韓國 컴퓨터情報學會 論文誌
第8卷 第2號, 2003. 6.

2003-8-2-2-1

웹 상에서의 비정형 시스템의 자율적 설계

최 정 우* 최 인 수**

A Web-Based Autonomous Design for Unstructured Systems

Jung-Woo Choi * In-Soo Choi **

요 약

비정형 시스템이란 각 시스템이 처한 환경변수에 따라 서로 다른 시스템의 설계와 구축이 필요하고 수시로 시스템의 변동과 수정이 필요한 시스템을 말한다. 지금까지는 이런 비정형 시스템은 대부분이 오프라인 상에서 구현되었기 때문에 시스템의 유지보수에 많은 투자가 필요하였으며 중앙 집중적 관리는 더더욱 어려웠었다. 이러한 방식의 시스템의 구현은 하나의 표준화된 틀에 맞추어 이루어졌기에 실세계를 정확하게 표현하지 못하였으며 비정형 시스템이 가진 특징을 적절하게 수용할 수 없었다. 이러한 문제를 해결하고자 시스템 내의 각 모듈의 설계를 자율적으로 할 수 있고 유지보수가 용이한 자율 시스템이라는 새로운 개념의 시스템 개발론을 제안하고 있다. 또한 이러한 자율 시스템 개발방법론을 활용한 웹 상에서의 시스템 구축 실례를 보여주고 있다.

Abstract

The unstructured system requires different system designs and realizations according to the environmental factors of the situation, as well as frequent system changes and revisions. Being realized off-line in most of the cases, such a system is in a great demand for maintenance and repair and very difficult to execute the central intensive management. In such a case, the unstructured system is usually fit into a single standardized framework, which makes it impossible to present the accurate picture of the real world and to properly reflect the features of the system. In an attempt to solve these problems, a new concept of system development theory was suggested; it is called the autonomous system, which enables each module to be designed autonomously and allows easy maintenance and repair. An example of system realization on the web using the development methodology of this autonomous system was presented.

* 현재 숭실대학교 산업·정보시스템공학과 박사과정
** 현재 숭실대학교 산업·정보시스템공학과 교수

Ⅰ. 서론

지금까지의 시스템 개발방법론은 정도의 차이는 있지만 모두 정형화된 관점에서 접근했던 것이 대부분이었다. 그러나 이러한 개발방법에는 다양한 여러 시스템들을 통합하여 표현하기에 한계가 있을 수밖에 없고, 또한 이 시스템들이 수시로 내부구조가 변경되는 가변적인 시스템이라면 내부구조의 변경이 필요할 때마다 시스템을 재구축 해야만 하는 번거로움이 뒤따르게 된다[1][2].

정형화된 구조론적인 관점에서 시스템을 구축하는 데에는 한계가 있기 마련이다. 실세계를 정형화시켜 가장 비슷하고 적절한 틀로 맞추는 과정에서 부득이하게 정확하게 표현하고 적용할 수 없는 부분들이 생겨나게 된다. 결과적으로 실세계를 모델링하여 시스템으로 구축하는 데에 있어서 정형화된 관점으로는 여러 문제점들이 발생하게 된다고 말 할 수 있는 것이다.

본 연구에서는 각자 처한 환경변수에 따라 서로 다른 시스템의 설계와 구축이 필요하고 수시로 시스템의 변동과 수정이 필요한 비정형 시스템을 예를 들고 있다. 일반적으로 비정형 시스템은 유지보수비용이 많이 든다. 환경의 변화, 또는 필요에 따라 수시로 내부모듈의 변화가 필요한 시스템이기 때문이다[1]. 이에 초기 시스템 설계 시 각 모듈의 설계가 필요한 것은 당연한 것이며 그렇다 하더라도 유지보수 및 수시로 시스템을 재 설계하는 것이 필수불가결 하다. 따라서 초기에 도입한 시스템을 계속 수정해주어야 하는 번거로움이 있으며 이에 관련한 비용문제도 발생하여 시스템의 효율성 면에서 많이 부족하다고 할 수 있다. 또한 오프라인(off-line)으로 구축된 응용프로그램이었기에 분산된 여러 시스템들을 관리하는 데에 많은 투자가 필요하였으며 중앙 집중적 관리는 또한 어려웠었다. 그러나 본 연구에서 소개될 자율 시스템의 적용으로 각 시스템 관리자의 의지에 따라 맞춤형 시스템의 설계가 가능하고 전체시스템의 수정과 관리 등도 사용자가 손쉽게 운영 가능하도록 연구에 목적을 두었다. 또한 웹을 기반으로 한 응용프로그램이기에 분산된 환경에서의 중앙 집중적 관리 또한 가능하게 될 것이다.

구조적 방법론이나 정보공학 방법론과 같은 시스템 개발방법론에서는 정보의 원천으로서의 메시지의 내용을 중시하였으나[3][4] 본 연구에서 논할 자율 시스템의 설계 사상에서는 이러한 정보의 원천으로서의 내용을 중시할 필요가 없으며 그 의미의 정확한 해석 또한 불필요하게 되고 있다. 구조적 방법론이나 정보공학 방법론에서는 정보전달자의 의미와 정보를 알아야 했지만 자율 시스템 개발방법론에서 이용된 기법에서는 그 의미내용들을 알 필요가 없다. 정보를 객체화시켜 전달하였기에 정보를 받는 사람은 그 정보의 의도와 의미를 알 필요가 없는 것이다.

그러므로 인터페이스 표준, 각종 개발 모듈의 이름(naming)표준, 시스템 아키텍처의 표준이 필요한 것이 아니다. 하나의 표준을 선택하여 구조적 설계를 해야만 한다는 것이 시스템 구축에 있어서 가장 어려운 문제점 중의 하나였는데 이에 대한 구체적인 해결책을 제시하여주는 것이 바로 자율 시스템이라 하겠다.

다시 말해, 사용자의 측면에서 보았을 때 요구하는 데이터는 전적으로 사용자의 자율적 의지로 구성이 되며 이후의 정보적 피드백이나 방법론적인 제약이 추가되거나 이를 필요로 하지 않는다.

본 연구에서는 자율 시스템이란 개념을 도입하여 기존의 정형화된 관점에서의 시스템 개발방법론과는 다른 자율적이고 가변성이 용이한 시스템의 구축에 관한 문제를 다루고자 한다. 또한 웹 상에서의 이 시스템의 접목과 활용가치, 효용성 등을 실 구축을 통하여 알아보고자 한다.

Ⅱ. 자율 시스템

2.1 자율 시스템의 등장배경

최근 정보기술의 급격한 변화에 따라 정보기술의 이용도 다양화되고, 그 응용분야도 다변화되고 있으며 소프트웨어 개발방법론도 기술 환경변화에 맞게 개선되고 있는 실정이다. 그러나 현실적으로는 객체지향 기법과 컴포넌트 기법과 같은 새로운 방법론이 출현되고 있음에도 불구하고 소프트웨어개발 프로젝트들에는 아직까지도 기존에 사용되고 있는 구조적 또는 정보공학 소프트웨어개발 방법론이 많이 사용되고 있다[4]. 그런데 기존의 구조적 또는 정보

공학 개발 방법론에는 대부분이 데이터 웨어하우스 및 웹 기술의 반영이 안 되어 있다는 것이 문제가 된다. 이러한 기존 방법론을 가지고 데이터 웨어하우스, 웹 기술을 적용하는 소프트웨어를 개발하는 프로젝트를 수행할 때에는 별도의 절차와 산출물이 필요하게 되고, 비효율적으로 프로젝트가 수행되게 된다. 이상의 문제점들 때문에 예를 들면 객체지향 기법과 컴포넌트 기법을 사용하는 새로운 방법론들을 사용하여 소프트웨어를 개발하여야 할 필요성이 대두되게 되는 것이다[4].

본 연구에서는 위에서 언급한 필요성에 부응할 수 있는 자율 시스템방법론이라는 새로운 개념의 개발방법론을 제시하고자 한다. 기존의 정형화된 구조적 또는 정보공학 개발 방법론과는 다른 자율적이고 가변성이 용이한 개발 시스템이며, 또한 객체지향 기법과 컴포넌트 기법의 개발 방법론과도 자율성과 가변성에 있어서 차별화 되는 개발 시스템인 자율 시스템을 본 연구에서 다루고자 한다.

2.2 구조화와 절차화

구조화란, 하나의 시스템에 있어서 시스템의 크기, 각 기능 단위의 독립성, 시스템 내의 인터페이스를 고려하여 시스템을 구체적인 물리적 구조로 설계하는 방식을 말한다. 이러한 구조화는 정형 구조화와 비정형 구조화로 대별할 수 있다. 정형 구조화란 시스템을 물리적 구조로 설계하는데 있어서 제약사항이나 기준사항들이 존재하여 하나의 구조적인 틀에 맞추어야만 하는 정형화된 구조화를 말하며, 이와 반대로 비정형 구조화란 제약사항이나 기준사항들이 존재하지 않거나 느슨한 제약사항이나 느슨한 기준사항들이 존재하여서 구조화가 가변적이며 자유롭게 되는 구조화를 말한다. 예를 들어, 비정형 구조를 지향할수록 본 연구에서는 구조화 자유도가 높다고 판정하고 있다.

절차화에 대해서는 프로그래밍 언어를 예로 들어 설명하고자 한다. 3세대 언어인 절차화 언어를 사용하자면 프로그래머는 컴퓨터가 수행해야 할 태스크를 단계별로 정확하게 명시해야 한다. 반면 4세대 언어인 비절차화 언어를 사용할 때에는 프로그래머는 결과를 얻는데 필요한 절차를 상세하게 명시하지 않아도 된다. 예를 들어 기계어, 어셈블리어, 절차어와 같이 절차화를 지향할수록 절차화 자유도가 낮게 되고, 비절차어, 자연어와 같이 비절차를 지향할수록 본 연구에서는 절차화 자유도가 높다고 판정하고 있다.

표 1. 프로그래밍 언어의 세대별 발전과정
Table. 1 The evolution of programming language

1세대	2세대	3세대	4세대	5세대
기계어	어셈블리어	절차어	비절차어	지능적 언어
길고 어려운 프로그래밍	반복적인 명령어들과 단축코드의 조합	절차적 구조적인 명령어들과 단축코드포함	비절차적, 구조적인 응용 생성적 언어과정	비절차적, 비구조적인 자연적 언어과정

시스템 소프트웨어와 응용소프트웨어를 만들기 위한 기초적인 환경을 제공하는 프로그래밍 언어는 〈표 1〉과 같은 1세대에서 5세대에 이르는 발전과정을 거쳐오고 있다[5]. 1세대부터 5세대까지의 발전과정 중에서 3세대와 4세대 그리고 5세대 언어의 발전을 주목할 필요가 있다. 3세대 언어인 절차어에서 4세대 언어인 비절차어로의 발전과 비절차어에서 5세대 언어인 지능적 언어로의 발전은 언어규칙이나 틀 또는 규정 중심의 체계로부터 자율적이고 능동적인 측면으로의 발전이라고 할 수 있겠다.

표 2. 프로그래밍 언어의 자유도
Table 2. Degree of freedom of programing languages

세대별 프로그래밍 언어	자유도	
	구조화 자유도	절차화 자유도
1 세대 - 기계어	낮음	낮음
2 세대 - 어셈블리어	↕	↕
3 세대 - 절차어	↕	↕
4 세대 - 비절차어	↕	↕
5 세대 - 지능적 언어	높음	높음

〈표 2〉에서 보는 바와 같이 프로그래밍 언어는 세대가 거듭할수록 구조화와 절차화 적인 관점에서의 변화가 보이게 된다. 절차화 부분에서는 4세대 언어가 비 절차화를 제시했으며 구조화 부분에서는 점점 그 정도가 약해지며 5세대 언어부터는 완전한 비 구조화를 표방하고 있다.

프로그래밍 언어의 발전과정과 더불어 앞서거니 뒤서거니 하면서 시스템 개발방법론도 발전을 거듭해 오고 있다. 1세대 언어인 기계어와 2세대 언어인 어셈블리어를 거쳐 3세대 언어인 절차어로의 프로그래밍 언어의 진보에 따라 프로그래밍 언어 중심의 구조적 방법론이 대두되게 되었다. 구조화적이며 절차화중심의 언어인 3세대언어를 사용하는 구조적 방법론은 절차 중심적으로 시스템영역을 분석하였다. 이러한 구조적 방법론은 기존의 업무를 개발해 놓은 시스템의 신속한 경영 의사결정 지원이 어렵다는 문제

로 인하여 절차화나 구조화의 제약사항들이 약한 정보공학 방법론이 출현하게 되었다. 정보공학 방법론은 SQL (Structured Query Language : 구조적 질의어)이라는 데이터베이스 조작언어와 3세대, 4세대 언어로 구현되어 졌다[4].

SQL을 정점으로 하는 4세대 언어는 3세대 언어에 비하여 개발자들이 코딩작업을 크게 덜어주고, 프로그램간의 종속성을 크게 완화시켰지만, 여전히 하드웨어에 종속적인 특성으로 인하여 프로그램의 재사용이 용이하지 않게 되고 있다. 또한 인터넷 기반 기술의 발전과 함께 원격의 분산 객체 기술에 해결책을 줄 수 없게 되었다. 이러한 문제점에 대한 해결책으로 등장한 것이 4세대 비절차 언어인 Java와 같은 순수 객체지향언어로 구현하는 객체지향 방법론이다. 그리고 최근 부각되고 있는 컴포넌트기반 개발방법론(Component Based Software Development : CBSD)는 역시 4세대 언어를 기반으로 하여 보편화된 인터넷 환경과 이 기종 컴퓨터 간의 연동기술의 발전으로 컴포넌트라고 정의되는 독립된 단위 기능의 소프트웨어 부품들로서 조립하여 시스템을 개발하는 방법론이다[4].

이와 같은 프로그래밍 언어의 발전과 이와 연관된 시스템 개발 방법론의 진보는 전술한 구조화 자유도와 절차화 자유도의 관점에서 프로그래밍 언어와 그 맥락을 같이 하고 있다. 개발 방법론에 따른 자유도를 살펴보면 〈표 3〉과 같다.

표 3. 개발 방법론의 자유도
Table 3. Degree of freedom of development methodologies

	자유도	
개발 방법론	구조화 자유도	절차화 자유도
구조적 방법론	낮음	낮음
정보공학 방법론	↑	↑
객체지향 방법론		
CBSD (컴포넌트기반 방법론)	↓	↓
자율 시스템 방법론	높음	높음

2.3 자율 시스템의 정의

자율 시스템(autonomous system)이란 구조화, 절차화의 측면에서의 자유도가 높은 시스템을 말한다. 각 모듈별 재설계와 새로운 모듈의 자율생성도 가능하게끔 하여 시스템 재 설계가 매우 용이한 시스템이라 할 수 있다. 구조화, 절차화의 관점에서 보았을 때 기존의 표준만을 지향한 시스템의 개발이 아니라 사용자의 자유로운 의지에 따라 자율생성이 가능한 시스템이라 할 수 있다. 자율생성이 가능하다는 점에 있어서 시스템 설계 시 객체의 정적인 특성만을 강조한 것이 아니라 객체의 동적인 특성과 생성된 객체 시스템과의 자연스러운 조화를 강조한 시스템이라 말할 수 있는 것이다.

각 시스템별 사용자들은 자율 시스템으로서 사용자 중심의 맞춤형 시스템 설계가 가능하다. 대용량 시스템의 유지보수 및 유연성, 사용자 편의성 등을 고려할 때 이와 같은 방법론은 최적의 솔루션을 제공할 수 있다 보여 진다. 시스템 내에 모듈의 변형이 필요할 때나 새로운 모듈의 생성이 필요할 때 각 시스템의 사용자들은 자율 시스템으로 모든 유지보수 및 재설계가 용이하며 사용자 중심의 시스템이 구축이 가능한 것이다.

기존의 일반화된 시스템에서는 시스템의 재 설계나 새로운 모듈의 추가 및 여러 변동 상황의 적용이 필요할 때 사용자는 개발자에게 요구사항들을 제시하게 된다. 그리고 충분한 의사소통이 이루어진 후에야 비로소 시스템의 재 설계가 개발자에 의해 이루어지게 되고 사용자는 계속 개발상황을 지켜보면서 조율을 해야만 하는 어려움이 따르게 된다. 이 과정에서 많은 재 설계비용이 소요되게 된다. 그러나 자율 시스템으로 설계된 시스템에서는 시스템을 사용하는 사용자 스스로가 시스템의 재 설계나 모듈의 추가 및 여러 변동 상황의 적용이 필요할 때 개발자의 도움 없이 시스템의 유연한 관리가 가능할 수 있는 것이다.

2.4 자율 시스템의 설계

자율 시스템의 설계에 있어서 핵심이 되는 부분은 사용자 스스로가 각자의 시스템에 맞게 수정 및 변형이 가능하게 하는데 있다.

시스템의 각 모듈의 명칭을 자유롭게 지정할 수 있으며 이렇게 생성된 각 모듈들의 구성을 모두 사용자의 자율에 맡겨진다. 자율생성이 가능하며 생성모듈의 구성이 자유롭게 이루어진다는 것은 사용자가 초기에 자신의 필요한 모델에 맞추어 시스템의 구축이 가능하게 할 수 있다.

또한, 여러 예외사항의 적용이나 특별한 분류의 경우 새로운 유형(pattern)을 관리자 임의의 제약조건으로 생성을 할 수 있게 하여 어떠한 경우일 지라도 시스템의 적용이 가능하게끔 한다.

최근 몇 년간 IT(Information Technology)방면의 비약적인 발전은 과거에 실행 불가능했던 많은 것들을 실행 가능하게끔 해줄 수 있는 기술적 발판을 만들어 주었다.

데이터베이스의 기술적 발전은 실세계를 좀더 사실적이며 구체적으로 표현할 수 있게 되었으며 시스템 구축상의 많은 편리성을 도모할 수 있었다. 또한 인터넷 분야의 빠른 성장을 가능케 하였던 웹(Web)기술의 비약적인 발전은 괄목할 만한 성과가 아닐 수 없는 것이다. 또한 사용자 중심의 여러 가지 업무분석 및 설계에 이용될 수 있는 작성하기 쉽고 이해하기 쉬운 다이어그램 및 작성 기준을 제시하여 과거의 프로그램 개발자 위주의 시스템 구축 방식을 사용자 위주로 변환시키는 계기를 만들어 주었다. 여기서 보여지는 자율시스템 역시 이러한 IT방면의 발전에 발판을 두고 가능하게끔 되게 한 것이었다.

본 연구에서 실제 아파트 단지의 관리빈 산정에 관한 비정형 시스템을 예로 들고 있다. 아파트 관리 시스템의 가장 큰 특징이 바로 일정한 표준을 가지고 있는 시스템이 아니라 수시로 환경의 변화에 따라 내부 모듈의 변화가 필요하고 각 아파트 단지별로 관리방법이 상이하여 시스템의 표준화가 어려운 시스템이라는 점이다. 실제로 아파트 A 단지는 아파트 B 단지나 C 단지와는 서로 다른 건물이며 또한 같은 단지 내의 아파트라 해도 각 건물별로 층수가 다르고 평수가 다양하여 각기 다른 관리체계와 관리비 영수체계를 가질 수 있는 것이다.

2.4.1 시스템 모듈의 자율생성원리

아파트 단지의 관리 시스템 같은 비정형 시스템은 처한 환경에 따라 각기 다른 모델을 제시하여야 하며 수시로 내부의 모듈이 변하는 시스템이다. 따라서 시스템 모듈을 자율적으로 생성, 삭제, 수정하는 부분은 매우 중요한 부분이라 할 수 있다.

시스템 모듈의 생성순서는 다음과 같다.

① 새로 생성할 모듈의 위치를 정한다.
② 새로 생성할 모듈의 이름을 명한다.
③ 시스템 내에서 동작할 수 있는 제약조건을 정하고 필요하다면 생성한 모듈에 연결될 하위모듈을 앞의 절차와 같이 생성한다.

이렇게 생성될 모듈의 위치와 이름을 정하여 시스템 내에 위치시키며 시스템 내에서 동작할 수 있는 제약조건을 명시하면 하나의 모듈이 생성되게 된다. 생성된 모듈의 하위모듈이 필요하다면 이 위치에서 앞에서와 같은 순서로 하위 모듈을 생성하는 것이다.

2.4.2 자율 시스템의 구성요소

본 연구의 4.3절에서 실제로 구현한 아파트관리시스템을 예시해 가면서 자율시스템의 구성요소를 설명하고자 한다. 자율시스템의 중요한 구성요소에는 항목관리, 유형관리, 그리고 Web DB Class의 3가지가 있다.

첫째, 항목관리란 필요한 모듈의 생성과 삭제, 수정을 담당하는 기능을 한다. 아파트관리시스템의 예를 들면 관리비영수에 필요한 요금항목을 생성하고 구성하는 기능을 담당하는 부분이다. 파일 구성은 다음과 같다.

```
pubinput/
        menu_manager.jsp - 항목리스트를 보여준다.
        menu_id.jsp - 항목을 선택하게 해준다.
        menu_input.jsp - 입력받은 데이터를 저장하여
                         항목을 생성한다.
        menu_inform.jsp - 항목을 만들기 위한 자료
                          입력 폼을 보여준다.
        menu del.jsp - 항목을 삭제한다.
```

둘째, 유형관리란 각 항목별 유형을 만들고 삭제하는 관리기능 및 각 유형에 할당된 제약사항의 입력, 입력된 제약사항의 각 단위별 계산 및 최종 항목의 처리결과를 만들어 낸다. 아파트 관리 시스템의 예를 들면 각 유형에 할당된 비용입력, 입력된 비용의 각 호별 계산 및 최종 항목 요금표를 만들어 내는 부분이라 할 수 있다. 파일 구성은 다음과 같다.

```
pubinput/
        aptype_apply.jsp - 유형정보를 활성화 한다.
        aptype_del.jsp - 유형을 삭제한다.
        aptype_inform.jsp - 새 유형을 추가하기 위한 입력
                            폼을 제공한다.
        aptype_input.jsp - 새 유형을 생성한다.
        aptype_manager.jsp - 유형의 목록을 보여준다.
        aptype_manager_sub.jsp - 각 유형의 세부데이터
                                 를 입력한다.
        aptype show.jsp - 최종 처리 결과를 보여준다.
```

셋째, Web DB Class라는 DB관련 클래스를 들 수 있다. 자바에서는 데이터베이스에 연결하고 쿼리를 수행하기 위해서 JDBC를 사용한다. 이것은 수많은 코드를 반복 입력해야만 했다. 때문에 이를 간단히 몇 과정으로 나누어서 사용할 수 있도록 하기 위해 만든 컴포넌트 개념의 클래스가 Web DB Class이다. 또한 이것은 사용의 편의를 위하여 웹 브라우저를 통하여 데이터베이스를 생성, 삭제 할 수 있는 관리 툴을 제공한다. 이 툴에서 생성한 데이터베이스 이름을 입력하는 것만으로 해당 데이터베이스에 자유롭게 연결이 가능하다. 파일 구성은 다음과 같다.

```
Package vwt.webdb;
public class Webdb2 extends Webdb{
  public String query;//쿼리를 입력한다.
  public String[][]result;//쿼리수행 결과가 저장된다.
  public string html;//쿼리결과를 HTML 형식으로 바꾼 내용
                     이 저장된다.
  public Webdb2(String vwtdbname) throws Exception
                     //생성자로서 관리툴에서 생성한 DB명을
                     입력할 수 있다.
  public Webdb2() throws Exception
                     //생성자로서 기본 DB에 연결된다.
  public void getQuery() throws Exception
                     //쿼리를 수행하고 결과를 가져온다.
  public void setQuery() therows Exception
                     //쿼리를 수행한다.
  public void resultToHtml(string nHtml)
                     //결과를 지정형식의 HTML 로 만든다.
```

이와 같이 이 세 가지의 구성요소는 자율시스템의 핵심 요소로서 전체 시스템 내에서 모듈의 자율생성과 구성을 담당하고 있다.

Ⅲ. 개발 방법론들 간의 비교

시스템 개발 방법론의 진보는 전술한 구조화 자유도와 절차화 자유도의 관점에서 프로그래밍 언어와 그 맥락을 같이 하고 있다. 이에 구조화 자유도와 절차화 자유도의 관점에서의 개발 방법론들 간의 비교분석을 하고자 한다.

3.1 구조화, 절차화의 자유도 산정

개발 방법론간의 구조화, 절차화의 자유도를 비교하기 위해 아래와 같은 방법을 사용하기로 한다.

먼저 SDLC(System Development Life Cycle) 에 따라 계획, 분석, 설계, 구현의 4단계로 구분하고 각 단계별로 가중치를 표 4와 같이 정한다[6].

표 7. SDLC 단계에 따른 가중치
Table 4. Weights of SDLC phases

SDLC 단계	계획	분석	설계	구현
가중치	15%	20%	35%	30%

개발 방법론의 SDLC 단계별 구조적 자유도와 절차적 자유도에 각각 1에서 6까지의 점수를 부여하고 표 4에서의 단계별 가중치를 곱하여 단계별 자유도 성향을 산정해 보기로 한다. 그리고 최종적으로 산정 된 각 단계별 자유도 성향을 합산하여 개발방법론의 구조적 자유도 성향과 절차적 자유도 성향의 값을 정하기로 한다.

예를 들어 구조적 자유도 성향과 절차적 자유도 성향을 알아보고자 하는 개발 방법론에서 설계 단계의 구조적 자유도는 4, 절차적 자유도는 3이라면 그림 1과 같이 표현되며 이에 대한 설계 단계의 구조적, 절차적 자유도는 다음과 같이 산정된다.

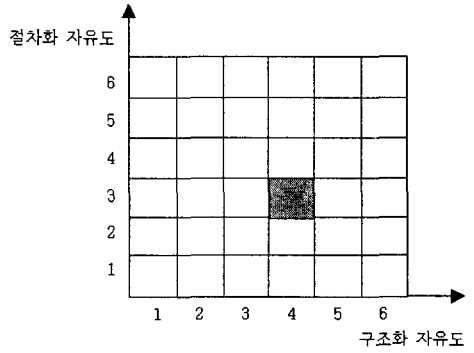

그림 1. 구조화 자유도와 절차화 자유도 매트릭스
Fig. Structural and procedural degree of freedom matrix

구하고자 하는 설계 단계의 구조적 자유도 성향은 이 단계의 구조적 자유도인 4에서 설계 단계의 가중치 0.35를 곱하여 1.40 이라는 값을 얻게 되며 구하고자 하는 설계단계의 절차적 자유도 성향 역시 이 단계의 절차적 자유도인 3에서 설계단계인 가중치 0.35를 곱하여 1.05라는 값을 얻게 된다.

3.2 개발 방법론들 간의 구조화, 절차화의 자유도 비교

개발 방법론들 간의 구조화, 절차화의 자유도를 비교하기 위하여 3.1의 구조화, 절차화의 자유도 산정기준을 이용하여 구조적 방법론, 정보공학 방법론, 객체지향 방법론, CBSD, 자율시스템 방법론을 비교하여 본다. 먼저 각 방법론들간의 구조적, 절차적인 비교를 통해 각각 1에서 6까지의 점수를 부여하고 3.1에서와 같은 자유도 산정 기준을 통하여 각 방법론들간의 구조적, 절차적 자유도 성향을 비교해 보도록 한다.

표 8. 비교 평가된 개발 방법론들 간의 구조화, 절차화 자유도
Table 5. Structural and procedural degree of freedom of development methodologies estimated by SDLC phases

	단계 / 자유도	계획	분석	설계	구현
구조적 방법론	구조적 자유도	2	1	1	1
	절차적 자유도	2	1	1	1
정보공학 방법론	구조적 자유도	2	3	3	3
	절차적 자유도	2	2	2	3
객체지향 방법론	구조적 자유도	2	4	4	4
	절차적 자유도	3	4	5	4
CBSD 방법론	구조적 자유도	4	5	5	5
	절차적 자유도	3	5	5	5
자율시스템 방법론	구조적 자유도	5	5	6	6
	절차적 자유도	4	5	6	6

〈표 5〉는 본 연구에서 비교 평가한 개발 방법론들 간의 구조화, 절차화의 자유도를 계획, 분석, 설계, 구현의 단계에 따라 1에서 6까지의 점수를 부여한 것이다.

개발 방법론들 간의 구조적, 절차적 자유도 성향을 종합하여 보면 〈표 6〉과 〈그림 2〉와 같이 표현될 수 있다.

표 9. 개발 방법론들 간의 구조화, 절차화 자유도 성향
Table 6. Estimated Structural and procedural degree of freedom inclination of development methodologies

	단계 / 자유도성향	계획	분석	설계	구현	Total
구조적 방법론	구조적 자유도 성향	0.30	0.20	0.35	0.30	1.15
	절차적 자유도 성향	0.30	0.20	0.35	0.30	1.15
정보공학 방법론	구조적 자유도 성향	0.30	0.60	1.05	0.90	2.85
	절차적 자유도 성향	0.30	0.40	0.70	0.90	2.30
객체지향 방법론	구조적 자유도 성향	0.30	0.80	1.40	1.20	3.70
	절차적 자유도 성향	0.45	0.80	1.75	1.20	3.20
CBSD 방법론	구조적 자유도 성향	0.60	1.00	1.75	1.50	4.85
	절차적 자유도 성향	0.45	1.00	1.75	1.50	4.70
자율시스템 방법론	구조적 자유도 성향	0.75	1.00	2.10	1.80	5.65
	절차적 자유도 성향	0.60	1.00	2.10	1.80	5.50

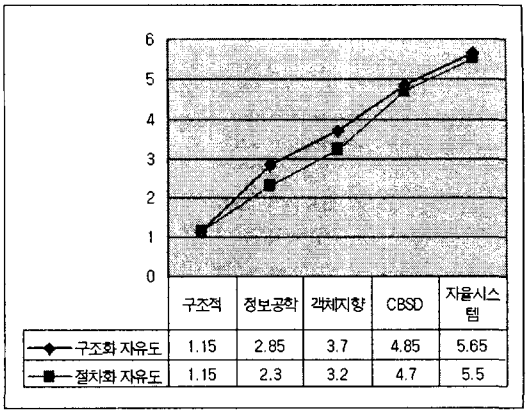

그림 2. 개발 방법론들 간의 구조화, 절차화 자유도 성향의 변화
Fig. 2 Structural and procedural degree of freedom inclination of development methodologies

개발 방법론이 진보되면서 구조적 자유도와 절차적 자유도 역시 그 정도가 향상됨을 확인할 수 있다.

Ⅳ. 자율 시스템의 웹 상에서의 구현

실제 자율 시스템을 웹 상에서 구현함에 있어서 웹 상에서의 구현의 이점과 그 실용화 방안 및 실현 가능성을 타진해 보도록 한다.

4.1 웹 상에서의 구현의 이점

인터넷 기반기술의 비약적인 발전은 오프라인(off-line)으로만 구현되었던 시스템구현의 기반 영역을 웹 상으로 이동시키고 있다. 특히 시스템을 웹으로 구현하였을 때에 오프라인을 기반으로 구현된 시스템과 몇 가지 차이가 있다. 예를 들어 지역적으로 널리 분산된 시스템의 경우에 웹을 기반으로 하여 구축을 한다면 여러 지역에 분산되어 있는 기간 시스템들 간에 효율적 정보교류가 가능해지며 이러한 분산시스템을 통제하는데 있어 적절한 해결책을 제시한다. 또한 정보통신망을 적극적으로 활용함으로써 조직의 계층 수가 축소되고, 수평화가 이루어질 수 있다. 그 결과 조직 내의 의사결정단계가 축소되고 한 관리자가 통제할 수 있는 범위(span of control)가 확대되어 업무처리가 신속 간소화될 수 있는 것이다.

본 연구에서 구현하고 하는 시스템은 분산 시스템이며 비정형시스템이다. 따라서 시스템의 웹 상으로의 구현은 필수적인 요소이다. 시간과 장소의 제약 없이 시스템을 관리, 운영할 수 있으며 지역적으로 분산된 각 기간시스템들 간의 개별적 특성을 각자 자율적으로 설계하며 구현할 수 있게 된다.

4.2 시스템 구현의 기반

시스템을 구현하는데 있어서의 개발 도구 및 개발 환경을 살펴보면 〈표 7〉과 같다.

표 10. 개발 환경
Table 7. Development environment

개발 환경	사 양
서버	Linux
웹서버	Apache
데이터베이스	Postgre SQL
기본 개발 언어	Java
서버 스크립트 언어	JSP
JSP 엔진	resin

기본적으로 개방 환경을 지향하는 개발환경에 중점을 두었으며 개발 언어로서는 순수 객체지향 언어인 자바(Java)를 이용하였다. 웹 상에서 구현하기 위한 서버 스크립트 언어로서는 JSP(Java Server Page)를 이용하여 기본 개발 언어와의 호환성을 최대한 배려하였다.

4.3 자율 시스템으로 구현한 A社의 아파트 관리시스템 사례

A기업의 실제 시스템의 개발과 적용을 위해 본 연구팀이 개발한 시스템의 간략한 소개를 할 것이다. 또한 이전 일반 시스템과의 비교를 통해 어떤 문제점들이 개선되었는가를 알아보도록 하겠다.

4.3.1 A社의 기업현황

1986년 설립된 A社는 공공기관, 기업체를 비롯하여 (아파트, 오피스텔, 복합매장) 관리 사무소의 전산정보 처리업무를 주력사업으로 동업종의 국내제일의 선도적인 역할을 10여 년 간 수행하여 왔다. 또한 아파트 500만 세대(5,000여 아파트단지) 입주민을 기반으로 한 사이버 공동체 사업의 일환으로 인터넷 사업-인터넷광고, 전자상거래 쇼핑몰, 멀티미디어 데이터베이스 컨텐츠, 인터넷/인트라넷 솔루션 개발, E-mail 마케팅을 새로운 사업으로 전개하고 있다. 한국의 아파트관리비 고지서의 특화된 업무영역에서 국내 최고의 위치를 확보하고 있는 상황에서 한국이 정보통신 대국이 되는 것과 발맞추어 '사이버 아파트'라는 거대한 영역으로 사업 확장을 목표하고 있다.

4.3.2 기술적인 문제점

아파트관리시스템 구현의 어려움을 주는 요소는 아파트 단지별 예외상황이나 개별적 특이사항 또는 시스템의 가변적 요소가 너무 많아서 전산화 작업에 많은 문제점을 가지고 있다는 것이었다.

이러한 문제에 대한 제약사항은 다음과 같다.

첫째, 공동 전기료 등은 아파트 단지별로 임의로 정해 부과한다.

둘째, 개별(세대별) 전기료는 단지별로 조견표 등록이 틀리며 부과세율 또한 유동적이다.

셋째, 각종 요금의 검침계가 건물의 설계대로 모두 틀리게 설치되어 있다.

넷째, 각종 항목들이 추가되거나 삭제 될 수 있다는 점은 데이터베이스의 유동적 생성을 필요로 한다는 것을 말하고 있다. 그렇다면 이러한 문제점들을 해결하고 나아가 이를 웹 기반으로 구축할 수 없는가에 대한 생각을 해볼 필요가 있다. 더욱이 A사는 500만 세대(5,000여 아파트단지)를 관리하는 전문 회사로 현재는 각 단지별로 월말에 data를 넘겨받아 일괄처리 방식을 취하고 있다. 이를 각 단지별로 개별입력 해 web을 통해 처리되는 네트워크 구성이 절실히 요구되고 있다. 이와 같은 문제점을 개선하기 위해 자율시스템으로의 구현에 초점을 맞추었다.

4.3.3 자율 시스템 구현으로의 문제점 해결

4.3.2절에서 언급된 문제들을 해결하기 위해 먼저 A社가 이전에 사용하던 시스템과 본 연구에서 제시한 자율 시스템과의 비교를 통해 새로 구축한 시스템이 실제 웹 상에서 적용되는 모습을 제시한다.

A社가 이전에 사용하였던 표준화 시스템은 다음과 같다.

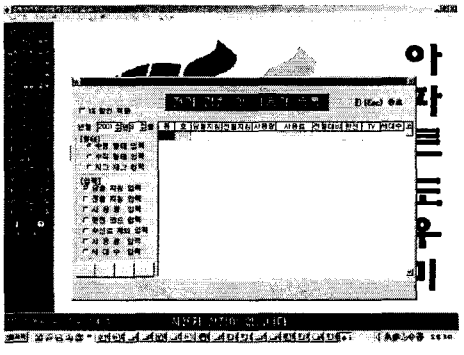

그림 3. A社의 아파트관리 시스템
Fig. 3 Apartment management system of A company

위 시스템상의 메뉴를 보면 기존의 문제점이었던 아파트 단지별의 예외사항들을 수용할 수 없는 시스템이었다. 또한 각 부과단지별로의 구조가 모두 다르므로 시스템 구축도 따로따로 단지별로 분리하여 이루어졌다. 위에서 언급한 바와 같이 공동 전기료 등은 아파트 단지별로 임의로 정해 부과한다는 것과 개별(세대별) 전기료는 단지별로 조견표 등록이 틀리며 부과 세율 또한 유동적이라는 점, 각종 항목들이 추가되거나 삭제 될 수 있다는 많은 예외조건들이 기존의 시스템으로는 해결할 수 없는 문제점들을 야기 시켰다. 그리하여 위의 문제해결은 전산화가 아닌 수작업으로 이루어지거나 새로운 시스템의 구축으로 이어지고 이와 같은 결과는 유지보수비용의 증대로 나타날 수밖에 없었다.

〈그림 4〉에서는 본 논문에서 제시한 자율시스템의 실제 구축 화면을 보여주고 있다.

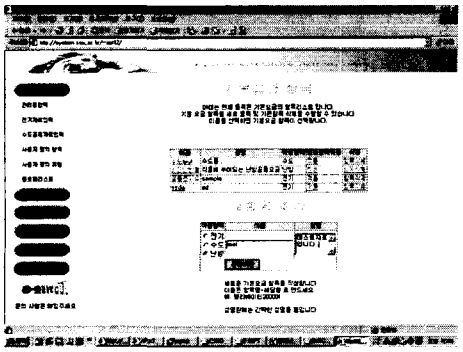

그림 4. 자율 시스템으로 구현한 아파트관리 시스템
Fig. 4 An apartment management system by the autonomous system methodology

기존의 문제점이었던 아파트 단지별의 예외사항들을 수용하기 위하여 필요한 시스템 모듈을 사용자 정의에 맡기어 자율생성이 가능하도록 구축하였다. 이것으로 인하여 각 단지별 시스템의 구성은 관리자의 책임 하에 자율적으로 이루어질 수 있으며 각 부문별의 예외사항적용은 관리자의 간단한 등록작업으로 해결될 수 있게 되었다. 공동 전기료 등은 아파트 단지별로 임의로 정해 부과한다는 것과 개별(세대별) 전기료는 단지별로 조견표 등록이 틀리며 부과 세율 또한 유동적이라는 점, 각종 항목들이 추가되거나 삭제 될 수 있다는 많은 예외조건들은 위와 같은 자율시스템으로서 해결될 수 있었다. 단지별로 서로 다른 시스템을 구축할 필요 없이 웹 기반의 하나의 기본적인 모듈시스템으로서 그 단지별 관리자가 그 단지에 맞는 전체 시스템을 만들어 가는 것이라고 할 수 있을 것이다. 또 하나의 문제점이었던 관리의 중앙 집중화는 자율시스템의 웹 상에서의 구축이라는 것으로 해결할 수 있었다. A社는 500만 세대(5,000여 아파트단지)를 관리하는 전문 회사로 현재는 각 단지별로 월말에 데이터를 넘겨받아 일괄처리 방식을 취하고 있다. 이를 각 단지별로 개별입력 해 웹을 통해 처리되는 네트워크 구성이 이루어짐으로서 A사의 중앙본부에서는 각 관리자별(단지별)의 시스템 체크 및 관리가 수월해 질 수 있었다.

다음에서는 실제로 자율 시스템 내에서 새로운 모듈을 생성하여 시스템 내에 위치시키는 과정을 보여준다.

새로운 시스템 모듈을 생성하여 시스템에 위치되는 과정은 다음과 같다.

(1) 새 항목을 추가한다.

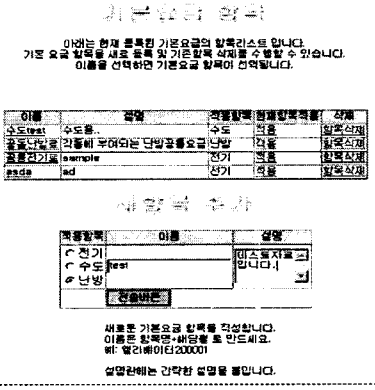

그림 5. 새로운 항목의 추가
Fig. 5 Addition of a new module

(2) 새 항목이 전체 시스템 내에 추가된 모습을 보여준다.

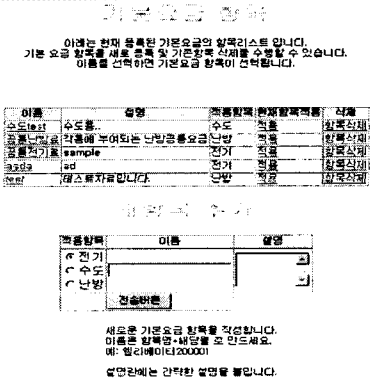

그림 6. 추가된 항목의 조회
Fig. 6 View of an added module

(3) 새로 작성된 항목을 적용할 시점을 정한다.

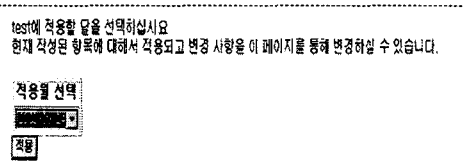

그림 7. 추가된 항목의 적용시점 선택
Fig. 7 Adoption period of an added module

(4) 새로 작성된 항목의 하위 유형들을 생성한다.

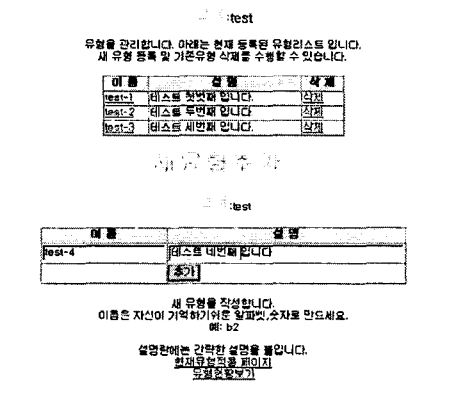

그림 8. 하위 유형의 추가
Fig. 8 Addition of a new sub-module

(5) 만든 유형을 적용할 범위를 정한다.

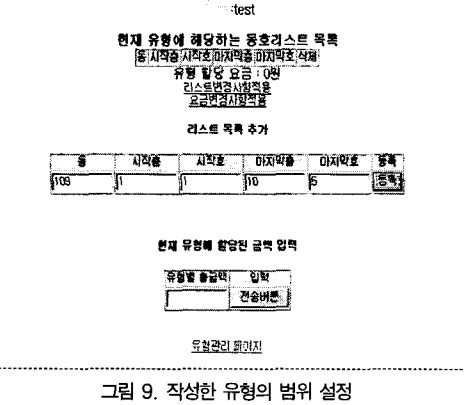

그림 9. 작성한 유형의 범위 설정
Fig. 9 Extent adjustment of an added sub-module

위와 같은 과정을 통해 시스템 모듈의 자율생성이 가능하며 시스템 구조 또한 사용자 스스로가 만들 수 있는 것이다.

V. 결론

지금까지 본 연구에서는 자율 시스템이라는 새로운 시스템개발 방법론을 채택함으로써 기존의 표준화된 시스템 개발 방법론으로 시스템을 설계하였을 경우에 발생하는 여러 어려움을 제거하고 실현 불가능했던 부분들을 구현시키는 것이 가능함을 보여주었다. 실제 자율시스템을 채택함으로써 비정형 시스템이라는 조건하에서도 완벽한 설계를 이룰 수 있었으며 수많은 예외상황들도 전체시스템에 적절히 수용시킬 수 있게 되었다. 또한 생성된 모듈의 수정 및 삭제 또한 자유롭게 이루어질 수 있었으며, 이러한 모든 프로세스가 웹 상에서 이루어진다는 면에서 무한한 가능성과 이점을 창출할 수 있을 것이라 생각한다.

본 연구에서 제시한 자율시스템의 적용범위는 현재 아파트 관리 시스템에 국한되어있지만 비정형 시스템의 성격을 가지고 있는 여러 다양한 시스템으로의 적용이 가능할 것이라 기대된다. 특히 시간이나 환경조건에 따라 여러 가지의 유형으로 변화가 필요한 시스템이라면 자율 시스템의 적용이 필수 불가결한 것이 될 수 있을 것이다. 또한 현재 운용중인 시스템들에 부분적으로 자율시스템을 적용하였을 시에 대한 가능성 및 실효성을 검증해보는 것도 추후 연구과제가 되리라 생각한다.

참고문헌

[1] Shoji Miyamoto, Kinji Mori and Kirokazu Ihara, "Autonomous Decentralized Control and it's Application to the Rapid Transit System", Computer in industry, Volume 5, Issue 2, pp. 115-124, 1984

[2] Hideo Hanamura, Isao Kaji and Kinji Mori, "Autonomous Consistency Technique in

Distributed Database with Heterogeneous Requirements", IPDPS 2000 Workshops, LNCS 1800, pp. 706 - 712, 2000.
[3] Leon Brillouin, "Science And Information Theory, Second Edition", Academic Press Inc., pp. 297 - 299, 1962.
[4] 김상하, 통합 객체지향 방법론 모델링 및 설계구축 실무, 으뜸정보교육출판, 1999
[5] Efraim Turban, Rex K. Rainer, Jr., and Richard Potter, "Introduction to Information Technology", John Wiley & Sons, Inc., pp. 113-132, 2001.
[6] Alan Dennis, Barbara H. Wixom, "Systems Analysis & Design, Second Edition", John Wiley & Sons, Inc., pp. 60-67, 2003.
[7] David Flanagan, "Database System Concepts", O'REILLY, 1990
[8] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen, "Object - Oriented Modeling and Design, Prentice - Hall International", Inc., 1991.
[9] Henry F. Korth, Abraham Silberschatz, "Database System Concepts Second Edition", McGraw-Hill, Inc., 1991.
[10] Raymond McLeod, Jr., George Schell, "Management Information Systems Eighth Edition", Prentice - Hall, 2001.
[11] James A. O'Brien, Introduction to Information Systems tenth edition McGraw - Hill, Inc., 2001.
[12] Geoff Cutts, Structured Systems Analysis and Design Methodology, Second Edition, Alfred Waller Limited Publishers, 1991.

저 자 소 개

최 정 우

숭실대학교 산업·정보시스템공학과 학사
숭실대학교 MIS 석사
현재 숭실대학교 산업·정보시스템공학과 박사과정
〈관심분야〉 MIS, 정보시스템분석·설계, 분산시스템

최 인 수

서울대학교 산업공학 공학박사
영국 College of Librarianship, Wales(Diploma), 日本 東洋大學 經營學部 객원교수
현재 숭실대학교 산업·정보시스템공학과 교수에 재직 중
<관심분야> 객체지향데이터베이스, 정보시스템 분석·설계, ERP, MIS