



정보처리기사실기

1>요구사항확인

dumok.net

요구사항확인

개발 대상 소프트웨어의 요구사항을 분석하고 개념 모델링을 수행할 수 있다. 디자인 패턴을 활용하여 설계할 수 있다.

1

현행 시스템 분석하기



2

요구사항 확인하기

3

분석 모델 확인하기





DUMOK.NET



출제비율

구분	챕터	출제문항수
1	현행시스템 분석하기	1
2	요구사항 확인하기	0
3	분석 모델 확인하기	2

dumok.net

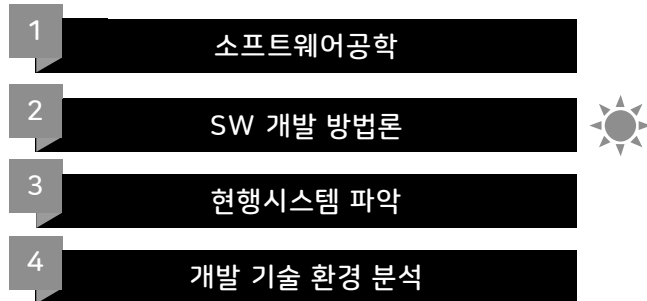
*PASS DNA*

dumok.net

1>현행시스템 분석하기

현행시스템 분석하기

개발 대상 시스템을 분석한다.



dumok.net

학습 목표 및 키워드



학습목표 및 키워드

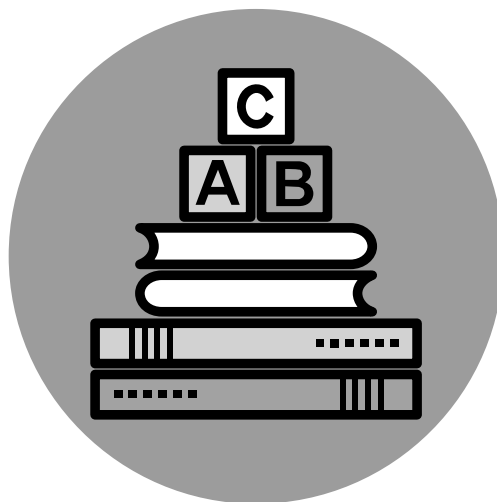
- 현행 시스템 분석을 위한 전통적 소프트웨어 개발 방법론을 설명할 수 있다.
 - 폭포수 모형, 나선형 모형, V-모델, 재공학, 역공학
- 최근 소프트웨어 개발 방법론을 설명할 수 있다.
 - 애자일 방법론
 - XP 핵심가치 (소통, 단순성, 피트백, 용기, 존중)
 - XP 12가지 실천사항
 - SCRUM 팀 역할 (제품 책임자, 스크럼 마스터, 스크럼 팀)
 - SCRUM 절차



학습목표 및 키워드

- 현행시스템을 분석 할 수 있다.
 - 시스템 구성, 시스템 기능, 인터페이스
 - 소프트웨어, 하드웨어, 네트워크
- 플랫폼의 개념을 설명할 수 있다.
 - 플랫폼의 성능 특성 분석 항목 (응답시간, 가용성, 사용률)
- 미들웨어를 설명할 수 있다.
- 오픈소스라이선스를 구분 할 수 있다.

dumok.net



Thank you



1/요구사항확인

2>요구사항 확인하기

dumok.net

요구사항 확인하기

고객이 요청한 요구사항은 개발하고 관리할 수 있다.

1

요구사항 개발

2

요구사항 관리



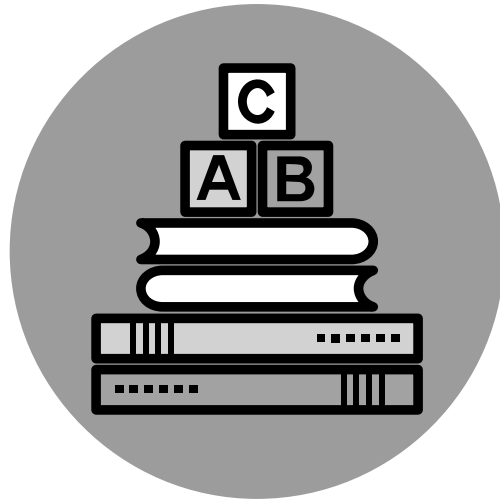
학습목표 및 키워드

- 요구사항의 정의와 요구공학의 개념을 서술 할 수 있다.
- 요구사항 개발 프로세스를 순서대로 서술 할 수 있다.
 - 도출 -> 분석 -> 명세 -> 확인
- 요구사항 도출 기법을 나열 하고 설명할 수 있다.
 - 문서조사, 관찰, 설문, 롤 플레잉, 워크숍, 브레인 스토밍, 인터뷰, 프로토타이핑, Use Case, 벤치마킹
- 요구사항 을 기능적/비기능적 요구사항으로 분류 할 수 있다.
- 요구사항 명세서를 작성 할 수 있다.
 - 시스템 정의서, 시스템 요구사항 명세서, 소프트웨어 요구사항 명세서 작성



학습목표 및 키워드

- 요구사항 타당성 검증 사항을 말할 수 있다.
 - 무결성, 일관성, 명확성, 기능성, 검증및 추적 가능성
- 요구사항 관리 프로세스를 순서대로 설명할 수 있다.
 - 협상 - 베이스라인 설정 - 변경관리 - 확인 및 검증
- 인수 테스트 방법을 설명 할 수 있다.
 - 알파 테스트, 베타 테스트
- 테스트 V 레벨을 설명 할 수 있다.



Thank you

dumok.net



PASS DNA

dumok.net

1/요구사항확인

3>분석 모델 확인하기

분석모델 확인하기

요구사항 분석을 위한 분석 모델을 알 수 있다.
UML 및 디자인 패턴의 종류를 구분 할 수 있다.

- 1 분석 모델
- 2 분석 모델 검증
- 3 개념 모델링
- 4 디자인 패턴

dumok.net

학습 목표 및 키워드



학습목표 및 키워드

- 구조적 분석 모델을 구분 할 수 있다.
 - 자료흐름도, 자료사전, 소단위 명세서, 개체 관계도, 상태전이도
- 요구사항 분석 모델 검증 절차를 알 수 있다.
 - 사례 모델 검증 - 개점 수준 클래스 검증 - 분석 클래스 검증
- 분석 클래스의 스테레오 타입을 구분 할 수 있다.
 - 경계, 제어, 엔티티
 - 스테레오 타입의 유형은 << >> Guillemet 으로 표현한다.
- 소프트웨어 개발 자동화 도구의 개념을 설명할 수 있다.
 - CASE의 장단점




학습목표 및 키워드

- 개념 모델링 UML 의 개념을 이해 할 수 있다.
- 럼바우 객체지향 분석 기법의 모델링 기법을 구분하고 산출물을 알 수 있다.
 - 객체 모델링 : 객체 다이어그램으로 표시
 - 동적 모델링 : 상태 다이어그램으로 표시
 - 기능 모델링 : 자료흐름도로 표시
- UML의 특성을 알 수 있다.
 - 비주얼화, 문서화, 명세화, 구축
- UML의 소프트웨어에 대한 관점을 구분할 수 있다.
 - 정적, 동적, 기능적

dumok.net



학습목표 및 키워드

- UML의 구성을 알 수 있다.
 - 사물, 관계, 다이어그램
- UML 접근 제어자를 구분할 수 있다.
 - Public(+), Private(-), Protect(#), Package(~)
- 연관 관계의 다중성 표현을 이해할 수 있다.
- 구조/행위 다이어그램을 구분 할 수 있다. 
 - 구조 : 클래스, 패키지, 복합구조, 오브젝트, 콤포넌트, 배치
 - 행위 : 유스케이스, 액티비티, 콜라보레이션, 스테이트, 인터액션, 시퀀스, 커뮤케이션, 인터액션 오버뷰, 타이밍



학습목표 및 키워드

- 클래스 다이어그램의 표현 방법과 관계 표현을 구분 할 수 있다.

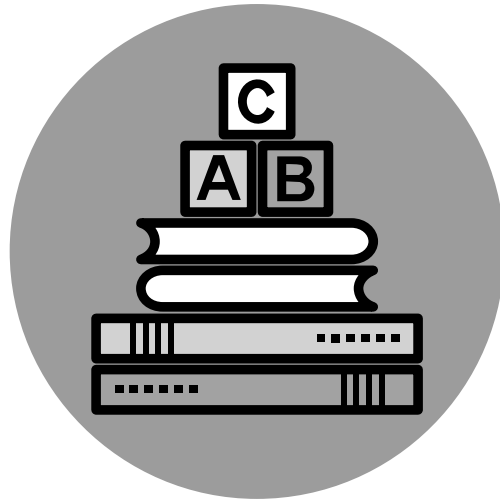
연관 관계	————→	<ul style="list-style-type: none"> 클래스 연결 상태 표시(개념상 서로 연결) 한 클래스가 다른 클래스에서 제공하는 기능을 사용할 때
의존 관계	----->	연관 관계와 동일하지만 메소드를 사용할 때와 같이 매우 짧은 시간만 유지
일반화 관계	————▷	객체 지향에서 상속 관계(S-A)를 표현하며, 한 클래스가 다른 클래스를 포함하는 상위 개념일 때 사용
집합/포함 관계	————◇	<ul style="list-style-type: none"> 클래스 사이 전체나 부분이 같은 관계 전체/부분 객체 라이프타임 독립적(전체 객체 삭제 → 부분 객체 남음)
	————◆	전체/부분 객체 라이프 타임 의존적(전체 객체 삭제 → 부분 객체 삭제)
실체화 관계	-----▷	책임 집합 인터페이스와 실제로 실현한 클래스들 사이의 관계



학습목표 및 키워드

- 디자인패턴을 설명하고 구성 요소를 나열 할 수 있다.
 - 이름, 문제 및 배경, 해법, 결과, 사례, 코드
- GoF 디자인패턴의 생성, 구조, 행위 패턴을 구분 하고 설명 할 수 있다.

생성(Creational)	구조(Structure)	행위(Behavioral)
Abstract Factory(추상 팩토리)	Adapter	Chain of responsibility
Builder	Bridge	Command
Factory Method	Composite	Interpreter
Prototype	Decorator	Iterator
Singleton	Façade	Mediator
	Flyweight	Memento
	Proxy	Observer
		State
		Strategy
		Template Method
		Visitor



Thank you

dumok.net