Software Creation: A Study on the Inside of Human Design Knowledge

Hassan ABOLHASSANI†, Student Member, Hui CHEN†, Behrouz Homayoun FAR†, and Zenya KOONO†, Members

SUMMARY This paper discusses the characteristics of human design knowledge. By studying a number of actual human made designs of excellent designers, the most frequent basic mental operations of a typical human designer have been found. They are: a design rule for hierarchical detailing reported previously, a micro design rule for generating a hierarchical expansion, dictionary operations to build a micro design rule and dictionaries. This study assumes a multiplicity of knowledge based on Zipf’s theory, “the principle of least effort.” Zipf’s principle may be proved and it becomes possible to understand the fundamental nature of human design.

key words: knowledge based software engineering, human design knowledge

1. Introduction

The software industry has been facing difficulties in responding to the strong need for software products. Among many solutions, various kinds of automatic design have been proposed. However, their use is limited. Due to the difficulties of ascertaining the characteristics of human designs, most of the applications are still focusing on the last stage of the design-coding [12].

The Software Creation Project has been conducting bottom-up research aimed at establishing a rational and scientific basis for enabling automatic design. The approach aims to determine the characteristics of human design skill. For the tool, a kind of “Acquisition by Modeling” has been developed [10], and it was used for systematic reconstruction of human design knowledge [4].

The final goal of this study is developing a tool with some kind of knowledge-based operations during the process of human design. Preceding research [5] on the simplest case of reusing past designs showed that a substantial part of a design may be performed by reusing past hierarchical expansions.

The motive for this study is to “Create a Hierarchical Detailing.” Considering the final goal, an intensive analysis was done. The results of the study help to establish a knowledge model for design, which is necessary for systematic acquisition of knowledge from an expert. The knowledge model and acquired knowledge is explained in this paper. The study covers detailed design, but the results may also be applicable to higher levels.

Once the knowledge model is established, the expert system or the automatic design system may be reconstructed systematically with ease. In view of this, the paper reports on the analysis phase or studies up to the detailed design’s knowledge structure.

From the Software Engineering viewpoint, this is a more detailed study of some general-purpose design methods. From the Knowledge Engineering viewpoint, it is a study to establish the knowledge structure model for the basic mental operations of a human designer.

Section 2, the basis of this study, summarizes previous work on automatic design by reusing past designs. It also gives a perspective on the need for creation ability during hierarchical detailing. Section 3 describes knowledge inside of design rules and introduces the procedure leading to the detailed structure.

Section 4, micro design rules, discusses characteristics of knowledge structures, micro design rules and dictionaries. Section 5, macros, introduces a form of knowledge structure between hierarchical detailing and pure knowledge micro design rules.

Section 6 introduces the entire knowledge structure thus found. Section 7 is the conclusion.

2. Basis of This Study

First, the proceeding research which provide the basis for this study are explained.

In order to purify the design process, a high maturity software organization is taken as an expert model. Since designers in such an organization have accumulated various improvements on their process or design procedure (including natural language), their process is highly standardized, or a solid hierarchical work process appears. Such solid work process is taken as the knowledge model.

The left part of Fig. 1 shows the hierarchical work process, and the right side shows a corresponding knowledge structure, systematically reproduced from the original work process [4], [10]. A design may be regarded as a hierarchical expansion chain from the initial specification to the source code, where each link is a hierarchical decomposition of human concept. In a highly

---

†The authors are with the Department of Information and Computer Sciences, Faculty of Engineering, Saitama University, Urawa-shi, 338–8570 Japan.
matured organization, each link or each small step (micro work process) of a design is well documented. This expansion is shown in the bottom left of the figure, and the elementary detailing is called a design rule.

In order to accumulate all the research in the project, a bottom up approach has been taken. The preceding study [5] assumed that all of a design is constructed automatically via the memory of past experiences. This is the model of a novice designer. As the person experiences more designs, accumulating design rules, a hierarchical decomposition is made more automatically, reusing design rules.

The degree of automatic detailing has been evaluated quantitatively, as shown by the bold curve in Fig. 2 [5] using the Learning Effect [8]. The abscissa shows the number of experiences of designs. The ordinate shows the degree of automatic hierarchical detailing. The bold line is the case of reuse only [5]. Though it starts from zero, it shows that the majority of detailing may be performed by reuse, where the number of experiences is large.

Thus, the important part of the inside of the design has been revealed by [5]. However, several problems still remain. The main problem is to find the knowledge of creation of new hierarchical detailing. This may be said to be the ability of an expert designer. The next
stage of the study is to find solutions for this creation problem.

The authors’ approach is shown in Fig. 2 by the dotted curves. To upgrade the degree of automatic detailing, there must be several levels of creation abilities, for example from novice, to moderate and then expert designers. Thus in enhancing design knowledge, the degree of automatic detailing advances. Each dotted curve in the figure shows the effect of adding one creation level to a previously attained level. Therefore this study is a trial to go beyond the bold curve in the figure.

“Acquisition by Modeling” [7] has been applied to resolve problems associated with knowledge acquisition from experts in the project. The approach is to construct a knowledge model, and then acquire knowledge systematically according to the model. This approach is taken in this study, and this paper clearly shows a mechanism for establishing the knowledge model as well as the knowledge.

3. Knowledge Inside Design Rules

3.1 Basic Assumptions

The aforementioned problem may not be solved by Knowledge Engineering at present. The project’s policy is “to learn from a human designer,” who designs more automatically as the level of expertise advances. Principles which explain human design have been sought.

Zipf’s theory, “the principle of least effort” [14], is well accepted in Ergonomics. It states that: to solve a problem, a human tries to use the simplest way first, which needs least effort, and if it fails the person tries to find another more complex way. This is repeated until the problem is solved finally. What Zipf states is apparently the existence of multiplicity of (design) knowledge.

Figure 2 shows effect of this theory for the design. The bold curve shows reuse of past experiences, which represents the simplest way in a design. Dotted curves show the model of more expert designers, which build a design in more complex ways. With more ability to solve a problem, the curve moves upward.

The qualitative aspect of this may be explained by Rasmussen’s theory [13]. He categorized human operational knowledge in three levels; “Skill-based,” “Rule-based” and “Knowledge-based,” where skill-based is the simplest kind in the sense that it needs least thought and is for frequent routine works resulting in quick response. Knowledge based is the most complex behavior, which entails more complex thinking processes and represents less occurring behavior, resulting in slower operation. This theory is a view from a different stand point on the same fact, “multiplicity of knowledge.”

The bottom right side of Fig. 1 shows the relationship between these theories and design rules. As shown, a multiplicity of design knowledge (Zipf’s theory) is depicted by a multiplicity of knowledge units inside level 112. When a designer is asked for a design rule, (s)he first tries to use past experiences (skill-based unit). If unable to find any, an attempt to find a simple rule for creating the design rule is made (rule-based unit). If still unsuccessful, another more complex rule is tried. In an extreme case, (s)he may use extensive knowledge to create the design rule (knowledge-based unit). Boxes (units) in the figure are labels of the classes. These kinds of behavior will be clarified by this study.

3.2 Approach for Analysis

As with preceding research [4], [5] in the project, a highly matured software organization with a hierarchical work process is assumed. In it, an expert designer creates a hierarchical detailing and records the design hierarchically in documents. Each small step in such documents is a design rule, as shown in the left side of Fig. 1.

Since the knowledge inside design rules is not included in the design documents, a digging down study has been made in this research [1]. Keeping the hierarchical nature as the basis, a researcher estimates the inside of creating a design rule. The bottom left part of Fig. 1 shows the idea [2], [3]. As a design consists of “cognition,” “decision” and “transformation,” the hierarchical detailing is decomposed to three fractional processes. Input is the parent concept, while output is the hierarchically detailed concept. Hereafter, the estimation of the inside is repeated. As the estimation deepers, several fundamental human mental operations become clear. After that, care has been taken to utilize them to result in several major types of mechanisms.

3.3 An Example of Analysis of a Design

Figure 3 shows an example design in four hierarchical levels. Each level of design is represented by a design document and each hierarchical decomposition of a function into the next level is a design rule. Natural language phrases are used to represent function and data symbols. Lower than level four are source code phrases, not shown in the figure.

Hierarchical decomposition of the “clock” function in level 1 to three serial functions in level 2 is a typical application of Myers’ Source-Transform-Sink division (STS)† [11]. This is a general method for function decomposition in a cascaded way, applicable in a wide range of applications. Section 3.4 shows the results of studying this method.

Another type of design rule in the example is based on the fundamental principle of Jackson’s Structured Programming method (JSP)†† [9]. Detailing of Transform to hands function of the level two to sub-functions
in the level three, and also detailing of *Transform to minute hand* function of the level three to sub-functions in the level four in the figure, show applications of JSP. It is also a general method for function decomposition in parallel way, which can be applied to many applications. This is described in Sect. 3.5.

### 3.4 STS Design Rule

The top of Fig. 4 shows a STS design rule, and the first step estimation of it is shown at the bottom. Cognition is a function with assigned input and output symbols. Decision is to apply STS division to divide hierarchically the function to three functions of Source, Transform and Sink. Transformation is shown in two stages. In the level 1-1, intermediate data points, specified as I-MAP and O-MAP in the figure, are found.

STS division is a method for structured design proposed by Myers in the context of Data Flow Design to decompose a function. The major steps of this decomposition method is: 1) Identifying major data flow, 2) Finding two points in data flow where data concepts there are most far from input and output concepts and exist in the most abstract forms, 3) Inserting three functions between: a) Input and it’s Most abstract form, b) Most abstract input and Most abstract output and c) Most abstract output and Output. Myers combined this with module design and named it composite design. STS division, however, is a universal principle applicable to any functional decomposition.

JSP division is another method for structured design proposed by Jackson in the context of design by data structures. The idea is that the correspondence of input and output hierarchical data structures determines the decomposition of a function to more detailed sub functions, resulting in a hierarchical decomposition of a function.

In the level 1-2, function symbols, in between two successive data symbols, are found, which complete the detailing. This estimation process was continued to find more inside detail of each intermediate step shown here.

By repeating these estimations, frequent human
mental operations, estimated to be used inside the Level 1-1, have been found. (In “Acquisition by modeling,” the researchers’ arbitrary decision is inevitable to some extent. Before reaching it, various mechanisms were proposed and evaluated quantitatively. For frequent operations, simpler mechanisms are preferred.) Figure 5 shows the results for finding I-MAP and O-MAP candidates, which were estimated to be fundamental steps inside every STS application. It is estimated that a human designer must form concept chains and abstractions starting from input and output concepts, respectively. Part (a) of the figure shows that for finding I-MAP candidates a human designer, having dictionary definitions of the concepts, forms a concept definition chain starting from input concept, 1-sec-clock. Starting from 1-sec-clock, a chain of concepts containing second, minute, hour and the like may be created by referring to a dictionary (uses of the dictionary is shown in rounded boxes in the figure).

Another basic operation is abstraction. One example is abstracting hour, minute and second as Time. This kind of operation is also formed by referring to a dictionary.

Part (b) in the figure shows a similar mechanism starting from the output symbol in a reverse direction. In this case also, forming (reverse) concept chains and abstraction are fundamental operations.

3.5 JSP Design Rule

Detailing of ‘Obtain hands’ (in the Fig. 3) from the level two to three functions in the level three is a kind of JSP design rule.

Like the STS design rule, similar estimations were recorded. Figure 6, shows the estimation in the order of elementary detailed design information. Three types of operations are shown in the figure. One is extending a concept to lower level concepts, which is shown from (a) to (b). It is applied five times in the example, specified by numbers in the figure. A human designer does this kind of operation using types of dictionary definitions.
which imply Part-Whole relationship.

Another kind of operation, shown in part (c) of the figure, is the creation of concept definition chains, which builds correspondences such as the Hour to Hour-hand.angle, Hour-hand.width and Hour-hand.length. This is like the construction of concept chains discussed in the case of the STS design rule. Dictionary type definitions are also used during this process.

The third type, shown in part (d), draws an upward correspondence to higher level concepts. For example, knowing the correspondence of Hour to components of Hour-hand, it is seen that Hour corresponds to Hour-Hand. Using these kind of operations, the correspondence structure of Time and Hands is created.

4. Micro Design Rules

From the aforementioned discussions, four types of basic human mental operations, which frequently appear during design, can be understood:

1. Building Concept Definition Chains (CDC).
2. Extending a Concept to its components (EXC).
3. Creating Abstraction for a set of concepts (CAB).
4. Upward Correspondence Structure to higher level concepts (UCS).

Such basic operations are called Micro Design Rules in this study. As discussed before, the necessary information for these kinds of operations are dictionary definitions. Thus dictionaries reside under Micro Design Rules.

These estimations have been applied to a number of real design samples. Then each elementary operation implied by the lowest levels of transformation steps in estimations were considered. Among them the more frequent patterns have been considered as micro design rules. Figure 7 lists them.

The first micro design rule named CDC, (Creates a Concept Definition Chain) starts from a given concept. Each concept in the chain corresponds to a definition in the dictionary related to previous concept in the chain. In the previous section, several examples of this rule were shown in the STS and JSP design rules.

The next is called CAB, as its function is to Create Abstraction from specified concepts. Applications of the rule have shown to create the most abstracted points, I-MAP and O-MAP.

The EXC micro design rule, Extends a Concept to lower level concepts or as described before builds a hier-
4.1 Dictionary Definitions

As mentioned in the previous section, each micro design rule needs some kind of dictionary definition. It has been found that the most frequent types of definitions used are:

1. Word category, specifically the noun and verbs are the main concern. Nouns represent data symbols and verbs are the main parts of function symbols.

2. Definition relationship: Relationships imply definition of a noun based on other nouns. For example the relationship between Minute and Second in the clock program example.

3. Part-Whole relationships: For extending a symbol to lower level concepts or the abstraction of a number of symbols to a single concept, this kind of relationship is necessary.

4. Synonym relationship: A concept may be represented by several words and it is necessary to understand such kind of similarity this relationship is necessary. For example, “buyer” and “customer” represents the same concept in a customer service application.

5. Macros

During the study it was found that some sequences of micro design rules frequently appear in a number of design rules. Such sequences were named Macro in this study.

The left side of Fig. 9 shows a macro example in the clock program. By referring to a dictionary, it is recognized that 1-sec clock and analog clock are related to some other concepts. This cognition activates CDC to build a concept definition chain from 1-sec clock concept, to find second, minute and hour, and then use CAB to abstract them as time. Then, use of CDC starting from analog clock results in concepts like hour hand, minute hand and second hand. By using CAB they are abstracted as the hands concept. Then an-
other macro, draw correspondence, is activated to establish correspondence between hour and hour hand, minute and minute hand, second and second hand, and finally between time and hands. Then SIO micro design rule selects time and hands as I-MAP and O-MAP concepts, respectively.

This macro is abstracted in a general form, as shown on the right side of Fig. 9. The cognition is that there are some dictionary definitions implying relationships between some concepts and the input concept, as well as relationships between some concepts and the output concept. The decision is to activate CDC and CAB micro design rules for input side and output side concepts, and then to activate another macro to draw correspondence of input side and output side concepts, and finally to activate SIO micro design rule to select I-MAP and O-MAP concepts. This is a frequent pattern for selection of MAP concepts inside many STS design rules.

Another frequent macro, which is referred to inside the above macro, is Draw Correspondence. Its application appears in most STS and JSP cases. It can be regarded as a series of applications of EXC for extending the input side concept and the output side concept to their lower level concepts. Then a series of applications of CDC for finding correspondence of lower level concepts, and then a series of applications of UCS to elevate correspondence to higher level concepts, are executed.

6. The Mechanism Underlay in a Design

The left side of Fig. 10 shows the fine structure of the design knowledge thus derived. It shows a hierarchical structure starting from a design and ending on a dictionary at the bottom. A design is hierarchically decomposed to a number of design rules, which may be already stored in memory. In a case where a design rule is not in memory, a process starts to create a design rule, and some macros may be activated. Likewise, a macro may be decomposed to applications of a number of micro design rules. Each micro design rule uses some kind of dictionary operation. And finally a dictionary provides the basic knowledge.

To clarify the structure, for each level of the structure in the left side of the figure, a corresponding example is shown in the right side. As a design rule, STS design rule of the clock program example is shown. As a macro inside this rule, draw correspondence is given. To select time and hands as I-MAP and O-MAP concepts, respectively, it is necessary to find whether it is possible to obtain hands from time. This is what draw correspondence macro is supposed to do.

As a step inside this macro, UPC is shown in the next level. Knowing the correspondence of concepts belonging to time and hands structures, the micro design rule may set time and hands as corresponding symbols. As a dictionary operation used by UPC, abstrac-
tion finder is shown. Having a set of concepts as hour, minute and second, it finds an abstraction like time from the dictionary. And finally at the lowest level a dictionary definition for the time concept is shown.

The structure corresponds to the multiplicity of Zipf’s theory postulated as the basis for the study. In a design, if a suitable design rule has been found, it is used at that instant, and this represents the simplest solution or reuse of past design. (It maybe regarded as a kind of dedicated dictionary.) If not, it goes down and a corresponding micro design rule is sought. On finding one, it works at the next instant. If not, it goes in another way. From this level, there are various possibilities.

During operation of micro design rules, repeated use of the lowest level dictionary is frequent. As a result, it becomes possible to perform a design in various ways. This explains the internal structure of human design knowledge as pointed out by Zipf.

It may also be discussed based on Rasmussen’s theory. For each level in the structure it may be possible to consider skill-based, rule-based and knowledge-based behavior. For instance at the design rule level, use of a past experience represent skill-based, while use of a predetermined algorithm such as macros corresponds to rule-based behavior. In an extreme case, where any predetermined algorithm is failed, micro design rules should be tried to create a new way for building the design rule. Apparently, this kind of behavior corresponds to knowledge-based behavior.

This also explains Learning Effect. If design becomes possible at a high level in Fig.10, the operation is fast, while it takes more time to operate at the lower levels. In the process of accumulating experiences, the number of layers, along with the components in a level, increase. As a result, the response becomes faster as more experience is gained.

The structure in Fig.10 represents a study of the last phase of design in this paper. But as every part of a work process has the same structure irrespectively of the position in the hierarchical work process, this result may also be applicable to any work process, even to the earliest and top level work process. Also, the authors believe that this structure may also be applicable to any human intentional activity.

Though much consideration have been given, the present analysis is based on estimations from outside as in the case of Cognitive Science. The authors recognize that there is a limitation. However, the structure of Fig.10 can explain various activities during design more clearly. The next problem is to clarify perceptions for identifying the way to a solution. It is necessary to continue intensive research toward revealing "what human design is."

The conversion from the most detailed natural language phrases to a source code is made in similar way.
as the hierarchical detailing of natural language concepts. But it must be made under the constraints of programming language grammar, which results in a different discussion. Since the main concern of the study is on natural language level, no further discussion about source code conversion is given in this paper.

Presently, the idea has been successfully applied in the design of an “Inventory Control System” and an experimental automatic software design system has been built successfully.

7. Conclusion

This paper describes a study on finding the knowledge inside software design. By studying frequent design patterns of detailed design phase of software, using a kind of “Acquisition by Modeling,” a fine structure of the design knowledge has been revealed.

It is a hierarchical structure of: design rule reported earlier, macros, micro design rules, and dictionaries with the associated operations.

The structure seems to be universal, and it will find many applications. One application is to extend it to the entire design knowledge structure. As it stands upon the most matured and well standardized experts knowledge structure, by deteriorating some part of the system, various kinds of poor or erroneous designs maybe explained. This will give rational and scientific basis for various “Software Engineering Practices,” which have been known on empirical basis.

Acknowledgment

The authors wishes to express their gratitude to those who have contributed to this study. Special thanks to Dr. Haruki Ueno (Professor, NACSIS) for his valuable comments and advice on this study.

References


Hassan Abolhassani has received his B.Sc. degree in Computer Software Engineering from Isfahan University, Iran 1990. He has received M.E. in Software Engineering from Sharif University of Technology, Iran 1993. He was graduated with honor of the master course and awarded by minister of education of Iran in 1993. From 1994 to 1997 he was a lecturer at the Computer Engineering Department of Isfahan University. He was successful to get the scholarship offered by ministry of higher education of Japan (Monbusho) to continue his study of the Ph.D. level. Since 1998 he is a Ph.D. candidate student in Information and Computer Science Department of Saitama University, Japan. His research interests include automatic software design, knowledge based software engineering, and methodologies for software development and management. He is a member of the Information Processing Society of Japan (IPSJ), and the Japan Society for Artificial Intelligence (JSAI).
Hui Chen received her B.Sc. degree in Computer Science from Shanghai Jiao Tong branch University, China in 1983. She received the M.E. and the Dr. Eng. degrees in information and Computer Science, both from Saitama University, Japan in 1995 and 1998 respectively. From 1983 to 1991, she was a lecturer at the Shanghai electricity and manufacturing technology school. Currently, she is a Research Associate in Department of Information and Computer Science, Saitama University, Japan. Her research interests include automatic software design, intelligent CASE tool and expert systems. She is a member of the Information Processing Society of Japan (IPSJ), and the Japan Society for Artificial Intelligence (JSAI).

Behrouz Homayoun Far received BSc. and MSc. degrees in Electronic Engineering in 1983 and 1986, respectively, from Tehran University, Iran. He has received his Ph.D. degree from Chiba University - Japan, in 1990. He is currently an Associate Professor at the Department of Information and Computer Sciences, Saitama University - Japan. The research fields of his interest are qualitative reasoning, automatic programming and distributed AI. Dr. Far was a secretary of SIG-KBSE/IEICE from 1995 to 1997. He is a member of the ACM, IEEE Computer Society, Japanese Society for Artificial Intelligence, IEICE and Information Processing Society of Japan.

Zenzai Koomo received B.E., M.Eng. and D.Eng. in 1959, 1961 and 1964 respectively all from The University of Tokyo. In 1964, he joined Hitachi, Ltd., and had been engaged in research, development and business management of electronic switching. His technical area has ranged from electronic circuits, I/O devices, processor from architecture and the implementation to the CAD, software/systems architecture, and development methods. In 1991, he transferred to Professor, Saitama University. He is a recipient of Inada young engineers award for 1963, and had served as an elected member of Board of Editors from 1983 to 1986. He is also a member of IPS Japan and JSAI.