REDISCOVERY OF PATTERN LANGUAGE FROM THE INFORMATION SYSTEMS VIEWPOINT

Kiminobu Kodama[†, ‡] **and Tadanori Mizuno**[‡]
[†] EXA Corporation,
580 Horikawa-cho, Saiwai-ku,
Kawasaki, Kanagana Pref. Japan.
[‡] Graduate School of Science and Technology,
Shizuoka University
3-5-1 Johoku, Hamamatsu 432-8011,
Shizuoka Pref., Japan.

ABSTRACT

Software engineering produced the significant results of design patterns, Pattern-Oriented Software Architecture, and so forth, by using the pattern language of architectonics, but it only focused on the aspect of the format of the pattern as know-how description. Focusing on another aspect of pattern language—"timeless way" as continuous activities—it seems that pattern language is an ideal mechanism to incorporate the requirements and the design constraints of information systems into itself as organization learning. In this paper, *the information systems cycle* is proposed for the building process of the enterprise information systems from such a viewpoint, and the meaning and the problem of incorporating the pattern language into the cycle are discussed.

Keywords: information systems cycle, pattern language, proto-requirements, urban planning approach

INTRODUCTION

In recent years, enterprise information systems (EIS) have become closely related to business strategy (MSSG, 2004). However, EIS still have many problems, such as the requirement of a great investment of money and time for the development, maintenance, and operation of systems. One serious problem is that the systems cannot change quickly in reaction to changes in the business environment; this could negatively affect the business base. The main reasons that systems development cannot conform to changes in the business environment are that the speed of change is too rapid for businesses to catch up and the cost of development is too high. Additionally, there are peripheral factors: the information systems manager might not recognize the problems and cannot change the information systems because no one in the organization possesses knowledge of the entire system.

A decrease in the planning power of EIS and a lack of flexibility in the organization have lead to a decrease in corporate power. Additionally, social influence is increased when trouble with information systems occurs and the ability to fulfill responsibilities is required. To solve the problem, conceptual data modeling and business process modeling projects have been attempted as methods for the information systems division to use to plan and manage their information systems using the guidance and education of project planning (Teshima, 2002). However, such a modeling does not generally produce satisfactory results. Even if the situation has already been taken into account, it is problematic to write only a superficial model, which can obstruct an essential discussion and a flexible conception of the business system.

Davis (2005) stated that it is often difficult for customers to write their requirements documents using any formal methods. Customers must not dedicate their time and their labor to such documents, but instead use their time to plan more essential business strategies. Based on these assumptions, this paper discusses the management and requirements for effective construction of information systems.

THE INFORMATION SYSTEMS CYCLE

The development of EIS can be compared to urban planning (Namba, 2005). In contemporary urban planning, only a few town blocks at a time are redeveloped gradually, accommodating residents' needs and making the best use of the existing infrastructure. Enterprise information system managers can adapt the concepts of architectonics for their own systems.

Learning from Architectonics

Zachman Framework
Zachman (1987) found a correspondence between the information systems process and the process of constructing a building. The building process advances through the architect's conversations with the owner and the builder. Zachman found similarities between the processes of the construction of a building and the production of information systems. He proposed that the construction of information systems should be based on a disciplined approach such as the construction and documentation processes.

The Owner, Designer, Contractor, and Architect
Vitruvius, an architect in the Roman era, described an architect as one who manages the trade-offs among the needs, design, and structure of a building. The needs are the owner's concerns, the design is the designer's concern, and the structure is the builder's concern. In the construction of information systems, there are also the roles of owner, designer, builder, and architect. Zachman (1987) found that the role of architect emerges from the structure of the other three roles and the processes among these roles. Zachman, however, considered the construction of only one building. The owner of EIS might be the organization, including the CIO and his/her assistants as the financial representatives of the enterprise, or might simply be a manager of accounting or manufacturing.

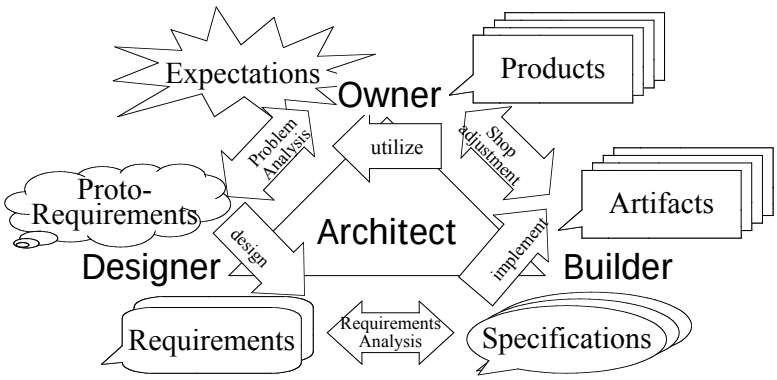

Figure 1. Information systems cycle model

Information Systems Cycle Model

The construction process of information systems that Zachman (1987) described can be redefined as a continuous cyclic process.

The relationship among the participants involved in the construction of information systems, including the owner, designer, builder, and architect and the products that flow among them, are identified in Figure 1. Raw materials are converted into end products in a stepwise manner by the series of actions of the participants as they play their own roles in a process called *the information systems cycle*.

Expectations, Proto-requirements, Requirements, Specifications, Artifacts, and Effects
The starting point of the information systems cycle is a declaration of *expectations* by the owner that establishes the targets of the process. The concrete states or the objectives by which the expectations are achieved are called *proto-requirements*. The expectations are translated into the proto-requirements through the interaction of the owner and the architect. The proto-requirements are converted into formal requirements by the designer, and the requirements must be approved by the owner. The *requirements* are translated into manufacturing s*pecifications* through the interaction of the designer and the builder. The builder converts the specifications into *artifacts* such as programs, data, or manuals, and after the artifacts pass inspection, they are delivered to the owner. At this time, some slight modifications called *shop adjustments* may be made and the owner accepts these as *products*. The changes made after delivery of the products are called *maintenance*. The owner obtains *effects* from the organization's use of them.

In general, EIS are constructed partially and gradually. Because it is rare that the expectations of the owner are satisfied within one cycle, cycles might be continuously repeated. Furthermore, the cycles may continue as long as the business continues to react to changes in the business environment.

The software process is considered here to be an internal process of the information systems cycle. This cycle is not a newly designed process but simply a reformation of the already known software processes. However, the software process can be clearly delineated by recognizing the components of the cycle, especially the proto-requirements, with which the owner can take the initiative in the cycle. The primary work of the owner and the architect in an information systems cycle is explained in the following sections.

The Declaration of Expectations by the Owner

The expectations are the declarations of the owner's vision. No designs should be included in the expectations. It is necessary for the owner to declare the expectations in his/her own words; accordingly, they are often described in the context of a story. Trust and motivation can be built among the team members when the story is narrated clearly.[1]

Analysis of Problems

The activity that leads to the proto-requirement stage is called *problem analysis*. Its purpose is to generate proto-requirements before the requirements are described. Problem analysis ensures the commitment of owners by presenting requests in language that they use. Owners thus recognize that they are the hosts of continuous activities. The architect works with the owner to define his or her proto-requirements.

The Positioning of Proto-Requirements

The proto-requirement stage falls between the stages of expectations and requirements. The content of the declarations varies according to the characteristics of the business and the developmental stages of its EIS. For instance, if the response time of the system influences the business, performance and safety requirements might be included in the proto-requirements. In an enterprise with insufficient information literacy, there might be proto-requirements that only paraphrase expectations. However, in such a case the architect should take care of the poor proto-requirements so as not to be confused in the following stages.

Proto-requirements include aspects of the entire EIS and parts of the individual business system. Although the proto-requirements are initially presented together, they are changed and differentiated as the information systems cycle progresses.

Early Testing of Proto-Requirements

It is necessary to test the proto-requirements immediately to confirm the existence of enterprise accommodation and commitment to them. Note that this is not a software test. Not only the specialist but also the owner and the stakeholders review the proto-requirements. Therefore, the presentation is carried out using comprehensible formats, such as user scenarios, demonstration videos, or storyboards. It is also useful to make the prototype a component of a core part of the proto-requirements.

Designing

The design work consists of converting the proto-requirements into the requirements for the solution plans. The requirements correspond to the drawings for both the artistic and structural designs of the building. The conversion work can be considered a designer's creation, with which it is unacceptable to interfere.

The requirements of EIS are rough sketches of the business system, consisting of the entire plan, the conceptual data model, the business process model, the conceptual use case, and state diagrams. These become input to the builder, although the designer may revise the requirements after receiving feedback on how to manufacture and implement the models more economically and quickly. The requirements are completed and tested immediately. The owner tests the requirements using general explanations and a demonstration of the systems provided by the designer. A more technical explanation is provided to the builder, who will estimate the manufacturing cost based on the requirements.

---

[1] A good example is the short story “Lightning of Senjogahara, Nikko” by Shinichiro Sakurai, the chief designer of Nissan Motors’ Skyline-R30 model. At the first design meeting for the new model, he narrated an image with a story of the car hurrying toward a lover waiting at a hotel in the rain, briefly illuminated by a flash of lightning.

REDISCOVERY OF PATTERN LANGUAGE AS ACTIVITIES

Pattern language is a concept of planning and designing towns and houses proposed in the 1970s by Christopher Alexander, a professor of architectonics. He recognized that towns or buildings that have a "quality without a name"[2] possess useful patterns handed down from past centuries but which are now forgotten. He thought that revealing and applying these patterns would aid in the successful planning of future towns. This idea had a great influence on software engineering in the late 1990s.

This idea yielded the concept of "*Design Patterns*" (Gamma, et al., 1995), which resulted in many pattern-catalogues. We owe much to the Hillside Group's Pattern Language of Programming (PLoP) initiative.[3] The pattern movement has contributed and continues to contribute to quality and productivity improvements in software production. However, the results only took into account the aspect of the pattern of the know-how description. The original pattern language is the set of activities called "*The Timeless Way of Building*" (Alexander, 1979).

Management of the Activities

Rules that provide the principles for the actions and criteria for the accomplishment of the project are formulated before starting each project. In "*The Oregon Experiment*"(Alexander, 1975), which was based on such rules, the "planning board" was organized to manage the construction project, and the board maintained the pattern language. The board consisted of the owner (university), the user (student), and the architect. None of the builders was included. In Alexander's other projects, there was no mention of having formally set up an organization such as a planning board, but both the owner and the users were involved in the activities.

How to Use the Pattern Language

The usage provided in the book "*A Pattern Language*" (Alexander, 1977) is now described. First, according to the theme of construction, one looks for and extracts the most appropriate pattern from among 253 patterns. Other patterns to which the pattern refers are extracted, if necessary. Next, only useful patterns are extracted from the following patterns. After any necessary changes and additions are made, the set of patterns is assumed to be "the pattern language" for the project. This is like a poem to be recited, Alexander said. Note that the order of the pattern language should be maintained during this procedure.

The example of the Eishin-Gakuen Higashino high school in Japan that he involved shows how the pattern language is spoken. In Figure 2, each underlined section is a pattern, and the pattern description corresponding to each section is made separately. The pattern language is spoken as if one were walking through the site, giving a vivid image of the school (not a schoolhouse, but a life) to be established in the future. The proto-requirements of information systems should be stated as they are in this example.

2-3. The Door Road
There is the door road toward the boundary from the front gate to the inside. The walls or the trees stand in a row on both sides of the door road, and it is very quiet.

2-6. The Important Center
Passing through the third gate, across the Central Plaza, there is the most important center of the high school and the university. Here, the place can be reached by passing through the many folds of the way, and there is quietness.

4-7. The Quiet Part of the Character Square-plus-Cross Center
The other half of the character Square-plus-Cross Center is more mysterious. Maybe it is the rear of the student hall that leads to the university corridor. It is possible to glance at the place from the entrance, or through the pillars of the arcade. But the place is so quiet that the high school students cannot go there. Whenever the high school students glance at the place, they yearn for the education that they'll receive there.

Figure 2. Example of proto-requirements in pattern language

[2] "Goodness which cannot be explained."
[3] http://hillside.net/

Each pattern is also a technological vocabulary that passes among the three people. If someone says "the door road," for instance, they all understand at once what that means. The pattern will be corrected if misunderstood.

The pattern language also contains the meaning of the design code, which restricts the designer's degree of freedom. Even though the degree of freedom is restricted, it is expected that the design efficiency will increase by the narrowing of the design space that the designer uses at the conversion. Because different designers will design similar models, the order of the landscape will be maintained.

THE CONCEPT OF INFORMATION SYSTEMS PATTERN LANGUAGE

Each plan based on the proto-requirements of the information systems cycle using the idea of the pattern language summarized below.

A Pattern Language for Information Systems

Proto-requirements are stated in pattern language that is easily understood by the stakeholders and enables early testing. The workflow when the pattern language is introduced into the information systems cycle is shown in Figure 3, where the process diagram is used (Eriksson & Penker, 2000).

The Timeless Way
The work of the organization that corresponds to the planning board is to write and maintain the master pattern language. It is shown in the cycle in the upper half of Figure 3. The construction planning of an individual business system is called a *program*. *Program management* determines the sequence and timing of the initiation of the programs. The evolution speed of EIS is controlled by this adjustment and the decision by the organization. The planning board includes a user's representative.

The Completion of Programs
Each project team is composed of an owner, a designer, a builder, and an architect. As shown in the lower half of Figure 3, three of these refer to the master pattern language differently. The owner refers to it as a dictionary of technical terms to state his or her proto-requirements, the designer refers to it as a design code to design the requirements, and the builder refers to it as a construction standard to produce the software products. Each one's work might progress concurrently as the expectations and requirements become clear through the statement of the proto-requirements spoken in the pattern language. The traceability from the expectations to the effects is ensured in the process.

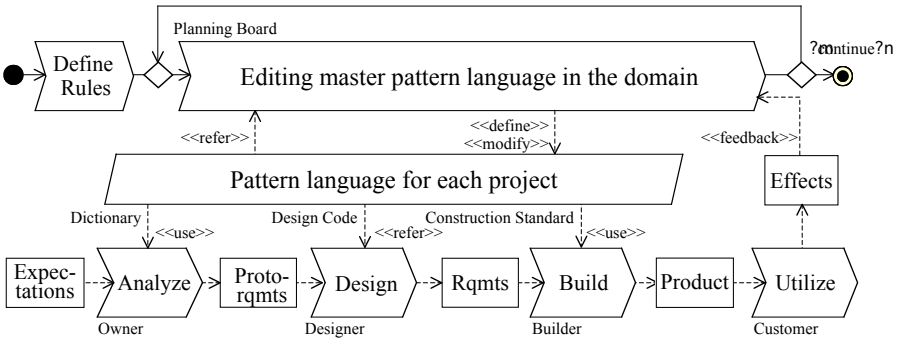

Figure 3. The information systems cycle with the pattern language

The users evaluate the quality of the products by operating them and reporting the results to the planning board. The planning board receives them and adds corrections to the master pattern language, if necessary. Through this cycle, it is expected that the knowledge of the principles or know-how of the business and the laws of the industry can be easily accumulated, and the quality of the entire EIS can be continuously improved.

Software Process Cycle

In the information systems cycle, the software process is one internal process generally considered to be the professional responsibility of software builders. For the owner, an information systems construction program is usually performed only once. In contrast, the builder has participated several times as a contractor in similar programs with other owners and has gained considerable knowledge. This knowledge is accumulated through a feedback loop for the builder called *the software cycle,* shown in the process diagram of Figure 4. Reusing such knowledge effectively will now be discussed as a software product line approach (Sugumaran, et al., 2006).

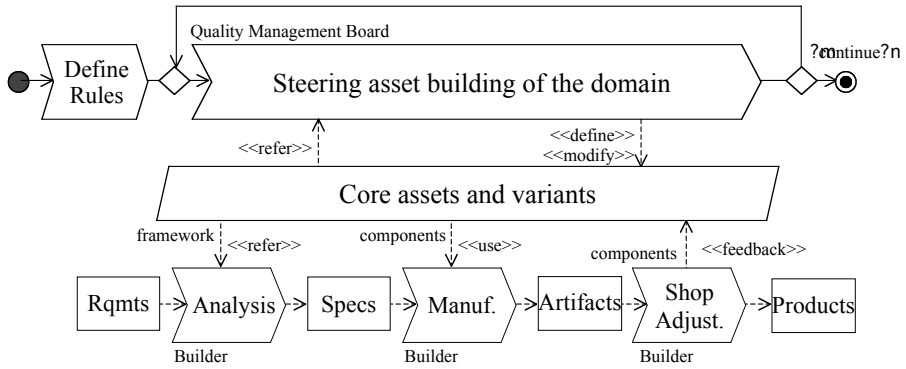

Figure 4. The software process cycle embedded in the software product line development

The Two Cycles Intersect Obliquely

We should consider that the information systems cycle of the owner and the software process cycle of the builder intersect obliquely (Figure 5). In the information systems cycle, the processes from providing the requirements to accepting the products are performed at this intersection. In the software process cycle, the processes from capturing the requirements to delivering the products are performed at this intersection.

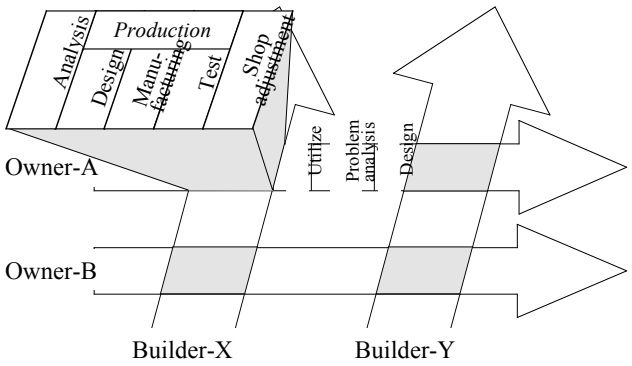

Figure 5. The intersection of the information systems cycle and the software process cycle

It should be emphasized that neither requirements nor proto-requirements exist on the upstream process of the software process cycle of the builder. Therefore, the requirements engineering requires an approach based on a sense of values different from software engineering. A discussion concerning the architect's role should not be avoided.

TOWARDS THE ACTUALIZATION OF THE PATTERN LANGUAGE FOR INFORMATION SYSTEMS

To perform actions based on the pattern language of information systems, it is useful to have a reference model corresponding to "*A Pattern Language*." The model language will be made according to the following procedures: (1) collect the required documents from past projects to use as proto-requirements again, (2) derive the technological vocabularies used in the proto-requirements and rewrite them into a pattern format, (3) correct or add each description of the patterns if there are duplications or gaps among collected patterns, and (4) adjust the duplications and gaps in the patterns again to align the patterns as a whole to derive a theme related to the society's or organization's business practices.

CONCLUSION

Information systems have come to play a significant role in business processes as businesses continue to expand and become more complicated. Software builders, not business owners, now plan, develop, and implement new information systems, excepting a few noteworthy companies. As the relationship between the purchaser and the contractor has emerged, the purchaser's responsibility in urban planning has been abandoned. As a result, architects have left. However, the use of software without the approval of the owner cannot yield satisfactory results.

Zachman's framework and the *Enterprise Architecture* allow the owner to regain sovereignty in planning, constructing, and operating EIS. This paper has redefined these processes as the information systems cycle, and proposed using pattern language activities in information systems to continue "the timeless way."

REFERENCES

Alexander, C. (1975). *The Oregon Experiment*, Oxford Univ. Press.

Alexander, C. et al. (1977). *A Pattern Language*, Oxford Univ. Press.

Alexander, C. (1979). *The Timeless Way of Building*, Oxford Univ. Press.

Alexander, C. (1985). *The Production of House*, Oxford Univ. Press.

Checkland, P. (1981). *Systems Thinking, Systems Practice*, John Wiley & Sons.

Davis, A. M. (2005). *Just Enough Requirements Management: Where Software Development Meets Marketing*, Dorset House Publishing.

Eriksson, H. and Penker, M. (2000). *Business Modeling with UML*, John Wiley & Sons.

Gamma, E. et al. (1995). *Design Patterns*, Addison Wesley, Reading, Massachusetts.

MSSG (Management System Study Group) Ed. (2004). *NTT DoCoMo Challenge for Real-time Management*, Nihon Kougyou Keizai Shinbun, Tokyo. (in Japanese).

Namba, Y. (2005). System integration and urban planning approach in enterprise information systemsÅGn *A discussion point of system integrations*, Nikka-Giren, pp. 79-92, (in Japanese).

Sugumaran, V., Park, S., and Kang, K. C. (2006). Software Product Line Engineering, *Communications of ACM*, Vol. 49, No. 12, pp. 29-32.

Teshima, A. (2002). Reproduction plan for information systems department being worriedÅC*Nikkei Computer*, No. 08/26, pp. 172-177, (in Japanese).

Zachman, J. A. (1987). A Framework for information systems architecture, *IBM SYSTEMS JOURNAL*, Vol. 26, No. 3, pp. 454-470.