교 육 학 석 사 학 위 논 문

응용프로그램의 프로토타입 표현에 관한 연구

2012년 2월

부경대학교 교육대학원

전산교육전공

옥 영 종

교 육 학 석 사 학 위 논 문

응용프로그램의 프로토타입 표현에 관한 연구

지도교수 여 정 모

이 논문을 교육학석사 학위논문으로 제출함.

2012년 2월

부경대학교 교육대학원

전산교육전공

옥 영 종

옥영종의 교육학석사 학위논문을 인준함.

2012년 2월 24일

주 심 공학박사 조 경 연 (인)

위 원 공학박사 서 경 룡 (인)

위 원 공학박사 여 정 모 (인)

목 차

그 림 목 차

표 목 차

A Study on The Prototype representation of application program

Young-Jong Ock

Department of Computer Education, Graduate School of Education
Pukyong National University

Abstract

A prototype needs to show the requirements of the users and help them understand better. These prototypes are useful in a variety of industries. But most existing prototypes depend on the visual representation. This method of visual representation has a limitation, which can't express the detailed contents. This paper suggests an representation method of linguistic formation as an alternative. The prototype of linguistic representation can demonstrate the detailed contents of the requirements, keep a good communication among those who are related to the project, and allow them to deal with a large number of possible problems which may happen from the developing process. As it's also easy to make the developing process documented, the prototype will be helpful to improve the software development.

Ⅰ. 서 론

프로토타입은 일상적으로 시작품으로 불리며, 최종적인 디자인 해결안을 실제로 구현하여 평가해 볼 수 있는 도구를 말한다[1]. 이것은 사용자가 요구한 시스템의 초기 일부분을 보여주어 사용자와 개발자 사이에 이해의 격차를 줄여준다. 그리하여 사용자의 요구사항 대로 개발하는데 유용하다.

프로토타입은 여러 산업분야에서 다양하게 활용되고 있다. 건축이나 선박 건조 시에 설계도나 모형을 만들어 시험한다. 자동차 업계에서 신 모델을 출시할 때는 필수적으로 컨셉트 카(Concept car)를 통해 소비자의 경향과 요구사항을 파악한다. 이와 같이 어떤 기술이 실제 생산에서 구현 가능한 것인지, 어떤 가치들이 시장 경쟁력을 강화시킬 수 있는지 충분히 시험을 해 본 후에야 양산 플랜트를 구축하고 있다.

소프트웨어 개발 과정에서도 페이퍼 프로토타입(Paper prototype)을 통해 사용자의 요구사항을 표현한다. 기존의 프로토타입 표현 기법들의 공통점은 바로 시각적인 표현에만 의존하고 있다는 것과 규칙화 되지 못한 표현 방법으로 인해 커뮤니케이션의 문제가 발생 할 수 있다는 것이다. 가장 많이 활용되고 있는 페이퍼 프로토타입은 요구사항의 상호관계가 복잡한 경우 나타내기에 복잡하고 세부적인 동작의 흐름을 표현하기 힘들며, 무엇보다도 표현 기법들이 표준화 되지 못해 여러 사람들과 공유하기 어렵다는 단점이 있다. 시각적 표현 기법의 프로토타입[2]은 대부분 디자인 작업의 결과로 작성된 것으로, 외형적인 측면에 너무 치중한 나머지 근본적인 문제의 본질이 숨겨질 수 있다.

이론적으로 개발자에게 주어지는 요구 명세서[3]는 최종적인 내용이 확정된 문서이어야 한다. 소프트웨어 프로젝트는 개발과정 가운데 요구사항이 지속적으로 수정된다. 하지만 시각적인 표현기법으로는 이러한 요구사항의

지속적인 변화에 신속히 대응하기 어려운 단점이 있다. 이러한 문제점의 대안으로 본 논문에서는 프로토타입의 새로운 표현 기법으로서 정형적인 언어(Formal language)를 기반으로 문자나 기호로 구성된 언어 표현 기법을 제안한다. 제안하는 표현기법은 프로타입을 기술하는 언어로서 문자나 기호는 규칙을 바탕으로 프로토타입을 표현하며, 이렇게 표현된 언어 프로토타입은 복잡한 상호관계를 보다 간단히 표현 할 수 있다. 이렇게 표현된 언어 프로토타입은 프로토타입 그 자체가 바로 소프트웨어 개발과정에서의 문서화로 이어 질수 있는 특징을 가지고 있다.

본 논문의 구성은 다음과 같다. 서론에 이어 제2장에서는 프로토타입과 관련된 요구사항과 프로토타이핑 그 외 관련된 내용들을 소개한다. 제 3장에서는 본 논문에서 제안하는 언어 프로토타입에 대하여 기술하고, 제 4장에서는 제안하는 언어 프로토타입을 적용한 설계 및 기존의 표현기법과 본 논문에서 제안하는 기법을 비교분석 하고, 제 5장에서 결론을 맺는다.

Ⅱ. 관련연구

이 장에서는 본 논문의 연구와 관련이 되는 요구사항, 프로토타이핑에 대한 이해와 특징에 대해 소개한다. 그리고 요구사항을 분명하게 표현하는 도구로 프로토타입에 대해 알아보고, 그 외 관련된 내용들도 소개한다.

2.1 요구사항

요구사항은 고객(시스템의 구축을 의뢰하는 사람) 또는 사용자(개발된 시스템을 사용하는 사람)가 개발될 시스템에 대해서 요구하는 모든 것을 의미하며, 구현되어야 하는 것에 대한 명세서이다[14]. 시스템이 어떻게 동작해야 하는지 또는 시스템 속성이나 특성에 대한 설명이다. 이것은 시스템 개발 과정의 제약조건이 될 수도 있다.

모든 소프트웨어 프로젝트는 요구사항(Requirement)을 가지고 있다. 소프트웨어 시스템을 구축 하는데 있어서 가장 어려운 한 부분은 무엇을 구축할 것인지, 즉 요구사항을 정확하게 판단하는 것이다.

소프트웨어 산업에서 가장 일반적으로 보고되는 문제들은 사용자의 요구사항을 파악하고 관리하는 것이다. 요구사항이란 '설계를 선택하게 하는 모든 것' 이라고 정의하며, IEEE 표준 용어집은 문제를 해결하거나 목적을 달성하기 위하여 사용자가 필요로 하는 조건 또는 기능이다. 계약, 표준, 명세 또는 정규화 된 문서를 충족하기 위하여 시스템 또는 시스템 컴포넌트가 처리하거나 충족시켜야 하는 조건 또는 기능에 대해 문서로 표현된 것이라 정의하고 있다. 이와 같이 요구사항에 대한 사용자의 관점과 개발자의 관점을 모두 포함하고 있다.

2.1.1 요구사항의 불확실성

요구사항을 분명하게 분석하는 것은 소프트웨어 프로젝트에서 매우 중요한 일이다. 올바른 요구사항이 없다면, 프로젝트를 주어진 일정과 예산 내에서 마치고 계획된 시스템의 사용자들을 만족시킬 가능성은 거의 사라지게 된다.

하지만 아무리 완벽한 요구사항 명세서라 할지라도 개발자와 사용자간에는 이해의 격차가 발생한다. 이러한 불명확한 요구사항은 이해당사자간의 합의를 이끌어 내지 못하여 추후 잦은 요구사항 변경을 초래하게 되고, 이는 전체 소프트웨어 개발 및 관리 활동을 어렵게 하는 요인이 된다[13]. 애매모호한 증상으로 인해 요구사항을 읽는 사람이 여러 가지로 해석할 수 있고, 여러 사람이 그 의미를 서로 다르게 이해한다는 것이다. 이러한 애매모호함은 여러 관련자에게 서로 다른 기대를 만들어 내고, 개발자가 잘못된 문제에 대한 솔루션을 구현하게 되면 제품이 개발자가 의도한 것과 다르게 동작할 것으로 기대의 격차를 해소하기 위하여 시간을 또 낭비하게 된다.

애매모호함을 제거하는 한 가지 방법은 서로 다른 관점을 제시하는 사람들이 요구사항을 보다 구체적으로 검토할 수 있도록 만드는 것이다. 단순히 요구사항 문서를 회람시켜 주석을 달게 하는 것만으로는 애매모호함을 제거하지 못한다. 서로 다른 검토자가 요구사항을 서로 다르게 해석하겠지만, 그 중 하나라도 제대로 된 것이라면 프로젝트 후반부까지는 애매모호함이 나타나지 않으며, 그 때는 이미 상당히 늦은 시기가 된다. 요구사항에 대해 테스트 케이스를 작성하고, 프로토타입을 만드는 것이 애매모호함을 발견하는 방법이다.

2.1.2 불확실한 요구사항으로 인한 발생비용

성공적인 시스템 개발을 위해 보장되어야 하는 활동 중의 하나는 사용자의 요구사항을 정확히 파악하는 것이다. 정확하지 않은 요구 명세서로 인한 영향은 시스템 개발의 비용 초과, 공정 지체, 품질 저하의 원인이 된다. 요구사항 오류는 재작업 비용의 70%~85%를 차지한다.

그림1에서와 같이 프로젝트 후반부에 발견되는 문제를 수정하는 비용은 문제 발생 즉시 수정하는 것보다 훨씬 더 많은 비용이 든다. 요구사항 에러를 예방하고 초기에 파악하는 것은 요구사항 오류로 인한 재작업을 크게 줄여 준다. 이러한 사용자의 요구사항을 분석하는 목적은 사용자의 요구가 무엇인지를 정확히 도출하기 위해 요구사항을 수집하며, 분석하고, 기록하는 활동이라 할 수 있다[4].

일반적으로 사용자의 요구사항을 정확하게 파악하는 것은 무척 어려운 일로 알려져 있다. 통계에 의하면 프로젝트 실패에 직접적인 영향을 미치는 요인으로 요구 사항을 파악하는 오류가 13.1%, 불충분한 사용자의 참여가 12.4%, 그리고 요구 사항의 변경이 8.1%로 사용자 요구와 관련된 오류 요인이 전체의 32% 가량을 차지하고 있는 것으로 보고되었다[2]. 통계에 나타나듯이 요구 사항을 분명하게 표현하는 것이 매우 중요하며, 요구사항을 분명하게 표현하기 위해서는 프로토타입이란 도구를 활용한다. 이렇게 요구사항을 표현하는 것이 프로젝트의 성패를 좌우하는 중요한 요소이다.

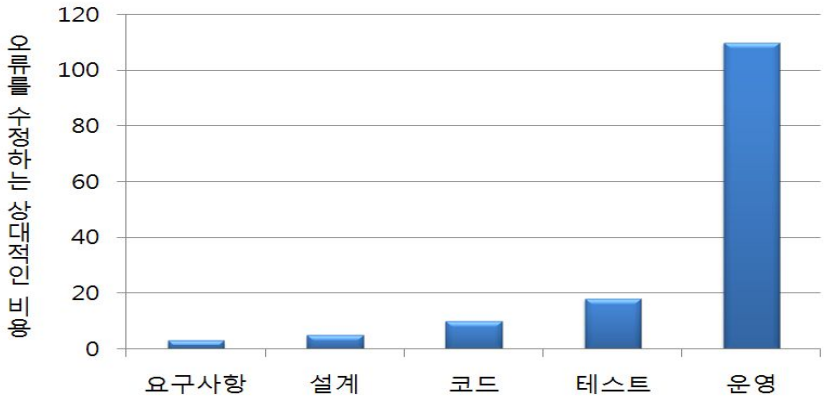

그림1. 요구사항 오류가 발견 시기에 따른 상대적인 비용

2.1.3 요구사항 명세서

요구사항 명세서(Requirements Specification)는 분석가와 클라이언트 사이에 해결할 문제를 서로 이해할 수 있도록 하는 문서이다[6]. 이것은 공식적이든 비공식적이든 간에 개발자와 클라이언트 사이의 계약의 기초가 된다. 요구사항 명세서는 다음과 같은 특징을 가지고 있다.

1) 명세서는 설계단계를 위한 출발점이다. 따라서 명세는 아주 정확한 수학적인 기술이 더 바람직한 반면에 다른 사용자가 이해할 수 있어야 한다. 그것은 읽기 쉬운 문서를 의미한다.

2) 명세서는 정확해야 한다. 정확성을 보장하는 절차는 없다. 요구사항 명세는 정확성을 평가하기 위해 사용자의 실제적인 요구와 다른 상위 문서에 대해 유효성을 인정받아야 한다.

3) **명세서는 명확해야 한다.** 사용하는 사람과 만드는 사람 양쪽 모두에게 명백해야 하며, 요구사항을 일관되게 이해할 수 있어야 한다.

4) **명세서는 완전해야 한다.** 이것은 기능성, 성능, 제약사항과 같이 모든 중요한 것은 문서화되어야 하므로 정확한 입력과 잘못된 입력 둘 다에 대한 응답이 명세 되어야 한다.

5) **명세서는 증명 가능하여야 한다.** 이것은 요구사항이 충돌 하는지 아닌지를 결정하기 위한 제한된 프로세스가 있다는 것을 의미한다. '시스템은 사용자에게 친숙하다' 와 같은 구는 증명가능하지 않다. 마찬가지로, '시스템의 응답시간은 보통 2초보다 적다' 에서처럼 측정할 수 없는 사용은 피해야 한다.

6) **명세서는 수정 가능해야 한다.** 이것은 소프트웨어 모델 사실의 일부분이다. 그러므로 이것은 변화가능하다. 상응하는 요구사항 명세서는 모델화된 사실과 함께 발전한다. 그러므로 문서는 변화를 즉시 수용할 수 있는 방식으로 구성되어야 한다. 쓸데없는 중복은 가능한 피해야 한다. 왜냐하면 변화로 인한 불일치의 위험이 있기 때문이다.

7) **명세서는 추적 가능하여야 한다.** 이것은 각 요구사항 또는 모든 요구사항의 근원과 원리는 추적 가능해야 한다. 분명하고 일관된 순위 도표는 다른 문서가 요구사항 명세의 부분들을 조회하는 것을 가능하게 만든다.

2.2 요구사항 분석 단계

요구사항 분석 단계에는 요구사항 수집, 분석, 검증, 명세의 작업이 있다. 이런 과정들은 소프트웨어가 포함된 제품의 요구사항을 수집하고 평가하고 문서로 만드는 것과 관련된 모든 활동을 포함하고 있다. 각각의 세부적인 절차는 다음과 같다.

2.2.1 요구사항 수집

프로젝트의 요구사항 수집은 어떠한 필수적인 사용자 요구사항도 배제해서는 안 되며, 특정 사용자 요구사항에 대해 모든 기능 요구사항을 추적할 수 있어야 한다. 또한 품질과 성능 기대와 같은 기능 이외의 요구사항도 적절한 소스에서 수집해야 한다.

2.2.2 요구사항 분석

요구사항 분석은 모든 관련자들이 요구사항을 이해하고 에러, 누락과 다른 문제들이 있는지를 정밀하게 검토하도록 요구사항을 다듬는 것을 말한다. 분석에는 고차원의 요구사항을 상세수준으로 분해하고, 프로토타입을 만들고 타당성을 평가하고 우선순위를 조정하는 것 등이 포함되어 있다. 분석의 목적은 관리자가 현실적인 프로젝트 예측을 하고 기술 스태프가 설계, 구성과 테스트 작업을 진행할 수 있도록 충분한 품질과 상세내용을 가진 요구사항을 개발하는 것이다.

2.2.3 요구사항 검증

요구사항 검증은 요구사항 설명이 적절한지를 검토하고, 바람직한 품질 특징을 알려주고 고객의 요구를 충족시킨다. 요구사항 명세서를 읽을 때 적합한 것으로 보이는 요구사항들이 개발자들에게는 문제로 밝혀질 수도 있다. 요구사항에서 테스트 케이스를 작성하면 종종 애매모호함과 무의미한 내용이 밝혀지는 경우가 많다. 요구사항이 시스템 테스트 또는 사용자 승인 테스트를 통해 설계와 최종 시스템 확인을 위한 안정적인 기반 역할 하기 위해서는 요구사항 검증과정에 적절한 검토 과정이 필요하다.

2.2.4 요구사항 명세

요구사항을 검토할 수 있는 방식으로 문서화가 필요하며, 요구사항 명세는 일관성 있고 쉽게 사용하며 검토할 수 있는 방식을 문서화해야 한다. 고객의 요구사항을 수집하고 이해했다면 이제 그것을 토대로 개발자들이 볼 수 있도록 상세하게 기록을 해야 한다. 다시 말해 요구사항 명세는 분석된 요구사항을 명확하고 완전하게 기록하는 것으로, 소프트웨어 시스템이 수행해야 할 모든 기능과 시스템에 관련된 구현 사의 제약 사항 및 개발자와 사용자가 합의한 성능에 관한 사항 등을 명세하는 것이다.

2.3 프로토타입

요구사항을 분명하게 표현하기 위한 방법으로 프로토타입이란 도구를 사용한다. 프로토타입은 활용되는 특성에 따라 하드웨어적인 프로토타입과 소프트웨어적인 프로토타입으로 구분할 수 있다. 하드웨어적인 프로토타입[3]은 모바일 같은 기계나, ATM(Automated Teller Machine)과 같은 기계를 디자인할 때 활용된다.

소프트웨어적인 프로토타입은 특정한 소프트웨어 프로그램을 개발할 때 활용된다. 본 논문에서는 응용프로그램을 프로토타입으로 표현을 제안하는 것이므로 소프트웨어적인 프로토타입에 중점으로 프로토타입 정의, 종류, 분류, 특징에 대해 설명한다.

2.3.1 프로토타이핑

프로토타이핑(Prototyping)은 요구사항을 도출하는 개발접근법의 하나로서 개발초기에 시스템의 프로토타입을 간단히 만들어 사용자에게 보여 준다. 사용자가 정보시스템을 직접 사용해 보게 함으로서 기능의 추가, 변경 및 삭제 등을 요구하면 이를 즉각 반영하여 정보시스템 설계를 다시 하고 프로토타입을 재구축하는 과정으로 사용자가 만족할 때까지 반복해 나가면서 시스템을 개선시켜 나가는 방식이다.

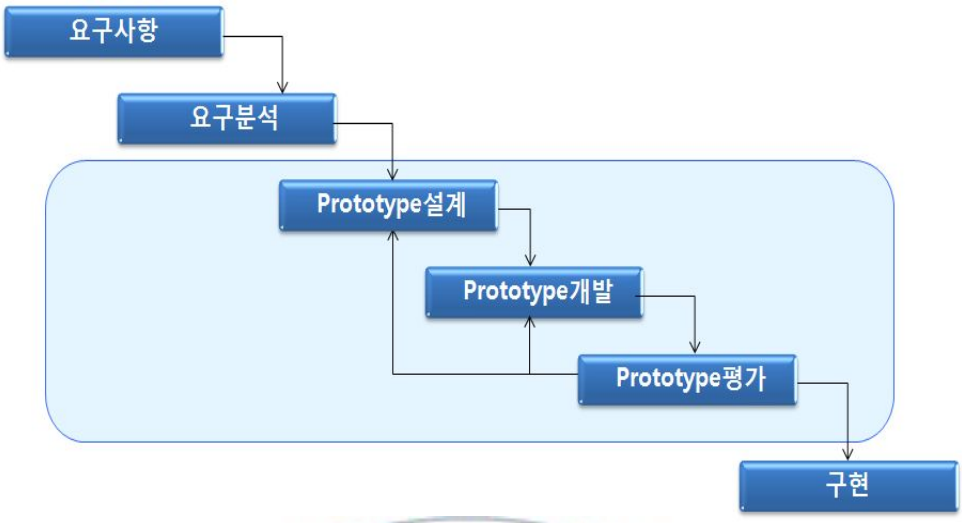

그림 2. 프로토타이핑의 프로세스

프로토타이핑은 그림2의 요구분석 단계로부터 시작한 시작하여, 프로토타입 설계, 프로토타입 개발, 프로토타입 평가의 반복적인 과정을 의미한다. 프로토타입 설계 과정에서는 프로토타입을 이용하면 시범 시스템을 쉽게 만들 수 있다. 프로토타입 설계 과정을 통해 제작된 프로토타입은 프로토타입 평가라는 과정을 거쳐 개발자와 고객간의 요구사항을 파악하는 의사소통의 도구로서 활용된다. 평가 과정을 통해서 새로운 요구사항을 다시 프로토타입으로 설계하여 반영할 수도 있다.

실제 소프트웨어의 첫 번째 버전을 만드는 경우도 있다. 이것은 실제 시스템이 설치될 환경과 같은 조건에서 사용될 시스템을 만드는 것으로 여기에 기능을 추가하고, 변경하고, 문서화하면 프로젝트의 최종 생산물이 된다. 프로토타이핑은 그림2와 같이 요구사항 분석 단계로부터 시작한다. 발주자나 사용자는 한 번에 완전한 요구를 낼 수 없기 때문에 프로토타입이 설계된다.

프로토타입이 구현된 후에 발주자와 개발자는 이를 평가하여 요구를 수정

한다. 새로운 요구에 따라 프로토타입을 수정하거나 보완하고 이를 확장하면 시스템이 구현되는 것이다. 이러한 프로토타이핑 과정을 통해서 소프트웨어 개발이 제대로 되고 있는지 확인할 수 있다.

2.3.2 프로토타입의 정의

프로토타입(Prototype)은 개발자와 사용자간의 요구사항을 분명하게 표현하고, 애매모호함과 불완전한 것을 더욱더 현실적으로 만들고 요구 사항에 대한 개발자와 사용자 간에 이해의 격차를 줄여준다[5]. 프로토타입은 일상적으로 시작품으로 불리며, 사용자 앞에 가상 시스템 또는 초기 일부분을 보여주어서 사용자의 사고를 촉진시키고 요구사항에 대한 개발자와 사용자 간의 대화를 원활하게 해주며, 실제로 구현하여 평가해 볼 수 있는 도구를 말한다[4].

2.3.3 프로토타입의 분류

프로토타입은 여러 산업분야에서 다양하게 활용되고 있다. 건축이나 토목과 같은 정적인 설계와 건축모형에서 뿐만 아니라 선박 건조 시에도 모형을 만들어 시험하는 것은 물론이고 자동차의 신 모델을 출시할 때는 필수적으로 컨셉트 카(Concept car)를 통해 소비자의 경향과 요구사항을 파악하고, 충분히 시험을 해 본 후에야 양산 플랜트를 구축하고 있다. 소프트웨어 개발 과정에서도 다양한 형태의 프로토타입 기법이 활용되고 있으며, 페이퍼 프로토타입(Paper prototype)이 대표적인 예라고 할 수 있다[6].

페이퍼 프로토타입은 융통성이 가장 좋은 표현 방법이다. 단순히 필기도

구뿐만 아니라 여러 가지 색과 모양을 가진 물건을 활용하여 버튼, 아이콘, 메뉴, 등을 나타낼 수 있다. 종이는 웹 인터페이스에 관련된 모든 것은 물론이요, 인터랙션을 표현해내는 데에도 가장 이상적인 매체다[7]. 페이퍼 프로토타입은 현업에서 가장 많이 활용되고 있으며, 최근 수십 년간 학계를 비롯하여 여러 기업에서 각광을 받고 있다.

페이퍼 프로토타입의 다음과 같은 장점을 가지고 있다. 페이퍼 프로토타입의[8] '종이'는 소프트웨어, 제품, 특히 모바일과 제스처를 이용한 인터랙션을 표현해 내는 데 적합한 몇 안 되는 매체이다. 어떤 인터랙션을 상상하든 그 아이디어를 종이로 된 프로토타입으로 옮기는 데는 채 몇 분 걸리지 않아 신속하게 제작이 가능하다.

페이퍼 프로토타입은 특별한 제작 프로그램이 필요치 않기 때문에 별도의 소프트웨어 라이선스가 필요 없다. 집이나 사무실에 앉아서 주변에 있는 모든 필기를 자유로이 사용할 수 있고, 요구사항의[9] 지속적인 변화에서 매우 빠르게 대처할 수 있고 쉽게 편집이 가능하다.

널리 쓰이고 있고 있는 페이퍼 프로토타입 역시 약점과 한계를 가지고 있다. 페이퍼 프로토타입은 멀리 떨어진 원격 작업에는 적합하지 못하며, 상호관계가 복잡한 경우 정확하게 표현하기 어려움과 실제 프로그램이 아니므로, 상상력을 동원하여 프로그램 실행 상태를 머릿속에 그려내야 하는 한계가 있다. 본 논문에서는 이러한 부분들을 보완하기 위한 방법을 제시한다.

프로토타입은 유형의 형태에 따라 수평적 프로토타입, 수직적 프로토타입으로 분류 할 수 있다. 각각의 정의와 특성은 다음과 같다.

1) 수평적 프로토타입

일반적으로 "소프트웨어 프로토타입"이라고 하는 경우 일반적으로 수평

적(Horizontal)을 의미한다. 수평적 프로토타입은 행동적 프로토타입(Behavioral prototype), 또는 모형(Mock-up)이라고도 한다. 이는 아키텍처의 모든 계층 또는 시스템 상세내용은 진행하지 않고 주로 사용자 인터페이스의 일부를 묘사하기 때문이다. 이런 유형의 프로토타입은 요구사항을 정교하게 만드는 동시에 의도한 시스템의 특정 동작을 조사할 수 있게 한다.

영화 세트와 같이 수평적 프로토타입은 실제로 기능을 구현하지 않고도 기능에 대해 상상할 수 있게 한다. 이것은 사용자 인터페이스 화면의 외관을 보여주고 화면 간을 이동할 수 있게 하지만 기능은 거의 가지고 있지 않다.

수평적 프로토타입은 사용할 기능 옵션, 사용자 인터페이스의 외관과 정보 아키텍처를 보여주어 유용한 작업을 하는 것처럼 보이지만 실제로는 어떤 작업도 하지 않는다. 사용자가 어떤 기능이 누락되었는지, 불필요한지를 판단할 수 있게만 하면 그것으로 수평적 프로토타입의 역할로 충분하다.

2) 수직적 프로토타입

수직적(Vertical) 프로토타입은 구조적 프로토타입, 또는 개념 검증(Proof of concept) 이라고 하며, 기술 서비스 계층을 통해 사용자 인터페이스에서 애플리케이션 기능의 일부를 구현한다. 수직적 프로토타입은 시스템 구현의 모든 수준을 다루고 있기 때문에 실제 시스템이 동작하는 것과 같이 동작하며, 제안된 아키텍처가 합리적이고 문제가 없는지 불확실한 경우 또는 알고리즘을 최적화시키고 제안된 데이터베이스 스키마를 평가하고 중요한 요구사항을 테스트할 경우에 수직적 프로토타입을 개발한다. 운영환경과 비슷한 환경에서 운영 제품을 사용해서 만들어지며, 중요한 인터페이스와 요구사항을 조사하고 설계의 위험을 낮추기 위해 사용된다.

2.3.4 프로토타입의 장점

프로토타입을 만드는 가장 큰 이유는 개발 프로세스의 불확실성을 조기에 해결하려는 것이다. 프로토타입은 요구사항에서 애매모호함과 불완전한 것을 찾아내고 해결하는 데 도움이 된다. 이러한 프로토타입은 다음과 같은 목적에 도움이 된다. 첫 번째로 프로토타입은 이해되지 않는 시스템의 일부분을 먼저 구현하는 것으로 이러한 과정을 통해 사용자는 프로토타입을 통해 요구사항과 관련된 문제를 지적하게 되고, 평가하게 되므로 사용자의 요구사항을 분명하고 완전하게 만든다. 두 번째는 설계 툴로 사용되는 프로토타입은 관련자들이 사용자와의 또 다른 상호작용 기법을 찾고, 시스템 유용성을 최적화시키고 잠재적인 기술 접근방법을 평가할 수 있게 한다.

프로토타입은 동작하는 설계를 통해 요구사항의 타당성을 입증할 수 있다. 구현의 툴로 사용되는 프로토타입은 제품의 일부를 초기에 기능적으로 구현한 것으로 작은 규모의 개발주기를 통해 완전한 제품으로 만들어져간다. 무엇보다 프로토타입을 만드는 가장 큰 이유는 개발 프로세스의 불확실성을 조기에 해결하려는 것이다. 이런 불확실성을 사용해서 시스템의 어느 부분을 프로토타입으로 만들고 프로토타입 평가에서 어떤 것을 알아 낼 것인지를 판단한다.

프로토타입은 요구사항에서 애매모호함과 불완전한 것을 찾아내고 해결하는 데 도움이 된다. 사용자, 관련자와 다른 관련자들은 제품의 명세가 만들어지고 설계되는 동안에 프로타입을 통해 구체적으로 생각할 수 있게 된다.

Ⅲ. 언어 프로토타입

이 장에서는 언어를 기반으로 하는 새로운 프로토타입 표현기법을 제안한다. 제안하는 표현기법은 문자나 기호로 구성된 언어 프로토타입이다.

언어 프로토타입에 대한 이해와 특징에 대해 소개하며, 그리고 표현규칙을 바탕으로 응용프로그램을 프로토타입으로 표현한다.

3.1 언어 프로토타입의 구성 요소

본 논문에서 제안하는 언어 프로토타입의 구성 요소는 객체(Object), 이벤트(Event), 주석(Comment)으로 구성되어 있다. 이러한 각각의 구성요소들은 규칙을 바탕으로 언어 프로토입으로 표현 된다.

언어 프로토타입의 구성 요소의 핵심은 바로 객체(Object)이다. 객체는 우리가 사는 생활 세계의 모든 물체들을 말한다. 즉, 자동차, 책상 등과 같이 눈에 보이는 모든 것을 의미한다.

이벤트(Event)는 객체가 외부의 자극에 대하여 반응하는 것을 의미하며, 이와 같은 이벤트 항목들에는 마우스 동작, 속성 값의 선택, 다른 화면으로 이동하거나 새 창으로 이동이 있다.

주석(Comments)은 해당 소스 코드의 의미를 보다 쉽게 전하기 위해서이다. 언어 프로토타입에서도 주석을 이용하여 프로토타입의 상세한 내용을 전달할 수 있다.

3.1.1 객체 표현

객체들은 하나의 독립된 형태로 존재 할 수 있고, 두 개의 객체가 서로 결합된 형태의 객체로 표현 될 수 있다. 하나의 기본적인 객체의 형태를 표현하기 위해 다음의 객체는 “nO[str:data]″의 형태와 같이 하나의 독립된 객체 형식을 정의할 수 있다. 여기서 n은 객체의 형제번호를 의미하고, O는 객체를 나타내는 객체의 약어가 들어간다. str은 객체명, data는 객체 데이터가 들어간다.

정의 1. 객체의 형제번호는 객체순서의 일련번호를 의미한다.■

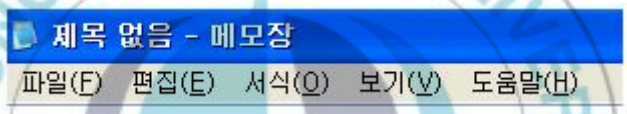

그림 3. 메모창의 메뉴 표시줄

예를 들어 그림3의 메뉴 표시줄의 객체의 형제번호는 객체들의 일렬의 순서대로 지정된다. 그림3의 경우 파일은 1에 해당하며, 편집은 2, 서식은 3, 보기는 4, 도움말은 5로 형제번호가 지정된다. 객체의 나열 형태가 가로형인 경우 왼쪽에서 오른쪽으로, 세로형인 경우 위에서 아래로 객체들의 형제번호가 지정된다.

정의 2. 객체들은 각각의 특성에 맞는 약어로 표현한다.■

예를 들어 Window의 경우 해당 객체를 나타낼 수 있는 'w'로 표기한다.

보조정의 2-1. 본 논문에서 거론하는 객체들은 <표 1>과 같이 표현한다.

보조정의 2-2. 새로운 객체들은 해당 객체의 특성에 맞는 약어로 지정하여 표현할 수 있다.

응용 프로그램에서 객체(Object)는 Object, Screen, Menu, Window, Group, Check Box, Text Box Button, Radio Button, List Box, Tab, Label, Table, Image등이 있다. 이러한 다양한 형태의 객체들은 객체의 특성에 맞는 약어로 표1과 같이 표현한다.

객체 이름	접두어
Object	o
Screen	s
Menu	m
Window	w
Group	g
Check Box	k
Text Box	x
Button	b
Radio Button	r
List Box	l
Tab	t
Label	v
Table	e
Image	i

<표 1> 객체 표기법

정의 3. str은 객체명이 들어간다. 없는 경우는 [:data]로 나타낸다.■

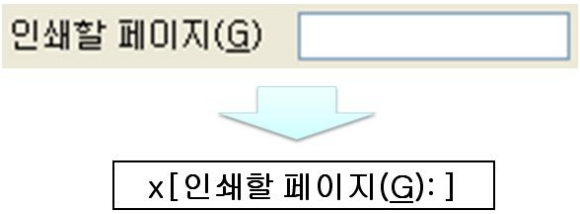

그림 4. 인쇄할 페이지 Text box

예를 들어 그림4와 같이 '인쇄할 페이지'라는 객체명을 가진 Text box를 정의3을 바탕으로 "x[인쇄할 페이지(G):]" 과 같이 표현 할 수 있다.

정의 4. data는 객체 데이터가 들어간다. 없는 경우는 [str]로 data가 미정이거나 공란이면 [str:]로 나타낸다. ■

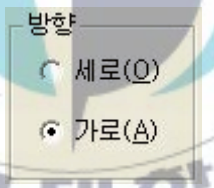

그림 5. 방향 Radio button

예를 들어 그림5와 같은 가로, 세로 데이터를 가지고 있는 Radio button을 정의 4를 바탕으로 "r[방향: 세로| 가로]" 과 같이 표현한다.

객체는 단독으로 존재 할 수 있지만 두 개 이상의 객체가 결합하여 하나의 객체로 표현할 수 있다.

규칙 1) 객체를 의미하는 각각의 문자들을 서로 결합하여 하나의 객체형태로 나타낸다. ■

그림 6.텍스트 박스와 레이블의 형태

예를 들어 레이블 객체는 텍스트 박스와 같은 다른 객체의 이름을 나타내주는 객체로서 다른 객체와 같이 사용되는 객체이다. 레이블 객체는 그림6과 같이 단독으로 사용되는 경우는 드물고, 폼에 놓여 진 어떤 컨트롤의 용도를 설명하기 위해서 그 옆에 문자열을 표기하는데 주로 사용된다.

이러한 경우 정의5를 바탕으로 객체를 의미하는 각각의 문자들을 서로 결합하여 그림7과 같이 표현 한다. 레이블 객체를 의미하는 v와 텍스트 박스 객체를 의미하는 x가 서로 결합하여 vx 로 표기하며, str 위치에는 객체명에 해당하는 “이름”, data 위치에는 객체의 data에 해당하는 “Text1”들어 간다.

그림 7.텍스트 박스객체와 레이블 객체의 결합 표현

3.1.2 이벤트 표현

대부분의 응용프로그램에서 파일을 불러오는 경우나 인쇄를 하는 경우에는 새로운 원도우창이 오픈하게 된다.

정의 5. 새로운 원도우창의 오픈은 다음과 같은 “→” 나타낸다.■

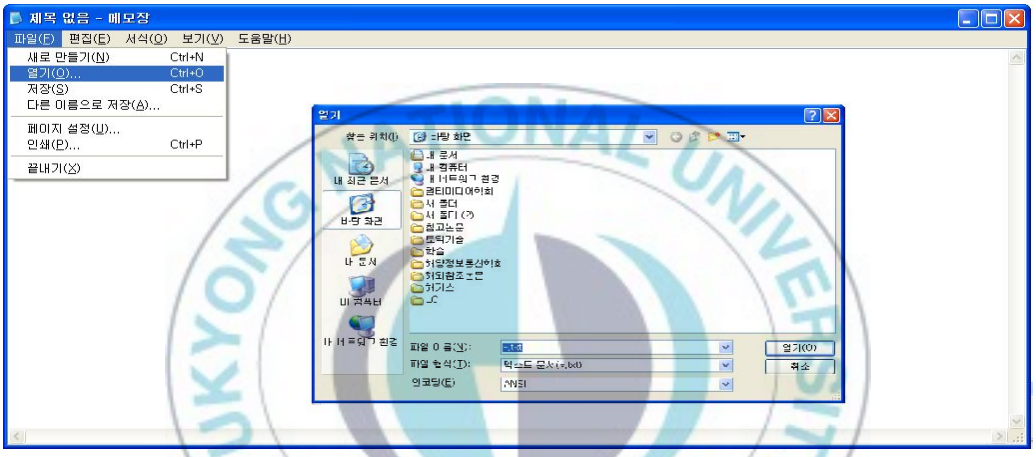

그림 8. 메모창의 원도우창 오픈

규칙 2) 원도우창의 오픈은 객체와 객체 사이에 “→”로 표기 한다.■

예를 들어 그림8의 메모창에서 파일을 찾을 경우 새로운 원도우창이 오픈한다. 이러한 새로운 원도우창의 오픈을 다음과 같이 “s[메모창]/1m[파일]/2m[열기.] → w[열기]” 표현할 수 있다.

“→” 기준으로 전자에 해당하는 객체는 마우스가 클릭하는 대상의 객체를 의미하며, 후자는 객체는 마우스 클릭으로 인해 새로 오픈하는 원도우창

의 객체를 의미한다.

마우스의 클릭 동작 또한 언어 프로토타입으로 표현할 수 있다. 마우스에서 클릭(Click)의 종료는 클릭과 더블클릭(Doubleclick) 두 가지로 구분할 수 있다.

클릭이라고 하면, 마우스의 단추를 한 번 누름, 또는 그런 행위를 의미한다. 더블클릭은 마우스의 단추를 연이어 두 번 누르는 것으로 주로 프로그램을 실행시키는 명령과 같은 기능을 한다. 일반적으로 더블클릭이라고 하면 마우스 왼쪽 버튼을 누르는 것을 의미한다.

정의 6. 마우스의 클릭의 동작은 "!(Exclamation mark)"로 표기하여 나타낸다. ■

예를 들어 "확인"이란 버튼의 마우스 클릭을 나타내기 위해서는 객체 옆에 Exclamation mark를 표기하여 다음과 같이 "!b[확인]" 표현할 수 있다.

마우스 왼쪽 버튼을 클릭하는 형태는 객체의 왼쪽에 '!' 표시 한다. 예를 들어 "확인"이란 버튼의 마우스 왼쪽 한번 클릭을 나타내기 위해서는 다음과 같이 "!b[확인]" 표현할 수 있다.

마우스 오른쪽 버튼을 클릭하는 형태는 객체의 오른쪽에 '!' 표시한다. 예를 들어 "확인"이란 버튼의 마우스 오른쪽 한번 클릭을 나타내기 위해서는 다음과 같이 "b[확인]!" 표현할 수 있다.

Exclamation mark가 하나 있으면 한 번 클릭, 두 개 있으면 더블 클릭

을 의미한다. 예를 들어 “확인”이란 버튼의 마우스 한 번 클릭은 다음과 같이 “!b[확인]” 표현하며, 마우스 더블 클릭은 다음과 같이 “!!b[확인]” 표현한다.

라디오 버튼은 체크박스와 비슷하나 하나만 선택할 수 있다는 특징을 가지고 있다. 마우스를 이용하여 하나의 속성 값을 선택한 경우 선택된 속성값 밑에 다음과 같은 “__” 밑줄 기호로서 해당 속성값의 선택을 나타낸다.

정의 7. “__” 밑줄 기호로서 default값을 나타낸다.■

예를 들어 회원가입 창에서 남자와 여자 성별을 선택해야하는 경우 “__” 밑줄 기호로서 다음과 같이 “r[남자|여자]” 표현할 수 있다.

3.1.3 주석 표현

주석문(Comments)은 언어 프로토타입으로 표현된 프로토타입의 설명글이다. 주석문을 삽입하여 프로젝트를 참여하는 모든 관계자들에게 언어 프로토타입을 좀 더 읽기 쉽고 이해하기 쉽도록 만드는 것이다.

정의 8. 주석은 *(Asterisk)로 시작한다.■

Asterisk 이후의 한 라인이 주석으로 처리된다. 여러 라인의 주석처리는 처음과 끝에 /*과 */를 붙여서 여러 줄의 주석을 작성할 수 있다.

예로 그림9와 같이 주석문을 삽입하면 보다 상세한 내용을 전달할 수 있다.

```
S[DA# Modeler]/1m[파일]/4m[저장] * 작업 중인 파일을 저장
```

그림9. 주석 표기의 예

3.2 언어 표현 규칙

언어 프로토타입은 객체, 이벤트, 주석과 같은 구성요소들이 결합과 흐름을 바탕으로 표현된다. 결합과 흐름에는 규칙이 존재한다.

규칙 3) 언어 프로토타입의 문장의 흐름은 왼쪽에서 오른쪽으로, 위에 서 아래로 표현한다. ■

객체들의 배열은 규칙3을 바탕으로 왼쪽에서 오른쪽으로 위에서 아래쪽으로 표현한다.

두 개 이상의 독립된 형태의 객체들이 존재하는 경우 동일한 레벨에서 객체들의 순서를 나열할 때 규칙 4를 바탕으로 객체들을 나열할 수 있다.

규칙 4) 동일한 레벨에서의 객체 순서를 표현할 때 객체와 객체 사이에 ,(Comma)를 기준으로 표현한다. ■

예를 들어 그림10과 같은 메모창의 메뉴 표시줄을 표현할 때 메모창의 메뉴 표시줄에 위치한 메뉴들은 동일한 레벨에 위치하므로, 연속하여 나열되는 객체들은 객체와 객체사이에 comma를 기준으로 표현한다. 메모창의 메

뉴 표시줄을 다음과 같이 “1m[파일],2m[편집],3m[서식],4m[보기],5m[도움말]” 표현된다.

상위 메뉴와 하위 메뉴와의 연결을 표현할 때 규칙 3을 바탕으로 표현한다.

규칙 5) 상위 객체와 하위 객체의 연결의 표현할 때 객체와 객체 사이에 /(Slash)를 기준으로 표현한다. ■

예를 들어 그림10과 같이 메모창의 파일 메뉴와 하위 메뉴에 해당하는 인쇄 메뉴항목을 표현하기 위해서는 규칙5를 바탕으로 상위 객체와 하위 객체를 연결할 수 있다. 상위객체와 하위 객체 사이에 slash를 기준으로 다음과 같이 “1m[파일]/6m[인쇄]” 표현된다.

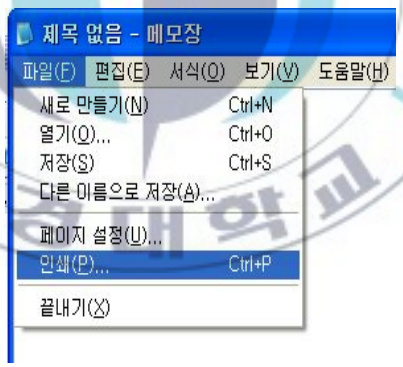

그림10. 메모창의 메뉴 표시줄

동일한 레벨에서 객체가 많은 경우 언어 프로토타입 문장이 길어질 수 있다. 이런 경우 이전 줄과 연속임을 의미하기 위하여 위해서 규칙6을 바탕으로 표현할 수 있다.

규칙 6) 현재 줄은 이전 줄과의 연속임을 표현할 때 객체와 객체 사이 에 ~(Swung Dash)를 기준으로 표현한다. ■

그림11. DA# Modeler의 보기메뉴

예를 들어 그림11과 같이 상위메뉴와 하위메뉴 객체들이 많은 경우 규칙 6을 바탕으로 그림12와 같이 표현된다.

```
3m[보기(V)]/1m[작업관리창],2m[전체레이아웃],
~ 3m[실체확대보기],4m[Note미리보기],
~ 5m[속성상속정보],6m[도구모음(T)],
~7m[창크기에맞추기],8m[실제크기],
~ 9m[보기비율],10m[모델Level],
~11m[동의어표시],12m[식별자상속처리]
~ 13m[관계명나타내기],14m[관계동사나타내기]
~ 15m[관계속성나타내기],16m[물리모델Align보기],
~ 17m[보조명],18m[모델정보나타내기],19m[실체찾기]
```

그림 12. DA# Modeler의 보기메뉴의 언어 프로토타입 표현

radio button과 list box와 같은 경우 세부적인 속성 값을 가지고 있으며 이러한 속성 값을 표현하기 위해서는 규칙7을 바탕으로 표현한다.

규칙 7) 속성 목록을 나열하기 위해서 속성과 속성 사이에 ‘ | ’ 기호로 속성을 나열한다. ■

예를 들어 그림13의 radio button의 Password, Externally, Globally 속성을 ‘ | ’ 기호를 기준으로 다음과 같이 표현 할 수 있다. “ r[Type: Password | Externally | Globally] ”

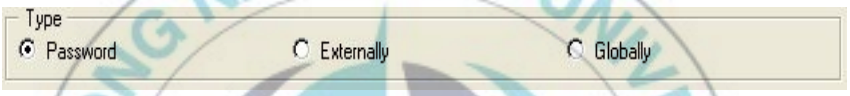

그림 13. Radio button

Radio button과 Check box는 비슷하나 하나를 선택할 수 있다는 특징을 가지고 있다. 마우스를 이용하여 하나의 속성 값을 선택한 경우 규칙8을 바탕으로 표현할 수 있다.

규칙 8) 선택된 속성 값 밑에 다음과 같은 “__” 밑줄 기호로서 해당 속성값의 선택을 나타낸다. ■

예를 들어 회원가입 창에서 남자와 여자 성별을 선택해야하는 경우 다음과 같이 “r[남자|여자]” 표현할 수 있다.

3.3 응용프로그램 프로토타입의 표현

응용프로그램(Application program)은 특정한 업무를 해결하기 위한 목적을 가지고 만들어진 프로그램을 의미한다. 윈도우즈 응용프로그램들은 공통적으로 그림14와 같은 아이콘, 제목을 나타내는 타이틀 바, 창의 크기를 제어하는 아이콘 표시 버튼, 전체화면 표시 버튼, 창 닫기 버튼, 메뉴, 도구모음, 상태 바 등으로 구성되어 있다.

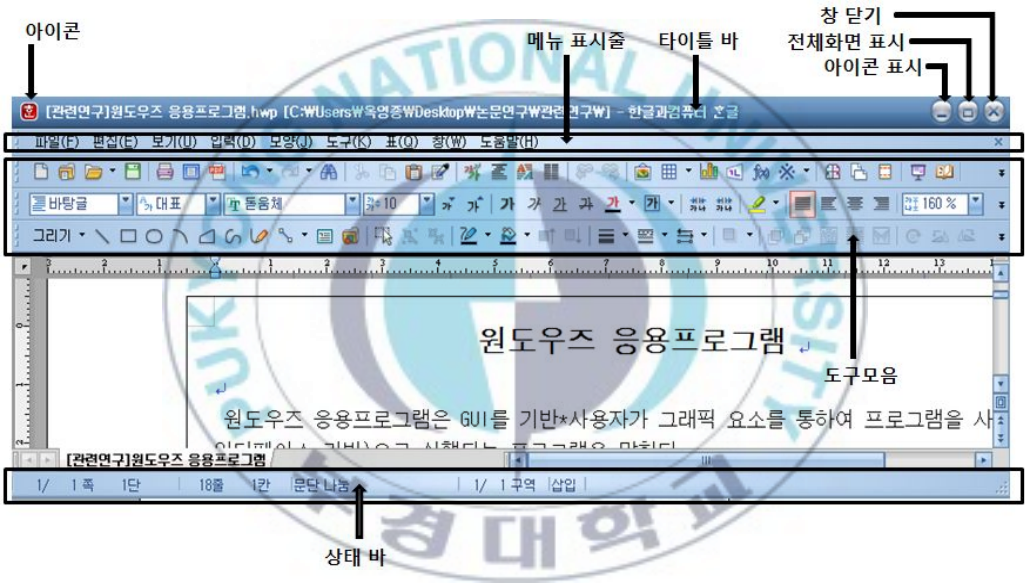

그림 14. 응용프로그램의 구성 요소

본 논문에서 제안하는 언어 프로로토타입은 언어를 기반으로 하여 원도우즈 응용프로그램의 Main menu, Sub menu, 페이지 설정 창, 동작의 흐름을 언어를 기반으로 하여 언어 프로토타입을 표현 할 수 있다.

1) Main menu의 표현 예시

응용프로그램의 메뉴 표시줄의 Main menu를 언어 표현 기법을 바탕으로 언어 프로토타입으로 표현할 수 있다.

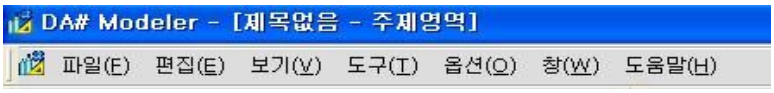

그림 15. DA# Modeler의 Main menu

그림 15. DA# Modeler의 Main menu에는 상위객체에 해당하는 DA# Modeler와 하위객체에 해당하는 메뉴표시줄이 있다. 상위객체와 하위객체의 연결은 정의된 규칙3을 바탕으로 두 객체간의 연결을 표현할 수 있다. 메뉴표시줄에 위치한 객체들은 동일한 레벨에 위치하므로 정의된 규칙2를 바탕으로 객체들을 나열할 수 있다. 이러한 규칙을 바탕으로 그림16과 같이 DA# Modeler의 Main menu를 언어 프로토타입을 표현 할 수 있다.

```
S[DA# Modeler]/1m[파일],2m[편집],3m[보기],4m[도구],5m[옵션],6m[창],7m[도움말]
```

그림 16. Main menu의 언어 프로토타입 표현

2) Sub menu의 표현 예시

그림17. DA# Modeler의 Sub menu

그림 17의 sub menu메뉴의 항목을 상위메뉴와 하위메뉴를 연결은 정의된 규칙5를 바탕으로 객체간의 연결을 표현할 수 있다. 하위메뉴의 객체들은 동일한 레벨에 위치하므로 위에서 아래의 순으로 순차적으로 표현한다. 이러한 규칙을 바탕으로 그림 18와 같이 언어 프로토타입으로 표현할 수 있다.

```
S[DA#Modeler]/1m[파일]/1m[새로만들기],2m[열기],3m[닫 기],~
4m[저장],5m[다른 이름으로 저장...],6m[Import],7m[Export],~
8m[모델신청],9m[A/R모델 올리기]],10m[A/R모델 내려받기]],~
11m[DA# A/R 연결],12m[인쇄...]13m[인쇄미리보기]],~
14m[페이지 설정],15m[모델정보],16m[종료]
```

그림18. 파일메뉴 속성 언어 프로토타입

3) 페이지 설정 창 표현 예시

대부분의 응용프로그램에서 인쇄항목을 선택하는 경우 새로운 원도우창이 오픈하게 된다. 그림19와 같은 페이지 설정 원도우창인 경우에도 그림 20과 같이 언어 프로토타입으로 표현할 수 있다.

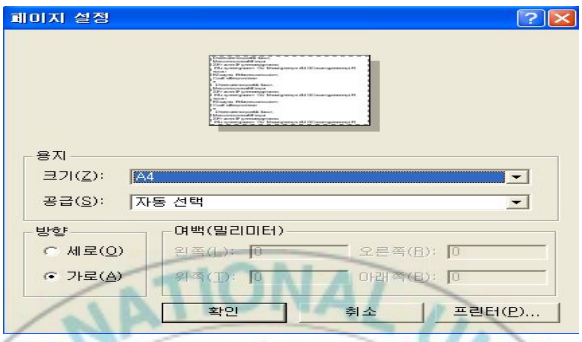

그림 19. 페이지 설정 원도우창

언어 프로토타입은 화면의 표현 뿐 만 아니라 시각적 표현기법으로 표현하기 어려운 세부적인 항목까지도 표현할 수 있다. 그림 19의 경우 크기와 공급의 콤보박스의 경우 세부적인 속성 값을 가지고 있다. 하지만 시각적 표현 기법으로는 이러한 세부적인 부분까지 표현하기 어려움이 있다.

화면에 보이지 않는 콤보박스의 세부 항목까지도 정의된 규칙5를 바탕으로 속성 목록을 나열할 수 있고, 선택된 속성 값의 선택은 밑줄로서 나타낸다. 이렇게 표현된 언어 프로토타입은 그림 20과 같이 나타낼 수 있다.

```
w[페이지 설정]
 0/1g[용지]
        1/1c[크기: A4(210x297mm)| sizes]
        1/2c[공급: 자동| 공급값들]
 0/2r[방향: 세로|가로]
 0/3g[여백(밀리미터): x[왼쪽(L):], x[오른쪽(L):], x[위쪽(L):], x[아래쪽(L):]]
 0/4b[확인], 0/5b[취소], 0/7b[프린터...]
```

그림 20. 페이지 설정 창 언어 프로토타입 표현

4) 응용프로그램 동작의 흐름 표현

원도우 프로그램의 특징은 한마디로 이벤트 중심의 프로그램이다. 원도우 프로그램이 이벤트를 중심으로 작동한다는 것이다. 프로그램이 실행된 상태에서 화면에 버튼을 누르기 전까지는 어떠한 동작도 하지 않고 있다가, 버튼이 눌렀을 때만 작동하는 것을 이벤트 중심의 프로그램이라고 한다.

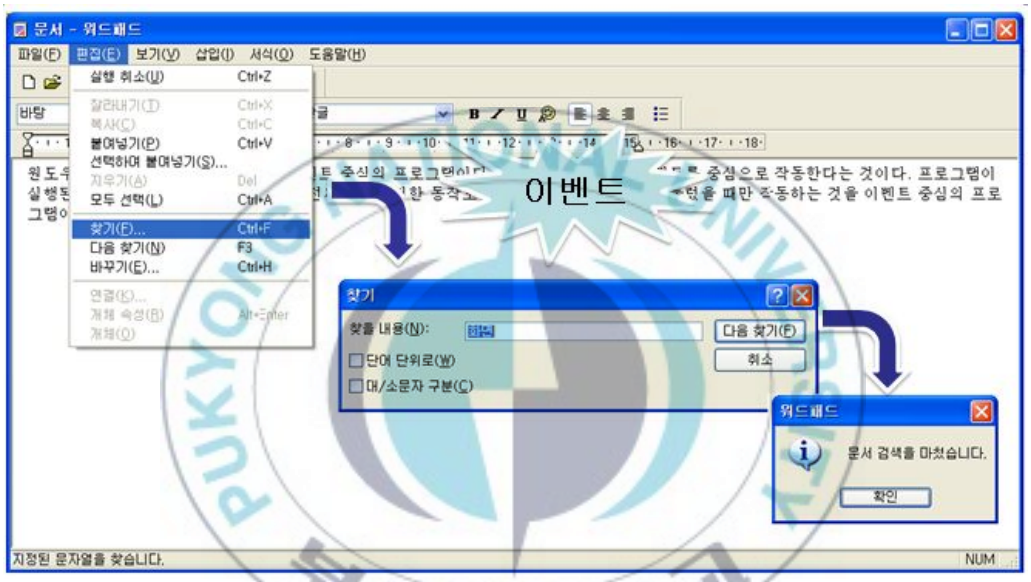

그림 21. 응용프로그램 동작의 흐름

그림21은 워드패드에서 단어를 검색하는 절차의 흐름을 표현한 것이다. 이와 같은 동작의 순차적인 흐름까지도 그림 22와 같이 언어 프로토타입으로 표현할 수 있다.

```
S[워드패드]/2m[편집]/8m[찾기]→w[찾기],vx[찾을내용(N):화면],b[다음찾기]→w[워드패드]
```

그림 22. 응용프로그램 동작의 흐름 언어 표현

3.4 언어 표현 기법의 문서화

시스템 개발 시 문서화 작업은 개발 과정 중 가장 중요한 부분이다. 문서화는 시스템 개발의 모든 단계에 걸쳐 수행된다. 대부분의 소프트웨어는 변화에 잘 적응하지 못하므로 이러한 문제점들을 해결하기 위해서는 개발 초기부터 각 개발 단계에 대한 체계적인 문서화 계획을 수립하고 실행해야 한다[7].

시스템이 개발되었다. 하더라도 이를 적절하게 운용하기 위해서는 사용자 지침서와 같은 사용자 문서가 필요하다. 프로젝트의 이해관계자가 요구사항을 명확히 이해하고 이를 통해 대화할 수 있도록 하기 위해서는 프로토타입을 문서로 기술해야 한다.

소프트웨어 프로젝트 개발과정에서 각각의 화면들을 언어로서 프로토타입으로 표현할 때 표현된 프로토타입은 프로젝트 개발과정의 문서화로 이어질 수 있다.

언어 프로토타입의 문서화는 규범적(Prescriptive)이면서 동시에 설명적(Descriptive)이다[10]. 다시 말해 규칙을 바탕으로 표현되어 있으며, 동시에 기호와 구성요소들은 각각의 의미를 가지고 있다. 이것은 의사 결정을 내리려 하는 사람들의 입장에서는 고려해야 할 제약사항이 명시돼 있어서 어떤 선택을 해야 하는지 미리 알 수 있고, 시스템을 파악하는 사람들의 입장에서 보면 시스템 설계에 대한 결정사항들이 하나씩 설명돼 있어서 시스템에 대해 올바로 이해할 수 있다.

소프트웨어 개발과정의 문서화할 때 가장 기본적인 규칙은 사용자의 시점으로 문서를 작성해야 한다는 것이다. 쓰기는 쉽고 읽기는 어려운 문서는 쓸모가 없다. '읽기 쉬운' 문서란 문서를 읽는 사람에 따라 동일한 내용으로 인지해야 한다는 것이다. 언어 표현 기법을 활용하면 이런 문서를 쉽고 유

용한 문서로 만드는 데 도움이 된다. 프로토타입의 주요 목적은 이해관계자들 사이에 의사소통 수단을 제공하는 것이 있다고 했다. 프로토타입의 문서화를 하면 이런 의사소통이 쉬워진다.

이해관계자를 오랜 기간 동안 함께 온 사람인지 또는 새로 온 사람인지를 나누어 생각해 볼 때 새로 온 이해관계자가 원하는 정보는 내용면에서는 오래된 이해관계자가 원하는 것과 비슷하겠지만 형식면에서는 적은 분량의 소개 정도를 기대한다는 점이 다를 것이다. 프로토타입의 문서화는 새로운 개발자나 사용자 등 개략적인 내용을 원하는 사람들을 교육시킬 때 요긴하게 사용할 수 있다. 프로토타입의 문서화는 이해 관계자들 간의 의사소통의 도구 뿐 만 아니라 소프트웨어 개발과정 전체의 품질향상에 도움이 된다.

Ⅳ. 언어 표현기법을 적용한 설계 및 분석

이 장에서는 본 논문에서 제안하는 언어 표현기법을 바탕으로 제작된 프로토타입으로 실제 화면 설계에 적용하였다. 분석에서는 동일한 형태의 표현기법이 없으므로 기존에 많이 활용되고 있는 시각적인 표현기법과 본 논문에서 제안하는 언어 표현기법에 대해 분석하였다.

4.1 언어 표현 기법을 적용한 설계

언어 표현 기법을 적용한 프로토타입을 바탕으로 실제 화면을 다음과 같은 결과로 구현할 수 있다. 일반적으로 회원가입 창의 경우 테이블형태의 양식 사용한다. 일반적인 회원가입 요구사항 명세인 경우 성명, 회원ID 비밀번호, 비밀번호 확인, 성별, 취미 등의 회원정보를 요구하며 마지막으로 회원등록, 다시작성, 취소하기를 버튼 컨트롤을 이용하여 처리한다.

이러한 요구사항의 정보들은 맨 위에서 아래로 순차적으로 표현하고, 동일한 행에서는 데이터는 왼쪽에서 오른쪽으로 표현할 수 있으며 첫 번째 성명을 입력하는 동일한 행에 사용되는 객체들은 label과 text box이므로 다음과 같이 “1e[:1v[성명],2x[:]]” 표현할 수 있다. 두 번째 회원ID인 경우에도 테이블의 형제번호가 두 번째이므로 “2e”로 표현하며 동일한 행에서도 “2e[: 1v[회원 ID], 2x[:]]” 와 같이 표현할 수 있다. 세 번째 비밀번호도 동일하게 “3e[: 1v[비밀 번호], 2x[:]]”로 표현할 수 있다. 네 번째 비밀번호 확인도 동일하게 “4e[: 1v[비밀 번호 확인], 2x[:]]” 다섯 번째 성별인 경우에는 남자와 여자 둘 중 하나이므로 radio button을 사용하여, 다음과 같이 “5e[: 1v[성별], 2r[: 남 | 여]]” 표현한다. 여섯 번째 취미인 경우 다양한 취미

선택을 할 수 있도록 check box를 사용하여 다음과 같이 “6e[:1x[취미], 2k[수영],3k[영화],4k[음악],5k[여행],6k[게임]]” 표현한다. 마지막으로 회원등록, 다시작성, 취소하기는 button을 사용하여 다음과 같이 “7e[:1b[회원등록], 2b[다시작성], 3b[취소하기]]” 표현한다. 이러한 행들의 표현은 그림 23과 같이 언어 프로토타입으로 표현된다.

```
1e[:1v[성명],2x[:]]
2e[:1v[회원ID],2x[:]]
3e[:1v[비밀번호],2x[:]]
4e[:1v[비밀번호확인],2x[:]]
5e[:1v[성별],2r[:남|여]]
6e[:1x[취미],2k[수영],3k[영화],4k[음악],5k[여행],6k[게임]]
7e[:1b[회원등록],2b[다시작성],3b[취소하기]]
```

그림 23. 언어 표현 기법 적용 설계

그림23과 같이 표현된 언어 프로토타입을 바탕으로 그림 24와 같은 회원가입 화면 창을 구현할 수 있다.

그림 24. 회원가입 화면 창

4.2 기존 표현기법과 제안기법의 비교분석

본 장에서는 기존의 프로토타입 표현 기법과 본 논문에서 제안하는 언어 표현기법의 특징들을 비교 분석하였다. 언어 표현기법과 시각적 표현 기법을 바탕으로 표현기법, 장·단점을 비교 분석하였다. 먼저 프로토타입의 표현 기법으로 시각적 표현기법은 이미지를 바탕으로 디자인적인 측면을 강조하고 있는 하여, 누구나 쉽게 프로토타입을 인식할 수 있는 장점을 가지고 있다.

하지만 시간적인 표현기법은 정적인 이미지를 강조하므로, 동적인 부분을 표현하는데 어려움과 상호관계가 복잡한 요구사항의 표현에 어려운 단점을 가지고 있다. 언어적 표현기법은 언어와 기호를 바탕으로 표현되며, 언어를 기반으로 복잡한 상호관계를 보다 효과적으로 표현 가능하며, 정적인 것뿐만 아니라 정적인 화면의 흐름까지도 언어로서 표현이 가능하다. 다만 언어와 기호를 바탕으로 표현되므로 미리 정의된 언어와 기호에 대한 인지가 필요하다. 시각적 표현기법과 언어적 표현기법의 특징에 대한 비교분석은 다음 <표 2>와 같다.

	시각적 표현기법	언어적 표현기법
표현 기법	시각적 요소	언어 기호
장점	쉽게 프로토타입을 인식 할 수 있음	세부적인 내용 표현 및 화면의 동작이 표현가능
문제점	상호관계가 복잡한 경우 표현하기 어려움	객체 표기법 및 언어표현 방법 인지 필요

<표 2> 시각적 표현기법과 언어적 표현기법의 비교

4.3 제안 기법의 분석 결과

본 논문에서 제안하는 언어 프로토타입은 다음과 같은 특징들을 가지고 있다. 가장 큰 특징으로 언어를 기반으로 하고 있다는 사실이다. 이것은 복잡한 상호관계의 요구사항도 언어를 바탕으로 하여 상세히 표현이 가능하다. 언어를 기반으로 하는 것이므로 요구사항을 말하는 것을 그대로 표현할 수 있고, 표현된 것을 그대로 말할 수 있다. 이것은 소프트웨어 프로젝트 진행과정에서 요구사항의 지속적인 변화에도 신속하게 대응할 수 있다는 것이다.

언어 프로토타입은 소프트웨어 시스템의 산출물의 가시화, 명세화, 구축, 문서화하는 데 사용될 수 있다. 소프트웨어의 요구 사항을 언어 형태로 작성하며, 작성된 표기법에 있어서는 명확한 정의가 존재한다. 따라서 개발자들 사이에 오류 없는 원활한 의사소통이 이루어질 수 있다.

언어 프로토타입은 소프트웨어 개발 과정의 분석, 설계, 구현 단계의 각 과정에서 필요한 모델을 정확하고, 완전하게 명세화할 수 있는 명세언어로서 활용이 가능하다. 명세화된 모델은 프로그램 소스 코드로 변환하여 구축할 수 있다. 또한 이미 구축되어 있는 소스 코드를 언어 프로토타입으로 역변환 하여 분석하는 역공학(Reverse Engineering)이 가능하다.

각각의 화면 단위를 언어 프로토타입을 표현한다면 상세한 내역에 대한 문서화로 이루어지며, 시스템을 테스트하는 언어제공 및 시스템 구축 후 유지보수 활용에 도움이 된다.

Ⅴ. 결론

프로토타입은 사용자의 요구사항을 가장 잘 표현할 수 있는 도구이다. 소프트웨어 프로젝트의 주요 실패의 원인은 사용자와 개발자간의 비효율적인 의사소통으로 인한 것이므로 프로토타입이란 도구를 통해 사용자와 개발자 사이의 이해의 격차를 줄여준다. 페이퍼를 이용한 프로토타입, PPT를 활용한 프로토타입 등의 기법들이 있다. 하지만 이러한 기존의 프로토타입 표현 기법들은 시각적인 표현에만 의존하고 있다는 사실이다.

페이퍼 프로토타입과 같이 시각적인 표현 기법에 의존하고 있는 프로토타입 기법들은 상호관계가 많은 경우에는 표현하기가 복잡해지며, 세부적인 동작의 흐름을 표현하는데 어려움과 표준화 되지 못한 표현 기법들도 인해 여러 사람들에게 공유하기가 어렵다는 단점이 있다.

시각적 표현 기법에 속하는 프로토타입들은 시각적인 디자인 측면에 너무 치중한 나머지 근본적인 문제의 본질이 숨겨질 수 있다. 따라서 본 논문에서는 요구사항의 상세한 표현과 세부적인 동작의 흐름을 효과적으로 표현하기 위해 언어적인 표현 기법을 제안한다.

제안하는 표현 기법은 정형적인 언어(formal language)로서 규칙을 바탕으로 문자나 부호로 구성된 언어 기반 프로토타입 표현기법이다. 이렇게 규칙을 바탕으로 표현된 언어 프로토타입은 요구사항을 문서화 할 수 있고, 프로토타입 표현의 형식을 규칙화 표준화된 문서화가 가능 하며, 프로젝트에 참여하는 이해 관계자들 간의 의사소통의 도구로서 활용될 수 있다. 각각의 개발 화면들을 언어로서 프로토타입을 표현한다면 또 하나의 소프트웨어 개발 언어(Language)로서 발전 가능하며, 표현된 언어들은 점차 소프트웨어 개발 향상에 도움이 될 것이라 기대된다.

참고문헌

[1] Sung-Kon Kim,"A Study on the Application of Prototype in Graphic User Interface Development Process", Korean Society of Design Science, No.37, pp.181-190, 2000

[2] Kang Zhang, Guang-Lei Song, Jun Kong, "Rapid Software Prototyping Using Visual Language Techniques", June 2004

[3] Todd Zaki, warfel, Prototping , insight, 2011

[4] 권기태, "소프트웨어 공학", 홍릉과학출판사, 2006

[5] 프로젝트 성공 잠재력 기준표, Standish Group, 2006

[6] 송영재, 김귀정, 변정우, 서영준, 최한용, 한정수, "객체지향 모델과 CDB 중심 소프트웨어공학", 이한출판사, 2004

[7] Min-young Cho, Won-sup Kim,"A Study on Prototyping Tools for Interaction Design of Product", Korean Society of Design Science, No.58, pp.119-128, 2006

[8] http://jibjung.egloos.com/21289

[9] 김성곤, "그래픽 사용자 인터페이스 개발 프로세스에서의 프로토타입 활동에 관한 연구", 한국디자인연구학회 디자인학연구, 2000

[10] Karl E. Wiegers, "Software Requirements Second Edition", July, 2003

[11] 렌 베스, 폴 클레먼츠, "릭 캐즈먼. 소프트웨어 아키텍처 이론과 실제", 에이콘, 2007

[12] 김희천, "문서화 지원을 위한 CASE 도구의 설계 및 구현," 한국정보과학회 정보과학회논문지, Vol 3, No. 1, pp. 37-50, 1997

[13]최정은, 최순규, 이선아 "소프트웨어 요구사항 관리 사례 연구"
한국정보과학회 학술발표논문지, Vol 29, No. 1
[14] 채흥석, 객체지향 CBD 개발 Bible, 한빛미디어, 2003

감사의 글

“꿈이 있어 도전할 수 있었고 미래에 대한 희망이 있어 행복합니다.”

인생에서 가장 보람되고, 한 사람을 변화 시킬 수 있는 좋은 선생님이 되기 위해 교육대학원의 진학하면서 지식적인 면이나 내적인 면에서 예전에 알지 못했던 많은 것들을 경험하고 배웠습니다. 지난 3년 이라는 시간을 돌아보며, 본 논문을 완성하기까지 많은 분들의 도움과 격려가 있었습니다. 아직 부족함이 많음을 느끼며, 작은 결실이나마 얻을 수 있기까지 도움을 주신 모든 분들께 작게나마 감사의 마음을 전하고 싶습니다.

항상 열정적이신 모습과 많은 가르침 그리고 조언과 격려로써 저의 부족함을 채워주시고, 저의 부족한 논문을 지도해 주신 여정모 교수님께 진심으로 감사드립니다. 또한, 바쁘신 중에도 저의 부족한 논문을 심사해 주시고 보다 좋은 논문이 될 수 있도록 관심과 지도를 해주신 조경연 교수님, 서경룡 교수님께 감사드립니다. 그리고 많은 가르침을 주시고 배움의 길로 인도하여 주셨던 박만곤 교수님, 박승섭 교수님, 윤성대 교수님, 김창수 교수님, 김영봉 교수님께도 감사를 드립니다. 그리고 교직과목에 교사로서 바른 가르침을 가르쳐 주신 원효헌 교수님, 김상곤 교수님께도 깊은 감사의 마음을 전합니다.

비록 짧은 교직생활 이지만 좋은 선생님이 될 수 있도록 격려해주시고 도와주신 부산남중 김형욱 교장선생님, 부산혜송학교 조순화 교감선생님, 분포중 박윤희 선생님, 용문중 김민성 선생님, 부산중 이윤수 선생님, 선화여중 이지연 선생님께 깊은 감사의 마음을 전합니다.

논문을 작성 할 수 있도록 많은 배려와 논문에 많은 관심을 가져 주신 SK C&C 금융사업본부 서진수 부장님, 최원준 차장님, 알투웨어 금융사업팀 우재호 이사님 그리고 함께 근무했던 선우에게도 감사의 마음을 전합니다.

짧지 않은 연구실 생활동안 누구보다도 항상 함께 하고, 도움을 주고, 함께 졸업하자고 격려해주신 박선이 박사님, 그동안 많은 교직과목을 같이 수강하고 옆에서 힘이 되어준 심혜정 선생님, 지난 시간들을 나를 믿고 함께 해주었던 데이터베이스 연구실 도유, 유신, 경우, 광중, 종준, 세한, 그리고 막내 성현이 그리고 Datapia 가족들 모든 식구들에게 감사의 마음을 전합니다. 그리고 교직 교과목에서 조별활동으로 함께했던 국어전공 김하연 선생님께도 감사의 마음을 전합니다.

저의 영원한 멘토 현석이형에게 깊은 감사를 전하고 싶습니다. 행복할 때나 힘들 때나, 항상 같이 저의 곁에서 사랑과 지혜로써 힘이 되어준 현석이형이 함께 있음이 저에겐 가장 큰 축복이었습니다. 그리고 멀리 서울에서 전공 지식에 많은 조언과 도움을 주신 재천이형에게도 감사의 마음을 전하며, 재천이형 가정에 언제나 축복이 가득하길 기원합니다.

그리고 대학원 시작부터 함께하며 항상 옆에서 모든 것을 받아주고 이해해준 보라, 기도로 응원해준 정명이형, 우리 셀에 영원한 막내 솔샘이, 영은이 나의 소중한 셀 가족들에게도 감사의 마음을 전합니다.

항상 밤늦게 귀가하여 피곤해하는 저를 끝까지 지켜보며 격려해주시며, 제가 가는 길을 지지해 주셨던 아버지 공부하는 동안 아낌없이 지원해주셨던

어머니께 사랑한다는 말을 전하고 싶습니다. 그리고 저를 믿어주셨던 할아버지와 할머니께도 감사의 마음을 전합니다.

“인생은 초판 보다 개정판이 아름답다” 는 표현처럼 비록 지난 3년의 시간동안 많은 아쉬움은 있지만 학교에서 수많은 학생들과의 시간들을 한편의 아름다움 추억들로 간직하고 또 다른 인생의 개정판을 꿈꾸며, 이제 새로운 도전을 준비 하려고 합니다. 꿈을 포기하지 않는 한 희망은 존재하는 것 처럼 저에게는 아직 꿈이 있습니다. 그리고 그 꿈을 향해 나아가려고 합니다. 앞으로 저의 삶에 역경도 있겠지만 언제든지 받아들일 준비가 되어있고 헤쳐 나갈 용기가 있습니다. 이 논문을 계기로 학문적으로 뿐만 아니라 실무적으로도 더욱 더 정진할 수 있도록 노력하겠습니다. 끝으로 논문에 도움을 주신 분들과 이 논문을 읽어주신 모든 분들의 건강과 가정의 평안을 기원합니다. 감사합니다.

2012년 1월

부경대학교 데이터베이스 연구실에서

옥영종 올림