Trends and Perspectives

Design and Operation of Large Systems

Nam P. Suh, Massachusetts Institute of Technology, Cambridge, Massachusetts

Abstract

Issues related to the design of large systems are explored based on the axiomatic approach to design. A large system has the characteristic of having to satisfy varying sets of functional requirements (FRs) over its lifetime, some of which are not known a priori. Such a system must therefore be designed to accommodate these varying and sometimes unknown sets of functional requirements. The design must also allow the system to have an expandable database of design parameters (DPs) to satisfy yet-unknown FRs and endow the system with "intelligence" to make correct decisions. The independence axiom and the information axiom are the basis for several theorems that are derived for large systems. These theorems characterize or establish the bounds on the large systems. The implications of these theorems are examined using organizational design and the design of intelligent manufacturing systems as examples. Finally, the human cognitive process, including human creativity, is discussed as an example of a large system.

Keywords: Large Systems, Manufacturing Systems, Design Theory, Optimization

Introduction

In recent years, the design and operation of large systems has attracted the interest of many researchers. This may, in part, be due to the ever-increasing size of technical systems, industrial enterprises, and government organizations. This trend is likely to continue in the future because rapid access to and management of information makes efficient operation of such large systems possible. Yet, the design and operation of large systems is still based on empiricism and heuristics. In the future, however, to be competitive industrial firms must have a rational strategy for design and management of large products, their enterprises, and their manufacturing operations; likewise, to be efficient in handling societal programs such as health care, governments need administrative systems that are rationally designed. Cost-efficient and responsive designs of such systems can be created reliably only through the development of a firm science base.

What is a large system? Without an operational definition of a large system, it will be difficult to describe the design process itself. The common definition of large systems, which is based on a number of parts or on physical size, is found to be insufficient because it is not consistent with the design process. In spite of the difficulty in developing a definition for large systems, most people, including researchers and engineers, seem to know intuitively what we mean by large systems. The telephone system for Boston, the government bureaucracy, an assembly plant for automobiles, a software system for control of nuclear power plants, and the Boeing 747 airplane seem to fit the term large systems because they are physically large, have many components, or have many lines of computer code. The examples cited may constitute large systems—but are they? We can answer this question only after the definition of a large system is clearly established.

The purpose of this paper is to lay the theoretical foundation for the design of large systems based on axiomatic design. To assist those readers who are not familiar with earlier work of the author on axiomatic design, the framework for axiomatic design will be presented first. Based on this understanding of the design process, a definition of a large system is given, followed by detailed discussions on large system design.

Axiomatic Design Framework

The basic idea for axiomatic design was advanced in the mid-1970s, and it was later published in a book in 1990. The basic postulate is that there are fundamental axioms that govern the design process. Two axioms were identified by examining the common elements that are always present in good design—be it product, process, or systems design. The first axiom is called the independence axiom, which states that the independence of functional requirements (FRs) must always be maintained,
where FRs are defined as the minimum number of independent requirements that characterize the design goals. The second axiom is called the information axiom, which states that, among those designs that satisfy the independence axiom, the design with the highest probability of functional success is the best design. Based on these design axioms, we can derive theorems and corollaries.

The design world of the axiomatic approach is made up of four domains: customer domain, functional domain, physical domain, and process domain. The domain structure is schematically illustrated in Figure 1. As shown in Table 1, all design tasks (product design, process design, system design, organization design, software design, and material design) are contained in these four domains. For example, in the case of manufacturing systems, customer attributes (CAs) may be the attributes desired by internal or external customers; functional requirements (FRs) may be flexibility, controllability, efficiency, and uniqueness; design parameters (DPs) may be the layouts (U-shaped cells and so on) and arrangements (the design of the manufacturing elements themselves as composed of physical elements); and process variables (PVs) may be machine tools, people, tooling/tools, material handling, and so on. In Figure 1, a domain on the left relative to another domain on the right repre-

### Table 1

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Attributes which consumers desire</td>
<td>Functional requirements specified for the product</td>
<td>Physical variables which can satisfy the functional requirements</td>
<td>Process variables that can control design parameters (DPs)</td>
</tr>
<tr>
<td>(b) Materials</td>
<td>Desired performance</td>
<td>Required properties</td>
<td>Microstructure</td>
<td>Processes</td>
</tr>
<tr>
<td>(c) Software</td>
<td>Attributes desired in the software</td>
<td>Output</td>
<td>Input variables or algorithms</td>
<td>Subroutines</td>
</tr>
<tr>
<td>(d) Organization</td>
<td>Customer satisfaction</td>
<td>Functions of the organization</td>
<td>Programs or offices</td>
<td>People and other resources that can support the programs</td>
</tr>
<tr>
<td>(e) Systems</td>
<td>Attributes desired of the overall system</td>
<td>Functional requirements of the system</td>
<td>Machines or components, subcomponents</td>
<td>Resources (human, financial, materials, etc.)</td>
</tr>
</tbody>
</table>
resents "what we want to achieve." The domain on the right represents the proposed design solution of "how we choose to satisfy the requirements specified in the left domain." To go from "what" to "how" requires mapping. During this mapping process, the independence axiom must be satisfied.

At a given level of design hierarchy, the set of functional requirements that define the specific design goals constitutes a vector, \( \{\text{FRs}\} \). Similarly, the set of design parameters in the physical domain that are the "hows" for the functional requirements also constitutes a vector, \( \{\text{DPs}\} \). The relationship between these two vectors can be written as follows:

\[
\{\text{FRs}\} = [A] \{\text{DPs}\} \tag{1}
\]

where \([A]\) is the design matrix. Eq. (1) is called the design equation for the product. The independence axiom may be stated as follows:

**Axiom 1: Independence Axiom**

Maintain the independence of functional requirements

To satisfy the independence axiom, the matrix must be either diagonal or triangular. When \([A]\) is diagonal, each of the functional requirements can be satisfied independently by means of one design parameter. Such a design is called an uncoupled design. When the matrix is triangular, the independence of the functional requirements can be guaranteed if and only if the design parameters are changed in the proper sequence. Such a design is called a decoupled or quasi-coupled design.

In process design, a similar relationship exists between the design parameter vector, \( \{\text{DPs}\} \), of the physical domain and process variable vector, \( \{\text{PVs}\} \), of the process domain, where the process variables are the "hows" with respect to the "whats" given by the design parameters. The design equation for a process may be written as follows:

\[
\{\text{DPs}\} = [B] \{\text{PVs}\} \tag{2}
\]

Substitution of Eq. (2) into Eq. (1) relates the functional requirements for a product to the process variables of the process domain; therefore, for concurrent engineering to be possible, the product \([A][B]\) must also be diagonal or triangular.

Functional requirements, design parameters, and process variables can be decomposed; however, contrary to the conventional assumption on decompositi-
tion content is a quantitative measure of complexity. According to Eq. (2), complex systems require more information to make the system function. According to this view, a large system is not necessarily complex, and even a small system can be complex if the probability of its success is low. Then why do we often think that a large system with many physical elements or components is more complex? The answer is that in axiomatic design, the information is related to tolerances. To construct a large system with many elements or components, tight tolerances are often specified for the components so that the system assembled of these components performs the required functions. To produce parts with tight tolerances using a given set of machines, more information may be required to make them, and thus a large system with many components appears to be complex.

The probability of success can be computed by comparing the design range (dr), which is the range of desired values for the functional requirements (or the design parameters in the case of process design) specified by the designer, with the system range (sr), which refers to the range of possible values that the proposed solution can provide to satisfy the functional requirements (or design parameters). Figure 2 illustrates these two ranges graphically. The system range is plotted as a probability density function (pdf) versus the functional requirement (FR). The overlap between the design range and the system range is called the common range (cr), and this is the only region where the design requirements are satisfied. For example, if a designer wishes to have a rod of 1 m ± 10 μm in length, the design range is from (1 m - 10 μm) to (1 m + 10 μm). If the tool chosen to cut the rod to the length is a hacksaw, the system range may be 1 m ± 3 mm. The probability of achieving the design goal is very small if the hacksaw is the only tool available to produce the rod; therefore, the proposed solution is very complex.

The probability of success is the area of the common range divided by the area under the system range shown in Figure 2. Then, the information content may be expressed as follows:

$$I = \log \left( \frac{A_{cr}}{A_{sr}} \right)$$ (4)

where $A_{sr}$ denotes the area under the system range and $A_{cr}$ is the area of the common range. Furthermore, because $A_{sr} = 1.0$ in most cases and there are $n$ FRs to satisfy, Eq. (4) may be revised as follows:

$$I = \sum_{i=1}^{n} \log \left( \frac{1}{A_{cr}} \right)$$ (5)

In an ideal uncoupled design, the number of functional requirements, design parameters, and process variables are the same, and the information content is zero. In this case, the design matrices are diagonal and $A_{sr} = A_{cr}$ for all requirements. This fact can be stated as a theorem. Many theorems and corollaries have been derived based on the two design axioms. Details of axiomatic design can be found in Suh.\textsuperscript{2}

**Definition of a Large System**

From the foregoing discussion of axiomatic design, it can be seen that the number of components in a piece of hardware or the number of lines of code in a software program will depend on the highest level functional requirements we want to satisfy and the number of layers of decomposition necessary to come up with the complete design solution. A good designer will minimize both the number of upper level functional requirements and the number of layers of hierarchical decomposition; therefore, if we define a large system based on the number of physical components or the size of software, it will be difficult to provide a true measure or metric that separates the size effects due to the inherent nature of the design task from those that result from inefficient design solutions. In other words, by this definition, anything can be made into a large system if we are very poor designers.

Instead of being defined by the total number of components, a large system has the following char-
acteristics: a large number of functional requirements at the highest level of specification or at the problem definition stage, and not all of these functional requirements must be satisfied at the same time. Often, a subset of these functional requirements must be satisfied at any given time, and the elements of this subset change as a function of time. This process repeats throughout the life of the system. For example, New England Telephone serves many customers, but the particular set of customers who must be linked together changes as a function of time; therefore, the system is dynamic and must be reconfigurable on demand. Similarly, the Boeing 747 airplane must be able to deal with different requirements throughout its life. On the other hand, many machine tools are static. They are designed to satisfy a fixed set of functional requirements at the highest level, which do not change as a function of time. For example, an extruder is designed to process polyethylene or polyvinyl chloride. It is not designed to change its configuration to process any thermoplastics at any time over its life. In this sense, many machine tools are small systems.

Whether there is a large number of functional requirements that must be satisfied or the subset of the functional requirements that must be satisfied changes as a function of time, the system must be designed such that the independence axiom is always satisfied. When this is done incorrectly, the system is difficult to operate due to the coupling of functional requirements by the design parameters chosen. Such a coupled system often cannot satisfy all of its functional requirements within their specified tolerances. The information content of a decoupled system may be large relative to an uncoupled system because decoupling introduces conditional probabilities. Many conventionally designed large systems tend to be complex due to poor design and, consequently, require a large information input to operate properly. When coupling is introduced in the design, the original goals cannot be satisfied by the design, and many undesirable compromises are made in functional requirements.

In view of the foregoing discussion, a large system may be defined as follows:

A system is a large system if the total number of the highest level FRs that the system must satisfy during its lifetime is large and if at different times the system is required to satisfy many different subsets of FRs.

According to this definition, a large building or even a large castle is not a large system, although they may have many physical components and many rooms, because their functional requirements remain more or less constant throughout their lifetime. An intelligent skyscraper or a hotel with a thousand beds, however, may be a large system according to the above definition, if the set of functional requirements it must satisfy changes continuously and unpredictably as a function of time due to changes in customer requirements.

Axiomatic Approach to the Design of a Large System

Suppose that we have to design a system that has to satisfy n FRs. As we search for design parameters in the physical domain that will enable us to satisfy the functional requirements, we may find that there is more than one DP, that can satisfy a given FR. This fact can be expressed as follows:

\[
\begin{align*}
\text{FR}_1 & \, \sim \, (D_{P1}^1, D_{P1}^2, \ldots, D_{P1}^r) \\
\text{FR}_2 & \, \sim \, (D_{P2}^1, D_{P2}^2, \ldots, D_{P2}^r) \\
\text{FR}_3 & \, \sim \, (D_{P3}^1, D_{P3}^2, \ldots, D_{P3}^r) \\
& \vdots \\
\text{FR}_n & \, \sim \, (D_{Pn}^1, D_{Pn}^2, \ldots, D_{Pn}^r)
\end{align*}
\]

(6)

The relations in Eq. (6) simply state that FR1 can be satisfied (indicated by \(\sim\)) by DP1, DP1, DP1, and so on. Similarly, FRn is satisfied by DPn, DPn, DPn, and so on. Eq. (6) does not say which DP3, for example, is the best solution for FR3. Furthermore, because all or a subset of the functional requirements must be satisfied at any given instant, one cannot say a priori which DP3 is the best solution without considering the other functional requirements that must be satisfied at the same time; that is, the choice of DP3 may be different depending on the chosen subset of functional requirements.

Eq. (6) represents the knowledge base or database for the large system. When additional design parameters are added to these equations, it is equivalent to expanding the knowledge base or the database. The richness of the system is defined by the size and quality of the database because the larger the num-
ber of available design parameters the greater the possibility for better designs.

As defined earlier, a large system is often required to satisfy a subset of its total set of functional requirements at any given time. Suppose the subsets of functional requirements change as a function of time as follows:

\[
\begin{align*}
@ t = 0, \quad & \text{the subsets are} \\
\{\text{FR}s\}_0 = \{\text{FR}1, \text{FR}5, \text{FR}7, \text{FR}n\} \\
@ t = T_1, \quad & \{\text{FR}s\}_1 = \{\text{FR}3, \text{FR}5, \text{FR}8, \text{FR}m\} \\
@ t = T_2, \quad & \{\text{FR}s\}_2 = \{\text{FR}3, \text{FR}9, \text{FR}10, \text{FR}n\}
\end{align*}
\]

Eq. (7) is written in vector notation form. For example, the equation \(\{\text{FR}s\}_0 = \{\text{FR}1, \text{FR}5, \text{FR}7, \text{FR}n\}\) is read as “the vector of FRs at time \(t = 0\) is composed of elements FR1, FR5, FR7, and FRn.”

To satisfy \(\{\text{FR}s\}_0\), the design parameters we choose—for example, \(\{\text{DP}s\}_0 = \{\text{DP}1, \text{DP}5, \text{DP}7, \text{DP}n\}\)—must be such that the independence of FR1, FR5, FR7, and FRn is assured according to the independence axiom. Once these design parameters are chosen, the large system operates just like any other system; however, at \(t = T_1\) a different subset of functional requirements must be satisfied. This means that the system must reconfigure or switch to satisfy \(\{\text{FR}s\}_1 = \{\text{FR}3, \text{FR}5, \text{FR}8, \text{FR}m\}\) independently. Similarly, at \(t = T_2\) the system must again reconfigure to satisfy \(\{\text{FR}s\}_2 = \{\text{FR}3, \text{FR}9, \text{FR}10, \text{FR}n\}\).

The search process for design parameters is straightforward, in principle. Consider the case of choosing design parameters to satisfy the specified functional requirements at \(t = T_2\), \(\{\text{FR}s\}_2\). When we are searching for an uncoupled design, we can choose DP3 that affects only FR3 and has no effect on any of the other functional requirements in the subset. Then we choose DP9 that only affects FR9. This process is continued until the entire subset of functional requirements is satisfied. If an uncoupled design cannot be developed because of the lack of proper design parameters in the database, then we must seek a decoupled design. In this case, we can choose any DP3 first to satisfy FR3. Then, we can look for a DP9 that does not affect FR3, yet DP9 may affect all other functional requirements. Similarly, we can choose DP10 that does not affect FR3 and FR9. Finally, when we are looking for DPn that can satisfy FRn, DPn cannot affect FR3, FR9, and FR10. A systematic way of choosing a correct set of design parameters was presented in earlier publications.

The switching mechanism to go from a given subset of design parameters to another must operate at an acceptable speed. In the case of the telephone system, the switching rate has become ever faster over the years since the invention of the telephone. In the case of the U.S. government, the time constant to change and satisfy the subset of functional requirements is at least six months, as indicated by the time it takes for a new administration to appoint subcabinet-level government officials. In the case of automotive companies, it used to take at least six months to change over the manufacturing line when new car models were introduced. Now, because of global competition, changeover time has been shortened considerably.

For a given subset of functional requirements, there may be many different sets of design parameters that are acceptable from the functional point of view. The best solution from among those proposed can be chosen based on an evaluation of each of the proposed solutions by measuring the information content. We can evaluate the information content for each and all functional requirements that comprise the subset and then sum them up to get the total information using Eq. (5).

An ideal large system is an uncoupled design with an infinite adaptability or flexibility. Infinite adaptability means that an acceptable set of design parameters can always be selected to satisfy the given subset of functional requirements. In this case, the database [that is, the knowledge base given by Eq. (6)] must be expandable because the system must be configured to adapt to a new situation by acquiring new data or knowledge. Otherwise, the system will have a limited usefulness and flexibility. Furthermore, a large system can function most effectively if any user can modify the database so that users can adapt it to a variety of applications.

To map from the physical domain to the process domain, we must establish a database similar to Eq. (6). Following the format and the system of notation used in Eq. (6), the database may be expressed as follows:

\[
\text{DP, } \{\text{PV}_r^a, \text{PV}_r^b, \ldots, \text{PV}_r^n\}
\]

The nature of the databases of plausible design parameters and process variables will be different.
depending on the hierarchical level of the functional requirements or design parameters. For example, the word \textit{vehicle} is at a higher level of abstraction than words such as \textit{bus} or \textit{car}. Similarly, the word \textit{extruder} is at a higher level of abstraction than the specification of a particular combination of \textit{screw} and \textit{barrel} that comprise the extruder. Thus, the lower level database in the decomposition hierarchy contains more specific and detailed information, whereas the highest level database tends to be more conceptual or abstract.

\textbf{On Designing the Best Large System}

The selection of a right set of functional requirements for given customer needs is the most important step in design. Equally important is the selection of the right design parameters and process variables at the highest level. Wrong decisions made at the highest level cannot be rectified at lower levels of the design hierarchy. Sometimes it is necessary to change the functional requirements until the correct design problem is formulated.

Given a set of functional requirements, the quality of the solutions that satisfy the independence axiom depends on the richness of the database available to the designer. Furthermore, the databases given in Eqs. (6) and (8) can be expanded by any designer; therefore, there is no unique or optimal solution from an absolute point of view.

When there are several equally acceptable solutions according to the independence axiom, we may be able to choose the best design among these solutions, if all the subsets of functional requirements that the system must satisfy during its life are known. In this case, the information content can be computed using Eq. (5). For example, if there are four possible designs for satisfying the three subsets of functional requirements specified in the example given by Eq. (7), we can compute the information content for each design. The design with the least information content among all four possible designs is the best solution, but only among the four proposed alternatives.

The difficulty arises in choosing the best solution when the subsets of functional requirements vary unpredictably over the lifetime of the system. In this case, the best design cannot be determined a priori; that is, the one design that requires the least amount of information for a given subset of functional requirements may not be the best for another subset. The strategy for designing a large system when the future needs cannot fully be predicted in terms of specific subsets of functional requirements is to provide the system with intelligence so that it can always choose the right sets of design parameters according to the independence axiom. To achieve this goal, the database must be expandable because the robustness of the system depends on the richness of the database.

Moreover, if it is possible to identify some of the subsets of functional requirements that the large system will encounter in its lifetime, then we may choose the design that is the best at least for these known subsets. Obviously, the probability of choosing the best design among all those possible increases as the number of subsets identified during the design stage approaches the complete set to be encountered during the actual operation.

Considering the foregoing reasoning, the following theorems may be stated:

\textbf{Theorem 1 (Importance of High-Level Decisions)}

The quality of design depends on the selection of FRs and the mapping from domains to domains. Wrong decisions made at the highest levels of design domains cannot be rectified through the lower level design decisions.

\textbf{Theorem 2 (The Best Design)}

The best design among the proposed designs for a large system that satisfies \( n \) FRs and the independence axiom can be chosen if the complete set of the subsets of FRs that the large system must satisfy over its life is known a priori.

\textbf{Theorem 3 (The Need for a Better Design)}

When the complete set of the subsets of FRs that a given large system must satisfy over its life is not known a priori, there is no guarantee that a specific design will always have the minimum information content for all possible subsets, and thus there is no guarantee that the same design is the best at \textit{all times}, even if there are designs that satisfy the FRs and the independence axiom.
Theorem 4 (Completeness)
The probability of choosing the best design for a large system increases as the known subsets of FRs that the system must satisfy approach the complete set that the system is likely to encounter during its life.

Theorem 5 (Infinite Adaptability Versus Completeness)
The large system with an infinite adaptability (or flexibility) may not represent the best design when the large system is used in a situation where the complete set of the subsets of FRs that the system must satisfy is known a priori.

Theorem 6 (Complexity)
A large system is not necessarily complex if it has a high probability of satisfying the FRs specified for the system.

Theorem 7 (Quality of Design)
The quality of design of a large system is determined by the quality of the database, the proper selection of FRs, and the mapping process.

A system consists of components or elements. The relationship between these physical components is sometimes called form. For a system to function, the interrelationship between the components and their integration is important; however, the specific interrelationship among these elements follows the functional requirements the system must satisfy. The simplest relationship among these elements is given when the design is uncoupled. When the design is coupled, a system with many components cannot function.

Implications on Organizational Design
As shown in Table 1, organizations such as governments or a large company can be designed or analyzed based on axiomatic design. In organizational design, the process variables of the process domain are resources, both human and financial. One of the problems in organizational design is how to provide the flexibility and ability to respond quickly when new functional requirements are imposed on the organization without the expenditure of additional resources. An organization cannot keep extra people for unforeseen events because of the cost and unpredictability of future needs; therefore, in the case of organizational design, we cannot create a rich database [that is, Eq. (8)] because of the cost. One cannot have a redundant system where a large number of design parameters (that is, lower level organizational units) are added in the database to satisfy a given functional requirement in anticipation of unforeseen situations, and conversely, a large number of process variables (that is, people) cannot be kept in an organization for unpredictable future needs.

When the database for a large organization is insufficient, we may or may not have acceptable design parameters and process variables when a new set of functional requirements must be satisfied. To respond to the ever-changing set of functional requirements with fixed resources, a possible solution is to have a means of quickly reorganizing the company (that is, change design parameters) to satisfy a new set of functional requirements and, if necessary, to hire the required personnel (process variables) who can perform tasks that can satisfy the design parameters and, ultimately, the new set of functional requirements. Such a policy also necessitates the elimination or retraining of people who cannot fulfill the new tasks. From a societal point of view, this may not be a desirable policy, but it is practiced frequently. Lifetime employment is possible only when either the functional requirements do not change or when the organization expands through internal growth, creating new jobs for those not needed in their old jobs or when employees have a variety of skills and commitment to lifelong education. When an organization depends on the ability and ingenuity of a fixed set of people to meet a new set of functional requirements, the organization may fail due to coupling of the functional requirements. This can be stated in the following theorem:

Theorem 8 (Design of Organizations)
In designing large organizations with finite resources, the most efficient organizational design is the one that specifically allows reconfiguration by changing the organizational structure and by having a flexible personnel policy when a new set of FRs must be satisfied.
It should be noted that in axiomatic design, zigzagging between the domains is required to create the hierarchies of functional requirements, design parameters, and process variables. When this is done for organizational design, the desirable organizational structure at a given level of the hierarchy is that which consists of either “functional subunits” or “product subunits,” but never a mixture of functional and product subunits at the same level of the hierarchy because the decomposition of the functional requirements must be done such that the decomposed functional requirements are independent of each other. When functional subunits are mixed with product subunits, it is most likely that their functional requirements will overlap and will not be independent. Consequently, any attempt to come up with an optimum design for the entire system may create a coupled design or a resource intensive design. This observation may be stated in the following theorem:

Theorem 9 (The Need for Modularity)
When a large system (for example, an organization) consists of several subunits, each unit must satisfy independent subsets of FRs so as to eliminate the possibility of creating a resource-intensive system or coupled design for the entire system.

Implications on the Design of an Intelligent Manufacturing System
In recent years, the design of an intelligent manufacturing system (IMS) has gained worldwide interest. Although the term intelligent in IMSs has not been defined by the manufacturing community, it must refer to the ability of the manufacturing system to reconfigure the production system, including machines, purchasing, inventory control, and factory layout, to respond to changing market demand for various products. The question is: “How do we design such a system?” To have intelligence, the system must have the ability to make correct decisions based on rules or principles. Axiomatic design may provide such a tool, and the purpose of this discussion is to show how such a manufacturing system may be designed.

Suppose the goal of a manufacturing system is to produce five products (that is, DP_A, DP_B, DP_C, DP_D, and DP_E) at the lowest possible cost. In addition, assume that at \( t = T_1 \) we have to produce 100 parts of DP_A and 1800 parts of DP_B; at \( t = T_2 \) we have to produce 300 parts of DP_C, 450 parts of DP_D, and 100 parts of DP_E; and at \( t = T_3 \) we have to produce 400 parts of DP_A and 500 parts of DP_D. Thus, the demand for these products is not the same and changes as a function of time. Furthermore, the demand for each product cannot be predicted a priori. To manufacture the products at the lowest possible cost, the manufacturing output must be maximized (that is, full-capacity production of reliable products), while the system input (such as capital investment, inventory, and labor content) must be minimized. For simplicity, labor content will be ignored in this discussion.

In typical manufacturing operations, the components for these products can be manufactured using several different kinds of machines in a preexisting manufacturing facility. These machines will be designated by \( \alpha, \beta, \chi, \delta, \epsilon, \) and \( \phi \). We will consider the design problem only at the highest level of the hierarchy without undertaking the decomposition process. In this case, the database may be represented as follows:

\[
\begin{align*}
DP_A & \subseteq (PV^\alpha, PV^\beta, PV^\chi, PV^\delta, PV^\epsilon, PV^\phi) \\
DP_B & \subseteq (PV^\alpha, PV^\beta, PV^\chi, PV^\delta, PV^\epsilon, PV^\phi) \\
DP_C & \subseteq (PV^\alpha, PV^\beta, PV^\chi, PV^\delta, PV^\epsilon, PV^\phi) \\
DP_D & \subseteq (PV^\alpha, PV^\beta, PV^\chi, PV^\delta, PV^\epsilon, PV^\phi) \\
DP_E & \subseteq (PV^\alpha, PV^\beta, PV^\chi, PV^\delta, PV^\epsilon, PV^\phi)
\end{align*}
\]

Eq. (9) states that the five products represented by \( DP_i \) can be made by any one of the six processes indicated by \( PV_i \). To ensure the manufacture of reliable products, we have to choose process variables such that the specific design parameters that must be satisfied at any given time will remain independent of one another. Suppose the opposite case where one tries to use one of the processes, say \( PV_i \), for all design parameters. Then, the design parameters must be made sequentially while other machines are standing idle. Then, we cannot utilize the full capacity of the machines.

When there is more than one set of process variables that can maintain the independence of design parameters, we have to choose the best from among the possible solutions by means of the information content. In manufacturing, the tolerance requirements for the parts and the capability of machine tools to pro-
duce parts within the required tolerance have significant effects on the ultimate choice of process variables because the functional independence and the information contents are affected by the tolerances. Given a subset of design parameters that we must produce, we can choose appropriate process variables so that the independence axiom is not violated. Then, the information contents can be computed by comparing the design range with the system range for each DPi. It can be seen that as the number of design parameters and process variables increases, this task can best be performed using a computer algorithm.

Because it was assumed that the choice of machine tools is limited to those available in the company, there is no assurance that the system configuration arrived at through the above design process is the best that would be used if the designer were free to choose any state-of-the-art machine tools; however, this limitation is often the reality because frequent investments of new capital equipment cannot be justified. Indeed, to justify the capital investment, it must also be included in the analysis. For example, the capital cost can be included as one of the design parameters or treated as a constraint.

So far the discussion has considered only the production capability of each machine tool and suggested a means for dealing with capital investment, but it has not included the questions of capacity utilization and inventory control. Production rate is obviously an important consideration that is related to these two different points of view. If DPa and DPb are, for example, two independent products, we will want to maximize their production rates, whereas if these parts are ultimately to be assembled together in a single product, we will want to match their production rates to minimize the inventory (such as by either slowing one of the machines or adding another machine). How can we deal with this problem?

One way is to specify the rate of production as additional design parameters and evaluate the information content; that is, the production rate of the specific process variables can be compared against the design range. For example, at \( t = T_1 \) we have to satisfy \( DP_a, DP_b, d(DP_a)/dt, \) and \( d(DP_b)/dt \), using the database of process variables specified in Eq. (9) and additional inputs involving \( d(PV_j)/dt \) terms, where the differential terms \( d(x)/dt \) indicate the rate of production. These additional terms will restrict the design choices further. This example is purposely made very simple to illustrate the point. Further refinements of the ideas presented in this paper are necessary for these ideas to be used in factories.

The Human Brain as a Large System and the Role of Creativity

The human brain and its associated cognitive processes are an example of a large system not only because there are a large number of neurons in the brain but because the system can and must respond to changing functional requirements during its lifetime. It also has the ability to self-organize and make its own design decisions. Its ability to create good design solutions depends on the database the brain contains, and the ability to synthesize good design solutions using an "algorithm" and by manipulating the database. How does a brain synthesize a design solution?

The brain accumulates its database through education and experience. In the absence of specific algorithmic instruction on decision-making techniques, the brain may derive a solution through association, analogy, extrapolation, interpolation, heuristics, and pattern recognition, some of which may be called inductive reasoning, while the rest may be called deductive reasoning; however, these processes are slow and fraught with possible errors. Furthermore, they are not systematic, requiring many iterations to complete the design. The question we should raise is: "Is there a better means of teaching the brain to be creative or produce better designs?" It may be interesting to examine this question in terms of the large systems design issues discussed in this paper.

Can the brain be taught to create new ideas, weed out bad ideas, and make correct design decisions based on the independence axiom? If we can teach a machine to be creative using the methodology and principles outlined in this paper, we may be able to teach young children to be more creative as well. Children's innate creative abilities may be enhanced by teaching the notion of the independence of functional requirements through carefully planned exercises and through the instruction of the information axiom by means of examples that show the importance of simplicity in synthesis and design. If this pedagogy is successful, it may be a way of proving
that creativity is not only a genetically given innate ability but also one that can be acquired.

At the Massachusetts Institute of Technology, we are attempting to demonstrate that we can teach a machine to be creative by developing a thinking design machine (TDM). The idea is to create a machine that can develop creative solutions based on the design axioms. When we examined the Invention Machine developed in Russia, we found that the types of features suggested by the machine were consistent with the design axioms; however, the underlying software incorporates many rules rather than relying on decision-making principles.

One significant difficulty in implementing the TDM is the conceptual leap required in assembling the selected design parameters into a systems solution. When human designers create design solutions, designers' knowledge and creativity play a major role during the mapping process from the functional domain to the physical domain as well as from the physical domain to the process domain and when they are integrated in a systems context. The ability to integrate the components into a systems solution distinguishes a creative and able designer from one who is not. Human brain power is so great that no human-made machine may ever mimic it, but it is an interesting intellectual exercise to investigate whether at least a small subset of brain functions can be imitated or improved.

To date, work on the TDM has focused on developing a software shell to aid designers of large systems. It is an interactive system for designers that can store the database, suggest all acceptable designs based on the database, analyze proposed solutions for independence, and evaluate the information content. The shell can be used for the design of products, processes, systems, software, and organizations.

Conclusions

This paper provides a theoretical framework for the design and operation of large systems. The framework consists of a working definition for large systems and specific theorems to aid decision-making. These theorems have been derived from the design axioms. Furthermore, the implications of these theorems on the design of organizations, intelligent manufacturing systems, and the role of the human brain in fostering creativity are discussed.

One of the ultimate goals of our research is to develop a thinking design machine. On the basis of the understanding we gain from our research efforts, we hope to create pedagogical tools for teaching young people to be more creative. If we accomplish these goals, we would have closed the loop in our understanding of creativity—going from design theory to cognitive science. We may then eliminate some of the myths and mysteries associated with creative human activities.

Acknowledgment

The author wishes to thank Professor Leonard Albano, Dean Joel Moses, and Derrick Tate for interesting discussions. Part of this paper was given at an invited lecture at the Intelligent Manufacturing Systems Conference in Tokyo in 1990.

References

7. Invention Machine (Cambridge, MA: Invention Machine Corp.).

Author's Biography

Nam P. Suh, the Ralph E. & Eloise E. Cross Professor of Manufacturing in the Department of Mechanical Engineering at Massachusetts Institute of Technology, joined the faculty in 1970. He is director of the MIT Manufacturing Institute and head of the mechanical engineering department. He founded the MIT-Industry Polymer Processing Program in 1973 and served as its director until 1984. He was the founding director of MIT’s Laboratory for Manufacturing and Productivity, which was established in 1977. From October 1984 to January 1988, he served as Assistant Director for Engineering at the National Science Foundation (NSF). His main objective at NSF was to implement a conceptual framework for stronger U.S. industrial competitiveness through improved engineering education and an infrastructure for research.

Dr. Suh received his SB and SM degrees from MIT and his PhD from Carnegie-Mellon University. He is the recipient of numerous honors and academic awards, and his professional interests include tribology, polymer processing, metal processing, manufacturing, and design.