스마트한 앱 설계방법

궁상환*

Smart Design for App

Sang Hwan Kung*

요 약 본 논문에서는 스마트폰에서 활용되는 앱 응용을 효율적으로 또 체계적으로 개발할 수 있도록 지원하는 앱 설계방법론을 제안한다. 최근의 스마트폰과 앱의 활용이 확산되면서 앱 개발에 대한 요구는 늘어나는 반면, 앱을 겨냥한 효율적인 방법론에 대한 연구는 미흡한 실정이다. 특히 앱은 아이디어를 컨텐츠 또는 프로그램으로 즉시 전환해야 하는 경우가 많아, 신속하고 체계적인 개발이 무엇보다 중요하다. 논문에서는 사용자 인터페이스(UI)를 중심으로 한 스마트한 앱 개발방법론을 소프트웨어 산출물의 요소와 설계절차의 관계를 고려하여 논리성 있는 단계적 접근방법을 토대로 제시한다. 제안된 설계방법의 검증을 위해서는 잘 알려진 일정관리 응용을 방법론의 절차에 따른 설계를 통해 타당성을 제시하고 있으며, 끝으로 잘 알려진 소프트웨어 개발 방법론들과 중요한 관점에서의 특성들을 비교해 봄으로써 제안된 방법론의 우수성을 입증하도록 하였다.

주제어 : 앱설계방법, 소프트웨어설계, 앱, 스마트폰, 사용자인터페이스

Abstract The paper describes the efficient and agile design methodology for the smart phone application, so called the App. Despite the number of smart phones as well as their applications is rapidly being increased recently, the smart and easy methodology for App development is not well developed and not matured in use yet. Since the App is converted to programs or contents from the new ideas, especially, it is important to design and implement the App rapidly and systematically. In order to resolve this problem, we adopt the UI centered approach first of all, emphasizing user-efficient utilization of applications. And we also try to build the whole process in streamlined procedure, making each step reasonable, and the input and output for the step well organized. The design methodology described here is evaluated by designing popular sample application, Scheduler, and also compared with some well-known software design methodologies.

Key Words : App Design Methodology, Software Design, App, Smart Phone, User Interface

1. 서론

스마트폰의 보편적 활용에 따라 많은 사용자가 개인 및 오피스를 위한 앱의 활용에 관심을 갖게 되었다[3]. 또한 아이디어를 중심으로 한 앱의 개발에도 더 많은 개발자들이 관심을 갖게 되었다.

그동안 다양한 소프트웨어 개발을 위해서는다양한 방법론이 개발되어 활용되어 왔음에도 불구하고, 앱의 설계 및 구현을 위한 효율적인 방법론에 대해서는 아직 연구가 미진한 상태이다. 특히, 스마트폰에서 활용되는 앱 응용은 소프트웨어 관점 뿐 아니라 사용자 인터페이스(UI)의 관점을 중요하게 고려해야 한다는 차원에서 기존의 방법론에 대한 개선이 요망되고 있다.

본 논문에서는 전통적인 어플리케이션의 개발방법론에 대한 개념과 특성을 분석하고, UI/UX에 대한 분석 및 설계방법을 살펴 본 후, 스마트폰 응용의 개발에 적합한 절차와 문서화 방법을 정의한다. 그리고 정의된 방법론에 따른 응용설계의 제시 및 기존의 방법론들과의 비교 및 검토한 결과를 결론으로 제시한다.

*백석대학교 정보통신학부 교수
논문접수: 2012년 7월 3일, 1차 수정을 거쳐, 심사완료: 2012년 7월 21일

2. 관련 연구

2.1. S/W 설계방법

2.1.1 구조적 설계방법

구조적 분석기법은 DeMarco가 제안한 처리 중심(process-oriented)의 소프트웨어 분석 및 설계방법이다. 이 방법은 하향식(top-down)의 분석원리를 이용한다[1]. 자료의 흐름을 프로그램 구조로 전환하는 과정은 다음의 4가지 단계로 구성된다.

- 정보흐름(information flow)의 유형을 결정한다.
- 흐름의 경계를 식별한다.
- 자료흐름도를 프로그램 구조도로 매핑한다.
- 구조를 분해하고(factoring) 제어계층을 정의한다.

2.1.2 객체지향 설계방법

Jacobson의 Object Oriented Software Engineering (OOSE)는 쓰임새(Use Case) 모델을 중심으로 한 객체지향 분석 및 설계 방법이다. 쓰임새란 시스템을 구성하는 단위기능이며, 모든 쓰임새의 집합은 결국 시스템의 완전한 기능을 설명하게 된다[6].

최근 객체지향 설계는 UML이라는 다이어그래밍 도구를 이용하는데, UML 기반의 객체지향 설계절차를 순서적으로 설명하면 다음과 같다.

o 쓰임새(Use Case) 다이어그램을 작성한다.
  - 각 쓰임새의 시나리오를 작성한다.
o 객체를 식별하여 개념적 클래스 다이어그램을 작성한다.
o 순차 다이어그램을 작성한다. : 메소드 식별
o 메소드의 호출을 연계하여 상세 클래스 다이어그램을 작성한다.

2.1.3 CBD(Component Based Development)

컴포넌트는 공개된 인터페이스를 갖는 객체의 집합이다. CBD는 소프트웨어를 컴포넌트라는 단위로 재사용할 수 있도록 개발한 설계방법론이다. 이 방법은 크게 두 가지 모델로부터 입력을 받아 설계를 진행한다. 하나는 Use Case 모델이며, 다른 하나는 비즈니스 개념 모델이다. Use Case 모델에서는 기능 중심의 컴포넌트를 식별하고, 비즈니스 모델에서는 시스템에 존재하는 중요한 데이터를 다루는 기능을 컴포넌트로 식별한다[2].

2.1.4 ABD(Architecture-Based Development)

아키텍처 설계방법들의 공통적인 특징은 시스템을 분해할 때, 요구되는 품질을 위한 아키텍처 패턴을 식별하고, 이 패턴에 의해 설계요소를 분해해 나간다. 논리적 관점, 동시성 관점, 배치 관점 등의 설계 View를 작성한다. 모든 View의 집합은 하나의 아키텍처 문서를 형성한다[4][5][7][8][9][10].

2.1.5 앱 개발방법

앱의 개발은 기획단계, 개발단계, 인도단계를 거쳐서 완성이 된다. 기획단계에서는 기획과 요구분석, 설계의 활동이 수행되며, 개발단계에서는 UI 디자인과 코딩, 테스트 활동이 수행된다. 한편, 최종단계인 인도단계에서는 통합 테스트, 인도, 운영 및 유지보수와 같은 활동을 수행한다.

한편, 분석과 설계단계에서는 UI 스케치, 스토리보드 작성, Navigation map 작성, UI에 대한 스펙 작성, Flow와 IO 정의를 수행하게 된다.

2.2 UI/UX 설계방법

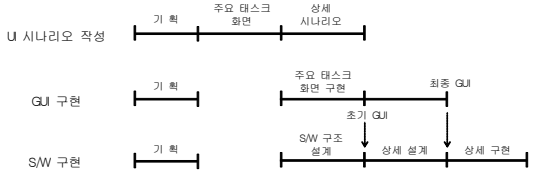
2.2.1 UI 시나리오 설계

앱의 활용이 대중화되면서 UI(User Interface)나 또는 UX(User eXperience)의 중요성이 높아져 가고 있다. 특히, UI의 설계는 그래픽이나 구체적인 S/W의 설계에 앞서, 먼저 사용자의 분석을 토대로 UI 시나리오를 충실히 작성하여야만 한다. UI/UX의 설계 절차는 다음과 같다[11].

o 시스템을 사용하는 사용자의 타입을 모델링한다.
  - 사용자의 유형과 사용 상황, 사용 패턴을 식별한다.
  - 사용빈도와 중요도를 토대로 우선순위를 결정한다.
o 사용자 태스크(task / task case)를 파악한다.
o 사용자 유형/태스크 매트릭스(matrix)를 작성한다.
o 태스크 카드를 작성한다. Process Map을 결정하는 기초자료가 된다.
o 콘텐츠 카드(기능구조)를 제작한다.
o 콘텐츠를 토대로 Wireframe(대표화면)을 작성한다.
o Paper Prototype을 제작하고 테스트한다.
o 완성된 인터액션을 기반으로 UI 시나리오를 작성한다.

2.2.2 UI 설계와 S/W 설계

UI 시나리오는 그래픽 작업인 GUI 구현과 프로그래밍 작업에 기초적인 입력으로 활용된다. 시나리오 작성 단계에서 주요 태스크에 관련된 화면에 대한 초기 드로잉이 작성되면, 이 내용을 토대로 GUI 구현을 위한 그래픽 작업을 착수한다. GUI 팀은 주요 태스크 화면에 대한 그래픽 작업을 수행하여 초기 GUI 그래픽 작업을 완료한다. 한편, S/W 팀에서는 초기 GUI를 만드는 작업과 병행하여 S/W 구조설계를 완료하며, 초기 GUI가 완료되면 이 자료를 토대로 S/W의 상세설계 작업을 수행한다. GUI 팀에서 최종적으로 GUI 작업을 완료하면, 이 결과를 이용하여 S/W 팀에서는 S/W 구현을 수행하고 완료한다.



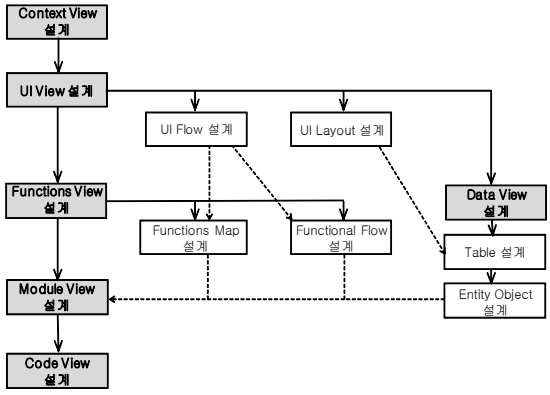
[그림 1] UI 설계와 S/W 설계

3. 앱 응용 설계절차

앱 응용설계의 첫 번째 작업은 Context View를 분석하는 작업이다. Context View는 사용자가 시스템을 보는 관점을 표현한다. 보통 Use Case 다이어그램을 이용하여 작성하며, 이해를 위해 시나리오도 작성한다.

다음의 UI View 설계단계는 사용자가 직접 보고 활용하는 작업 화면을 설계하는 단계이다. 각각의 UI 화면에 대한 설계와 각각의 화면이 절차적으로 이용되는 UI Flow를 작성한다. 이러한 UI 작성의 흐름은 작업자의 업무흐름이며, 설계되는 UI 화면은 작업자가 작업을 위해 사용하는 서류양식에 대비된다.

Functions View의 설계에서는 기능적인 관점에서 어떻게 기능들이 연계되는 가라는 분석과 여기에 속한 기능들의 그룹을 형성하는 작업을 수행한다. 기능의 흐름을 설계하는 Functional Flow 설계 작업은 이전 단계의 UI Flow 설계결과가 직접적인 입력 자료로 활용된다. Functional Flow의 각 function들은 관련된 function을 중심으로 더 큰 function이나 function의 그룹으로 조합되어 Functions Map으로 작성된다. Functions Map은 Use Case 다이어그램을 이용하여 표현되며, 초기단계의 Context View가 상세히 구체화된 산출물이라고 볼 수 있다.



[그림 2] 스마트 앱 설계 절차

Data View는 UI View의 UI Layout으로부터 도출한다. 한편, UI Layout에 표현된 입출력 필드는 응용의 중요한 데이터가 되므로, 각 화면과 연관된 입출력 자료들은 하나의 테이블 엔티티로 설계된다. 특히, 이 엔티티들은 객체지향에서 하나의 엔티티 객체가 될 수 있으므로, 엔티티-객체간 전환을 통해 중요한 엔티티를 설계한다.

Module View는 응응 시스템을 구성하는 중요한 모듈의 구조적인 측면을 표현한다. 이 View에 나타나는 모듈들은 상세 설계 후 프로그램으로 구현될 모듈이 된다. Module View는 앞서 설계된 Functions Map의 function들을 식별함과 동시에, 이들 사이의 계층적 구조를 참고하여 작성된다. 또한, Functional Flow에 표현된 function들 간의 접근관계를 참조하며, 이를 토대로 모듈간의 계층구조와 모듈간의 접근관계를 표현한다.

Code View 설계에서는 Module View에서 식별된 각각의 모듈을 위해 구체화하는 작업이 수행된다. 즉, 모듈(클래스)을 위한 변수와 메소드를 설계하고, 각 메소드가 수행하는 절차를 다이어그램이나 설명으로 기술한다.

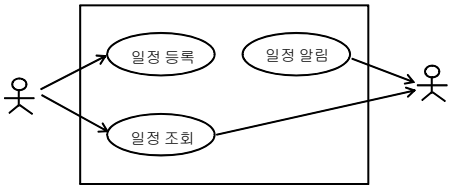
4. 일정관리응용 사례연구

예제 응용은 가장 전형적인 앱 응용으로 알려진 일정관리를 선정하여, 제안 방법론에서 정의하는 단계적인 절차에 따라 설계를 진행하여 절차를 검증하고자 하였다.

4.1. 컨텍스트 View의 설계

4.1.1 컨텍스트 다이어그램

먼저 Use Case 다이어그램을 이용하여 일정관리 응용을 특징적으로 표현한다. [그림 3]은 일정관리를 크게 일정등록과 일정조회, 그리고 일정 도래시의 자동적인 통보의 세 가지 기능으로 보여주고 있다.



[그림 3] Context 다이어그램

4.1.2 서비스 시나리오

서비스 시나리오는 Context 다이어그램을 보다 쉽게 이해할 있도록 도와주는 설명이다.

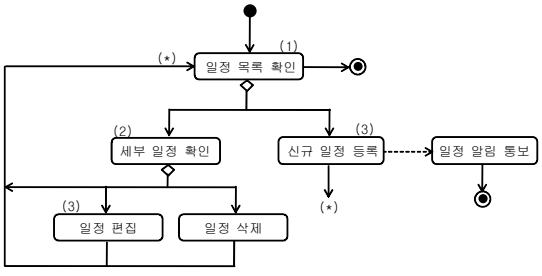
o 새로운 일정을 등록한다.
  - 시간이 되면 알림 메시지가 도착한다.

o 일정을 조회한다.
  - 일정목록이 조회된다.
  - 특정한 일정을 조회한다.(목록에서 선택)
  - 일정의 상세한 내용이 출력된다.

4.2. UI View의 설계

4.2.1 UI Flow의 설계

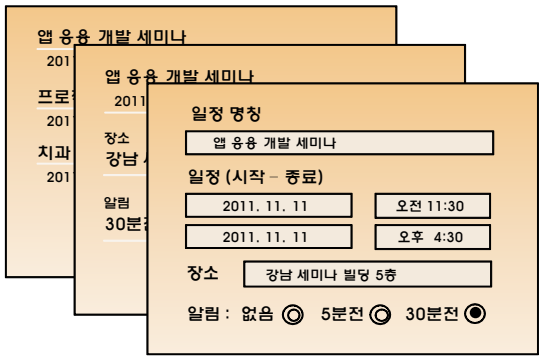
UI Flow는 반복적인 분석작업을 통해 최적의 UI를 완성할 수가 있다. 일정관리에서는 두 가지 단계를 통해 최적화를 시도하도록 한다. 먼저 일정관리에서 가장 시급한 문제는 일정을 넣는 것이라고 할 수 있다. 그러나 일단 일정이 등록되고 나면, 사용자는 일정목록을 확인하는 문제를 더 중요한 문제로 인식한다. 이러한 개선은 UI 최적화를 통해 달성할 수가 있는 것이다.



[그림 4] 사용자 이용 중심의 UI Flow

4.2.2 UI Layout의 설계

UI Flow의 각 작업 단계는 각 작업에 필요한 입출력 화면을 이용하여 수행된다. 일정관리의 작업용 UI 화면은 다음의 [그림 5]와 같다.

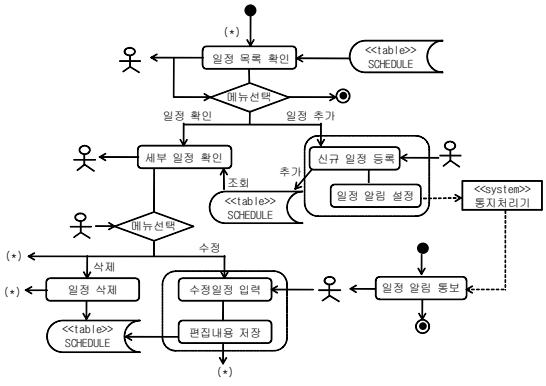


[그림 5] UI 화면설계

4.3 Functions View의 설계

4.3.1 Functional Flow의 설계

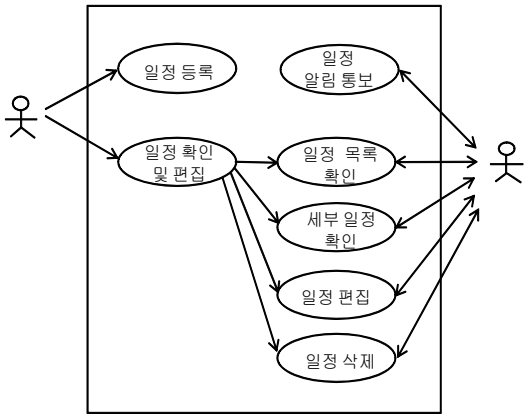
다음의 [그림 9]는 Activity 다이어그램으로 작성한 Functional Flow를 보여 준다. 여기서는 사용자 인터페이스, DB 인터페이스, 통신 인터페이스, 시스템(안드로이드) 인터페이스와 같은 인터페이스를 이용한 입력과 출력이 설계과정에서 도출되어야 한다.



[그림 6] Functional Flow

4.3.2 Functions Map의 설계

다음의 [그림 7]은 일정관리를 위한 Use Case 기반의 Functions Map을 보여 준다. 일정확인 및 편집 Use Case는 일정목록 확인, 세부일정 확인, 일정 편집, 일정 삭제 등의 Use Case를 포함하여 계층적인 기능관계를 표현한다.



[그림 7] Use Case기반의 Functions Map

4.4 Data View의 설계

4.4.1 데이터베이스 테이블의 설계

사용자의 입력이 요구되거나 사용자에게 보여 져야 하는 정보들은 하나로 조합되며, 각각은 데이터베이스 테이블의 필드가 된다. 다음의 [그림 8]은 일정(SCHEDULE) 테이블에 대한 설계를 도시한다.

일 정 ID	일 정 명 칭
ID	N A M E
i n t	S t r i n g

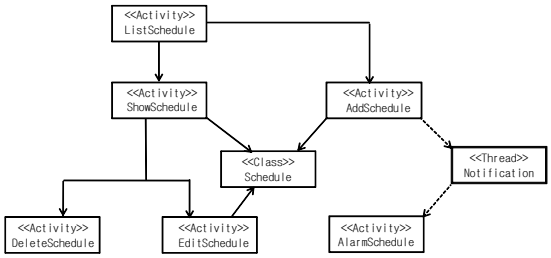
[그림 8] 일정 테이블의 설계

4.4.2 엔티티 객체의 설계

엔티티 객체는 중요한 자료에 대한 정보를 토대로 식별된다. 예제 응용에서는 데이터베이스의 일정 테이블로부터 엔티티 객체인 Schedule 객체를 도출할 수가 있다.

4.5 Module View의 설계

Module View는 모듈간의 논리적 관계를 보여 주는 동시에, 접근관계 및 실행관점을 동시에 표현한다. 각 모듈은 Activity나 Service, Thread, Class와 같은 유형으로 설계된다. 다음의 [그림 9]는 일정관리 응용을 위한 Module View를 보여 준다.



[그림 9] Module View

위 그림의 Module View에 나타난 각각의 모듈에 대한 유형과 역할은 [표 1]에 정리되어 있다.

[표 1] 일정관리 응용의 모듈

모듈명	모듈유형	모듈 설명
ListSchedule	Class	일정목록확인 모듈, main 프로그램
ShowSchdule	Class	세부일정확인 모듈
AddSchedule	Class	신규일정등록 모듈
DeleteSchedule	Class	일정삭제 모듈
EditSchedule	Class	일정편집 모듈
AlarmSchedule	Class	일정알림통보 모듈
Notification	Thread	알림통보용 시스템 모듈
Schedule	Class	일정정보 기본 클래스

5. 평가 및 결론

본 연구의 개발방법론은 스마트폰을 위한 앱 응용을 분석하고 설계하는 절차와 방법을 정의하고 있다. 설계의 대상이 되는 설계요소들을 View라는 구조적인 체계로 정립하고, 모든 View를 위한 논리적이고 자연스러운 절차를 정의하였다. 각 View는 절차적인 문서의 용도 외에, 각각의 View가 모여지면 전체가 하나의 설계문서가 되도록 고려하였다.

제안 방법론의 평가는 관련 연구에서 분석한 객체지향 방법(OOD; Object Oriented Design), 구조적 분석설계 방법(SAD; Structured Analysis and Design), 컴포넌트 기반 설계 방법(CBD; Component Based Development), 아키텍처 기반 설계방법(ABD; Architecture Based Development)를, 본 연구에서 제안하고 있는 스마트폰 앱 개발 방법(SDA; Smart Design for App)과 비교하여 보았다. 비교대상 관점은 소프트웨어 개발방법론이 가져야 하는 특성 즉, 절차적 완성도, 산출물 완성도, 모듈 도

출의 용이성, 사용 용이성과 함께, 앱의 특성이 갖는 모바일 적합성을 비교요소에 포함하여 평가하였다. 또한, 각 관점은 우수와 미흡 사이를 5 ~ 1로 점수를 부여하였다. 이러한 비교에서, 논문의 제안하는 방법론은 다른 방법론들에 비해 모든 비교 관점에서 우위에 있는 것으로 평가되었다.

[표 2] 방법론 간 비교

비교 관점	OOD	SAD	CBD	ABD	SDA
절차적 완성도	3	4	5	3	5
산출물 완성도	5	5	5	3	5
모듈 도출 용이성	4	5	3	3	5
사용 용이성	4	4	4	3	5
모바일 적합성	4	4	3	2	5
평가 합계	20	22	20	14	25

참 고 문 헌

[1] 궁상환, 소프트웨어공학, OK 프레스, 2007.

[2] 김경주, 조남규 공역, UML Components, 도서출판 인터비전, 2001.

[3] 정보통신산업진흥원, 주간기술동향 1446호, 최근 Smart Work 동향, 박신정, 2010. 12. 15.

[4] Erich Gamma, Richard Helm, Ralph Johnson and John Vissides, Design Patterns, Addison Wesley, 2000.

[5] Felix Bachmann, Len Bass, Gay Chastek, Patric Donohoe, Fabio Perzzi, Architecture Based Design Method, Technical Report CMU/SEI-2000-TR-001, CMU Software Engineering Institute, 2000.

[6] Ivar Jacobson, Object Oriented Software Engineering : A Use Case Driven, ACM Press, 1992.

[7] Len Bass, Mark Klein, Felix Bachmann, "Quality Attribute Design Primitives and the Attribute Driven Design Method", 4th International Workshop on Product Family Engineering Bilbao, Spain, 2001.

[8] Less Bass and Rick Kazman, Architecture-Based Development, CMU Software Engineering Institute, Technical Report CMU/SEI-99-TR-007, ESC-TR-99-007, 1999.

[9] Rob Wojcik and et al, Attribute-Driven Design(ADD), Version 2.0, Technical Report CMU/SEI-2006-TR-023, CMU Software Engineering Institute, 2006.

[10] Robert T. Monroe, Andrew Kompanek, Ralph Melton, and David Galan, Architectural Styles, Design Patterns, and Objects, IEEE Software, January, 1997.

[11] T아카데미, 모바일 UX/UI 기획, Ver 2.2, 2011.

궁 상 환

· 1977년 2월 : 숭실대학교 전자계산학과 졸업(이학사)
· 1983년 8월 : 고려대학교 경영대학원 전자정보처리학과 졸업(경영학석사)
· 1988년 2월 : 충북대학교 대학원 전자계산학과 졸업(이학박사)
· 1981년 7월 ~ 1998년 2월 : 한국전자통신연구원 책임연구원
· 2001년 3월 현재 : 백석대학교 정보통신학부 교수
· 관심분야 : 소프트웨어 설계, 소프트웨어 아키텍처, 분산시스템
· E-Mail: kung@bu.ac.kr