1. UI 요구사항 확인

* UI 요구사항 확인

1. UI 개념

사용자와 시스템 사이에서 의사소통 할 수 있도록 고안된 물리적, 가상의 매개체

정보 기기나 소프트웨어의 화면 등에서 사람이 접하게 되는 화면이다

UX는 UI를 포함하고 있다

1. UI 유형
2. CLI : 정적인 텍스트 기반 인터페이스, 명령어를 텍스트로 입력하여 조작
3. GUI : 그래픽 반응 기반 인터페이스, 그래픽 환경을 기반 마우스나 전자펜을 이용
4. NUI : 직관적 사용자 반응 기반 인터페이스, 키보드나 마우스 없이 신체부위이용
5. OUI : 유기적 상호 작용 기반 인터페이스, 현실에 존재하는 모든 사물이 입출력장치로
6. UI 설계 원칙
7. 직관성 : 누구나 쉽게 이해하고, 쉽게 사용할 수 있어야됨
8. 유효성 : 정확하고 완벽하게 사용자의 목표가 달성될 수 있도록 제작
9. 학습성 : 초보와 숙련자 모두가 쉽게 배우고 사용 할 수 있게 제작
10. 유연성 : 사용자의 요구사항을 최대한 수용, 실수를 방지 할 수 있도록 제작
11. UI 요구사항 확인
12. UI 요구사항 개요

사용자가 정보시스템을 구축하여 얻고자 하는 최종 목적의 기준

1. UI 품질 요구사항

기능성 : 실제 사용시 정확하지 않은 결과가 발생할 확률과 관련하여 시스템의 동작을 관찰 하기 위한 품질 기준

신뢰성 : 시스템이 일정한 시간 또는 작동되는 시간 동안 의도하는 기능을 수행함을 보증하는 품질 기준

사용성 : 사용자와 컴퓨터 사이에 발생하는 어떠한 행위를 정확하고 쉽게 인지할 수 있는 품질 기준

효율성 : 할당된 시간에 한정된 자원으로 얼마나 빨리 처리 할 수 있는가에 대한 기준

유지보수성 : 요구사항을 개선하고 확장하는 데 있어 얼마나 용이한가에 대한 품질 기준

이식성 : 다른 플랫폼에서도 많은 추가 작업 없이 얼마나 쉽게 적용이 가능한가에 대한 품질 기준

* UI 지침

1. 개념

UI 표준에 따라 사용자 인터페이스 설계, 개발 시 지켜야할 세부사항을 규정하는 가이드라인이다.

1. 소프트웨어 개발 단계별 UI 지침
2. 목표 정의

성공적인 수행을 위해 내부 관계자에게 UI 개발 필요성 및 목표를 공유하고 개발 범위를 수립하기 위한 활동을 한다

3C 분석 : 고객, 경쟁 자사, 경쟁 사를 비교하고 어떻게 차별화 해서 이길것인지

SWOT 분석 : 기업의 내부 환경과 외부 환경을 분석하고 전략을 수립하는 방법

시나리오 플래닝 : 불확실성이 높은 상황 변화를 사전에 예측하고 시나리오를 설계

워크숍 : 소집단 정도의 인원으로 연구회 및 세미나

1. 프로젝트 계획

사용자 분석 및 최신 트렌드, 경쟁사 동향을 통해 파악된 핵심 기능을 토대로 UI 개발

프로파일 : 어떤 시스템을 일정 범위 내에서 한정적으로 특징 지우는 그룹화된 값

리서치 : 이미 존재하던 지식의 발견, 해설, 정정, 재확인 등에 초점

1. 요구사항 정의

페르소나 : 사용자의 다양한 목적과 관찰된 행동 패턴을 응집시켜 놓은 가상의 사용자

브레인스토밍 : 아이디어 목록을 통해서 특정한 문제에 대한 해답을 찾고자 하는 회의

요구사항 매트릭스 : 페르소나의 목적을 기준으로 만든 요구사항 표

정황 시나리오 : 요구사항 정의에 사용되는 초기 시나리오

1. 설계 및 구현

UI 설계 시안을 토대로 실제 설계 및 구현을 위해서 모든 화면에 대한 UI 상태 설계 단계를 진행

1. 테스트

사용성 테스트 : 사용자가 직접 제품을 사용하면서 시나리오에 맞추어 과제 수행

1. UI 화면 설계
2. 와이어프레임 : 화면구성을 협의하거나 서비스의 간략한 흐름을 공유하기 위해 화면단위의 레이아웃 설계
3. 스토리보드 : 정책, 프로세스, 콘텐츠 구성, 와이어프레임 ( UI, UX ), 기능정의, 데이터베이스 연동 등 서비스 구축을 위한 모든 정보가 담겨 있는 설계 산출물
4. 프로토타입 : 정적인 화면으로 설계된 와이어프레임 또는 스토리보드에 동적인 효과를 적용하여 실제 구현된 것처럼 시뮬레이션을 할 수 있는 모형

* UI 프로토타입 제작 및 목업

1. 프로토타입
2. 개념

컴퓨터 시스템이나 소프트웨어의 설계 또는 성능, 구현 가능성, 윤용 가능성을 평가하거나 요구사항을 좀더 잘 이해하고 결정하기 위하여 전체적인 기능을 간략한 형태로 구현한 시제품

1. 의의

시스템 설계 및 개발에 소요되는 총 비용과 노력 절감

1. 장점 및 단점

사용자 설득과 이해가 쉬움, 개발 시간 감소, 요류 사전 발견을 통한 예방 가능

수정 과정 증가시, 작업 시간 증가 위험 존재

1. 목업

제품이나 서비스의 디자인을 시각적으로 보여주기 위한 모형

1. UI 설계

* UML

1. UML 개념

객체 지향 소프트웨어 개발 과정에서 산출물을 명세화, 시각화, 문서화할 때 사용되는 모델링 기술과 방법론을 통합하여 만든 표준화된 범용 모델링 언어

1. UML의 특징

UML은 방법론을 통합한 것으로, 표준화된 모델링 기법을 제공한다

1. 가시화 언어

개념 모델 작성 시 오류가 적고 의사소통이 용이

1. 구축 언어

다양한 프로그래밍 언어로 실행 시스템의 예측 가능

1. 명세화 언어

정확한 모델 제시, 완전한 모델 작성 가능

1. 문서화 언어

시스템에 대한 평가 및 의사소통의 문서

1. UML의 구성 요소
2. 사물 : 추상적인 개념으로, 주제를 나타내는 요소 단어 관점에서 명사 또는 동사
3. 관계 : 사물의 의미를 확장하고 명확히 하는 요소 사물과 사물을 연결 하여 관계 표현
4. 다이어그램 : 사물과 관계를 모아 그림으로 표현한 형태
5. UML 다이어그램

구분에 따라 구조적 ( 정적 ) 다이어그램, 행위적 ( 동적 ) 다이어그램으로 구성됨

1. 구조적 / 정적 다이어그램

시스템의 구조 및 구성요소 들 간의 관계를 나타내는 다이어그램이다

클래스 : 클래스의 속성 및 연산과 클래스 간 정적인 관계를 표현한 다이어그램

객체 : 객체와 객체 사이의 관계로 표현, 연관된 모든 인스턴스를 표현

컴포넌트 : 시스템을 구성하는 컴포넌트와 그들 사이의 의존관계를 나타낸다

배치 : 컴포넌트 사이의 종속성을 표현하고 물리적 요소들의 위치를 표현한다

복합체 구조 : 클래스나 컴포넌트가 복합 구조를 갖는 경우 내부 구조를 표현

패키지 : 유스케이스나 클래스 등의 모델 요소들을 그룹화한 패키지들의 관계

1. 행위적 / 동적 다이어그램

유스케이스 : 사용자의 관점에서 표현

시퀀스 : 객체 간 동적 상호 작용을 시간적 개념을 중심으로 메시지 흐름으로 표현

커뮤니케이션 : 동작에 참여하는 객체들이 주고받는 메시지, 객체 간의 연관 표현

상태 : 하나의 객체가 자신이 속한 클래스의 상태변화, 상호작용에 따라 상태 변화 표현

활동 : 시스템이 어떤 기능을 수행하는지를 객체의 처리 로직이나 조건에 따른 처리의 흐름을 순서대로 표현 하는 다이어그램

타이밍 : 객체 상태 변화와 시간 제약을 명시적으로 표현하는 다이어그램

컴포넌트, 배치 다이어그램은 구현 단계에서 사용되는 다이어그램이다

1. UML 스테리오 타입

‘<<>>’ 길러멧 기호를 사용하여 표현

1. <<include>> : 유스케이스가 어떤 시점에 반드시 다른 유스케이스를 실행하는 표현 관계
2. <<extend>> : 유스케이스가 어떤 시점에 유스케이스를 실행할 수도 아닐 수도 있는 관계
3. <<interface>> : 모든 메서드가 추상 메서드이며 바로 인스턴스를 만들 수 없는 클래스로 추상 메서드와 상수 만으로 구성된 클래스
4. <<entity>> : 일반적으로 정보 또는 오래 지속되는 연관된 행위를 형상화 하는 클래스로 유스케이스 처리 흐름이 수행되는 과정에서 기억 장치에 저장되어야 할 정보를 표현
5. <<boundary>> : 시스템과 외부 엑터와 의 상호 작용 담당
6. <<control>> : 시스템이 제공하는 기능의 로직 및 제어를 담당하는 클래스
7. UML의 유형
8. 클래스 다이어그램

객체 지향 모델링 시 클래스의 속성 및 연산과 클래스 간 정적인 관계를 표현한 다이어그램이다

구성요소로는 클래스, 속성, 연산, 메서드, 접근 제어자가 있다

클래스 : 공통의 속성, 연산, 관계, 의미를 공유하는 객체들의 집합

속성 : 클래스의 구조적 특성에 이름을 붙인 것, 인스턴스가 보유 할 수 있는 값

연산 ( 메서드 ) : 이름, 타입, 매개변수들과 연관된 행위를 호출하는데 요구되는 제약사항들을 명시하는 클래스의 행위적인 특징

접근 제어자 : 클래스에 접근할 수 있는 정도를 표현

* - : 클래스 내부 접근만 허용 (private)
* + : 클래스 외부 접근을 허용 (pubilc)
* # : 동일 패키지/파생 클래스에서 접근 가능 (protected)
* ~ : 동일 패키지 클래스에서 접근 가능 (default)

클래스 간의 관계는 연관, 집합, 포함(복합), 일반화, 의존, 실체화 관계가 있다

* 연관 : 클래스가 서로 개념적으로 연결된 관계
* 의존 : 하나의 클래스가 또 다른 클래스를 사용하는 관계
* 일반화 : 하나의 사물이 다른 사물에 비해 일반적인지 구체적인지를 표현하는 관계
* 실체화 : 집합 관계의 특수한 형태로, 포함하는 사물의 변화가 포함되는 사물에게 영향을 미치는 관계
* 포함 : 포함하는 사물의 변화가 포함되는 사물에게 영향을 미치는 관계
* 집합 : 하나의 객체에 여러 개의 독립적인 객체들이 구성되는 관계

1. 유스케이스 다이어그램

시스템이 제공하고 있는 기능 및 그와 관련된 외부 요소를 사용자의 관점에서 표현

구성요소로는 유스케이스, 액터, 시스템, 시나리오, 이벤트의 흐름 이 있다

* 유스케이스 : 시스템이 제공해야 하는 서비스 , 기능을 표현하는 요소
* 액터 : 사용자가 시스템에 대해 수행하는 역할을 표현하는 요소
* 시스템 : 전체 시스템의 영역을 표현하는 요소
* 시나리오 : 발생되는 이벤트의 흐름
* 이벤트의 흐름 : 사람, 시스템, 하드웨어, 시간의 흐름

유스케이스 다이어그램의 관계는 액터와 유스케이스, 유스케이스와 유스케이스 사이에서 나타날 수 있으며 포함, 확장, 일반화 관계가 있다

포함(include) : 유스케이스를 수행할 때 다른 유스케이스가 반드시 수행되는 관계

확장(extend) : 특정 조건일 때 한 유스케이스로만 확장되는 관계

일반화(Generalization) : 추상적인 액터와 좀더 구체적인 액터 사이에 맺어주는 관계

1. 시퀀스 다이어그램

객체 간 상호 작용을 메시지 흐름으로 표현한 다이어그램, 시간적 개념을 중심으로 모델링 하는 과정이며 객체의 오퍼레이션과 속성을 상세히 정의해야 한다.

시퀀스 다이어그램 구성요소로는 객체, 생명선, 활성화, 메시지가 있다

객체 : 객체는 위쪽에 표시되며 아래로 생명선을 가지는 요소 사각형 안에 밑줄 친 이름

생명선 : 객체로부터 뻗어 나가는 점선

실행 : 오퍼레이션(함수)이 실행되는 시간을 의미 하는 요소 직사각형이 길어질수록 수행시간이 긺

메시지 : 객체간 의 상호 작용을 위한 요소 다른 객체로의 메시지 전달

1. 패키지 다이어그램

시스템의 서로 다른 패키지들 사이의 의존 관계를 표현하기 위한 다이어그램

패키지 다이어그램 구성요소로는 패키지, 의존관계가 있다

* 패키지 : 객체들을 그룹화한 요소
* 의존 : 하나의 패키지가 다른 패키지를 사용하는 관계 의존성의 성질을 나타내기 위해 스테리오 타입을 붙일 수 있다

1. 활동 다이어그램

시스템이 어떤 기능을 수행하는지를 객체의 처리 로직이나 조건에 따른 처리의 흐름을 순서대로 표현하는 다이어그램, 오퍼레이션이나 처리 과정이 수행되는 동안 일어나는 일들을 단계적으로 표현한다, 활동 다이어그램은 하나의 유스케이스 안이나, 유스케이스 사이에서 발생 하는 복잡한 처리의 흐름을 명확하게 표현 할 수 있다

활동 다이어그램 구성요소는 시작점, 전이, 액션/액티비티, 조건, 노드, 병합 노드, 포크 노드, 조인 노드, 구획면이 있다

액션 : 더 이상 분해할 수 없는 단일 작업

액티비티 : 몇개의 액션으로 분리될 수 있는 작업

1. 상태 다이어그램

하나의 객체가 자신이 속한 클래스의 상태 변화 혹은 다른 객체와의 상호 작용에 따라 상태가 어떻게 변화하는지 표현하는 다이어그램이다, 어떤 이벤트에 의해 객체 자신이 속한 클래스의 상태 변화나 객체 간 상호 작용하는 과정에서의 상태 변화를 표현한다

상태 다이어그램 구성요소는 상태, 시작 상태, 종료 상태, 전의, 이벤트, 전이 조건이 있다

* 상태 (state) : 객체가 존재할 수 있는 조건을 표현하는 요소, 둥근 사각형 안에 객체의 상태 기술
* 시작 상태 ( Initial State ) : 속이 채워진 원으로 표현
* 종료 상태 ( Final State ) : 원안에 속이 채워진 원으로 표현
* 전이 ( Transition ) : 객체의 상태가 다른 상태로 변경되는 상태를 표현, 화살표
* 이벤트 ( Event ) : 상태의 변화를 주는 현상을 표현, 전이 위에 이벤트 이름 표기
* 전이 조건 ( Transition Condition ) : 특정 조건 만족 시 전이가 발생하기 위해 사용되는 요소

1. 커뮤니케이션 다이어그램

시퀀스 다이어그램과 같이 동작에 참여하는 객체들이 주고받는 메시지를 표현하고, 메시지뿐만 아닌 객체 간의 연관까지 표현하는 다이어그램, 시스템이나 객체들이 메시지를 주고받으며 시간의 흐름에 따라 상호 작용하는 과정을 표현한 다이어그램

커뮤니케이션 다이어그램 구성요소는 액터, 객체, 링크, 메시지가 있다

* 액터 : 시스템으로부터 서비스를 요청하는 외부 요소
* 객체 : 메시지를 주고받은 주체 콜론(:) 을 기준으로 앞에는 객체명, 뒤쪽은 클래스명 기술
* 링크 : 객체들 간의 관계를 표현 링크에 메시지를 표현, 실선
* 메시지 : 객체 간의 상호 작용을 위한 요소, 화살표의 방향은 메시지를 받는 쪽으로 향하게 표현

1. 컴포넌트 다이어그램

시스템을 구성하는 물리적인 컴포넌트와 그들 사이의 의존 관계를 나타내는 다이어그램

코드 컴포넌트 기반의 물리적 구조로 표현되며 실질적 프로그래밍 작업에 사용한다

컴포넌트 다이어그램 구성요소는 컴포넌트, 인터페이스, 의존 관계가 있다

* 컴포넌트 : 모듈화된 소프트웨어 시스템의 구성요소, 탭이 달린 직사각형
* 인터페이스 : 스테레오 타입을 이용 <<interface>> 로 표기
* 의존 : 화살표 모양의 점선으로 연결