesFunction*.[∃*hasInput*.Affinity ⊓ ∃*hasOutput*.Force] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*hasStructure*. StarfishSpine ⊓ ∃*fromDomain*.Biological_Domain

In Table 5.2, the strategies MutConn-BrittleStarLigaments, MutConn-FeatherStarArmLigaments, MutConn-FeatherStarArmLigaments, MutConn-SeaCucumberDermis, and MutConn-SeaUrchinTooth share a common parent strategy of MutuableConnectivity but have different structures. All the concepts listed in Table 5.2 share a common parent of "Stiffness Change", where all the strategies listed are strategies for controlling stiffness.

5.5.3 Strategy Retrieval

In Section 5.5.2, the testbed repository was presented. In this section, retrieval of strategies from the repository is discussed. Specifically, in Section 5.5.3.1, different types of queries used in engineering design are discussed how to structure these queries to retrieve strategies. In Section 5.5.3.2, precision of the retrieval strategy is test empirically.

5.5.3.1 Querying the Repository

In the proposed repository, designers use queries to describe the intent of their search in a manner that allows it to be compared to other concepts in the ontology. Retrieval is based on comparing the prescriptive function convention of the function structure to the descriptive function and behavior representation utilized in the repository.

In the search for solutions, there are several types of relevant queries that are envisioned in this research: (1) search for a strategy to satisfy a function using input/output flows of energy, material, and signal (2) search for a strategy that refines a particular behavior (function) using a verb-noun pairing (3) search for the strategies that a particular system or structure uses to satisfy a function and/or behavior and (4) search for strategies from a particular domain that satisfy a function and/or behavior.

Verb-noun pairings of function are considered purpose functions and input/output flow pairings are considered action functions [56]. In defining function, the intention, or purpose, of the device or component is expressed using a verb-noun pairing. Action functions, on the other hand, are defined at much lower levels of design specification.

At higher levels of the design specification, when only the intention, or purpose, of the device is known, a behavior-based (verb-noun) search is advantageous. This allows comparison of the intended function of the to-be-designed device to the actual behavior that a strategy satisfies. Using DL, this query is structured as $\exists$ *refinesBehavior*.[$\exists$*hasAction.*Action $\sqcap$ $\exists$*hasAttribute*.attribute], where the verb is termed "action" and the noun is termed "attribute" in the representation. For instance, consider a designer that is searching for a strategy that can "change the color of an artifact". The query will be structured as $\exists$ *refinesBehavior*.[$\exists$*hasAction.*Change $\sqcap$ $\exists$*hasAttribute*.Color], where "change" is the action (verb) and "color" is the attribute (noun) of interest. In this case, the designer is searching for a strategy for achieving the intended function of changing color, such as "variable refraction". Since strategies are defined in terms of the behaviors that they refine (or intended functions that they satisfy), the query node will subsume all strategies that refine a behavior of "change color".

At more refined levels of design specification, when input/output flows are known, a function-based input/output search is useful. This search allows comparison of the flows of the intended device to that of the function that the strategy satisfies. Using

DL, this query is structured as ∃*satisfiesFunction*.[∃*hasInput*.Flow ⊓ ∃*hasOutput*.Flow]. For instance, consider a designer that is searching for "a strategy that produces (outputs) color based on an input of mechanical force". A query will be structured as ∃ *satisfiesFunction*.[∃*hasInput*.Force ⊓ ∃*hasOutput*.Color]. This search is often used to refine the verb-noun search, giving a much more specific set of strategies. For instance, consider a designer searching for " a strategy the changes color using an input of mechanical force". Using DL, a query can be structured as ∃ *refinesBehavior*.[∃*hasAction*.Change ⊓ ∃*hasAttribute*.Color] ⊓ ∃ *satisfiesFunction*.[∃*hasInput*.Force ⊓ ∃*hasOutput*.Color].

Designers can also search for the strategies that a particular system uses to satisfy a function or behavior. For instance, the designer is searching for the strategy that human muscle uses to generate force. This type of search is useful when browsing for novel strategies to mimic or in narrowing the search field of possible strategies. Using DL, the query is structured as ∃*hasStructure*.Structure.

Using the domain concept, designers can specify the domain of application in which the strategy is applied. This is particularly useful when designers only want to retrieve novel biological strategies or in the case where designers are looking for ready-made strategies and solutions from the engineering domain. In this case, the designer would add the domain concept as ∃*fromDomain*.Domain to an existing search.

To test the precision of retrieval through subsumption, several queries were developed to represent different scenarios upon which the repository may be searched. The queries used for this study are displayed in

Table 5.3.

Table 5.3 displays the query description in natural and DL language, as well as strategies considered as relevant through comparison.

Table 5.3 Query Concepts

Query	**Description (Natural Language)**	**Description (DL Representation)**	**Relevant**
Q1	A strategy that allows one to increment stiffness of a system	∃ *refinesBehavior*. [∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness]	Magnetorheological effect, Electrorheological effect, Crossbridge effect, Hydro-clustering, MC-(All)
Q2	A strategy that allows one to change the stiffness of a system	∃ *refinesBehavior*.[∃*hasAction*.Change ⊓ ∃*hasAttribute*.Stiffness]	Magnetorheological effect, Electrorheological effect, Crossbridge effect, Hydro-clustering, MC-(All)
Q3	An engineering strategy that allows one to change the stiffness of a system	∃ *refinesBehavior*.[∃*hasAction*.Change ⊓ ∃*hasAttribute*.Stiffness] ⊓ ∃*fromDomain*.Engineering_Domain	Magnetorheological effect, Electrorheological effect, Hydro-clustering
Q4	A strategy that transforms chemical energy into mechanical energy	∃ *satisfiesFunction*.[∃*hasInput*.Chemical ⊓ ∃*hasOutput*.Mechanical]	Crossbridge effect, MC-(All)
Q5	Any strategy that produces force	∃ *satisfiesFunction*.[∃*hasInput*.Flows ⊓ ∃*hasOutput*.Force]	Magnetorheological effect, Electrorheological effect, Crossbridge effect, Hydro-clustering, MC-(All)
Q6	A strategy that allows a system to change shape	∃ *refinesBehavior*.[∃*hasAction*.Change ⊓ ∃*hasAttribute*.Shape-Physical]	None
Q7	A strategy that increments the stiffness of the system and has a chemical affinity input and force output	∃ *satisfiesFunction*.[∃*hasInput*.Affinity ⊓ ∃*hasOutput*.Force]] ⊓ ∃ *refinesBehavior*.[∃*hasAction*.Increment ⊓ ∃*hasAttribute*.Stiffness]	Crossbridge effect, MC-(All)
Q8	The strategy that the human muscle in isometric contraction	∃*hasStructure*.Human_Musle-IsomContraction	Crossbridge effect

Using the testbed repository presented in Section 5.5.2, the queries in

Table 5.3 were processed. These results are presented in the following section.

5.5.3.2 Retrieval results

Protégé represents subsumption through a concept hierarchy. In the hierarchy, subsumption is represented using arrows, with the tail representing the subsumed concept and the head represented the subsumer concept. It should also be noted that subsumption extends to all the children of the subsumed concept. Using the queries above for retrieval, the following subsumption hierarchy, displayed in Figure 5.18, was computed.

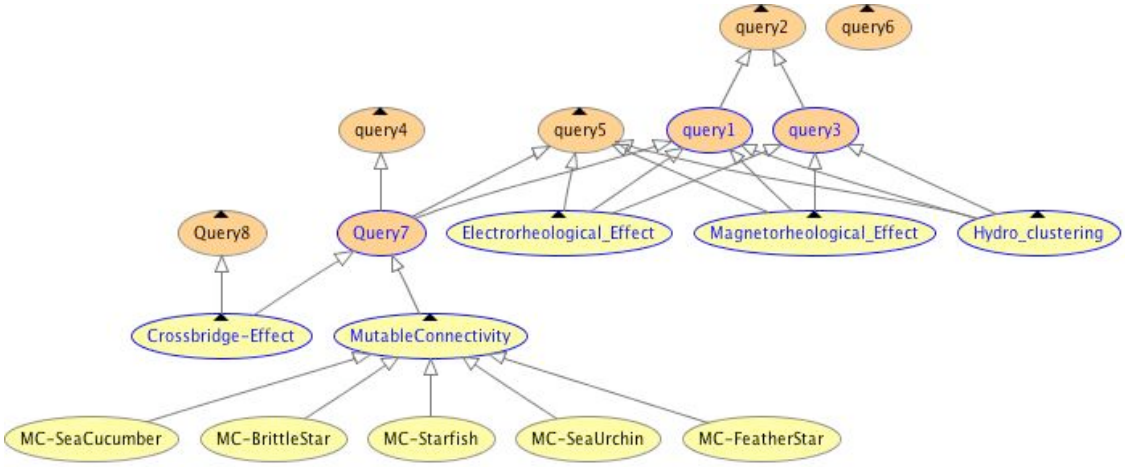


Figure 5.18 Subsumption hierarchy

The results from the test-bed for the individual queries are described below:

Query 1: Query 1 seeks to find a strategy that allows one to increment stiffness in a system. Through subsumption, all the strategies are retrieved. Since all the retrieved strategies refine a behavior that increments stiffness, the precision is calculated as 1.

Query 2: Query 2 seeks a strategy that allows one to change the stiffness of a system. Through subsumption, all the strategies are retrieved. Since the action 'change' subsumes 'increment' in the action taxonomy, the query retrieves all the strategies all the strategies that refine a behavior that increments stiffness. Precision is also calculated as 1 in this case.

Query 3: Query 3 seeks an engineering strategy that allows one to change the stiffness of a system. Query 3 subsumes strategies Magnetorheological fluid, Electrorheological fluid, and Hydro-clustering. Since all the concepts from the engineering domain that increment stiffness are retrieved, precision is calculated as 1.

Query 4: Query 4 seeks a strategy that transforms chemical energy into mechanical energy. Query 4 subsumes strategies Crossbridge effect and Mutable Connectivity. These strategies have an input of chemical affinity (a type of Chemical energy) and an output of force (a type of Mechanical energy), thus the precision is calculated as 1.

Query 5: Query 5 seeks any strategy that produces force. Since all the strategies have an output of force, all the strategies are retrieved. Precision is calculated as 1.

Query 6: Query 6 seeks a strategy that allows a system to change shape. None of the strategies refine a strategy that refines shape, therefore none are subsumed by Query 6.

Query 7: Query 7 seeks a strategy that increments the stiffness of the system and has a chemical affinity input and force output. Query 7 subsumes strategies Crossbridge effect and Mutable Connectivity, which both have inputs of affinity and outputs of force and behaviors that increment stiffness. In query 7, precision is calculated as 1.

Query 8: Query 8 seeks the strategy of the human muscle in isometric contraction. The query retrieves the crossbridge effect, which is indeed the strategy of the human muscle. Therefore, the precision of this query is 1.

In this section, the precision of subsumption in DL was empirically tested. Using the test queries on the testbed repository, subsumption-based retrieval was found to be precise. In every case except for when no relevant results were found (Query 6), the precision was found to be 1.

5.5.4 Comparison with current approaches

In this section, the performance of the strategy repository is compared with the performance of the Biomimicry Database [21] and the Functional Keyword Search [22, 23, 29] (reviewed in Section 1.2.2). In Section 5.5.3, the precision of the subsumption-based retrieval strategy utilized in the Strategy Repository was empirically tested. However, in order to get a full picture of the performance of a retrieval strategy, retrieval recall must also be considered. Recall is defined using Equation 5.3 [85].

$$\text{Recall} = \frac{\text{Number of retrieved relevant documents}}{\text{Number of relevant documents}} \qquad \text{Equation 5.3}$$

In this comparison, retrieval performance is defined in terms of a retrieval effectiveness score, or an F_1 score [146]. This F_1 score, defined by Equation 5.4, takes into account retrieval precision and recall in a single metric.

$$F_1 \text{ score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \qquad \text{Equation 5.4}$$

Based on Equation 5.4, an F_1 score of 1 means that all retrieved results are relevant and all the relevant results were retrieved. A low score means that a large number of irrelevant results were retrieved and/or only a small number of the relevant results was retrieved.

With respect to the Strategy Repository, using the results from Section 5.5.3.2, the F_1 was found to be 1 for all queries except *Query 6* (when no results were retrieved). This is because the subsumption algorithm in DL not only ensures that all the strategies retrieved are relevant (Section 5.5.3.2), it also assures that all the relevant results are retrieved. Therefore, subsumption in DL ensures both high precision and recall in retrieval of biological strategies.

Biomimicry Database

The Biomimicry Database utilizes a searchable database of biological information to identify biological analogs. In this study, the effectiveness of the database is tested by running test queries into the database and computing the precision, recall, and F_1 score for the results. First, the queries were formulated and the relevant results were determined by browsing the repository. Next, the queries were performed and the performance metrics calculated. The queries, relevant results that should be retrieved for these queries, the actual retrieved results, and the performance metrics are displayed in Table 5.4.

Table 5.4 Query Results for the Biomimicry Database

	Query	***Relevant Results***	***Retrieved Results***	***Precision***	***Recall***	***F_1***
Q1	Create color	• Diffraction • Photonic Crystals • Pigments • Structural Color • Thin-film Interference • Emission of Colored Light • Scattering	• Diffraction • Photonic Crystals • Pigments • Structural Color • Thin-film Interference	1	0.71	0.83
Q2	Color creation	• Diffraction • Photonic Crystals • Pigments • Structural Color • Thin-film Interference • Emission of Colored Light • Scattering	• Diffraction • Scattering • Thin-film Interference	1	0.42	0.60

Table 5.4 Continued

	Query	***Relevant Results***	***Retrieved Results***	***Precision***	***Recall***	F_1
Q3	Change temperature	• Color Change for Thermoregulation • Conventional Air Heating/Cooling Direct Evaporative Cooling • Emissivity for Temperature Regulation • Evaporative Cooling • Varying Insulation • Varying Posture • Regional Heterothermy	• Direct Evaporative Cooling	1	0.13	0.22
Q4	Regulate temperature	• Color Change for Thermoregulation • Conventional Air Heating/Cooling • Countercurrent Heat Exchange • Emissivity for Temperature Regulation • Evaporative Cooling • Fur Insulation • Gular Flutter • Varying Insulation • Varying Posture • Regional Heterothermy	• Varying Posture	1	0.10	0.18
Q5	Generate heat	• Non-shivering Thermogenesis (NST) • Brown Adipose Tissue	• Brown Adipose Tissue	1	0.50	0.67

In the results found in Table 5.4, the retrieval effectiveness was found to be less than one in every case, with query effectiveness going as low as 0.18 in the case of Q4.

The low retrieval performance is a product of the keyword-based search strategy utilized in the database. In the retrieval strategy used, an exact match of terms is needed for retrieval. Although the exact keyword match aids in increasing precision, the tightened criteria for exact matching excludes many relevant results. This results in low recall scores, thus lowering the overall effectiveness of the retrieval strategy. For instance, consider Queries 1 and 2, where a strategy was sought for color creation and queries were structured as both "create color" and "color creation". As can be seen by the results, Q1 retrieved more relevant strategies than Q2, even though the only difference between the searches is the verb tense. The lower effectiveness scores means that relevant and potentially useful strategies are not identified. This can be contrasted with the subsumption-based retrieval strategy, where a classification system of the keywords is utilized and where exact keyword matches are not needed for retrieval. The classification based matching and the rigorous subsumption algorithm used in the Strategy Repository results in higher effectiveness scores, and thus better performance, when compared to the keyword-based system used in the Biomimicry Database.

Functional Keyword Search

For comparison of the performance of the Functional Keyword Search to that of the Strategy Repository, results published in literature were used. In [147], Stroble and co-authors demonstrated the use of an organized verb-noun search through biological literature for a smart flooring example. The text was searched using the verb "detect" and a series collocated nouns. A sample of the retrieved results is displayed in Figure 5.19. The relevant results are denoted by bold text.

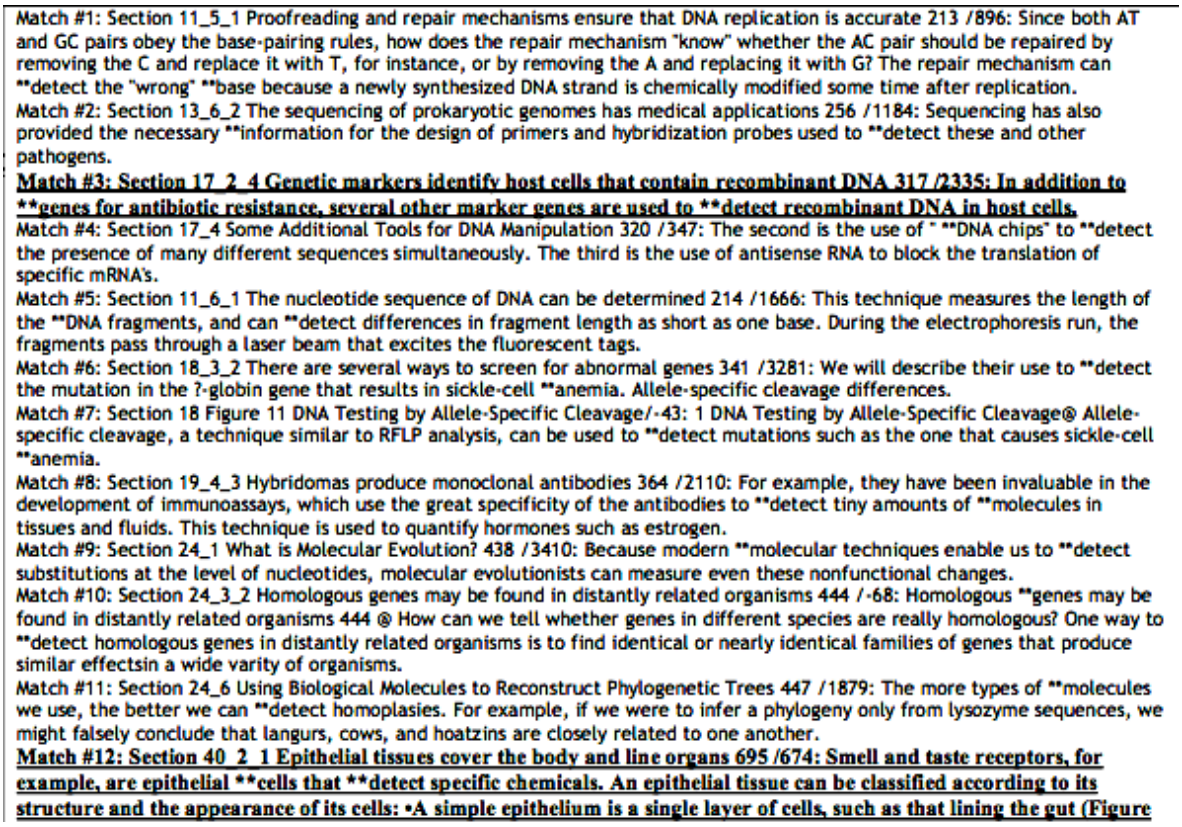

Match #1: Section 11_5_1 Proofreading and repair mechanisms ensure that DNA replication is accurate 213 /896: Since both AT and GC pairs obey the base-pairing rules, how does the repair mechanism "know" whether the AC pair should be repaired by removing the C and replace it with T, for instance, or by removing the A and replacing it with G? The repair mechanism can **detect the "wrong" **base because a newly synthesized DNA strand is chemically modified some time after replication.
Match #2: Section 13_6_2 The sequencing of prokaryotic genomes has medical applications 256 /1184: Sequencing has also provided the necessary **information for the design of primers and hybridization probes used to **detect these and other pathogens.
Match #3: Section 17_2_4 Genetic markers identify host cells that contain recombinant DNA 317 /2335: In addition to **genes for antibiotic resistance, several other marker genes are used to **detect recombinant DNA in host cells.
Match #4: Section 17_4 Some Additional Tools for DNA Manipulation 320 /347: The second is the use of " **DNA chips" to **detect the presence of many different sequences simultaneously. The third is the use of antisense RNA to block the translation of specific mRNA's.
Match #5: Section 11_6_1 The nucleotide sequence of DNA can be determined 214 /1666: This technique measures the length of the **DNA fragments, and can **detect differences in fragment length as short as one base. During the electrophoresis run, the fragments pass through a laser beam that excites the fluorescent tags.
Match #6: Section 18_3_2 There are several ways to screen for abnormal genes 341 /3281: We will describe their use to **detect the mutation in the ?-globin gene that results in sickle-cell **anemia. Allele-specific cleavage differences.
Match #7: Section 18 Figure 11 DNA Testing by Allele-Specific Cleavage/-43: 1 DNA Testing by Allele-Specific Cleavage® Allele-specific cleavage, a technique similar to RFLP analysis, can be used to **detect mutations such as the one that causes sickle-cell **anemia.
Match #8: Section 19_4_3 Hybridomas produce monoclonal antibodies 364 /2110: For example, they have been invaluable in the development of immunoassays, which use the great specificity of the antibodies to **detect tiny amounts of **molecules in tissues and fluids. This technique is used to quantify hormones such as estrogen.
Match #9: Section 24_1 What is Molecular Evolution? 438 /3410: Because modern **molecular techniques enable us to **detect substitutions at the level of nucleotides, molecular evolutionists can measure even these nonfunctional changes.
Match #10: Section 24_3_2 Homologous genes may be found in distantly related organisms 444 /-68: Homologous **genes may be found in distantly related organisms 444 ® How can we tell whether genes in different species are really homologous? One way to **detect homologous genes in distantly related organisms is to find identical or nearly identical families of genes that produce similar effectsin a wide varity of organisms.
Match #11: Section 24_6 Using Biological Molecules to Reconstruct Phylogenetic Trees 447 /1879: The more types of **molecules we use, the better we can **detect homoplasies. For example, if we were to infer a phylogeny only from lysozyme sequences, we might falsely conclude that langurs, cows, and hoatzins are closely related to one another.
Match #12: Section 40_2_1 Epithelial tissues cover the body and line organs 695 /674: Smell and taste receptors, for example, are epithelial **cells that **detect specific chemicals. An epithelial tissue can be classified according to its structure and the appearance of its cells: •A simple epithelium is a single layer of cells, such as that lining the gut (Figure

Figure 5.19 Sample results from the Functional Keyword Search [147]

In the study, a total of 22 matches were found, with only 8 of them being deemed relevant to the query. Because the total number of relevant results in the text was not reported, the effectiveness score could not be calculated. However, from these results, the precision was calculated to be 0.36. This low precision means that a large number of irrelevant results were retrieved. This can negatively impact the idea generation process, as the user will have to go through and filter out the irrelevant results. Depending on the number of irrelevant results retrieved, this secondary filtering process could potentially be a very time consuming and tedious process. Because the subsumption-based retrieval strategy used in the Strategy Repository results in a precision and recall of 1, the secondary filtering of irrelevant results is avoided.

5.6 CLOSURE AND VALIDATION

Current keyword-based retrieval strategies utilized in the current approaches to bio-inspired design suffer from inefficient retrieval, either retrieving too many and/or

irrelevant results. This requires the designer to go through an additional step of filtering out relevant results. In this chapter, to improve on the current inefficiencies in retrieval, we address the following question:

" How can hierarchical Petri net representations of biological systems be structured to aid retrieval of relevant strategies from a knowledge repository?"

To answer this question, it was hypothesized that

Hypothesis 3: An ontology of concepts from hierarchical Petri net representations of biological systems can be represented using Description Logics. Subsumption in Description Logics will enable consistent and precise retrieval of relevant biological strategies from a knowledge repository.

To validate this hypothesis, an ontology of concepts from hierarchical Petri net representations of biological systems was implemented using Description Logics in Sections 5.1 and 5.2. Then, subsumption in Description Logics was used to enable consistent and precise retrieval of relevant biological strategies from a knowledge repository. The two primary components validated in this chapter were the consistency (Section 5.5.1) and precision (Section 5.5.2) of subsumption-based retrieval.

Consistency in retrieval is important because it ensures that our retrieval results will always be the same. To validate consistency in our hypothesis, we presented mathematical evidence in Section 5.5.1 that subsumption in DL was consistent, thus retrieval based on subsumption in DL will also be consistent. Precision is particularly important in retrieval as it shows that all the results returned will be directly relevant to the query, thus eliminating the need for a secondary filtering process. To test precision in retrieval, we do so empirically by developing a test repository of biological and engineering strategies. We then test the precision of subsumption-based retrieval using subsumption hierarchies formed by queries (formal search nodes). In Section 5.5.3, subsumption-based retrieval was found to be precise. This means that all the strategies retrieved in every query were relevant.

In all, subsumption in DL was found to enable both consistent and precise retrieval of strategies from a repository. Because of this, the use of DL as a structuring mechanism for an ontology of hierarchical Petri net concepts is also justified. Although the repository structure was found to be beneficial to the user, these advantages do bear a cost. The drawback to this approach is the fact that it is information front-loaded, meaning that a significant pre-work is needed to build the repository to a point that it is useful beyond narrow scopes. This pre-work includes finding the biological systems and associated strategies, as well as encoding these strategies for retrieval using DL. The Biomimicry Database bears a similar front-loaded cost, whereas the Functional Keyword Search alleviates much of this cost by searching through biological literature. By using an existing resource, the information does not need to be previously extracted and encoded.

It should also be noted that the retrieval results from Section 5.5 were found using a sparsely populated state of the repository, compared to the vast number of biological and engineering strategies within their respective domains. Therefore, the question becomes, "What should we expect from a much larger repository?" To answer this question, we examine how the retrieval consistency, retrieval precision, and computational time are expected to change as the repository grows. Consistency in DL assures that the order in which the repository is built does not matter and precision assures only relevant results are retrieved. Because of the mathematics-based tableau algorithm for calculating subsumption, the consistency and precision in retrieval are expected to be consistent as the repository grows. However, as the repository grows, the consistency and precision in retrieval afforded by DL comes at the cost of computational complexity. In this research, the attribute language with full existential quantification ($\mathcal{ALE}$) was used. With respect to complexity, $\mathcal{ALE}$ has a computational complexity of NP (non-deterministic polynomial time), meaning that problems of this class cannot be solved in polynomial time. NP refers to how the computation time for comparing two concepts in the ontology for subsumption relationships varies with the size of the concepts [85]. As the repository grows and new taxonomies are added, strategies can be described using additional concepts from these ontologies to aid in retrieval. As the

number of additional concepts used to describe the strategy grows, the computational complexity and computational time will grow in non-deterministic polynomial time.

To examine the performance of the DL-based repository as the number of concepts in the repository grows, we consider experiments performed by Moller et al. [148] on the scalability of description logic instance retrieval. In this study, the authors described a set of university databases using DL. The authors measured the computational time of a constant set of 14 queries as the number of concepts (universities) in the repository grew. To give an idea of the number of concepts considered in the experiments, 1 university had approximately 1714 individuals, 53738 concept assertions, and 49336 role assertions. The experiments were conducted on an AMD 64-bit processor, 4GB, Linux operating system. The results from this experiment are displayed in Figure 5.20.

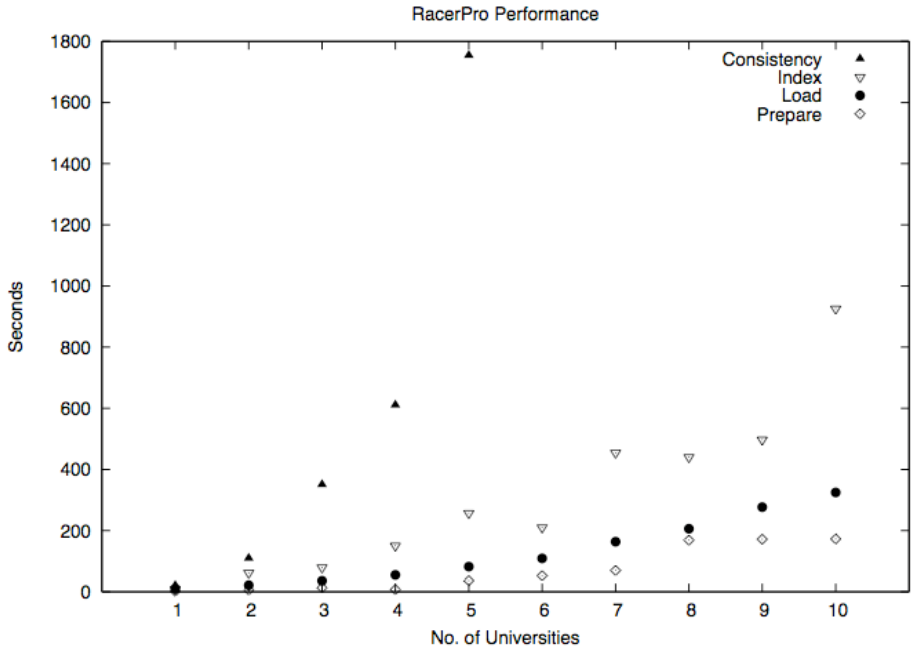


Figure 5.20 Computational time versus concept number in DL [148]

In the experiments, the authors found linear relationships between the indexing, loading, and preparation times and the number of universities. This means that the computational time will grow linearly with the size of the repository. Yim [75] and Udoyen [85] also

found linear growth of computation time with the number of concepts in their respective repositories.

Validation

With respect to our validation strategy, displayed in Figure 5.21, the validity of the strategy repository was addressed in this chapter.

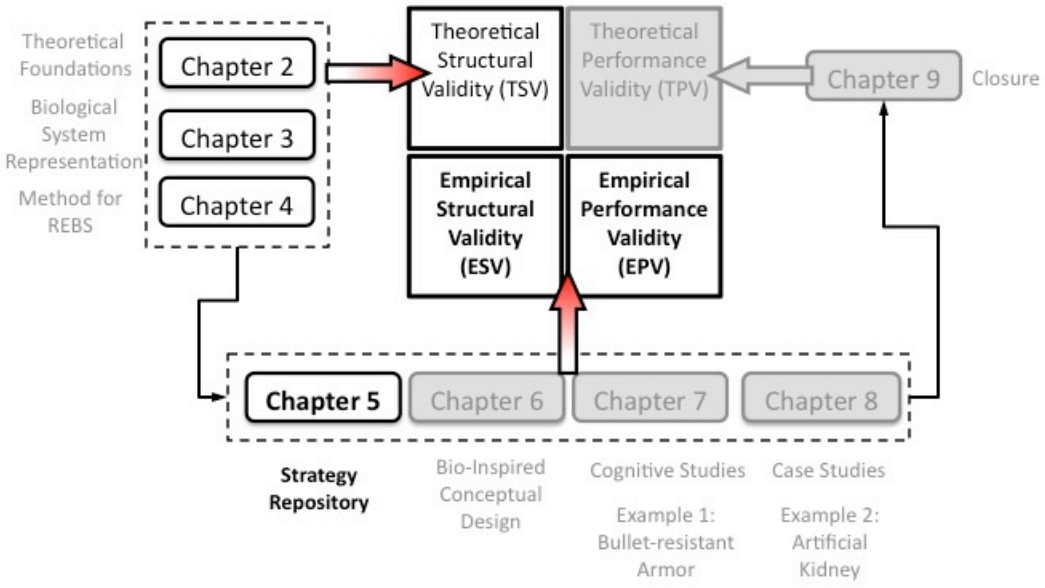


Figure 5.21 Validaton Overview in Chapter 5

In Chapter 2, the constructs of the proposed repository and retrieval method, engineering ontologies and Description Logics, were addressed. In this chapter, Empirical Structural Validity and Empirical Performance Validity follow the Theoretical Structural Validity from Chapter 2.

Empirical Structural Validity (ESV)

ESV includes accepting the appropriateness of the example problems used to verify method performance. In this research, the repository is used to store and efficiently retrieve biological and engineering strategies in Conceptual Design. This repository structure is tested within this context using a testbed repository. Using this testbed, the precision of the retrieval method was tested. The queries used to test the

retrieval method are structured after typical requests made by designers, thus the test method is deemed appropriate.

Empirical Performance Validity (EPV)

EPV involves accepting the usefulness of the method for some representative example problems. In this case, the usefulness of the repository and the retrieval method is tested using the testbed repository and test queries. It was found that subsumption in DL leads to consistent and precise retrieval of biological strategies. Comparing this to the typical keyword search, in which precision is < 1, this repository structure and retrieval method is deemed useful.

In addition, the retrieval performance of subsumption-based retrieval was compared to that of the Biomimicry Database and the Functional Keyword Search. The performance, measured with an effectiveness score F_1, was found to be less than 1 for both alternatives. This was compared to an effectiveness score equal to 1 for subsumption-based retrieval. With that, the subsumption-based retrieval strategy is deemed useful.

CHAPTER 6 BIO-INSPIRED CONCEPT GENERATION

In Chapter 3, the proposed hierarchical Petri net (hPN) representation for biological systems was presented. Building on this representation, a method for extracting biological strategies from these representations, the method for Reverse Engineering Biological Systems, was presented. In Chapter 5, a repository for storing and retrieving these strategies from a knowledge base was presented. In this chapter, the constructs of the method for Reverse Engineering Biological systems and the strategy repository are synthesized into approaches for Bio-Inspired Concept Generation. The dissertation outline is displayed in Figure 6.1.

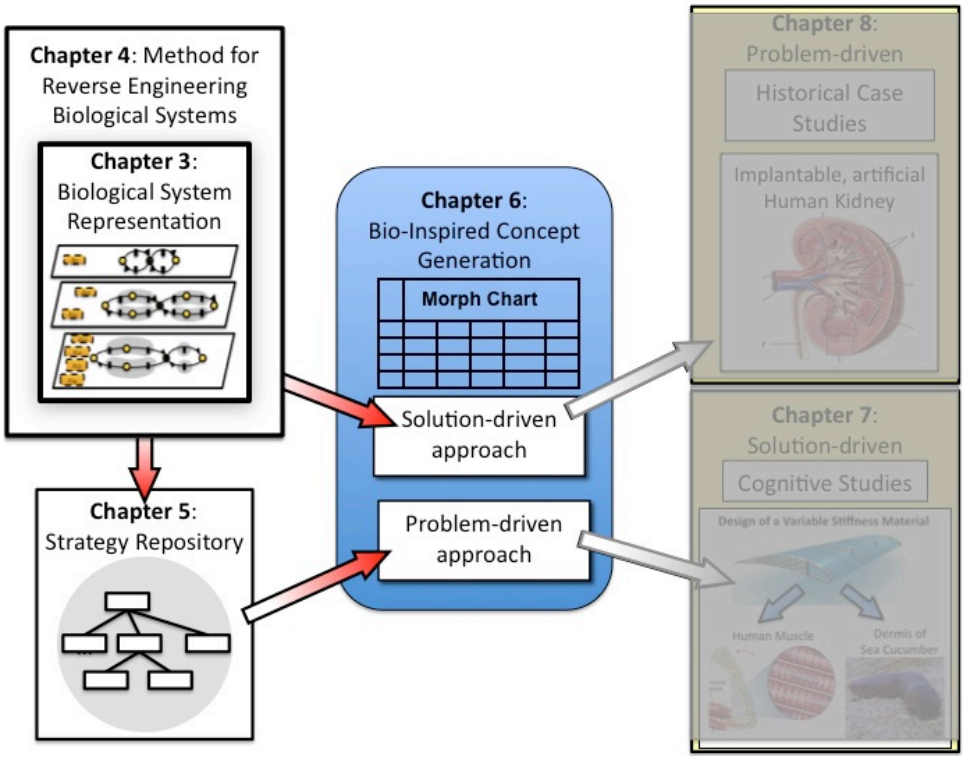


Figure 6.1 Dissertation Outline and Chapter 6

Bio-Inspired Concept Generation typically follow two distinct approaches [20]: problem-based and solution-driven. In the problem-based approach, the designer begins with an engineering problem and searches for solutions to this problem through the engineering design process. In the solution-driven approach, the designer begins with a

biological solution and attempts to mimic the behavior of this system in the engineering domain. These approaches will be addressed in the following sections.

6.1 THE FOUNDATION

In Chapter 4, the Method for Reverse Engineering Biological Systems was presented. This method is used as a means for systematically decomposing the function, behavior, and structure of biological systems using hierarchical Petri nets. This decomposition process has value to both engineers seeking to be inspired by biological strategies (problem-based approach) and those seeking to reverse engineer and extract a design strategy from a biological system (solution-driven approach). In the problem-based approach, the proposed method can be used indirectly in populating the strategy repository (Chapter 5). This strategy repository is then used to retrieve relevant biological strategies to inspire design. In the solution-driven approach, the proposed method can be used directly as a means of analyzing the biological system and extracting a design strategy from this system.

In Chapter 5, the strategy repository was developed as a means of retrieving relevant biological strategies in Conceptual Design. This repository also has value in both the problem-based and solution-driven approach. In the problem-based approach, the strategy repository can be used to access biological strategies, which are then used to inspire novel engineering systems. In the solution-based approach, the strategy repository can be used as a means for locating engineering systems, as well as other biological systems, that mimic the strategy of the reverse engineered system. This will aid in generating new ideas to mimic the strategy of the biological system.

Building upon this foundation, the problem-based and solution-driven approaches to bio-inspired Conceptual Design are presented in Sections 6.2 and 6.3, respectively.

6.2 PROBLEM-BASED BIO-INSPIRED CONCEPT GENERATION

6.2.1 Conceptual Design

In the problem-based approach to Bio-Inspired Design, the designer follows the systematic design method and uses biological strategies in the search for solutions. The Pahl and Beitz Systematic Design Method is divided into four phases: Planning and Clarifying the Task, Conceptual Design, Embodiment Design, and Detail Design. The phase of particular interest in this research is Conceptual Design (displayed in Figure 6.2). The goal of the Conceptual Design phase is for the designer to determine the principle of a solution, or the concept. In the Conceptual Design phase, the designer takes the specification of the artifact being designed and develops it into a concept through various steps. Abstraction identifies the critical specifications and requirements of the artifact. Using a solution-neutral problem statement, a function structure is created to identify functional relationships of different sub-functions of the solution. Solution variants are established for the sub-functions by identifying working principles, then assembling them into working structures and combining feasible combinations of those working structures. These variants are then evaluated based on technical and economic criteria, and the best identified as concepts to be embodied and detailed in the Embodiment and Detail Design phases.

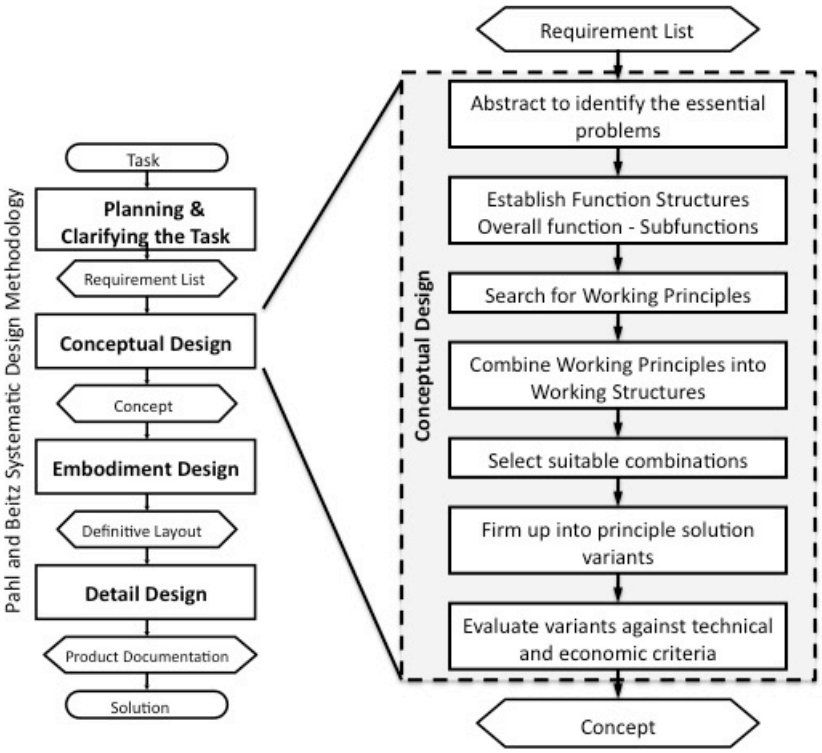


Figure 6.2 Conceptual Design phase of the Pahl and Beitz Systematic Design Methodology

One of the key steps in Conceptual Design is the 'Search for Working Principles'. In this step, the designer searches for specific solutions (working principles) to the sub-functions defined in the function structure. In this search for solutions, the aim of the designer should be to canvas as much of the design space as possible. To aid in this process, Pahl and Beitz [5] suggest several methods for identifying solutions for each sub-function, including literature search, analyzing natural and other known technical systems, and intuition-based methods (see Section 2.1). It should be noted that it is beneficial to use multiple methods for identifying solutions so as to widen the solution field as broadly as possible.

6.2.2 Problem-based Bio-Inspired Conceptual Design development

As stated in Section 6.1, this method is intended to be used as an alternative to other solution finding methods. Specifically, with respect to the Conceptual Design phase of Pahl and Beitz, we use this method in the Search for Working Principles. We also specify the inputs to the method as the functions and/or subfunctions elaborated in

the Establish Function Structures step of Conceptual Design, and the outputs being several design solutions (working principles) for consideration in the morphological matrix. The method placement is displayed in Figure 6.3.

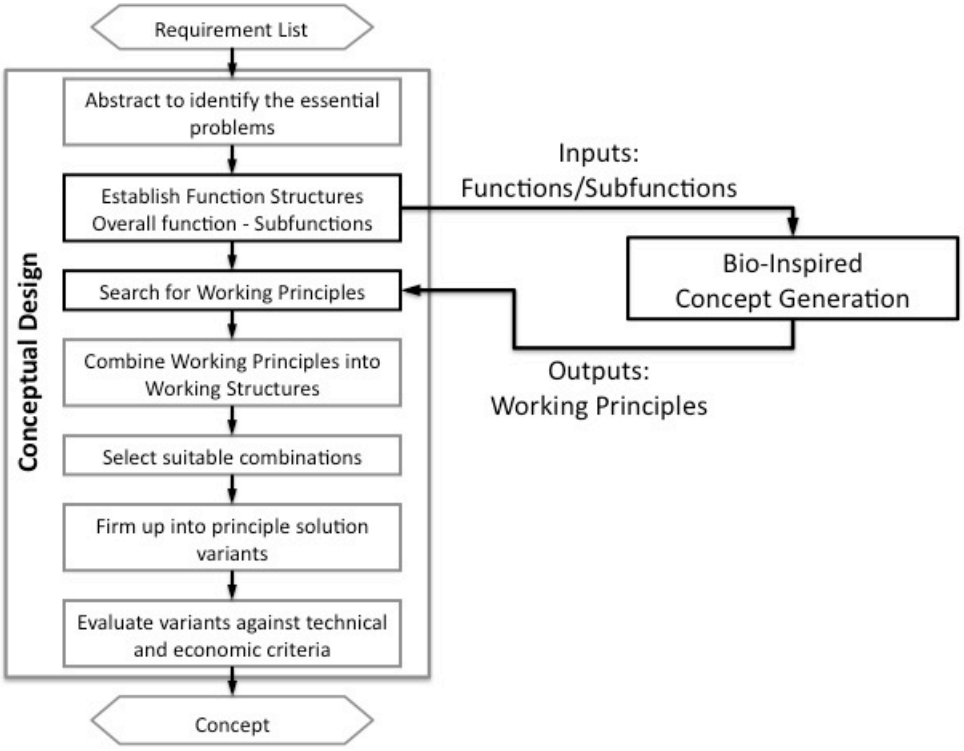


Figure 6.3 Bio-inspired Concept Generation and Conceptual Design

Now that the proposed method has been interfaced with the systematic design method, we now turn to development of the method itself. The problem-based approach to Bio-Inspired Concept Generation can be divided into three key steps: Detail requirements for function of interest, Identify biological strategies, Generate ideas. These key steps are displayed in Figure 6.4.

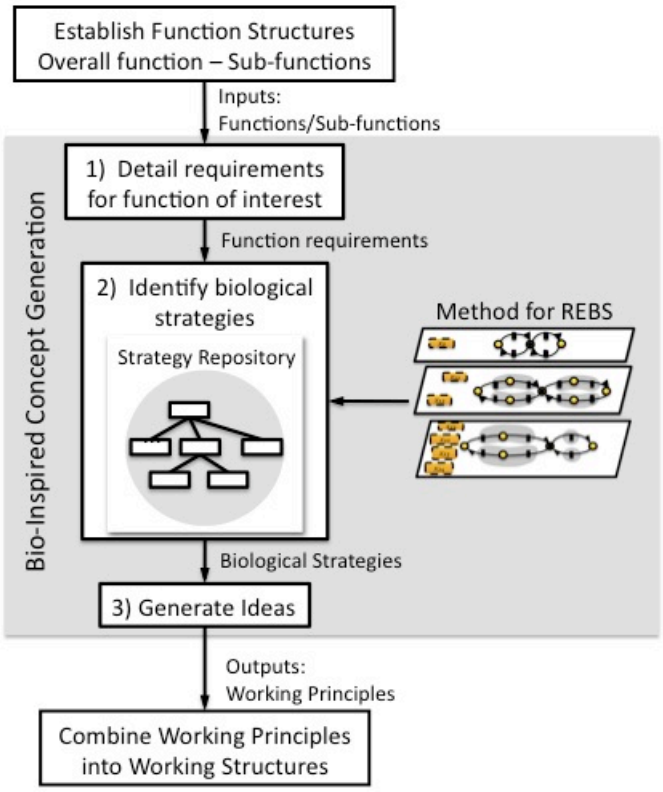


Figure 6.4. Problem-based Bio-Inspired Concept Generation

As seen in Figure 6.4, based on the requirements for the functions and sub-functions of interest identified from the function structure, candidate biological systems exhibiting these same functions are identified. The identified biological strategies are then used to stimulate ideas in idea generation. Specific steps for the proposed method for Bio-Inspired Concept Generation in the problem-based context are detailed as follows:

1) *Detail the sub-function of interest*

   In this initial step of the approach, the requirements and specifications for the sub-function of interest are identified. First, the input and output flows of the function are documented. Next, the environment of use of the component, as well as any other information deemed critical to carry out the function, is documented. The more information considered in this step eases the search for feasible working principles in the following steps.

2) *Identify biological strategies*

In this step, biological strategies are identified. The first step in identifying biological strategies is identifying analogous biological systems. These analogous systems solve a similar problem or fulfill a similar function to the function of interest (detailed in Step 1). Behavioral strategies used by the biological system to fulfill this function of interest are then extracted.

In this research, biological systems and their associated strategies are identified using the strategy repository defined in Chapter 5. Several retrieval scenarios are discussed in Section 5.5.3. Additional approaches for identifying biological strategies include: a review of biological literature, a functional keyword search through biological literature [136, 22, 23], and the use searchable databases of biological strategies [12, 3, 24, 26, 13, 21]. The use of experts in biology and related fields provides a good starting point for this search.

3) *Generate ideas*

In this step, the retrieved biological strategies are used to stimulate the generation of working principles and populate the morphological matrix. Using the biological strategies, the designer is tasked with the search for engineering technologies that mimic the strategies. These technologies are then entered into the morphological matrix and Conceptual Design continues with the selection and combination of working principles.

In the problem-based approach to Conceptual Design, the strategy repository allows direct access to biological strategies, which otherwise are difficult and time-consuming to identify. The method for Reverse Engineering Biological Systems is used to populate the repository with novel biological systems and strategies.

6.2.3 Method Characteristics and Validation

In this research, the proposed method for Reverse Engineering Biological Systems (Chapter 4) and the strategy repository (Chapter 5) are validated in the context of problem-based and solution-driven Conceptual Design. In this section, the problem-based approach was presented. The problem-based approach to Bio-inspired Conceptual Design is intended for problems with the following characteristics:

1. Problem-based, meaning the designer begins with a design problem and proceeds through the design process systematically in the search for solutions
2. Open solution field – there are multiple solutions that can satisfy a given problem
3. The designer wishes to be inspired by biological strategies, implying a transfer of strategy at a high level of abstraction
4. Novelty of solution is valued

In Chapter 7, the problem-based approach is tested using cognitive studies (Section 7.1) and a comprehensive example of the design of a hybrid, bullet resistant armor system (Section 7.2). In the cognitive studies, students are given a design problem and asked to generate solutions. In the studies, some of the students are exposed to biological strategies in the idea generation process, where others are either exposed to no strategy or a human-engineered strategy. The ideas generated are compared on a basis of novelty and variety. In the comprehensive example, we start with a design problem and proceed through the design process systematically. In this example, the strategies are retrieved using the strategy repository, as opposed to being given, as in the case of the cognitive studies. The novelty of the concept developed is compared to other concepts found in industry. As can be seen from the discussion above, both the cognitive studies and the comprehensive example possess the characteristics of the problems in which the proposed approach is intended, including: starting with a problem with an open solution field, bio-inspiration is desired, and novelty is valued. Therefore, these tests are accepted.

6.3 SOLUTION-DRIVEN BIO-INSPIRED CONCEPTUAL DESIGN

In the solution-driven approach to Bio-inspired Design, the designer takes a reverse engineering approach to engineering design. In this approach, the designer begins with a biological solution and seeks to decompose the system physically and functionally. With respect to the Pahl and Beitz systematic design method, the proposed approach is used as a means of adaptive design. In adaptive design, the designer begins with analysis of an existing solution and decomposes the system into a function structure.

Depending on the requirements of the design, the functions can be modified by variation, addition, or omission of individual sub-functions or by changes in their combination [5]. The method for solution-driven Bio-inspired Conceptual Design is developed in Section 6.2.1.

6.3.1 Solution-driven Bio-Inspired Conceptual Design development

The method for solution-driven bio-inspired Conceptual Design is displayed in Figure 6.5.

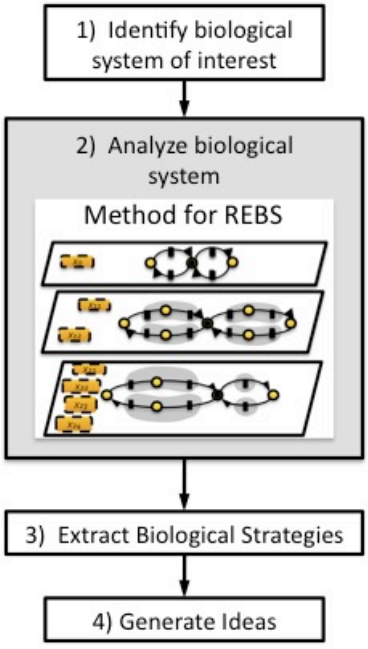


Figure 6.5 Solution-driven Bio-Inspired Concept Generation

After the biological system is identified, it is analyzed using the Method for Reverse Engineering Biological Systems put forth in Chapter 4. Using this method, the biological strategies are extracted from the system and used as a basis for the design of a new engineering system. The four key steps are detailed as follows:

1) *Identify biological system of interest*

   In the first step, the biological system of interest is identified. Identified systems typically possess some form of novel quality or feature that can be leveraged in the engineering domain.

2) *Analyze biological system*

In this step, the biological system is analyzed using the method for Reverse Engineering Biological Systems. In the first step, the system is decomposed into its physical units. A behavioral model is then created using the hierarchical Petri net representation.

3) *Extract biological strategies*

Using this representation, the biological strategy is extracted using the method presented in Section 4.2. Specifically, the strategy (S') of behavior (t) is denoted by the behavioral path ($\sigma(\bullet S^t, S^t \bullet)$) of its subnet S^t, or S'(t) = $\sigma(\bullet S^t, S^t \bullet)$, where the behavior from an initial state of the system to another state is defined as $\sigma(M_0, M_n) = t_1, t_2, \cdots t_n$.

4) *Generate Ideas*

In this step, the biological strategies are used to stimulate the generation of working principles. In this approach, engineering solutions are found that mimic the behavior of the biological strategies. The current strategy repository has also been set up to aid in identifying engineering solutions that satisfy a particular function or behavior. These technologies are then entered into the morphological matrix and Conceptual Design continues with the selection and combination of these working principles.

In the solution-based approach, the method for Reverse Engineering Biological Systems aids the designer in systematically decomposing the function, behavior, and structure of the biological system of interest. Using the hierarchical Petri net representation developed, the biological strategies can be easily extracted from the system and used to generate engineering alternatives. The strategy repository adds additional value in aiding the designer in locating engineering solutions that also share this same strategy.

6.3.2 Method Characteristics and Validation

In this section, the solution-based approach to Bio-inspired Conceptual Design was presented. The solution-driven is intended for problems with the following characteristics:

1. Solution-driven, meaning a reverse engineering approach is followed in mimicking a novel feature or behavior from an analogous system
2. Target system can be systematically decomposed
3. Quality of solution is valued
4. The designer has the initial time to invest in reverse engineering an analogous system

In Chapter 8, the solution-driven approach is tested using historical case studies (Section 8.1) and a comprehensive example of the design of a novel renal replacement system. In the historical case studies, advances in the bio-inspired systems of aviation and renal replacement are documented and compared to the behavior of their respective systems with respect to performance. In the comprehensive example, the human kidney is decomposed using the proposed method for Reverse Engineering Biological Systems. The strategy from the kidney is then used to design a new renal replacement therapy that closely mimics this strategy.

Both of the historical case studies consider advances in fields that were historically bio-inspired, whereby aviation seeks to defy gravity like birds and renal replacement seeks to replace the function of the kidney. In both cases, significant effort was put into studying their biological counterparts and mimicking them in an effort to design better performing systems. Because both case studies involved design systems based on an understanding and mimicking of nature, the historical case studies are accepted as a means for validation. The comprehensive example details the solution-driven approach to developing a renal replacement therapy that closely mimics the human kidney. The behavior of the kidney is decomposed and the strategy extracted is used to develop a quality renal replacement therapy. Since the characteristics of the example closely mimic that of the intended problem of the proposed approach, the example problem is accepted as a means for validation.

6.4 CLOSURE AND VALIDATION

In this chapter, the method for Reverse Engineering Biological systems and the strategy repository are synthesized into approaches for Bio-Inspired Concept Generation. These approaches include a problem-based and solution-driven approach to concept generation. In Section 6.1, the Conceptual Design process was detailed. The problem-based approach utilizing the tools introduced in this work was presented in Section 6.2. This method was integrated into the Conceptual Design process and systematic steps for implementation were presented. In Section 6.3, the solution-driven approach was detailed. Specifically, the proposed method and the strategy repository were synthesized into a method to aid in reverse engineering and mimicking novel behavior from biological systems.

With respect to validation, Empirical Structural Validity (ESV) is addressed in this chapter. The validation strategy for this chapter is displayed in Figure 6.6.

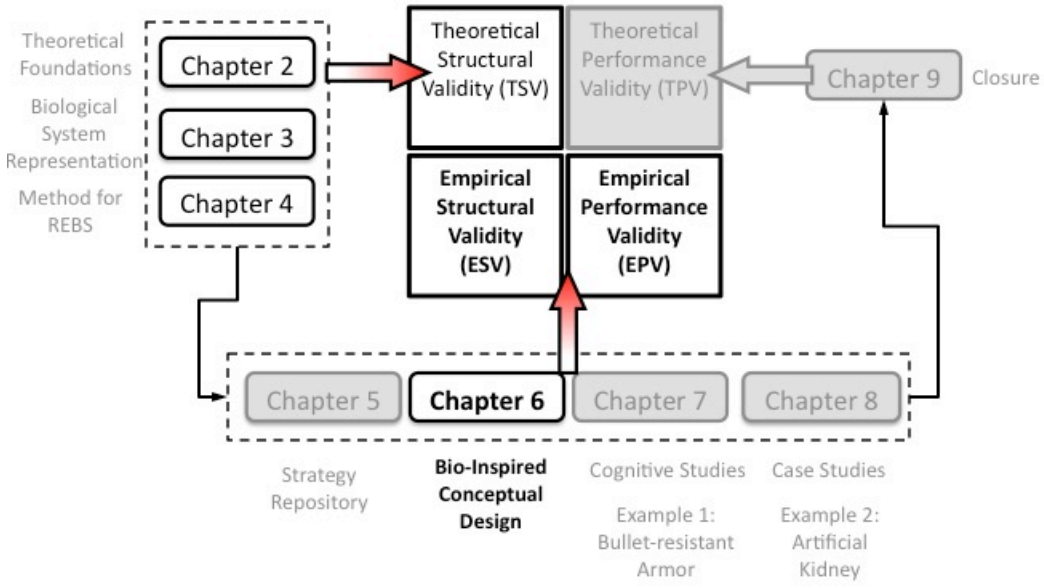


Figure 6.6 Validation Strategy and Chapter 6

ESV involves accepting the appropriateness of the example problems that are used to verify the method performance. In Section 6.1.3, the characteristics for which the proposed method for problem-based Bio-inspired Conceptual Design is intended are presented. These characteristics were then compared to the characteristics of the cognitive studies and comprehensive example used for validation. Based on this

comparison, the tests were accepted as appropriate for validating the proposed approach in the problem-based context. In Section 6.2.2, the characteristics for which the solution-driven approach to Bio-inspired Conceptual Design is intended are presented. These results were compared to the characteristics of the historical case studies and comprehensive example used for validation of the solution-driven approach. These tests were accepted as appropriate after comparing them to the characteristics for which the proposed method is intended.

CHAPTER 7 PROBLEM-BASED BIO-INSPIRED CONCEPTUAL DESIGN

In the problem-based approach to Conceptual Design, the designer is tasked with searching for novel and innovative solutions to engineering problems. However, humans are imperfect search engines [6] and tend to focus on a narrow part of the design space and overlook many valuable solutions [4]. According to the theory of bounded rationality [7-9], the space searched is bounded by the limited cognitive abilities of the designer [2]. To overcome this limitation, designers often employ several techniques to aid in idea generation (i.e, see [5]). These techniques aid the designer in expanding and exploring his/her design space more efficiently.

The goal of this research is to aid the designer in generating design ideas in Conceptual Design through the use of biological strategies. The proposed method for Reverse Engineering Biological Systems (Chapter 4) is used to systematically extract biological strategies from biological systems using the hierarchical Petri net representation (Chapter 3). A strategy repository was structured in Chapter 5 to aid in consistently retrieving these strategies. In Chapter 6, the constructs of the Method for Reverse Engineering Biological Systems and the accompanying strategy repository were synthesized into two approaches for Bio-inspired Conceptual Design: the problem-based and solution-driven approach. The role of this chapter in the dissertation plan is displayed in Figure 7.1.

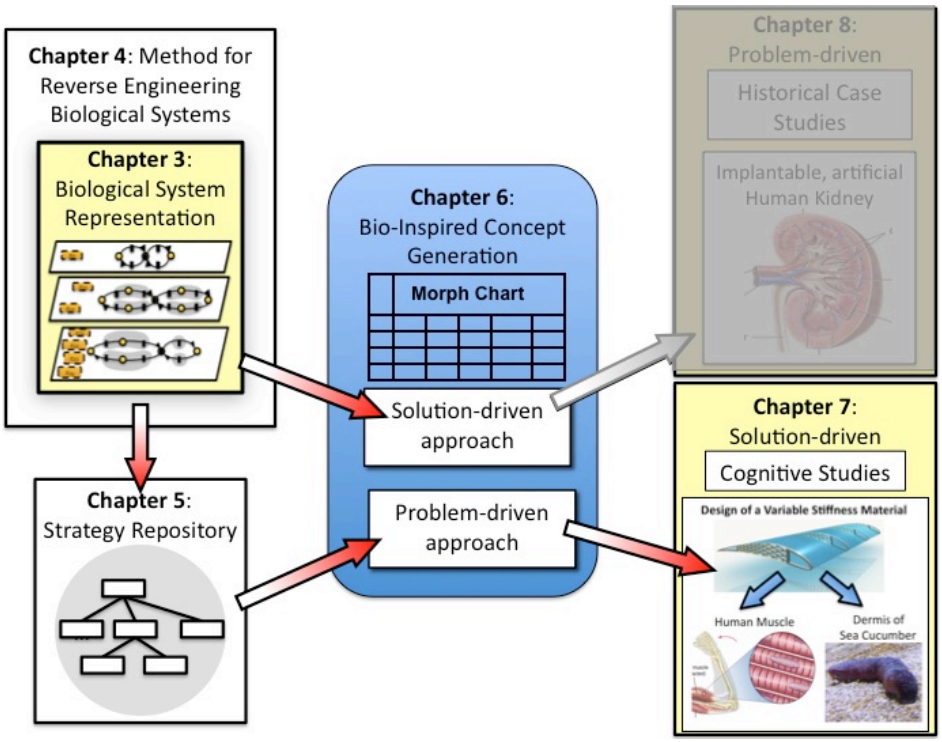


Figure 7.1 Dissertation plan and Chapter 7

In this chapter, the impact of biological strategies in the problem-based approach is reviewed. Specifically, in this chapter, the following research question is addressed:

RQ4: "What is the impact of biological strategies in the conceptual design process?"

To answer this question, it was hypothesized in Chapter 1 that:

Hypothesis 4: *4(a)* Exposure to biological strategies will increase the novelty of design ideas generated and *4(b)* will increase the variety of design ideas generated.

To validate this hypothesis, cognitive studies were performed on Mechanical Engineering students using biological strategies in ideation (Section 7.1). In Section 7.2, a comprehensive example of the problem-based approach to Bio-inspired Conceptual Design is presented. In this study, the proposed approach is used in the development of a hybrid bullet resistant armor system.

7.1 COGNITIVE STUDIES

7.1.1 Background

Theories on the effects of exposure to examples in idea generation have converged on a dual influence model of both negative (design fixation) and positive (cognitive stimulation) effects [4]. Design fixation refers to conformity effects that result in subsequent designs after exposing designers to example solutions. Cognitive stimulation refers to the stimulation of new ideas that occurs as a result of exposure to the ideas of others. Jansson and Smith [149] found that participants generated a larger amount of ideas containing features of examples to which they were exposed than did a control group not exposed to the examples. Similar results were found in studies by Purcell and co-authors [150, 151], but only when the principles from the example problem were considered to involve the same knowledge base as the expertise of the mechanical engineering participants. On the contrary, students in Industrial Design did not show these conformity effects. Conformity effects were also found in several other studies [48, 152, 153].

Examples have been shown to also positively influence idea generation through cognitive stimulation, with the knowledge embodied in the examples stimulating ideas that designers would not otherwise have been able to access [4]. In a study on idea generation in groups, Paulus and Yang [154] found enhanced performance in group brainstorming due to cognitive stimulation. Nijstad [155] found that participants exposed to ideas from a wide variety of categories surveyed more categories of ideas, compared to that of participants not exposed to any ideas and participants exposed to ideas from fewer categories. The authors also argued that problems with larger design spaces are more likely to show these stimulation effects than ones with smaller spaces. Dugosh [156] also found stimulation effects in group brainstorming and increased amounts in cases where participants were asked to attend to the ideas of others.

In this research, the value of bio-inspired techniques in aiding the designer expand and explore his/her design space in idea generation is assessed. Specifically, the impact of biological examples on design space exploration and expansion was quantified.

Defined metrics [35, 157] relating the novelty and variety of ideas generated in ideation to expansion and exploration of the design space were used (see Section 2.4). These metrics for novelty and variety are used to assess the value of biological examples in ideation. To examine this two experimental studies are conducted in which participants are exposed to biological examples in the idea generation process. These results are then compared to that of participants receiving no examples and to those receiving human-engineered examples.

7.1.2 Experimental Methods

To test Hypothesis 4, two studies were performed. In the first study, participants generated conceptual designs to solve conflicting design requirements of portability and effectiveness for a leg immobilization device designed for use in the wilderness. Three conditions were designed to test Hypotheses *4(a)* and *4(b)*: (1) a condition in which participants received a biological example after some time in ideation, (2) a condition in which participants received a human-engineered example after some time, and (3) an unaided condition where no example was presented. The unaided condition established a baseline for unaided concept generation. The human-engineered condition established a baseline for concept generation using within-domain design examples. In the biological condition, we assessed the value of biological design examples. Comparisons to test our hypotheses were made both within groups and between groups. The within-group analysis compared the design ideas generated before and after exposure to the design example, while the between-group analysis compared the ideas generated after exposure to a design example in each of the different conditions. In Study 2, a similar procedure was followed to test the robustness of the results from Study 1.

7.1.2.1 Study 1 – Leg Immobilization versus Portability

Participants

Twenty-six mechanical engineering students from Georgia Institute of Technology participated in the Study 1. The students were recruited from a senior level capstone design course. To be eligible for the course, students must have at least three

years of undergraduate engineering coursework and at least one semester of a formal undergraduate design course. All participants were volunteers and did not receive payment or course credit for participation in the study.

Materials

All participants were asked to solve the same design problem individually. The design problem given in Study 1 is displayed in Figure 7.2. In Study 1, participants were asked to solve a design problem with conflicting design requirements of leg immobilization and portability.

Mountain-TREK

Mountain-TREK (MTREK) is an outdoor wilderness company that organizes backpacking trips to the mountains throughout the year. During these trips, MTREK utilizes trip guides to lead a group of participants through these wilderness expeditions. For safety reasons, MTREK requires each of its guides to carry emergency kits containing an assortment of medical supplies. These kits contain items that can be used in the case of sickness, insect bites, wounds, trauma, etc. Due to the other items outfitted in the guides' packs, available space is limited. In extreme hiking conditions, MTREK has noticed a significant risk of leg and ankle dislocations and fractures.

Design challenge

Due to the potential for leg injuries, MTREK is now requiring guides to carry additional supplies to treat these injuries. In this design challenge, MTREK has hired you to design a device that can be used to immobilize a joint or limb in case of an extreme injury. This device must (1) be as light and small as possible when stored in the guides' packs but (2) rigid enough and large enough to immobilize the leg of an average-sized male.

Figure 7.2 Design problem for Study 1

Experimental Procedure

This design study was carried out in a classroom setting on the second meeting of the senior capstone design course. Two experimenters, including the author, administered the study. At the onset of the study, participants were given the design problem, several sheets of paper, and a black pen by the experimenters. Participants were assigned to either the biological (N = 9), the human-engineered (N = 9), or the unaided

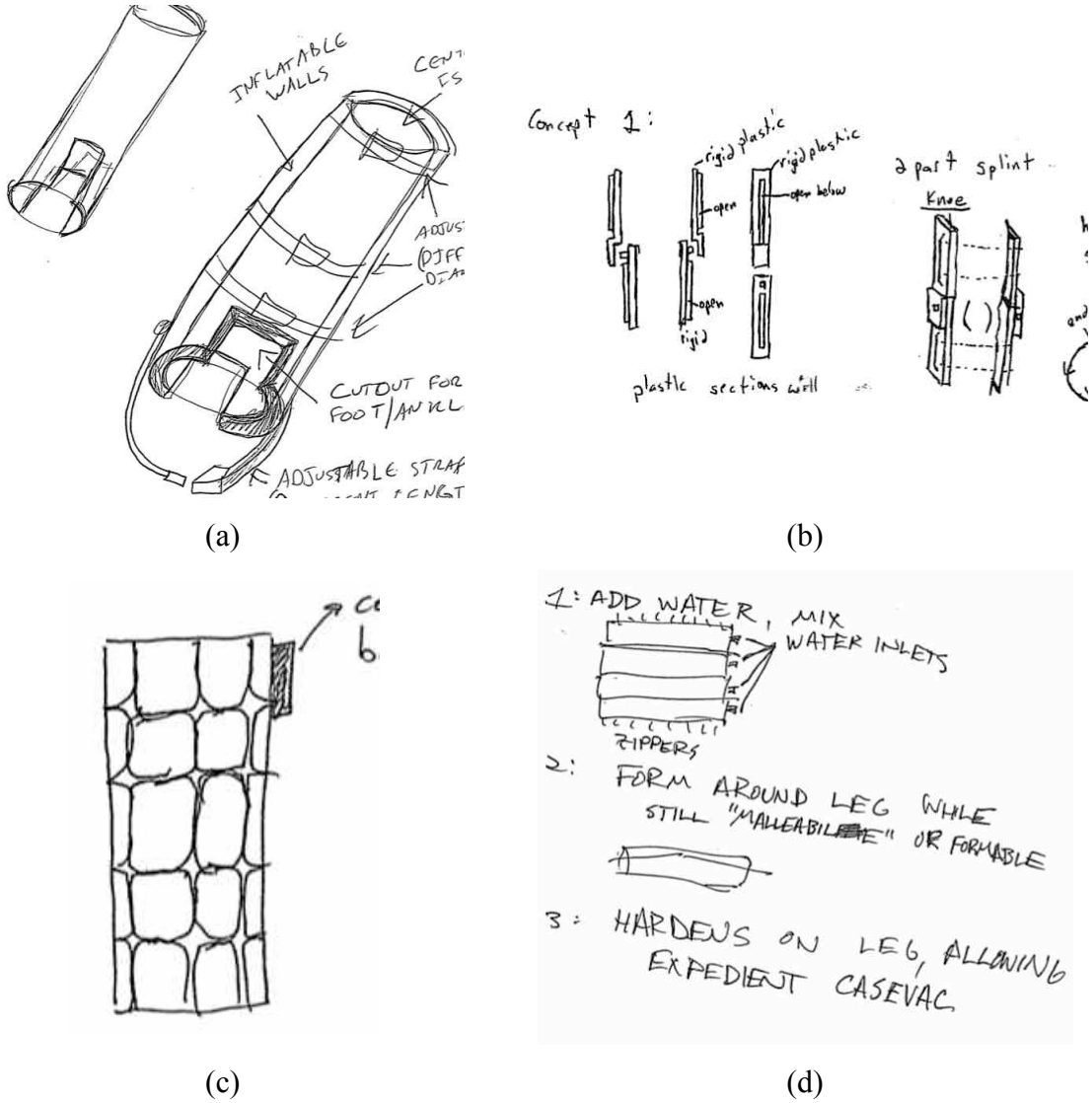
(N = 8) condition in an alternating fashion based on seating location. The condition assignment was designated by the color paper they received, although the students were unaware of this. The students received verbal instructions on how to proceed. Students were asked to generate and label distinct design ideas, with no explicit instructions being given regarding quantity or creativity of the design ideas (so as not the bias the participants). After the instruction and time for questions, the students were given 20 minutes to generate design ideas for the design problem. After the 20 minutes for initial idea generation, the participants were given blue pens to enable distinguishing design work from the initial 20 minutes from the second 20 minutes. The participants in the biological and human-engineered conditions were handed their respective design aids and verbally instructed that they were receiving design aids that may or may not prove useful to them and that they were not required to use the aids. Specific procedures for each condition are described as follows:

Biological Condition - Participants in the biological condition were given a biological design example at the midpoint of the study that described the variable-stiffness behavior of the mutable connective tissue of the sea cucumber. The design example included a pictorial and textual description of the biological systems and is displayed in Figure C. 1 of Appendix C.

Human-engineered Condition – Participants in the human-engineered condition received a design example with a pictorial and textual description of a human-engineered system with similar behavior to that of the biological system. The design example described the variable-stiffness behavior of electro-rheological fluids and is displayed in Figure C. 2 of Appendix C.

Unaided Condition – After the initial 20 minutes of concept generation, participants in the unaided condition received no design example and were instructed to continue with concept generation.

After an additional 20 minutes to generate design ideas, the students were stopped and given a post survey to evaluate their prior knowledge and experience as well as their understanding of the design examples given to them. Figure 7.3 displays examples of design solutions generated in Study 1.



(a) (b)

(c) (d)

Figure 7.3 Sample of design ideas from Study 1 (a) Inflatable (b) multiple-part snap (c) electrorheological fluid chambers with power source (d) chemically-rigidizable

7.1.2.2 Study 2 – Protection versus Comfort

Participants

Twenty-one mechanical engineering graduate students from Georgia Institute of Technology participated in Study 2. The students were recruited from a design research

laboratory. All students had undergraduate degrees in an engineering discipline and conducted research in a field related to engineering design.

Materials

The design problem given in Study 2 is displayed in Figure 7.4. In this study, participants were asked to solve a design problem with conflicting design requirements of level of protection and comfort.

Hybrid Armor

Combat-Zone (CZ) is a defense contractor that develops specialty armor for military and state and local police units. Current armor comes in two varieties: (1) hard armor (i.e. ceramic plates or shields) for high risk situations, and (2) soft armor (i.e. Kevlar) for low risk, everyday wear. Soft armor provides comfort and portability for the user, but offers little protection in the face of high-caliber projectiles or explosions. Hard armor, on the other hand, provides a maximum level of protection but are too cumbersome and heavy for long-time use.

Design Challenge

CZ is currently trying to develop a form of hybrid armor for use in high and low risk situations. There are no restrictions on the design, only that the armor must (1) not restrict the user in everyday activities and (2) be protective enough for use in high risk situations, should one come about suddenly.

Figure 7.4 Design Challenge from Study 2

Experimental Procedure

Study 2 was carried out in a conference room setting where the design research lab meetings were normally held. The author administered the study. At the onset of the study, participants were given a packet containing the design problem, several sheets of paper, the design example, and a black writing utensil by the experimenters. The design example was located at the back of the packet and the students were instructed not to flip

through the packet. As in the previous study, the domain of design example (biological vs. human-engineered) was again manipulated in this study. Within their respective domain, the type of design example was also manipulated. Participants in the biological condition (*N=10*) were presented either a variable-stiffness (*N=4*) or a shape-changing (*N=6*) biological example. Participants in the human-engineered condition (*N=11*) were likewise presented with either a variable-stiffness (*N=5*) or shape-changing (*N=6*) human-engineered example. Verbal instructions were then given on how to proceed. As in Study 1, the students were asked to generate and label distinct design ideas, with no explicit instructions given regarding originality or the number of ideas that the students were expected to generate. Particular attention was paid to not biasing the participants.

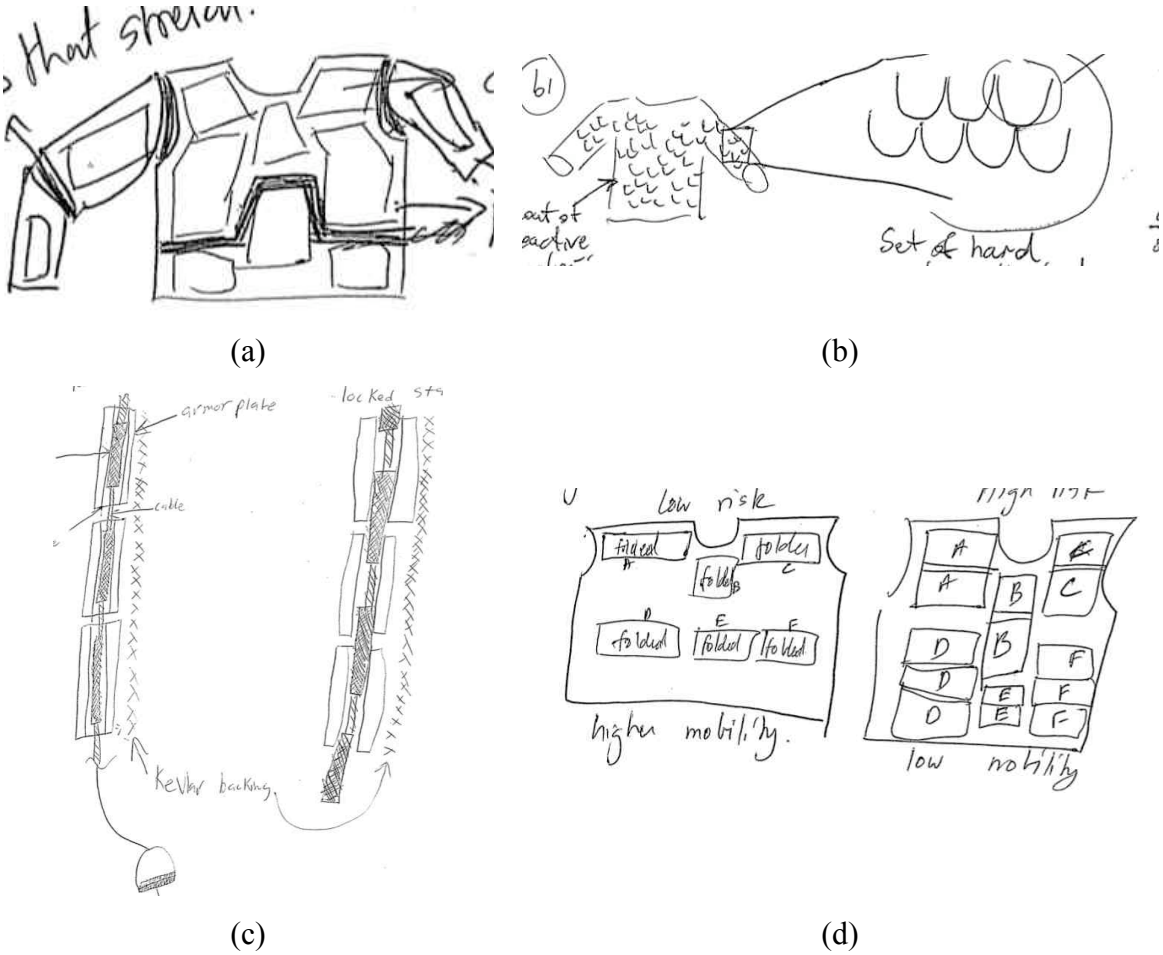
After the instructional period, the students were given 20 minutes to generate design ideas for the design problem. After the 20 minutes of initial idea generation, the students were asked to flip to the back of their study packets to the design aid. At this time, the students were given green pens and verbally instructed that the design aids may or may not prove useful to them and that they are not required to use the aids. Specific procedures for each condition are described as follows:

Concept generation using biological design examples - In the biological condition, after 20 minutes of initial concept generation, participants were given a biological design example describing either (1) the variable-stiffness behavior of human muscle in isometric contraction or (2) the shape-changing behavior of plants possessing nastic movement. The design example included a pictorial and textual description of the biological phenomena of interest. The design examples presented are displayed in Figure C. 3 and Figure C. 4 of Appendix C.

Concept generation using human-engineered design examples - Similar to that of the biological test condition, participants in the human-engineered condition were given human-engineered examples after the initial 20 minutes of concept generation. The participants in this condition received a pictorial and textual description of either (1) the variable-stiffness behavior of electro-rheological fluids or (2) the shape-changing

behavior of shape memory polymers. The human-engineered design examples used are displayed in Figure C. 5 and Figure C. 6 of Appendix C.

After an additional 20 minutes of generating design ideas the students were stopped and given a post survey. Examples of design solutions generated during the study are displayed in Figure 7.5.



(a) (b)

(c) (d)

Figure 7.5 Sample design solutions from Study 2 (a) Segmented armor (b) armor scales (c) pull-string activated variable-stiffness armor (d) foldable armor plates

7.1.3 Data Analysis

To investigate the impact of biological design examples on idea generation, we examined how exposure to biological examples aided designers in expanding their design space (increased novelty of ideas generated) and exploring their design space (increased

variety of ideas generated). Metrics for novelty and variety were assessed using the drawings and descriptions generated by the participants. The experimenters first coded the participants' design ideas. These ideas were categorized with respect to the working principle used to solve the design problem. After coding the results separately, the experimenters compared and agreed upon a common categorization. After this point, the metrics for novelty and variety were calculated based on this categorization. These metrics are defined and discussed in Section 2.4.

7.1.4 Results

The novelty and variety of the design ideas generated by the participants were calculated using the metrics defined in Section 2.4. The results were analyzed using two-tailed, nonparametric *Wilcoxon signed-ranked tests* and *Mann-Whitney U tests* at an alpha level of 0.05 ($\alpha = 0.05$). The more conservative nonparametric tests were chosen due to the small sample sizes resulting in non-normally distributed results.

7.1.4.1 Results from Study 1

Novelty

Hypothesis 4a stated that exposure to biological design examples would increase the novelty of design ideas generated. This hypothesis was tested by comparing the novelty scores of the participants in the biological condition before and after the design example was introduced. These results were then compared to that of the human-engineered and unaided conditions. The mean and standard deviation (SD) of the novelty scores for the three conditions are displayed in Table 7.1.

Table 7.1 Novelty scores for Study 1

	Control (SD)	Bio-Inspired (SD)	Human-engineered (SD)
Before	0.31(0.25)	0.34 (0.09)	0.30 (0.20)
After	0.41 (0.33)	0.76 (0.24)	0.89 (0.18)

Analysis of the results from Table 7.1 yields the following:

- Participants in both the biological and human-engineered condition showed statistically-significant increases in novelty ($W = 1$, $p < 0.01$ and $W = 1$, $p < 0.01$, respectively) after being exposed to the example; participants in the unaided condition showed no significant change in the novelty of their design ideas.
- Participants in both the biological and human-engineered conditions generated design ideas with higher novel ($U = 68$, $p < 0.01$) than participants in the unaided condition, but the novelty scores of participants in the biological and human-engineered condition did not differ significantly from each other.

These results supported the hypothesis that exposure to biological examples leads to design ideas of greater novelty. However, we also found the same effect with participants exposed to the human-engineered example.

Variety

Hypothesis 4b stated that exposure to biological examples would lead to a greater variety of design ideas. This hypothesis was tested by comparing the variety of the design ideas produced by the participants before and after the design examples were introduced. The mean and standard deviation (SD) of the variety scores of participants in the three conditions are displayed in Table 7.2.

Table 7.2 Variety Scores for Study 1

	Control	Bio-Inspired	Human-engineered
Before	5.8(5.6)	12.1 (7.8)	13.1 (8.7)
After	5.6 (6.8)	9.4 (7.6)	3.2 (4.3)

The results from Table 7.2 are summarized as follows:

- Participants in the biological and unaided conditions showed no statistically-significant change in the variety of design ideas during the two phases of the study while participants in the human-engineered condition showed a significant decrease ($W = 5$, $p < 0.05$) in the variety of design ideas generated after receiving the example.

- The variety of design ideas of participants in the biological and unaided conditions did not differ significantly.

The hypothesis that exposure to biological examples will lead to a greater variety of design ideas was unsupported with these results. However, while the biological example did not increase the variety of design ideas relative to the unaided condition, it did not decrease the variety of the design ideas, which had resulted in the human-engineered condition.

7.1.4.2 Results from Study 2

As in Study 1, the novelty and variety of the design ideas generated by the participants in Study 2 were calculated using the metrics defined in Section 2.3. The results were analyzed using the nonparametric *Wilcoxon signed-rank* and *Mann-Whitney U* tests at an alpha level of 0.05 ($\alpha = 0.05$).

Novelty

Hypothesis 4a stated that exposure to biological design examples in ideation would increase the novelty of design ideas generated. In this study, we aimed to build upon the results from Study 1 and provide further support for Hypothesis 4a using multiple examples. Table 7.3 displays the results for the novelty scores of the participants in Study 2.

Table 7.3 Novelty scores for Study 2

	Biological (SD)	Human-engineered (SD)
Before	0.42 (0.23)	0.52 (0.27)
After	0.93 (0.06)	0.76 (0.28)

Analysis of the results in Table 7.3 yields the following:

- Participants in the biological condition showed a statistically-significant increase ($W = 0$, $p < 0.01$) in the novelty scores after exposure to the design example.

- Participants in the human-engineered condition showed a statistically-significant increase *(W* = 6, $p < 0.05$) in the novelty of design ideas after exposure to the design example.

The results for this study further support Hypothesis 4a in that exposure to biological examples aids the designer in generating higher novelty solutions. Participants in the biological condition showed significant increases in novelty after being exposed to the biological example.

Variety

Hypothesis 4b stated that the exposure to biological strategies in idea generation will increase the variety of the design ideas generated. Table 7.4 displays the variety scores for the participants in Study 2.

Table 7.4 Variety Score for Study 2

	Biological (SD)	Human-engineered (SD)
Before	13.8 (10.9)	14.7 (9.5)
After	7.1 (9.64)	4.73 (5.8)

Analysis of the results in Table 7.4 are summarized below.

- The biological design example showed no statistically-significant effect on the variety of design ideas generated.
- The human-engineered design example significantly decreased the variety ($W = 3$, $p < 0.01$) of design ideas generated.

These results further support the results from Study 1 in that exposure to biological examples had no significant effect on the variety of ideas generated. With that, we could not find evidence to support Hypothesis 4b.

7.1.5 Discussion

The results from the studies presented in this chapter supported the hypothesis (*4a*) that exposure to biological examples in the idea generation process increases the

novelty of design ideas. The results from the study did not support the hypothesis that exposure to biological examples in the idea generation process increases the variety of design ideas generated, although biological examples do not have the negative effects on variety that occur with human-engineered examples.

In this work, novelty was defined as the uniqueness of an idea with respect to the unaided group of ideas generated by the group of participants. Hypothesis 4a stated that exposure to biological examples in the idea generation process would increase the novelty of ideas generated, and it was found that exposure to biological examples did indeed increase the novelty of ideas generated. This increase in novelty signifies that participants were able to generate ideas that otherwise would not have been accessed without exposure to the biological example. Shah et al. [35] correlated this increase in novelty to a broadening of the design space of the designer. In the first study, participants not receiving any examples showed no increase in the novelty of their design ideas during the study, while those receiving a human-engineered example showed an increase in novelty in both studies.

Hypothesis 4b stated that exposure to biological examples would increase the variety of the design ideas generated. This hypothesis was unsupported by the results from the two studies. In both studies, the variety before and after exposure to the biological example showed no statistically-significant difference in variety. Although exposure to the biological examples did not increase the variety of ideas generated, it did however seem to maintain the variety of the generated ideas (note: we can not definitively conclude this from the above studies). Exposure to examples in idea generation has been shown to cause fixation[149-151], decreasing the variety of the generated ideas. In both studies, a significant decrease occurred in the variety of ideas generated after exposure to the human-engineered example, while participants in the first study that were not exposed to an example showed no change in variety. To find a possible explanation for these phenomena, we examined the abstraction level at which characteristics of the example problem were transferred to the design ideas generated.

Knowledge can be transferred from design examples at varying levels of abstraction. To assess the level of transfer of ideas, we first decomposed the design

example into its key characteristics. Next, scores were calculated by analyzing the level at which the key characteristics were transferred to the design ideas of the particiapants. The physical principle and working principle levels of transfer were considered. The results from both studies are displayed in Table 7.5.

Table 7.5 Level of Transfer scores

Level of transfer	Bio-Inspired	Human-engineered
Physical Principle	1.46	1.71
Working Principle	0.37	1.31

While there was no difference in the number of ideas generated or the number of ideas possessing characteristics at the physical principle of the example, participants exposed to human-engineered examples transferred significantly more characteristics ($U = 300$, $p < 0.01$) at the working principle level of abstraction than those exposed to biological examples. In the human-engineered case, this fixation at lower levels of abstraction could have constrained the variety of ideas in idea generation. In the biological case, transferring principles at a high level of abstraction could have allowed a greater variety of ideas to still be generated.

In reviewing these results, a case for the validity of the results with respect to bias can also be made. In these studies, two sources of bias could be imagined. The first source is that of the experimenters influencing the results when giving the instructions. In these studies, the experimenters gave no specific instructions to the participants regarding the number and/or type of ideas that they should generate. They also gave explicit instructions that the participants did not have to use the design aids. The other source of bias can come from coding the results. To reduce the bias in coding the results, the experimenters first coded the ideas separately. Then, the categorizations were compared and a common categorization found. After this point, the scores calculated for novelty and variety lacked the subjectivity of the experimenters. The metrics were simply calculated based on the categorizations.

In Section 7.1, the benefits of biological strategies in Conceptual Design were presented. In the cognitive studies, access to relevant biological strategies was assumed. In the Section 7.2, a comprehensive example of problem-based Bio-inspired Conceptual Design of a hybrid bullet resistant armor system is presented.

7.2 COMPREHENSIVE EXAMPLE: DEVELOPMENT OF HYBRID, BULLET RESISTANT ARMOR

In Section 7.1, cognitive studies were performed on mechanical engineering students using biological strategies in idea generation. These studies tested the value of the end product of the Method for Reverse Engineering Biological Systems, biological strategies, in the context of the problem-based approach to Bio-Inspired Concept Generation. In the cognitive studies, access to relevant strategies was assumed. In this section, a comprehensive example of the development of hybrid, bullet resistant armor is presented. The purpose of this example is to detail the entire process of problem-based Bio-Inspired Concept Generation. In this example, the strategy repository developed in Chapter 5 is used to identify relevant biological strategies.

The defense community is currently focusing on equipping military troops and law enforcement personnel with better performing and more comfortable body armor. Presently, body armor comes in two types: hard and soft. Hard body armor (displayed in Figure 7.6) incorporates thick ceramic or metal plates, inserted into pockets in the vest covering vital areas, and deflects bullets. Such vests are rigid, the plates make them cumbersome and heavy, and they restrict the wearer's movement.

Figure 7.6 Steel armor plates for hard body armor

In contrast, soft body armor (displayed in Figure 7.7) operates on a different principle. Soft armor is comprised of a strong, dense net of fibers which absorb the impact energy of a bullet and disperse the load evenly over the rest of the vest. One such material is Kevlar®, manufactured by DuPont, which is lightweight but nearly five times stronger than steel of the same weight. When bullets impact the armor, they are caught in the web of fibers, absorbing and dispersing the impact energy, and deforming the bullet. Additional energy is absorbed by each layer in the vest until the bullet stops penetrating any further. The layers of material working together permit the breadth of the vest to assist in preventing the bullet from penetrating. This also prevents "non-penetrating" injuries, such as blunt trauma to internal organs.

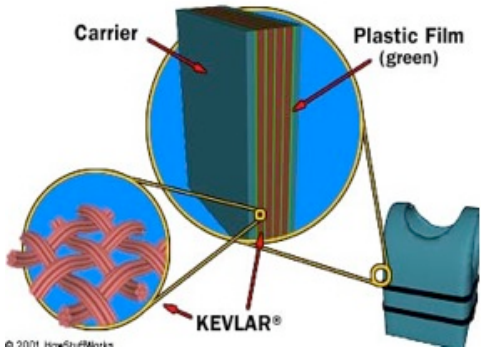


Figure 7.7 Kevlar® based Soft Body Armor (HowStuffWorks)

7.2.1 The Problem

Current vest styles necessitate tradeoffs between the level of protection and the level of comfort and flexibility. As the level of the threat increases, vests must be heavier and bulkier in order to protect the wearer and disperse the projectile load. More protective vests incorporate materials which are rigid, heavy and bulky and are therefore impractical for routine use. Such vests are typically reserved for use in tactical situations, and worn for short periods of time when confronted with higher threat levels. In lower risk situations, soft armor is appropriate because it is flexible and light. However, this compromises the level of protection the vest offers. Military and police alike are searching for vests that offer more protection without increasing bulk or decreasing comfort.

7.2.2 Conceptual Design of Hybrid, Bullet Resistant Armor

In the problem-based approach to Bio-inspired Conceptual Design, the designer begins with an engineering problem and searches for solutions to these problems. Given the problem presented in Section 7.2.1, a comprehensive example of the use of the proposed approach is presented in this section.

The steps for problem-based approach to Bio-inspired Conceptual Design include (1) identifying and detailing the sub-function of interest, (2) identifying candidate biological systems, and (3) idea generation.

6) Identify and detail the sub-function of interest

In this initial step, the requirements and specifications for the function/sub-function of interest are identified. The specific function of the current vest is to absorb/disperse the energy from projectile impact, with functional inputs of mechanical energy (force) and outputs of mechanical energy (reaction force). The vest is used as a means for protection in high and low risk situations.

In the current vest, impact absorption/dispersion is enacted by the mechanical stiffness of the material, either in a plate (in hard armor) or weave (in soft armor) configuration. In order to allow varying levels of protection and comfort for different conditions, the stiffness of the vest material can be controlled.

7) Identify candidate biological systems

In this step, candidate biological systems of interest are identified based on the requirements and specifications of the sub-function put forth in Step 1. In the proposed approach, the strategy repository developed in Chapter 5 is utilized to aid in identifying relevant biological strategies. Since the goal is to develop a system that allows control over the stiffness of the vest, a search for strategies that "increment stiffness" is input in the repository. This search and the retrieved strategies are displayed in Figure 7.8.

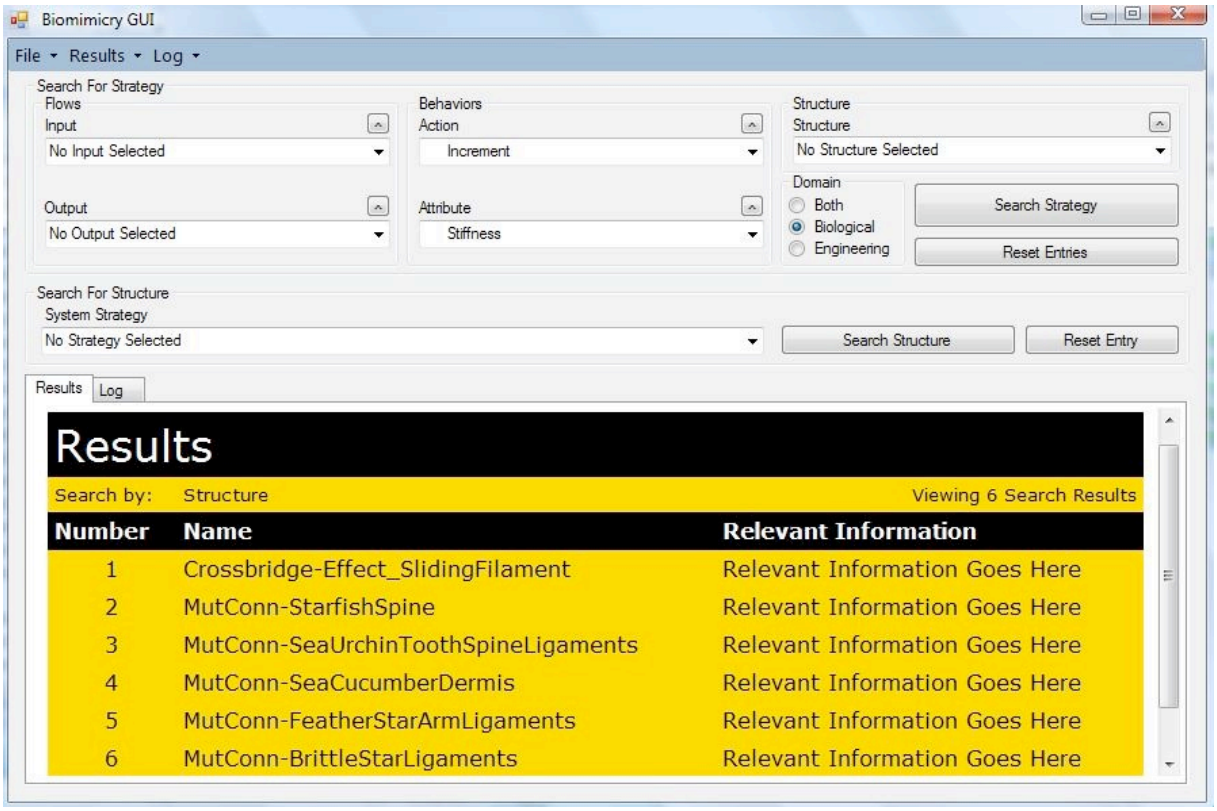


Figure 7.8 Repository search and results

Two primary strategies were retrieved from the repository: Crossbridge Effect (Result 1) and Mutable Connectivity (Results 2-6). These strategies are defined as follows:

Mutable Connectivity: *"Stiffness in the dermis is changed by controlling the association of the collagen fibril bundles"*

Crossbridge Effect: *"Stiffness in the Muscle Fiber is changed by controlling the bridging of the actin and myosin filaments of Myosin"*

By comparing these two strategies, the similarity in strategy between these two very different biological systems can be seen. Specifically, the stiffness of both systems is controlled by an association, or bridging, of the major structural elements. In domain-insensitive terms, a unified strategy can be expressed as follows:

"Stiffness can be changed by controlling the association, or bridging, between free, rigid elements in a system"

8) Idea Generation

In this step, the strategies are used to stimulate engineering concepts. Specifically, engineering strategies that mimic the retrieved strategies are sought. Using the strategy from Step 2 as a starting point, several ideas were generated that utilize this strategy, including:

- *Electrorheological (ER) and Magnetorheological (MR) fluids*- Electrorheological (ER) fluids are fluids that experience increased yield stress in the presence of electric fields. ER fluids consist of electrically polarizable particles suspended within a non-conductive fluid medium. In the absence of an electric field, ER fluids behave as Newtonian fluids, whereas in the presence of such field, they immediately solidify [158]. Magnetorheological (MR) fluids are considered the magnetic analog to ER fluids. Consisting of small magnetic particles dispersed in a carrier fluid, the shear yield stress of these fluids exhibits a strong dependence on the magnetic field applied. When used within composite materials, these fluids add the ability of active control of the material properties of a composite material [159-162]. With these fluids, an external field (magnetic/electric) is used to control the stress transfer between the rigid elements (particle suspensions).
- *Shape Memory Polymers (SMPs)*- SMPs are polymers that can be deformed into one shape, and under thermal activation, be restored back to its original shape. Under this thermal activation, the SMP also changes stiffness and becomes flexible. SMPs return to their predefined shape under thermal cycling, whereas typical high-strength polymers or liquids are not able to recover this strain. This recovery strain is needed because tension must be kept on the skin during deformation to eliminate buckling. The tailorable mechanical properties of these SMPs also make them attractive, especially when reinforced with a higher stiffness element, such as a carbon nanofiber. With SMP composites, thermal activation is used to control the state of the polymer, thus controlling the stress transfer between the reinforcement.
- *Phase Change Materials* - Phase change materials (PCM) can be defined as materials (commonly polymers) formulated to undergo phase transitions at

prescribed temperatures [163]. Once a solid state PCM reaches the prescribed temperature, it liquifies, and absorbs heat without any additional temperature change. Once the ambient temperature drops, the PCM solidifies releasing the stored latent heat [164]. PCM are commonly used for thermal energy storage for insulation and electronics and recently as nonvolatile memory in computer microchips. These phase change materials can be used to control the stress transfer between rigid elements in a matrix material. In the flexible state, a composite material is heated and the PCM changes to a liquid state, thus effectively inhibiting stress transfer between the rigid elements in the composite.

Next, the concept is firmed up into working principles. A possible working principle utilizing electrorheological fluids is displayed in Figure 7.9.

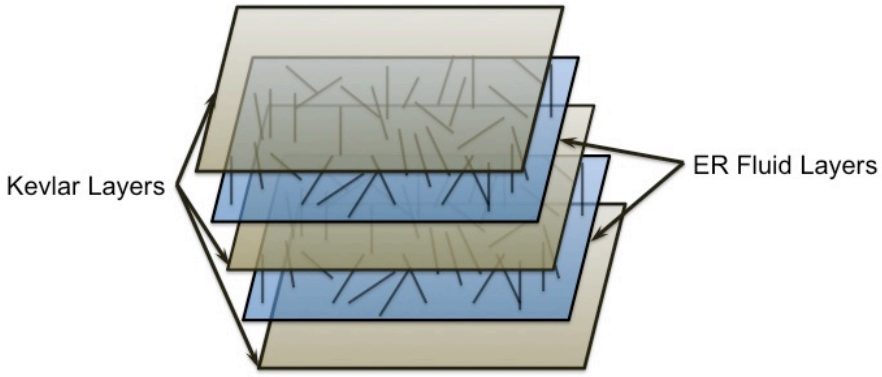


Figure 7.9 Hybrid Armor Concept

In this design concept, layers of woven soft armor are alternated with layers of ER fluid, with the ER fluid controlling the coupling between the soft armor layers. When activated, the ER fluid rigidizes and forms a rigid load absorption layer. This layer also couples the soft armor layers, allowing stress transfer between these layers and increased load dispersion. The ER fluid is composed of high aspect ratio particles. These long fibers allow for increased load dispersion over traditional spherical particles. This concept allows the user to actively modify the properties of the armor for the situation. thus allowing the flexibility and comfort of a traditional soft armor vest in low risk situations and the rigidity needed in higher risk situations.

7.2.3 Discussion

The goal of this research is to aid the designer in idea generation through the use of biological strategies. This concept utilizes the strategy of controlled association from the mutable connectivity and crossbridge effect strategies retrieved from the repository. This strategy allows the conflicting design requirements of comfort and protection posed in the design problem to be solved.

Assessment of Novelty

Novelty can be viewed as the uniqueness of a given design idea with respect to a universal world of ideas. In this case, the active armor concept is compared to others found in industry. As mentioned in Section 7.2.1, there are currently two forms of armor available in industry: hard and soft armor. When surveying the research community, one instance of active armor was found. The Institute of Soldier Nanotechnology at MIT is currently developing active vests utilizing MR Fluids as the active stiffness mechanism. Although using a similar strategy, there are a couple fundamental differences between the two concepts, including the (1) use of high aspect ratio particles in the current design versus spherical in the MIT concept and (2) use of an electrical activation method versus a magnetic. Even still, one could argue that these concepts are indeed similar. Even if this is the case, novelty is measured using a relative number of instances found in the universal world of ideas. Given that, both ideas are deemed novel when looking at the lack of total concepts found for active armor. It should also be noted that using the biological strategy of controlled association, many more novel strategies can be generated from the micro to macro level of system development.

7.3 CLOSURE AND VALIDATION

Currently, there is little empirical evidence as to the effects of bio-inspired techniques in the idea generation process. The aim of this work was to quantify the value of these techniques in ideation as it relates to design space expansion and exploration. The research question addressed in this chapter is as follows:

"What is the impact of biological strategies in the conceptual design process?"

With this question, this research seeks to assess the value of bio-inspired design in the conceptual design process. In this research, the value is assessed using two different contexts: (1) problem-based Conceptual Design, where the designer seeks to be inspired by biological strategies in the ideation process and (2) solution-driven Conceptual Design, where the designer seeks better solutions to engineering problems by mimicking the biological strategies. In this chapter, the value of biological strategies and the proposed approach for problem-based Conceptual Design are assessed.

Specifically, in the problem-driven context, it was hypothesized that (Hypothesis 4a) exposure to biological design examples in ideation will increase the novelty of design ideas generated and (Hypothesis 4b) exposure to biological design examples in ideation will increase the variety of design ideas generated. In Section 7.1 two experimental studies were presented in which senior and graduate mechanical engineering students were exposed to biological examples in the idea generation process, and these results were compared to participants receiving no examples and to those receiving human-engineered examples. Exposure to biological examples was found to increase the novelty of design ideas generated after exposure without decreasing the variety of design ideas generated. In Section 7.2, a comprehensive example of the design of hybrid, bullet resistant armor was presented. In this study, the designs generated using the problem-based approach were found to be novel relative to other solutions currently found in industry.

The results of the work presented in this dissertation have a number of implications in engineering design. One of the primary goals of engineering design is to discover new and innovative solutions to encountered problems. In the development of these solutions, idea generation is key. Dylla has demonstrated significant correlation between the amount of design space considered in idea generation and the quality of the final design [1]. It follows that methods for aiding designers in expanding and exploring their design space should yield better designs. The results show that search strategies that include biological examples help expand the design space of the designer, while also negating the negative effects on variety typically seen with exposure to examples. These

results have been found for senior and graduate mechanical engineering students, but should be extendable beyond the current sample group.

Empirical Performance Validity

The validation strategy in this dissertation is displayed in Figure 7.10.

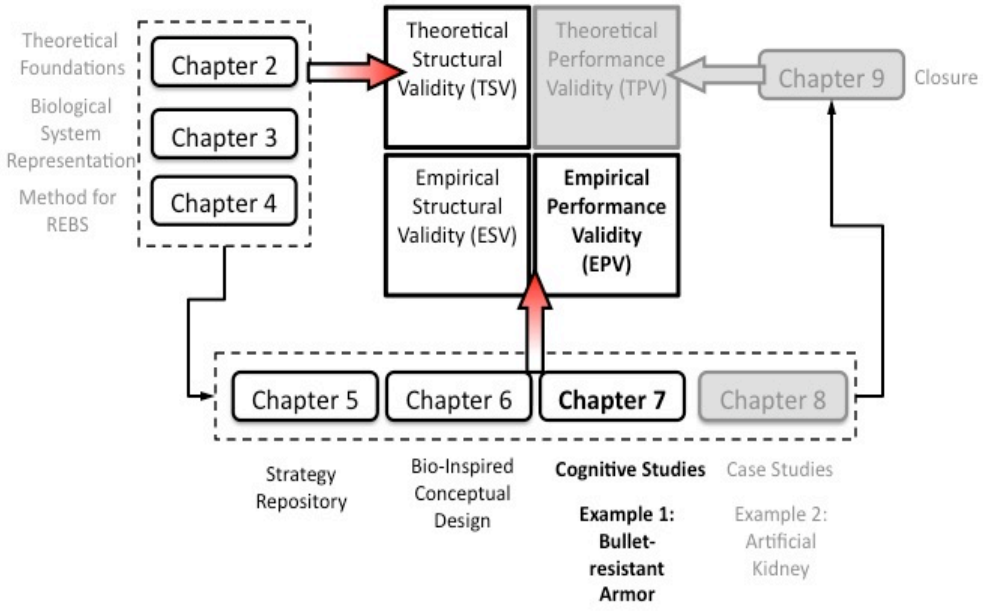


Figure 7.10 Validation Strategy and Chapter 7

Empirical Performance Validation involves accepting the usefulness of the method for some representative example problems. In Section 7.1, cognitive studies were used to show the value of biological strategies in problem-based Conceptual Design. In the cognitive studies, participants generated design ideas that were more novel after exposure to the biological strategies than before. With respect to variety, there was no significant difference found in the variety of design ideas before and after exposure to the biological strategies. Therefore, the proposed approach, which utilizes biological strategies to aid in idea generation is deemed useful. In Section 7.2, a comprehensive example of the design of a hybrid, bullet resistant armor system using the problem-based approach was presented. In this example, the design generated was found to be novel compared to current protection systems found in industry, thus showing usefulness in the proposed approach.

CHAPTER 8 SOLUTION-DRIVEN BIO-INSPIRED CONCEPTUAL DESIGN

In the solution-driven approach, the designer begins with a biological solution and attempts to mimic some novel feature or behavior of this system through engineering design. The specific goal in this approach is to develop innovative solutions by reverse engineering biological solutions. In this research, the value of biological strategies in this context is assessed. In this Chapter 7, the impact of biological strategies on the Conceptual Design process in the problem-based context was assessed. In this chapter, the impact of these strategies in the solution-driven context is explored. The dissertation plan is displayed in Figure 8.1.

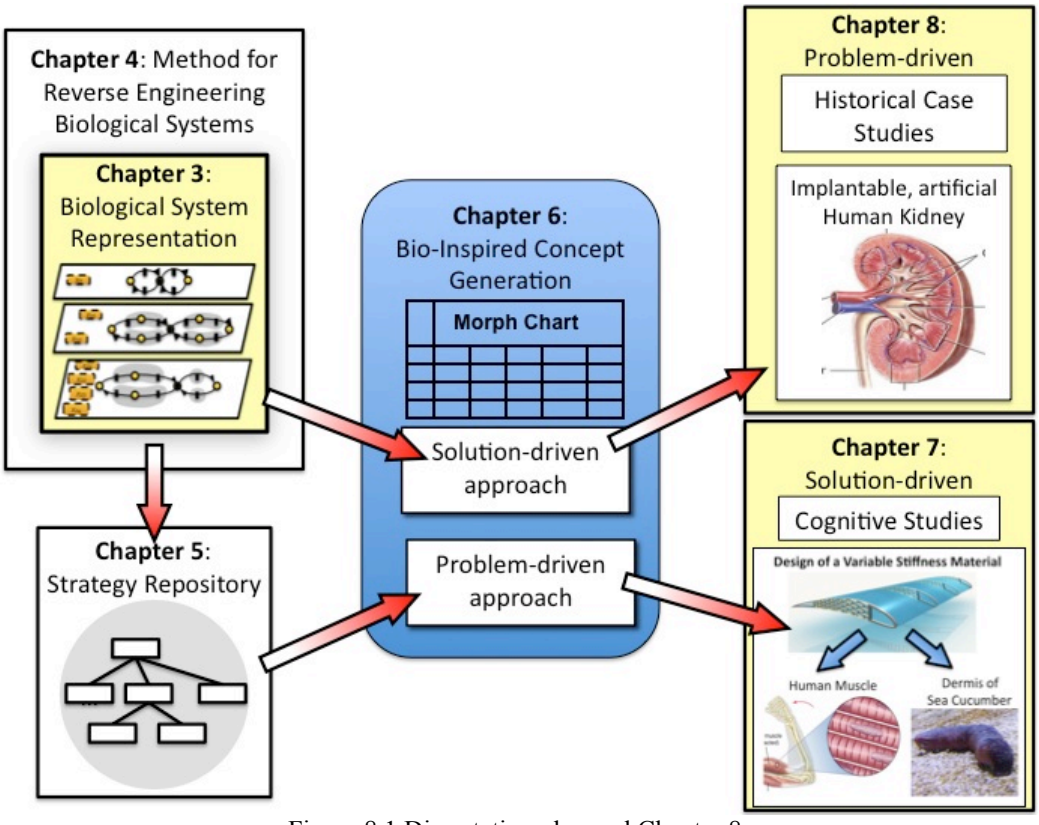


Figure 8.1 Dissertation plan and Chapter 8

Continuing from Chapter 7, the following research question is addressed in this chapter:

RQ4: "What is the impact of biological strategies in the conceptual design process?"

To answer this question, it is hypothesized in Chapter 1 that:

Hypothesis 4c: Bio-inspired engineering systems possessing a deeper level of biological system behavior will perform better than those possessing superficial behavioral similarities.

In essence, the value of rich behavioral descriptions in bio-inspired design is assessed with this hypothesis. To validate this hypothesis, historical case studies on bio-inspired systems are presented (Section 8.1). In Section 8.2, a case study of the solution-driven approach to Bio-inspired Conceptual Design is present. In this study, the proposed approach is used in the development of a novel renal replacement therapy.

8.1 CASE STUDIES

In Chapter 7, the value of biological strategies in the ideation process was explored with respect to the novelty and variety of design ideas generated. In this chapter, the value of rich behavioral descriptions in bio-inspired design is explored. Specifically, it is hypothesized that bio-inspired engineering systems possessing a deeper level of biological system behavior will perform better than those possessing superficial behavioral similarities. In Section 8.1.1, this hypothesis is tested in the field of aircraft flight control. In Section 8.1.2, this hypothesis is tested further in the field of renal replacement therapy.

8.1.1 Avian Flight

The adaptability and control in avian flight has intrigued engineers for centuries. The morphing ability of the wings over varying conditions has been a primary source of

interest. In this case, we explore control in avian flight and its varying levels of impact on engineering innovations in the area of aviation.

8.1.1.1 Avian Flight Control

An annotated figure of the bird wing is displayed in Figure 8.2. The primary load bearing structures of the bird wing are a series of interconnected bones, much in the same fashion as the human arm [165]. This structure consists of the upper and lower arm bones, and the hand bone. Several joints between these bones allow the range of motion of flight, including the shoulder, elbow, and wrist joint. The skin and feathers, attached to the bone structure, produce the aerodynamic shape necessary for flight [165]. The main feathers of the bird wing used in flight include the primaries, secondaries, alula, and coverts. The primaries are connected to the hand bone and can be individually controlled. The seconaries are connected to the lower arm bone, while the alula feathers are attached to the bird's thumb. These feathers can be rotated to aid in slower flight. The covert feathers cover the secondary and primary feathers and help smooth airflow over the wings.

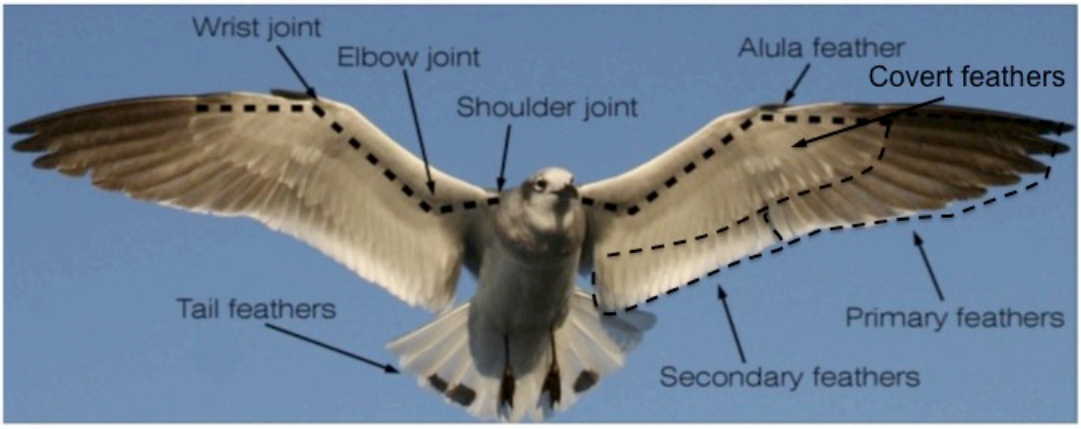


Figure 8.2 Skeletal and feather feathers of the bird wing [165]

For control in varying flight conditions, the bird has the ability to change the shape of its wing, both in planform and profile view. Examples of the large range of configuration for the Bald Eagle is displayed in Figure 8.3.

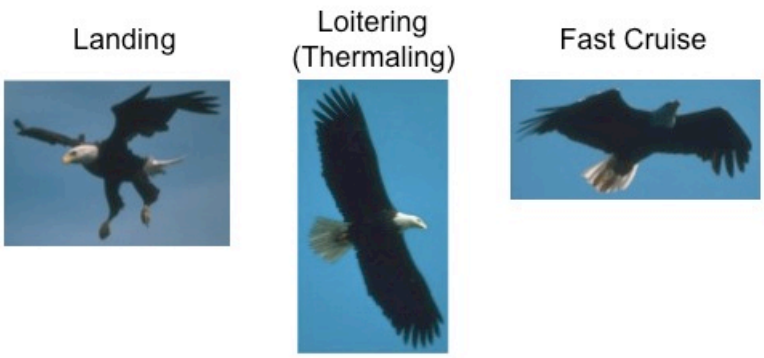
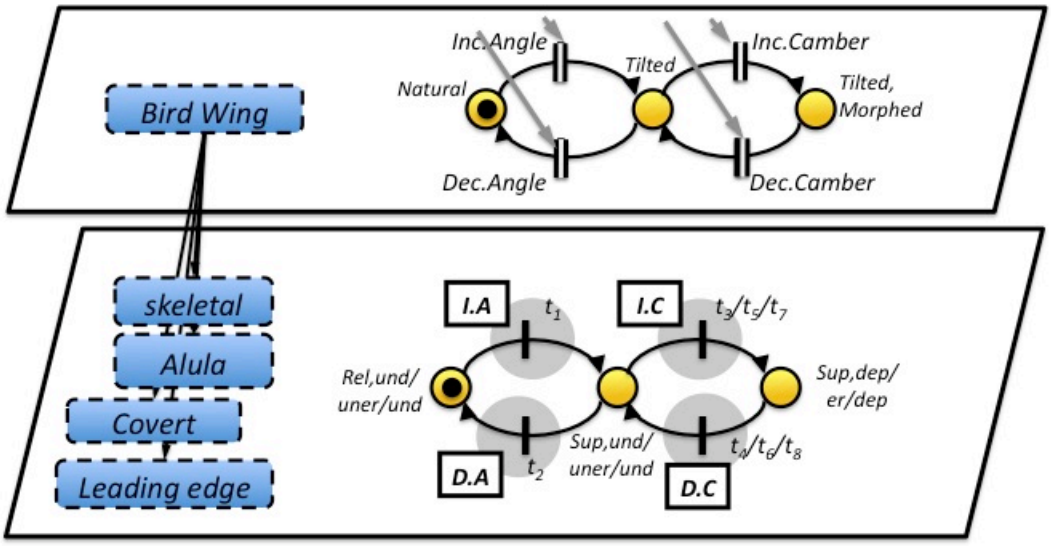


Figure 8.3 Bald Eagle in various flight configurations [166]

Specifically, one of the main problems of flying animals is the manipulation of lift while maneuvering and in unfavorable flight conditions such as landing and take-off. To increase lift in these conditions, birds can manipulate the surface of the wing, such as increasing the angle of incidence and/or the camber, or curvature, of the wing [167]. However, one of the limits on the camber and angle of incidence of the wing is stall. When this point is reached, the airstream separates from the wing's upper surface, producing a sudden fall in lift and increase in drag of the wing. To generate lift and overcome this stall condition, birds alter several features of their wings, including the leading edge, alula, and covert feathers. The alula feathers are projected on the leading edge to direct wind over the wing, which prevents stall at high angles of incidence. The covert feathers, under turbulence, also rise to prevent flow separation for high lift. Birds have been also shown to project feathers on the leading edge of its wings to increase the camber of the wing [168]. The tails of certain birds, such as swifts, swallows, and fork-tailed falcon, also act in conjunction with the wing to increase the overall camber and increase the overall area of the wing [167].

A hierarchical Petri net model for the flight control behavior of the bird wing is displayed in Figure 8.4. As seen in the figure, the bird wing has three physical states with respect to flight control: natural, tilted, and tilted/morphed. In the natural state, the wing has the incidence angle and camber most efficient for soaring. To increase lift in either or both wings, the bird increases the angle of incidence of the wing, moving into the tilted state of the wing. For additional lift in unsteady conditions such as takeoff and landing, the bird increases the camber of its wing, causing the tilted/morphed state of the wing.

The behavior of the wing can also be viewed at a deeper level of abstraction. In the natural state, the skeletal components are in a relaxed state, the alula and leading edge feathers are undeployed, and the covert feathers are unerect. To increase lift, the bird supinates its wrist, causing the tilted state of the wing. To further increase lift when the critical angle for stall is reached, the alula and leading edge feathers deploy and the covert feathers erect. This increases the camber of the wing and brings the state of the wing to a tilted and morphed state.



t_1	Skeletal (supinate wrist)	t_2	Skeletal (relax wrist)
t_3	Alula (if ang≥$ang_{critical}$, deploy)	t_4	Alula (if ang≥$ang_{critical}$, relax)
t_5	Covert(if ang≥$ang_{critical}$, erect)	t_6	Covert(if ang≥$ang_{critical}$, relax)
t_7	L. Edge (if ang≥$ang_{critical}$, deploy)	t_8	L. Edge (if ang≥$ang_{critical}$, relax)

Figure 8.4 Hierarchical Petri net model of the flight control mechanisms of the bird wing

Using the method presented in Chapter 4, the following strategy can be extracted from the model:

Strategy(*Natural, Tilted/morphed*) = Bird Wing (*increase angle, increase camber*)

In natural language, the strategy is as follows:

"Lift is controlled by tilting and morphing the wing"

The behavior of the wing can be expanded to also reflect how the behavior of its components contribute to its behavior. The expanded strategy can be extracted as follows:

Strategy (*Natural, Tilted/Morphed*) = Skeletal (*supinate wings*), if ang≥$ang_{critical}$ (alula (*deploy*)/covert (*erect*)/leading edge feathers (*deploy*))

In natural language, this strategy reads:

"Lift is controlled in the wing by first supinating the wing to increase the angle of attack. As the angle approaches the critical angle for stall in the wing, the bird then increases the camber of the wing by deploying the leading edge and alula feathers. The covert feathers also become erect to direct flow over the wing."

In the following section, we present specific innovations in flight control in modern aviation.

8.1.1.2 Brief history of control surfaces in flight control

Figure 8.5 displays a brief history of innovations in control and manipulation in flight over time.

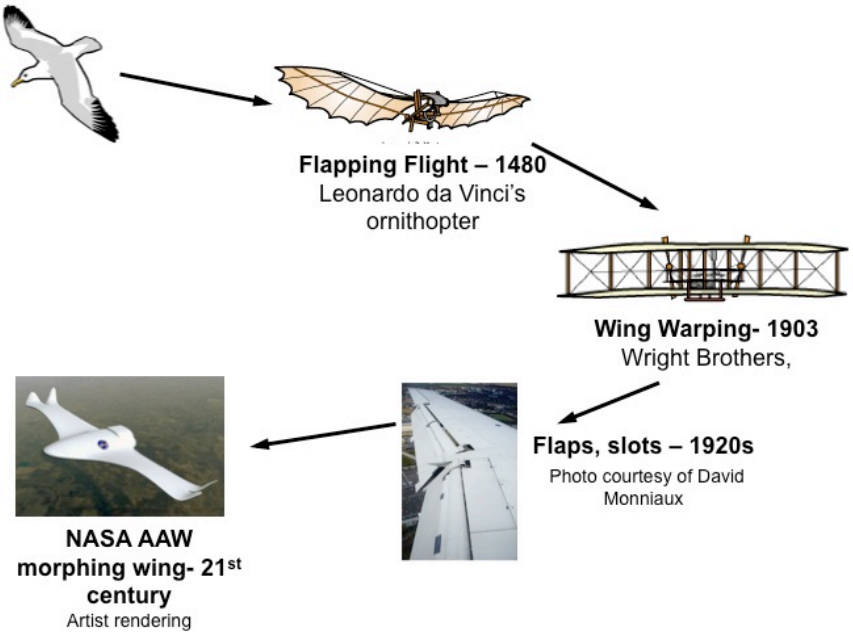


Figure 8.5 Brief history of flight

Beginning in the 1500s, Leonardo da Vinci started devising ways to mimic flight by observing birds. Attempts at flight were rather unsuccessful until the idea of wing warping by the Wright brothers in the early 1900s. Building on the idea of flight control using wing warping, engineers further devise ways of manipulating lift through the use of wing flaps and slats. Scientists at DARPA and NASA have further devised methods for control using smooth wing morphing during flight. These key innovations will be discussed in the following sections.

In this study, we look at several key leaps in innovation that significantly impacted the control of the aircraft. We then compare these key innovations to the behavioral strategies extracted from the control mechanisms in bird flight. Lastly, we correlate this similarity to the performance of the aircraft.

The progress and failures of Pre-20th century flight

Humans attempting to imitate natural flight can be traced back to 8 B.C. with the ancient Greek legend of Daedalus and Icarus. In the 16th century, during the Italian Renaissance, Leonardo da Vinci designed several flying machines, called ornithopters (Greek for "bird" and "wing"). Leonardo di Vinci based his machines on the flapping wings of birds, believing that the body could power and control the aircraft.

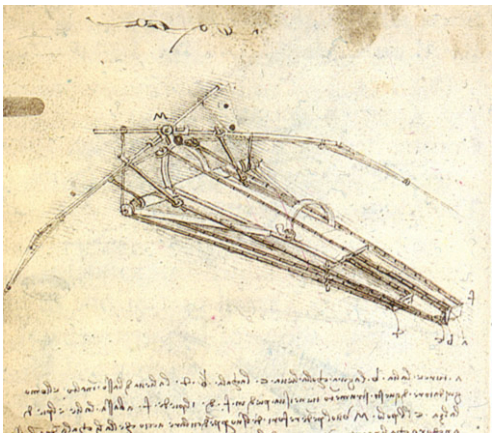


Figure 8.6 Ornithopters

Following da Vinci's vision, in the 18th century, there were many designs relying on the shape of birds, including "Passarola", or "Great Bird", designed by Father Laurence de Gusmao. During this century, there were also many fictional stories

adopting the view that flight was simply a matter of wings, including "A Dissertation on the Art of Flying" by Samuel Johnson. At the end of this century, in 1799, George Cayley defined the problem of heavier-than-air human flight as "to make a surface support a given weight by the application of power to the resistance of the air". In defining the forces of lift, drag, and thrust, he was the first to distinguished between the mechanism for lift (wing) and thrust (assisters) [169].

The 19th century was the period of the glider. In 1804, Cayley designed and tested his first hand-launched glider [169]. This design gave idea to the configuration of the modern airplane; Cayley's design consisted of a wing, fuselage, empennage, rudder and elevator. Building on the glider concept, two people made significant contributions to modern, human-powered flight: Horatio Phillips and Otto Lilienthal. Phillips tested several cambered airfoil shapes in a wind tunnel for aerodynamic performance, including that of the rook. Lilienthal, famous for the book "*Bird Flight as a Basis for Aviation*", studied the flight of birds and constructed aircraft based on his studies. Through his experiments, Lilienthal contributed greatly to the advancement of the glider modern flight [169]. However, Lilienthal faulted in his fixation on previous ornithopter designs to power flight, which were limited as they relied on human power. Octave Chanute, a civil engineer, also experimented with flight during this period and advanced the science of flight through his work with gliders [169].

During the period prior to the 20th century, human flight had progressed from di Vinci's human-powered ornithopter to the level of the stable glider. Many inventors explored with the idea of flight based on observing bird flight. Early in the history of flight, inventors became fixated on flight as merely a flapping of wings. While the flapping powered flight, the key to flight at this level, as discovered by Cayley, was the balance of lift, drag, weight and thrust. Driven by Cayley's airfoils, much attention was paid to the shape of the airfoil and maximizing the lift for stable flight, while not paying much attention to dynamic control in flight. This stable flight led to flying machines that would "proceed on a straight and level course with the pilot intervening only when a change in direction or altitude was required" [170].

With respect to successfully imitating the bird flight, leading up to the 20th century, flight had only progressed to mimicking of the static behavior and form of bird wings to produce lift. Although incremental advances were made to the glider, the engineers and scientists of this day had yet to understand the importance of control and maneuverability to sustained flight.

The Age of the Wright Brothers

The first key innovation in flight control came with the Wright Brothers and their method for producing differential lift in the wings of a glider for roll control. During the 20th century, after taking interest in the works of Lilienthal, the Wright brothers, Otto and Wilber Wright, began to study flight. At this time, the Wright brothers recognized the true problem of modern flight, that of control (stability vs. maneuverability). Wilber, an avid bird watcher, wrote in 1900 to Octave Chanute, "My observation of the flight of buzzards leads me to believe that they regain their lateral balance when partly overturned by a gust of wind by a torsion at the tips of their wings. If the rear edge of the right wing is twisted upward and the left downward, the bird becomes an animated windmill and instantly begins to turn a line from its head to its tail being the axis…..In the apparatus I intend to employ and make use of the torsion principle" [10]. In watching birds, the Wrights found that birds controlled roll in flight changing the angle and shape of each of its wings to produce differential lift. This asymmetry in lift would cause rotation about the centerline of the birds. The Wright brothers understood that by controlling the wing, they could control and maneuver the plane easier. This idea was the first to provide the much needed dynamic control over the lift of the wing, allowing increased steering and maneuverability. The Wright brothers' 1903 *Flyer* is displayed in Figure 8.7.

Figure 8.7. Wright brothers' 1903 Flyer

The period through the 1920s was regarded as the period of strut-and-wire biplanes, which built upon the Flyer of the Wright Brothers. From 1905 until the end of WWI, no significant advances in flight control were made. Towards the latter part of this period, these biplanes utilized rigid wings, thus wing warping was limited. Instead, these designs utilized ailerons, or small wing flaps, for roll control in rigid wing aircraft. The period that followed, the era of the mature propeller-driven airplane, was a period of improvement.

High lift surfaces and increased control

The second key innovation in flight control involved more control over the wing surface. During the 1930s, wind loading (weight of the plane/ wing planform area) almost quadrupled. This was mainly due to the fact that airplanes flew faster and were able to generate more lift. With the higher wing loading came higher required takeoff and landing speeds. To allow planes to fly slower during these times for balance, planes utilized several features such as flaps, which increased the lift of the airplane. These flaps were designed directly after that of ailerons, but both were simultaneously deflected in the same direction to increase the camber of the airfoil and provide lift [171]. To increase lift while also preventing stall (or sudden reduction in lift), development continued on advancing control surfaces, including slots, spoilers, leading and trailing edge flaps, and slats. An annotated diagram [172] of these high lift surface features is displayed in Figure 8.8.

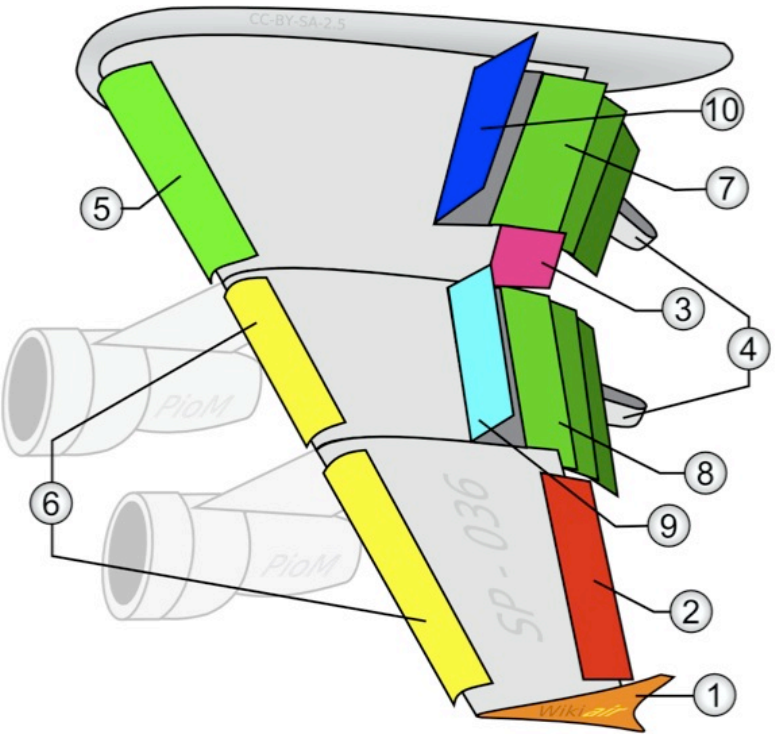


Figure 8.8 High-lift features – (1) Wingtip, (2) Low Speed Aileron, (3) High Speed Aileron, (4) Flap track fairing, (5) Krüger flaps, (6) Slats, (7) Three slotted inner flaps, (8) Three slotted outer flaps, (9) Spoilers, and (10) Spoilers Air-brakes [172]

More recently (1985-1988), as part of the NASA Ames Mission Adaptive Wing (MAW), scientists developed a smooth camber control to reduce drag produced by discontinuous surfaces. The MAW wing had an internal mechanism to flex the wing and produce optimal camber configurations for different flight conditions [173]. Even more recently, FlexSys Inc. [174], used compliant mechanisms to achieve smooth shape change of the leading and trailing edge flaps. With their hinge-less, smoothly contoured control surfaces, FlexSys demonstrated high actuation rates, large deflections, and large shape variability.

8.1.1.3 Performance assessment and Validation of hypothesis

In this study, we look at the role of key innovations in wing surfaces, and control over these surfaces, in the advancement of aerodynamic performance in flight. We look at how these innovations have led to increased control of aerodynamic parameters, such as the lift coefficient. The lift coefficient, C_L , can be defined as a characteristic of the cross section of the airfoil and the angle of attack of the wing. C_L helps define both the

lift generated over varying flight conditions and the maneuverability of the aircraft. Controlling the shape of the airplane wing in flight allows us direct manipulation of the lift coefficient.

Manipulation of lift is particularly important for stability at low flight speeds, such as in takeoff and landing, and for maneuverability of the aircraft. Low flight speeds are desirable at take-off and landing. However, at low flight speeds, the air over the wing becomes turbulent and causes a sudden decrease in lift (stall). Control over the shape (and C_L) of the airfoil can increase the lift of the wing and delay the onset of stall, thus allowing lower flight speeds for takeoff and landing. A low turning radius is important for maneuverability of the aircraft. Control over the shape of the wing helps to reduce the turning radius of the aircraft. The lift coefficient is inversely proportional to turning radius, thus increasing the lift coefficient reduces the turning radius of the aircraft. Therefore, since a high lift coefficient of the wing is desired for flight stability and maneuverability, the $C_{L,\,max}$ of aviation technologies will be compared to their level of biological similarity in this study.

Our hypothesis states that bio-inspired engineering systems possessing a deeper level of biological system behavior will perform better than those possessing superficial behavioral similarities. Specifically, in this section, we make the claim that deeper behavioral similarity between the flight mechanics of the bird wings and the control mechanisms developed in aviation can be correlated to increased performance in lift control, $C_{L,\,max}$. The amount of lift control (% change in lift coefficient) allowed by the key innovations in aviation is displayed in Figure 8.9.

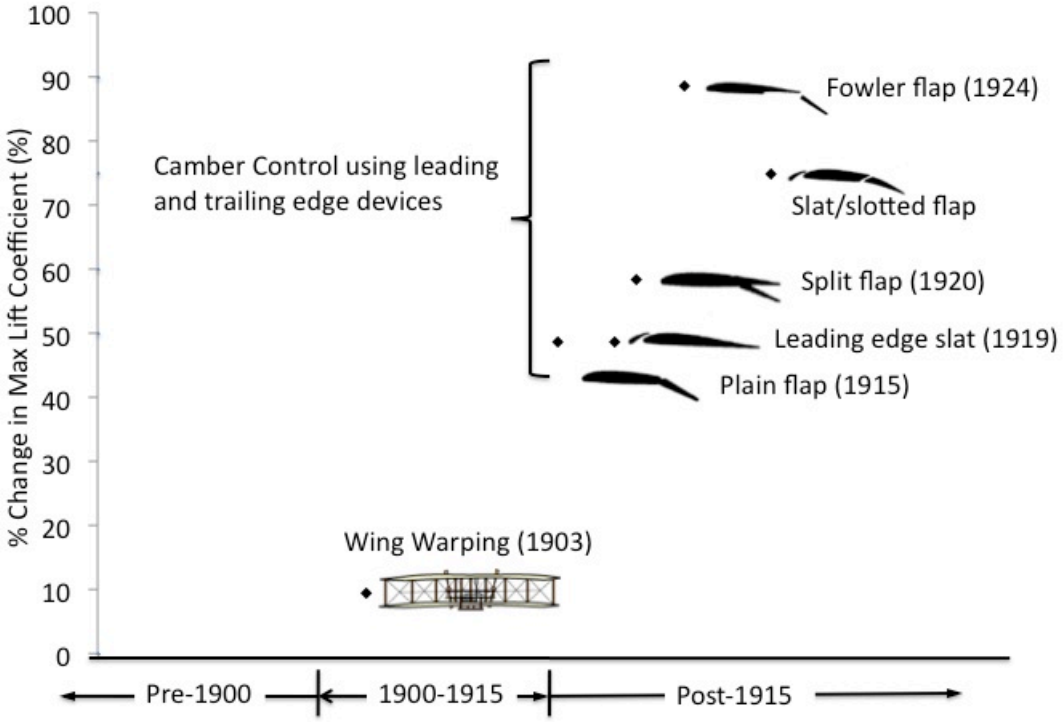


Figure 8.9 Key innovations in aviation (values used were derived from typical max lift coefficient for airfoil shapes [175])

These key innovations and advances in performance can now be correlated to the level of behavioral similarity to that of the bird wing. In Table 8.1, the correlation between the level of avian behavioral strategy used and the performance (amount of lift control) is presented.

Table 8.1 Summary Table for Aviation Case

Level of Decomposition - *Strategy*	**% Change in Max Lift (%)**	**Description**
Level G(-1) – *Wing shape providing lift*	0	Pre-1900 – Aviation focused mimicking of the static behavior and form of bird wings to enable flight. This led to severals glider design that were used to produce lift, but not sustained flight.
Level G0 - *"Lift is controlled by tilting and morphing the wing"*	10	1900-1915 - Wing warping was developed by the Wright Brothers after observing the means by which birds controlled the shape of their wings. Wing warping allowed more stable flight by affording the ability to manipulate lift by up to 10%.

Table 8.1 Continued

Level of Decomposition - *Strategy*	**% Change in Max Lift (%)**	**Description**
Level G1 - *"Lift is controlled in the wing by first supinating the wing to increase the angle of attack. As the angle approaches the critical angle for stall in the wing, the bird then increases the camber of the wing by deploying the leading edge and alula feathers. The covert feathers also become erect to direct flow over the wing."*	50-100	Post-1915 – By observing and utilizing the control features of the bird wing, engineers developed high lift control surfaces, such as ailerons, flaps, and slats, to enable increased lift manipulation in the aircraft. This allowed sustained flight and control in flight in varying flight conditions, and thus better flight performance.

As seen in Table 8.1, performance with respect to lift control (maneuverability) in aviation increases as the level of behavioral similarity increases. Mimicking the overall form and static behavior of bird flight yielded only the development of static gliders, which were not able to sustain flight. Using the behavioral strategy of tilting and morphing the wing for control of lift (from studying birds in flight), the Wright brothers were able to achieve a nearly 10% increase in lift through wing warping. Wing warping, the first key innovation in flight control, allowed control over roll about the centerline of the plane and high radius turning. This control was significant as it has been attributed as the key innovation enabling sustained flight.

As one goes deeper into the behavioral strategy of the bird wing, additional parallels can be drawn. The richer biological strategy of controlling lift in unfavorable conditions by controlling the camber of the wings surface through deployment of special feathers on the wing's surface, such as the alula, leading edge, and covert feathers. This strategy is used in situations such as take-off and landing. This strategy of controlling the specific control surfaces of the wing, as opposed to simply wing warping, is used in current aviation technologies and has enabled truly sustained flight and maneuverability in flight. Specifically, in the post-1915 era, wing warping was replaced by ailerons on the trailing edge of the wing. Larger flaps and slots offered even more control over the

surface of the wing, allowing the angle of incidence as well as camber of the wing to be morphed. These led to increased angle of incidences achievable without stall, and increased aircraft control for maneuvering and take-off and landing.

As seen in the aviation case, advances in flight depended strongly on the understanding of how birds fly. Wing designs possessing deeper level of similarity to the actual behavior of the bird wing perform better with respect to manipulating lift. This supports our hypothesis. From this, we can realize the value of having richer behavioral models of biological systems, which aid in the understanding of biological technologies. In section 8.1.2, further support is provided for our hypothesis by looking at renal replacement therapies.

8.1.2 Renal Replacement Therapy

In this case, we explore key innovations in the field of renal replacement therapy. All living organisms must constantly regulate their condition in order to maintain life. The human kidneys play an important role in this homeostasis by providing several functions that help to maintain a healthy balance inside the body. The kidneys perform the following primary functions: (1) removal of metabolic waste products and foreign substances from the plasma, (2) regulation of plasma ionic composition, (3) regulation of plasma osmolarity, (4) regulation of plasma volume, and (5) regulation of plasma pH. While the primary functions of the kidney include removal of wastes and regulation of the plasma, it is also responsible for secreting hormones and enzymes. The kidney is a part of the urinary system, which consists of two kidneys, two ureters, the urinary bladder and the urethra. Blood is supplied to the kidney by the renal arteries, and the clean blood exits the kidney through the renal veins. The waste cleared from the blood exits as urine. This urine then flows through the ureters to the bladder, where it is stored until it is excreted.

8.1.2.1 The Human Kidney

The basic functional unit of the human kidney is called a nephron (displayed in Figure 8.10). It is composed of five main parts: the renal corpuscle, the proximal tubule, the loop of Henle, the distal tubule, and the collecting ducts. In these parts, the kidney

utilizes four main functions to provide homeostasis: filtration, reabsorption, secretion, and excretion. First it filters most substances across the glomerulus (Figure 8.10), a collection of capillaries surrounded by the Bowman's capsule, which together form the renal corpuscle. There are many factors affecting glomerular filtration, including: (1) size and charge of the molecules being filtered, (2) Size of the filtration slits of the glomerulus and charge of the glomerular basement membrane, and (3) several hemodynamic factors. Hemodynamic features include blood flow, convection, diffusion, the glomerular capillary pressure difference, and the Bowman's capsule pressure difference [176].

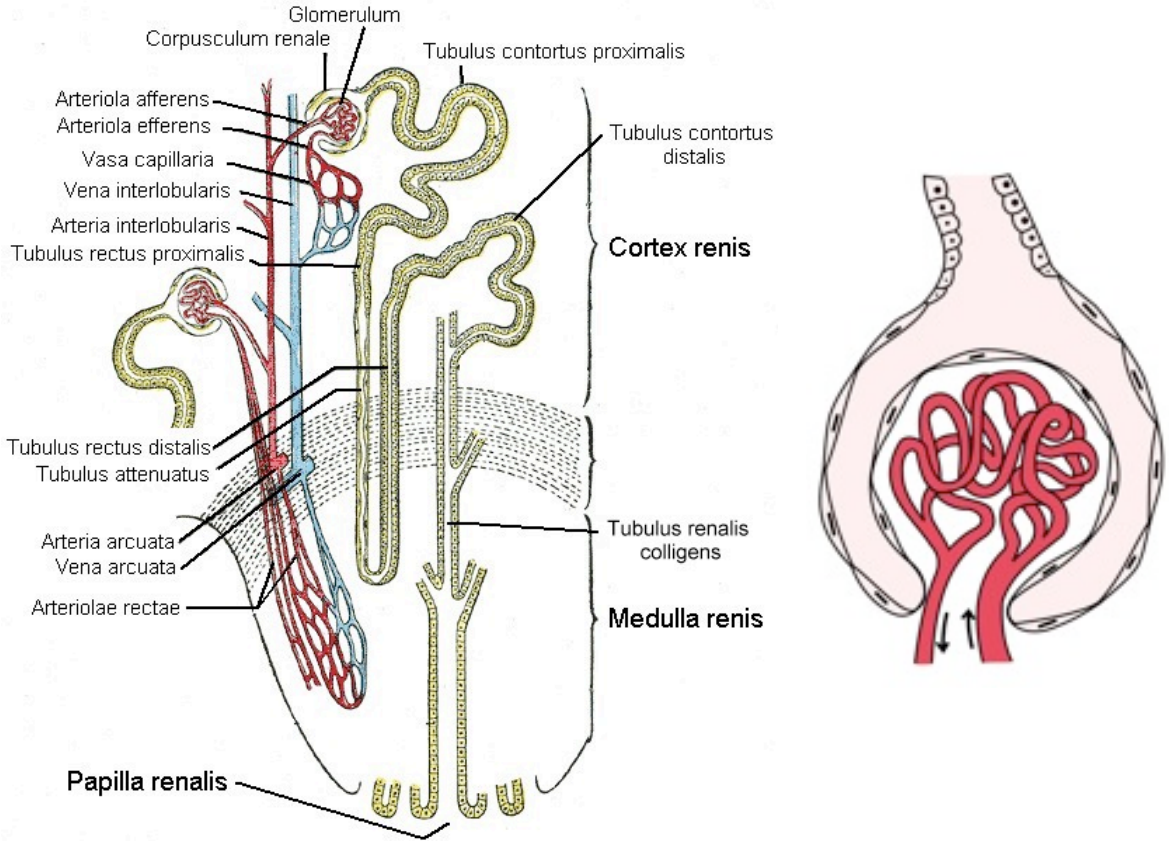


Figure 8.10. A complete nephron is shown on the left. To the right, the dark-red glomerulus is surrounded by the pink Bowman's capsule, forming the renal corpuscle

Next, important substances are reabsorbed back into the blood stream at the proximal tubule, loop of Henle, the distal tubule, or collecting ducts. Lastly, substances that were not filtered across at the glomerulus can be excreted from the kidney at one of the later stages.

Approximately 625 mL of plasma flows through the kidney every minute. Of this 625 mL, the kidney filters approximately 20%, or 125 mL, across this membrane every minute. However, only 1mL/min of urine is excreted. This discrepancy between the filtered and excreted amounts is accounted for by reabsorption and secretion. After removing so much from the blood, the human kidney then works to reabsorb back into the blood those substances that are beneficial for homeostasis. As an example, all glucose found in the blood in the renal artery is filtered out of the blood in the renal corpuscle. However, glucose is vital for homeostasis, so it is immediately reabsorbed in the proximal tubule.

A hierarchical Petri net model of the kidney for removal of waste from the blood is displayed in Figure 8.11. The generation of this model is detailed in Section 8.2. The model of the kidney displays how the composition of the blood (represented as numerical tokens) changes with respect to the different kidney processes. The PN arcs represent the different rates at which the composition of the blood changes.

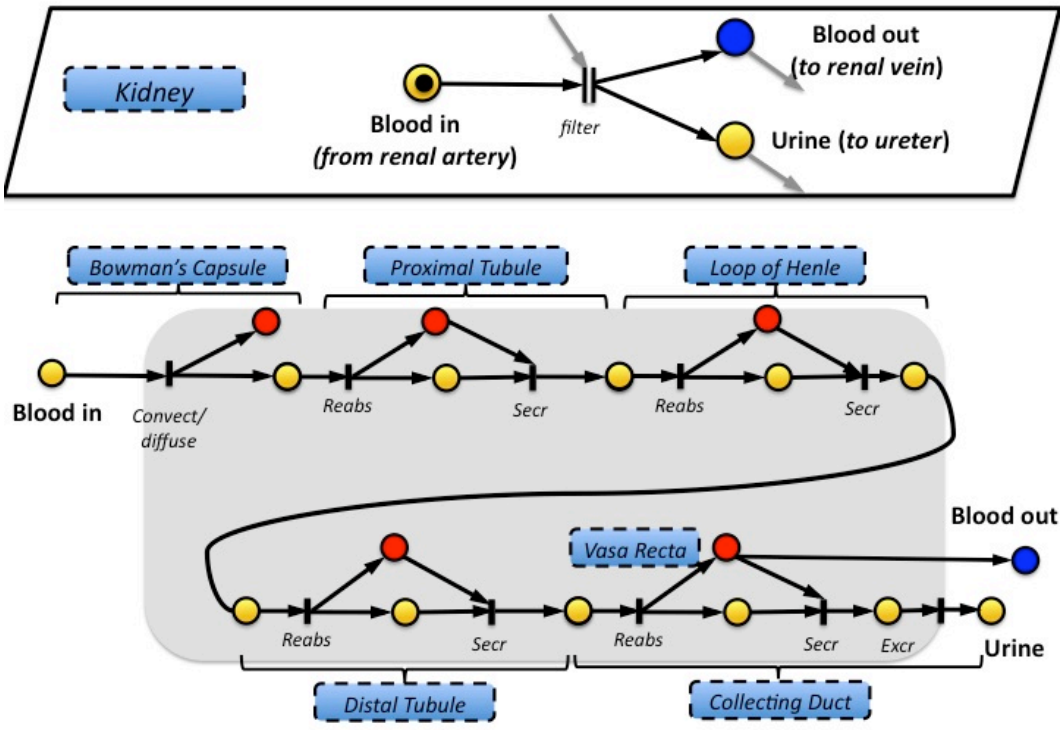


Figure 8.11 Hierarchical Petri net model of waste removal in the kidney

As seen in Figure 8.11, at the kidney level of abstraction, dirty blood comes in from the renal vein and wastes are filtered out into the urine and the remaining clean blood returns to the body through the renal artery. The subsystem level of decomposition includes the Bowman's capsule, Proximal Tubule, Loop of Henle, Distal Tubule, and the Collecting Duct. At this level, the blood plasma and its solutes are convected and diffused across the Bowman's capsule. The solutes in the plasma then go through several steps of reabsorption and secretion, finally being excreted through the collecting duct as urine.

Based on the hierarchical net in Figure 8.11, the strategy for the kidney is extracted as follows:

Strategy($Composition_{iniital}$, *composition 1/composition 2*) = (Kidney (*filter*))

In natural language, the strategy of the kidney is as follows:

"The composition of blood in the kidney is modified through filtration"

The strategy of the kidney can now be expanded to reflect the behavior of the glomerulus, nephron, and ureter as follows:

Strategy($Composition_{iniital}$, *composition 1/composition 2*) = *BowCap* (Conv./Diff.), *PrTub* (Reabs/Secr), *LHen* (Reabs/Secr), *DisTub* (Reabs/Secr), *ColDuct* (Reabs/Secr/Excr)

In natural language, the strategy of the kidney is as follows:

"Filtration in the kidney is performed by removing mostly all substances from the blood through convection/diffusion in the Bowman's Capsule and reabsorbing and secreting needed substances in the Proximal tubule, Loop of Henle, and Distal Tubule, and Collecting Duct. The remaining solutes are excreted through the Collecting Duct."

In the following section, the development of several renal replacement therapies is reviewed.

8.1.2.2 Artificial Kidney Development

In this section we focus on the development of two key forms of renal replacement therapy, including: (1) hemodialysis and (2) hemodiafiltration. These renal replacement therapies are discussed in the following sections.

Hemodialysis

The first major innovation in treating kidney desease came in the 1940s. Williem Kolff is credited with constructing the first hemodialysis machine in 1943. His artificial kidney utilized blood flowing through cellophane tubing in a rotating drum assembly. This drum rotated in a tank of dialyzer medium [177]. In the 1950s, Kolff's invention was advanced enough to solve the problem of acute renal failure. It was not until the works of Belding Scribner that a solution for chronic end stage renal disease was envisioned. Scribner devised an idea of using Teflon tubes inserted into the artery and veins. After treatment, these tubes were connected using a U-shaped device, completing the circulatory circuit. This advancement allowed direct access to the circulatory system and ended the need for making incisions every time. Although several incremental advances have been made with traditional hemodialysis, the basic ciruit and principle has remained the same.

In hemodialysis, displayed in Figure 8.12, solutes are filtered by diffusion across a semipermeable membrane. The primary component of modern hemodialysis is the dialyzer (labeled 'filter' in Figure 8.12), which contains the semipermeable membrane. In the dialyzer, blood from the patient flows along one side of the membrane countercurrent to that of the dialysate, creating a concentration gradient across the membrane. The general principle of hemodialysis is that small molecules will diffuse across the membrane to areas of lower concentration. The concentration of solutes to be filtered is zero, while concentrations of those solutes to be kept in the blood is equal to that of the blood. Based on the laws of diffusion, the larger the molecule, the slower the rate of transfer across the membrane will be. Given that, molecules of small molecular weight, such as urea (60 dA), are filtered efficiently, while molecules with higher molecular weights, such as creatinine (113 daltons), less efficiently[178]. The waste, which is filtered across the membrane into the dialysate, is disposed of with the dirty dialysate.

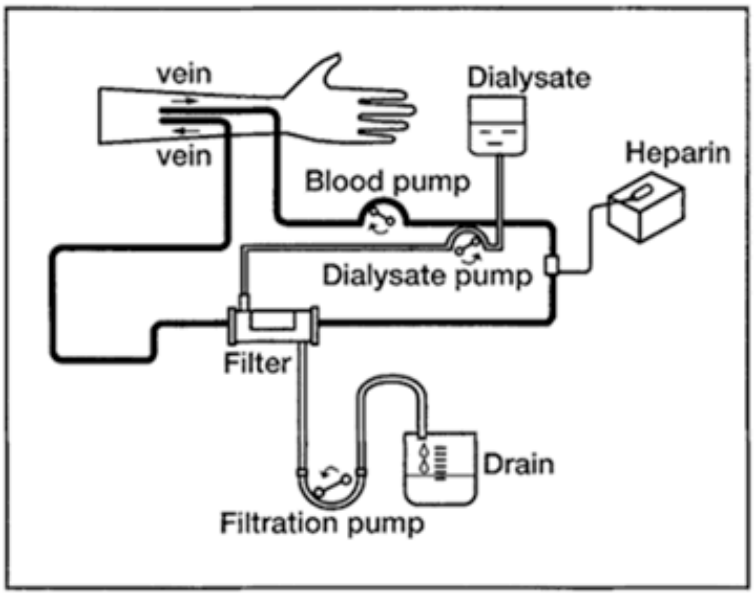


Figure 8.12 Hemodialysis circuit [179]

Hemodiafiltration

Another key innovation came by way of combining hemodialysis with a convective renal replacement therapy known as hemofiltration. This hybrid process is called hemodiafiltration. In the 1960s, hemofiltration was introduced to enhance the removal of larger substances from the blood and improve hemodynamic tolerance [178]. In the 1970s, building on benefits of hemofiltration, work began on creating a renal replacement therapy harnessing the benefits of hemodialysis (small molecular weight substance removal) and hemofiltration (middle molecular weight solute removal). Hemodiafiltration works in a similar fashion to the human kidney; a large amount of filtrate is filtered from the blood and then desirable components are replaced. In hemodiafiltration, blood and dialysate are pumped through the filter in a counter-current manner. Similar to that of glomerular filtration, water and substances up to a molecular weight of 20,000 Da are convected and diffused across the membrane and into the dialysis fluid. Desirable substances are then replaced in the distal part of the hemodiafiltration circuit using a replacement fluid. The typical composition of the replacement fluid is displayed in Table 8.2.

Table 8.2 Hemodiafiltration replacement fluid

Component	Value (mmol/L)
Sodium	140
Potassium	0-4
Calcium	1.6
Magnesium	0.75
Chloride	101
Lactate	45
Glucose	11

The typical hemodiafiltration circuit is displayed in Figure 8.13.

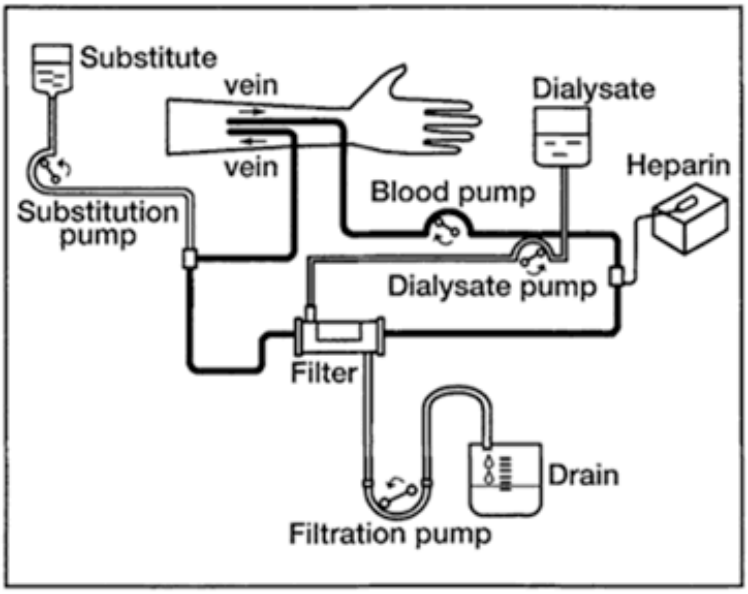


Figure 8.13 Hemodiafiltration circuit [179]

As displayed in the figure, blood and dialysate are pumped through the filter and the filtrate is drained. The substitution fluid containing the desired substances is then infused into the blood in the distal part of the circuit.

8.1.2.3 Performance assessment and Validation of Hypothesis

In this study, we compare these renal replacement techniques on the basis of solute removal and mortality rate. A summary of results from recent studies is as follows:

Studies on small molecular weight solute removal

- Clearance of small solutes (< 500 Da) such as urea (60 Da) and creatinine (113 Da) is largely dependent on diffusion processes. Hemodialysis and hemodiafiltration both showed effective removal of these small solutes, while hemofiltration did not. There was little difference between hemodialysis and hemodiafiltration in clearance of these small solutes [180].
- Studies by Ward, et al. [181] showed small improvements (10-15%) of urea and creatinine removal for hemodiafiltration and high-flux hemodialysis.

Studies on medium molecular weight solute removal

- In a study by Ahrenholz, et al. [182], the experimenters showed a 123% improvement in clearance of inulin (5200 Da) in hemodiafiltration versus high-flux hemodialysis. [180].
- β_2-microglobulin (11,800 Da) is not removed at all by hemodialysis because it is larger than the typical hemodialysis membrane pore. Kerr et al. [183]reported 54.8% reduction of β_2-microglobulin of high-flux hemodialysis and 62.7% reduction in hemofiltration after a 3 hour session. Lorney et al [184]reported a 49.7% reduction using high-flux hemodialysis, compared to 72.7% with hemodiafiltration, in a 4 hour session. In a 245-minute session, Maduell et al. [180] reported -0.2, 60, and 75% reductions with hemodialysis, high-flux hemodialysis, and hemodiafiltration, respectively. [180]
- In a study by Ward et al. [181], hemodiafiltration resulted in greater removal of β_2-microglobulin than high-flux hemodialysis, as indicated by a significantly higher pre- to posttreatment change in concentration (73 ± 1% versus 58 ± 1%, respectively).
- In renal failure, β_2-microglobulin accumulates in the body and can be deposited in bone and joints in the form of amyloid. In the HEMO study [185], a significant relationship was found in pre-dialysis β_2-microglobulin and all causes of mortality; mortality increased by 11% for every 10mg/L rise in β_2-microglobulin concentration [186].

- Dember and Jaber [187] estimated yearly accumulation of 111 g, 97 g, and 51 g for hemodialysis (4 hrs, 3 times per week), high-flux hemodialysis (4 hrs, 3 times per week), and hemofiltration (2 hrs, 6 times per week), respectively. Hemodiafiltration should show results similar to that of hemofiltration [186]

Recent studies on Mortality

- In the Dialysis Outcomes and Practice Patterns (DOPPS) study [188], after adjusting for demographic and other contributors to mortality, experimenters reported a 35% better survival rate with hemodiafiltration (11.9 deaths/100 patient years) versus hemodialysis (14.2 deaths/100 patient years).
- In analysis of data from The European Clinical database, Jirka et al.[189] reported a 35.3% better survival rate with hemodiafiltration versus hemodialysis. This study was also adjusted for other contributors to mortality.

In summary, in studies of small molecular weight solute removal such as urea and creatinine, both hemodiafiltration and hemodialysis showed efficient removal, with hemodiafiltration showing slight improvement. In studies of middle molecular weight solute removal, hemodiafiltration was shown to significantly improve the removal of β_2-microglobulin. β_2-microglobulin amounts were shown to positively correlate to increased mortality. In specific large scale studies on mortality, hemodiafiltration was shown to have a nearly 35% improvement in survival rate. A table summarizing the use of kidney filtration strategies in renal replacement systems is displayed in Table 8.3.

Table 8.3 Summary Table for Renal Replacement Therapy Case

Level of Decomposition - *Strategy*	**Renal Therapy**	**Description**
Level G0 - *"The composition of blood in the kidney is modified through filtration"*	Hemodialysis	Removal of waste from the blood through one-step filtration process. **Shows only good small solute removal.**
Level G1 - *"Filtration in the kidney is performed by removing mostly all substances from the blood through convection/diffusion in the Bowman's Capsule and reabsorbing and secreting needed substances in the Proximal tubule, Loop of Henle, and Distal Tubule, and Collecting Duct. The remaining solutes are excreted through the Collecting Duct."*	Hemodiafiltration	Removal of waste through a two-step filtration and reabsorption process. **Shows good small, medium, and large solute removal.**

Our hypothesis states that bio-inspired engineering systems possessing a deeper level of biological system behavior will perform better than those possessing superficial behavioral similarities. In the early development of renal replacement therapies, engineers developed hemodialysis as a means to replace the function of the kidney and treat kidney disease. Hemodialysis mimics the general behavioral strategy of the kidney in filtering the blood of waste, however it is not very efficient at removing harmful middle molecular weight solutes from the blood and does not have a particularly high survival performance. In the 1970s, scientists developed hemodiafiltration as a means of improving on the performance of renal replacement therapy. Hemodiafiltration functions in a similar fashion as that of the human kidney; hemodiafiltration filters large amounts of water and substances from the blood through convection and diffusion, then replaces the substances needed for bodily function. This setup has shown improved performance in removal of small and middle molecular weight solutes, as well as significant improvements in the survival rates of patients receiving this form of treatment. Through this case study, we have shown that renal replacement therapies possessing a deeper level

of similarity perform better with respect to mortality and solute removal, thus providing support for our hypothesis.

In Section 8.1, a case was made that bio-inspired engineering systems possessing a deeper level of similarity to biological system behavior will perform better than those possessing superficial behavioral similarities. This was shown through 2 case studies: avian flight control compared to innovations in aviation (Section 8.1.1) and the human kidney compared to current renal replacement therapies (Section 8.1.2).

In Section 8.2, a comprehensive example of solution-driven Bio-inspired Conceptual Design is presented. In this case study, the human kidney is reverse engineered and a novel renal replacement therapy is designed.

8.2 COMPREHENSIVE EXAMPLE: CONCEPTUAL DESIGN OF A NOVEL RENAL REPLACEMENT THERAPY

In Section 8.1, 2 historical case studies of aviation and renal replacement therapy were presented. In these studies, it was found that bio-inspired engineering systems possessing a deeper level of similarity to biological system behavior perform better than those possessing superficial behavioral similarities. Using this finding as motivation, in this section, a novel renal replacement therapy that closely mimics the behavior of the kidney is developed. In this example, the process of solution-driven Bio-inspired Concept Generation is detailed. Specifically, the Method for Reverse Engineering Biological Systems is used to decompose the behavior of the kidney and extract the behavioral strategy. This strategy is then used in the development of the renal replacement therapy system.

8.2.1 The Problem

The functional output of a human kidney is measured by its glomerular filtration rate (GFR). End-Stage Renal Failure (ESRD) is a disease inflicting hundreds of thousands of patients worldwide that is defined by a GFR below 15% of normal kidney function. Patients diagnosed with ESRD have two survival options: receive a donor kidney, or begin dialysis.

While organ transplant is the best option, there are simply not enough donor kidneys available. Consequently, most patients begin peritoneal dialysis – a reasonable form of treatment that uses the body's own peritoneum as a filter – allowing users to retain many aspects of their normal pre-ESRD lifestyle. After a few years of peritoneal dialysis, the filtration ability of the peritoneum becomes inadequate for peritoneal dialysis and patients are forced to begin traditional hemodialysis.

Hemodialysis requires most patients to visit a clinic three times a week for three to five hour treatment sessions. Blood is circulated out of the body, cleansed through dialysis, and then returned to the body. Unfortunately, modern hemodialysis fails in two major ways: it is unable to perfectly clean the blood, and it is a very invasive process.

Advancements have been made by several private companies to bring hemodialysis into the home, releasing patients of the need to travel to a clinic for treatment. While this is a considerable advancement, the actual hemodialysis process patients endure is fundamentally the same: needle sticks, blood filtered across a membrane, patients tied to a stationary machine for hours on end.

In Section 8.1.2, it was found that renal replacement therapies that more closely mimic the behavior of the human kidney on the systems level perform better. Therefore, in this study, the solution-driven approach is used to conceptually design a novel RRT system that closely mimics the behavior of the kidney. The specific problem statement for this study is as follows:

Problem Statement: Design a renal replacement therapy system that closely mimics the waste removal function of the kidney. Specifically, we wish to design a renal therapy that allows the regulation of solutes that are reabsorbed into the blood. This regulation will allow for more hemodynamic stability and continuous waste regulation of the blood.

8.2.2 Solution-driven Conceptual Design

The solution-driven approach includes the following steps: (1) Identify biological systems of interest, (2) Analyze system, (3) Extract biological strategies, and (4) Generate ideas.

Step 1: Identify biological systems of interest

In this step, the biological system of interest is identified. In this study, the human kidney is identified as the system of interest. As mentioned earlier, the goal of this study is the design of a more biologically-correct renal therapy. Specifically, a system is sought that mimics the waste removal function of the human kidney at the systems level.

Step 2: Analyze system

In this step, the Method for Reverse Engineering Biological Systems is used to systematically analyze the function, behavior, and structure of the human kidney. Specifically, the waste removal function of the kidney is analyzed.

1) Define root system:

In this step, the designer defines the root biological system of interest. The root system is the human kidney. As displayed in Figure 8.14, the human kidney inputs dirty blood and outputs urine and clean blood. There is also a control signal dictating kidney operation.

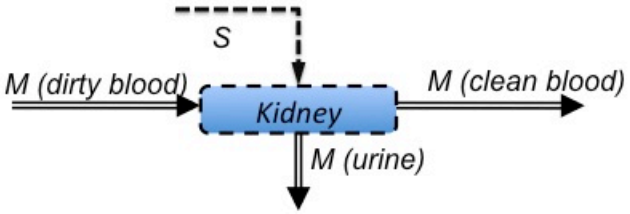


Figure 8.14 Root System – Human Kidney

2) Define standalone behavior:

In step 2, the designer defines the behavior of the root system using the Petri Net modeling formalism. The behavior of kidney is displayed in Figure 8.15. As opposed to being modeled as a discrete system, the human kidney is modeled as a continuous system with the state of the system dictated by the composition of solutes at different places. The behavior of the kidney, *filter*, is represented by a transition. The composition of the blood before and the blood and urine after filtration is represented within the places.

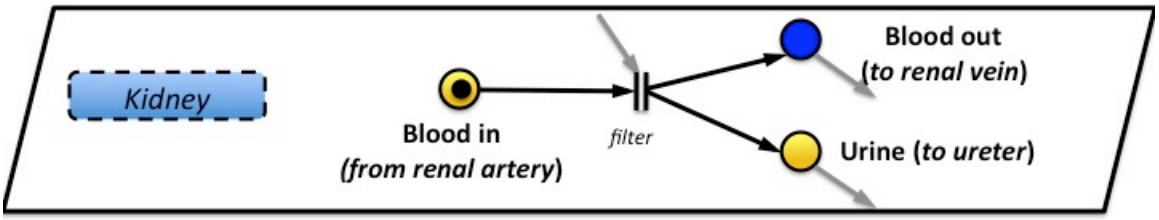


Figure 8.15 PN model of the overall behavior of the human kidney

3) Decompose system and sub-systems

In step 3, the system is decomposed into its subsystems. The functional unit of the kidney is the nephron. The nephron can be further decomposed into its different subsystems, including the Bowman's capsule, proximal tubule, loop of Henle, distal tubule, and the collecting duct.

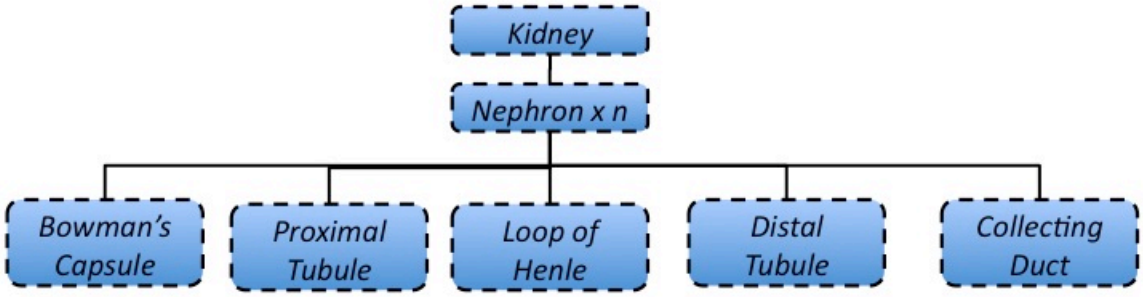


Figure 8.16 Structural Decomposition of the Human Kidney

Next, the interactions between the components are modeled and displayed in Figure 8.17. As seen in the figure, blood solutes from the Bowman's capsule flow through the proximal tubule, loop of Henle, and the distal tubule, before exiting at the collecting duct as urine. Solutes are also exchanged with the Vasa Recta before leaving the kidney in the clean blood.

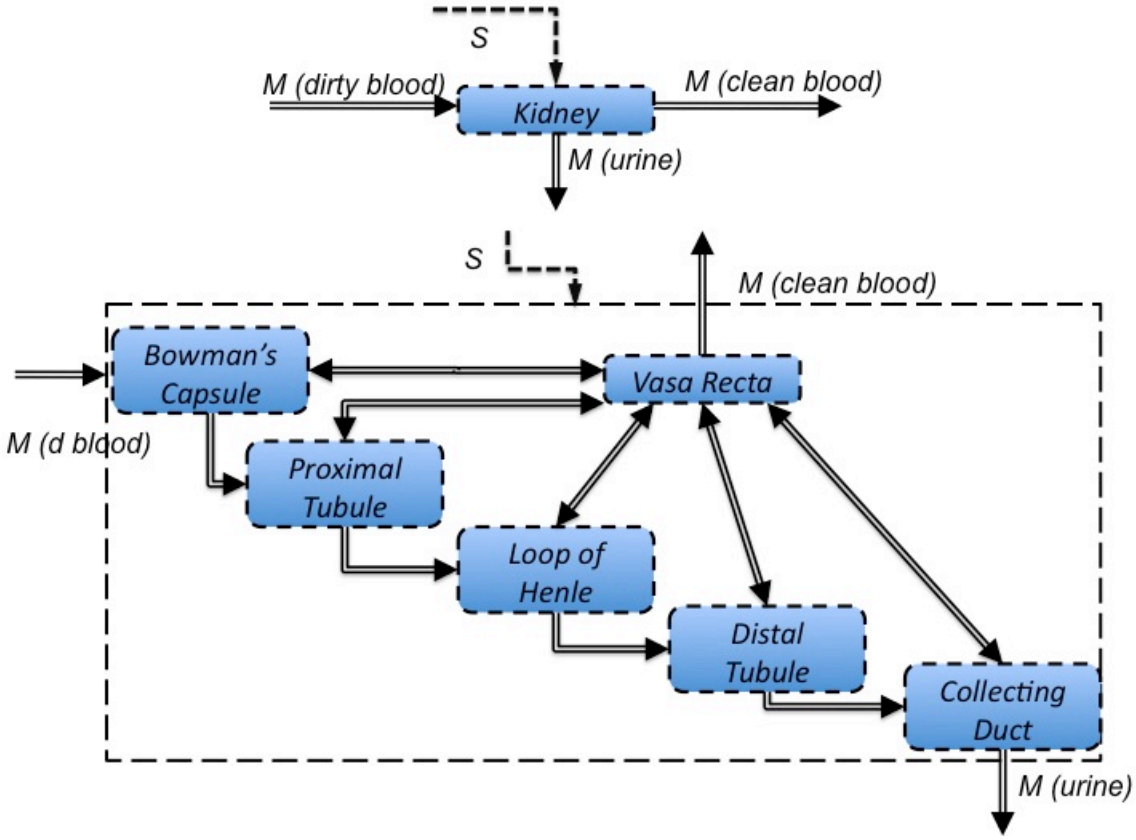


Figure 8.17 Kidney subsystem interactions

4) Define standalone behaviors of sub-systems

Following the procedure from Step 2, the standalone behaviors of the subcomponents are identified. The individual behaviors of the kidney subsystems displayed in Figure 8.16 are defined in Figure 8.18. In this figure, the behavior of 3 solutes (Ca^{2+}, glucose, and urea) are represented. Consider the behavior of the proximal tubule. The composition of Ca^{2+}, Glucose, and Urea at entry to the proximal tubule are 540, 800, and 933 mmol/day, respectively. In this tubule, 70 % of Ca^{2+}, 100% of Glucose, and 50% of Urea are reabsorbed into the blood.

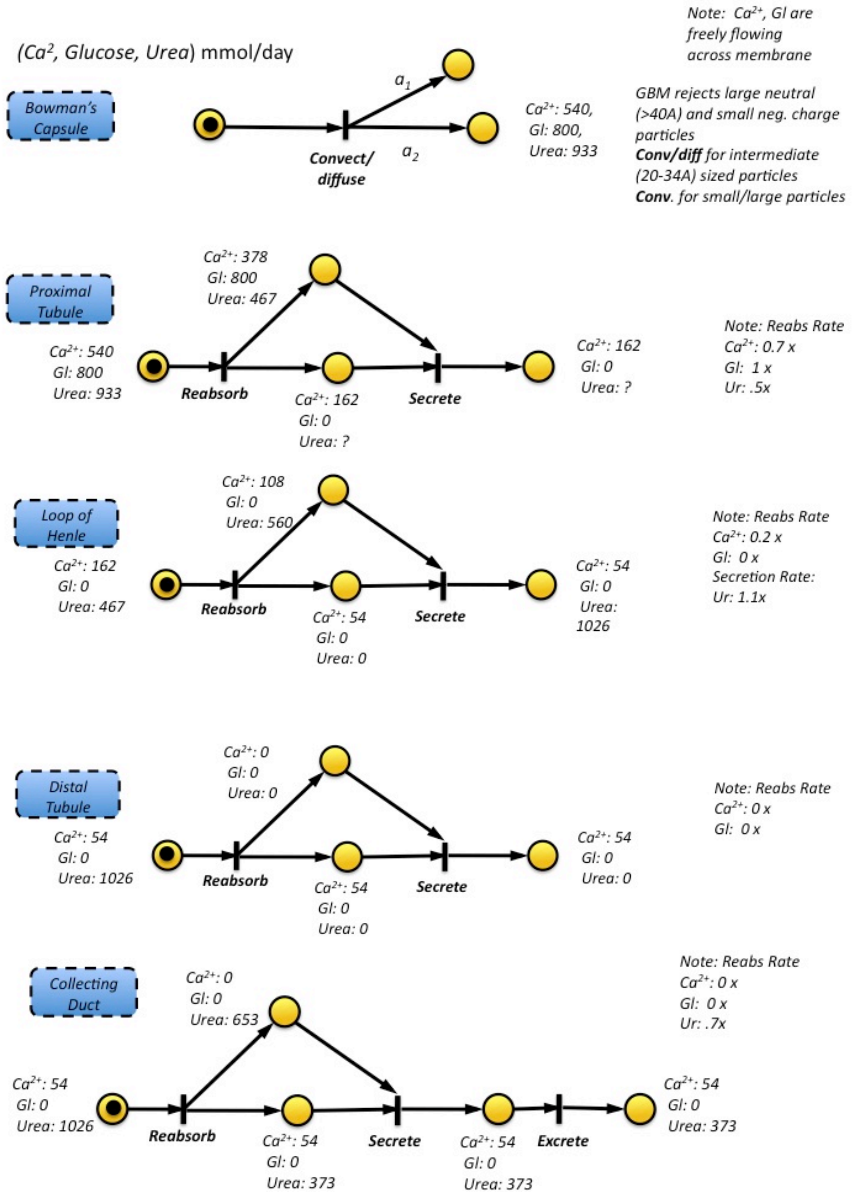


Figure 8.18 Standalone behaviors of the Kidney subsystems

5) Define interface relationships between subsystems

In this step, the interface relationships between subsystems are defined using external arcs and are displayed in Figure 8.19. Double-dashed lines are defined between the places of the subsystems, indicating these places are synchronized.

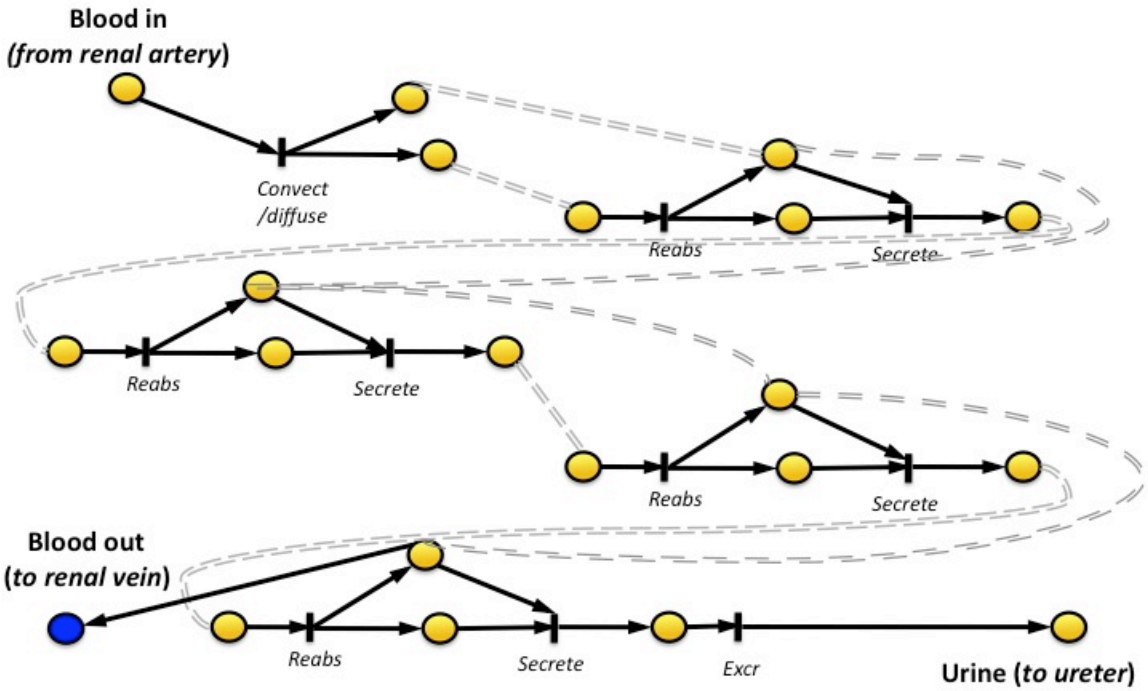


Figure 8.19 Interface relationships between the Kidney subsystems

6 - 8) Generate combined behavioral model, Identify Subnets, and Create Macrotransitions

Since steps 6-8 are iterative, the discussion below is combined. Figure 8.20 displays the hierarchical Petri net model for the human kidney. After generating the reachability graph, the subnet of the filter transition is identified. In the figure, the subnet of the transition 'filter' is defined using a gray outline in the lower net. The transition *filter* is now defined as a macrotransition because it has a subnet associated with it.

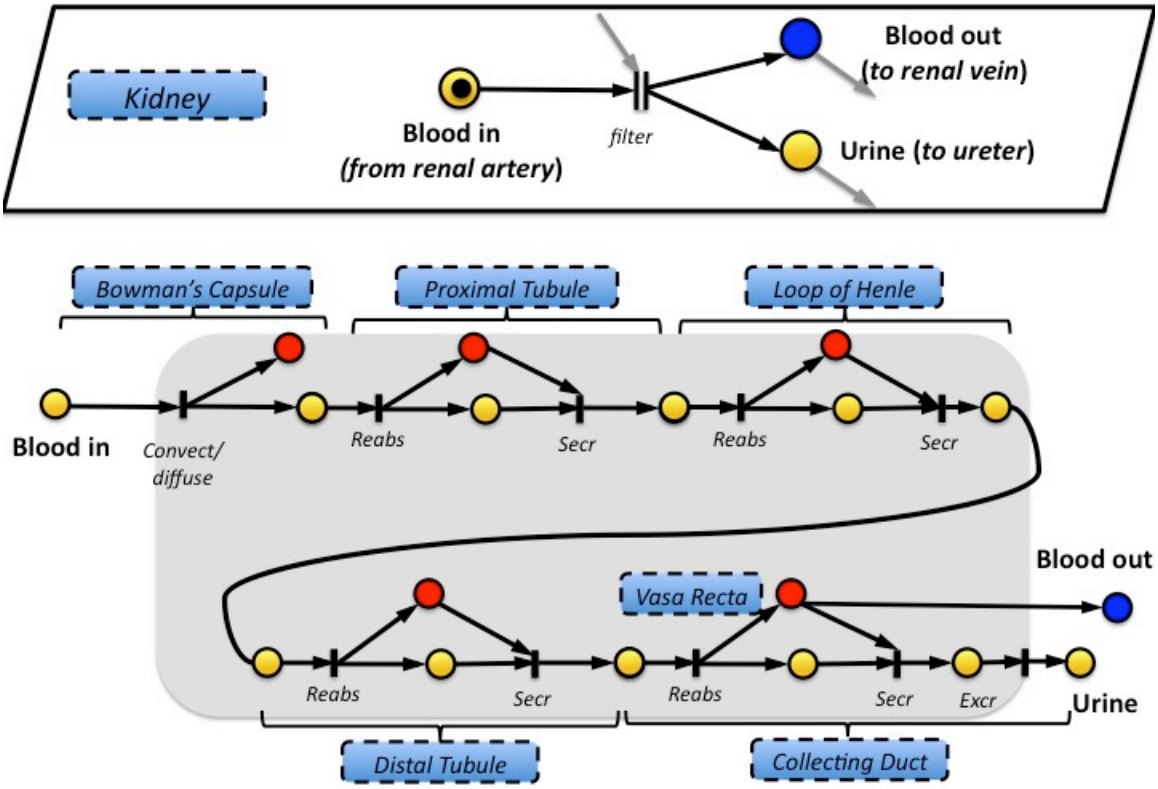


Figure 8.20 Hierarchical Petri net representation of the Human Kidney

Step 3: Extract Biological Strategies

In this step, strategy is systematically extracted from the subnet representing the behavior of the lower level systems. Using the method defined in Section 4.1, strategy is extracted from the model as follows:

Strategy (Filtration) = *BowCap* (Conv./Diff.), *PrTub* (Reabs/Secr), *LHen* (Reabs/Secr), *DisTub* (Reabs/Secr), *ColDuct* (Reabs/Secr/Excr)

The strategy of the Human Kidney to filter specific solutes can also be defined:

Strategy (Filtr. - Glucose) = *BowCap* (Conv./Diff.), *PrTub* (Reabs 100%)

Strategy (Filtr.- Ca^{2+}) = *BowCap* (Conv./Diff.), *PrTub* (Reabs 70%), *LHen* (Reabs 20%), *ColDuct* (Excr 10%)

Strategy (Filtr. - Urea) = *BowCap* (Conv./Diff.), *PrTub* (Reabs 50%), *LHen* (Secr 60%), *ColDuct* (Reabs 70% / Excr 40%)

In natural language, the strategy of the Human Kidney can be defined as:

"Filtration in the kidney is performed by removing mostly all substances from the blood through convection/diffusion in the Bowman's Capsule and reabsorbing and secreting needed substances in Proximal tubule, Loop of Henle, and Distal Tubule, and Collecting Duct . The remaining solutes are excreted through the Collecting Duct."

Step 4: Idea Generation

In this step, the biological strategies are used to stimulate the generation of working principles. Based on the strategy extracted for waste removal in the kidney in Step 3, the following high-level function structure, displayed in Figure 8.21, was generated. This model will help guide the conceptual design process.

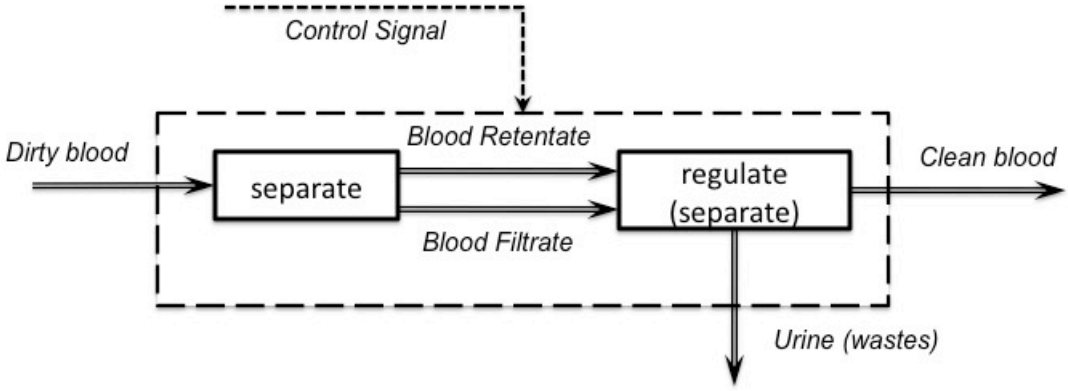


Figure 8.21 High level function structure of the Kidney

In the model, the dirty blood is first separated into blood filtrate and retentate. In the regulation step, substances in the filtrate are separated in order to selectively reabsorb the needed substances into the blood and remove the unneeded substances (waste) in the form of urine. After reabsorbing the needed substances and secreting the unneeded ones, the now clean blood is circulated back into the body. The rate of reabsorption and secretion of the substances is regulated by the body. Since the goal of this case study is the conceptual design of a new renal therapy that closely mimics waste removal in the kidney, we use this model as a base for our design.

Based on the function structure, the Strategy Repository is used to retrieve potential strategies to use in mimicking the kidney. The primary function of the kidney is separation, therefore, the repository is used to retrieve strategies for separation. The search is structured as follows:

$\exists$ *satisfiesFunction*.[$\exists$*hasInput*.Mixture $\sqcap$ $\exists$*hasOutput*.Mixture $\sqcap$ $\exists$*hasOutput*.Mixture] $\sqcap$ $\exists$ *refinesBehavior*.[$\exists$*hasAction*.Separate] $\sqcap$ $\exists$*fromDomain*.Engineering_Domain.

With this search, an engineering strategy that separates a mixture into two mixtures is sought. The repository search is displayed in Figure 8.22.

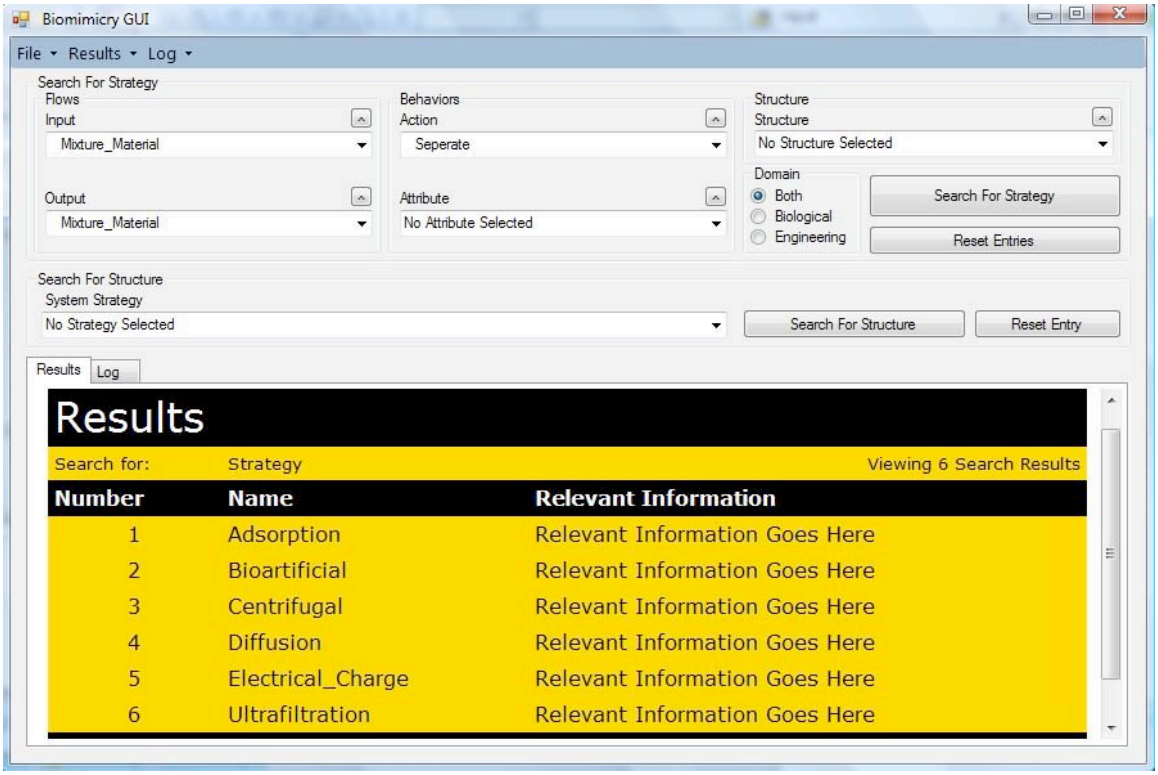


Figure 8.22 Repository search for separation strategies

The retrieved strategies are described briefly as follows:

Diffusion - Diffusion is a membrane-based separation strategy in which a solute concentration gradient is the primary driver. In diffusion, solutes flow across a membrane from areas of higher concentration to areas of lower concentration. Control over which solutes in a mixture cross the membrane is usually done by modifying the relative concentration between both sides of the membrane and by altering the properties of the membrane itself.

Ultrafiltration - Ultrafiltration is a membrane-based separation strategy in which hydrostatic pressure forces a mixture across a semi-permeable membrane. Large solutes in the mixture are retained, while smaller solutes and the liquid pass through the membrane.

Electrical Charge - Another separation strategy is driven by the electric charge of the solutes in a mixture. This process, termed electrophoresis, is commonly used to sort ions in a fluid. A fluid, full of ions to be sorted, is mixed with a buffer solution and run between two oppositely charged plates. As the fluid and buffer flow parallel along the plates, the charged ions are pulled towards the plates based upon their electric charge.

Bio-artificial - Bio-artificial separation uses actual mammalian renal cells as part of a membrane-based filtering system. Living renal cells can be suspended onto a polymer membrane scaffold, and behave as actual renal cells in the kidney: pumping solutes in and out of the blood.

Centrifugation - Centrifugation is a separation strategy involving the use of centrifugal force for the separation of substances in a mixture. In this case, the heavier components of the mixture move away from the axis of the centrifuge, while lighter components move towards the axis.

Adsorption - Adsorption is a separation strategy that uses chemical affinity to separate specific substances from a mixture. In this case, a sorbent is used to adsorb the unwanted substances from the mixture.

These strategies can now be used to stimulate working principles for the artificial human kidney. First, the first separation process, where most of the substances are removed from the blood, is addressed. The goal of this separation process is simply the removal of many of the substances, including wastes and needed substances, from the blood. For this process, many of the strategies can be used, however, the membrane-based ultrafiltration strategy was chosen. Ultrafiltration is used in this case to separate the plasma from the whole blood. The plasma contains both needed and unneeded substances. In ultrafiltration, control over which solutes cross the membrane barrier is

primarily done by either modifying the pore size of the membrane or the filtration pressure (blood pressure).

The goal of the second separation process is regulation of substances reabsorbed back into the blood. This means that the traditional membrane-based systems won't work, as they don't allow active control of which solutes are removed. This separation must be performed on a continuous basis, as is done in the human kidney. For this second separation process, electrophoresis is considered. With electrophoresis, the substances in the blood are sorted by electrical charge. Using this strategy, the filtered solutes can be reclaimed and directed back into the blood. Because the electrical input can be varied, the substances that are separated can also be controlled on a continuous basis. This allows for regulation of solutes similar to that of the kidney.

Based on the discussion above, the two specific principles chosen for use in our renal replacement system are ultrafiltration-based separation followed by electrophoresis-based separation. Thus, the functions in the kidney function structure from Figure 8.21 can be replaced by specific working principles, as displayed in Figure 8.23.

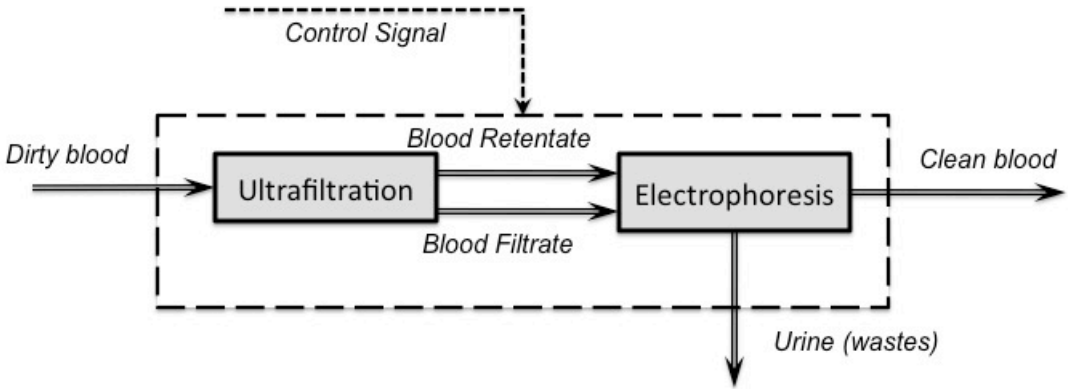


Figure 8.23 Specific working principles for renal replacement therapy concept

Based on the working principles in Figure 8.23, the following concept, displayed in Figure 8.24, is developed.

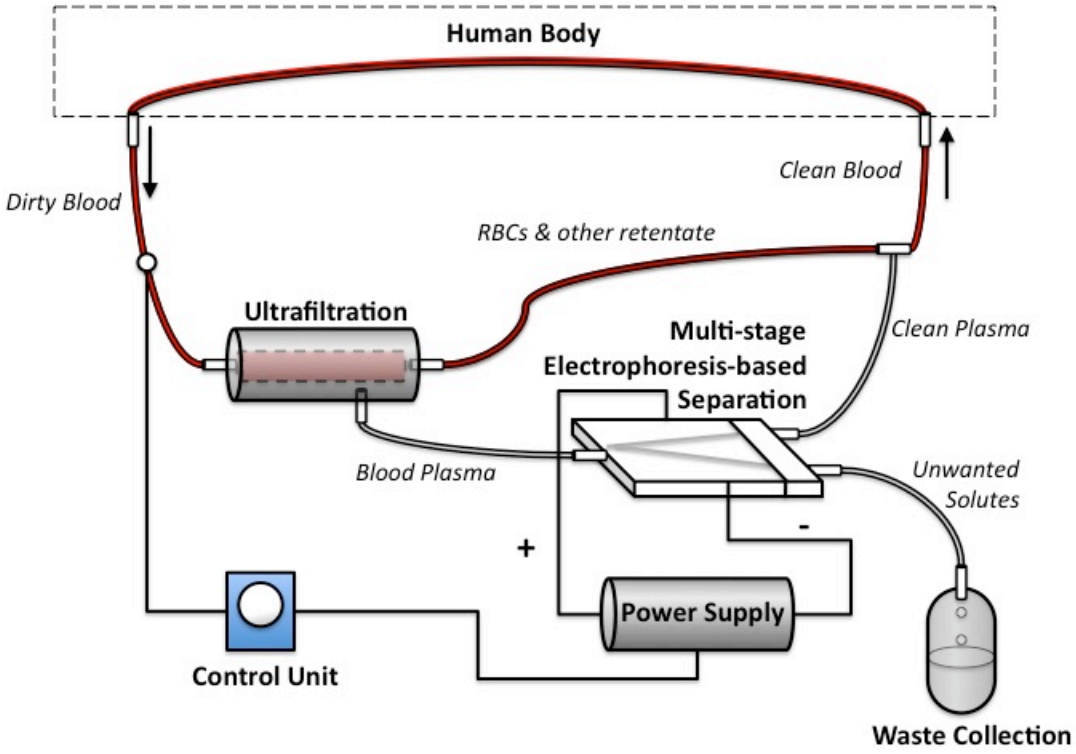


Figure 8.24 Renal replacement therapy (RRT) concept

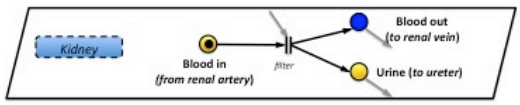
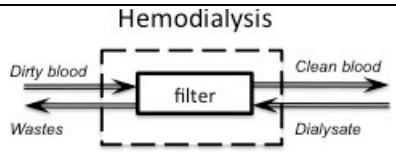
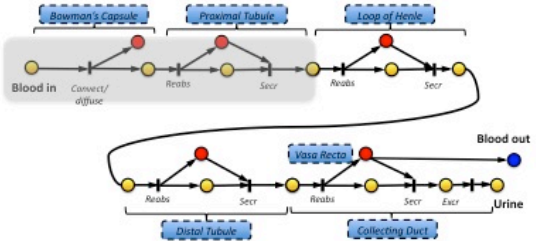
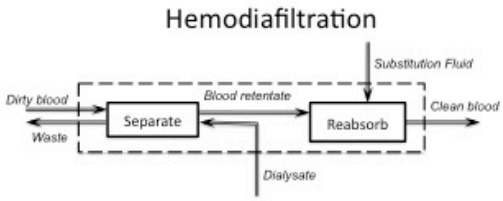
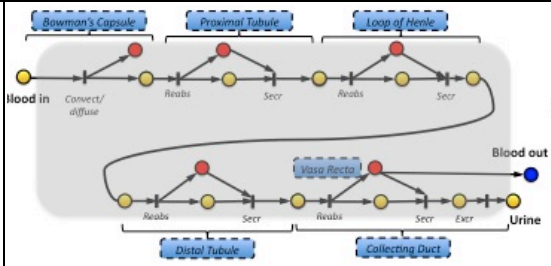
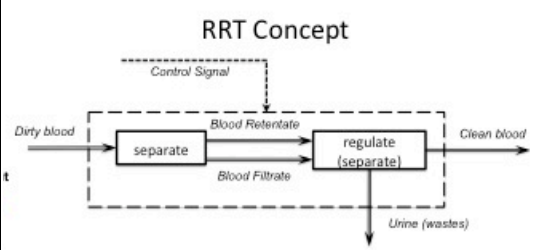
The concept displayed in Figure 8.24 combines ultrafiltration for the glomerular filtration function of the kidney and electrophoresis for the substance regulation function of the kidney. In the concept, dirty blood leaving the body is filtered by ultrafiltration to separate the blood plasma from the whole blood. This plasma is then sent to a multi-stage electrophoresis system, whereby unwanted substances, such as urea and Beta-2-microglobulin, are separated from the plasma. The multi-phase system is used to get better resolution between substances in the separation process. The clean plasma is then recombined with the blood retentate and returned to the body. The waste is collected for disposal. A control unit is used to regulate the concentration of solutes that are removed by the multi-stage electrophoresis system.

8.2.3 Comparison and Discussion of performance

The goal of this design example was to design a renal replacement therapy that more closely mimics the waste removal strategy of the kidney. This comparison is summarized in Table 8.4 (referencing Figure 8.20). In Section 8.1.2, hemodialysis and

hemodiafiltration were compared on the basis of their behavioral similarity to the kidney. Hemodialysis can be seen as similar to the first level of behavior of the human kidney, which is simply filtration of the blood. Hemodiafiltration, on the other hand, possesses a deeper level of behavioral similarity by first filtering most of the substances from the blood, and then replacing the needed substances using a substitution fluid. Although this method is more similar to the actual behavior of the kidney than hemodialysis, it still lacks in allowing regulation of the reabsorbed solutes. In the kidney, needed substances are regulated through multiple steps of secretion and reabsorption throughout the nephron. The RRT concept developed in Section 8.2.2 allows for continuous regulation of the needed solutes using a multi-stage electrophoresis system.

Table 8.4 Renal Replacement Therapy Comparison

	Kidney Strategy	Renal Therapy Strategy
G0		
G1		
G1		

The question now becomes, "What type of advantage would this RRT concept have over existing renal therapies?" As this is just a research concept, the performance of the system can only be theorized. To get a true estimate of performance, many years of

development and trials are needed. However, the RRT concept presented in Figure 8.24 has several theorized advantages over the current renal therapies, including:

1. Selectivity in filtration
2. Two-stage processing
3. Continuous solute regulation

One of the key advantages of the RRT concept is its ability to selectively reabsorb molecules. There are two obvious ways to filter out individual molecules: by size or by charge. Size selectivity is limited by the manufacturing capabilities of current membrane technologies. In order to selectively filter by charge, a membrane must be made with the exact pore size of the selected molecule. Today's technology is currently not capable of such precision manufacturing (pore sizes are around fifteen nanometers in size). Even if this was possible, the membrane would still allow any molecule smaller than this cutoff point through the membrane. Charge selectivity, on the other hand, allows filtration of molecules by charge. The strength of the filter is defined by the electric field and the resolution of the charges of the molecules. Thus, the electrophoresis-based reabsorption strategy allows individual molecules to be selectively reabsorbed, which is not currently feasible with current membrane-based technologies.

Another key advantage of the RRT concept is the two-stage solute processing. In the RRT concept, ultrafiltration is used to separate all of the substances from the blood and multi-stage electrophoresis is used to selectively reabsorb the needed substances. It is believed that it is much more efficient to remove all the substances from the blood and selectively reabsorb the needed substances than to try to only remove selected waste from the blood in a "one shot" fashion, as is done in current dialysis technologies.

In addition, the electrophoresis-based separation process in the RRT concept allows for continuous solute regulation. In the RRT concept, solute regulation can be performed on a continuous basis, as opposed to intermittently when using a replacement fluid in hemodiafiltration. A continuous-based therapy will help to increase the hemodynamic stability and biocompatibility of the treatment.

8.3 CLOSURE AND VALIDATION

The research question posed at the onset of this chapter is as follows:

"What is the impact of biological strategies in the conceptual design process?"

In essence, with this question, the value of bio-inspired design in the conceptual design process is assessed. In this research, the value of biological strategies is assessed in two different contexts: (1) problem-based Conceptual Design, where the designer seeks to be inspired by biological strategies in the ideation process and (2) solution-driven Conceptual Design, where the designer is seeking better solutions to engineering problems by mimicking the biological strategies. The value of the biological strategies and the proposed method in the problem-driven context was assessed in Chapter 7. In this chapter, the value was assessed in the solution-driven context.

Specifically, in the solution-driven context, it was hypothesized that Bio-inspired engineering systems possessing a deeper level of biological system behavior will perform better than those possessing superficial behavioral similarities. In Section 8.1, historical case studies were presented and it was concluded that systems possessing deeper levels of similarity did perform better with respect to the chosen metrics. In both the avian flight control and the artificial human kidney case studies, support was found for this hypothesis. Specifically, in the avian flight control case (Section 8.1.1), it was concluded that the ability to manipulate lift in maneuvering and control increased as the level of similarity between the behavior of the bird wing and flight control surfaces on aircraft increased. In the artificial kidney example presented in Section 8.1.2, it was concluded that the survival rate and removal of solutes increased with the level of similarity of the renal replacement therapy to that the human kidney. Since it is concluded that there is value in high levels of behavioral similarity with biological systems, there is also value in rich representations of this behavior. These representations allow more behavioral knowledge to be extracted and represented from biological systems. Specifically, the hierarchical Petri net representation allows for biological system behavior and strategy to be represented at multiple levels of refinement.

In Section 8.2, these historical case studies were followed by a case study in solution-based Bio-Inspired Conceptual Design. In this case, a novel renal replacement

therapy was designed and compared to others in industry. In this case, it was found that the proposed method for solution-driven Conceptual Design resulted in not only a renal therapy that was more similar to that of the kidney, but also one that was hypothesized to be better performing.

Empirical Performance Validity

The validation strategy in this dissertation is displayed in Figure 8.25.

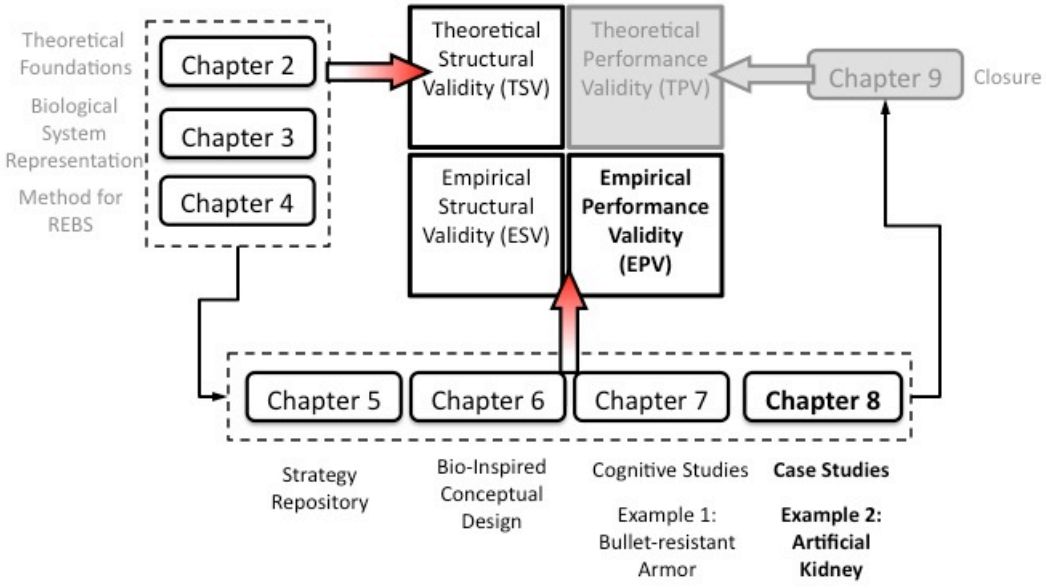


Figure 8.25 Validation Strategy and Chapter 8

Empirical Performance Validation involves accepting the usefulness of the method for some representative example problems. In Section 8.1, historical case studies were used to show the value of rich behavioral models in solution-driven Conceptual Design. In the historical case studies, bio-inspired systems that more closely mimicked the behavior of the target biological system were found to perform better. In Section 8.2, a case study on the design of a novel renal replacement therapy system was presented. In this study, several advantages of the design generated were found over current renal therapies.

CHAPTER 9 CLOSURE AND CONTRIBUTIONS

In this chapter, the research questions and their respective hypotheses are revisited. The specific contributions to the body of knowledge are also reviewed in this chapter.

9.1 REVISITING THE RESEARCH QUESTIONS

Due to its inherent difficulties, bio-inspired design has thus far followed an ad hoc path. Although several researchers have developed approaches for identifying and transferring biological strategies to the engineering domain, several shortcomings were identified in Section 1.2.3. These shortcomings include the lack of research on representing biological systems so that strategies can be easily accessed and comprehended, inefficient identification of these strategies, and a lack of empirical evidence on the advantage of these biological strategies in Conceptual Design. Given these shortcomings, the author set out to answer the following primary research question in this research:

Primary Research Question:
How can we aid the designer in more "effective" idea generation in Conceptual Design?

To answer this question, the following hypothesis is proposed:

Primary Research Hypothesis:
Building upon a rich behavioral model of biological systems and a strategy repository, the proposed approaches to Bio-inspired Conceptual Design can be used to aid the designer in (1) identifying relevant biological strategies and (2) using biological strategies in Conceptual Design to produce 2a) a larger variety of design ideas (2b) design ideas of greater novelty and (2c) higher quality design ideas.

The fundamental claims of the primary hypothesis of this research include that of biological representation, efficient retrieval, and assessing the impact of biological strategies on Conceptual Design. To validate these claims, several sub-research questions were proposed and are discussed in the following sections.

9.1.1 Research Question and Hypothesis 1

Representations play a key role in understanding complex systems, especially when these systems can't be experienced directly. However, representing and understanding biological systems is very difficult. It is believed that an explicit representation of biological systems can aid in bridging the gap in the transfer of these technologies to engineering systems. This motivation led to the following question:

Question 1: "*What type of representation can be used to model the behavior of biological systems?"*

To answer this question, Hypothesis 1 stated that

Hypothesis 1: A representation based on (1) a causal behavioral description and (2) hierarchical Petri nets can be used to model the behavior of biological systems.

In Chapter 3, Hypothesis 1 was validated through qualitative evaluation of the causal behavioral description and the hierarchical Petri net representation against several representation criteria derived from the psychology and design literature. Specifically, the Structure-Behavior-Function (SBF) model, Function Behavior-State (FBSt) model, Functional Rationale (FR), Function-Behavior-Structure (FBS) model, Function-Environment-Behavior-Structure (FEBS) model, and the Causal Behavioral Model (SAPPhIRE) were evaluated against requirements for representing biological systems. These requirements include hierarchical representation, explicit dynamic representation, explicit representation of the environment, behavior-centric approach, and completeness and uniqueness of representation. After evaluation, a derivative of the SAPPhIRE

Model, termed the causal behavioral description (Section 3.2.4), was found to meet these representation requirements.

In Section 3.3, the representation, or expression, of the causal behavioral description was considered. In this case, several expressions were evaluated, including the textual, static diagrammatic, and dynamic diagrammatic expressions. The Petri net representation, a dynamic diagrammatic expression, of the causal behavioral description was found to meet the requirements put forth for computational offloading, inference, validity, consistency, isomorphism, model complexity, and behavior verification. In Section 3.4, building on the Petri net representation validated in Section 3.3, the hierarchical Petri net representation was developed. Therefore, after qualitative evaluation, a representation based on a causal behavioral description and hierarchical Petri nets was found to meet the requirements for modeling the behavior of biological systems.

9.1.2 Research Question and Hypothesis 2

The purpose of the Method for Reverse Engineering Biological Systems is to aid the designer extracting behavioral strategies from biological systems. Consistent strategy extraction requires consistency in behavior across hierarchical levels of the Petri net representation. This motivation led to the following research question:

Question 2: *How can the behavior of biological systems be hierarchically represented using Petri nets, while preserving the fundamental properties at each hierarchical level?*

In answering this question, it was hypothesized in Hypothesis 2 that:

Hypothesis 2: The systematic method for Reverse Engineering Biological Systems will ensure that the fundamental properties of boundedness, reachability, and liveness will be preserved across hierarchical levels.

To validate this hypothesis, mathematical proofs for the preservation of boundedness, liveness, and reachability across hierarchical levels when using the proposed method were presented in Section 4.3. Preservation of these properties was found to be a direct result of behavioral mapping in the proposed method. In behavioral mapping, a reachability graph is used to systematically combine the individual behaviors of the subsystems, while the subnet definition is used to map the behaviors of lower-level subnets to the more abstract behavioral net. Specifically, boundedness is preserved by the combined behavioral graph generation, where $M(p) \leq 1$, and the subnet definition, where $M(\bullet S^t)=M(\bullet t)$ and $M(S^t \bullet)=M(t \bullet)$. Liveness is also preserved by the combined behavioral graph, where a transition t only appears if it is live. By subnet definition, $\sigma(\bullet S^t, S^t \bullet) \subseteq \sigma(\bullet t, t \bullet)$, or the firing order is unchanged. Therefore, liveness is preserved. By definition, the reachability graph $R(M_0)$ used to generate the combined behavioral graph, generates all reachable states, M, from state M_0. By subnet definition, $M(\bullet S^t)=M(\bullet t)$ and $M(S^t \bullet)=M(t \bullet)$, and reachability is preserved. From this, it was concluded that the fundamental properties were preserved when following the proposed method for Reverse Engineering Biological Systems.

Two illustrative examples detailing the proposed method were presented in Section 4.4. In Section 4.4.1, the method for Reverse Engineering Biological Systems was used to extract the behavioral strategy from the mutable connective tissue of echinoderms. In Section 4.4.2, the proposed method was used to extract the behavioral strategy from muscle fiber in isometric contraction.

9.1.3 Research Question and Hypothesis 3

Identification of relevant biological strategies is a key issue in bio-inspired design. Current approaches are useful for storing and providing access to biological information, however, the generic keyword-based retrieval process utilized by these approaches suffers from providing too many and/or irrelevant results [30]. In this research, it is believed that structuring biological information using ontologies can lead to more accurate and efficient retrieval of biological strategies. The following research question was posed:

Question 3: *How can hierarchical Petri net representations of biological systems be structured to aid retrieval of relevant strategies from a knowledge repository?*

To answer this question, Hypothesis 3 stated:

Hypothesis 3: An ontology of concepts from hierarchical Petri net representations of biological systems can be represented using Description Logics and that the subsumption algorithm in Description Logics will enable consistent and precise retrieval of relevant biological strategies from a knowledge repository.

To validate this hypothesis, an ontology of concepts from the hierarchical Petri net representation of biological systems was structured using taxonomies for energy, material, and signal flows, actions, attributes, strategies, domain, and structures. Specifically, the ontology was structured by relationships between the system strategy and its functional (flows), behavioral (actions, attributes), structural, and domain concepts. This ontology was then encoded using Description Logics.

With respect to retrieval, mathematical evidence that subsumption in DL ensures consistency in retrieval was presented in Section 5.5.1. Specifically, subsumption in DL can be shown to impose a partial order relation on entities when subsumption is computed. Since subsumption, in general, is found to compute a consistent and correct hieararchy, retrieval through subsumption is also consistent and correct. Precision in subsumption-based retrieval was validated empirically in Section 5.5.3. Specifically, a strategy repository testbed was developed and used to empirically test the precision of subsumption-based retrieval. Test queries were formulated and used to retrieve strategies from the repository. Precision was calculated to be 1 for every query except when no matches were found, therefore, it was concluded that subsumption in DL ensured precise retrieval of biological strategies. Because subsumption in DL was found to enable both consistent and precise retrieval, the use of DL as a structuring mechanism for an ontology of hierarchical Petri net concepts is also justified.

Additionally, in Section 5.5.4, the retrieval performance of subsumption-based retrieval was compared to that of the Biomimicry Database and the Functional Keyword

Search. The performance of the alternatives were compared using a retrieval effectiveness score, F_1. In the study, the retrieval effectiveness was found to be less than 1 for both the Biomimicry Database and the Functional Keyword Search. This was compared to an effectiveness score of 1 for subsumption-based retrieval. Because of this, it was concluded that subsumption-based retrieval was useful.

9.1.4 Research Question and Hypothesis 4

In this research, bio-inspired design is used in two different contexts: problem-based and solution-driven Conceptual Design. Until this point, there has been very little research in quantifying its impact on the designer in Conceptual Design. This motivates the following research question:

Question 4: *What is the impact of biological strategies in the Conceptual Design process?*

In answering this question, it was hypothesized in Hypothesis 4 that

Hypothesis 4: *4(a)* Exposure to biological strategies will increase the novelty of design ideas generated and *4(b)* will increase the variety of design ideas generated. Additionally, *4(c*) bio-inspired engineering systems possessing a deeper level of biological system behavior will perform better than those possessing superficial behavioral similarities.

Hypotheses 4(a) and *4(b)* were validated in the context of problem-based Conceptual Design. In Section 7.1, two experimental studies were presented in which mechanical engineering students were exposed to biological examples in the idea generation process; these results were then compared to participants receiving no examples and to those receiving human-engineered examples. In Section 7.1.4, exposure to biological examples was found to increase the novelty of design ideas generated after exposure, thus providing support for *Hypothesis 4(a).* This result agrees with others found in literature where distant analogies (biological systems) have been positively

related to more original designs [43] and been considered the main drivers of truly innovative thought [46]. Additionally, exposure to biological examples was found to preserve the variety of design ideas generated. This result did not support *Hypothesis 4(b)*, where it was hypothesized that the variety of ideas would increase. Although no support for *Hypothesis 4(b)* was found, the results were still favorable. This contradicts results found in the literature [149, 150, 48, 152, 151, 153] where exposure to design examples has been shown to have conformity or fixation effects on the resulting design. In Section 7.2, these cognitive studies were followed with a comprehensive example problem-based Bio-inspired Conceptual Design. In this example, a hybrid bullet resistant armor system was designed and compared to other designs in industry. Based on a survey of other technologies, the system was found to be novel.

Hypothesis 4(c) was validated in the solution-driven Conceptual Design context. In Section 8.1, historical case studies were performed on bio-inspired engineering systems, including that of aviation and renal replacement therapy. It was found that the bio-inspired systems possessing a deeper level of behavioral similarity did indeed perform better than those possessing only superficial similarities. Specifically, in Section 8.1.1, aircraft wings possessing more similarity to the actual behavior of the bird wing perform better with respect to manipulating lift for flight control. Also, in Section 8.1.2, hemodiafiltration, which possesses more behavioral similarity with the human kidney than hemodialysis, was also found to perform better largely because of this similarity. It was concluded that advances in both aviation and renal replacement therapy were a result of a better understanding of and similarity to their respective source biological systems. In Section 8.2, the historical case studies were followed by a comprehensive example solution-based Bio-Inspired Conceptual Design. In this case, a novel renal replacement therapy closely mimicking the behavior of the kidney was designed. This concept was compared to other renal replacement therapies and was hypothesized to have many performance benefits.

A summary of the hypothesis validation performed in this dissertation is displayed in Table 9.1.

Table 9.1 Summary of Hypothesis Validation

	Hypothesis	Validation Tests
Hyp 1	A representation based on (1) a causal behavioral description and (2) hierarchical Petri nets can be used to model the behavior of biological systems	- Qualitative evaluation with respect to requirements put forth for representation of biological systems ***(Chapter 3)***
Hyp 2	Using the systematic method for Reverse Engineering Biological Systems will insure that the fundamental properties of boundedness, reachability, and liveness will be preserved across hierarchical levels.	Find mathematical evidence of boundedness, reachibility, and liveness for hiearchical Petri net representation ***(Chapter 4)***
Hyp 3	An ontology of concepts from hierarchical Petri net representations of biological systems can be represented using Description Logics. Subsumption in Description Logics will enable consistent and precise retrieval of relevant biological strategies from a knowledge repository.	- Find mathematical evidence of consistency through subsumption ***(Chapter 5)*** - Evaluate retrieval precision in various scenarios using test queries ***(Chapter 5)***
Hyp 4a	Exposure to biological strategies will increase the novelty of design ideas generated and will increase the variety of design ideas generated and	- Cognitive studies on mechanical engineering students ***(Chapter 7)*** - Problem-based Conceptual Design Example (Design of Hybrid Bullet-Resistant Armor) ***(Chapter 7)***
Hyp 4b	Bio-inspired engineering systems possessing a deeper level of biological system behavior will perform better than those possessing superficial behavioral similarities.	- Historical case studies on bio-inspired design ***(Chapter 8)*** - Solution-driven Conceptual Design Example (Design of a wearable, artificial kidney) ***(Chapter 8)***

As seen in Table 9.1 and the prior discussion, each of the hypotheses was thoroughly tested and validated. The second part of the validation strategy in this dissertation involves the Validation Square, which is discussed in Section 9.2.

9.2 VALIDATION SUMMARY

The focus of this dissertation is the proposed method for Reverse Engineering Biological systems and its accompanying hierarchical Petri net representation, therefore,

validation of the method as a whole is also of primary importance. To do so, the Validation Square [190] is employed. This validation strategy is presented in the following sections.

9.2.1 Theoretical Structural Validity (TSV)

The first step in validating the method for Reverse Engineering Biological Systems is evaluating the theoretical structural validity of the proposed method. TSV involves checking the individual constructs and assumptions upon which the method is built, as well as checking the internal consistency of the method.

In Chapter 2, the theoretical foundations of the method for Reverse Engineering Biological Systems and the Strategy Repository were validated through review of the relevant literature. In Section 2.1, relevant literature on systematic design and idea generation was reviewed. In Section 2.2, representations in engineering design were reviewed. This review included general models of cognitive processes in idea generation, mental models, and common representations used in engineering design. The foundations of the strategy repository, engineering ontologies and Description Logics were reviewed in Section 2.3. In Section 2.4, metrics for empirical evaluation of idea generation techniques were reviewed.

In Chapter 3, the foundations of the specific representation used in the proposed method, the hierarchical Petri net representation, were reviewed. Specifically, a rigorous assessment of both engineering representations and representation expression against several key requirements for representing biological systems was presented.

The latter part of TSV involves a check of the internal consistency of the method. In Chapter 4, the Method for Reverse Engineering Biological Systems was presented. The check of internal consistency was performed using the systematic steps and process flowchart presented in Section 4.2.

9.2.2 Empirical Structural Validity

Empirical Structural Validity involves accepting the appropriateness of the example problems that are used to verify the method performance. In this research, the

Method for Reverse Engineering is used in both a problem-driven and the solution-based conceptual design context. In the problem-based approach, the designer begins with an engineering problem and searches for solutions to this problem through the engineering design process. In the solution-driven approach, the designer begins with a biological solution and attempts to mimic the behavior of this system in the engineering domain. Example problems were developed for each approach. In Section 6.1.3, the cognitive studies (Section 7.1) and the comprehensive example on bullet resistant armor development (Section 7.2) were both accepted as appropriate to verify the performance of the proposed method in the problem-driven context. These studies test the impact of the proposed method when using biological strategies in the search for solutions. In Section 6.2.2, the historical case studies presented in Section 8.1 and the comprehensive example on renal replacement therapy development presented in Section 8.2 were both accepted as appropriate for testing the proposed method in the solution-driven context. These studies test the impact of the proposed method in aiding the designer in designing engineering systems that successfully mimic novel biological behavior.

The ESV of the strategy repository is considered in Chapter 5. The repository structure and retrieval method were tested using a testbed repository (Section 5.5.2.1). Specifically, test queries to the testbed were used to test the precision of subsumption in DL. The queries used to test the retrieval method were structured after typical requests made by designers, thus the test method is deemed appropriate.

9.2.3 Empirical Performance Validation

Empirical Performance Validation involves accepting the usefulness of the method for some representative example problems. In this research, the goal of the proposed method is to aid the designer in generating ideas in Conceptual Design using biological strategies. Therefore, the usefulness of the method is measured by its ability to aid the designer in generating a large variety of novel solutions (in the problem-based approach) and in generating quality solutions (in the solution-driven approach).

In the problem-based approach, the proposed method was tested using cognitive studies (Section 7.1) and a comprehensive example of the design of hybrid, bullet

resistant armor (Section 7.2). In the cognitive studies, the biological strategies were found to aid the designer in generating novel solutions while preserving the variety of design ideas generated. In the example problem, the designs generated using the proposed method were found to be novel relative to other solutions currently found in the market.

In the solution-driven context, the proposed method was tested using historical case studies (Section 8.1) and a comprehensive example of the design of a novel renal replacement therapy (Section 8.2). In the historical case studies, bio-inspired systems that more closely mimicked the behavior of the target biological system were found to perform better. In the comprehensive example , several advantages of the design generated were found over current renal therapies.

With respect to the strategy repository, the precision of subsumption-based retrieval was tested using test queries and a repository testbed (Chapter 5). Retrieval using subsumption in DL was found to have a precision of 1. This was found to have a significant advantage over keyword-based retrieval methods, which typically have a precision less than 1.

9.2.4 Theoretical Performance Validation

Success in the previous validation steps helps to build a case for this generality. Although a case can be made for generality, every validation strategy relies ultimately on a "leap of faith" [190].

The scope of the proposed method and strategy repository is idea generation in Conceptual Design. Theoretical Performance Validation involves building confidence in the generality of the method and its usefulness beyond the example problems. The goal of the proposed method and repository is to aid the designer in generating a large variety of novel solutions (in the problem-based Conceptual Design approach) and quality design solutions (in the solution-driven Conceptual Design approach). Theoretical Performance Validity is inferred from the Theoretical Structural Validity (TSV), Empirical Structural Validity (ESV), and Empirical Performance Validity (EPV) presented throughout this dissertation. Specifically, with TSV, the individual constructs of the proposed method

were accepted, as well as demonstrated the internal consistency of the way the constructs were put together in the method. With ESV, the example problems (cognitive studies case studies, and comprehensive examples) were found to be appropriate for testing the proposed method in both the problem-based and solution-driven Conceptual Design context. With respect to EPV, the proposed method and repository were found to be useful for the cognitive studies and case studies presented. Collectively, these validation steps show that the method is useful within the scope of the example problems presented.

In extending the usefulness of the proposed method beyond the scope of the example problems, the general class of problems in which the method is useful is examined. In the problem-based approach, the general class of problems has the following characteristics:

1) Problem-based, meaning the designer begins with a design problem and proceeds through the design process systematically in the search for solutions
2) Open solution field – there are multiple solutions that can satisfy a given problem
3) The designer wishes to be inspired by biological strategies, implying a transfer of strategy at a high level of abstraction
4) Novelty of solution is valued

Given the general class of problems, the proposed method can be extended beyond the scope of example problems and generalized for applicability in all problem-based Conceptual Design scenarios.

With respect to the solution-driven approach, the general class of problems has the following characteristics:

1) Solution-driven, meaning a reverse engineering approach is followed in mimicking a novel feature or behavior from an analogous system
2) Target system can be systematically decomposed
3) Quality of solution is valued
4) The designer has the initial time to invest in reverse engineering an analogous system

Given this general class of problems, the proposed method can be extended beyond the scope of the example problems and generalized for applicability in all solution-driven Conceptual Design scenarios.

9.3 REVIEW OF RESEARCH GAP AND CONTRIBUTIONS

Based on the review of current approaches to bio-inspired design reviewed in Section 1.2.2, several gaps were identified. These gaps include:

1) Biological representations to bridge the gap between biological and engineering domains
2) Systematic method to guide decomposition and strategy extraction
3) Efficient retrieval of relevant solutions to aid in identifying relevant solutions
4) Lack of empirical evidence to support bio-inspired design.

Biological System Representation

In the reviewed approaches, a significant gap was identified between the biological and engineering research being performed by biologists in their respective fields and the analogical translation that aims to transfer knowledge from biology to engineering. To bridge this gap between biology and engineering, a hierarchical Petri net representation for biological systems was developed. This representation offers a mathematically-founded representation that allows qualitative simulation of biological system behavior. To aid the engineer in understanding biological phenomena, this representation also offers a multi-leveled view of system behavior, allowing both behavior abstraction and refinement in a single model.

The hierarchical Petri net representation was used to structure an ontology of biological concepts. This ontology, when implemented in a repository, is used to aid the identification and retrieval of relevant biological strategies to stimulate idea generation. For engineers seeking to mimic novel biological behavior, positive correlation was found between the performance and the level of biological similarity of bio-inspired systems. Given that, there is value in rich behavioral models of biological system behavior, such as

the hierarchical Petri net representation. The proposed method aids in systematically decomposing the biological system behavior and creating these rich behavioral models.

Systematic Decomposition

Systematic and consistent decomposition of the behavior of biological systems is key to extracting complete and correct biological strategies. Along with hierarchical representation, this systematic decomposition is key to bridging the gap between biological knowledge and engineering design. To address the gap, the Method for Reverse Engineering Biological Systems was developed as a means to aid the engineer in systematically decomposing biological systems using the hierarchical Petri net representation and extracting behavioral strategies from these systems. This method was found to have value for both engineers seeking to be inspired by biological strategies and those attempting to mimic them. For those seeking bio-inspiration, the biological strategies extracted using the proposed method were found to aid designers in generating novel design solutions, without sacrificing the variety of these ideas (as in the case of human-engineered strategies).

Retrieval of relevant solutions

Efficient identification of relevant biological strategies to use in Conceptual Design is key to harnessing biological technologies in engineering, however, this identification is one of the most difficult tasks in bio-inspired design. In this research, a strategy repository was developed to aid in the identification of relevant biological strategies. This repository was structured using an engineering ontology of concepts from the hierarchical Petri net representation and implemented using Description Logics. Subsumption in Description Logics was shown to enable consistent and precise retrieval of biological strategies. This consistency and precision in retrieval can aid in reducing the time needed for a secondary "weeding" process of irrelevant results. Although the current approaches to biological databases offer access to these biological solutions, the generic keyword-based retrieval mechanisms utilized by these approaches often suffer from providing either too many and/or irrelevant results [30]. When compared to the keyword-

based search strategies utilized by the Biomimicry Database and the Functional Keyword search, subsumption-based retrieval was shown to outperform these keyword-based strategies in retrieval effectiveness. Effectiveness was defined by a combined metric for both retrieval precision and recall.

Lack of empirical evidence to support bio-inspired design

Although significant advantages of bio-inspired design have been theorized by examining scattered examples of successful cases, there has been a lack of research on how the use of biological strategies in engineering impacts the designer and the products that follow. In this research, cognitive studies were performed on engineering students exposed to biological strategies in the idea generation process. These results were then compared to students exposed to either no strategies or human-engineered strategies in idea generation. Students exposed to biological strategies showed a significant increase in the novelty of their design ideas after exposure, while no significant difference was found between the variety of the design ideas produced before and after exposure. This is a significant result since examples in idea generation have been shown to typically be fixating, which would reduce the variety of design ideas produced.

9.4 RESEARCH LIMITATIONS

Although there are several advantages to using the research presented in this dissertation, it does not come without its limitations. In this section, the main limitations of this research are discussed. These limitations open up avenues for future work, which is also discussed in this section.

The first limitation to this research relates to the hierarchical Petri net representation and the Method for Reverse Engineering Biological Systems. The main limitation of this representation, as is the case for all representations, is that the quality of the information that can be extracted from the representation is strongly dependent on the quality of information used to build the representation. In other words, "Garbage in = Garbage out". The proposed method is used as a means for facilitating systematic and consistent behavior decomposition. Although the proposed method aids in systematizing

the decomposition and strategy extraction process, the strategy extracted from the representation is only as correct and complete as the information used to build it. Because of this, it is recommended that persons with a modest level of biological knowledge construct these representations. However, since this approach uses a behavior-centric approach to representing the system, the problem of subjectivity that usually plagues function-based representations is alleviated.

The second limitation to this research relates to the Strategy Repository. The Strategy Repository is constructed using defined taxonomies of concepts for functional, behavioral, and structural information. Currently, the user is restricted to this same vocabulary for querying the repository. It should be noted that although the structured queries increase the performance of the retrieval strategy, they lack the expressiveness found in natural language-based systems. The tradeoff between retrieval performance and query expressiveness can be better managed using grammar processing systems, such as Wordnet [27]. These grammar processing systems can be used to expand the query space and increase the expressiveness of the retrieval strategy. Currently, Sungshik Yim is working on grammar templates using Wordnet to help improve the expressiveness of the queries.

The size of the repository is also a limitation. In order for the repository to become truly useful, it needs to be populated much more than its current state. Populating the repository to a level that extends beyond the current scope will take the efforts of many different individuals and research groups. We believe that this limitation can be overcome using a crowd-sourcing scheme, similar to that of Wikipedia. By allowing the repository to be open to and governed by the users, the repository can become populated very quickly. This will also aid in reducing the upfront cost of this type of repository.

The final limitation comes by way of the cognitive studies performed. In these studies, a relatively small sample of Mechanical Engineering students at Georgia Institute of Technology were used. This small sample size limits the level of generality that can be claimed from the studies. As part of future work, these cognitive studies should be

performed on a larger set of diverse participants. This will help expand further the case for value in bio-inspired design.

9.5 BEYOND TOMORROW (FUTURE WORK)

In Section 9.4, the current limitations of the work are addressed. In this section, we look beyond these limitations and propose several research directions that will aid in completing this work. The overall vision of the proposed method for Reverse Engineering Biological Systems and the Strategy Repository is that of an ideation tool, similar to that of brainstorming and the Delphi method (others reviewed in Section 2.2). As with the other idea generation tools, these tools can be used to aid the designer in systematically expanding and exploring his/her design space. These tools can also be used beyond the current scope of biological systems. Specifically, the proposed method can be used beyond the current scope of biological systems as a means for decomposing the function, structure, and behavior of any type of physical system. In addition, the repository can be used to store and retrieve knowledge and design strategies outside the current scope of biological systems. The strategy repository is seen as a tool to bridge the knowledge gap between many otherwise distant domains. These domains can include the electrical, chemical, and materials domains.

Given this overall vision for the research, there are several research issues that must be addressed to build upon its current state. These future research directions are focused in the areas of method validation, the expressiveness of the retrieval algorithm, systematically increasing variety, and model simulation. The first area identified for future research is that of method validation. In the current research, the method for REBS, as a whole, was not validated. To truly prove value, an unbiased comparison of this method versus others for decomposing biological systems must be performed. This study should be performed in a controlled environment using either students or design/engineering professionals.

Another research area identified deals with the expressiveness of the DL used to build the repository. In its current state, strategies are retrieved from the repository by using a very limited vocabulary, which is also used to build the repository. Although this

method is useful with respect to retrieval performance, it lacks in allowing a natural expression of a search query by the user. Future work should include the use of grammar processing templates, which allow the use of natural language sentence formulation for querying the repository, but then convert this query to one that is understandable for DL retrieval. Also, the Description Logic $\mathcal{ALE}$ was used in this research. Although not as expressive as other attribute languages, $\mathcal{ALE}$ is also less computationally complex than the more expressive languages. Future work in this area should include the use of a more expressive DL, while also reducing the computational complexity of the inference algorithms.

Extending the method for increasing the variety of the design ideas generated using biological strategies is also an area of future work. In this work, Hypothesis 4b (increased variety of design ideas) was not validated. It is believed that this hypothesis was not validated due to the use of only one biological strategy in the studies. Variety can be increased through the use of multiple design strategies as inspiration in idea generation. To test this hypothesis, studies must be conducted with the number of design strategies used as an experimental factor. Based on these studies, the proposed method can be extended to aid the designer in systematically increasing the variety of his/her design ideas.

Lastly, behavioral simulation is also an area of future work. Currently, the hierarchical Petri net representation is used to represent the behavior of biological systems. One of the advantages of this type of representation is that it allows the hierarchical and dynamic simulation of biological system behavior. The systems modeled in this dissertation were simple enough to model and simulate manually. However, with more complex biological sytems, software tools are needed to aid in producing and simulating these hierarchical and dynamic representations and handling the complexity usually involved in modeling complex biological systems. Adding computer support and simulation capabilities to the representation can give an easy way of building and interacting with these hierarchical and dynamic models. Two software tools that hold promise are STELLA [191] and SysML [192], which allow for formal and dynamic modeling of physical systems. STELLA (Structural Thinking Experimental

Learning Laboratory with Animation) utilizes an icon-based graphical interface to model and simulate system behavior. Using the software, STELLA allows the simulation of discrete and continuous system behavior using user-defined mathematical relationships and mappings. The main advantage of this type of model is the simple, easy-to-use interface used to build and simulate the system behavior. This software also allows multi-leveled, hierarchical system modeling and simulation. This will allow hierarchical Petri net models of biological systems to be built and simulated using the software.

Another possible tool is SysML, which is a generic modeling language made for modeling systems and processes. SysML allows the modeling of systems using different types of diagrams, including structure diagrams, behavioral diagrams, parametric diagrams, and requirement diagrams. A generic, top-level SysML model is displayed in Figure 9.1.

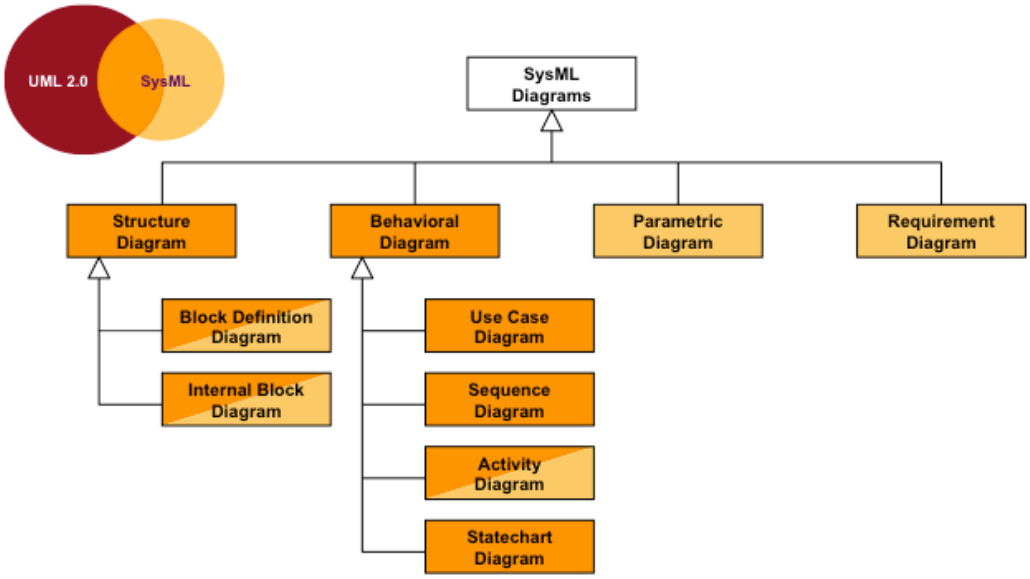


Figure 9.1 SysML model diagram [192]

In SysML, structure diagrams are designed to represent the physical components of the systems, as well as the relationships between these components. Behavioral diagrams utilize state diagrams, use case diagrams, activity diagrams, and sequence diagrams to represent the behavior of the system. Requirement diagrams are used show the different relationships between the requirements of the system. In essence, SysML allows one to model and simulate the functional, behavioral, and structural aspects of these systems.

Although providing easy user interfaces and complete system modeling, it should be noted that neither STELLA nor SysML currently enable the ability to represent system behavior using hierarchical Petri nets. However, there are several dedicated programs for modeling hierarchical Petri nets. One such program is QPME (Queueing Petri net Modeling Environment) [193], developed by the Databases and Distributed Systems Group at the Technische Universitat Darmstadt. QPME offers a simple, graphical software tool for modeling Petri nets and allows one to analyze the formal properties these systems. The primary advantage of QPME is its ability to handle and analyze very large and complex systems. QPME also supports the modeling of hierarchical (queueing) Petri nets and allows simulation across multiple levels of hierarchy. QPME is made of two primary components, a hierarchical Petri net editor (QPE) and a simulator (SimQPN). QPE provides the graphical interface for modeling the system behavior, while SimQPN is a portable and platform-independent simulation tool for simulating the net [193]. A screenshot of the QPE interface is displayed in Figure 9.2.

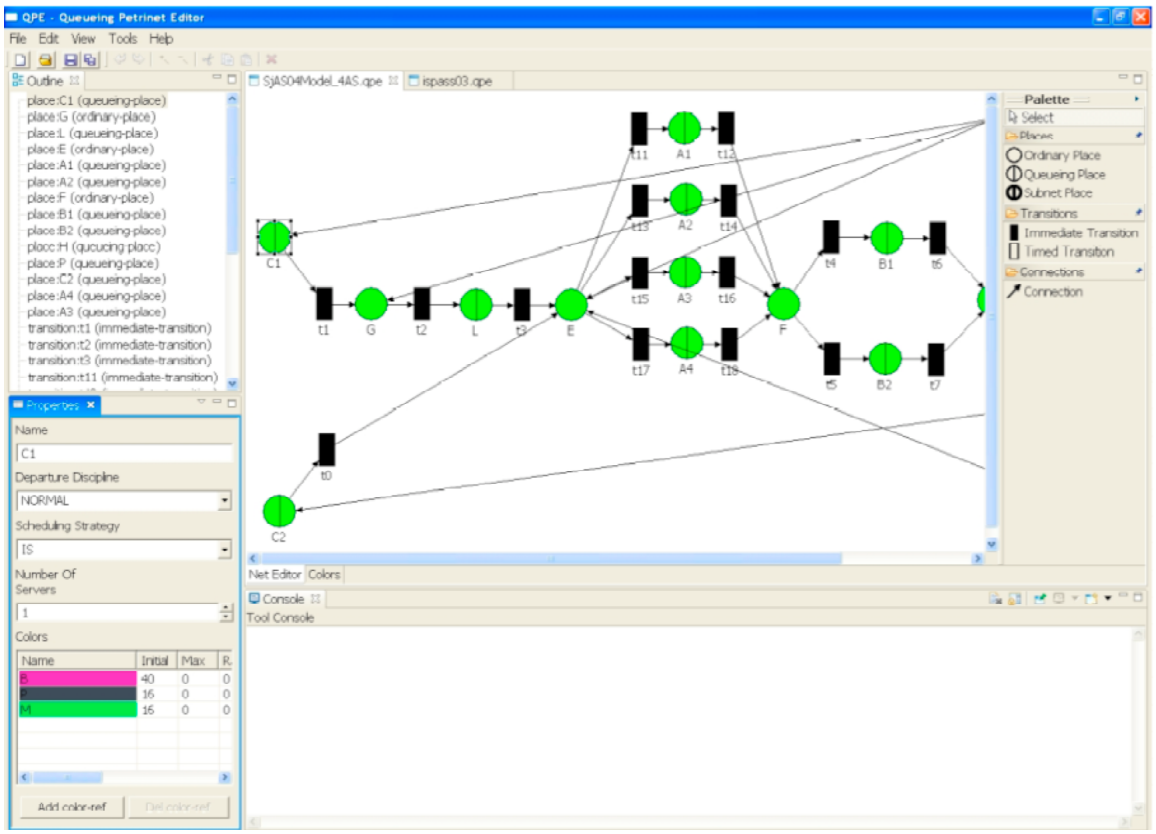


Figure 9.2 QPME Interface [193]

9.6 CLOSING THOUGHTS

In closing, I present a few remarks on the 'value' of the research presented in this thesis. As presented earlier, this work presented in this dissertation is used to aid the designer in more "effective" idea generation in Conceptual Design. This research was focused on developing a hierarchical Petri net representation and method for Reverse Engineering Biological Systems to aid in bridging the gap between the biological and engineering domains and developing a repository and retrieval method to aid in identifying biological strategies.

This question of concern in this closing section is, "*What is the value of the work presented in this dissertation to the businesses and engineers that seek novel and innovative solutions?*" Value can be defined as 'benefit' divided by 'cost'. In the context of design, 'benefit' considers the added advantage to using the tools presented in this research and 'cost' considers such factors as time that typically plague the design process.

In the problem-based approach to Bio-inspired Conceptual Design, the strategy repository is used as means to identify relevant biological strategies in Conceptual Design. These strategies are then used to stimulate ideas for new and innovative concepts for design. With respect to the benefit of this research, these strategies were shown to increase the novelty of the design ideas produced without sacrificing the variety of design ideas produced. This increase in novelty can be correlated to a broadening of the designer's design space. A broad design space provides a better opportunity for developing a truly innovative and winning design. With respect to cost, the majority of the cost is taken upfront in populating the repository. Although this cost may be significant, the designer utilizing the repository is shielded from this responsibility.

In the solution-driven approach, the method for Reverse Engineering Biological Systems is used to decompose biological system behavior and extract biological strategies. The true benefit of the proposed method is that it gives a systematic way of handling the complexity found in biological systems and guiding the designer more quickly and directly through behavioral decomposition and strategy extraction. The cost of the proposed method also relates to the systematic nature of the method. The systematic nature of the method can sometimes be more time-consuming and labor-intensive than ad hoc approaches. However, the benefits of the systematic procedure are argued to outweigh the costs.

In closing, the primary conclusions of this research can be summarized as follows:

- A new representation framework for biological systems was developed. Considering several requirements from both the psychological and design domain, the hierarchical Petri net representation was found to be acceptable for representing the physical behavior of biological systems.
- The Method for Reverse Engineering Biological Systems was shown to facilitate systematic decomposition of the behavior of biological systems.
- The Method for Reverse Engineering Biological Systems was shown to preserve reachability, boundedness, and liveness across hierarchical levels of

the representation. This enables consistency in behavioral representation and strategy extraction.

- Exposure to biological strategies in the design process was shown to increase the novelty of design ideas generated for Mechanical Engineering students, thus increasing the likelihood of developing a novel solution.
- Exposure to biological strategies in the design process was shown to preserve the variety of design ideas generated for Mechanical Engineering students, thus negating the fixation effects typically associated with exposure in idea generation.
- Using historical case studies, bio-inspired designs showing closer similarity to their analogous biological systems were shown to outperform those with less similarity.
- Subsumption in Description Logics was shown to enable consistent and precise retrieval of biological strategies.
- When compared to the keyword-based retrieval strategies used in current approaches to biological strategy retrieval, subsumption-based retrieval outperformed these retrieval strategies with respect to retrieval effectiveness.
- Description Logics was found to be useful in structuring an ontology of biological and engineering concepts.

APPENDIX A – REPOSITORY USER INTERFACE CODE

In Appendix A, the computer code used to build the user interface for the strategy repository is presented. The user interface to the repository was programmed by Patrick Chang, a graduate student in Mechanical Engineering at Georgia Institute of Technology, under the advisement of Jamal Wilson and David Rosen.

Form CLASS

```
using System;
using System.IO;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Xml;
using System.Xml.XPath;

namespace RepositoryGUI
{
   public partial class GUI : Form
   {
      #region Variables

      InitializeKnowledgeBase kb;
      public string uri;
```

```
public string reasonerAddress;
public string input;
public string output;
public string action;
public string attribute;
public string structure;
public string domain;
public string strategy;

#endregion

#region Constructor

public GUI()
{
   InitializeComponent();
   reasonerAddress = "http://localhost:8080";
   kb = new InitializeKnowledgeBase(reasonerAddress);
   uri = "";
   input = "";
   output = "";
   action = "";
   attribute = "";
   structure = "";
   domain = "";
   strategy = "";
}

#endregion
```

```
#region Form Startup Method

private void GUI_Load(object sender, EventArgs e)
{
   try
   {
      kb.newKB();
   }
   catch
   {
      MessageBox.Show("ERROR: Cannot find RacerPro! Please open RacerPro before running this program.", "Error Message");
      Application.Exit();
   }

   input_comboBox.Items.Clear();
   output_comboBox.Items.Clear();
   action_comboBox.Items.Clear();
   attribute_comboBox.Items.Clear();
   structure_comboBox.Items.Clear();
   strategy_comboBox.Items.Clear();

   input_comboBox.Items.Add("No Input Selected");
   input_comboBox.Items.Add("Energy_Flow");
   input_comboBox.Items.Add("   Acoustic");
   input_comboBox.Items.Add("      Particle_Velocity");
   input_comboBox.Items.Add("      Pressure");
   input_comboBox.Items.Add("   Biological");
   input_comboBox.Items.Add("      ParticleVelocity");
   input_comboBox.Items.Add("      PressureBiol");
```

```
input_comboBox.Items.Add("   Chemical");
input_comboBox.Items.Add("      Affinity");
input_comboBox.Items.Add("      ReactionRate");
input_comboBox.Items.Add("   Electrical");
input_comboBox.Items.Add("      Current");
input_comboBox.Items.Add("      ElectromotiveForce");
input_comboBox.Items.Add("   Electromagnetic");
input_comboBox.Items.Add("      Optical");
input_comboBox.Items.Add("         Intensity_Optical");
input_comboBox.Items.Add("         Velocity_Optical");
input_comboBox.Items.Add("      Solar");
input_comboBox.Items.Add("         Intensity_Solar");
input_comboBox.Items.Add("         Velocity_Solar");
input_comboBox.Items.Add("   Human");
input_comboBox.Items.Add("      Force_Human");
input_comboBox.Items.Add("      Velocity_Human");
input_comboBox.Items.Add("   Hydraulic");
input_comboBox.Items.Add("      Pressure_Hydraulic");
input_comboBox.Items.Add("      VolumetricFlow_Hydraulic");
input_comboBox.Items.Add("   Magnetic");
input_comboBox.Items.Add("      MagneticFluxRate");
input_comboBox.Items.Add("      MagnetomotiveForce");
input_comboBox.Items.Add("   Mechanical");
input_comboBox.Items.Add("      Rotational");
input_comboBox.Items.Add("         Torque");
input_comboBox.Items.Add("         Velocity");
input_comboBox.Items.Add("      Translational");
input_comboBox.Items.Add("         Force");
input_comboBox.Items.Add("         LinearVelocity");
input_comboBox.Items.Add("   Pneumatic");
```

```
input_comboBox.Items.Add("     MassFlow_Pneumatic");
input_comboBox.Items.Add("     Pressure_Pneumatic");
input_comboBox.Items.Add("   Radioactive_Nuclear");
input_comboBox.Items.Add("     DecayRate");
input_comboBox.Items.Add("     Intensity");
input_comboBox.Items.Add("   Thermal");
input_comboBox.Items.Add("     HeatFlow");
input_comboBox.Items.Add("     Temperature");
input_comboBox.Items.Add("Material_Flow");
input_comboBox.Items.Add("   Gas_Material");
input_comboBox.Items.Add("   Human_Material");
input_comboBox.Items.Add("   Liquid_Material");
input_comboBox.Items.Add("   Mixture_Material");
input_comboBox.Items.Add("     Colloidial");
input_comboBox.Items.Add("     Gas-Gas");
input_comboBox.Items.Add("     Liquid-Gas");
input_comboBox.Items.Add("     Liquid-Liquid");
input_comboBox.Items.Add("     Solid-Gas");
input_comboBox.Items.Add("     Solid-Liquid");
input_comboBox.Items.Add("     Solid-Liquid-Gas");
input_comboBox.Items.Add("     Solid-Solid");
input_comboBox.Items.Add("   Plasma_Material");
input_comboBox.Items.Add("   Solid_Material");
input_comboBox.Items.Add("     Composite");
input_comboBox.Items.Add("     Object");
input_comboBox.Items.Add("     Particulate");
input_comboBox.Items.Add("Signal_Flow");
input_comboBox.Items.Add("   Control_Signal");
input_comboBox.Items.Add("     Analog");
input_comboBox.Items.Add("     Discrete");
```

```
input_comboBox.Items.Add("   Status_Signal");
input_comboBox.Items.Add("      Auditory");
input_comboBox.Items.Add("      Olfactory");
input_comboBox.Items.Add("      Tactile");
input_comboBox.Items.Add("      Taste");
input_comboBox.Items.Add("      Visual");

output_comboBox.Items.Add("No Output Selected");
output_comboBox.Items.Add("Energy_Flow");
output_comboBox.Items.Add("   Acoustic");
output_comboBox.Items.Add("      Particle_Velocity");
output_comboBox.Items.Add("      Pressure");
output_comboBox.Items.Add("   Biological");
output_comboBox.Items.Add("      ParticleVelocity");
output_comboBox.Items.Add("      PressureBiol");
output_comboBox.Items.Add("   Chemical");
output_comboBox.Items.Add("      Affinity");
output_comboBox.Items.Add("      ReactionRate");
output_comboBox.Items.Add("   Electrical");
output_comboBox.Items.Add("      Current");
output_comboBox.Items.Add("      ElectromotiveForce");
output_comboBox.Items.Add("   Electromagnetic");
output_comboBox.Items.Add("      Optical");
output_comboBox.Items.Add("         Intensity_Optical");
output_comboBox.Items.Add("         Velocity_Optical");
output_comboBox.Items.Add("      Solar");
output_comboBox.Items.Add("         Intensity_Solar");
output_comboBox.Items.Add("         Velocity_Solar");
output_comboBox.Items.Add("   Human");
output_comboBox.Items.Add("      Force_Human");
```

```
output_comboBox.Items.Add("     Velocity_Human");
output_comboBox.Items.Add("   Hydraulic");
output_comboBox.Items.Add("     Pressure_Hydraulic");
output_comboBox.Items.Add("     VolumetricFlow_Hydraulic");
output_comboBox.Items.Add("   Magnetic");
output_comboBox.Items.Add("     MagneticFluxRate");
output_comboBox.Items.Add("     MagnetomotiveForce");
output_comboBox.Items.Add("   Mechanical");
output_comboBox.Items.Add("     Rotational");
output_comboBox.Items.Add("       Torque");
output_comboBox.Items.Add("       Velocity");
output_comboBox.Items.Add("     Translational");
output_comboBox.Items.Add("       Force");
output_comboBox.Items.Add("       LinearVelocity");
output_comboBox.Items.Add("   Pneumatic");
output_comboBox.Items.Add("     MassFlow_Pneumatic");
output_comboBox.Items.Add("     Pressure_Pneumatic");
output_comboBox.Items.Add("   Radioactive_Nuclear");
output_comboBox.Items.Add("     DecayRate");
output_comboBox.Items.Add("     Intensity");
output_comboBox.Items.Add("   Thermal");
output_comboBox.Items.Add("     HeatFlow");
output_comboBox.Items.Add("     Temperature");
output_comboBox.Items.Add("Material_Flow");
output_comboBox.Items.Add("   Gas_Material");
output_comboBox.Items.Add("   Human_Material");
output_comboBox.Items.Add("   Liquid_Material");
output_comboBox.Items.Add("   Mixture_Material");
output_comboBox.Items.Add("     Colloidial");
output_comboBox.Items.Add("     Gas-Gas");
```

```
output_comboBox.Items.Add("      Liquid-Gas");
output_comboBox.Items.Add("      Liquid-Liquid");
output_comboBox.Items.Add("      Solid-Gas");
output_comboBox.Items.Add("      Solid-Liquid");
output_comboBox.Items.Add("      Solid-Liquid-Gas");
output_comboBox.Items.Add("      Solid-Solid");
output_comboBox.Items.Add("   Plasma_Material");
output_comboBox.Items.Add("   Solid_Material");
output_comboBox.Items.Add("      Composite");
output_comboBox.Items.Add("      Object");
output_comboBox.Items.Add("      Particulate");
output_comboBox.Items.Add("Signal_Flow");
output_comboBox.Items.Add("   Control_Signal");
output_comboBox.Items.Add("      Analog");
output_comboBox.Items.Add("      Discrete");
output_comboBox.Items.Add("   Status_Signal");
output_comboBox.Items.Add("      Auditory");
output_comboBox.Items.Add("      Olfactory");
output_comboBox.Items.Add("      Tactile");
output_comboBox.Items.Add("      Taste");
output_comboBox.Items.Add("      Visual");

action_comboBox.Items.Add("No Action Selected");
action_comboBox.Items.Add("Branch");
action_comboBox.Items.Add("   Distribute");
action_comboBox.Items.Add("   Seperate");
action_comboBox.Items.Add("      Divide");
action_comboBox.Items.Add("      Extract");
action_comboBox.Items.Add("      Remove");
action_comboBox.Items.Add("Channel");
```

```
action_comboBox.Items.Add("   Export");
action_comboBox.Items.Add("   Guide");
action_comboBox.Items.Add("      Rotate");
action_comboBox.Items.Add("      Translate");
action_comboBox.Items.Add("   Import");
action_comboBox.Items.Add("   Transfer");
action_comboBox.Items.Add("      Transmit");
action_comboBox.Items.Add("      Transport");
action_comboBox.Items.Add("Connect");
action_comboBox.Items.Add("   Couple");
action_comboBox.Items.Add("      Join");
action_comboBox.Items.Add("      Link");
action_comboBox.Items.Add("   Mix");
action_comboBox.Items.Add("Control");
action_comboBox.Items.Add("   Actuate");
action_comboBox.Items.Add("   Regulate");
action_comboBox.Items.Add("      Decrease");
action_comboBox.Items.Add("      Increase");
action_comboBox.Items.Add("Convert");
action_comboBox.Items.Add("Magnitude");
action_comboBox.Items.Add("   Change");
action_comboBox.Items.Add("      Condition");
action_comboBox.Items.Add("      Decrement");
action_comboBox.Items.Add("      Increment");
action_comboBox.Items.Add("      Shape");
action_comboBox.Items.Add("   Stop");
action_comboBox.Items.Add("      Inhibit");
action_comboBox.Items.Add("      Shape");
action_comboBox.Items.Add("Provision");
action_comboBox.Items.Add("   Store");
```

```
action_comboBox.Items.Add("     Collect");
action_comboBox.Items.Add("     Contain");
action_comboBox.Items.Add("   Supply");
action_comboBox.Items.Add("Signal");
action_comboBox.Items.Add("   Indicate");
action_comboBox.Items.Add("     Display");
action_comboBox.Items.Add("     Track");
action_comboBox.Items.Add("   Process");
action_comboBox.Items.Add("   Sense");
action_comboBox.Items.Add("     Detect");
action_comboBox.Items.Add("     Measure");
action_comboBox.Items.Add("Support");
action_comboBox.Items.Add("   Position");
action_comboBox.Items.Add("   Secure");
action_comboBox.Items.Add("   Stabilize");

attribute_comboBox.Items.Add("No Attribute Selected");
attribute_comboBox.Items.Add("Physical_Attributes");
attribute_comboBox.Items.Add("   Color");
attribute_comboBox.Items.Add("   Shape-Physical");
attribute_comboBox.Items.Add("   Length");
attribute_comboBox.Items.Add("   Radius");
attribute_comboBox.Items.Add("     Inner_Radius");
attribute_comboBox.Items.Add("     Outer_Radius");
attribute_comboBox.Items.Add("   Diameter");
attribute_comboBox.Items.Add("     Inner_Diameter");
attribute_comboBox.Items.Add("     Outer_Diameter");
attribute_comboBox.Items.Add("   Width");
attribute_comboBox.Items.Add("   Height");
attribute_comboBox.Items.Add("   Thickness");
```

```
attribute_comboBox.Items.Add("   Area");
attribute_comboBox.Items.Add("      Cross_Sectional_Area");
attribute_comboBox.Items.Add("      Surface_Area");
attribute_comboBox.Items.Add("   Volume");
attribute_comboBox.Items.Add("   Mass");
attribute_comboBox.Items.Add("Intrinsic_Attributes");
attribute_comboBox.Items.Add("   Mechanical-Intrinsic");
attribute_comboBox.Items.Add("      Stiffness");
attribute_comboBox.Items.Add("         Axial_Stiffness");
attribute_comboBox.Items.Add("            Modulus_of_Elasticity");
attribute_comboBox.Items.Add("      Rotational_Stiffness");
attribute_comboBox.Items.Add("      Bending_Stiffness");
attribute_comboBox.Items.Add("      Strength");
attribute_comboBox.Items.Add("         Tensile_Strength");
attribute_comboBox.Items.Add("         Yield_Strength");
attribute_comboBox.Items.Add("         Ultimate_Strength");
attribute_comboBox.Items.Add("         Breaking_Strength");
attribute_comboBox.Items.Add("         Compressive_Strength");
attribute_comboBox.Items.Add("         Shear_Strength");
attribute_comboBox.Items.Add("         Torsional_Strength");
attribute_comboBox.Items.Add("      Ductility");
attribute_comboBox.Items.Add("      Malleability");
attribute_comboBox.Items.Add("      Strain");
attribute_comboBox.Items.Add("      Strain_Hardening");
attribute_comboBox.Items.Add("      Hardness");
attribute_comboBox.Items.Add("      Scratch_Hardness");
attribute_comboBox.Items.Add("      Indention_Hardness");
attribute_comboBox.Items.Add("      Rebound_Hardness");
attribute_comboBox.Items.Add("      Toughness");
attribute_comboBox.Items.Add("         Impact_Toughness");
```

```
                attribute_comboBox.Items.Add("    Poisson’s_Ratio");
                attribute_comboBox.Items.Add("    Fatigue_Limit");
                attribute_comboBox.Items.Add("    Permeability-Intrinsic-Mechanical");
                attribute_comboBox.Items.Add("    Porosity");
                attribute_comboBox.Items.Add("    Elasticity");
                attribute_comboBox.Items.Add("    Plasticity");
                attribute_comboBox.Items.Add("    Brittleness");
                attribute_comboBox.Items.Add("    Density");
                attribute_comboBox.Items.Add("      Linear_Density");
                attribute_comboBox.Items.Add("      Area_Density");
                attribute_comboBox.Items.Add("      Volume_Density");
                attribute_comboBox.Items.Add("  Electrical-Intrinsic");
                attribute_comboBox.Items.Add("    Electrical_Conductivity");
                attribute_comboBox.Items.Add("    Electrical_Resistivity");
                attribute_comboBox.Items.Add("    Permittivity");
                attribute_comboBox.Items.Add("    Dielectric_Constant");
                attribute_comboBox.Items.Add("    Dielectric_Strength");
                attribute_comboBox.Items.Add("    Capacitance");
                attribute_comboBox.Items.Add("  Thermodynamic_Intrinsic");
                attribute_comboBox.Items.Add("                Absorptivity-Intrinsic-
Thermodynamic");
                attribute_comboBox.Items.Add("    Conductivity");
                attribute_comboBox.Items.Add("    Diffusivity");
                attribute_comboBox.Items.Add("    Heat_of_Vaporization");
                attribute_comboBox.Items.Add("    Heat_of_Fusion");
                attribute_comboBox.Items.Add("    Entropy");
                attribute_comboBox.Items.Add("    Enthalpy");
                attribute_comboBox.Items.Add("    Emissivity");
                attribute_comboBox.Items.Add("
Coefficient_of_Thermal_Expansion");
```

```
attribute_comboBox.Items.Add("     Specific_Heat");
attribute_comboBox.Items.Add("     Flammability");
attribute_comboBox.Items.Add("     Melting_Point");
attribute_comboBox.Items.Add("     Boiling_Point");
attribute_comboBox.Items.Add("     Triple_Point");
attribute_comboBox.Items.Add("     Flash_Point");
attribute_comboBox.Items.Add("     Curie_Point");
attribute_comboBox.Items.Add("   Chemical-Intrinsic");
attribute_comboBox.Items.Add("     Reactivity");
attribute_comboBox.Items.Add("     Stability");
attribute_comboBox.Items.Add("     Corrosion_Resistance");
attribute_comboBox.Items.Add("     Adhesion");
attribute_comboBox.Items.Add("   Optical-Intrinsic");
attribute_comboBox.Items.Add("     Absorptivity-Intrinsic-Optical");
attribute_comboBox.Items.Add("     Reflectivity");
attribute_comboBox.Items.Add("     Refractive_Index");
attribute_comboBox.Items.Add("     Photosensitivity");
attribute_comboBox.Items.Add("     Luminosity");
attribute_comboBox.Items.Add("   Magnetic-Intrinsic");
attribute_comboBox.Items.Add("     Permeability-Intrinsic-Magnetic");
attribute_comboBox.Items.Add("   Biological-Intrinsic");
attribute_comboBox.Items.Add("     Toxicity");
attribute_comboBox.Items.Add("   Fluid-Intrinsic");
attribute_comboBox.Items.Add("     Viscosity");
attribute_comboBox.Items.Add("       Dynamic_Viscosity");
attribute_comboBox.Items.Add("       Kinematic_Viscosity");
attribute_comboBox.Items.Add("       Volume_Viscosity");
attribute_comboBox.Items.Add("       Bulk_Viscosity");
attribute_comboBox.Items.Add("       Shear_Viscosity");
attribute_comboBox.Items.Add("       Extensional_Viscosity");
```

```
                attribute_comboBox.Items.Add("       Surface_Tension");
                attribute_comboBox.Items.Add("    Dimensionless_Numbers");
                attribute_comboBox.Items.Add("       Cauchy_Number");
                attribute_comboBox.Items.Add("       Euler_Number");
                attribute_comboBox.Items.Add("       Froude_Number");
                attribute_comboBox.Items.Add("       Mach_Number");
                attribute_comboBox.Items.Add("       Strouhal_Number");
                attribute_comboBox.Items.Add("       Weber_Number");
                attribute_comboBox.Items.Add("Extrinsic_Attributes");
                attribute_comboBox.Items.Add("  Mechanical-Extrinsic");
                attribute_comboBox.Items.Add("    Force-Extrinsic-Mechanical");
                attribute_comboBox.Items.Add("       Gravitational_Force");
                attribute_comboBox.Items.Add("       Spring_Force");
                attribute_comboBox.Items.Add("       Normal_Force");
                attribute_comboBox.Items.Add("       Friction_Force");
                attribute_comboBox.Items.Add("       Weight");
                attribute_comboBox.Items.Add("  Electrical-Extrinsic");
                attribute_comboBox.Items.Add("    Current-Extrinsic-Electrical");
                attribute_comboBox.Items.Add("    Resistance-Extrinsic-Electrical");
                attribute_comboBox.Items.Add("    Voltage");
                attribute_comboBox.Items.Add("    Charge");
                attribute_comboBox.Items.Add("    Flux-Extrinsic-Electrical");
                attribute_comboBox.Items.Add("  Thermodynamic-Extrinsic");
                attribute_comboBox.Items.Add("                Temperature-Extrinsic-
Thermodynamic");
                attribute_comboBox.Items.Add("       Autoignition_Temperature");
                attribute_comboBox.Items.Add("       Critical_Temperature");
                attribute_comboBox.Items.Add("                Pressure-Extrinsic-
Thermodynamic");
```

```
                attribute_comboBox.Items.Add("                    Resistance-Extrinsic-
Thermodynamic");
                attribute_comboBox.Items.Add("    Flux-Extrinsic-Thermosynamic");
                attribute_comboBox.Items.Add("  Chemical-Extrinsic");
                attribute_comboBox.Items.Add("    Acidity");
                attribute_comboBox.Items.Add("    Specific_Internal_Surface_Area");
                attribute_comboBox.Items.Add("    Concentration");
                attribute_comboBox.Items.Add("  Optical-Extrinsic");
                attribute_comboBox.Items.Add("    Transmittance");
                attribute_comboBox.Items.Add("    Magnification");
                attribute_comboBox.Items.Add("    Focal_Length");
                attribute_comboBox.Items.Add("    Aperture_Size");
                attribute_comboBox.Items.Add("  Magnetic-Extrinsic");
                attribute_comboBox.Items.Add("    Magnetic_Moment");
                attribute_comboBox.Items.Add("    Magnetization");
                attribute_comboBox.Items.Add("    Intensity_of_Magnetization");
                attribute_comboBox.Items.Add("    Magnetic_Field_Strength");
                attribute_comboBox.Items.Add("    Magnetic_Induction");
                attribute_comboBox.Items.Add("  Acoustic-Extrinsic");
                attribute_comboBox.Items.Add("    Absorption");
                attribute_comboBox.Items.Add("    Frequency");
                attribute_comboBox.Items.Add("
Frequency_of_Damped_Free_Vibration");
                attribute_comboBox.Items.Add("      Excitation_Frequency");
                attribute_comboBox.Items.Add("    Period");
                attribute_comboBox.Items.Add("
Period_of_Free_Damped_Oscillation");
                attribute_comboBox.Items.Add("    Wavelength");
                attribute_comboBox.Items.Add("    Quality");
                attribute_comboBox.Items.Add("    Logarithmic_Decrement");
```

```
attribute_comboBox.Items.Add("   Environmental-Extrinsic");
attribute_comboBox.Items.Add("      Embodied_Energy");
attribute_comboBox.Items.Add("      Embodied_CO2");
attribute_comboBox.Items.Add("      Embodied_Water");
attribute_comboBox.Items.Add("   Fluid-Extrinsic");
attribute_comboBox.Items.Add("      Mass_Rate_of_Flow");
attribute_comboBox.Items.Add("      Head");
attribute_comboBox.Items.Add("      Pressure-Extrinsic-Fluid");
attribute_comboBox.Items.Add("      Volumetric_Flow_Rate");
attribute_comboBox.Items.Add("      Internal_Energy");
attribute_comboBox.Items.Add("      Specific_Weight");
attribute_comboBox.Items.Add("Kinematic_Attributes");
attribute_comboBox.Items.Add("   Position_and_Orientation");
attribute_comboBox.Items.Add("      Angle");
attribute_comboBox.Items.Add("         Roll");
attribute_comboBox.Items.Add("         Pitch");
attribute_comboBox.Items.Add("         Yaw");
attribute_comboBox.Items.Add("      Displacement");
attribute_comboBox.Items.Add("         X-Coordinate");
attribute_comboBox.Items.Add("         Y-Coordinate");
attribute_comboBox.Items.Add("         Z-Coordinate");
attribute_comboBox.Items.Add("      Velocity-Kinematic");
attribute_comboBox.Items.Add("         Linear_Speed");
attribute_comboBox.Items.Add("         Linear_Velocity");
attribute_comboBox.Items.Add("         Angular_Speed");
attribute_comboBox.Items.Add("         Angular_Velocity");
attribute_comboBox.Items.Add("         Acceleration");
attribute_comboBox.Items.Add("         Linear_Acceleration");
attribute_comboBox.Items.Add("         Angular_Acceleration");
attribute_comboBox.Items.Add("Time_Attributes");
```

```
                attribute_comboBox.Items.Add("Energy_Attributes");
                attribute_comboBox.Items.Add("   Kinetic_Energy");
                attribute_comboBox.Items.Add("      Electrical_Energy");
                attribute_comboBox.Items.Add("      Mechanical_Energy");
                attribute_comboBox.Items.Add("   Potential_Energy");
                attribute_comboBox.Items.Add("      Chemical_Energy");
                attribute_comboBox.Items.Add("Power_Attributes");

                structure_comboBox.Items.Add("No Structure Selected");
                structure_comboBox.Items.Add("African_Reed_Frog");
                structure_comboBox.Items.Add("BrittleStar-InvertebralLigaments");
                structure_comboBox.Items.Add("Butterfly_Wing");
                structure_comboBox.Items.Add("Cuttlefish_Chromatophore");
                structure_comboBox.Items.Add("Dichrotic_Filter");
                structure_comboBox.Items.Add("Dielectric_EAP");
                structure_comboBox.Items.Add("Dielectric_Mirror");
                structure_comboBox.Items.Add("Electric_Valve");
                structure_comboBox.Items.Add("ER_Fluid");
                structure_comboBox.Items.Add("FeatherStar-ArmLigaments");
                structure_comboBox.Items.Add("Firefly_LE-Organ");
                structure_comboBox.Items.Add("Flower_Bending");
                structure_comboBox.Items.Add("Flower_Opening-Closing");
                structure_comboBox.Items.Add("Gas_Lighting");
                structure_comboBox.Items.Add("Glow_Worm");
                structure_comboBox.Items.Add("Human_Muscle-
Isometric_Contraction");
                structure_comboBox.Items.Add("Ionic_EAP");
                structure_comboBox.Items.Add("Light_Emitting_Diode");
                structure_comboBox.Items.Add("Liquid_Crystal_Display");
                structure_comboBox.Items.Add("MR_Fluid");
```

```
structure_comboBox.Items.Add("NIPAM_Polymer");
structure_comboBox.Items.Add("Peacock_Feather");
structure_comboBox.Items.Add("Photonic_Ink");
structure_comboBox.Items.Add("Piezoelectric_Stack");
structure_comboBox.Items.Add("Piston");
structure_comboBox.Items.Add("Pollen_Tube_Growth");
structure_comboBox.Items.Add("Rhododendron_Leave_Curling");
structure_comboBox.Items.Add("Root_Growth");
structure_comboBox.Items.Add("Rotary_Actuators");
structure_comboBox.Items.Add("SeaCucumberDermis");
structure_comboBox.Items.Add("SeaUrchin-ToothandSpineLigaments");
structure_comboBox.Items.Add("Shape_Memory_Alloy");
structure_comboBox.Items.Add("Shape_Memory_Polymer");
structure_comboBox.Items.Add("Shear-Thickening_Fluid");
structure_comboBox.Items.Add("Starfish-Spine");
structure_comboBox.Items.Add("Stem_Growth");
structure_comboBox.Items.Add("Tie_Rod_Cylinders");
structure_comboBox.Items.Add("Vacuum_Generators");
structure_comboBox.Items.Add("Zebrafish_Chromatophore");

strategy_comboBox.Items.Add("No Strategy Selected");
strategy_comboBox.Items.Add("Actuation");
strategy_comboBox.Items.Add("   Chemically_Induced_Actuation");
strategy_comboBox.Items.Add("   Electrically_Induced_Actuation");
strategy_comboBox.Items.Add("   Hydraulic_Actuation");
strategy_comboBox.Items.Add("   Pneumatic_Actuation");
strategy_comboBox.Items.Add("Color_Modulation");
strategy_comboBox.Items.Add("   Color_Change");
strategy_comboBox.Items.Add("                                    Multi-
Layered_Thin_Film_Interface");
```

```
                strategy_comboBox.Items.Add("
Temperature_Induced_Volume_Change");
                strategy_comboBox.Items.Add("     Variable_Diffraction");
                strategy_comboBox.Items.Add("   Color_Creation");
                strategy_comboBox.Items.Add("     Structural_Color");
                strategy_comboBox.Items.Add("   Color_Filtration");
                strategy_comboBox.Items.Add("Luminescence");
                strategy_comboBox.Items.Add("   Artificial_Light_Generation");
                strategy_comboBox.Items.Add("     Electroluminescence");
                strategy_comboBox.Items.Add("     Thermoluminescence");
                strategy_comboBox.Items.Add("   Natural_Light_Generation");
                strategy_comboBox.Items.Add("     Bioluminescence");
                strategy_comboBox.Items.Add("Piezoelectric_Effect");
                strategy_comboBox.Items.Add("   Piezoelectric_Electricity_Generation");
                    strategy_comboBox.Items.Add("   Piezoelectric_Force_Generation");
                strategy_comboBox.Items.Add("Shape_Modulation");
                strategy_comboBox.Items.Add("   Nastic_Movement");
                strategy_comboBox.Items.Add("     Chemonasty");
                strategy_comboBox.Items.Add("     Hydronasty");
                strategy_comboBox.Items.Add("     Photonasty");
                strategy_comboBox.Items.Add("                                        Temp-
Induced_Shape_Memorization");
                strategy_comboBox.Items.Add("   Tropic_Movement");
                strategy_comboBox.Items.Add("     Chemotropism");
                strategy_comboBox.Items.Add("     Gravitropism");
                strategy_comboBox.Items.Add("     Hydrotropism");
                strategy_comboBox.Items.Add("     Thermotropism");
                strategy_comboBox.Items.Add("Stiffness_Change");
                strategy_comboBox.Items.Add("   Crossbridge-Effect_SlidingFilament");
                strategy_comboBox.Items.Add("   Electrorheological_Effect");
```

```
            strategy_comboBox.Items.Add("   Hydro_clustering");
            strategy_comboBox.Items.Add("   Magnetorheological_Effect");
            strategy_comboBox.Items.Add("   MutableConnectivity");
            strategy_comboBox.Items.Add("      MutConn-BrittleStarLigaments");
            strategy_comboBox.Items.Add("                                    MutConn-
FeatherStarArmLigaments");
            strategy_comboBox.Items.Add("      MutConn-SeaCucumberDermis");
            strategy_comboBox.Items.Add("                                    MutConn-
SeaUrchinToothSpineLigaments");
            strategy_comboBox.Items.Add("      MutConn-StarfishSpine");

            input_comboBox.SelectedIndex = 0;
            output_comboBox.SelectedIndex = 0;
            action_comboBox.SelectedIndex = 0;
            attribute_comboBox.SelectedIndex = 0;
            structure_comboBox.SelectedIndex = 0;
            strategy_comboBox.SelectedIndex = 0;

            uri = kb.getURI();
            results_rtBox.Text = "The uri for the reasoner is: " + uri + "\n\n";
            results_rtBox.Text += kb.loadRepository() + "\n";
            results_rtBox.Text += "Repository has been loaded. Ready for query." +
"\n";

            results_webBrowser.Navigate(Environment.CurrentDirectory          +
"/Files/default.html");
        }

        #endregion
```

```
        #region Button Clicking Methods

        private void searchForStrategy_button_Click(object sender, EventArgs e)
        {
            formatVariables();

            searchForStrategy query = new searchForStrategy(uri, reasonerAddress,
input, output, action, attribute, structure, domain);

            results_rtBox.AppendText("\nThe XML query for the strategy search is
shown here:\n\n" + query.getQueryString() + "\n");
            results_rtBox.AppendText("\nThe reasoner results are shown here:\n\n" +
query.getResultsString());

            results_webBrowser.Navigate(Environment.CurrentDirectory +
"/Files/Results.xml");
        }

        private void searchForStructure_button_Click(object sender, EventArgs e)
        {
            formatVariables();

            searchForStructure query = new searchForStructure(uri, reasonerAddress,
strategy);

            results_rtBox.AppendText("\nThe XML query for the structure search is
shown here:\n\n" + query.getQueryString() + "\n");
            results_rtBox.AppendText("\nThe reasoner results are shown here:\n\n" +
query.getResultsString());
```

```
            results_webBrowser.Navigate(Environment.CurrentDirectory +
"/Files/Results.xml");
        }

        private void resetSearchForStrategy_button_Click(object sender, EventArgs
e)
        {
            resetSearchForStrategy();
        }

        private void resetSearchForStructure_button_Click(object sender, EventArgs
e)
        {
            resetSearchForStructure();
        }

        #endregion

        #region Other Methods

        private void formatVariables()
        {
            input = input_comboBox.Text;
            input = input.Replace(" ", "");

            output = output_comboBox.Text;
            output = output.Replace(" ", "");

            action = action_comboBox.Text;
            action = action.Replace(" ", "");
```

```
    attribute = attribute_comboBox.Text;
    attribute = attribute.Replace(" ", "");

    structure = structure_comboBox.Text;
    structure = structure.Replace(" ", "");

    strategy = strategy_comboBox.Text;
    strategy = strategy.Replace(" ", "");

    if (engineering_radioButton.Checked)
    {
        domain = "Engineering_Domain";
    }
    else if (biological_radioButton.Checked)
    {
        domain = "Biological_Domain";
    }
    else
    {
        domain = "Domain";
    }
}

private void resetSearchForStrategy()
{
    input_comboBox.SelectedIndex = 0;
    output_comboBox.SelectedIndex = 0;
    action_comboBox.SelectedIndex = 0;
    attribute_comboBox.SelectedIndex = 0;
```

```
            structure_comboBox.SelectedIndex = 0;
            both_radioButton.Checked = false;
            biological_radioButton.Checked = false;
            engineering_radioButton.Checked = false;
            results_webBrowser.Navigate(Environment.CurrentDirectory +
"/Files/default.html");
        }

        private void resetSearchForStructure()
        {
            strategy_comboBox.SelectedIndex = 0;
            results_webBrowser.Navigate(Environment.CurrentDirectory +
"/Files/default.html");
        }

        #endregion

        #region Tool Strip Button Clicking Methods

        private void close_toolStripMenuItem_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void saveResults_ToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Stream myStream;
            SaveFileDialog saveFileDialog = new SaveFileDialog();
```

```
saveFileDialog.Filter = "Text Files (*.txt)|*.txt|All Files (*.*)|*.*";
saveFileDialog.FilterIndex = 1;
saveFileDialog.RestoreDirectory = true;

if (saveFileDialog.ShowDialog() == DialogResult.OK)
{
   if ((myStream = saveFileDialog.OpenFile()) != null)
   {
      XmlDocument xDoc = new XmlDocument();
      xDoc.Load(Environment.CurrentDirectory + "/Files/Results.xml");

      XmlNamespaceManager nsMgr = new XmlNamespaceManager(xDoc.NameTable);
      nsMgr.AddNamespace("ns", "http://dl.kr.org/dig/2003/02/lang");

      XmlNodeList xmlNodeList = xDoc.SelectNodes("Results/ns:responses/ns:conceptSet/ns:synonyms/ns:catom/@name", nsMgr);

      StreamWriter sWriter = new StreamWriter(myStream);

      sWriter.WriteLine("Results");
      sWriter.WriteLine();
      sWriter.WriteLine("Number of Results: " + xmlNodeList.Count);
      sWriter.WriteLine();
      sWriter.WriteLine("Number" + "\t" + "Name");
      for (int i = 0; i < xmlNodeList.Count; i++)
      {
         sWriter.WriteLine(i + 1 + ". " + "\t" + xmlNodeList[i].InnerText.ToString());
```

```
                }
                sWriter.WriteLine();
                sWriter.WriteLine("File Saved - " + DateTime.Now);

                sWriter.Close();
                myStream.Close();
            }
        }
    }

    private void launchResults_ToolStripMenuItem_Click(object sender,
EventArgs e)
    {
        System.Diagnostics.Process.Start(Environment.CurrentDirectory +
"/Files/Results.xml");
    }

    private void clearLog_ToolStripMenuItem_Click(object sender, EventArgs
e)
    {
        results_rtBox.Clear();
    }

    #endregion

    #region Find Parents Buttons

    private void inputParent_button_Click(object sender, EventArgs e)
    {
        formatVariables();
```

```
            findParents query = new findParents(uri, reasonerAddress, input);

            results_rtBox.AppendText("\nThe query formed to find the input parent is shown here:\n\n" + query.getQueryString() + "\n");
            results_rtBox.AppendText("\nThe results from the parent search are shown here:\n\n" + query.getResultsString());

            string comboName = query.getParentName();
            if (comboName != "None")
            {
               int comboLoc = input_comboBox.FindString(comboName);
               while (comboLoc == -1)
               {
                  comboName = "   " + comboName;
                  comboLoc = input_comboBox.FindString(comboName);
               }
               input_comboBox.SelectedIndex = comboLoc;
            }
            else
            {
               input_comboBox.SelectedIndex = 0;
            }
         }

         private void outputParent_button_Click(object sender, EventArgs e)
         {
            formatVariables();

            findParents query = new findParents(uri, reasonerAddress, output);
```

```
            results_rtBox.AppendText("\nThe query formed to find the output parent is shown here:\n\n" + query.getQueryString() + "\n");
            results_rtBox.AppendText("\nThe results from the parent search are shown here:\n\n" + query.getResultsString());

            string comboName = query.getParentName();
            if (comboName != "None")
            {
               int comboLoc = output_comboBox.FindString(comboName);
               while (comboLoc == -1)
               {
                  comboName = "   " + comboName;
                  comboLoc = output_comboBox.FindString(comboName);
               }
               output_comboBox.SelectedIndex = comboLoc;
            }
            else
            {
               output_comboBox.SelectedIndex = 0;
            }
         }

         private void actionParent_button_Click(object sender, EventArgs e)
         {
            formatVariables();

            findParents query = new findParents(uri, reasonerAddress, action);
```

```
            results_rtBox.AppendText("\nThe query formed to find the action parent is shown here:\n\n" + query.getQueryString() + "\n");
            results_rtBox.AppendText("\nThe results from the parent search are shown here:\n\n" + query.getResultsString());

            string comboName = query.getParentName();
            if (comboName != "None")
            {
               int comboLoc = action_comboBox.FindString(comboName);
               while (comboLoc == -1)
               {
                  comboName = "   " + comboName;
                  comboLoc = action_comboBox.FindString(comboName);
               }
               action_comboBox.SelectedIndex = comboLoc;
            }
            else
            {
               action_comboBox.SelectedIndex = 0;
            }
         }

         private void attributeParent_button_Click(object sender, EventArgs e)
         {
            formatVariables();

            findParents query = new findParents(uri, reasonerAddress, attribute);

            results_rtBox.AppendText("\nThe query formed to find the attribute parent is shown here:\n\n" + query.getQueryString() + "\n");
```

```
            results_rtBox.AppendText("\nThe results from the parent search are shown here:\n\n" + query.getResultsString());

            string comboName = query.getParentName();
            if (comboName != "None")
            {
               int comboLoc = attribute_comboBox.FindString(comboName);
               while (comboLoc == -1)
               {
                  comboName = "   " + comboName;
                  comboLoc = attribute_comboBox.FindString(comboName);
               }
               attribute_comboBox.SelectedIndex = comboLoc;
            }
            else
            {
               attribute_comboBox.SelectedIndex = 0;
            }
         }

         private void structureParent_button_Click(object sender, EventArgs e)
         {
            formatVariables();

            findParents query = new findParents(uri, reasonerAddress, structure);

            results_rtBox.AppendText("\nThe query formed to find the structure parent is shown here:\n\n" + query.getQueryString() + "\n");
            results_rtBox.AppendText("\nThe results from the parent search are shown here:\n\n" + query.getResultsString());
```

```
            string comboName = query.getParentName();
            if (comboName != "None")
            {
                int comboLoc = structure_comboBox.FindString(comboName);
                while (comboLoc == -1)
                {
                    comboName = "   " + comboName;
                    comboLoc = structure_comboBox.FindString(comboName);
                }
                structure_comboBox.SelectedIndex = comboLoc;
            }
            else
            {
                structure_comboBox.SelectedIndex = 0;
            }
        }

        #endregion
    }
}
```

InitializeKnowledgeBase Class

```
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;
using System.Reflection;
using System.Xml;
using System.Xml.XPath;

namespace RepositoryGUI
{
   class InitializeKnowledgeBase
   {
      #region Variables

      private Network nw;
      private string uri;
      private string reasonerAddress;
      private string repositoryFile;

      #endregion

      #region Constructor

      public InitializeKnowledgeBase(string ra)
      {
         nw = new Network();
         uri = "";
         reasonerAddress = ra;
```

```
            repositoryFile = "";
        }

        #endregion

        #region Accessors

        public string getURI()
        {
            return uri;
        }

        public string getRepository()
        {
            return repositoryFile;
        }

        #endregion

        #region Other Methods

        public void newKB()
        {
            string newKBStr = "<newKB
xmlns=\"http://dl.kr.org/dig/2003/02/lang\"/>";
            string xStr = nw.InvokeReasoner(reasonerAddress, newKBStr);

            XmlDocument xDoc = new XmlDocument();
            xDoc.LoadXml(xStr);
```

```
            XmlNamespaceManager nsMgr = new
XmlNamespaceManager(xDoc.NameTable);
            nsMgr.AddNamespace("ns", "http://dl.kr.org/dig/2003/02/lang");

            XmlNode xNode;
            xNode = xDoc.SelectSingleNode("/ns:response/ns:kb/@uri", nsMgr);

            uri = xNode.Value.ToString();
        }

        public string loadRepository()
        {
            XmlDocument xDoc = new XmlDocument();
            xDoc.Load(Environment.CurrentDirectory + "/Files/Repository.xml");

            XmlNamespaceManager nsMgr = new
XmlNamespaceManager(xDoc.NameTable);
            nsMgr.AddNamespace("ns", "http://dl.kr.org/dig/2003/02/lang");
            nsMgr.AddNamespace("xsi", "http://www.w3.org/2001/XMLSchema-
instance");

            XmlElement tellElem = xDoc.CreateElement("", "tells",
"http://dl.kr.org/dig/2003/02/lang");

            XmlAttribute tellURIAttr = xDoc.CreateAttribute("uri");
            tellURIAttr.Value = uri;
            tellElem.Attributes.Append(tellURIAttr);

            XmlAttribute tellXSIAttr = xDoc.CreateAttribute("xsi",
"schemaLocation", "http://www.w3.org/2001/XMLSchema-instance");
```

```
            tellXSIAttr.Value = "http://dl.kr.org/dig/2003/02/lang";
            tellElem.Attributes.Append(tellXSIAttr);

            XmlNodeList xNodeList = xDoc.SelectNodes("/ns:Start/*", nsMgr);
            for (int i = 0; i < xNodeList.Count; i++)
            {
               tellElem.AppendChild(xNodeList[i]);
            }

            repositoryFile = tellElem.OuterXml.ToString();

            return nw.InvokeReasoner(reasonerAddress, repositoryFile);
        }

        #endregion
    }
}
```

Network Class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net;
using System.IO;

namespace RepositoryGUI
{
   class Network
   {
      #region Variables

      private HttpWebRequest request;
      private HttpWebResponse response;

      #endregion

      #region Invoke Reasoner Method

      public string InvokeReasoner(string url, string msg)
      {
         try
         {
            request = (HttpWebRequest)WebRequest.Create(url);
            request.Method = "post";
```

```
            ASCIIEncoding encoding = new ASCIIEncoding();
            byte[] byte1 = encoding.GetBytes(msg);
            // Set the content type of the data being posted.
            request.ContentType = "text/xml";

            // Set the content length of the string being posted.
            request.ContentLength = byte1.Length;
            request.Pipelined = true;
            request.KeepAlive = true;

            Stream newStream = request.GetRequestStream();
            newStream.Write(byte1, 0, byte1.Length);

            response = (HttpWebResponse)request.GetResponse();

            //response.ContentType = "text/xml";

            Stream receiveStream = response.GetResponseStream();
            StreamReader readStream = new StreamReader(receiveStream,
Encoding.UTF8);

            string retStr = "";
            retStr = readStream.ReadToEnd();

            readStream.Close();
            receiveStream.Close();
            newStream.Close();
            response.Close();
            return retStr;
        }
```

```
            catch (Exception ex)
            {
                return ex.Message;
            }
        }

        #endregion
    }
}
```

searchForStrategy Class

```
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;
using System.Reflection;
using System.Xml;
using System.Xml.XPath;

namespace RepositoryGUI
{
   class searchForStrategy
   {
      #region Variables

      private string uri;
      private string reasonerAddress;
      private string input;
      private string output;
      private string action;
      private string attribute;
      private string structure;
      private string domain;
      private XmlDocument xDoc;
      Network nw;
      private string queryXML;
      private string reasonerResults;

      #endregion
```

```
        #region Constructor

        public searchForStrategy(string u, string ra, string i, string o, string ac, string
at, string st, string d)
        {
            uri = u;
            reasonerAddress = ra;
            input = i;
            output = o;
            action = ac;
            attribute = at;
            structure = st;
            domain = d;
            xDoc = new XmlDocument();
            nw = new Network();
            queryXML = "";
            reasonerResults = "";

            formatVariables();
            createQuery();
            findStrategy();
            formatResults();
        }

        #endregion

        #region Search Methods

        private void createQuery()
```

```
{
    XmlElement askElem = xDoc.CreateElement("", "asks", "");

    XmlAttribute askNSAttr = xDoc.CreateAttribute("xmlns");
    askNSAttr.Value = "http://dl.kr.org/dig/2003/02/lang";
    askElem.Attributes.Append(askNSAttr);

    XmlAttribute askURIAttr = xDoc.CreateAttribute("uri");
    askURIAttr.Value = uri;
    askElem.Attributes.Append(askURIAttr);

    XmlAttribute askXSIAttr = xDoc.CreateAttribute("xsi",
"schemaLocation", "http://www.w3.org/2001/XMLSchema-instance");
    askXSIAttr.Value = "http://dl.kr.org/dig/2003/02/lang";
    askElem.Attributes.Append(askXSIAttr);

    XmlElement descElem = xDoc.CreateElement("", "descendants", "");

    XmlAttribute descAttr = xDoc.CreateAttribute("id");
    descAttr.Value = "q1";
    descElem.Attributes.Append(descAttr);

    XmlElement andElem = xDoc.CreateElement("", "and", "");

    askElem.AppendChild(descElem);
    descElem.AppendChild(andElem);

    andElem.AppendChild(createFlowNode(input, output));
    andElem.AppendChild(createBehaviorNode(action, attribute));
    andElem.AppendChild(createStructureNode(structure));
```

```
    andElem.AppendChild(createDomainNode(domain));

    queryXML = askElem.OuterXml.ToString();
}

private void findStrategy()
{
    reasonerResults = nw.InvokeReasoner(reasonerAddress, queryXML);
}

private void formatVariables()
{
    if (input == "NoInputSelected")
    {
        input = "Flows";
    }
    if (output == "NoOutputSelected")
    {
        output = "Flows";
    }
    if (action == "NoActionSelected")
    {
        action = "Actions";
    }
    if (attribute == "NoAttributeSelected")
    {
        attribute = "Attributes";
    }
    if (structure == "NoStructureSelected")
    {
```

```
        structure = "Structure";
    }
}

#endregion

#region Display Methods

public void formatResults()
{
    string fileLocation = "Files/Results.xml";
    XmlTextWriter textWriter = new XmlTextWriter(fileLocation, null);

    string pi1 = "version='1.0' encoding='UTF-8'";
    textWriter.WriteProcessingInstruction("xml", pi1);

    string pi2 = "type='text/xsl' href='formatStrategyResults.xsl'";
    textWriter.WriteProcessingInstruction("xml-stylesheet", pi2);

    XmlDocument doc = new XmlDocument();
    doc.LoadXml(reasonerResults);

    XmlElement xmlElem = doc.CreateElement("", "Results", "");

    XmlNodeList xmlNodeList = doc.SelectNodes("/*");
    for (int i = 0; i < xmlNodeList.Count; i++)
    {
        xmlElem.AppendChild(xmlNodeList[i]);
    }
```

```
    doc.LoadXml(xmlElem.OuterXml.ToString());
    doc.Save(textWriter);

    textWriter.Close();
}

#endregion

#region Node-Creating Methods

private XmlElement createFlowNode(string i, string o)
{
    XmlElement fSomeElem = xDoc.CreateElement("", "some", "");

    XmlElement fRatomElem = xDoc.CreateElement("", "ratom", "");

    XmlAttribute fRatomAttr = xDoc.CreateAttribute("name");
    fRatomAttr.Value = "satisfiesFunction";
    fRatomElem.Attributes.Append(fRatomAttr);

    XmlElement andElem = xDoc.CreateElement("", "and", "");

    XmlElement iSomeElem = xDoc.CreateElement("", "some", "");

    XmlElement iRatomElem = xDoc.CreateElement("", "ratom", "");

    XmlAttribute iRatomAttr = xDoc.CreateAttribute("name");
    iRatomAttr.Value = "hasInput";
    iRatomElem.Attributes.Append(iRatomAttr);
```

```
XmlElement iCatomElem = xDoc.CreateElement("", "catom", "");

XmlAttribute iCatomAttr = xDoc.CreateAttribute("name");
iCatomAttr.Value = i;
iCatomElem.Attributes.Append(iCatomAttr);

XmlElement oSomeElem = xDoc.CreateElement("", "some", "");

XmlElement oRatomElem = xDoc.CreateElement("", "ratom", "");

XmlAttribute oRatomAttr = xDoc.CreateAttribute("name");
oRatomAttr.Value = "hasOutput";
oRatomElem.Attributes.Append(oRatomAttr);

XmlElement oCatomElem = xDoc.CreateElement("", "catom", "");

XmlAttribute oCatomAttr = xDoc.CreateAttribute("name");
oCatomAttr.Value = o;
oCatomElem.Attributes.Append(oCatomAttr);

fSomeElem.AppendChild(fRatomElem);
fSomeElem.AppendChild(andElem);
andElem.AppendChild(iSomeElem);
iSomeElem.AppendChild(iRatomElem);
iSomeElem.AppendChild(iCatomElem);
andElem.AppendChild(oSomeElem);
oSomeElem.AppendChild(oRatomElem);
oSomeElem.AppendChild(oCatomElem);

return fSomeElem;
```

```
}

private XmlElement createBehaviorNode(string ac, string at)
{
   XmlElement bSomeElem = xDoc.CreateElement("", "some", "");

   XmlElement bRatomElem = xDoc.CreateElement("", "ratom", "");

   XmlAttribute bRatomAttr = xDoc.CreateAttribute("name");
   bRatomAttr.Value = "refinesBehavior";
   bRatomElem.Attributes.Append(bRatomAttr);

   XmlElement andElem = xDoc.CreateElement("", "and", "");

   XmlElement acSomeElem = xDoc.CreateElement("", "some", "");

   XmlElement acRatomElem = xDoc.CreateElement("", "ratom", "");

   XmlAttribute acRatomAttr = xDoc.CreateAttribute("name");
   acRatomAttr.Value = "hasAction";
   acRatomElem.Attributes.Append(acRatomAttr);

   XmlElement acCatomElem = xDoc.CreateElement("", "catom", "");

   XmlAttribute acCatomAttr = xDoc.CreateAttribute("name");
   acCatomAttr.Value = ac;
   acCatomElem.Attributes.Append(acCatomAttr);

   XmlElement atSomeElem = xDoc.CreateElement("", "some", "");
```

```
    XmlElement atRatomElem = xDoc.CreateElement("", "ratom", "");

    XmlAttribute atRatomAttr = xDoc.CreateAttribute("name");
    atRatomAttr.Value = "hasAttribute";
    atRatomElem.Attributes.Append(atRatomAttr);

    XmlElement atCatomElem = xDoc.CreateElement("", "catom", "");

    XmlAttribute atCatomAttr = xDoc.CreateAttribute("name");
    atCatomAttr.Value = at;
    atCatomElem.Attributes.Append(atCatomAttr);

    bSomeElem.AppendChild(bRatomElem);
    bSomeElem.AppendChild(andElem);
    andElem.AppendChild(acSomeElem);
    acSomeElem.AppendChild(acRatomElem);
    acSomeElem.AppendChild(acCatomElem);
    andElem.AppendChild(atSomeElem);
    atSomeElem.AppendChild(atRatomElem);
    atSomeElem.AppendChild(atCatomElem);

    return bSomeElem;
}

private XmlElement createStructureNode(string st)
{
    XmlElement someElem = xDoc.CreateElement("", "some", "");

    XmlElement ratomElem = xDoc.CreateElement("", "ratom", "");
```

```
    XmlAttribute ratomAttr = xDoc.CreateAttribute("name");
    ratomAttr.Value = "hasStructure";
    ratomElem.Attributes.Append(ratomAttr);

    XmlElement catomElem = xDoc.CreateElement("", "catom", "");

    XmlAttribute catomAttr = xDoc.CreateAttribute("name");
    catomAttr.Value = st;
    catomElem.Attributes.Append(catomAttr);

    someElem.AppendChild(ratomElem);
    someElem.AppendChild(catomElem);

    return someElem;
}

private XmlElement createDomainNode(string d)
{
    XmlElement someElem = xDoc.CreateElement("", "some", "");

    XmlElement ratomElem = xDoc.CreateElement("", "ratom", "");

    XmlAttribute ratomAttr = xDoc.CreateAttribute("name");
    ratomAttr.Value = "fromDomain";
    ratomElem.Attributes.Append(ratomAttr);

    XmlElement catomElem = xDoc.CreateElement("", "catom", "");

    XmlAttribute catomAttr = xDoc.CreateAttribute("name");
    catomAttr.Value = d;
```

```
            catomElem.Attributes.Append(catomAttr);

            someElem.AppendChild(ratomElem);
            someElem.AppendChild(catomElem);

            return someElem;
        }

        #endregion

        #region Accessors

        public string getQueryString()
        {
            return queryXML;
        }

        public string getResultsString()
        {
            return reasonerResults;
        }

        #endregion
    }
}
```

searchForStructure

```
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;
using System.Reflection;
using System.Xml;
using System.Xml.XPath;

namespace RepositoryGUI
{
   class searchForStructure
   {
      #region Variables

      private string uri;
      private string reasonerAddress;
      private string strategy;
      private XmlDocument xDoc;
      Network nw;
      private string queryXML;
      private string reasonerResults;

      #endregion

      #region Constructor

      public searchForStructure(string u, string ra, string s)
      {
```

```
    uri = u;
    reasonerAddress = ra;
    strategy = s;

    xDoc = new XmlDocument();
    nw = new Network();
    queryXML = "";
    reasonerResults = "";

    formatVariables();
    createQuery();
    findStructure();
    formatResults();
}

#endregion

#region Search Methods

private void createQuery()
{
    XmlElement askElem = xDoc.CreateElement("", "asks", "");

    XmlAttribute askNSAttr = xDoc.CreateAttribute("xmlns");
    askNSAttr.Value = "http://dl.kr.org/dig/2003/02/lang";
    askElem.Attributes.Append(askNSAttr);

    XmlAttribute askURIAttr = xDoc.CreateAttribute("uri");
    askURIAttr.Value = uri;
    askElem.Attributes.Append(askURIAttr);
```

```
            XmlAttribute askXSIAttr = xDoc.CreateAttribute("xsi",
"schemaLocation", "http://www.w3.org/2001/XMLSchema-instance");
            askXSIAttr.Value = "http://dl.kr.org/dig/2003/02/lang";
            askElem.Attributes.Append(askXSIAttr);

            XmlElement descElem = xDoc.CreateElement("", "descendants", "");

            XmlAttribute descAttr = xDoc.CreateAttribute("id");
            descAttr.Value = "q1";
            descElem.Attributes.Append(descAttr);

            XmlElement andElem = xDoc.CreateElement("", "and", "");

            askElem.AppendChild(descElem);
            descElem.AppendChild(andElem);

            andElem.AppendChild(createStrategyNode(strategy));

            queryXML = askElem.OuterXml.ToString();
        }

        private void findStructure()
        {
            reasonerResults = nw.InvokeReasoner(reasonerAddress, queryXML);
        }

        private void formatVariables()
        {
            if (strategy == "NoStrategySelected")
```

```
    {
        strategy = "SystemStrategy";
    }
}

#endregion

#region Display Methods

public void formatResults()
{
    string fileLocation = "Files/Results.xml";
    XmlTextWriter textWriter = new XmlTextWriter(fileLocation, null);

    string pi1 = "version='1.0' encoding='UTF-8'";
    textWriter.WriteProcessingInstruction("xml", pi1);

    string pi2 = "type='text/xsl' href='formatStructureResults.xsl'";
    textWriter.WriteProcessingInstruction("xml-stylesheet", pi2);

    XmlDocument doc = new XmlDocument();
    doc.LoadXml(reasonerResults);

    XmlElement xmlElem = doc.CreateElement("", "Results", "");

    XmlNodeList xmlNodeList = doc.SelectNodes("/*");
    for (int i = 0; i < xmlNodeList.Count; i++)
    {
        xmlElem.AppendChild(xmlNodeList[i]);
    }
```

```
    doc.LoadXml(xmlElem.OuterXml.ToString());
    doc.Save(textWriter);

    textWriter.Close();
}

#endregion

#region Node-Creating Methods

private XmlElement createStrategyNode(string s)
{
    XmlElement someElem = xDoc.CreateElement("", "some", "");

    XmlElement ratomElem = xDoc.CreateElement("", "ratom", "");

    XmlAttribute ratomAttr = xDoc.CreateAttribute("name");
    ratomAttr.Value = "hasStrategy";
    ratomElem.Attributes.Append(ratomAttr);

    XmlElement catomElem = xDoc.CreateElement("", "catom", "");

    XmlAttribute catomAttr = xDoc.CreateAttribute("name");
    catomAttr.Value = s;
    catomElem.Attributes.Append(catomAttr);

    someElem.AppendChild(ratomElem);
    someElem.AppendChild(catomElem);
```

```
            return someElem;
        }

        #endregion

        #region Accessors

        public string getQueryString()
        {
            return queryXML;
        }

        public string getResultsString()
        {
            return reasonerResults;
        }

        #endregion
    }
}
```

Program Class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

namespace RepositoryGUI
{
   static class Program
   {
      /// <summary>
      /// The main entry point for the application.
      /// </summary>
      [STAThread]
      static void Main()
      {
         Application.EnableVisualStyles();
         Application.SetCompatibleTextRenderingDefault(false);
         Application.Run(new GUI());
      }
   }
}
```

APPENDIX B – BIOLOGICAL AND ENGINEERING STRATEGIES

In Appendix B, the description logic descriptions of the biological and engineering strategies used to populate the Strategy Repository in Section 5.5 are displayed.

Strategy	DL Description
Crossbridge-Effect_SlidingFilament	∃ satisfiesFunction.[∃hasInput.Affinity ⊓ ∃hasOutput.Force] ⊓ ∃ refinesBehavior.[∃hasAction.Increment ⊓ ∃hasAttribute.Stiffness] ⊓ ∃hasStructure.Human_Musle-IsomContraction ⊓ ∃fromDomain.Biological_Domain
Electrorheological_Effect	∃ satisfiesFunction.[∃hasInput.Current ⊓ ∃hasOutput.Force] ⊓ ∃ refinesBehavior.[∃hasAction.Increment ⊓ ∃hasAttribute.Stiffness] ⊓ ∃hasStructure.ER_Fluid ⊓ ∃fromDomain.Engineering_Domain
Hydro_clustering	∃ satisfiesFunction.[∃hasInput.Force ⊓ ∃hasOutput.Force] ⊓ ∃ refinesBehavior.[∃hasAction.Increment ⊓ ∃hasAttribute.Stiffness] ⊓ ∃hasStructure.Shear-Thickening_Fluid ⊓ ∃fromDomain.Engineering_Domain
Magnetorheological_Effect	∃ satisfiesFunction.[∃hasInput.MagneticFluxRate ⊓ ∃hasOutput.Force] ⊓ ∃refinesBehavior.[∃hasAction.Increment ⊓ ∃hasAttribute.Stiffness] ⊓ ∃hasStructure.MR_Fluid ⊓ ∃fromDomain.Engineering_Domain
MutConn-BrittleStarLigaments	∃ satisfiesFunction.[∃hasInput.Affinity ⊓ ∃hasOutput.Force] ⊓

	∃ refinesBehavior.[∃hasAction.Increment ⊓ ∃hasAttribute.Stiffness] ⊓ ∃hasStructure. BrittleStar-InvertebralLigaments ⊓ ∃fromDomain.Biological_Domain
MutConn-FeatherStarArmLigaments	∃satisfiesFunction.[∃hasInput.Affinity ⊓ ∃hasOutput.Force] ⊓ ∃refinesBehavior.[∃hasAction.Increment ⊓ ∃hasAttribute.Stiffness] ⊓ ∃hasStructure. FeatherStarArmLigaments ⊓ ∃fromDomain.Biological_Domain
MutConn-SeaCucumberDermis	∃ satisfiesFunction.[∃hasInput.Affinity ⊓ ∃hasOutput.Force] ⊓ ∃ refinesBehavior.[∃hasAction.Increment ⊓ ∃hasAttribute.Stiffness] ⊓ ∃hasStructure.SeaCucumberDermis ⊓ ∃fromDomain.Biological_Domain
MutConn-SeaUrchinTooth	∃ satisfiesFunction.[∃hasInput.Affinity ⊓ ∃hasOutput.Force] ⊓ ∃ refinesBehavior.[∃hasAction.Increment ⊓ ∃hasAttribute.Stiffness] ⊓ ∃hasStructure. SeaUrchinToothSpineLigaments ⊓ ∃fromDomain.Biological_Domain
MutConn-StarfishSpine	∃ satisfiesFunction.[∃hasInput.Affinity ⊓ ∃hasOutput.Force] ⊓ ∃ refinesBehavior.[∃hasAction.Increment ⊓ ∃hasAttribute.Stiffness] ⊓ ∃hasStructure. StarfishSpine ⊓ ∃fromDomain.Biological_Domain
Chemically_Induced_Actuation	∃ satisfiesFunction ((∃ hasInput Chemical) ⊓ (∃

	hasOutput Force)) ⊓ ∃ refinesBehavior ((∃ hasAction Convert) ⊓ (∃ hasAttribute Force-Extrinsic-Mechanical)) ⊓ ∃ hasStructure Human_Muscle-Isometric_Contraction ⊓ ∃ fromDomain Engineering_Domain
Electrically_Induced_Actuation	∃ satisfiesFunction ((∃ hasInput Electrical) ⊓ (∃ hasOutput Force)) ⊓ ∃ refinesBehavior ((∃ hasAction Convert) ⊓ (∃ hasAttribute Force-Extrinsic-Mechanical)) ⊓ ∃ hasStructure Ionic_EAP ⊓ ∃ hasStructure Dielectric_EAP ⊓ ∃ hasStructure Electric_Valve ⊓ ∃ fromDomain Engineering_Domain
Hydraulic_Actuation	∃ satisfiesFunction ((∃ hasInput Liquid_Material) ⊓ (∃ hasOutput Force)) ⊓ ∃ refinesBehavior ((∃ hasAction Convert) ⊓ (∃ hasAttribute Force-Extrinsic-Mechanical)) ⊓ ∃ hasStructure Piston ⊓ ∃ fromDomain Engineering_Domain
Pneumatic_Actuation	∃ satisfiesFunction ((∃ hasInput Gas_Material) ⊓ (∃ hasOutput Force)) ⊓ ∃ refinesBehavior ((∃ hasAction Convert) ⊓ (∃ hasAttribute Force-Extrinsic-Mechanical)) ⊓ ∃ hasStructure Rotary_Actuators ⊓ ∃ hasStructure Tie_Rod_Cylinders ⊓ ∃ hasStructure Vacuum_Generators ⊓ ∃ fromDomain Engineering_Domain
Multi-Layered_Thin_Film_Interface	∃ satisfiesFunction ((∃ hasInput Mechanical) ⊓ (∃ hasOutput Visual)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Color)) ⊓ ∃ hasStructure Zebrafish_Chromatophore ⊓ ∃ hasStructure Cuttlefish_Chromatophore ⊓ ∃ fromDomain Biological_Domain

Temperature_Induced_Volume _Change	∃ satisfiesFunction ((∃ hasInput Temperature) ⊓ (∃ hasOutput Visual)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Color))⊓ ∃ hasStructure NIPAM_Polymer⊓ ∃ fromDomain Biological_Domain
Variable_Diffraction	∃ satisfiesFunction ((∃ hasInput Electrical) ⊓ (∃ hasOutput Visual)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Color))⊓ ∃ hasStructure Photonic_Ink ⊓∃ fromDomain Biological_Domain
Structural_Color	∃ satisfiesFunction ((∃ hasInput Intensity_Optical) ⊓ (∃ hasOutput Visual)) ⊓ ∃ refinesBehavior ((∃ hasAction Convert) ⊓ (∃ hasAttribute Color)) ⊓ ∃ hasStructure Peacock_Feather ⊓ ∃ hasStructure African_Reed_Frog ⊓∃ hasStructure Butterfly_Wing ⊓ ∃ fromDomain Biological_Domain
Color_Filtration	∃ satisfiesFunction ((∃ hasInput Intensity_Optical) ⊓ (∃ hasOutput Intensity_Optical)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Color)) ⊓ ∃ hasStructure Dielectric_Mirror ⊓ ∃ hasStructure Dichrotic_Filter ⊓ ∃ fromDomain Engineering_Domain
Electroluminescence	∃ satisfiesFunction ((∃ hasInput Electrical) ⊓ (∃ hasOutput Intensity_Optical)) ⊓ ∃ refinesBehavior ((∃ hasAction Convert) ⊓ (∃ hasAttribute Luminosity)) ⊓ ∃ hasStructure Light_Emitting_Diode ⊓ ∃ hasStructure Liquid_Crystal_Display ⊓ ∃ fromDomain Engineering_Domain
Thermoluminescence	∃ satisfiesFunction ((∃ hasInput Temperature) ⊓ (∃ hasOutput Intensity_Optical)) ⊓ ∃ refinesBehavior ((∃

	hasAction Convert) ⊓ (∃ hasAttribute Luminosity)) ⊓ ∃ hasStructure Gas_Lighting ⊓ ∃ fromDomain Engineering_Domain
Bioluminescence	∃ satisfiesFunction ((∃ hasInput Chemical) ⊓ (∃ hasOutput Intensity_Optical)) ⊓ ∃ refinesBehavior ((∃ hasAction Convert) ⊓ (∃ hasAttribute Luminosity)) ⊓ ∃ hasStructure Firefly_LE-Organ ⊓ ∃ hasStructure Glow_Worm ⊓ ∃ fromDomain Biological_Domain
Piezoelectric_Electricity_Generation	∃ satisfiesFunction ((∃ hasInput Force) ⊓ (∃ hasOutput Electrical)) ⊓ ∃ refinesBehavior ((∃ hasAction Convert) ⊓ (∃ hasAttribute Electrical_Energy)) ⊓ ∃ hasStructure Piezoelectric_Stack ⊓ ∃ fromDomain Engineering_Domain
Piezoelectric_Force_Generation	∃ satisfiesFunction ((∃ hasInput Electrical) ⊓ (∃ hasOutput Force)) ⊓ ∃ refinesBehavior ((∃ hasAction Convert) ⊓ (∃ hasAttribute Force-Extrinsic-Mechanical)) ⊓ ∃ hasStructure Piezoelectric_Stack ⊓ ∃ fromDomain Engineering_Domain
Chemonasty	∃ satisfiesFunction ((∃ hasInput Chemical) ⊓ (∃ hasOutput Mechanical)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Shape-Physical)) ⊓ ∃ fromDomain Biological_Domain
Hydronasty	∃ satisfiesFunction ((∃ hasInput Liquid_Material) ⊓ (∃ hasOutput Mechanical)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Shape-Physical)) ⊓ ∃ fromDomain Biological_Domain
Photonasty	∃ satisfiesFunction ((∃ hasInput Intensity_Optical) ⊓ (∃ hasOutput Mechanical)) ⊓ ∃ refinesBehavior ((∃

	hasAction Change) ⊓ (∃ hasAttribute Shape-Physical)) ⊓ ∃ hasStructure Flower_Opening-Closing ⊓ ∃ fromDomain Biological_Domain
Temp-Induced_Shape_Memorization	∃ satisfiesFunction ((∃ hasInput Temperature) ⊓ (∃ hasOutput Mechanical)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Shape-Physical)) ⊓ ∃ hasStructure Shape_Memory_Polymer ⊓ ∃ hasStructure Shape_Memory_Alloy ⊓ ∃ fromDomain Enginering_Domain
Chemotropism	∃ satisfiesFunction ((∃ hasInput Chemical) ⊓ (∃ hasOutput Mechanical)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Shape-Physical)) ⊓ ∃ hasStructure Pollen_Tube_Growth ⊓ ∃ fromDomain Biological_Domain
Gravitropism	∃ satisfiesFunction ((∃ hasInput Force) ⊓ (∃ hasOutput Mechanical)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Shape-Physical)) ⊓ ∃ hasStructure Stem_Growth ⊓ ∃ hasStructure Root_Growth ⊓ ∃ fromDomain Biological_Domain
Hydrotropism	∃ satisfiesFunction ((∃ hasInput Liquid_Material) ⊓ (∃ hasOutput Mechanical)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Shape-Physical)) ⊓ ∃ hasStructure Flower_Bending ⊓ ∃ fromDomain Biological_Domain
Thermotropism	∃ satisfiesFunction ((∃ hasInput Temperature) ⊓ (∃ hasOutput Mechanical)) ⊓ ∃ refinesBehavior ((∃ hasAction Change) ⊓ (∃ hasAttribute Shape-Physical)) ⊓

	∃ hasStructure Rhododendron_Leave_Curling ⊓ ∃ fromDomain Biological_Domain

APPENDIX C – COGNITIVE STUDY DOCUMENTS

In Appendix C, the documents from the cognitive studies in Section 7.1 are presented.

Biological analogy

Echinoderms (ie. Sea cucumbers) possess the ability to control the tensile properties (stiffness) of their skin by regulating the stress transfer between collagen fibril bundles. Interactions between these fibril bundles are regulated by special cells controlled by the sea cucumber's neural system. In its low stiffness state, the individual collagen fibril bundles are allowed to slide past one another. When signaled by the neural system, the special cells release a binding agent, called stiparin, which causes the individual fiber bundles to become linked. This causes the high-stiffness state of the skin.

The skin tissue can be modeled as a flexible composite of discontinuous fibrils within a viscous liquid medium. The force transferred through the solution to the fibrils depends on the size and orientation of the fibrils. Once activated, these fibrils become linked into a network of larger, continuous fibers (Figure 1). This increased size leads to an increased contribution on their part to the stiffness of the skin.

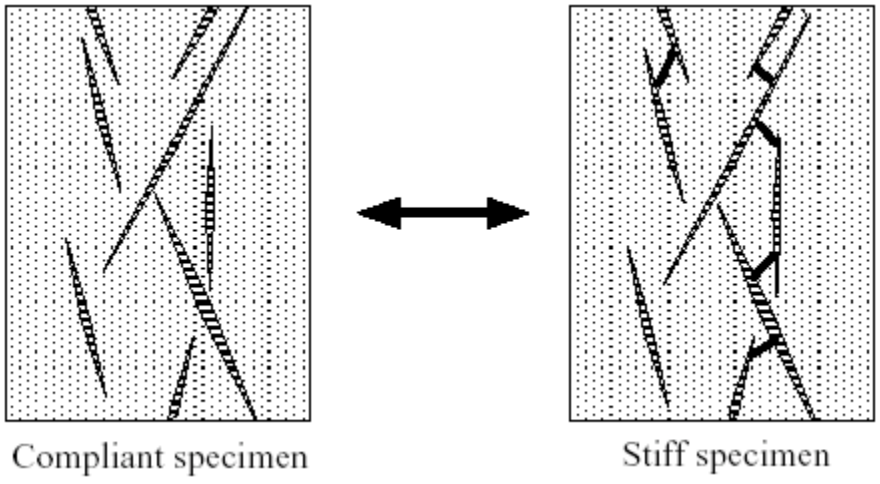


Figure 1. Model of the Echinoderm skin

Figure C. 1 Biological Design Example for Study 1 [129]

Electrorheological (ER) fluids

Electrorheological (ER) fluids are fluids that experience increased yield stress in the presence of electric fields. ER fluids consist of extremely small non-conducting particles suspended in an electrically insulating carrier fluid medium. In the absence of an electric field, ER fluids behave as typical fluids (Figure 1a). When an electric field is applied, these particles bind and the fluid immediately 'solidifies' with a yield point determined by the electric field strength (Fluid 1b).

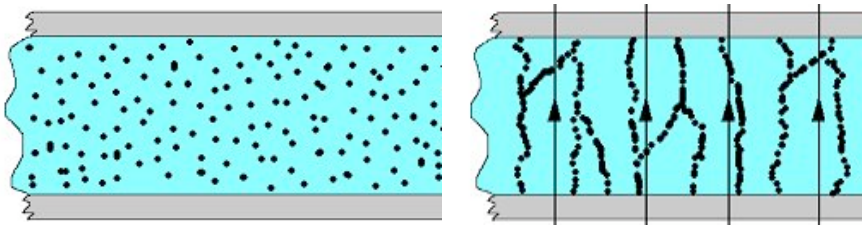


Figure 1. (a) ER fluid with zero electric field applied (b) ER fluid solidifies when electric field is applied

One application of this technology is in the US Army's planned Future Force Warrior project. In this project, the Army plans to create bullet-resistant armor using the ER fluid, whereby the stiffness of the armor can be actively-controlled.

Figure C. 2 Human-Engineered Design Example for Study 1 [194]

Nastic shape change

Many plants exhibit shape change and structural movement by way of nastic movement. In response to external stimuli, nastic movements are rapid, reversible responses caused by a change in the internal pressure due to movement of water within the cells. In different species, external stimuli for nastic movement in plants include light, chemical, water, temperature, and touch.

Figure 1. Venus Fly trap, which closes to trap insects when touched

Due to internal pressure control, nastic plants can change from one shape to another based on an external stimulus.

Figure C. 3 Variable-Stiffness Biological Design Example for Study 2 [195]

Human muscle in Isometric contraction

The stiffness of human muscle can be controlled under isometric (fixed-length) contraction. The basic unit of the muscle, the sarcomere, is composed of actin and myosin filaments. Stiffness change is caused by the chemical activation of crossbridges on the myosin myofilament, which bond to the actin myofilaments. The stiffness is directly proportional to the number of cross-bridges activated between the actin and myosin filaments.

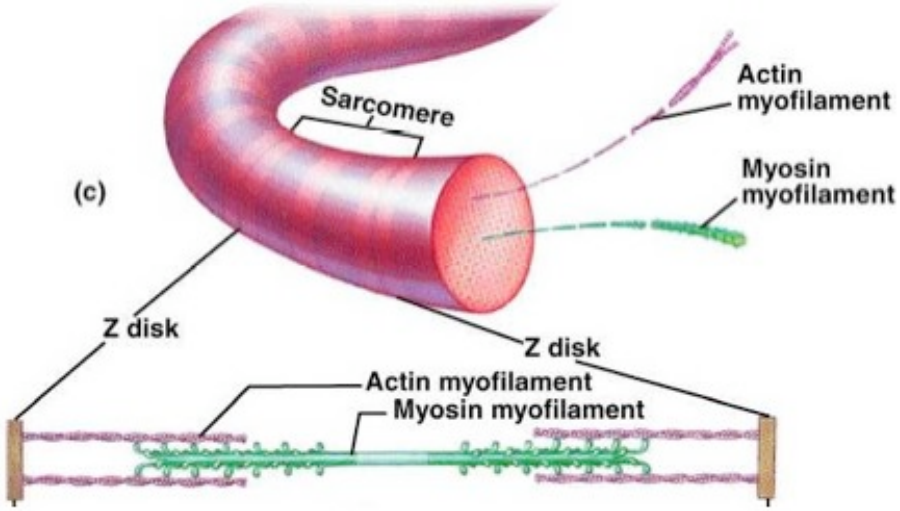


Figure 1. Sarcomere showing bridged and unbridged myosin and actin myofilaments

Stiffness of the muscle is controlled by the association (binding) of otherwise independent actin and myosin myofilaments in the sarcomere.

Figure C. 4 Shape-Changing Biological Design Example for Study 2 [196]

Electrorheological (ER) fluids

Electrorheological (ER) fluids are fluids that experience increased yield stress in the presence of electric fields. ER fluids consist of extremely small non-conducting particles suspended in an electrically insulating carrier fluid medium. In the absence of an electric field, ER fluids behave as typical fluids (Figure 1a). When an electric field is applied, these particles bind and the fluid immediately 'solidifies' with a yield point determined by the electric field strength (Fluid 1b).

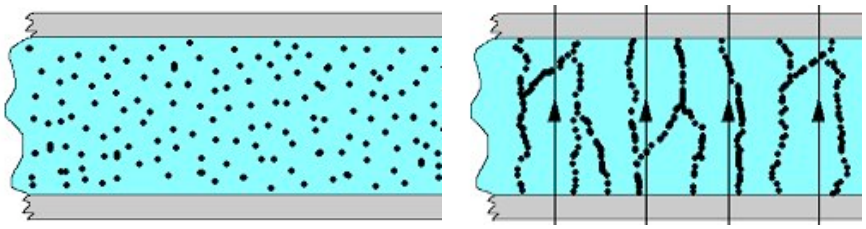


Figure 1. (a) ER fluid with zero electric field applied (b) ER fluid solidifies when electric field is applied

Stiffness in the ER fluid is changed by controlling the association (binding) between otherwise independent non-conducting particles.

Figure C. 5 Variable-Stiffness Human-Engineered Design Example for Study 2 [194]

Shape memory polymers

Shape memory polymers are polymers that can reversibly change shape via an external stimulus. SMPs are 2 –part block copolymers containing (1) a ‘switching‘ coil segment and (2) a rigid rod segment. Under external stimuli (ie. thermal, electric/magnetic field, light, or pH), the switching segment softens and allows the polymer to change to another predetermined shape.

Figure 1. Shape Memory Polymer

Due to the shape memory effect, SMPs can change from one shape to another based on an external stimulus.

Figure C. 6 Shape-Changing Human-Engineered Design Example for Study 2[197]

REFERENCES

[1] N. Dylla (1991). *Thinking Methods and Procedures in Mechanical Design*. Dissertation, Mechanical Design, Technical University of Munich.

[2] K. Schild, C. Herstatt and C. Luthje (2004). "How to Use Analogies for Breakthrough Innovations." Technical University of Hamburg, Hamburg, Germany.

[3] J. F. V. Vincent and D. L. Mann (2002). "Systematic Technology Transfer from Biology to Engineering. "*Philosophical Transactions of the Royal Society of London A*, 360, 159-73.

[4] M. K. Perttula (2006). *Idea Generation in Engineering Design: Application of Memory Search Perspective and Some Experimental Studies*. Doctoral Dissertation, Department of Mechanical Engineering, Helsinki University of Toronto.

[5] G. Pahl and W. Beitz (1996). *Engineering Design: A Systematic Approach* (2 ed.). London: Springer-Verlag.

[6] J. A. Busby and P. A. Lloyd (1999). "Influences on Solution Search Processes in Design Organisations. "*Research in Engineering Design*, 11, 158-71.

[7] H. A. Simon (1976). *Administrative Behavior*. New York, N.Y.: The Free Press.

[8] H. A. Simon (1983). *Reason in Human Affairs*. Stanford, CA: Stanford University Press.

[9] H. A. Simon (1996). *The Sciences of the Artificial* (3rd ed.). Cambridge, MA: MIT Press.

[10] S. Vogel (1998). *Cats' Paws and Catapults: Mechanical Worlds of Nature and People*. New York: W.W. Norton & Co.

[11] Y. Bar-Cohen (2006). *Biomimetics: Biologically Inspired Technologies*. Boca Raton: CRC Press.

[12] J. F. V. Vincent (2002). "Stealing Ideas from Nature." In S. Pellegrino (Ed. *Deployable Structures* (pp. 51-8). Italy: Springer Wein New York.

[13] J. F. V. Vincent, O. A. Bogatyreva, N. R. Bogatyrev, A. Bowyer and A.-K. Pahl (2006). "Biomimetics: Its Practice and Theory. "*Journal of the Royal Society Interface*, 3, 471-82.

[14] O. H. Schmitt (1969). "Some Interesting and Useful Biomimetic Transforms." *Thired International Biophysics Congress*, Boston, MA, August 29 - September 3, 1969.297.

[15] J. Benyus. "Case Studies." (2008). August 7, 2008 <http://www.biomimicryinstitute.org>.

[16] S. R. Bhatta and A. K. Goel (1996). "From Design Experiences to Generic Mechanisms : Model-Based Learning in Analogical Design. "*Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 10(2).

[17] S. R. Bhatta and A. K. Goel (1997). "A Functional Theory of Design Patterns. "*International Joint Conference on Artificial intelligence*, 15(1), 294-300.

[18] A. K. Goel, S. R. Bhatta and E. Stroulia (1997). "Kritik: An Early Case-Based Design System." In M. L. Maher and P. Pu (Eds.), *Issues and Applications of Case-Based Reasoning in Design* (pp. 87-132). Mahwah, NJ: Lawrence Erlbaum Associate.

[19] P. W. Yaner and A. K. Goel (2006). "From Form to Function: From Sbf to Dssbf." In J. S. Gero (Ed. *Design Computing and Cognition '06* (pp. 423-41). Springer Netherlands.

[20] S. Vattam, M. Helms and A. K. Goel (2007). "Biologically-Inspired Innovation in Engineering Design: A Cognitive Study." Design Intelligence Laboratory, School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA.

[21] http://database.portal.modwest.com/. Accessed

[22] I. Chiu and L. H. Shu (2004). "Natural Language Analysis for Biomimetic Design." *2004 DETC Design Theory and Methodology*, Salt Lake City, Utah. ASME, Paper No.DETC2004-57250

[23] I. Chiu and L. H. Shu (2005). "Bridging Cross-Domain Terminology for Biomimetic Design." *2005 IDETC Design Theory and Methodology*, Long Beach, CA. ASME, Paper No.DETC2005-84908

[24] A. Chakrabarti, P. Sarkar, B. Leelavathamma and B. S. Nataraju (2005). "A Functional Representation for Aiding Biomimetic and Artificial Inspiration of New Ideas. "*Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 19, 113-32.

[25] J. Benyus and D. Baumeister. "The Design Spiral." (2007). <www.biomimicryinstitute.org>.

[26] H. A. Bruck, A. L. Gershon, I. Golden, S. K. Gupta, L. S. G. Jr., E. B. Magrab and B. W. Spranklin (2006). "New Educational Tools and Curriculum Enhancements for Motivating Engineering Students to Design and Realize Bio-Inspired Products." In W. I. o. T. C. A. Brebbia (Ed. *Design and Nature Iii: Comparing Design in Nature with Science and Engineering*. WIT Press.

[27] Wordnet (2008). Available from: http://wordnet.princeton.edu/

[28] G. Altshuller (1984). *Creativity as an Exact Science*. New York, NY: Gordon & Breach.

[29] V. Vakili, I. Chiu, L. H. Shu, D. A. McAdams and R. B. Stone (2007). "Including Functional Models of Biological Phenomena as Design Stimuli." *ASME 2007 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Las Vegas, NV, September 4-7 (ASME, Ed.). ASME, Paper No.DETC2007-35776

[30] Z. Li, V. Raskin and K. Ramani (2008). "Developing Engineering Ontology for Information Retrieval. "*Journal of Computing and Information Science in Engineering*, 8(1), 011003-1-13.

[31] E. J. Chikofsky and J. H. C. II (1990). "Reverse Engineering and Design Recovery: A Taxonomy. "*IEEE Software*, 13-7.

[32] L. Chittaro and A. N. Kumar (1998). "Reasoning About Function and Its Applications. "*Artificial Intelligence in Engineering*, 12, 331-6.

[33] B. Buckley (2000). "Interactive Multimedia and Model-Based Learning in Biology. "*International Journal of Science Education*, 22(9), 895-935.

[34] Y.-M. Deng, S. B. Tor and G. A. Britton (2000). "Abstracting and Exploring Functional Design Information for Conceptual Mechanical Product Design. "*Engineering with Computers*, 16, 36-52.

[35] J. J. Shah, N. Vargas-Hernandez and S. M. Smith (2002). "Metrics for Measuring Ideation Effectiveness. "*Design Studies*, 24, 111-34.

[36] C. C. Seepersad, K. Pedersen, J. Emblemsvag, R. R. Bailey, J. K. Allen and F. Mistree (2005). "The Validation Square: How Does One Verify and Validate a Design Method?" In W. Chen, K. Lewis and L. Schmidt (Eds.), *Decision-Based Design: Making Effective Decisions in Product and Systems Design*. NY: ASME Press.

[37] C. B. Williams (2003). *Platform Design for Customizable Products and Processes with Non-Uniform Demand*, Mechanical Engineering, Georgia Institute of Technology.

[38] M. Messer (2008). *A Systematic Approach for Integrated Product, Materials, and Design-Process Design*. PhD Dissertation, GWW School of Mechanical Engineering, Georgia Institute of Technology.

[39] H. Casakin and G. Goldschmidt (1999). "Expertise and the Use of Visual Analogy: Implications for Design Education. "*Design Studies*, 20, 153-75.

[40] T. W. Mak and L. H. Shu (2004). "Use of Biological Phenomena in Design by Analogy." *ASME 2004 Design Engineering Technical Conference*, Salt Lake City, UT, September 28-October 2, 2004 (ASME, Ed.). ASME, Paper No. DETC2004-57303.

[41] B. T. Christensen and C. D. Schunn (2007). "The Relationship of Analogical Distance to Analogical Function and Pre-Inventive Structure: The Case of Engineering Design. "*Memory & Cognition*, 35(1), 29-38.

[42] J. Gregan-Paxton and D. R. John (1997). "Consumer Learning by Analogy: A Model of Internal Knowledge Transfer. "*Journal of Consumer Research*, 24(3), 266-84.

[43] D. W. Dahl and P. Moreau (2002). "The Influence and Value of Analogical Thinking During New Product Ideation. "*Journal of Marketing Research*, XXXIX, 47-60.

[44] D. Gentner and A. B. Markman (1997). "Structure Mapping in Analogy and Similarity. "*American Psychologist*, 52(1), 45-56.

[45] P. N. Johnson-Laird (Ed. (1989). *Analogy and the Exercise of Creativity*. New York, NY: Cambridge University Press.

[46] T. B. Ward (1998). "Analogical Distance and Purpose in Creative Thought: Mental Leaps Versus Mental Hops." In K. Holyoak, D. Gentner and B. Kokinov (Eds.), *Advances in Analogy Research: Integration of Theory and Data from the Cognitive, Computational, and Neural Sciences* (pp. 221-30). Sofia: New Bulgarian University.

[47] O. Benami and Y. Jin (2002). "Creative Stimulation in Conceptual Design." *2002 DETC Design Theory and Methodology*, Montreal, Canada (ASME, Ed.). ASME, Paper No.DETC2002/DTM-34023

[48] T. B. Ward (1993). "Structured Imagination: The Role of Category Structure in Exemplar Generation. "*Cognitive Psychology*, 27(1), 1-40.

[49] R. A. Finke, T. B. Ward and S. M. Smith (1992). *Creative Cognition: Theory, Research, and Applications*. Cambridge: A Bradford Book.

[50] A. B. Markman (1999). *Knowledge Representation*. Lawrence Erlbaum Associates.

[51] M. D. Williams, J. D. Hollan and A. L. Stevens (1983). "Human Reasoning About a Simple Physical System." In D. Gentner and A. L. Stevens (Eds.), *Mental Models*. L. Erlbaum Associates.

[52] M. J. Van Wie (2002). *Designing Product Architecture: A Systematic Method*. PhD Disssertation, School of Mechanical Engineering, The University of Texas at Austin.

[53] B. Y. White and J. R. Frederiksen (1990). "Causal Model Progressions as a Simple Physical System. "*Artificial Intelligence*, 42(1), 99-157.

[54] S. M. Kannapan and K. M. Marshek (1991). "Design Synthetic Reasoning. "*Mechanism and Machine Theory*, 26(7), 711-39.

[55] J. Winsor and K. MacCullum` (1994). "A Review of Functionality Modeling in Design. "*The Knowledge Engineering Review*, 9(2), 163-99.

[56] Y.-M. Deng (2002). "Function and Behavior Representation in Conceptual Mechanical Design. "*Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 16, 343-62.

[57] F. T. Brown (2007). *Engineering System Dynamics: A Unified Graph-Centered Approach*. Boca Raton: CRC Press.

[58] R. Zurawski (2005). "Petri Net Models, Functional Abstractions, and Reduction Techniques: Applications to the Design of Automated Manufacturing Systems. "*IEEE Transactions of Industrial Electronics*, 52(2), 595-609.

[59] T. Murata (1989). "Petri Nets: Properties, Analysis and Applications. "*Proceedings of the IEEE* 77(4), 541-80.

[60] T. Koga and K. Aoyama (2004). "Product Behavior and Topological Structure Design System by Step-by-Step Decomposition." *ASME 2004 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Salt Lake City, Utah, September 28-October 2, 2004. ASME, Paper No.DETC2004-57513

[61] E. Chang, X. Li and L. C. Schmidt (2001). "The Need for a Form, Function, and Behavior-Based Representation System." Designer Assistance Tool Laboratory, University of Maryland. <http://www.enme.umd.edu/DATLab>.

[62] Y. Umeda, T. Tomiyama and H. Yoshikawa (Eds.). (1990). *Function, Behavior, and Structure*. Berlin: Springer-Verlag.

[63] Y. Umeda, T. Tomiyama and H. Yoshikawa (1995). "Fbs Modeling: Modeling Scheme of Function for Conceptual Design." *9th International Workshop on Qualitative Reasoning*, Amsterdam, NL, May 11-19, 2005. Paper No.

[64] Y. Umeda and T. Tomiyama (1997). "Functional Reasoning in Design. "*IEEE Expert*, 12(2), 42-8.

[65] B. Chandrasekaran, A. K. Goel and Y. Iwasaki (1993). "Functional Representation as Design Rationale. "*IEEE Computer*, 26(1), 48-56.

[66] B. Chandrasekaran and J. R. Josephson (2000). "Function in Device Representation. "*Engineering with Computers*, 16, 162-77.

[67] J. S. Gero (1990). "Design Prototypes: A Knowledge Representation Schema for Design " *AI Magazine*, 11(4), 26-36.

[68] L. Qian and J. S. Gero (1996). "Function-Behavior-Structure Paths and Their Role in Analogy-Based Design. "*Artificial Intelligence in Engineering*, 10, 289-312.

[69] J. S. Gero and U. Kannengiesser (2006). "A Function-Behaviour-Structure Ontology of Processes." In J. S. Gero (Ed. *Design Computing and Cognition '06* (pp. 407-22). Springer.

[70] S. Szykman, R. D. Sriram, C. Bochenek, J. W. Racz and J. Senfaute (2000). "Design Repositories: Engineering Design's New Knowledge Base." National Institute of Science and Technology (NIST), Gaithersburg, MD.

[71] A. Aamodt and E. Plaza (1994). "Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches "*AI Communications*, 7(1), 39-59.

[72] F. Marir and I. Watson (1994). "Case-Based Reasoning: A Categorized Bibliography. "*The Knowledge Engineering Review*, 9(4), 382-419.

[73] I. Watson and F. Marir (1994). "Case-Based Reasoning: A Review. "*The Knowledge Engineering Review*, 9(4), 355-81.

[74] R. L. d. Mantaras and E. Plaza (1997). "Case-Based Reasoning: An Overview. "*AI Communications*, 10, 21-9.

[75] S. Yim (2007). *A Retrieval Method (Dfm Framework) for Automated Retrieval of Design for Additive Manufacturing Problems*. PhD Dissertation, GWW School of Mechanical Engineering, Georgia Institute of Technology.

[76] J. J. Shah and P. K. Wright (2000). "Developing Theoretical Foundations of Dfm." *International Design Engineering Technical Conferences and Computers and Information in Engineering Conferences*, Baltimore, MD. ASME

[77] J. Kim, P. Will, S. R. Ling and B. Neches (2003). "Knowledge-Rich Catalog Services for Engineering Desing. "*Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 17(4), 349-66.

[78] Z. Li, M. Liu and K. Ramani (2004). "Reveiw of Product Information Retreival: Representation and Indexing." *ASME 2004 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Salt Lake City, UT, September 28-October 2 (ASME, Ed.). Paper No.DETC2004-57749

[79] Z. Li, D. C. Anderson and K. Ramani (2005). "Ontology-Based Design Knowledge Modeling for Product Retrieval." *International Conference on Engineering Design*, Melbourne, August 15-18. Paper No.ICED05/462.1

[80] S. Ahmed, S. Kim and K. M. Wallace (2007). "A Methodology for Creating Ontologies for Engineering Design. "*Journal of Computing and Information Science in Engineering*, 7, 132-40.

[81] Z. Li and K. Ramani (2007). "Ontology-Based Design Information Extraction and Retrieval. "*Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 21, 137-54.

[82] N. F. Noy and D. L. McGuinness (2001). "Ontology Development 101: A Guide to Creating Your First Ontology." Stanford Knowledge Systems Laboratory.

[83] Z. Li, M. Liu, D. C. Anderson and K. Ramani (2005). "Semantics-Based Design Knowledge Annotation and Retrieval." *ASME 2005 International Design Engineering Technical Conferences & Computer and Information in Engineering Conference*, Long Beach, CA, September 24-28. ASME, Paper No.DETC2005-85107

[84] J. B. Kopena, C. B. Cera and W. C. Regli (2005). "Conceptual Design Knowledge Management and the Semantic Web." *International Design Engineering Technical Conferences and Computers and Information in Engineering Conferences*, Long Beach, CA, September 24-28 (ASME, Ed.). Paper No.DETC2005-85310

[85] N. Udoyen (2006). *Information Modeling for Intent-Based Retrieval of Parametric Finite Element Analysis Models*. PhD Dissertation, GWW School of Mechanical Engineering, Georgia Institute of Technology.

[86] S. Yim, J. Wilson and D. Rosen (2008). "Development of an Ontology for Bio-Inspired Design Using Description Logics." *International Conference on Product Lifecycle Management*, Seoul, Korea.

[87] F. Azuaje, W. Dubitzky, N. Black and K. Adamson (2000). "Retrieval Strategies for Case-Based Reasoning: A Categorized Bibliography. "*The Knowledge Engineering Review*, 15, 371-9.

[88] F. Baader, D. Calvanese, D. McGuinness, D. Nardi and P. F. Patel-Schneider (2002). "The Description Logic Handbook.

[89] J. Z. Pan (2004). *Description Logics: Reasoning Support for the Semantic Web*, School of Computer Science,, University of Manchester.

[90] D. Ullman (2003). *The Mechanical Design Process* (3rd ed.). McGraw-Hill.

[91] D. W. McShea (2001). "The Hierarchical Structure of Organisms: A Scale and Documentation of a Trend in the Maximum. "*Paleobiology*, 27(2), 405-23.

[92] G. A. J. M. Jagers op Akkerhuis (2008). "Analysing Hierarchy in the Organization of Biological and Physical Systems. "*Biological Reviews*, 83(1), 1-12.

[93] Human Muscle. ≤http://www.cliffsnotes.com/WileyCDA/CliffsReviewTopic/topicArticleId-8741,articleId-8717.html>. Accessed May 7, 2008

[94] D. W. McShea (2000). "Functional Complexity in Organisms: Parts as Proxies. "*Biology and Philosophy*, 15, 641-68.

[95] S. A. Kauffman (1993). *The Origins of Order: Self-Organization and Selection in Evolution*. New york: Oxford University Press, Inc.

[96] J. W. Valentine (2006). *On the Origin of Phyla*. University of Chicago Press.

[97] X. F. Zha and H. Du (2001). "Mechanical Systems and Assemblies Modeling Using Knowledge-Insensitive Petri Nets Formalisms. "*Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 15, 145-71.

[98] J. G. Miller (1978). *Living Systems*. New York: McGraw-Hill, Inc.

[99] D. W. McShea and E. P. Venit (2001). "What Is a Part?" In G. P. Wagner (Ed. *The Character Concept in Evolutionary Biology* (pp. 259-84). San Diego: Academic Press.

[100] G. P. Wagner and L. Altenberg` (1996). "Complex Adaptations and the Evolution of Evolvability. "*Evolution*, 50(3), 967-76.

[101] B. Chandrasekaran and J. R. Josephson (1996). "Representing Function as Effedt." *Fifth International Workshop on Advances in Functional Modeling of Complex Technical Systems*, Paris, France, July 1997 (M. Modarres, Ed.). The Center for Technology Risk Studies, University of Maryland.3-16.

[102] M. Scaife and Y. Rogers (1996). "External Cognition: How Do Graphical Representations Work? "*International Journal of Human-Computur Studies*, 45, 185-213.

[103] T. N. Madhusudan, K. Sycara and D. Navin-Chandra (1995). "Device Representation for Behavioral Synthesis of Mechatronic Devices." Carnegie Mellon University. <http//:citeseer.ist.psu.edu/46706.html>.

[104] R. Zurawski and M. Zhou (1994). "Petri Nets and Industrial Applications: A Tutorial. "*IEEE Transactions of Industrial Electronics*, 41(6), 567-83.

[105] J. D. Summers and J. J. Shah (2004). "Representation in Engineering Design: A Framework for Classification." *ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Salt Lake City, Utah, USA, September 28-October 2, 2004. ASME, Paper No.DETC2004-57514

[106] J. H. Larkin and H. A. Simon (1987). "Why a Diagram Is (Sometimes) Worth Ten Thousand Words. "*Cognitive Science*, 11, 65-99.

[107] C. Gurr, J. Lee and K. Stenning (1998). "Theories of Diagrammatic Reasoning: Distinguishing Component Problems. "*Minds and Machines*, 8(533-557), 1998.

[108] M. K. Kaiser, D. R. Proffitt, S. M. Whelan and H. Hecht (1992). "Influence of Animation on Dynamical Judgments. "*Journal of Experimental Psychology: Human Perception and Performance*, 18(3), 669-89.

[109] S. Jones and M. Scaife (2000). "Animated Diagrams: An Investigation into the Cognitive Effects of Using Animation to Illustrate Dynamic Processes." In M. Anderson and P. Cheng (Eds.), *Theory & Applications of Diagrams. Lecture Notes in Artificial Intelligence, No. 1889* (Vol. 231-244). Berlin: Springer-Verlag.

[110] K. Stenning and J. Oberlander (1995). "A Cognitive Theory of Graphical and Linguistic Reasoning: Logic and Implementation. "*Cognitive Science*, 19, 97-140.

[111] M. Peleg, D. Rubin and R. B. Altman (2005). "Using Petri Net Tools to Study Properties and Dynamics of Biological Systems. "*Journal of the American Medical Informatics Association*, 12(2), 181-99.

[112] M. Peleg, I. Yeh and R. B. Altman (2002). "Modeling Biological Processes Using Workflow and Petri Net Models. "*Bioinformatics*, 18, 825-37.

[113] M. Chen and R. Hofestaedt (2003). "Quantitative Petri Net Model of Gene Regulated Metabolic Networks in the Cell. "*Silico Biology*, 3(3), 347-65.

[114] H. Matsuno, Y. Tanaka, H. Aoshima, A. Doi, M. Matsui and S. Miyano (2003). "Biopathways Representation and Simulation on Hybrid Functional Petri Net. "*Silico Biology*, 3(3), 389-404.

[115] K. Voss, M. Heiner and I. Koch (2003). "Steady State Analysis of Metabolic Pathways Using Petri Nets. "*Silico Biology*, 3(3), 367-87.

[116] I. Zevedei-Oancea and S. Schuster (2003). "Topological Analysis of Metabolic Networks Based on Petri Net Theory. "*Silico Biology* 3, 323-45.

[117] I. Koch, B. H. Junker and M. Heiner (2005). "Application of Petri Net Theory for Modelling and Validation of the Sucrose Breakdown Pathway in the Potato Tuber "*Bioinformatics*, 21(7), 1219-26.

[118] J. L. Peterson (1977). "Petri Nets. "*Computing Surveys*, 9(3), 223-52.

[119] K.-H. Lee and J. Favrel (1985). "Hierarchical Reduction Method for Analysis and Decomposition of Petri Nets. "*IEEE transactions on Systems, Man, and Cybernetics*, 15(2), 272-80.

[120] B. Farwer and K. Misra (2003). "Modelling with Hierarchical Object Petri Nets. "*Fundamenta Informaticae*, 55(2), 129-47.

[121] L. Zerguini (2004). "A Novel Hierarchical Method for Decomposition and Design of Workflow Models. "*Transactions of the Society for Design and Process Science*, 8(2), 65-74.

[122] B. Farwer and M. Varea (2006). "Separation of Control and Data Flow in High-Level Petri Nets: Transforming Dual Flow Nets in Object Petri Nets. "*Fundamenta Informaticae*, 72, 123-37.

[123] K. Van Hee, I. A. Lomazova, O. Oanea, A. Serebrenik, N. Sidorova and M. Voorhoeve (2006). "Nested Nets for Adaptive Systems." *27th International Conference on Applications and Theory of Petri Nets* June 26-30.241-60.

[124] M. G. Rekoff Jr. (1985). "On Reverse Engineering." *IEEE Trans. Systems, Man, and Cybernetics*, March-April 1985.244-52.

[125] G. P. Wagner (1995). "Adaptation and the Modular Design of Organisms." In F. Morán, A. Morán, J. J. Merelo and P. Chacón (Eds.), *Advances in Artificial Life* (pp. 317-28). Berlin: Springer Verlag.

[126] T. Motokawa (1984). "Viscoelasticity of Holothurian Body Wall. "*Journal of Experimental Biology*, 109, 63-75.

[127] J. A. Trotter and T. J. Koob (1995). "Evidence That Calcium-Dependent Cellular Processes Are Involved in the Stiffening Response of Holothurian Dermis and That Dermal Cells Contain Organic Stiffening Factor. "*The Journal of Experimental Biology*, 198, 1951-61.

[128] F. A. Thurmond and J. A. Trotter (1996). "Morphology and Biomechanics of the Microfibrillar Network of Sea Cucumber Dermis. "*The Journal of Experimental Biology*, 199, 1817-28.

[129] G. K. Szulgit and R. E. Shadwick (2000). "Dynamic Mechanical Characterization of a Mutable Collagenous Tissue: Response of Sea Cucumber Dermis to Cell Lysis and Dermal Extracts. "*The Journal of Experimental Biology*, 203, 1539-50.

[130] J. A. Trotter, J. Tipper, G. Lyons-Levy, K. Chino, A. H. Heuer, Z. Liu, M. Mrksich, C. Hodneland, W. S. Dillmore, T. J. Koobs-Edmunds, K. Kadler and D. Holmes (2000). "Towards a Fibrous Composite with Dynamically Controlled

Stiffness: Lessons from Echinoderms. "*Biochemical Society Transactions*, 28, 357-62.

[131] T. Motokawa and A. Tsuchi (2003). "Dynamic Mechanical Properties of Body-Wall Dermis in Various Mechanical States and Their Implications for the Behavior of Sea Cucumbers. "*Biological Bulletin*, 205, 261-75.

[132] A. Summers (2003). "Catch and Release." *Natural History*, 112(9)

[133] N. Takemae and T. Motokawa (2005). "Mechanical Properties of the Isolate Catch Apparatus of the Sea Urchin Spine Joint: Muscle Fibers Do Not Contribute to Passive Stiffness Changes. "*Biological Bulletin*, 208, 29-35.

[134] D. Moffett, S. Moffett and C. Schauf (1993). *Human Physiology: Foundations and Frontiers*. Dubuque, Iowa: WCB Publishers.

[135] W. J. Germann and C. L. Stanfield (2002). *Principles of Human Physiology*. San Francisco: Benjamin Cummings.

[136] V. Vakili and L. H. Shu (2001). "Towards Biomimetic Concept Generation." *2001 DETC Design Theory and Methodology*, Pittsburgh, PA, September 9-12. ASME

[137] S. Ahmed and K. M. Wallace (2004). "Identifying and Supporting the Knowldedge Needs of Novice Designers within the Aerospace Industry. "*Journal of Engineering Design*, 15(5), 475-92.

[138] F. Baader, D. Calvanese, D. McGuinness, D. Nardi and P. F. Patel-Schneider (2002). *The Description Logic Handbook*. Cambridge University Press.

[139] G. M. Mocko (2006). *A Knowledge Framework for Integrating Multiple Perspectives in Decision-Centric Design*, School of Mechanical Engineering, Georgia Institute of Technology.

[140] M. C. Daconta, L. J. Obrst and K. T. Smith (2003). *The Semantic Web*. Indianapolis, Indiana: Wiley.

[141] S. Powers (2003). *Practical Rdf*. Sebastopol, CA: O'Reilly & Associates.

[142] Web Ontology Language (Owl). http://www.w3.org/2004/OWL/. Accessed

[143] S. Bechhofer (2003). "The Dig Description Logic Interface: Dig/1.1." University of Manchester, Manchester, U.K. <www.sts.tu-harburg.de/~r.f.moeller/racer/interface1.1.pdf>.

[144] B. Ganter and R. Wille (1999). *Formal Concept Analysis: Mathematical Foundations.* New York: Springer.

[145] N. Udoyen (2006). *Information Modeling for Intent-Based Retrieval of Parametric Finite Element Analysis Models*, GWW Woodruff school of Mechanical Engineering, Georgia Institute of Technology.

[146] C. J. Van Rijsbergen (1979). *Information Retrieval*. London: Buttersworth.

[147] J. K. Stroble, R. B. Stone, D. A. McAdams, I. Chiu, L. H. Shu and S. E. Watkins (2008). "Generating Biological Solutions to Engineering Problems Using an Organized Search." *ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference* Brooklyn, NY, August 3-6 (ASME, Ed.). ASME, Paper No.DETC2008/DTM-49368

[148] R. Moller, V. Haarslev and M. Wessel. "On the Scalability of Description Logic Instance Retrieval." (2006). <http://www.sts.tu-harburg.de/~r.f.moeller/racer/HaMW05.pdf.>.

[149] D. G. Jannson and S. M. Smith (1991). "Design Fixation. "*Design Studies*, 12(1).

[150] A. T. Purcell, P. Williams, J. S. Gero and B. Colbron (1993). "Fixation Effects: Do They Exist in Design Problem Solving? "*Environment and Planning B: Planning and Design*, 20(3), 333-45.

[151] A. T. Purcell and S. J. Gero (1996). "Design and Other Types of Fixation. "*Design Studies*, 17(4), 363-83.

[152] R. L. Marsh, J. D. Landau and J. L. Hicks (1996). "How Examples May (and May Not) Constrain Creativity. "*Memory & Cognition*, 24(3), 669-80.

[153] R. L. Marsh, M. L. Bink and J. L. Hicks (1999). "Conceptual Priming in a Generative Problem-Solving Task. "*Memory & Cognition*, 27(2), 355-63.

[154] P. B. Paulus and H.-C. Yang (2000). "Idea Generation in Groups: A Basis for Creativity in Organizations. "*Organizational Behavior and Human Decision Processes*, 82(1), 76-87.

[155] B. A. Nijstad, W. Stroebe and H. F. Lodewijkx (2002). "Cognitive Stimulation and Interference in Groups: Exposure Effects in an Idea Generation Task. "*Journal of Experimental Social Psychology*, 38(6), 535-44.

[156] K. L. Dugosh, P. B. Paulus, E. J. Roland and H.-C. Yang (2000). "Cognitive Stimulation in Brainstorming. "*Journal of Experimental Social Psychology*, 79(5), 722-35.

[157] B. A. Nelson, J. O. Wilson, J. Yen and D. Rosen (2008).

[158] J. Aboudi (1999). "Effective Behavior and Dynamic Response Modeling of Electro-Rheological and Magneto-Rheological Fluid Composites. "*Smart Materials and Structures*, 8, 106-15.

[159] Y. K. Kang, J. Kim and S.-B. Choi (2001). "Passive and Active Damping Characteristics of Smart Electro-Rheological Composite Beams. "*Smart Materials and Structures*, 10, 724-9.

[160] K.-D. Cho, I. Lee and J.-H. Han (2005). "Dynamic Characteristics of Er Fluid-Filled Composite Plate Using Multielectrode Configuration. "*Journal of Intelligent Material Systems and Structures*, 16, 411-9.

[161] H. J. Choi, S. J. Park, S. T. Kim and M. S. Jhon (2005). "Electrorheological Application of Polyaniline/Multi-Walled Carbon Nanotube Composites. "*Diamond & Related Materials*, 14, 766-9.

[162] H. Pu and F. Jiang (2005). "Towards High Sedimentation Stability: Magnetorheological Fluids Based on Cnt/Fe3o4 Nanocomposites. "*Nanotechnology*, 16, 1486-9.

[163] R. Wirtz, T. Zhao and Y. Jiang (2004). "Thermal and Mechanical Characterics of a Multi-Functional Thermal Energy Storage Structure." *IEEE: 2004 International Society Conference on Thermal Phenomena*.549-56.

[164] K. Pielichowski and K. Flejtuch (2005). "Recent Developments in Polymeric Phase Change Materials for Energy Storage: Poly(Ethylene Oxide)/Stearic Acid Blends. "*Polymers and Advanced Technologies*, 16, 127-32.

[165] M. Abdulrahim and R. Lind (2006). "Using Avian Morphology to Enhance Aircraft Maneuverability." *AIAA Atmospheric Flight Mechanics Conference*, Keystone, CO, August 21-24. AIAA, Paper No.AIAA 2006-6643

[166] J. Bowman, M. B. Sanders and D. T. Weisshar (2002). "Evaluating the Impact of Morphing Technologies on Aircraft Performance." *43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Denver, CO, April 22-25. AIAA, Paper No.AIAA 2002-1631

[167] U. M. L. Norberg (2002). "Structure, Form, and Function of Flight in Engineering and the Living World. "*Journal of Morphology*, 252, 52-81.

[168] A. C. Carruthers, G. K. Taylor, S. M. Walker and A. L. R. Thomas (2007). "Use and Function of a Leading Edge Flap on the Wings of Eagles." *45th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, January 8-11. AIAA, Paper No.AIAA 2007-43

[169] A. M. Millbrooke (2006). *Aviation History*. Englewood, CO: Jeppensen Sanderson.

[170] E. T. Raymond and C. C. Chenoweth (1993). *Aircraft Flight Control Actuation System Design*. Warrendale, PA: Society of Automotive Engineers.

[171] J. John D. Anderson (1997). *A History of Aerodynamics and Its Impact on Flying Machines*. Cambridge, UK: Cambridge University Press.

[172] Plane Wing by Piotr Jaworski Available from: commons.wikimedia.org/wiki/Image:PlaneWing.png

[173] A. K. Jha and J. N. Kudva (2004). "Morphing Aircraft Concepts, Classifications, and Challenges." *Smart Structures and Materials 2004: industrial and Commercial Applications of Smart Structures and Technologies*, Bellingham, WA (E. H. Anderson, Ed.). SPIE

[174] Flexsys, Inc. www.flxsys.com. Accessed February 2, 2008

[175] Cheffers Uk. http://www.cheffers.co.uk. Accessed April 6, 2008

[176] J. C. Jennette, R. H. Hepinstall, J. L. Olson, M. M. Schwartz and F. G. Silva (2006). *Hepinstall's Pathology of the Kidney (Sixth Edition)* (Sixth ed.). Wolters Kluwer Health.

[177] Davita, Inc. http://www.davita.com/dialysis. Accessed May 27, 2008

[178] P. D. L. G. Forni and M. D. P. J. Hilton (1997). "Continuous Hemofiltration in the Treatment of Acute Renal Failure. "*The New England Journal of Medicing*, 336(18), 1303-9.

[179] Asahi-Kasei, Inc. http://www.asahi-kasei.co.jp/medical/en/apheresis/therapies/crrt.html. Accessed May 27, 2008

[180] F. Maduell (2005). "Hemodiafiltration. "*Hemodialysis International*, 9, 47-55.

[181] R. A. Ward, B. Schmidt, J. Hullin, G. F. Hillerbrand and W. Samtleben (2000). "A Comparison of on-Line Hemodiafiltration and High-Flux Hemodialysis: A Prospective Clinical Study. "*Journal of the American Society of Nephrology*, 11, 2344-50.

[182] P. Ahrenholz, R. E. Winkler, W. Ramlow, M. Tiess and W. Muller (1997). "On-Line Hemodiafiltration with Pre- and Postdilution: A Comparison of Efficacy. "*International Journal of Artificial Organs*, 20(2), 81-90.

[183] P. B. Kerr, A. Argiles, J. L. Flavier, B. Canaud and C. M. Mion (1992). "Comparison of Hemodialysis and Hemodiafiltration: A Long-Term Longitudinal Study. "*Kidney International*, 41(4), 1035-40.

[184] W. Lornoy, Y. Becaus, J. M. Billiouw, L. Sierens and P. v. Malderen (1998). "Remarkable Removal of Beta-2-Microglobin by on-Line Hemodiafiltration. "*american Journal of Nephrology*, 18(2), 105-8.

[185] A. K. Cheung, M. V. Rocco, G. Yan, J. K. Leypoldt, N. W. Levin, T. Greene, L. Agodoa, J. Bailey, G. J. Beck, W. Clark, A. S. Levey, D. B. Ornt, G. Schulman, S. Schwab, B. Teehan, G. Eknoyan and H. S. Group (2006). "Serum {Beta}-2 Microglobulin Levels Predict Mortality in Dialysis Patients: Results of the Hemo Study. " *Journal of the American Society of Nephrology*, 17, 546-55.

[186] J. J. B. Petrie, T. G. Ng and C. M. Hawley (2008). "Review Article: Is It Time to Embrace Haemodiafiltration for Centre-Based Haemodialysis. "*Nephrology*, 13, 269-77.

[187] L. M. Dember and B. L. Jaber (2006). "Dialysis-Related Amyloidosis: Late Finding or Hidden Epidemic? "*Seminars in Dialysis*(19), 105-9.

[188] B. Canaud, J. L. Bragg-Gresham, M. R. Marshall, S. Desmeules, B. W. Gillespie, T. Depner, P. Klassen and F. K. Port (2006). "Mortality Risk for Patients Receiving Hemodiafiltration Versus Hemodialysis: European Results from Dopps. "*Kidney International*, 69(2087-2093).

[189] T. Jirka, S. Cesare, D. Benedetto, M. Perera Chang, P. Ponce, N. Richards, C. Tetta, L. Vaslaky, B. Canaud, J. L. Bragg- Gresham, M. R. Marshall, S. Desmeules, B. W. Gillespie, T. Depner, P. Klassen and F. K. Port (2006). "Mortality Risk for Patients Receiving Hemodiafiltration Versus Haemodialysis. "*Kidney International*, 70, 1524.

[190] K. Pedersen, J. Emblemsvag, J. K. Allen and F. Mistree (2000). "Validating Design Methods and Reserch: The Validation Square." *ASME Design Theory and Methodology*, Baltimore, MD (ASME, Ed.). ASME

[191] Stella: Systems Thinking for Education and Research. http://www.iseesystems.com/softwares/Education/StellaSoftware.aspx. Accessed September 30, 2008

[192] An Overview of Sysml, the Systems Modeling Language. www.nohau.se/images/nec/A3-Graham-Bleakley-NEC2006AnOverviewofSysMLGBI-Logix.pdf. Accessed September 30, 2008

[193] Qpme. http://sdq.ipd.uka.de/people/samuel_kounev/projects/QPME. Accessed October 25, 2008

[194] Electrorheological Fluids. http://www.juliantrubin.com/encyclopedia/electricity/magnetorheological_fluid.html. Accessed

[195] Venus Fly Trap. http://bottleworld.net/?m=200609. Accessed January 3, 2008

[196] Human Muscle. http://academic.wsc.edu/faculty/jatodd1/351/ch6outline.html. Accessed January 3, 2008

[197] Shape Memory Polymers. http://everythang.wordpress.com/2006/10/23/self-repairing-car-parts-the-fender-un-bender/. Accessed January 3, 2008