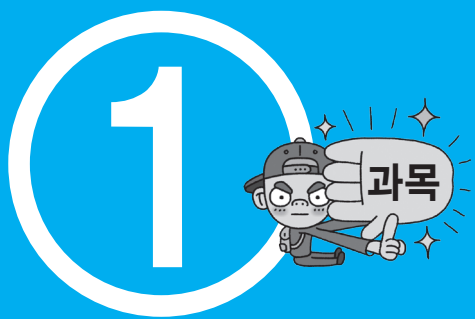


소프트웨어 설계


베리나이쓰~
SNG
구독!

1장 요구사항 확인

이 장에서 꼭 알아야 할 키워드 Best 10

1. 폭포수 모형 **2.** 나선형 모형 **3.** 애자일 **4.** 스크럼 **5.** XP(eXtreme Programming) **6.** 기능 요구사항
7. 비기능 요구사항 **8.** 프로토타이핑 **9.** UML **10.** UML 다이어그램

SECTION 001 ● LIVE

소프트웨어 생명 주기

전문가의 조언

일반적으로 소프트웨어는 요구사항을 분석해서 설계하고 그에 맞게 개발한 후 소프트웨어의 품질이 항상 최상의 상태를 유지할 수 있도록 관리하는데, 이러한 과정을 단계로 나눈 것을 소프트웨어 생명 주기라고 합니다. 소프트웨어 생명 주기의 의미를 기억해 두세요.

소프트웨어 개발 방법론

소프트웨어 개발 방법론은 소프트웨어 개발과 유지보수 등에 필요한 여러 가지 작업들의 수행 방법과 이러한 작업들을 좀 더 효율적으로 수행하기 위해 필요한 각종 기법 및 도구를 체계적으로 정리하여 표준화한 것입니다.

전문가의 조언

폭포수 모형은 한 단계가 완전히 끝나야만 다음 단계로 넘어가는 개발 방법론이라는 것을 염두에 두고 특징을 정리하세요.

매뉴얼

매뉴얼은 프로그램들의 사용과 운영에 대한 내용이 기술되어 있는 문서입니다.

1 소프트웨어 생명 주기(Software Life Cycle)

소프트웨어 생명 주기는 소프트웨어 개발 방법론*의 바탕이 되는 것으로, 소프트웨어를 개발하기 위해 정의하고 운용, 유지보수 등의 과정을 각 단계별로 나눈 것이다.

- 소프트웨어 생명 주기는 소프트웨어 개발 단계와 각 단계별 주요 활동, 그리고 활동의 결과에 대한 산출물로 표현한다. 소프트웨어 수명 주기라고도 한다.
- 소프트웨어 생명 주기를 표현하는 형태를 소프트웨어 생명 주기 모형이라고 하며, 소프트웨어 프로세스 모형 또는 소프트웨어 공학 패러다임이라고도 한다.
- 개발자는 문제의 유형이나 개발 방법 등에 따라 특정 모형을 선택하여 사용할 수도 있고, 개별적인 모형을 사용할 수도 있다.
- 일반적으로 사용되는 소프트웨어 생명 주기 모형에는 폭포수 모형, 프로토타입 모형, 나선형 모형, 애자일 모형 등이 있다.

2 폭포수 모형(Waterfall Model)

폭포수 모형은 폭포에서 한번 떨어진 물은 거슬러 올라갈 수 없듯이 소프트웨어 개발도 이전 단계로 돌아갈 수 없다는 전제하에 각 단계를 확실히 매듭짓고 그 결과를 철저하게 검토하여 승인 과정을 거친 후에 다음 단계를 진행하는 개발 방법론이다.

- 폭포수 모형은 소프트웨어 공학에서 가장 오래되고 가장 폭넓게 사용된 전통적인 소프트웨어 생명 주기 모형으로, 고전적 생명 주기 모형이라고도 한다.
- 소프트웨어 개발 과정의 한 단계가 끝나야만 다음 단계로 넘어갈 수 있는 선형 순차적 모형이다.
- 제품의 일부가 될 매뉴얼*을 작성해야 한다.
- 각 단계가 끝난 후에는 다음 단계를 수행하기 위한 결과물이 명확하게 산출되어야 한다.
- 두 개 이상의 과정이 병행하여 수행되지 않는다.

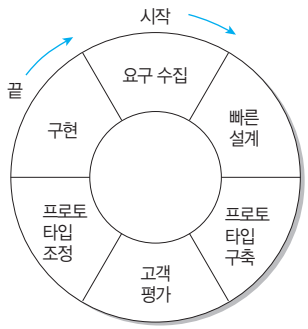
타당성 검토 → 계획 → 요구 분석 → 설계 → 구현(코딩) → 시험(검사) → 유지보수

3 프로토타입 모형(Prototype Model, 원형 모형)

프로토타입 모형은 사용자의 요구사항을 정확히 파악하기 위해 실제 개발될 소프트웨어에 대한 견본(시제)품(Prototype)을 만들어 최종 결과물을 예측하는 모형이다.

- 시제품은 사용자와 시스템 사이의 인터페이스에 중점을 두어 개발한다.

- 시스템의 일부 혹은 시스템의 모형을 만드는 과정으로서 요구된 소프트웨어를 구현하는데, 이는 추후 구현 단계에서 사용될 골격 코드가 된다.
- 소프트웨어의 개발이 완료된 시점에서 오류가 발견되는 폭포수 모형의 단점을 보완하기 위한 모형이다.

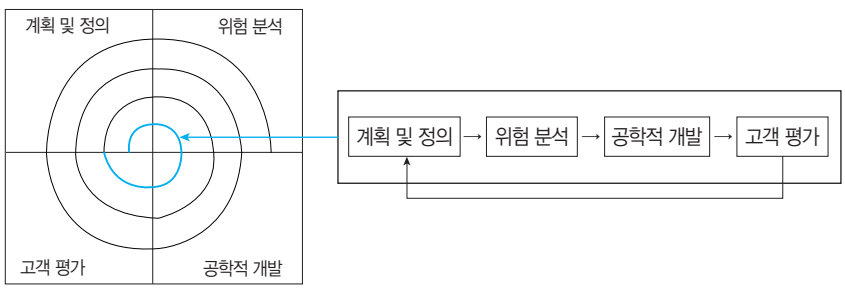


프로토타입 모형

4 나선형 모형(Spiral Model, 점진적 모형)

나선형 모형은 보헴(Boehm)이 제안한 것으로, 폭포수 모형과 프로토타입 모형의 장점에 위험 분석 기능을 추가한 모형이다.

- 나선을 따라 돌듯이 여러 번의 소프트웨어 개발 과정을 거쳐 점진적으로 완벽한 최종 소프트웨어를 개발하는 것으로, 점진적 모형이라고도 한다.
- 소프트웨어를 개발하면서 발생할 수 있는 위험을 관리하고 최소화하는 것을 목적으로 한다.
- 점진적으로 개발 과정이 반복되므로 누락되거나 추가된 요구사항을 첨가할 수 있고, 정밀하며, 유지보수 과정이 필요 없다.



전문가의 조언

나선형 모형은 나선을 따라 돌듯이 소프트웨어 개발 과정을 여러 번 반복하면서 진행하는 개발 방법입니다. 원이 점점 더 커지는 나선형 모형의 그림을 생각하면서 나선형 모형의 의미와 특징을 정리하세요.

5 애자일 모형(Agile Model)

애자일은 '민첩한', '기민한'이라는 의미로, 고객의 요구사항 변화에 유연하게 대응할 수 있도록 일정한 주기를 반복하면서 개발과정을 진행한다.

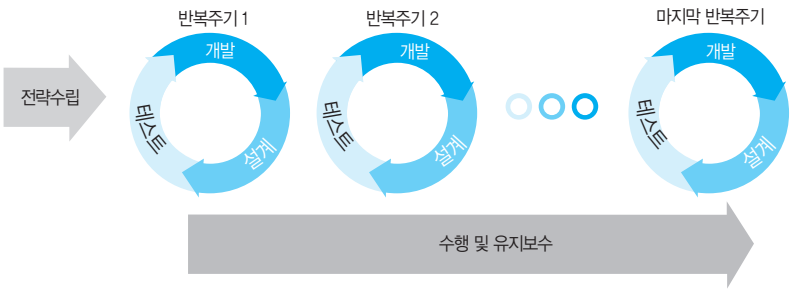
- 애자일 모형은 어느 특정 개발 방법론이 아니라 좋은 것을 빠르고 낭비 없게 만들기 위해 고객과의 소통에 초점을 맞춘 방법론을 통칭한다.
- 애자일 모형은 스프린트(Sprint) 또는 이터레이션(Iteration)이라고 불리는 짧은

전문가의 조언

애자일 모형은 고객의 다양한 요구사항의 변화에 유연하게 대응하기 위해 일정한 개발 주기를 반복하는 것이 핵심이라는 것을 염두에 두고 특징을 정리하세요. 특히 폭포수 모형과 비교되는 특징들은 확실히 숙지하세요.

개발 주기를 반복하며, 반복되는 주기마다 만들어지는 결과물에 대한 고객의 평가와 요구를 적극 수용한다.

- 각 개발주기에서는 고객이 요구사항에 우선순위를 부여하여 개발 작업을 진행한다.
- 소규모 프로젝트, 고도로 숙달된 개발자, 급변하는 요구사항에 적합하다.
- 애자일 모형을 기반으로 하는 소프트웨어 개발 모형에는 스크럼(Scrum), XP(eXtreme Programming), 칸반(Kanban), Lean, 크리스탈(Crystal), ASD(Adaptive Software Development), FDD(Feature Driven Development), DSDM(Dynamic System Development Method), DAD(Disciplined Agile Delivery) 등이 있다.



잠깐만요

애자일 선언(Agile Manifesto)

2001년 17명의 애자일 전문 개발자가 공통의 관점을 정리해 '애자일 SW 개발 선언문'을 만들었습니다. 선언문에는 애자일 개발 철학이 담겨있는 4가지 핵심 가치와 애자일 개발을 실무에 적용할 때 기준이 되는 12가지 실행 지침이 담겨있는데, 그 내용은 다음과 같습니다.

애자일 개발 4가지 핵심 가치

1. 프로세스와 도구보다는 개인과 상호작용에 더 가치를 둔다.
2. 방대한 문서보다는 실행되는 SW에 더 가치를 둔다.
3. 계약 협상보다는 고객과 협업에 더 가치를 둔다.
4. 계획을 따르기 보다는 변화에 반응하는 것에 더 가치를 둔다.

애자일 개발 12가지 실행 지침

1. 유용한 소프트웨어를 빠르고, 지속적으로 제공하여 고객을 만족시킨다.
2. 개발 막바지라도 요구사항 변경을 적극 수용한다.
3. 몇 개월이 아닌 몇 주 단위로 실행되는 소프트웨어를 제공한다.
4. 고객과 개발자가 프로젝트 기간에 함께 일한다.
5. 개발에 대한 참여 의지가 확실한 사람들로 팀을 구성하고, 필요한 개발 환경과 지원을 제공하며, 일을 잘 끝낼 수 있도록 신뢰한다.
6. 같은 사무실에서 얼굴을 맞대고 의견을 나눈다.
7. 개발의 진척도를 확인하는 1차 기준은 작동하는 소프트웨어이다.
8. 지속 가능한 개발을 장려하고 일정한 속도로 개발을 진행한다.
9. 기술적 우수성과 좋은 설계에 지속적인 관심을 기울이면 민첩성이 향상된다.
10. 단순화를 추구한다.
11. 최상의 아키텍처, 명확한 요구사항, 최상의 설계는 자기 스스로 일을 주도하는 조직적인 팀으로부터 나온다.
12. 더 효과적인 팀이 될 수 있는 방안을 정기적으로 깊이 고민하고 그에 따라 팀의 행동을 조정한다.

6 폭포수 모형과 애자일의 비교

구분	폭포수 모형	애자일
새로운 요구사항 반영	어려움	지속적으로 반영
고객과의 의사소통	적음	지속적임
테스트	마지막에 모든 기능을 테스트	반복되는 일정 주기가 끝날 때마다 테스트
개발 중심	계획, 문서(매뉴얼)	고객

기출문제 따라잡기 Section 001

출제예상
1. 소프트웨어 수명 주기 모형 중 폭포수 모형에 대한 설명으로 옳지 않은 것은?

① 적용 사례가 많다.
② 단계별 정의가 분명하다.
③ 단계별 산출물이 명확하다.
④ 요구사항의 변경이 용이하다.

한 단계를 수행하고 다음 단계로 넘어가면 이전 단계로 다시 돌아갈 수 없는 것이 폭포수 모형입니다. 즉 요구사항 변경이 매우 어렵습니다.

출제예상
2. 나선형(Spiral) 모형에 대한 설명으로 옳지 않은 것은?

① 여러 번의 개발 과정을 거쳐 점진적으로 완벽한 소프트웨어를 개발한다.
② 대규모 시스템의 소프트웨어 개발에 적합하다.
③ 위험성 평가에 크게 의존하기 때문에 이를 발견하지 않으면 문제가 발생할 수 있다.
④ 실제 개발될 소프트웨어에 대한 시제품을 만들어 최종 결과물을 예측하는 모형이다.

본격적인 개발을 하기 전에 미리 만들어보는 시제품을 프로토타입이라고 하죠? 프로토타입을 만들어 최종 결과물을 예측해 볼 수 있는 모형이 뭐였죠?

출제예상
3. 다음 중 애자일 프로세스 모델이 아닌 것은?

① XP(eXtreme Programming)
② ASD(Adaptive Software Development)
③ 스크럼(Scrum)
④ SM(Spiral Model)

이중에 전통적인 개발 모델이 하나 섞여 있네요. 잘 찾아보세요.

출제예상
4. 애자일(Agile) 선언문에 대한 설명으로 옳지 않은 것은?

① 계획에 따르기보다는 변화에 대응하는 것에 더 가치를 둔다.
② 방대한 문서보다 제대로 실행되는 소프트웨어에 더 가치를 둔다.
③ 프로세서와 도구보다 개인과 그들의 협업에 더 가치를 둔다.
④ 고객과의 협력보다는 계약 협상에 더 가치를 둔다.

애자일은 계약 협상 보다는 고객과의 협력에 더 가치를 둡니다.

▶ 정답 : 1. ④ 2. ④ 3. ④ 4. ④

SECTION 002 ● LIVE

스크럼(Scrum) 기법

전문가의 조언

스크럼이란 럭비 경기에서 양 팀이 서로 대치해 있는 대형을 일컫는 것으로 팀의 중요성을 강조하는 용어입니다. 먼저 스크럼의 개념을 이해하고 스크럼 팀의 구성원과 각 구성원들의 역할을 잘 기억해 두세요.

1 스크럼의 개요

스크럼이란 럭비에서 반칙으로 경기가 중단된 경우 양 팀의 선수들이 럭비공을 가운데 두고 상대팀을 밀치기 위해 서로 대치해 있는 대형을 말한다. 스크럼은 이처럼 팀이 중심이 되어 개발의 효율성을 높인다는 의미가 내포된 용어이다.

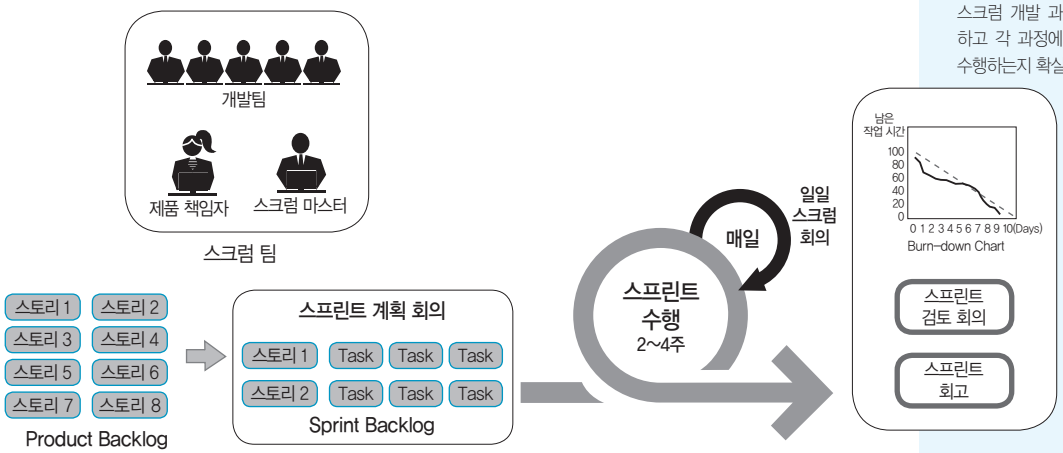
- 스크럼은 팀원 스스로가 스크럼 팀을 구성(self-organizing)해야 하며, 개발 작업에 관한 모든 것을 스스로 해결(cross-functional)할 수 있어야 한다.
- 스크럼 팀은 제품 책임자, 스크럼 마스터, 개발팀으로 구성된다.
- **제품 책임자(PO; Product Owner)**
  - 이해관계자*들 중 개발될 제품에 대한 이해도가 높고, 요구사항을 책임지고 의사 결정할 사람으로 선정하는데, 주로 개발 의뢰자나 사용자가 담당한다.
  - 이해관계자들의 의견을 종합하여 제품에 대한 요구사항을 작성하는 주체다.
  - 요구사항이 담긴 백로그(Backlog)*를 작성하고 백로그에 대한 우선순위를 지정한다.
  - 팀원들이 백로그에 스토리*를 추가할 수는 있지만 우선순위를 지정할 수는 없다.
  - 제품에 대한 테스트를 수행하면서 주기적으로 요구사항의 우선순위를 갱신한다.
- **스크럼 마스터(SM; Scrum Master)**
  - 스크럼 팀이 스크럼을 잘 수행할 수 있도록 객관적인 시각에서 조언을 해주는 가이드 역할을 수행한다. 팀원들을 통제하는 것이 목표가 아니다.
  - 일일 스크럼 회의를 주관하여 진행 사항을 점검하고, 개발 과정에서 발생된 장애 요소를 공론화하여 처리한다.
- **개발팀(DT; Development Team)**
  - 제품 책임자와 스크럼 마스터를 제외한 모든 팀원으로, 개발자 외에도 디자이너, 테스터 등 제품 개발을 위해 참여하는 모든 사람이 대상이 된다.
  - 보통 최대 인원은 7~8명이 적당하다.

이해관계자(利害關係者, Stakeholder)
소프트웨어 개발과 관련해서 이해관계자는 소프트웨어 개발 의뢰자, 소프트웨어 개발자, 소프트웨어 사용자 등입니다.

백로그(Backlog)
백로그란 제품 개발에 필요한 요구사항을 모두 모아 우선순위를 부여해 놓은 목록을 말합니다.

스토리(Story)
백로그에 담겨질 요구사항은 단어 형태로 표현된 것이 아니라 '고객은 상품 주문을 위해 로그인을 수행해야 한다.'와 같이 이야기를 서술하는 형태로 표현합니다. 그래서 백로그에 작성되는 요구사항을 스토리 또는 사용자 스토리라고 합니다.

2 스크럼 개발 프로세스



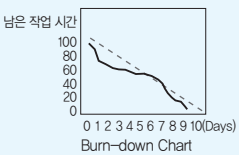
- **제품 백로그(Product Backlog)**
  - 제품 개발에 필요한 모든 요구사항(User Story)을 우선순위에 따라 나열한 목록이다.
  - 개발 과정에서 새롭게 도출되는 요구사항으로 인해 지속적으로 업데이트된다.
  - 제품 백로그에 작성된 사용자 스토리를 기반으로 전체 일정 계획인 릴리즈 계획(Release Plan)을 수립한다.
- **스프린트 계획 회의(Sprint Planning Meeting)**
  - 제품 백로그 중 이번 스프린트에서 수행할 작업을 대상으로 단기 일정을 수립하는 것이다.
  - 스프린트에서 처리할 요구사항(User Story)을 개발자들이 나눠서 작업할 수 있도록 태스크(Task)라는 작업 단위로 분할한 후 개발자별로 수행할 작업 목록인 스프린트 백로그(Sprint Backlog)를 작성한다.
- **스프린트(Sprint)**
  - 실제 개발 작업을 진행하는 과정으로, 보통 2 ~ 4주 정도의 기간 내에서 진행한다.
  - 스프린트 백로그에 작성된 태스크를 대상으로 작업 시간(양)을 추정한 후 개발 담당자에게 할당한다.
  - 태스크를 할당할 때는 개발자가 원하는 태스트를 직접 선별하여 담당할 수 있도록 하는 것이 좋다.
  - 개발 담당자에게 할당된 태스크는 보통 할 일(To Do), 진행 중(In Progress), 완료(Done)의 상태를 갖는다.
- **일일 스크럼 회의(Daily Scrum Meeting)**
  - 모든 팀원이 매일 약속된 시간에 약 15분 정도의 짧은 시간동안 진행 상황을 점검한다.
  - 회의는 보통 서서 진행하며, 남은 작업 시간은 소멸 차트(Burn-down Chart)*에 표시한다.

전문가의 조언

스크럼 개발 과정의 순서를 기억하고 각 과정에서는 무슨 작업을 수행하는지 확실히 파악해 두세요.

소멸 차트(Burn-down Chart)

소멸 차트는 해당 스프린트에서 수행할 작업의 진행 상황을 확인할 수 있도록 시간의 경과에 따라 남은 작업 시간을 그래프로 표현한 것입니다. 초기에 추정했던 전체 작업 시간은 작업이 진행될수록 점점 줄어(Burn-down) 들게 됩니다.



– 스크럼 마스터는 발견된 장애 요소를 해결할 수 있도록 도와준다.

- **스프린트 검토 회의(Sprint Review)**
  - 부분 또는 전체 완성 제품이 요구사항에 잘 부합되는지 사용자가 포함된 참석자 앞에서 테스팅을 수행한다.
  - 스프린트의 한 주당 한 시간 내에서 진행한다.
  - 제품 책임자(Product Owner)는 개선할 사항에 대한 피드백을 정리한 후 다음 스프린트에 반영할 수 있도록 제품 백로그를 업데이트한다.
- **스프린트 회고(Sprint Retrospective)**
  - 스프린트 주기를 되돌아보며 정해놓은 규칙을 잘 준수했는지, 개선할 점은 없는지 등을 확인하고 기록한다.
  - 해당 스프린트가 끝난 시점에서 수행하거나 일정 주기로 수행한다.

기출문제 따라잡기 Section 002

출제예상

1. 다음이 설명하는 프로세스 모델은 무엇인가?

- 팀원들이 스스로 팀을 구성하며, 개발 작업의 모든 것을 스스로 해결할 수 있어야 한다.
- 개발에 필요한 요구사항에 우선순위를 부여한 제품기능 목록(Product Backlog)을 작성한다.
- 개발 주기를 의미하는 스프린트는 2 ~ 4주 정도의 기간으로 진행한다.
- 스프린트 회고(Retrospective)를 통해 스프린트 동안 발생한 문제점을 파악하고 이에 대한 해결 방안을 모색한다.

① 익스트림 프로그래밍(XP)
② 크리스털(Crystal)
③ 스프린트(Sprint)
④ 스크럼(Scrum)

스프린트, 제품기능 목록, 스프린트 회고만 보면 어떤 프로세스 모델인지 알 수 있겠죠?

출제예상

2. 스크럼의 팀 구성 요소 중 이해관계자들의 의견을 종합하여 백로그(Backlog)를 작성하는 주체는 누구인가?

① 스크럼 마스터(SM)
② 제품 책임자(PO)
③ 고객(Customer)
④ 개발팀(DT)

백로그(Backlog)는 제품에 대한 요구사항입니다. 보기 중 제품과 관련된 주체를 찾으면 되겠죠?

출제예상

3. 다음의 스크럼(Scrum) 개발 과정을 진행 순서에 맞게 올바르게 나열한 것은?

ㄱ. 스프린트(Sprint)
ㄴ. 스프린트 회고(Sprint Retrospective)
ㄷ. 일일 스크럼 회의(Daily Scrum Meeting)
ㄹ. 스프린트 검토 회의(Sprint Review)
ㅁ. 스프린트 계획 회의(Sprint Planning Meeting)

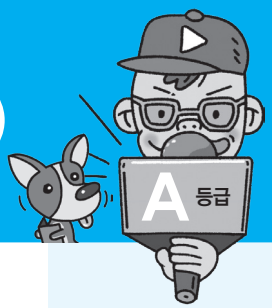
① ㅁ → ㄷ → ㄱ → ㄴ → ㄹ
② ㅁ → ㄱ → ㄷ → ㄹ → ㄴ
③ ㅁ → ㄷ → ㄱ → ㄹ → ㄴ
④ ㅁ → ㄹ → ㄱ → ㄴ → ㄷ

계획한 내용을 토대로 일정 기간 동안 스프린트를 수행하면서 진행 상황을 매일 점검하고 하나의 스프린트가 끝나면 검토한 후 진행을 되돌아봅니다.

▶ 정답 : 1. ④ 2. ② 3. ②

SECTION 003 LIVE

XP(eXtreme Programming) 기법

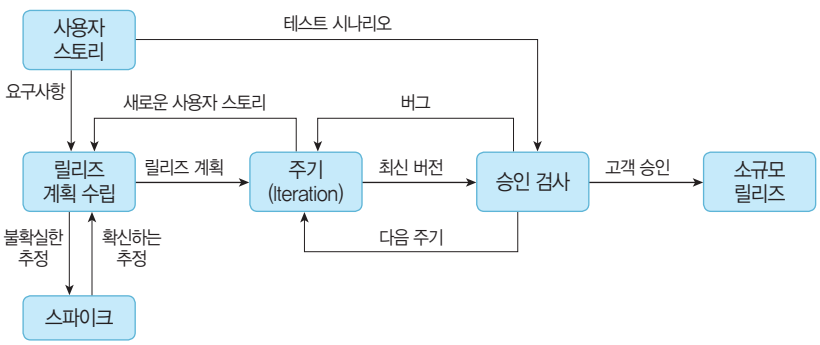


1 XP(eXtreme Programming)

XP(eXtreme Programming)는 수시로 발생하는 고객의 요구사항에 유연하게 대응하기 위해 고객의 참여와 개발 과정의 반복을 극대화하여 개발 생산성을 향상시키는 방법이다.

- XP는 짧고 반복적인 개발 주기, 단순한 설계, 고객의 적극적인 참여를 통해 소프트웨어를 빠르게 개발하는 것을 목적으로 한다.
- 릴리즈*의 기간을 짧게 반복하면서 고객의 요구사항 반영에 대한 가시성*을 높인다.
- 릴리즈 테스트마다 고객을 직접 참여시킴으로써 요구한 기능이 제대로 작동하는지 고객이 직접 확인할 수 있다.
- 비교적 소규모 인원의 개발 프로젝트에 효과적이다.
- **XP의 5가지 핵심 가치** : 의사소통(Communication), 단순성(Simplicity), 용기(Courage), 존중(Respect), 피드백(Feedback)

2 XP 개발 프로세스



- **사용자 스토리(User Story)**
  - 고객의 요구사항을 간단한 시나리오로 표현한 것이다.
  - 내용은 기능 단위로 구성하며, 필요한 경우 간단한 테스트 사항(Test Case)도 기재한다.
- **릴리즈 계획 수립(Release Planning)**
  - 몇 개의 스토리가 적용되어 부분적으로 기능이 완료된 제품을 제공하는 것을 릴리즈라고 한다.
  - 부분 혹은 전체 개발 완료 시점에 대한 일정을 수립한다.

전문가의 조언

몇 개의 요구사항이 적용된 일부 기능이 완성될 때마다 이를 고객에게 보여주고 이에 대한 반응을 확인하는 과정을 최종 제품이 완성될 때까지 지속적으로 반복한다는 XP의 기본 원리를 생각하면서 개념과 특징을 정리하세요.

릴리즈(Release)
릴리즈는 몇 개의 요구사항이 적용되어 부분적으로 기능이 완료된 제품을 제공하는 것을 말합니다.

가시성(Visibility)
일반적으로 가시성이란 대상을 확인할 수 있는 정도를 의미합니다. 릴리즈 기간을 짧게 반복하면서 개발 과정에서 제품 소프트웨어의 일부 기능이 구현될 때마다 고객에게 이를 확인시켜주면, 고객은 요구사항이 잘 반영되고 있음을 직접적으로 알 수 있다는 의미입니다.

전문가의 조언

XP 개발 과정의 순서를 기억하고 각 과정에서는 무슨 작업을 수행하는지 확실히 파악해 두세요.

- **스파이크(Spike)**
  - 요구사항의 신뢰성을 높이고 기술 문제에 대한 위험을 감소시키기 위해 별도로 만드는 간단한 프로그램이다.
  - 처리할 문제 외의 다른 조건은 모두 무시하고 작성한다.
- **이터레이션(Iteration)**
  - 하나의 릴리즈를 더 세분화 한 단위를 이터레이션(Iteration)이라고 한다.
  - 일반적으로 1~3주 정도의 기간으로 진행된다.
  - 이 기간 중에 새로운 스토리가 작성될 수 있으며, 작성된 스토리는 진행 중인 이터레이션 혹은 다음 이터레이션에 포함될 수 있다.
- **승인 검사(Acceptance Test, 인수 테스트)**
  - 하나의 이터레이션 안에서 계획된 릴리즈 단위의 부분 완료 제품이 구현되면 수행하는 테스트이다.
  - 사용자 스토리 작성 시 함께 기재한 테스트 사항에 대해 고객이 직접 수행한다.
  - 테스트 과정에서 발견한 오류 사항은 다음 이터레이션에 포함한다.
  - 테스트 이후 새로운 요구사항이 작성되거나 요구사항의 상대적 우선순위가 변경될 수 있다.
  - 테스트가 완료되면 다음 이터레이션을 진행한다.
- **소규모 릴리즈(Small Release)**
  - 릴리즈를 소규모로 하게 되면, 고객의 반응을 기능별로 확인할 수 있어, 고객의 요구사항에 좀 더 유연하게 대응할 수 있다.
  - 계획된 릴리즈 기간 동안 진행된 이터레이션이 모두 완료되면 고객에 의한 최종 테스트를 수행한 후 릴리즈, 즉 최종 결과물을 고객에게 전달한다.
  - 릴리즈가 최종 완제품이 아닌 경우 다음 릴리즈 일정에 맞게 개발을 계속 진행한다.

XP의 주요 실천 방법(Practice)

실천 방법	내용
Pair Programming (짝 프로그래밍)	다른 사람과 함께 프로그래밍을 수행함으로써 개발에 대한 책임을 공동으로 나눠 갖는 환경을 조성합니다.
Test-Driven Development (테스트 주도 개발)	• 개발자가 실제 코드를 작성하기 전에 테스트 케이스를 먼저 작성하므로 자신이 무엇을 해야할지를 정확히 파악합니다. • 테스트가 지속적으로 진행될 수 있도록 자동화된 테스팅 도구(구조, 프레임워크)를 사용합니다.
Whole Team (전체 팀)	개발에 참여하는 모든 구성원(고객 포함)들은 각자 자신의 역할이 있고 그 역할에 대한 책임을 가져야 합니다.
Continuous Integration (계속적인 통합)	모듈 단위로 나눠서 개발된 코드들은 하나의 작업이 마무리될 때마다 지속적으로 통합됩니다.

Design Improvement (디자인 개선) 또는 Refactoring (리팩토링)	프로그램 기능의 변경 없이, 단순화, 유연성 강화 등을 통해 시스템을 재구성합니다.
Small Releases (소규모 릴리즈)	릴리즈 기간을 짧게 반복함으로써 고객의 요구 변화에 신속히 대응할 수 있습니다.

기출문제 따라잡기 Section 003

출제예상

1. 익스트림 프로그래밍 테스팅에 대한 설명으로 옳지 않은 것은?

① 테스트는 최종 완제품을 고객에게 전달하기 바로 전에 수행한다.
② 각 사용자 스토리에 대해 테스트 케이스를 작성한다.
③ 실제 코드를 작성하기 전에 테스트 케이스를 우선 작성한다.
④ 자동화된 테스팅 도구 사용을 권장한다.

XP의 특징 중 하나는 짧은 개발 주기를 반복하며 그때마다 만들어진 제품을 테스트하여 새로이 발생되는 요구사항을 적극 반영하는 것입니다.

출제예상

2. 다음 중 익스트림 프로그래밍(XP)에 대한 설명으로 옳지 않은 것은?

① 테스트 이후 새로운 요구사항이 작성되거나 요구사항의 상대적 우선순위가 변경될 수 있다.
② 고객의 요구사항에 좀 더 유연하게 대응할 수 있도록 릴리즈 규모를 크게 한다.
③ 하나의 릴리즈를 더 세분화 한 단위를 이터레이션(Iteration)이라고 한다.
④ 모든 개발자들이 전체 코드에 대한 공동 책임을 가지며, 개발자 누구든지 어떤 코드라도 변경할 수 있다.

부분적으로 기능이 완료된 제품을 제공하는 것을 릴리즈라고 합니다. 고객의 요구사항에 좀 더 유연하게 대응하려면 릴리즈로 확인하는 기능이 좀 더 세분화되어야 합니다.

출제예상

3. 애자일 방법 중에 제일 많이 알려진 것이 익스트림 프로그래밍(eXtreme Programming : XP)인데, 다음 중 XP와 가장 연관성이 적은 것은?

① Whole Team
② Large Releases
③ Pair Programming
④ Continuous Integration

XP의 주요 실천 방법 중 하나는 고객의 요구 변화에 신속히 대응할 수 있도록 릴리즈 기간을 짧게 반복하는 것입니다.

출제예상

4. 익스트림 프로그래밍(XP)의 5가지 핵심 가치에 속하지 않는 것은?

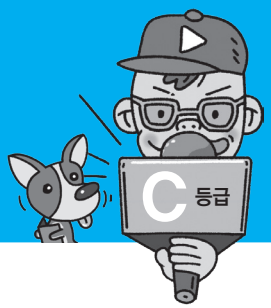
① 의사소통(Communication)
② 단순성(Simplicity)
③ 용기(Courage)
④ 효율성(Efficiency)

익스트림 프로그래밍(XP)의 5가지 핵심 가치는 보기로 제시된 내용 외에 존중(Respect)과 피드백(Feedback)이 있습니다.

▶ 정답 : 1. ① 2. ② 3. ② 4. ④

SECTION 004 LIVE

현행 시스템 파악

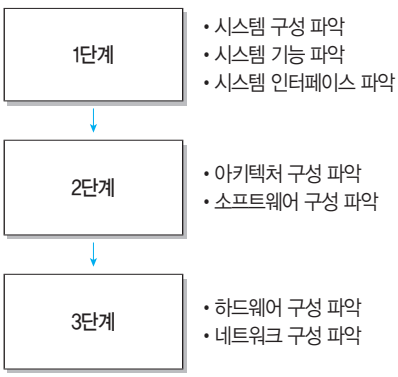


전문가의 조언

향후 개발하려는 시스템의 개발 범위를 명확히 설정하려면 우선 현행 시스템이 어떻게 구성되어 있는지를 파악해야 합니다. 제시된 각 과정별로 무엇을 파악하고 어떻게 정리하는지 알아두세요.

1 현행 시스템 파악 절차

새로 개발하려는 시스템의 개발 범위를 명확히 설정하기 위해 현행 시스템의 구성과 제공 기능, 시스템 간의 전달 정보, 사용되는 기술 요소, 소프트웨어, 하드웨어, 그리고 네트워크의 구성 등을 파악한다.



2 시스템 구성 파악

현행 시스템의 구성은 조직의 주요 업무를 담당하는 기간 업무와 이를 지원하는 지원 업무로 구분하여 기술한다.

- 조직 내에 있는 모든 정보시스템의 현황을 파악할 수 있도록 각 업무에 속하는 단위 업무 정보시스템들의 명칭, 주요 기능들을 명시한다.

예 금융기관의 여신관리 업무와 고객관리 업무 시스템 현황

구분	시스템명	시스템 내용	비고
여신관리 업무	여신기획 관리 시스템	여신기획 관리를 위한 여신요율 책정, 연간 여신운용지침 수립 등의 기능을 제공하는 시스템	
	여신상담 관리 시스템	여신상담 관리를 위한 거래처정보 관리, 여신상담, 대출의향서 발급 기능을 제공하는 시스템	
고객관리 업무	고객등록 처리 시스템	고객의 기본 정보를 관리하기 위한 등록, 변경, 조회 삭제 등의 기능을 제공하는 시스템	

3 시스템 기능 파악

현행 시스템의 기능은 단위 업무 시스템이 현재 제공하는 기능들을 주요 기능과 하부 기능, 세부 기능으로 구분하여 계층형으로 표시한다.

예 여신상담 관리 시스템의 주요 기능과 하부, 세부 기능

단위 업무 시스템	Level 1 주요 업무 기능	Level 2 세부 업무 기능	Level 3 세부 업무 기능 활동	비고
여신상담 관리	여신기획 관리	여신요율 책정		
		연간 여신운용지침 수립		
	여신상담 관리	거래처정보 관리	거래처정보 등록	
			신용정보 관리	
		여신상담	대상거래 파악	
			상담결과 보고	
			신용조사 의뢰	

4 시스템 인터페이스 파악

현행 시스템의 인터페이스에는 단위 업무 시스템 간에 주고받는 데이터의 종류, 형식, 프로토콜, 연계 유형, 주기 등을 명시한다.

- 데이터를 어떤 형식*으로 주고받는지, 통신규약*은 무엇을 사용하는지, 연계 유형*은 무엇인지 등을 반드시 고려해야 한다.

예 여신상담 관리 시스템의 인터페이스 현황

송신 시스템	수신 시스템	연동 데이터	연동 형식	통신규약	연계 유형	주기
여신상담 관리 시스템	여신관리센터	연체 정보	XML	TCP/IP	EAI	하루(일)
여신상담 관리 시스템	여신금융협회	부도 정보	XML	X.25	FEP	수시

- **데이터 형식** : XML, 고정 포맷, 가변 포맷 등
- **통신규약** : TCP/IP, X.25 등
- **연계 유형** : EAI, FEP 등

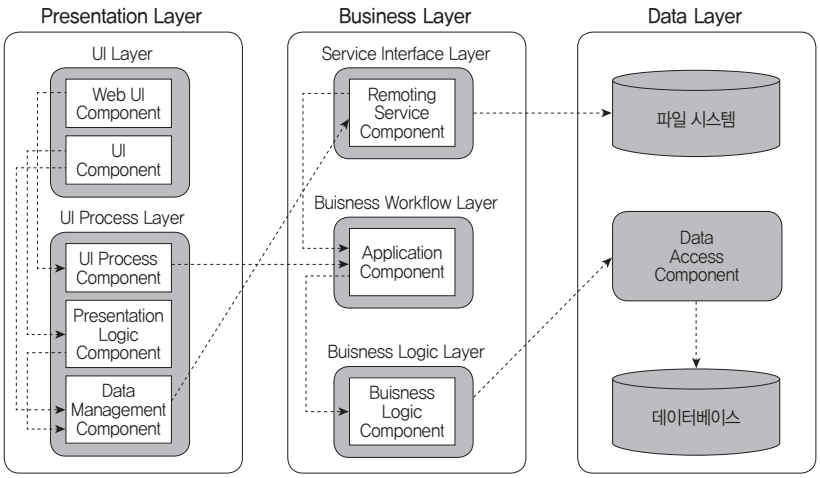
5 아키텍처 구성 파악

현행 시스템의 아키텍처* 구성은 기간 업무 수행에 어떠한 기술 요소들이 사용되는지 최상위 수준에서 계층별로 표현한 아키텍처 구성도로 작성한다.

- 아키텍처가 단위 업무 시스템별로 다른 경우에는 가장 핵심이 되는 기간 업무 처리 시스템을 기준으로 표현한다.

시스템 아키텍처(System Architecture)
시스템 아키텍처는 시스템 내부에서 각각의 하위 시스템들이 어떠한 관계로 상호 작용하는지 파악할 수 있도록 구성이나 동작 원리를 표현한 것을 말합니다.

예 회원 정보 관리 시스템 아키텍처 구성도



6 소프트웨어 구성 파악

소프트웨어 구성에는 단위 업무 시스템별로 업무 처리를 위해 설치되어 있는 소프트웨어들의 제품명, 용도, 라이선스 적용 방식, 라이선스 수 등을 명시한다.

- 시스템 구축비용 면에서 소프트웨어 비용이 적지 않은 비중을 차지하므로, 상용 소프트웨어*의 경우 라이선스 적용 방식*의 기준과 보유한 라이선스의 파악이 중요하다.

예 단위 업무 시스템별 소프트웨어 현황

구분	시스템명	SW 제품명	용도	라이선스 적용 방식	라이선스 수
여신관리 업무	거래처정보 관리 시스템	Apache Tomcat	WAS	오픈 소스 Apache License	1
		MySQL	데이터베이스	GPL 또는 상용	1
		UNIX	운영체제	GNU GPL	1
	대출의향서 발급 시스템	Sage	ERP	상용	1
		Oracle	데이터베이스	GPL 또는 상용	1
		Windows 10	운영체제	DSP	5

- **상용 소프트웨어** : 정식으로 대가를 지불하고 사용해야 하는 것으로, 해당 소프트웨어의 모든 기능을 정상적으로 사용할 수 있습니다.
- **라이선스 적용 방식** : 사이트, 서버, 프로세서(CPU), 동시 사용, 코어(Core), 사용자 수 등

※ **코어(Core)** : 각종 연산을 수행하는 CPU의 핵심 요소로, 코어의 개수에 따라 싱글 코어, 듀얼 코어, 트리플 코어 등으로 구분하며, 코어의 개수가 많을수록 처리속도가 빨라집니다.

7 하드웨어 구성 파악

하드웨어 구성에는 단위 업무 시스템들이 운용되는 서버의 주요 사양*과 수량, 그리고 이중화의 적용 여부를 명시한다.

- 서버의 이중화*는 기간 업무의 서비스 기간, 장애 대응 정책에 따라 필요 여부가 결정된다.
- 현행 시스템에 이중화가 적용된 경우 대부분 새로 구성될 시스템에도 이중화가 필요하므로 이로 인한 비용 증가와 시스템 구축 난이도가 높아질 가능성을 고려해야 한다.

예 단위 업무 시스템별 하드웨어 현황

구분	시스템명	서버 용도	제품명	주요 사양	수량	이중화
여신 관리 업무	여신정보 관리 시스템	AP 서버	HPE ProLiant DL360 Gen10 서버	• CPU : 2.6GHz 8core/24T × 2ea, 30MB Cache • Memory : 8GB RDIMM, 2133MT/s × 8ea • HDD : 300GB 15k RPM SAS 2.5″ × 3ea + 600GB 15k RPM SAS 2.5″ × 4ea • RAID Controller : 2GB 캐시	1	N
		DB 서버	HPE Integrity Superdome 2 서버	• CPU : 3.2GHz 12core/24T × 2ea, 50MB Cache • Memory : 16GB RDIMM, 2133MT/s × 8ea • HDD : 1TB 15k RPM SAS 2.5″ × 3ea + 2TB 15k RPM SAS 2.5″ × 4ea • RAID Controller : 4GB 캐시	1	Y

8 네트워크 구성 파악

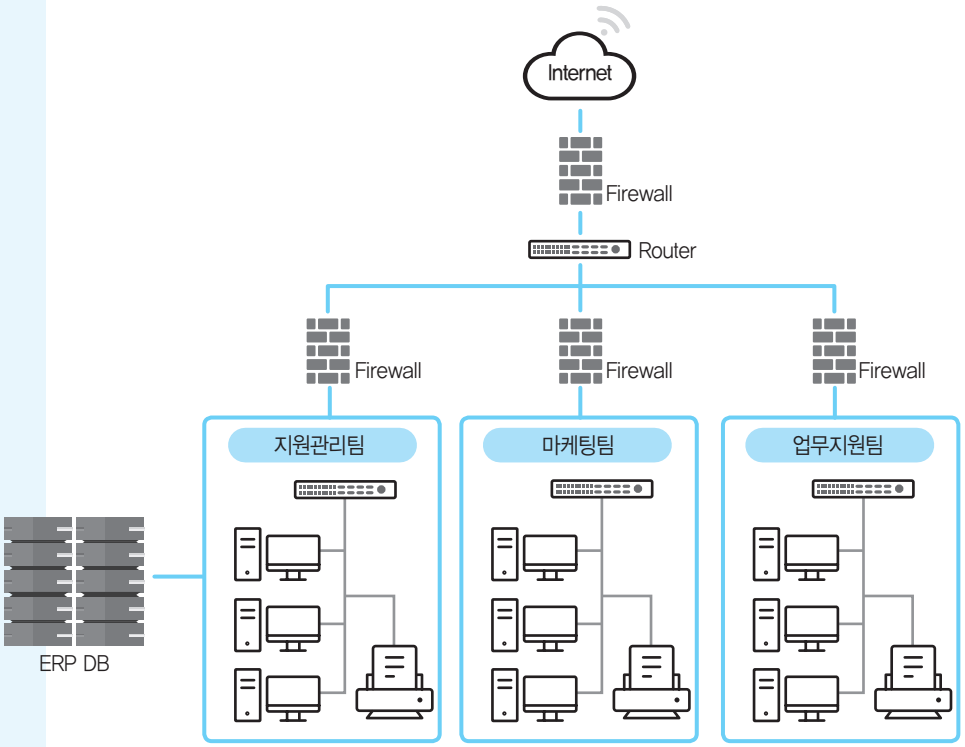
네트워크 구성은 업무 시스템들의 네트워크 구성을 파악할 수 있도록 서버의 위치, 서버 간의 네트워크 연결 방식을 네트워크 구성도로 작성한다.

- 네트워크 구성도를 통해 서버들의 물리적인 위치 관계를 파악할 수 있고 보안 취약성을 분석하여 적절한 대응을 할 수 있다.
- 네트워크에 장애가 발생한 경우 발생 원인을 찾아 복구하기 위한 용도로 활용될 수 있다.

서버의 주요 사양
서버의 CPU 처리 속도, 메모리 크기, 하드디스크의 용량 등을 파악해서 명시합니다.

서버의 이중화(Replication)
서버의 이중화란 운용 서버의 장애 시 대기 서버로 서비스를 계속 유지할 수 있도록, 운용 서버의 자료 변경이 예비 서버에도 동일하게 복제되도록 관리하는 것을 의미합니다.

예 자원관리팀, 마케팅팀, 업무지원팀의 인터넷 접속을 위한 네트워크 구성도



따라잡기

기출문제 따라잡기 Section 004

출제예상

1. 다음 중 현행 시스템 파악 과정에 대한 설명으로 잘못된 것은?

① 시스템 구성은 조직의 주요 업무를 담당하는 기간 업무와 이를 지원하는 지원 업무로 구분하여 기술한다.

② 소프트웨어 구성을 파악할 때 상용 소프트웨어의 경우 라이선스 적용 방식의 기준과 보유한 라이선스의 파악이 중요하다.

③ 아키텍처 구성을 파악할 때는 단위 업무 시스템 간에 주고받는 데이터의 종류, 형식, 프로토콜, 연계 유형, 주기 등을 명시한다.

④ 네트워크 구성을 파악하면 서버들의 물리적인 위치 관계를 파악할 수 있고 보안 취약성을 분석하여 이에 대한 적절한 대응을 할 수 있다.

아키텍처 구성은 기간 업무 수행에 어떠한 기술 요소가 사용되었는지 최상위 수준에서 계층별로 표현한 것입니다. 단위 업무 시스템 간에 주고받는 데이터의 종류, 형식, 프로토콜, 연계 유형, 주기 등은 시스템 인터페이스 파악 시 기술할 내용입니다.

출제예상

2. 다음 중 현행 시스템 파악 과정에 속하지 않는 것은?

① 시스템 구성 파악

② 시스템 인터페이스 파악

③ 네트워크 구성 파악

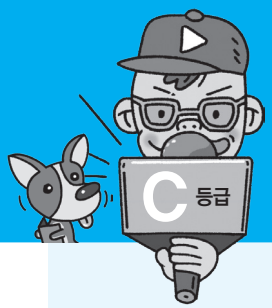
④ 업무에 따른 사무실 배치 파악

현행 시스템 파악은 말 그대로 현재 사용하고 있는 정보 시스템에 대한 현황을 파악하는 것입니다. 시스템과 거리가 먼 것을 찾아보세요.

▶ 정답 : 1. ③ 2. ④

SECTION 005 LIVE

개발 기술 환경 파악



1 개발 기술 환경의 정의

개발하고자 하는 소프트웨어와 관련된 운영체제(Operating System), 데이터베이스 관리 시스템(Database Management System), 미들웨어(Middle Ware)* 등을 선정할 때 고려해야 할 사항을 기술하고, 오픈 소스 사용 시 주의해야 할 내용을 제시한다.

2 운영체제(OS, Operating System)

운영체제는 컴퓨터 시스템의 자원*들을 효율적으로 관리하며, 사용자가 컴퓨터를 편리하고 효율적으로 사용할 수 있도록 환경을 제공하는 소프트웨어이다.

- 컴퓨터 사용자와 컴퓨터 하드웨어 간의 인터페이스로서 동작하는 시스템 소프트웨어의 일종으로, 다른 응용 프로그램이 유용한 작업을 할 수 있도록 환경을 제공해준다.
- 컴퓨터 운영체제의 종류에는 Windows, UNIX, Linux, Mac OS 등이, 모바일 운영체제에는 iOS, Android 등이 있다.

3 운영체제 관련 요구사항 식별 시 고려사항

운영체제 관련 요구사항 식별 시 다음과 같은 사항을 고려해야 한다.

구분	내용
가용성*	• 시스템의 장시간 운영으로 인해 발생할 수 있는 운영체제 고유의 장애 발생 가능성 • 메모리 누수*로 인한 성능 저하 및 재가동 • 보안상 발견된 허점을 보완하기 위한 지속적인 패치 설치로 인한 재가동 • 운영체제의 결함 등으로 인한 패치 설치를 위한 재가동
성능	• 대규모 동시 사용자 요청에 대한 처리 • 대규모 및 대용량 파일 작업에 대한 처리 • 지원 가능한 메모리 크기(32bit, 64bit)
기술 지원	• 제작업체의 안정적인 기술 지원 • 여러 사용자들 간의 정보 공유 • 오픈 소스* 여부(Linux)
주변 기기	• 설치 가능한 하드웨어 • 여러 주변기기 지원 여부
구축 비용	• 지원 가능한 하드웨어 비용 • 설치할 응용 프로그램의 라이선스 정책 및 비용 • 유지관리 비용 • 총 소유 비용(TCO)*

전문가의 조언

- 운영체제, 데이터베이스 관리 시스템, 웹 애플리케이션 서버, 그리고 오픈 소스의 개념과 특징을 정리하세요. 각각에 대한 요구사항 식별 시 고려사항은 내용이 어렵지 않아 한번 읽어보는 것만으로도 충분히 이해할 수 있으니 가볍게 읽으면서 정리하세요.
- 소프트웨어 개발과 관련된 미들웨어에는 다양한 종류가 있으나 여기서는 미들웨어 중 웹 애플리케이션 서버(WAS: Web Application Server)와 관련된 고려사항만 다루겠습니다.

미들웨어(Middle Ware)
미들웨어는 운영체제와 해당 운영체제에 의해 실행되는 응용 프로그램 사이에서 운영체제가 제공하는 서비스 이외에 추가적인 서비스를 제공하는 소프트웨어입니다.

자원
자원이란 시스템에서 사용할 수 있는 CPU, 주기억장치, 보조기억장치, 프린터, 파일 및 정보 등을 의미합니다.

- **가용성** : 프로그램이 주어진 시점에서 요구사항에 따라 운영될 수 있는 능력
- **메모리 누수** : 응용 프로그램이 더 이상 사용하지 않는 메모리를 반환하지 않고 계속 점유하고 있는 현상
- **오픈 소스** : 누구나 별다른 제한 없이 사용할 수 있도록 소스 코드를 공개해 무료로 사용이 가능한 소프트웨어
- **총 소유 비용(TCO; Total Cost of Ownership)** : 어떤 자산을 획득하려고 할 때 지정된 기간 동안 발생할 수 있는 모든 직간접 비용들로, 하드웨어 구매, 소프트웨어 구매 및 라이선스, 설치, 교육, 지속적인 기술 지원, 유지보수, 가동 중지로 인한 손실, 에너지 등의 비용이 있습니다.

4 데이터베이스 관리 시스템(DBMS)

DBMS(DataBase Management System)는 사용자와 데이터베이스 사이에서 사용자의 요구에 따라 정보를 생성해 주고, 데이터베이스를 관리해 주는 소프트웨어이다.

- DBMS는 기존의 파일 시스템이 갖는 데이터의 종속성과 중복성의 문제를 해결하기 위해 제안된 시스템으로, 모든 응용 프로그램*들이 데이터베이스를 공용할 수 있도록 관리해 준다.
- DBMS는 데이터베이스의 구성, 접근 방법, 유지관리에 대한 모든 책임을 진다.
- DBMS의 종류에는 Oracle, IBM DB2, Microsoft SQL Server, MySQL, SQLite, MongoDB, Redis 등이 있다.

응용 프로그램
응용 프로그램은 조직이나 기업체에서 특정 부서에 정보를 제공하기 위해 데이터베이스에 접근하여 운영되는 프로그램으로, 데이터베이스는 여러 개의 응용 프로그램들이 공동으로 사용합니다.

5 DBMS 관련 요구사항 식별 시 고려사항

DBMS 관련 요구사항 식별 시 다음과 같은 사항을 고려해야 한다.

구분	내용
가용성	• 시스템의 장시간 운영으로 인해 발생할 수 있는 운영체제 고유의 장애 발생 가능성 • DBMS의 결함 등으로 인한 패치 설치를 위한 재가동 • 백업이나 복구의 편의성 • DBMS 이중화 및 복제 지원
성능	• 대규모 데이터 처리 성능(분할 테이블 지원 여부) • 대용량 트랜잭션 처리 성능 • 튜닝 옵션의 다양한 지원 • 최소화된 설정과 비용 기반 질의 최적화* 지원
기술 지원	• 제작업체의 안정적인 기술 지원 • 여러 사용자들 간의 정보 공유 • 오픈 소스 여부
상호 호환성	• 설치 가능한 운영체제의 종류 • JDBC*, ODBC*와의 호환 여부
구축 비용	• 라이선스 정책 및 비용 • 유지관리 비용 • 총 소유 비용(TCO)

- **비용 기반 질의 최적화** : 사용자의 질의에 대한 최적의 실행 방법을 결정하기 위한 것으로, 질의에 대한 다양한 실행 방법을 만들고 각각의 방법에 대해 비용을 추정합니다. 비용 추정은 실행에 필요한 소요 시간과 자원 사용량을 기준으로 추정하며, 추정된 비용이 가장 최소인 방법을 선택하게 됩니다.
- JDBC(Java DataBase Connectivity) : 자바에서 DB에 접근하여 데이터를 조회, 삽입, 수정, 삭제할 수 있도록 자바와 DB를 연결해 주는 인터페이스
- ODBC(Open DataBase Connectivity) : 응용 프로그램에서 DB에 접근하여 데이터를 조회, 삽입, 수정, 삭제할 수 있도록 응용 프로그램과 DB를 연결해 주는 표준 인터페이스

6 웹 애플리케이션 서버(WAS; Web Application Server)

웹 애플리케이션 서버는 정적인 콘텐츠 처리를 하는 웹 서버와 달리 사용자의 요구에 따라 변하는 동적인 콘텐츠를 처리하기 위해 사용되는 미들웨어이다.

- 데이터 접근, 세션 관리, 트랜잭션 관리 등을 위한 라이브러리를 제공한다.
- 주로 데이터베이스 서버와 연동해서 사용한다.
- 웹 애플리케이션 서버의 종류에는 Tomcat, GlassFish, JBoss, Jetty, JEUS, Resin, WebLogic, WebSphere 등이 있다.

7 웹 애플리케이션 서버(WAS) 관련 요구사항 식별 시 고려사항

웹 애플리케이션 서버(WAS) 관련 요구사항 식별 시 다음과 같은 사항을 고려해야 한다.

구분	내용
가용성	• 시스템의 장시간 운영으로 인해 발생할 수 있는 고유의 장애 발생 가능성 • WAS의 결함 등으로 인한 패치 설치를 위한 재가동 • 안정적인 트랜잭션 처리 • WAS 이중화 지원
성능	• 대규모 트랜잭션 처리 성능 • 다양한 설정 옵션 지원 • 가비지 컬렉션(GC; Garbage Collection)※의 다양한 옵션
기술 지원	• 제조업체의 안정적인 기술 지원 • 여러 사용자들 간의 정보 공유 • 오픈 소스 여부
구축 비용	• 라이선스 정책 및 비용 • 유지관리 비용 • 총 소유 비용(TCO)

8 오픈 소스 사용에 따른 고려사항

오픈 소스(Open Source)는 누구나 별다른 제한 없이 사용할 수 있도록 소스 코드를 공개한 것으로 오픈 소스 라이선스를 만족하는 소프트웨어이다.

- 오픈 소스를 사용하는 경우에는 라이선스의 종류, 사용자 수, 기술의 지속 가능성 등을 고려해야 한다.

전문가의 조언

클라이언트의 웹 브라우저에서 특정 웹 사이트에 접속하면 웹 서버(Web Server)는 데이터베이스에 접속하여 해당 사이트에 포함된 각종 콘텐츠를 보여줍니다. 이러한 콘텐츠에는 텍스트나 이미지와 같이 정적인 자료도 있지만 주식 시세 정보나 날씨 위성 정보와 같이 실시간으로 변하는 동적인 자료도 있습니다. 실시간으로 변하는 동적인 자료는 웹 서버에서 직접 처리할 수 없으므로 동적인 자료 처리를 웹 애플리케이션 서버(Web Application Server)에 요청합니다. 웹 애플리케이션 서버가 JSP나 서블릿(Servlet)과 같은 프로그램을 구동하여 동적인 자료를 처리한 후 해당 정보를 웹 서버로 보내면, 웹 서버는 이를 클라이언트로 보내는 것입니다.

가비지 컬렉션(Garbage Collection)

가비지 컬렉션은 실제로는 사용되지 않으면서 가용 공간 리스트에 반환되지 않는 메모리 공간인 가비지(Garbage, 쓰레기)를 강제로 해제하여 사용할 수 있도록 하는 메모리 관리 기법입니다.

기출문제 따라잡기 Section 005

출제예상

1. 운영체제에 대한 설명으로 잘못된 것은?

① 사용자가 컴퓨터를 편리하고 효과적으로 사용할 수 있도록 환경을 제공한다.

② 컴퓨터 사용자와 컴퓨터 하드웨어 간의 인터페이스로서 동작하는 일종의 하드웨어 장치이다.

③ 다른 응용 프로그램이 유용한 작업을 할 수 있도록 환경을 제공한다.

④ 종류에는 Windows, UNIX, Linux, iOS 등이 있다.

운영체제가 소프트웨어라는 것만 알면 쉽게 풀 수 있는 문제입니다.

출제예상

2. 다음 중 데이터베이스 관리 시스템(DBMS)이 아닌 것은?

① Oracle

② MySQL

③ Microsoft SQL Server

④ Android

보기 중에 모바일 운영체제에 속하는 것이 있으니 찾아보세요.

출제예상

3. 요구사항 식별 시 고려사항 중 가용성과 관련된 내용이 아닌 것은?

① 시스템의 장시간 운영으로 인해 발생할 수 있는 고유의 장애 발생 가능성

② DBMS의 결함 등으로 인한 패치 설치를 위한 재가동

③ WAS 이중화 지원

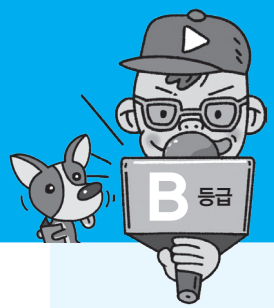
④ 설치할 응용 프로그램의 라이선스 정책 및 비용

가용성이란 프로그램이 주어진 시점에서 요구사항에 따라 운영될 수 있는 능력을 말합니다. 이를 위해 고려할 사항과 관련이 없는 것을 찾아보세요.

▶ 정답 : 1. ② 2. ④ 3. ④

SECTION 006 LIVE

요구사항 정의



1 요구사항의 개념 및 특징

요구사항은 소프트웨어가 어떤 문제를 해결하기 위해 제공하는 서비스에 대한 설명과 정상적으로 운영되는데 필요한 제약조건 등을 나타낸다.

- 요구사항은 소프트웨어 개발이나 유지 보수 과정에서 필요한 기준과 근거를 제공한다.
- 요구사항은 개발하려는 소프트웨어의 전반적인 내용을 확인할 수 있게 하므로 개발에 참여하는 이해관계자*들 간의 의사소통을 원활하게 하는 데 도움을 준다.
- 요구사항이 제대로 정의되어야만 이를 토대로 이후 과정의 목표와 계획을 수립할 수 있다.

2 요구사항의 유형

요구사항은 일반적으로 기술하는 내용에 따라 기능 요구사항(Functional requirements)과 비기능 요구사항(Non-functional requirements)으로 구분하며, 기술 관점과 대상의 범위에 따라 시스템 요구사항(System requirements)과 사용자 요구사항(User requirements)으로 나눈다.

유형	내용
기능 요구사항 (Functional requirements)	• 시스템이 무엇을 하는지, 어떤 기능을 하는지에 대한 사항 • 시스템의 입력이나 출력으로 무엇이 포함되어야 하는지, 시스템이 어떤 데이터를 저장하거나 연산을 수행해야 하는지에 대한 사항 • 시스템이 반드시 수행해야 하는 기능 • 사용자가 시스템을 통해 제공받기를 원하는 기능
비기능 요구사항 (Non-functional requirements)	• 시스템 장비 구성 요구사항 : 하드웨어, 소프트웨어, 네트워크 등의 시스템 장비 구성에 대한 요구사항 • 성능 요구사항 : 처리 속도 및 시간, 처리량, 동적 · 정적 적용량, 가용성 등 성능에 대한 요구사항 • 인터페이스 요구사항 : 시스템 인터페이스와 사용자 인터페이스에 대한 요구사항으로 다른 소프트웨어, 하드웨어 및 통신 인터페이스, 다른 시스템과의 정보 교환에 사용되는 프로토콜과의 연계도 포함하여 기술 • 데이터 요구사항 : 초기 자료 구축 및 데이터 변환을 위한 대상, 방법, 보안이 필요한 데이터 등 데이터를 구축하기 위해 필요한 요구사항 • 테스트 요구사항 : 도입되는 장비의 성능 테스트(BMT)나 구축된 시스템이 제대로 운영되는지를 테스트하고 점검하기 위한 테스트 요구사항 • 보안 요구사항 : 시스템의 데이터 및 기능, 운영 접근을 통제하기 위한 요구사항 • 품질 요구사항 : 관리가 필요한 품질 항목, 품질 평가 대상에 대한 요구사항으로 가용성*, 정합성*, 상호 호환성*, 대응성*, 신뢰성, 사용성, 유지 · 관리성, 이식성*, 확장성*, 보안성 등으로 구분하여 기술

전문가의 조언

요구사항이란 말 그대로 어떠한 문제를 해결하기 위해 필요한 조건이나 제약사항을 요구하는 것이며, 소프트웨어는 사용자의 요구사항을 충족시키기 위해 설계되고 개발됩니다. 즉 소프트웨어 설계 및 개발 과정 전반에 걸쳐 요구사항을 다루게 되므로 요구사항의 개념과 특징을 잘 알아두는 것이 좋습니다.

이해관계자(利害關係者)

소프트웨어 개발과 관련해서 이해관계자는 소프트웨어 개발 의뢰자, 소프트웨어 개발자, 소프트웨어 사용자 등이 있습니다.

전문가의 조언

요구사항은 크게 기능과 비기능으로 구분할 수 있습니다. 기능 요구사항은 '사용자는 회원ID와 비밀번호를 입력하여 로그인할 수 있다.'와 같이 말 그대로 기능에 관한 요구사항이고, 비기능 요구사항은 "시스템은 1년 365일, 하루 24시간 운용이 가능해야 한다."와 같이 대부분 품질이나 제약사항과 관련이 있습니다. 이를 염두에 두고 요구사항을 기능과 비기능으로 구분할 수 있도록 정리하세요.

- **가용성** : 사용하고자 할 때 언제라도 사용할 수 있는 정도
- **정합성** : 데이터의 값이 서로 모순 없이 일관되게 일치하는 정도
- **상호 호환성** : 다른 소프트웨어와 정보를 교환할 수 있는 정도
- **대응성** : 발생한 상황에 대처하는 정도
- **이식성** : 다양한 하드웨어 환경에서도 운용 가능하도록 쉽게 수정될 수 있는 정도
- **확장성** : 규모나 범위를 넓힐 수 있는 정도

비기능 요구사항 (Non-functional requirements)	• 제약사항 : 시스템 설계, 구축, 운영과 관련하여 사전에 파악된 기술, 표준, 업무, 법 · 제도 등의 제약조건 • 프로젝트 관리 요구사항 : 프로젝트의 원활한 수행을 위한 관리 방법에 대한 요구사항 • 프로젝트 지원 요구사항 : 프로젝트의 원활한 수행을 위한 지원 사항이나 방안에 대한 요구사항
사용자 요구사항 (User requirements)	• 사용자 관점에서 본 시스템이 제공해야 할 요구사항 • 사용자를 위한 것으로 친숙한 표현으로 이해하기 쉽게 작성된다.
시스템 요구사항 (System requirements)	• 개발자 관점에서 본 시스템 전체가 사용자와 다른 시스템에 제공해야 할 요구사항 • 사용자 요구사항에 비해 전문적이고 기술적인 용어로 표현된다. • 소프트웨어 요구사항이라고도 한다.

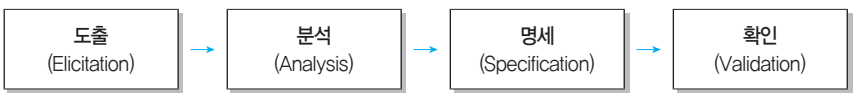
전문가의 조언

요구사항은 '도출 → 분석 → 명세 → 확인' 과정을 거치는데, 각 단계의 명칭을 보면 해당 단계에서 무엇을 수행하는지 대략적인 윤곽을 잡을 수 있습니다. 요구사항 개발 과정을 순서대로 기억하고 각 단계에서는 무엇을 수행하는지 파악해 두세요.

3 요구사항 개발 프로세스

요구사항 개발 프로세스는 개발 대상에 대한 요구사항을 체계적으로 도출하고 이를 분석한 후 분석 결과를 명세서(Specification Document)에 정리한 다음 마지막으로 이를 확인 및 검증하는 일련의 구조화된 활동이다.

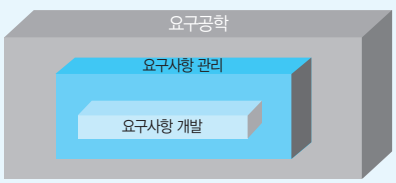
- 요구사항 개발 프로세스가 진행되기 전에 개발 프로세스가 비즈니스 목적에 부합되는지, 예산은 적정한지 등에 대한 정보를 수집, 평가한 보고서를 토대로 타당성 조사(Feasibility Study)가 선행되어야 한다.
- 요구사항 개발은 요구 공학(Requirement Engineering)의 한 요소이다.



요구공학(Requirements Engineering)

요구공학은 무엇을 개발해야 하는지 요구사항을 정의하고, 분석 및 관리하는 프로세스를 연구하는 학문입니다.

- 점점 복잡하고 대형화되어가는 소프트웨어 개발 환경에 따라 사용자 요구사항도 더욱 복잡해지고 잦은 변경이 발생하는 데, 이는 요구사항에 문제가 발생할 가능성을 높이며 요구사항 관리가 잘못될 수 있는 원인이 됩니다.
- 요구공학은 요구사항 변경의 원인과 처리 방법을 이해하고 요구사항 관리 프로세스의 품질을 개선하여 소프트웨어 프로젝트 실패를 최소화하는 것을 목표로 합니다.



4 요구사항 도출(Requirement Elicitation, 요구사항 수집)

요구사항 도출은 시스템, 사용자, 그리고 시스템 개발에 관련된 사람들이 서로 의견을 교환하여 요구사항이 어디에 있는지, 어떻게 수집할 것인지를 식별하고 이해하는 과정이다.

- 요구사항 도출은 소프트웨어가 해결해야 할 문제를 이해하는 첫 번째 단계이다.
- 요구사항 도출 단계에서 개발자와 고객 사이의 관계가 만들어지고 이해관계자(Stakeholder)가 식별된다.
- 이 단계에서는 다양한 이해관계자 간의 효율적인 의사소통이 중요하다.
- 요구사항 도출은 소프트웨어 개발 생명 주기(SDLC; Software Development Life Cycle) 동안 지속적으로 반복된다.
- 요구사항을 도출하는 주요 기법에는 인터뷰, 설문, 브레인스토밍*, 워크샵, 프로토타이핑*, 유스케이스* 등이 있다.

5 요구사항 분석(Requirement Analysis)

요구사항 분석은 개발 대상에 대한 사용자의 요구사항 중 명확하지 않거나 모호하여 이해되지 않는 부분을 발견하고 이를 걸러내기 위한 과정이다.

- 사용자 요구사항의 타당성을 조사하고 비용과 일정에 대한 제약을 설정한다.
- 내용이 중복되거나 하나로 통합되어야 하는 등 서로 상충되는 요구사항이 있으면 이를 해결한다.
- 도출된 요구사항들을 토대로 소프트웨어의 범위를 파악한다.
- 도출된 요구사항들을 토대로 소프트웨어와 주변 환경이 상호 작용하는 방법을 이해한다.

6 요구사항 명세(Requirement Specification)

요구사항 명세는 요구사항을 체계적으로 분석한 후 승인될 수 있도록 문서화하는 것을 의미한다.

- 요구사항을 문서화할 때는 기능 요구사항은 빠짐없이 완전하고 명확하게 기술해야 하며, 비기능 요구사항은 필요한 것만 명확하게 기술해야 한다.
- 요구사항은 사용자가 이해하기 쉬우며, 개발자가 효과적으로 설계할 수 있도록 작성되어야 한다.
- 설계 과정에서 잘못된 부분이 확인될 경우 그 내용을 요구사항 정의서에서 추적할 수 있어야 한다.

브레인스토밍(Brain Storming)
브레인스토밍은 3인 이상이 자유롭게 의견을 교환하면서 독창적인 아이디어를 산출해 내는 방법입니다.

프로토타이핑(Prototyping)
프로토타이핑은 프로토타입(견본품)을 통해 효과적으로 요구 분석을 수행하면서 명세서를 산출하는 작업으로, 가장 단순한 형태는 설명을 위해 종이에 대략적인 순서나 형태를 그려 보여주는 것입니다.

유스케이스(Use Case)
유스케이스는 사용자의 요구사항을 기능 단위로 표현하는 것입니다.

소프트웨어 요구사항 명세서

소프트웨어 요구사항 명세서(SRS; Software Requirement Specification)는 업계 표준 용어로 소프트웨어가 반드시 제공해야 하는 기능, 특징, 제약조건 등을 명시합니다.

- 시스템의 모든 동작뿐만 아니라 성능, 보안, 사용성과 같은 품질도 기술되어야 합니다.
- 프로젝트 유형에 맞게 양식을 만들어 사용합니다.
- 소프트웨어 요구사항 명세서에 포함되는 시스템 기능, 데이터, 외부 인터페이스, 품질 요구사항은 요구사항 단위별로 개별 요구사항 명세서를 작성합니다.
- 다음은 여러 유형의 프로젝트에 유용하게 사용할 수 있는 소프트웨어 요구사항 명세서 양식입니다.

1. 소개
   1.1 목적
   1.2 문서 규칙
   1.3 프로젝트 범위
   1.4 참조
2. 전반적인 설명
   2.1 제품 관점
   2.2 사용자 클래스 및 특징
   2.3 운영환경
   2.4 설계 및 구현 제약 조건
   2.5 가정 및 의존성
3. 시스템 기능
   3.1 시스템 기능
      3.1.1 설명
      3.1.2 기능적 요구사항
4. 데이터 요구사항
   4.1 논리 데이터 모델
   4.2 데이터 사전
   4.3 보고서
   4.4 데이터 수집, 무결성, 보존 및 폐기
5. 외부 인터페이스 요구사항
   5.1 사용자 인터페이스
   5.2 소프트웨어 인터페이스
   5.3 하드웨어 인터페이스
   5.4 통신 인터페이스
6. 품질 속성
   6.1 사용성
   6.2 성능
   6.3 보안
   6.4 안전
   6.X 기타
7. 국제화 및 현지화 요구
8. 기타 요구 사항

부록 A : 용어 사전
부록 B : 분석 모델

7 요구사항 확인(Requirement Validation, 요구사항 검증)

요구사항 확인은 개발 자원을 요구사항에 할당하기 전에 요구사항 명세서가 정확하고 완전하게 작성되었는지를 검토하는 활동이다.

- 분석가가 요구사항을 정확하게 이해한 후 요구사항 명세서를 작성했는지 확인(Validation)하는 것이 필요하다.
- 요구사항 명세서의 내용이 이해하기 쉬운지, 일관성은 있는지, 회사의 기준에는 맞는지, 그리고 누락된 기능은 없는지 등을 검증(Verification)하는 것이 중요하다.
- 요구사항 문서는 이해관계자들이 검토해야 한다.
- 일반적으로 요구사항 관리 도구를 이용하여 요구사항 정의 문서들에 대해 형상 관리*를 수행한다.

형상 관리(SCM; Software Configuration Management)
소프트웨어 개발 단계의 각 과정에서 만들어지는 프로그램, 프로그램을 설명하는 문서, 데이터 등을 통칭하여 형상이라고 합니다. 형상 관리는 소프트웨어의 개발 과정에서 만들어지는 형상들의 변경 사항을 관리하는 일련의 활동을 말합니다.

따라잡기

기출문제 따라잡기 Section 006

출제예상
1. 병원 진료관리시스템의 기능적 요구사항으로 옳지 않은 것은?

① 담당 의사는 자신이 담당한 환자의 진료 내용을 입력 또는 수정한다.
② 환자 정보 관리자는 환자의 정보를 등록, 삭제할 수 있다.
③ 환자는 자신이 진료한 내역을 조회할 수 있다.
④ 시스템 장애로 인한 정지시간이 한 달에 1시간을 넘지 않아야 한다.

성능에 관한 내용은 외적인 품질 속성, 즉 비기능 요구사항에 해당됩니다.

출제예상
2. 다음은 '호텔 예약 시스템'의 요구사항을 나열한 것이다. 기능적 요구사항에 대한 설명으로 가장 옳지 않은 것은?

① 예약 시 고객의 정보를 입력하는 방법을 결정해야 한다.
② 영수증과 예약 확인서에 어떤 정보를 표시할지 결정해야 한다.
③ 예약 대행 여행사와 고객이 호텔 정보 데이터베이스에 접근할 때 어떤 정보를 얻을 수 있는지를 결정해야 한다.
④ 해외 분점 호텔의 고객 정보까지 관리하기 위해 시스템을 확장할 수 있도록 설계해야 한다.

확장성에 관한 내용은 내적인 품질 속성, 즉 이것도 비기능 요구사항에 해당합니다.

출제예상
3. 요구분석 단계를 순서대로 바르게 나열한 것은?

가. 요구사항 검증
나. 요구사항 명세화
다. 타당성 조사
라. 요구사항 추출 및 분석

① 다 → 라 → 가 → 나
② 라 → 다 → 나 → 가
③ 라 → 가 → 다 → 나
④ 다 → 라 → 나 → 가

개발에 대한 타당성이 충족되면 작성된 요구사항을 분석하여 정리한 후 검증하는 과정을 진행합니다.

출제예상
4. 비기능 요구사항에 대한 설명으로 옳지 않은 것은?

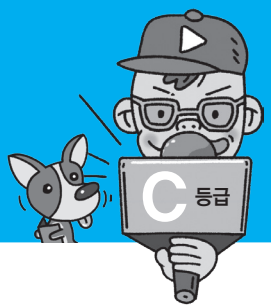
① 예산의 범위, 조직의 비전, 상호 호환성, 보안성, 안전성과 같은 사용자의 필요에 의해 발생한다.
② 시스템이 제공하는 서비스의 품질에 관한 것이다.
③ 시스템에서 제공되는 서비스나 기능에 대한 제약 사항에 관한 것이다.
④ 시스템이 특정 입력에 대해 어떻게 반응하는지, 사용자의 요구에 대해 시스템이 어떻게 동작해야 하는지에 관한 사항이다.

시스템이 무엇을 하는지, 어떻게 하는지 등에 관한 기능이나 동작에 관한 것이 기능 요구사항입니다. 이와 관련된 내용을 찾아보세요.

▶ 정답 : 1. ④ 2. ④ 3. ④ 4. ④

SECTION 007 ● LIVE

요구사항 분석 기법



전문가의 조언

요구사항은 '도출 → 분석 → 명세 → 확인' 과정을 거친다고 했죠? 순서상으로 보면 분석은 요구사항을 명세하기 전에 요구사항이 제대로 도출되었는지 분석하는 과정이란 것을 알 수 있습니다. 이번 섹션에서는 요구사항을 분석할 때 사용하는 기법들에 대해 학습합니다. 먼저 요구사항 분석이 무엇인지 개념을 정확히 정리하고 요구사항 분석에 사용되는 기법 중 무슨 기법을 말하는지 바로 알 수 있도록 각각의 특징을 파악해 두세요.

1 요구사항 분석 기법

요구사항 분석 기법은 개발 대상에 대한 사용자의 요구사항 중 명확하지 않거나 모호한 부분을 걸러내기 위한 방법이다.

- 요구사항 분석 기법에는 요구사항 분류, 개념 모델링, 요구사항 할당, 요구사항 협상, 정형 분석 등이 있다.

2 요구사항 분류(Requirement Classification)

요구사항을 명확히 확인할 수 있도록 다음과 같은 분류 기준을 고려한다.

- 기능 요구사항과 비기능 요구사항으로 분류한다.
- 하나 이상의 상위 요구사항에서 유도된 것인지 또는 이해관계자나 다른 원천(Source)으로부터 직접 발생한 것인지 분류한다.
- 개발할 제품에 관한 것인지 개발 과정(프로세스)에 관한 것인지 분류한다.
- 우선순위에 따라 분류한다.
- 소프트웨어에 미치는 영향의 범위에 따라 분류한다.
- 소프트웨어 생명 주기 동안에 변경될 가능성이 있는지 여부에 따라 분류한다.

3 개념 모델링(Conceptual Modeling)

요구사항을 보다 쉽게 이해할 수 있도록 현실 세계의 상황을 단순화하여 개념적으로 표현한 것을 모델이라고 하며, 이러한 모델을 만드는 과정을 모델링이라고 한다.

- 모델은 문제가 발생하는 상황을 쉽게 이해시키고 해결책을 설명할 수 있으므로 실세계 문제에 대한 모델링은 소프트웨어 요구사항 분석의 핵심이다.
- 개념 모델은 문제의 주체인 개체(Entity)*들과 그들 간의 관계 및 종속성*을 반영한다.
- 요구사항을 이해하는 이해관계자별로 관점이 다양하므로 그에 맞게 개념 모델도 다양하게 표현되어야 한다.
- 개념 모델의 종류에는 유스케이스 다이어그램(Use Case Diagram), 데이터 흐름 모델(Data Flow Model), 상태 모델(State Model), 목표기반 모델(Goal-Based Model), 사용자 인터액션(User Interactions), 객체 모델(Object Model), 데이터 모델(Data Model) 등이 있다.
- 모델링 표기는 주로 UML*(Unified Modeling Language)을 사용한다.

• **개체(Entity)** : 현실 세계에서는 사람, 자동차, 컴퓨터, 고양이 등과 같이 우리 주위에서 사용되는 물질적이거나 개념적인 것으로, 요구사항 분석에서는 요구기능이 적용되는 시스템이나 기능을 사용하는 사람 또는 그러한 기능과 연동되는 다른 시스템을 의미합니다.

• **종속성(從屬性, Dependency)** : A가 어떤 행동을 할 때 반드시 B를 통해서만 수행할 수 있다면 A는 B에 종속적이라고 표현하는데, 이와 같이 서로 의존적인 관계를 의미합니다.

UML
UML에 대한 내용은 Section 009에서 자세히 공부합니다.

4 요구사항 할당(Requirement Allocation)

요구사항 할당은 요구사항을 만족시키기 위한 구성 요소를 식별하는 것이다.

- 식별된 구성 요소들 간에 어떻게 작용하는지 분석하는 과정에서 추가적인 요구사항이 발견될 수 있다.

5 요구사항 협상(Requirement Negotiation)

요구사항 협상은 요구사항이 서로 충돌될 경우 이를 적절히 해결하는 과정이다.

- 요구사항이 다음과 같은 이유로 서로 충돌되는 경우 어느 한 쪽으로 맞추기보다는 적절한 기준점을 찾아 합의해야 한다.
  - 두 명의 이해관계자가 요구하는 요구사항이 서로 충돌되는 경우
  - 요구사항과 자원이 서로 충돌되는 경우
  - 기능 요구사항과 비기능 요구사항이 서로 충돌되는 경우
- 요구사항이 서로 충돌되는 경우에 각각에 우선순위를 부여하면, 무엇이 더 중요한지를 인식할 수 있으므로 문제 해결에 도움이 될 수 있다.

6 정형 분석(Formal Analysis)

정형 분석은 구문(Syntax)과 의미(Semantics)를 갖는 정형화된 언어를 이용해 요구사항을 수학적 기호로 표현*한 후 이를 분석하는 과정이다.

- 정형 분석(Formal Analysis)은 요구사항 분석의 마지막 단계에서 이루어진다.

정형 명세(Formal Specification)
정형화된 언어를 이용해 수학적 기호로 기술하는 것을 정형 명세라고 합니다.

기출문제 따라잡기 Section 007

출제예상
1. 다음 중 요구사항 분석 기법이 아닌 것은?

① 요구사항 분류 ② 개념 모델링
③ 정형 분석 ④ 요구사항 제거

요구사항 분석 기법의 종류를 기억하라고 했죠? 이 문제를 틀렸다면, 다시 한 번 정리하고 넘어가세요.

출제예상
2. 요구사항 분석 기법 중 다음이 설명하는 것은 무엇인가?

- 구문(Syntax)과 의미(Semantics)를 갖는 정형화된 언어를 이용해 요구사항을 수학적 기호로 표현한 후 이를 분석하는 과정이다.
- 요구사항 분석의 마지막 단계에서 이루어진다.

① 요구사항 분류 ② 요구사항 협상
③ 정형 분석 ④ 개념 모델링

정형화된 언어를 이용해서 요구사항을 분석하는 과정은 무엇일까요?

출제예상
3. 요구사항 분석 기법 중 개념 모델링에 대한 설명이 아닌 것은?

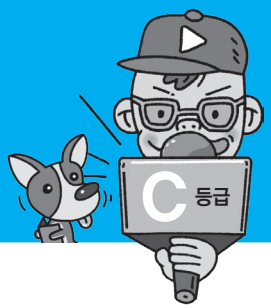
① 개념 모델은 소프트웨어 요구사항 분석의 핵심이다.
② 요구사항을 이해하는 이해관계자별로 다른 의견이 도출되지 않도록 하나의 통일된 개념 모델로 표현한다.
③ 개념 모델은 문제의 주체인 개체(Entity)들과 그들 간의 관계 및 종속성을 반영한다.
④ 모델링 표기는 주로 UML(Unified Modeling Language)을 사용한다.

개념 모델의 종류에는 유스케이스 다이어그램, 데이터 흐름 모델, 상태 모델 등이 있습니다. 이렇게 다양한 이유는 이해관계자별로 관점이 다양하기 때문입니다.

▶ 정답 : 1. ④ 2. ③ 3. ②

SECTION
008
LIVE

요구사항 확인 기법



전문가의 조언

요구사항이 제대로 도출되었는지 체계적으로 분석한 후 개발 작업 전에 승인을 위해 문서화하는 것이 명세인데, 이번 섹션에서는 문서화된 요구사항 명세서를 최종적으로 확인하고 검증할 때 사용되는 기법들에는 어떤 것들이 있는지, 각 기법의 특징에는 무엇들이 있는지 파악해 두세요.

1 요구사항 확인 기법

요구사항 확인 기법은 요구사항 개발 과정을 거쳐 문서화된 요구사항 관련 내용을 확인하고 검증하는 방법이다.

- 요구사항에 자원이 배정되기 전에 문제 파악을 위한 검증을 수행해야 한다.
- 요구사항 확인 기법에는 요구사항 검토(Requirement Reviews), 프로토타이핑(Prototyping), 모델 검증(Model Verification), 인수 테스트(Acceptance Tests) 등이 있다.

2 요구사항 검토(Requirement Reviews)

요구사항 검토는 문서화된 요구사항을 훑어보면서 확인하는 것으로 가장 일반적인 요구사항 검증 방법이다.

- 요구사항 검토자들은 요구사항 검토를 통해 명확하지 않은 내용은 없는지, 가정이 잘못되지는 않았는지, 정해놓은 기준을 벗어나지는 않는지 등을 찾아낸다.
- 요구사항 검토자 그룹을 구성할 때는 구성 방법이 중요하다. 예를 들어 고객 중심 프로젝터에 대한 검토자 그룹에는 고객 대표자가 꼭 포함되어야 하기 때문이다.
- 검토는 시스템 정의서(System Definition Document), 시스템 사양서(System Specification), 소프트웨어 요구사항 명세서(SRS; Software Requirements Specification Document) 등을 완성한 시점에 이루어진다.

3 프로토타이핑(Prototyping)

프로토타이핑은 초기 도출된 요구사항을 토대로 프로토타입(Prototype)을 만든 후 대상 시스템의 개발이 진행되는 동안 도출되는 요구사항을 반영하면서 지속적으로 프로토타입을 재작성하는 과정이다.

- 상품이나 서비스가 출시되기 전에 개발 대상 시스템 또는 그 일부분을 개략적으로 만든 원형을 프로토타입이라고 한다.
- 프로토타이핑을 수행하면서 새로운 요구사항이 도출될 수 있다.
- 소프트웨어 요구사항에 대한 소프트웨어 엔지니어의 해석이 맞는지 확인하기 위한 수단으로 주로 사용된다.

- 프로토타이핑은 다음과 같은 장 · 단점이 있다.

장점	단점
• 빠르게 제작할 수 있으며, 반복되는 제작을 통해 발전된 결과물을 얻을 수 있다. • 최종 시스템을 완성하기 전에 추가/변경 요구사항이나 아이디어 등에 대한 피드백이 가능하다. • 이해하기 쉬워 사용자와 개발자 또는 개발자 사이의 의사소통이 원활해진다. • 개발될 시스템의 사용에 대한 문제점을 시스템 완성 전에 식별할 수 있다. • 프로토타입이 개선될수록 변동 가능한 요구사항들이 감소한다.	• 사용자의 관심이 핵심에서 벗어나 프로토타입 제작에만 집중될 수 있다. • 개발 대상의 일부만을 대상으로 프로토타입이 제작된 경우 대상 범위를 잘못 이해하여 사용성이 과대평가 될 수 있다. • 지속적이고 반복적인 프로토타입의 개선으로 인한 비용이 부담될 수 있다.

4 모델 검증(Model Verification)

모델 검증이란 요구사항 분석 단계에서 개발된 모델이 요구사항을 충족시키는지 검증하는 것이다.

- 객체 모델의 경우 객체들 사이에 존재하는 의사소통 경로(Communication Path)를 검증(Verify)하기 위하여 정적 분석(Static Analysis)*을 수행하는 것이 유용하다.

정적 분석(Static Analysis)
정적 분석은 실행을 통해서 확인하는 것이 아니라 명세서의 정확성이나 일관성 등을 확인하거나 분석 도구를 사용해 확인하는 방법입니다. 직접 실행을 통해 확인하는 방법은 동적 분석(Dynamic Analysis)이라고 합니다.

5 인수 테스트(Acceptance Tests)

인수 테스트는 사용자가 실제로 사용될 환경에서 요구사항들이 모두 충족되는지 사용자 입장에서 확인하는 과정이다.

- 각각의 요구사항을 어떻게 확인할 것인지에 대한 계획을 세워야 한다.
- 인수 테스트의 종류*에는 사용자 인수 테스트, 운영상의 인수 테스트, 계약 인수 테스트, 규정 인수 테스트, 알파 검사, 베타 검사가 있다.

인수 테스트의 종류
인수 테스트의 종류에 대한 자세한 내용은 Section 052를 참조하세요.

기출문제 따라잡기

출제예상

1. 요구사항 확인에 대한 설명으로 잘못된 것은?

① 요구사항 확인 기법은 요구사항 개발 과정을 거쳐 문서화된 요구사항 관련 내용을 확인하고 검증하는 방법이다.

② 일반적으로 요구사항 관리 도구를 이용하여 요구사항 정의 문서들에 대해 형상 관리를 수행한다.

③ 요구사항 문서는 이해관계자들이 검토해야 한다.

④ 요구사항에 자원이 배정된 후에 문제 파악을 위한 검증을 수행한다.

요구사항에 대한 분석과 확인이 완료된 후에 자원을 배정합니다.

출제예상

2. 다음 중 요구사항 확인 기법이 아닌 것은?

① 요구사항 검토 ② 정형 분석

③ 모델 검증 ④ 인수 테스트

보기 중에 요구사항 분석 기법이 있네요. 이 문제를 틀렸으면 요구사항 확인 기법의 종류를 다시 한 번 정리하고 넘어가세요.

출제예상

3. 요구사항 확인 기법 중 하나인 프로토타이핑의 장점이 아닌 것은?

① 빠르게 제작할 수 있으며, 반복되는 제작을 통해 발전된 결과물을 얻을 수 있다.

② 이해하기 쉬워 사용자와 개발자 또는 개발자 사이의 의사소통이 원활해진다.

③ 지속적이고 반복적인 프로토타입의 개선으로 인해 비용이 증가한다.

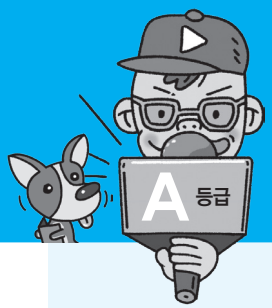
④ 프로토타입이 개선될수록 변동 가능한 요구사항들이 감소한다.

제시된 보기 중 부정적인 내용을 찾아보세요. 그것은 프로토타이핑의 단점에 해당됩니다.

▶ 정답 : 1. ④ 2. ② 3. ③

SECTION 009 ● LIVE

UML(Unified Modeling Language)



1 UML(Unified Modeling Language)의 개요

UML은 시스템 분석, 설계, 구현 등 시스템 개발 과정에서 시스템 개발자와 고객 또는 개발자 상호간의 의사소통이 원활하게 이루어지도록 표준화한 대표적인 객체지향 모델링 언어*이다.

- UML은 Rumbaugh(OMT), Booch, Jacobson 등의 객체지향 방법론의 장점을 통합하였으며, 객체 기술에 관한 국제표준화기구인 OMG(Object Management Group)에서 표준으로 지정하였다.
- UML을 이용하여 시스템의 구조를 표현하는 6개의 구조 다이어그램과 시스템의 동작을 표현하는 7개의 행위 다이어그램을 작성할 수 있다.
- 각각의 다이어그램은 사물과 사물 간의 관계를 용도에 맞게 표현한다.
- UML의 구성 요소에는 사물, 관계, 다이어그램 등이 있다.

2 사물(Things)

사물은 모델을 구성하는 가장 중요한 기본 요소로, 다이어그램 안에서 관계가 형성될 수 있는 대상들을 말한다.

- 사물에는 구조 사물, 행동 사물, 그룹 사물, 주해 사물이 있다.

사물	내용
구조 사물 (Structural Things)	• 시스템의 개념적, 물리적 요소를 표현 • 클래스(Class), 유스케이스(Use Case), 컴포넌트(Component)*, 노드(Node) 등
행동 사물 (Behavioral Things)	• 시간과 공간에 따른 요소들의 행위를 표현 • 상호작용(Interaction), 상태 머신(State Machine) 등
그룹 사물 (Grouping Things)	• 요소들을 그룹으로 묶어서 표현 • 패키지(Package)
주해 사물 (Annotation Things)	• 부가적인 설명이나 제약조건 등을 표현 • 노트(Note)

3 관계(Relationships)

관계는 사물과 사물 사이의 연관성을 표현하는 것으로, 연관 관계, 집합 관계, 포함 관계, 일반화 관계, 의존 관계, 실체화 관계 등이 있다.

연관(Association) 관계

연관 관계는 2개 이상의 사물이 서로 관련되어 있음을 표현한다.

전문가의 조언

많은 사람들이 모여 작업을 수행하다 보면 같은 대상물을 보고도 서로 다르게 표현하여 의사소통에 문제가 생기는 경우가 있습니다. 이런 문제를 해결하는 가장 좋은 방법은 공통된 표현법을 만들어 사용하는 것이죠. UML은 공통된 표현법을 사용해 개발할 대상물을 다이어그램으로 표현하는 도구입니다. 소프트웨어 개발 참여자들은 UML로 표현된 다이어그램으로 개발에 관한 의견을 서로 교환합니다. 이러한 UML의 개념과 특징을 잘 정리해 두세요.

모델링 언어

모델링 언어란 우리가 만들고자 하는 것을 시각적으로 표현할 수 있는 표기법, 도구 등을 의미합니다.

전문가의 조언

사람, 자동차, 컴퓨터, 동물 등과 같이 우리 주위에서 사용되는 물질적이거나 개념적인 것을 개체(Entity)라고 합니다. 이러한 개체를 컴퓨터 내부에 추상적으로 표현한 것을 사물(Things) 또는 객체(Object)라고 하는데, 다이어그램을 표현할 때는 사물보다는 객체라는 표현을 주로 사용합니다. 사물들은 이름을 통해 그 역할을 유추할 수 있으니 굳이 사물들의 역할을 외우려 노력하지 마세요. 여기서는 사물들의 종류만 기억하고 각 사물들의 구체적인 표현 형태는 가볍게 읽어보세요.

컴포넌트(Component)

컴포넌트는 문서, 소스코드, 파일, 라이브러리 등과 같은 모듈화된 자원으로, 재사용이 가능합니다.

전문가의 조언

관계는 UML을 학습하는 데 있어 반드시 알아야 할 중요한 개념이자 요소이므로 각 관계들의 개별적인 개념과 표현 방법을 확실히 숙지해야 합니다.

- 사물 사이를 실선으로 연결하여 표현하며, 방향성은 화살표로 표현한다.
- 서로에게 영향을 주는 양방향 관계의 경우 화살표를 생략하고 실선으로만 연결한다.
- 연관에 참여하는 객체의 개수를 의미하는 다중도(Multiplicity)를 선 위에 표기한다.

다중도	의미
1	1개의 객체가 연관되어 있다.
n	n개의 객체가 연관되어 있다.
0..1	연관된 객체가 없거나 1개만 존재한다.
0..* 또는 *	연관된 객체가 없거나 다수일 수 있다.
1..*	연관된 객체가 적어도 1개 이상이다.
n..*	연관된 객체가 적어도 n개 이상이다.
n..m	연관된 객체가 최소 n개에서 최대 m개이다.

전문가의 조언

사물, 즉 객체는 유스케이스, 클래스, 컴포넌트와 같이 별도의 표현 형태가 있는 경우를 제외하고는 기본적으로 사각형으로 표현됩니다.

예제 1 사람이 집을 소유하는 관계이다. 사람은 자기가 소유하고 있는 집에 대해 알고 있지만 집은 누구에 의해 자신이 소유되고 있는지 모른다는 의미이다.

해설

- '사람' 쪽에 표기된 다중도가 '1'이므로 집은 한 사람에 의해서만 소유될 수 있다.
- '집' 쪽에 표기된 다중도가 '1'이므로 사람은 집을 하나만 소유할 수 있다.

예제 2 선생님은 학생을 가르치고 학생은 선생님으로부터 가르침을 받는 것과 같이 선생님과 학생은 서로 관계가 있다.

해설

- '선생님' 쪽에 표기된 다중도가 '1..*'이므로 학생은 한 명 이상의 선생님으로부터 가르침을 받는다.
- '학생' 쪽에 표기된 다중도가 '1..*'이므로 선생님은 한 명 이상의 학생을 가르친다.

집합(Aggregation) 관계

집합 관계는 하나의 사물이 다른 사물에 포함되어 있는 관계를 표현한다.

- 포함하는 쪽(전체, Whole)과 포함되는 쪽(부분, Part)은 서로 독립적이다.
- 포함되는 쪽(부분, Part)에서 포함하는 쪽(전체, Whole)으로 속이 빈 마름모를 연결하여 표현한다.

예제 프린터는 컴퓨터에 연결해서 사용할 수 있으며, 다른 컴퓨터에 연결해서 사용할 수도 있다.

포함(Composition) 관계

포함 관계는 집합 관계의 특수한 형태로, 포함하는 사물의 변화가 포함되는 사물에게 영향을 미치는 관계를 표현한다.

- 포함하는 쪽(전체, Whole)과 포함되는 쪽(부분, Part)은 서로 독립될 수 없고 생명주기를 함께한다.
- 포함되는 쪽(부분, Part)에서 포함하는 쪽(전체, Whole)으로 속이 채워진 마름모를 연결하여 표현한다.

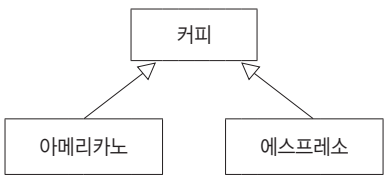
예제 문을 열 수 있는 키는 하나이며, 해당 키로 다른 문은 열 수 없다. 문이 없어지면 키도 더 이상 필요하지 않다.

일반화(Generalization) 관계

일반화 관계는 하나의 사물이 다른 사물에 비해 더 일반적인지 구체적인지를 표현한다.

- 예를 들어 사람은 여자와 남자보다 일반적인 개념이고 반대로 여자와 남자는 사람보다 구체적인 개념이다.
- 보다 일반적이 개념을 상위(부모), 보다 구체적인 개념을 하위(자식)라고 부른다.
- 구체적(하위)인 사물에서 일반적(상위)인 사물 쪽으로 속이 빈 화살표를 연결하여 표현한다.

예제 아메리카노와 에스프레소는 커피이다. 다시 말하면, 커피에는 아메리카노와 에스프레소가 있다.



의존(Dependency) 관계

의존 관계는 연관 관계와 같이 사물 사이에 서로 연관은 있으나 필요에 의해 서로에게 영향을 주는 짧은 시간 동안만 연관을 유지하는 관계를 표현한다.

- 하나의 사물과 다른 사물이 소유 관계는 아니지만 사물의 변화가 다른 사물에도 영향을 미치는 관계이다.
- 영향을 주는 사물(이용자)이 영향을 받는 사물(제공자) 쪽으로 점선 화살표를 연결하여 표현한다.

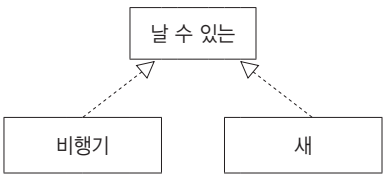
예제 등급이 높으면 할인율을 적용하고, 등급이 낮으면 할인율을 적용하지 않는다.

실체화(Realization) 관계

실체화 관계는 사물이 할 수 있거나 해야 하는 기능(행위, 인터페이스)으로 서로를 그룹화 할 수 있는 관계를 표현한다.

- 사물에서 기능 쪽으로 속이 빈 점선 화살표를 연결하여 표현한다.

예제 비행기는 날 수 있고 새도 날 수 있다. 그러므로 비행기와 새는 날 수 있다는 행위로 그룹화 할 수 있다.



전문가의 조언

다이어그램이 무엇인지, 구조적 다이어그램에는 어떤 것들이 있는지, 행위 다이어그램에는 어떤 것들이 있는지 정도만 알아두세요.

4 다이어그램(Diagram)

다이어그램은 사물과 관계를 도형으로 표현한 것이다.

- 여러 관점에서 시스템을 가시화한 뷰(View)를 제공함으로써 의사소통에 도움을 준다.
- 정적 모델링에서는 주로 구조적 다이어그램을 사용하고 동적 모델링에서는 주로 행위 다이어그램을 사용한다.
- 구조적(Structural) 다이어그램의 종류

클래스 다이어그램 (Class Diagram)	• 클래스와 클래스가 가지는 속성, 클래스 사이의 관계를 표현한다. • 시스템의 구조를 파악하고 구조상의 문제점을 도출할 수 있다.
객체 다이어그램 (Object Diagram)	클래스에 속한 사물(객체)들, 즉 인스턴스(Instance)를 특정 시점의 객체와 객체 사이의 관계로 표현한다.
컴포넌트 다이어그램 (Component Diagram)	• 실제 구현 모듈인 컴포넌트 간의 관계나 컴포넌트 간의 인터페이스를 표현한다. • 구현 단계에서 사용되는 다이어그램이다.
배치 다이어그램 (Deployment Diagram)	• 결과물, 프로세스, 컴포넌트 등 물리적 요소들의 위치를 표현한다. • 노드와 의사소통(통신) 경로로 표현한다. • 구현 단계에서 사용되는 다이어그램이다.
복합체 구조 다이어그램 (Composite Structure Diagram)	클래스나 컴포넌트가 복합 구조를 갖는 경우 그 내부 구조를 표현한다.
패키지 다이어그램 (Package Diagram)	유스케이스나 클래스 등의 모델 요소들을 그룹화한 패키지들의 관계를 표현한다.

- 행위(Behavioral) 다이어그램의 종류

유스케이스 다이어그램 (Use Case Diagram)	• 사용자의 요구를 분석하는 것으로 기능 모델링 작업에 사용한다. • 사용자(Actor)와 사용 사례(Use Case)로 구성되며, 사용 사례 간에는 여러 형태의 관계로 이루어진다.
시퀀스 다이어그램 (Sequence Diagram)	상호 작용하는 시스템이나 객체들이 주고받는 메시지를 표현한다.

커뮤니케이션 다이어그램 (Communication Diagram)	시퀀스 다이어그램과 같이 동작에 참여하는 객체들이 주고받는 메시지를 표현하는데, 메시지뿐만 아니라 객체들 간의 연관까지 표현한다.
상태 다이어그램 (State Diagram)	하나의 객체가 자신이 속한 클래스의 상태 변화 혹은 다른 객체와의 상호 작용에 따라 상태가 어떻게 변화하는지를 표현한다.
활동 다이어그램 (Activity Diagram)	시스템이 어떤 기능을 수행하는지 객체의 처리 로직이나 조건에 따른 처리의 흐름을 순서에 따라 표현한다.
상호작용 개요 다이어그램 (Interaction Overview Diagram)	상호작용 다이어그램 간의 제어 흐름을 표현한다.
타이밍 다이어그램 (Timing Diagram)	객체 상태 변화와 시간 제약을 명시적으로 표현한다.

기출문제 따라잡기 Section 009

출제예상

1. 결과물, 프로세스, 컴포넌트 등 물리적인 자원의 위치를 표시하는 것으로 구현 단계에서 사용되는 UML 다이어그램은?

① 컴포넌트(Component) 다이어그램
② 통신(Communication) 다이어그램
③ 배치(Depolyment) 다이어그램
④ 상태(State) 다이어그램

물리적인 자원들이 어떻게 위치하고 있는지 그 배치 현황을 표현하는 것은 무엇일까요?

출제예상

2. UML(Unified Modeling Language)의 구성 요소 중 사물(Thing)에 속하지 않는 것은?

① 구조(Structural) 사물 ② 분석(Analysis) 사물
③ 행동(Behavioral) 사물 ④ 주해(Annotation) 사물

UML의 사물(Thing)에는 보기에 제시된 3가지와 그룹(Grouping) 사물이 있습니다.

출제예상

3. 다음 중 동적인 행위를 표현하기 위한 UML 다이어그램이 아닌 것은?

① 시퀀스(Sequence) 다이어그램
② 상태(State) 다이어그램
③ 활동(Activity) 다이어그램
④ 배치(Deployment) 다이어그램

보기 중 물리적인 요소들의 위치와 같이 정적인 요소를 표현하는 다이어그램을 찾아보세요.

출제예상

4. 다음 중 UML 다이어그램이 아닌 것은?

① 사용사례 다이어그램(Use-Case Diagram)
② 순차 다이어그램(Sequence Diagram)
③ 클래스 다이어그램(Class Diagram)
④ 변화 다이어그램(Change Diagram)

UML 다이어그램에는 Class, Object, Component, Deployment, Use-Case, Sequence, Communication, State, Activity 다이어그램 등이 있습니다.

출제예상

5. UML 다이어그램 중 구현 단계에서 사용하기에 가장 적합한 것은?

① 유즈케이스 다이어그램
② 컴포넌트 다이어그램
③ 활동 다이어그램
④ 클래스 다이어그램

구현 단계에서 사용하는 다이어그램이 2개가 있었죠? 그 중 하나가 배치(Deployment) 다이어그램이었고요.

▶ 정답 : 1. ③ 2. ② 3. ④ 4. ④ 5. ②

예·상·문·제·은·행

1. 소프트웨어 생명 주기(Life Cycle) 모델 중 아래 보기가 설명하는 모형은?

> a. 고객과의 의사소통(Communication)을 통해 계획 수립과 위험 분석, 구축, 고객 평가의 과정을 거쳐 소프트웨어를 개발한다.
> b. 가장 큰 장점인 위험 분석 단계에서 기술과 관리의 위험 요소들을 하나씩 제거해 나감으로써 완성도 높은 소프트웨어를 만들 수 있다.
> c. 반복적인 작업을 수행하는 점증적 생명 주기 모델이다.
> d. 비용이 많이 들거나 시간이 많이 소요되는 대규모 프로젝트나 큰 시스템을 구축할 때 유리하다.

① 프로토타입(Prototype) 모델
② 폭포수(Waterfall) 모델
③ 나선형(Spiral) 모델
④ RAD 모델

2. 소프트웨어 공학 패러다임에 해당하지 않는 것은?

① 폭포수 모형
② 프로토타입 모형
③ 나선형 모형
④ 3세대 기법

3. 소프트웨어 공학에서 가장 폭넓게 사용되고 있는 모형은?

① 폭포수 모형(Waterfall Model)
② 프로토타입 모형(Prototype Model)
③ 나선형 모형(Spiral Model, 점증적 모형)
④ 4세대 기법(4GT)

4. 다음 중 소프트웨어 개발 모형이 가장 적절하게 선택된 경우는?

① 구축하고자 하는 시스템의 요구사항이 불분명하여 프로토타입 모형을 선택하였다.
② 개발 중에도 고객의 요구사항에 맞게 수정 작업을 할 수 있도록 폭포수 모형을 선택하였다.
③ 위험 분석을 통해 점증적으로 시스템을 개발할 수 있도록 폭포수 모형을 선택하였다.
④ 응용 분야가 단순하고 설치 시점에 제품 설명서가 요구됨에 따라 나선형 모형을 선택하였다.

5. 프로토타입 모델 개발 방법이 가장 적절하게 적용될 수 있는 경우는?

① 테스트 작업이 중요하지 않을 경우
② 고객이 빠른 시간 내에 개발의 완료를 요구할 경우
③ 구축하고자 하는 시스템의 요구사항이 불명확한 경우
④ 고객이 개발 과정에는 참여하지 않고자 하는 경우

6. 고전적 생명 주기 모델에 프로토타입 모형의 장점과 위험 분석 기능을 추가한 패러다임은?

① Waterfall Model
② Spiral Model
③ Jackson Model
④ 4GT

7. 소프트웨어 생명 주기 모형 중 Spiral Model에 대한 설명으로 옳지 않은 것은?

① 대규모 시스템에 적합하다.
② 초기에 위험 요소를 발견하지 못할 경우 위험 요소를 제거하기 위해 많은 비용이 소요될 수 있다.
③ 소프트웨어를 개발하면서 발생할 수 있는 위험을 관리하고 최소화하는 것을 목적으로 한다.
④ 소프트웨어 개발 과정의 앞 단계가 끝나야만 다음 단계로 넘어갈 수 있는 선형 순차적 모형이다.

8. 소프트웨어 생명 주기(Life Cycle) 모형 중 프로토타입(Prototype) 모형의 가장 큰 장점이라고 볼 수 있는 것은?

① 개발 비용의 절감
② 4세대 언어의 적용
③ 개발 단계의 명확성
④ 요구사항의 정확한 파악

9. 소프트웨어 개발 모형 중 나선형 모델의 활동 과정이 아닌 것은?

① 계획 및 정의
② 위험 분석
③ 개발
④ 유지보수

10. 여러 번의 개발 과정을 거쳐 완벽한 최종 소프트웨어를 개발하는 점진적 모형으로 보헴이 제안한 소프트웨어 생명주기 모델은?

① 4GT Model
② Spiral Model
③ Waterfall Model
④ Prototype Model

해설은 59쪽에 있습니다.

11. 소프트웨어 공학의 전통적인 개발 방법인 선형 순차 모형의 순서를 옳게 나열한 것은?

① 구현 → 분석 → 설계 → 테스트 → 유지보수
② 유지보수 → 테스트 → 분석 → 설계 → 구현
③ 분석 → 설계 → 구현 → 테스트 → 유지보수
④ 테스트 → 설계 → 유지보수 → 구현 → 분석

12. 폭포수 모델에 대한 설명으로 옳지 않은 것은?

① 소프트웨어 개발 과정의 각 단계가 순차적으로 진행된다.
② 앞 단계에서 발견하지 못한 오류를 다음 단계에서 발견했을 때 오류 수정이 용이하다.
③ 두 개 이상의 과정이 병행 수행되거나 이전 단계로 넘어가는 경우가 없다.
④ 개발 과정중에 발생하는 새로운 요구나 경험을 설계에 반영하기 힘들다.

13. 다음의 특징에 맞는 개발 접근방식은?

> • 유용한 소프트웨어를 빠르고, 지속적으로 제공하여 고객을 만족시킨다.
> • 개발 막바지에도 요구사항 변경을 환영한다.
> • 같은 사무실에서 얼굴을 맞대고 의견을 나눈다.
> • 동기가 부여된 개발자로 팀을 구성하고, 신뢰하고, 개발환경을 제공하고 지원한다.

① 컴포넌트 기반 개발(Component Based Development)
② 정보 공학 개발(Information Engineering Development)
③ Agile Programming
④ 객체 지향 개발(Object Oriented Development)

14. 폭포수 모형과 애자일을 비교 했을 때 애자일의 특징이 아닌 것은?

① 새로운 요구사항의 반영이 쉽다.
② 개발에 있어 계획보다는 고객을 중심으로 한다.
③ 개발이 완료되면 최종적으로 모든 기능을 테스트한다.
④ 지속적으로 고객과 의사소통을 수행한다.

15. 다음 중 스크럼에 대한 설명으로 잘못된 것은?

① 제품 개발에 필요한 모든 요구사항(User Story)을 우선순위에 따라 나열한 제품 백로그를 사용한다.
② 소멸(Burn-down) 차트를 통해 작업의 진행 상황을 확인할 수 있다.
③ 스프린트 검토 회의에서 개선할 사항에 대한 피드백이 정리되면 스크럼 마스터는 이를 다음 스프린트에 반영할 수 있도록 제품 백로그를 업데이트한다.
④ 스프린트 동안 진행될 작업들을 개발자별로 할당할 때는 개발자들이 자신에게 맞는 작업을 스스로 선별하여 담당할 수 있도록 하는 것이 좋다.

16. 다음에 제시된 XP(eXtreme Programming)의 개발 프로세스를 순서에 맞게 나열한 것은?

> ㄱ. 주기(Iteration)
> ㄴ. 승인 검사(Acceptance Test)
> ㄷ. 릴리즈 계획 수립
> ㄹ. 소규모 릴리즈

① ㄱ – ㄴ – ㄷ – ㄹ
② ㄴ – ㄷ – ㄹ – ㄱ
③ ㄷ – ㄴ – ㄱ – ㄹ
④ ㄷ – ㄱ – ㄴ – ㄹ

17. 다음 중 오픈 소스 사용에 따른 고려사항에 속하지 않는 것은?

① 라이선스의 종류
② 사용자 수
③ 기술의 지속 가능성
④ 라이선스의 비용

18. 사용자 요구사항 추출(Elicitation) 방법 중 시스템 수행 결과를 설명하기 위해 종이에 화면 순서를 기술하여 고객과 사용자에게 보여주는 것과 관련된 것은?

① 인터뷰
② 설문
③ 브레인스토밍
④ 프로토타이핑

19. 요구 분석에 대한 설명으로 옳지 않은 것은?

① 고객이나 개발자 누구나 알 수 있는 내용은 요구사항에서 생략하여도 무방하다.
② 요구사항은 소프트웨어를 개발하고 검증하는 기반이 된다.
③ 요구사항은 명확하고 구체적이며 검증이 가능해야 한다.
④ 요구사항은 크게 기능적인 요구사항과 비기능적 요구사항으로 분류된다.

▶ **정답 :** 1. ③ 2. ④ 3. ① 4. ① 5. ③ 6. ② 7. ④ 8. ④ 9. ④ 10. ② 11. ③ 12. ② 13. ③ 14. ③ 15. ③ 16. ④ 17. ④ 18. ④ 19. ①

예·상·문·제·은·행

해설은 60쪽에 있습니다.

20. 다음은 서점 시스템의 요구사항에 대한 내용이다. 비기능 요구사항에 대한 설명은?

① 사용자는 로그인 또는 비로그인을 통해 책을 구매할 수 있어야 한다.

② 사용자가 책을 현금으로 구매하였을 경우 현금영수증 처리를 할 수 있어야 한다.

③ 동시에 100명 이상이 주문을 요청해도 처리할 수 있어야 한다.

④ 사용자가 마이페이지에 저장해 놓은 도서 목록은 일정 기간 동안 그대로 저장되어 있어야 한다.

21. 요구사항 협상은 요구사항이 서로 충돌하는 경우 이를 적절히 해결하는 과정이다. 다음 중 요구사항이 서로 충돌되어 적절한 기준점을 찾아 합의해야 하는 경우가 아닌 것은?

① 두 명의 이해관계자가 요구하는 요구사항이 서로 충돌되는 경우

② 요구사항이 서로 우선순위가 다른 경우

③ 요구사항과 자원이 서로 충돌되는 경우

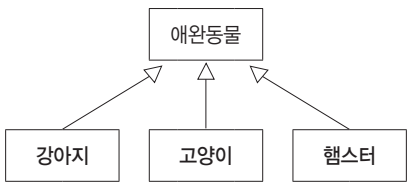
④ 기능 요구사항과 비기능 요구사항이 서로 충돌되는 경우

22. 객체지향 개발방법론 UML의 관계 표시법 중 의존(Dependency)을 표현하는 기호는?

① ———— ② - - - - - - - - - ->

③ - - - - - - - - ▷ ④ ————▷

23. UML의 다이어그램에서 관계를 완성하고자 한다. 다음 관계의 표현으로 가장 적합한 것은?



① 연관 관계(association)

② 일반화 관계(generalization)

③ 집단화 관계(aggregation)

④ 의존 관계(dependency)

▶ **정답 :** 20. ③ 21. ② 22. ② 23. ②

예·상·문·제·은·행·해·설

1. Section 001

- **프로토타입(Prototype) 모델** : 사용자의 요구사항을 정확히 파악하기 위해 실제 개발될 소프트웨어에 대한 견본품을 만들어 최종 결과물을 예측하는 모형
- **폭포수(Waterfall) 모델** : 폭포에서 한번 떨어진 물은 거슬러 올라 갈 수 없듯이 소프트웨어 개발도 각 단계를 확실히 매듭짓고 그 결과를 철저하게 검토하여 승인 과정을 거친 후에 다음 단계를 진행하며 이전 단계로 넘어갈 수 없는 방식
- **RAD 모델** : 소프트웨어의 구성 요소를 사용하여 매우 빠르게 선형 순차적 모델을 적용시킴으로써 빠른 개발 주기를 가지는 점진적 소프트웨어 개발 방식

2. Section 001

소프트웨어 공학 패러다임 : 폭포수 모형(Waterfall Model), 프로토타입 모형(Prototype Model), 나선형 모형(점증적 모형, Spiral Model), 4세대 기법(4GT)

3. Section 001

- **폭포수 모형(Waterfall Model)** : 폭포에서 한번 떨어진 물은 거슬러 올라갈 수 없듯이, 소프트웨어 개발도 각 단계를 확실히 매듭짓고 그 결과를 철저하게 검토하여 승인 과정을 거친 후에 다음 단계를 진행하며 이전 단계로 넘어갈 수 없는 방식으로 가장 오래, 가장 폭넓게 사용되고 있음
- **프로토타입 모형(Prototype Model)** : 사용자의 요구사항을 정확히 파악하기 위해 실제 개발될 소프트웨어에 대한 견본품(Prototype)을 만들어 최종 결과물을 예측하는 모형
- **나선형 모형(Spiral Model, 점증적 모형)** : 폭포수 모형과 프로토 타입 모형의 장점에 위험 분석 기능을 추가한 모형
- **4세대 기법(4GT)** : 사용자와 개발자가 쉽게 접근하고 사용할 수 있는 4세대 언어(4th Generation Language)를 이용하여 개발자가 조사한 요구사항 명세서로부터 원시 코드를 자동 생성할 수 있게 해주는 모형

4. Section 001

② 개발 중에도 고객의 요구사항에 맞게 수정 작업을 하려면 폭포수 모형이 아니라 개발 중간에 수정이 가능한 프로토타입 모형을 선택해야 한다.

③ 위험 분석을 통해 점증적으로 시스템을 개발하려면 폭포수 모형이 아니라 나선형 모형을 선택해야 한다.

④ 응용 분야가 단순하고 설치 시점에 제품 설명서가 요구되었을 경우에는 나선형 모형이 아니라 폭포수 모형을 선택해야 한다.

5. Section 001

프로토타입 모형(Prototype Model, 원형 모형)은 사용자의 요구사항을 정확히 파악하기 위해 실제 개발될 소프트웨어에 대한 견본(시제)품(Prototype)을 만들어 최종 결과물을 예측하는 모형이다.

6. Section 001

- **폭포수 모형(Waterfall Model)** : 소프트웨어 개발의 각 단계를 확실히 매듭짓고 그 결과를 철저하게 검토하여 승인 과정을 거친 후에 다음 단계를 진행하는 방식으로 가장 오래, 가장 폭넓게 사용되고 있다.
- **4세대 기법(4GT)** : 사용자와 개발자가 쉽게 접근하고 사용할 수 있는 4세대(4th Generation Language) 언어를 사용하는 모형

7. Section 001

나선형 모형(Spiral Model)은 폭포수 모형과 프로토타입 모형의 장점에 위험 분석 기능을 추가한 모형으로, 나선을 따라 돌듯이 여러 번의 소프트웨어 개발 과정을 거쳐 점진적으로(프로토타입을 지속적으로 발전시켜) 완벽한 최종 소프트웨어를 개발하는 것이다. 소프트웨어 개발 과정의 앞 단계가 끝나야만 다음 단계로 넘어갈 수 있는 선형 순차적 모형은 폭포수 모형이다.

9. Section 001

나선형 모형(Spiral Model, 점진적 모형)의 활동 과정은 계획 및 정의(Planning) → 위험 분석(Risk Analysis) → 공학적 개발(Engineering) → 고객 평가(Customer Evaluation) 순이다.

11. Section 001

폭포수 모형의 개발 순서는 '타당성 검토 → 계획 → 요구사항 및 분석 → 설계 → 구현 → 시험 → 유지보수'이다.

12. Section 001

폭포수 모형은 앞 단계가 끝나야만 다음 단계를 수행하는 것으로, 폭포가 물을 거슬러 올라갈 수 없는 것과 같이 현재 단계에서 이전 단계의 작업을 다시 수행할 수는 없다.

13. Section 001

- 지문에 제시된 내용은 애자일(Agile) 기법의 특징이다.
- **객체 지향 개발(Object Oriented Development)** : 현실 세계의 개체(Entity)를 기계의 부품처럼 하나의 객체(Object)로

예·상·문·제·은·행·해·설

만들어, 기계적인 부품들을 조립하여 제품을 만들 듯이 소프트웨어를 개발할 때에도 객체들을 조립해서 작성할 수 있도록 하는 개발 기법

14. Section 001

애자일은 짧은 개발 주기를 반복하며, 반복되는 주기마다 만들어지는 결과물에 대한 고객의 평가와 요구를 적극 수용한다. 개발이 완료되면 최종적으로 모든 기능을 테스트하는 것은 폭포수 모형의 특징이다.

15. Section 002

스프린트 검토 회의에서 개선할 사항에 대한 피드백이 정리되면 제품 책임자(PO; Product Owner)는 이를 다음 스프린트에 반영할 수 있도록 제품 백로그를 업데이트한다.

17. Section 005

오픈 소스란 누구나 제한 없이 무료로 사용할 수 있도록 소스 코드를 공개한 것이다.

18. Section 006

- 인터뷰 : 상대를 직접 만나서 이야기하면서 의견을 나누는 방법
- 설문 : 어떤 주제에 대하여 미리 준비한 질문을 내어 물어보는 방법
- 브레인스토밍 : 3인 이상이 자유롭게 의견을 교환하면서 독창적인 아이디어를 산출해 내는 방법

19. Section 006

요구사항은 제품 개발 과정에서 검증 테스트를 위한 기반이 되므로 개발에 필요한 모든 요소가 빠짐없이 완전하게 기술되어야 한다.

20. Section 006

③번은 시스템의 품질에 관한 요구사항으로 비기능 요구사항에 해당한다.

21. Section 007

이미 우선순위가 부여된 요구사항은 충돌이 발생하지 않는다.

22. Section 009

①번은 연관 관계, ③번은 실체화 관계, ④번은 일반화 관계를 표현하는 기호이다.

23. Section 009

애완동물을 구체적으로 표현하면 강아지, 고양이, 햄스터가 되고, 반대로 강아지, 고양이, 햄스터를 일반적으로 표현하면 애완동물이 된다.

2 장 화면 설계

이 장에서 꼭 알아야 할 키워드 Best 10

1. 사용자 인터페이스(UI) **2.** 내비게이션 **3.** 와이어프레임 **4.** 스토리보드 **5.** 프로토타입 **6.** UI 요구사항
7. 품질 요구사항 **8.** UI 설계서 **9.** 유용성 평가 **10.** UI 시나리오

SECTION 010 ● LIVE

사용자 인터페이스

전문가의 조언

스마트 폰을 사용할 때는 화면을 손가락으로 터치하고 TV 채널을 변경할 때는 리모콘을 누르죠? 터치화면과 리모콘이 바로 인터페이스입니다. 이처럼 인터페이스는 사람이 기계나 프로그램을 편리하게 사용할 수 있도록 하는 연결점이라고 생각하면 됩니다. 사용자 인터페이스의 개념을 중심으로 특징과 종류 등을 정리해 두세요.

인터페이스(Interface)

인터페이스는 서로 다른 두 시스템이나 소프트웨어 등을 서로 이어주는 부분 또는 접속 장치를 의미합니다.

1 사용자 인터페이스*(UI, User Interface)의 개요

사용자 인터페이스(UI)는 사용자와 시스템 간의 상호작용이 원활하게 이뤄지도록 도와주는 장치나 소프트웨어를 의미한다.

- 초기의 사용자 인터페이스는 단순히 사용자와 컴퓨터 간의 상호작용에만 국한되었지만 점차 사용자가 수행할 작업을 구체화시키는 기능 위주로 변경되었고, 최근에는 정보 내용을 전달하기 위한 표현 방법으로 변경되었다.
- **사용자 인터페이스의 세 가지 분야**
  - 정보 제공과 전달을 위한 물리적 제어에 관한 분야
  - 콘텐츠의 상세적인 표현과 전체적인 구성에 관한 분야
  - 모든 사용자가 편리하고 간편하게 사용하도록 하는 기능에 관한 분야

2 사용자 인터페이스(UI)의 특징

- 사용자의 만족도에 가장 큰 영향을 미치는 중요한 요소로, 소프트웨어 영역 중 변경이 가장 많이 발생한다.
- 사용자의 편리성과 가독성을 높임으로써 작업 시간을 단축시키고 업무에 대한 이해도를 높여준다.
- 최소한의 노력으로 원하는 결과를 얻을 수 있게 한다.
- 수행 결과의 오류를 줄인다.
- 사용자의 막연한 작업 기능에 대해 구체적인 방법을 제시해 준다.
- 정보 제공자와 공급자 간의 매개 역할을 수행한다.
- 사용자 인터페이스를 설계하기 위해서는 소프트웨어 아키텍처를 반드시 숙지해야 한다.

소프트웨어 아키텍처

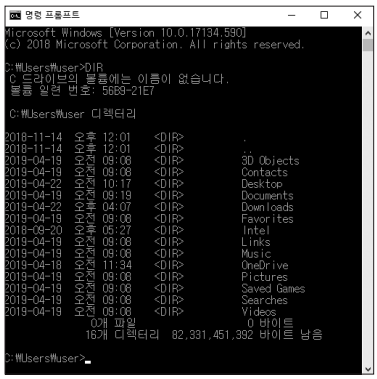
건물을 짓기 위해 설계도를 작성하듯 소프트웨어 아키텍처는 개발할 소프트웨어의 기본 틀을 만드는 것으로, 복잡한 소프트웨어 개발 과정을 체계적으로 접근하기 위한 밑그림입니다.

- 소프트웨어 아키텍처는 개발하고자 하는 소프트웨어의 특성과 본질을 파악하고 다양한 시각에서 모형화 합니다.
- 소프트웨어 아키텍처는 전체 시스템의 전반적인 구조를 설계합니다.
- 소프트웨어 아키텍처는 소프트웨어 시스템의 구축 및 개선을 용이하도록 합니다.
- 소프트웨어 아키텍처는 작업자들 간의 상호 이해, 타협 및 의사소통을 원활하게 하기 위해 사용됩니다.

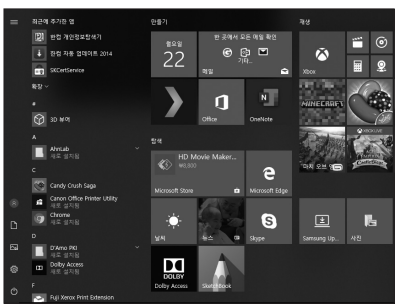
3 사용자 인터페이스의 구분*

사용자 인터페이스는 상호작용의 수단 및 방식에 따라 다음과 같이 구분된다.

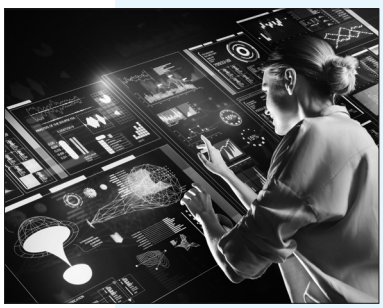
- **CLI(Command Line Interface)** : 명령과 출력이 텍스트 형태로 이뤄지는 인터페이스
- **GUI(Graphical User Interface)** : 아이콘이나 메뉴를 마우스로 선택하여 작업을 수행하는 그래픽 환경의 인터페이스
- **NUI(Natural User Interface)** : 사용자의 말이나 행동으로 기기를 조작하는 인터페이스



CLI



GUI



NUI

4 사용자 인터페이스의 기본 원칙

사용자 인터페이스의 기본 원칙에는 직관성, 유효성, 학습성, 유연성이 있다.

- **직관성** : 누구나 쉽게 이해하고 사용할 수 있어야 한다.
- **유효성** : 사용자의 목적을 정확하고 완벽하게 달성해야 한다.
- **학습성** : 누구나 쉽게 배우고 익힐 수 있어야 한다.
- **유연성** : 사용자의 요구사항을 최대한 수용하고 실수를 최소화해야 한다.

5 사용자 인터페이스의 설계 지침

사용자 인터페이스를 설계할 때 고려할 사항은 사용자 중심, 일관성, 단순성, 결과 예측 가능, 가시성, 표준화, 접근성, 명확성, 오류 발생 해결 등이다.

- **사용자 중심** : 사용자가 쉽게 이해하고 편리하게 사용할 수 있는 환경을 제공하며, 실사용자에 대한 이해가 바탕이 되어야 한다.
- **일관성** : 버튼이나 조작 방법 등을 일관성 있게 제공하므로 사용자가 쉽게 기억하고 습득할 수 있게 설계해야 한다.
- **단순성** : 조작 방법을 단순화시켜 인지적 부담을 감소시켜야 한다.
- **결과 예측 가능** : 작동시킬 기능만 보고도 결과를 미리 예측할 수 있게 설계해야 한다.

다른 사용자 인터페이스

- VUI(Voice User Interface) : 사람의 음성으로 기기를 조작하는 인터페이스
- OUI(Organic User Interface) : 모든 사물과 사용자 간의 상호작용을 위한 인터페이스로, 소프트웨어가 아닌 하드웨어 분야에서 사물 인터넷(Internet of Things), 가상현실(Virtual Reality), 증강현실(Augmented Reality), 혼합현실(Mixed Reality) 등과 함께 대두되고 있음

- **가시성** : 메인 화면에 주요 기능을 노출시켜 최대한 조작이 쉽도록 설계해야 한다.
- **표준화** : 기능 구조와 디자인을 표준화하여 한 번 학습한 이후에는 쉽게 사용할 수 있도록 설계해야 한다.
- **접근성** : 사용자의 연령, 성별, 인종 등 다양한 계층이 사용할 있도록 설계해야 한다.
- **명확성** : 사용자가 개념적으로 쉽게 인지할 수 있도록 설계해야 한다.
- **오류 발생 해결** : 오류가 발생하면 사용자가 쉽게 인지할 수 있도록 설계해야 한다.

기출문제 따라잡기 Section 010

출제예상

1. 사용자 인터페이스(UI)를 설계할 때 일반적으로 고려할 사항으로 틀린 것은?

① 일관성 : 전체 구조나 구성 요소는 일관성 있게 설계해야 한다.
② 개인성 : 사용자의 경험이나 능력 등 개인의 특성에 맞게 설계해야 한다.
③ 가시성 : 메인 화면에 주요 기능을 시각적으로 노출시켜 정보 흡수와 작업 수행이 쉽도록 설계해야 한다.
④ 사용 편리성 : 정보접근이 용이하고 기억하기 쉬워야 한다.

사용자 인터페이스는 개인이 아닌 다양한 계층에서 사용할 수 있도록 설계해야 합니다.

출제예상

2. 사용자 인터페이스를 설계할 때 주의해야 할 사항으로 가장 적합하지 않은 것은?

① 사용자가 콘텐츠를 쉽게 찾을 수 있도록 단순하고 간결하게 구성한다.
② 콘텐츠의 연결이 일관성이 있고 논리적이어야 한다.
③ 세부 콘텐츠를 먼저 배치한 후에 중요한 콘텐츠를 배치한다.
④ 텍스트와 그래픽 요소를 적절히 조화시킨다.

상식적으로 생각해도 풀 수 있는 문제네요. 구성 요소를 배치할 때 세부 콘텐츠와 중요 콘텐츠 중 어떤 것을 먼저 배치해야 할까요?

출제예상

3. 소프트웨어 아키텍처에 관한 설명으로 가장 옳지 않은 것은?

① 아키텍처의 결정은 비기능적 요구사항과 큰 관련이 있다.
② 물리적 구성을 바탕으로 정의하는 시스템의 상세 설계도이다.
③ 초기 계획 단계에서 작성되어 개발 및 유지 보수 작업 등에 영향을 준다.
④ 작업자들 간의 상호 이해, 타협 및 의사 소통을 위해 사용된다.

아키텍쳐는 시스템의 논리적인 구성을 정의하는 하는 것으로, 초기 설계 또는 개략 설계라고도 합니다.

출제예상

4. 소프트웨어 사용자 인터페이스에 대한 설명으로 가장 적합하지 않은 것은?

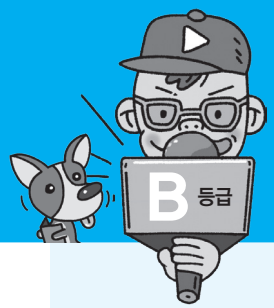
① 좋은 사용자 인터페이스를 설계하려면 실사용자를 분석할 필요가 있다.
② 좋은 사용자 인터페이스를 설계하려면 인간의 사물에 대한 인식 방법, 지각 능력 등을 분석할 필요가 있다.
③ 사용자 인터페이스에 대한 사용자의 요구 사항을 알아야 한다.
④ 모든 사용자에게 동일한 사용자 인터페이스를 제공한다.

사용자 인터페이스는 사용자에 따라 다른 종류의 사용자 인터페이스를 제공할 수 있습니다.

▶ 정답 : 1. ② 2. ③ 3. ② 4. ④

SECTION 011 ● LIVE

UI 표준 및 지침



1 UI 표준 및 지침

UI 표준과 지침을 토대로 기술의 중립성(웹 표준), 보편적 표현 보장성(웹 접근성), 기능의 호환성(웹 호환성)이 고려되었는지 확인한다.

- **UI 표준** : 전체 시스템에 포함된 모든 UI에 공통적으로 적용될 내용으로, 화면 구성이나 화면 이동 등이 포함된다.
- **UI 지침** : UI 요구사항, 구현 시 제약사항 등 UI 개발 과정에서 꼭 지켜야 할 공통의 조건을 의미한다.

웹의 3요소

웹의 3요소는 웹 사이트 개발 시 고려할 사항으로 웹 표준, 웹 접근성, 웹 호환성을 말합니다.

- **웹 표준**(Web Standards) : 웹에서 사용되는 규칙 또는 기술을 의미하는 것으로, 웹 사이트 작성 시 이용하는 HTML, JavaScript 등에 대한 규정, 웹 페이지가 다른 기종이나 플랫폼에서도 구현되도록 제작하는 기법 등을 포함합니다.
- **웹 접근성**(Web Accessibility) : 누구나, 어떠한 환경에서도 웹 사이트에서 제공하는 모든 정보를 접근하여 이용할 수 있도록 보장하는 것을 의미합니다.
- **웹 호환성**(Cross Browsing) : 하드웨어나 소프트웨어 등이 다른 환경에서도 모든 이용자에게 동등한 서비스를 제공하는 것을 의미합니다.

전문가의 조언

- UI 표준과 UI 지침은 '한국형 웹 콘텐츠 접근성 지침서'와 '전자정보 웹 표준 준수 지침서'를 토대로 구성하였습니다. 웹 콘텐츠 설계 시 사용성 확보를 위해 준수해야 할 지침과 전자정부 시스템 구축 시 반영해야 할 최소한의 규약을 숙지해 두세요.
- 사용자 인터페이스(User Interface)를 줄여 'UI'라고 합니다. 본 교재에서는 사용자 인터페이스를 'UI'로 표기하겠습니다.

2 한국형 웹 콘텐츠 접근성 지침(KWCAG; Korean Web Content Accessibility Guidelines)

'한국형 웹 콘텐츠 접근성 지침'은 장애인이 비장애인과 동등하게 접근할 수 있는 웹 콘텐츠의 제작 방법을 제시한다.

- '한국형 웹 콘텐츠 접근성 지침'의 목적은 웹 콘텐츠 저작자, 웹 사이트 설계자 등이 접근성이 보장된 웹 콘텐츠를 쉽게 제작할 수 있도록 도와주는 것이다.
- '한국형 웹 콘텐츠 접근성 지침'에는 웹 접근성의 준수 여부를 평가할 수 있는 요구조건과 이를 모두 준수할 경우 얻을 수 있는 기대 효과가 제시되어 있다.

전문가의 조언

'한국형 웹 콘텐츠 접근성 지침'은 2015년 3월에 미래창조과학부가 발표한 '한국형 웹 콘텐츠 접근성 지침 2.1'을 기준으로 작성되었습니다.

대체 수단
대체 수단은 멀티미디어 콘텐츠에 포함된 음성(대화)을 대체할 수 있는 자막, 대본, 수화 등을 의미합니다.

소리
동영상, 오디오, 음성, 배경 음악 등 콘텐츠에 포함된 모든 소리는 자동으로 재생되면 안되므로 멈춤, 일시정지, 음량 조절 등 제어가 가능한 수단을 함께 제공해야 합니다. 단, 3초 미만으로 재생되는 소리는 자동 재생이 허용됩니다.

광과민성 발작
광과민성 발작은 사람이 장시간 불규칙적으로 깜빡거리거나 번쩍이는 빛의 자극으로 인해 발생하는 발작으로, 초당 3~50회 주기로 깜빡이거나 번쩍이는 콘텐츠는 제공하지 않아야 합니다.

마크업 언어(Markup Language)
마크업 언어는 태그 등을 이용하여 문서의 포맷이나 구조 등을 지정하는 언어로, 종류에는 HTML, SGML, XML 등이 있습니다.

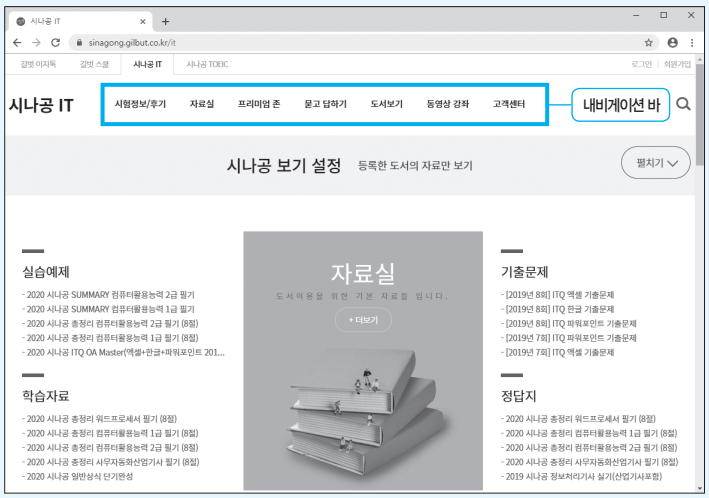
- 웹 콘텐츠 접근성(사용성) 지침 준수를 위한 고려 사항

지침		고려 사항
인식의 용이성	대체 텍스트	텍스트가 아닌 이미지 등의 콘텐츠에는 그 의미를 인식할 수 있는 대체 텍스트를 제공해야 한다.
	멀티미디어 대체 수단*	동영상, 음성 등 멀티미디어 콘텐츠에 대한 이해도를 높일 수 있도록 대체 수단을 제공해야 한다.
	명료성	콘텐츠는 색이나 명도, 방향, 모양, 크기, 소리* 등에 관계없이 명확하게 전달될 수 있어야 한다.
운용의 용이성	키보드 접근성	콘텐츠는 키보드만으로도 접근할 수 있어야 한다.
	충분한 시간 제공	콘텐츠를 읽고 사용하는 데 충분한 시간을 제공해야 한다.
	광과민성 발작* 예방	광과민성 발작을 일으킬 수 있는 콘텐츠는 제공하지 않아야 한다.
	쉬운 내비게이션	반복되는 영역은 건너뛸 수 있도록 하거나 용도나 목적을 이해할 수 있도록 링크 텍스트를 제공하는 등 콘텐츠를 쉽고 편리하게 내비게이션 할 수 있어야 한다.
이해의 용의성	가독성	콘텐츠는 읽고 이해하기 쉬워야 한다.
	예측 가능성	콘텐츠의 기능과 실행 결과는 예측이 가능해야 한다.
	콘텐츠의 논리성	콘텐츠는 선형 구조로 작성되어야 하고, 논리적인 순서를 제공해야 한다.
	입력 도움	입력 오류를 방지하거나 정정할 수 있어야 한다.
견고성	문법 준수	웹 콘텐츠는 마크업 언어(Markup Language)*의 문법을 준수해야 한다.
	접근성	웹 애플리케이션은 접근성이 있어야 한다.

내비게이션(Navigation)

내비게이션은 사용자가 사이트에서 원하는 정보를 빠르게 찾을 수 있도록 안내하는 것으로 사용자가 중심이 되어야 합니다.

- 내비게이션은 원하는 정보를 쉽고 빠르게 찾을 수 있도록 다양한 경로나 방법을 제공해야 합니다.
- 내비게이션은 메뉴, 사이트 맵, 버튼, 링크 등으로 구성되는데, 이들 구성 요소는 사용자가 직관적으로 찾아 사용할 수 있도록 설계되어야 하고, 사용자가 혼동하지 않도록 전체 페이지에서 일관성이 있어야 합니다.
- **내비게이션 구조의 요소**
  - **메뉴(단추)** : 계층 구조를 표현하는 기본 요소로, 사용자가 원하는 페이지로 이동할 수 있게 합니다.
  - **링크** : 원하는 페이지로 이동할 수 있게 하는 하이퍼링크를 의미합니다.
  - **이미지 맵** : 그림에 하이퍼링크를 연결하여 원하는 페이지로 이동할 수 있게 합니다.
  - **사이트 맵** : 사이트의 전체 구조를 한 눈에 알아볼 수 있도록 트리 구조 형태로 만든 것입니다.
  - **사이트 메뉴바** : 사이트의 좌측이나 우측에 메뉴, 링크 등을 모아둔 것입니다.
  - **내비게이션 바** : 메뉴를 한 곳에 모아 놓은 그래픽이나 문자열 모음입니다.
  - **디렉터리** : 주제나 항목을 카테고리별로 표현한 방식입니다.



3 전자정부 웹 표준 준수 지침

'전자정부 웹 표준 준수 지침'은 정부기관의 홈페이지 구축 시 반영해야 할 최소한의 규약을 정의한 것으로, 모든 사람이 시스템 환경에 구애받지 않고 정부기관의 홈페이지를 이용할 수 있도록 하기 위한 것이다.

- '전자정부 웹 표준 준수 지침'에는 이를 준수할 경우의 기대 효과가 제시되어 있다.
- 전자정부 웹 표준 준수 지침 사항

내용의 문법 준수	• 모든 웹 문서는 적절한 문서타입을 명시해야 한다. • 명시한 문서타입에 맞는 문법을 준수해야 한다. • 모든 페이지는 사용할 인코딩※ 방식을 표기해야 한다.
내용과 표현의 분리	• 논리적인 마크업 언어를 사용하여 웹 문서를 구조화해야 한다. • 사용된 스타일 언어는 표준적인 문법을 준수해야 한다.
동작의 기술 중립성 보장	• 스크립트의 비표준 문법을 확장하는 것은 배제해야 한다. • 스크립트 비사용자를 위해 대체 텍스트나 정보를 제공해야 한다.
플러그인※의 호환성	플러그인은 다양한 웹 브라우저에서 호환되는 것을 사용해야 한다.
콘텐츠의 보편적 표현	• 메뉴는 다양한 브라우저에서 접근할 수 있어야 한다. • 웹 사이트를 다양한 인터페이스로 이용할 수 있어야 한다.
운영체제에 독립적인 콘텐츠 제공	제공되는 미디어는 운영체제에 종속적이지 않은 범용적인 포맷을 사용해야 한다.
부가 기능의 호환성 확보	실명인증, 전자인증 등의 부가 기능은 다양한 브라우저에서 사용할 수 있어야 한다.
다양한 프로그램 제공	• 정보를 열람하는 기능은 다양한 브라우저에서 사용할 수 있어야 한다. • 별도의 다운로드가 필요한 프로그램은 윈도우, 리눅스, 맥킨토시 중 2개 이상의 운영체제를 지원해야 한다.

전문가의 조언

'전자정부 웹 표준 준수 지침'은 2008년 4월 행정안전부에서 고시한 지침입니다.

인코딩(Encoding)

인코딩은 문자들의 집합을 컴퓨터에 저장하거나 통신에 사용할 목적으로 부호화하는 방법을 의미합니다.

플러그인(Plug-In)

플러그인은 응용 프로그램에 추가되어 특정한 기능을 수행하도록 구현된 프로그램 모듈을 의미합니다.

기출문제 따라잡기 Section 011

출제예상

1. 장애인이 비장애인과 동등하게 웹 콘텐츠를 접근할 수 있도록 설계하기 위한 지침 사항이 아닌 것은?

① 멀티미디어 콘텐츠에는 자막, 수화 등을 제공해야 한다.

② 차트와 같이 복잡한 콘텐츠는 사용자가 해당 콘텐츠를 파악할 수 있도록 대체 텍스트를 제공해야 한다.

③ 콘텐츠에 포함된 소리는 자동으로 재생되어야 한다.

④ 모든 콘텐츠는 시각적으로 구분될 수 있도록 설계해야 한다.

콘텐츠에 포함된 소리는 사용자가 요구할 경우에만 재생할 수 있도록 해야 합니다.

출제예상

2. 장애 유무 등에 상관없이 웹 사이트에서 제공하는 모든 기능을 운용할 수 있도록 하기 위한 지침 사항이 아닌 것은?

① 모든 기능은 키보드만으로도 사용할 수 있어야 한다.

② 시간제한이 있는 콘텐츠는 응답시간을 조절할 수 있어야 한다.

③ 광과민성 증후가 발생하지 않도록 초당 3~50회 주기로 깜빡이거나 번쩍이는 콘텐츠를 제공해야 한다.

④ 주변 맥락을 통해 용도나 목적지를 명확하게 이해할 수 있도록 링크 텍스트를 제공해야 한다.

초당 3~50회 주기로 깜빡이거나 번쩍이는 콘텐츠는 광과민성 증후가 발생할 수 있으므로 제공하지 말아야 합니다.

출제예상

3. 다음 설명에 해당하는 것은?

> 정보 구조가 완성되면 이들 정보 사이를 자유롭게 이동할 수 있어야 하고, 위계적인 구조 외에도 사이트 맵, 검색 창, 링크 등 다양한 경로와 방법을 통해 원하는 정보를 쉽고 빠르게 접근할 수 있어야 한다.

① 사용자 인터페이스 ② 내비게이션

③ 시나리오 ④ 프로토타입

운전할 때 목적지까지 길을 안내하듯이 웹 사이트에서도 정보를 찾을 수 있도록 도와주는 것이 내비게이션입니다.

출제예상

4. 다음 중 내비게이션에 대한 설명으로 가장 거리가 먼 것은?

① 내비게이션은 전체적으로 일관성이 있도록 만들어야 한다.

② 사용자들의 다양한 설치 환경을 고려해서 만들어야 한다.

③ 전체 내용을 한꺼번에 볼 수 있도록 메뉴는 될 수 있으면 많이 만들어야 한다.

④ 각 페이지를 연결하는 링크가 끊어진 곳이 없어야 한다.

내비게이션으로 화면에 100개가 넘는 메뉴가 있다고 상상해 보세요. 사용자가 원하는 메뉴를 쉽게 찾을 수 있을까요?

출제예상

5. 사용자가 웹 페이지를 쉽게 이동하고 탐색할 수 있도록 해주는 내비게이션 구조의 요소들에 대한 설명이 틀린 것은?

① 이미지 맵 : 웹사이트의 전체 구조를 한눈에 알아볼 수 있도록 트리 구조 형태로 만든 것으로 지도와 같은 역할을 한다.

② 사이트 메뉴바 : 웹사이트의 좌측이나 우측에 메뉴, 링크 등을 모아둔 것을 말한다.

③ 디렉터리 : 주제나 항목을 카테고리 별로 계층적으로 표현하는 방식이다.

④ 내비게이션 바 : 메뉴를 한 곳에 모아놓은 그래픽이나 문자열 모음을 말한다.

사이트의 전체 구조를 한눈에 알아볼 수 있도록 트리 구조 형태로 만드는 것은 사이트 맵입니다.

출제예상

6. 다음 중 전자정부 웹 표준의 준수 지침에 대한 설명으로 틀린 것은?

① 웹 페이지의 모든 문서는 적절한 인코딩 방식을 지정해야 한다.

② 웹 서비스에 사용된 스크립트는 비표준 문법의 확장도 고려해야 한다.

③ 모든 웹 문서는 반드시 문서 타입을 명시해야 한다.

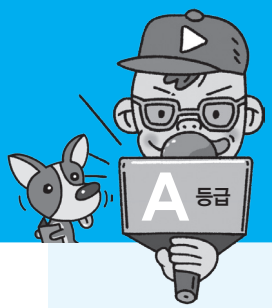
④ 콘텐츠의 구조를 파악하여 구조적인 페이지를 작성한다.

비표준 스크립트는 모든 브라우저에서 정상적으로 작동한다는 보장이 없기 때문에 스크립트의 비표준 문법을 확장하는 것은 배제되어야 합니다.

▶ 정답 : 1. ③ 2. ③ 3. ② 4. ③ 5. ① 6. ②

SECTION
012
LIVE

UI 설계 도구



1 UI 설계 도구

UI 설계 도구는 사용자의 요구사항에 맞게 UI의 화면 구조나 화면 배치 등을 설계할 때 사용하는 도구로, 종류에는 와이어프레임, 목업, 유스케이스, 프로토타입, 스토리보드 등이 있다.

- UI 설계 도구로 작성된 결과물은 사용자의 요구사항이 실제 구현되었을 때 화면은 어떻게 구성되는지, 어떤 방식으로 수행되는지 등을 기획단계에서 미리 보여주기 위한 용도로 사용된다.

2 와이어프레임(Wireframe)

와이어프레임은 기획 단계의 초기에 제작하는 것으로, 페이지에 대한 개략적인 레이아웃이나 UI 요소 등에 대한 뼈대를 설계하는 단계이다.

- 와이어프레임을 제작할 때는 각 페이지의 영역 구분, 콘텐츠, 텍스트 배치 등을 화면 단위로 설계한다.
- 개발자나 디자이너 등이 레이아웃을 협의하거나 현재 진행 상태 등을 공유하기 위해 와이어프레임을 사용한다.
- **와이어프레임 툴** : 손그림, 파워포인트, 키노트, 스케치, 일러스트, 포토샵 등

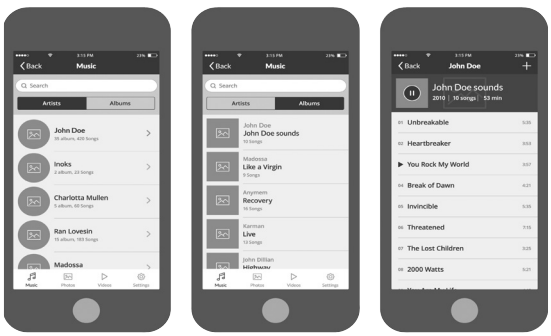
와이어프레임 작성 예

전문가의 조언

건물을 짓기 전에 건물에 대한 설계도를 그리듯이 UI를 제작할 때도 와이어프레임, 목업, 프로토타입 등을 이용하여 UI에 대한 설계를 먼저 해야 합니다. UI 설계 도구의 종류를 기억하세요. 그리고 각각의 특징을 서로 구분할 수 있을 정도로만 알아두세요.

3 목업(Mockup)

- 목업은 디자인, 사용 방법 설명, 평가 등을 위해 와이어프레임보다 좀 더 실제 화면과 유사하게 만든 정적인 형태의 모형이다.
- 시각적으로만 구성 요소를 배치하는 것으로 실제로 구현되지는 않는다.
- **목업 툴** : 파워 목업, 발사믹 목업 등

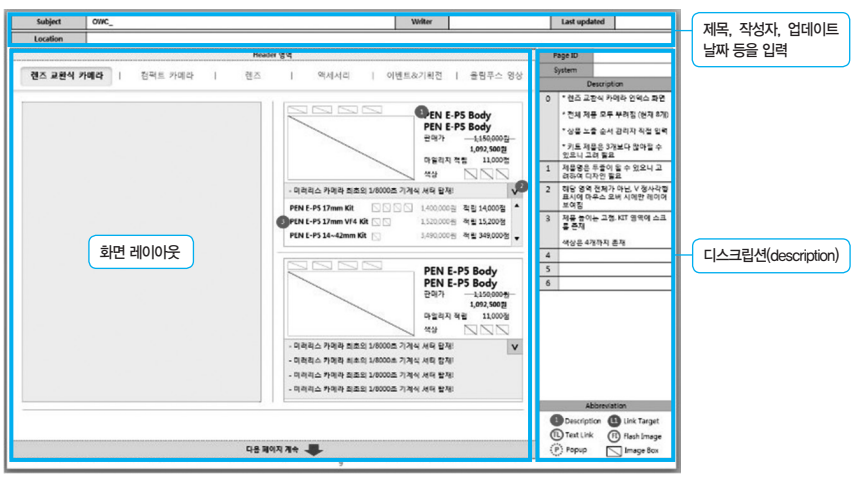


목업 작성 예

4 스토리보드(Story Board)

스토리보드는 와이어프레임에 콘텐츠에 대한 설명, 페이지 간 이동 흐름 등을 추가한 문서이다.

- 디자이너와 개발자가 최종적으로 참고하는 작업 지침서로, 정책, 프로세스, 콘텐츠 구성, 와이어프레임, 기능 정의 등 서비스 구축을 위한 모든 정보가 들어 있다.
- 스토리보드는 상단에는 제목, 작성자 등을 입력하고, 좌측에는 UI 화면, 우측에는 디스크립션(Description)을 기입한다.
- 디스크립션(Description)은 화면에 대한 설명, 전반적인 로직, 분기처리, 예외처리 등을 작성하는 부분으로, 명확하고 세부적으로 작성해야 한다.
- **스토리보드 툴** : 파워포인트, 키노트, 스케치, Axure 등



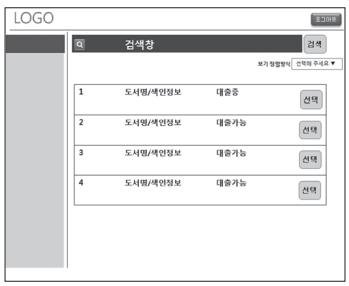
스토리보드 작성 예

5 프로토타입(Prototype)

프로토타입은 와이어프레임나 스토리보드 등에 인터랙션*을 적용함으로써 실제 구현된 것처럼 테스트가 가능한 동적인 형태의 모형이다.

- 프로토타입은 사용성 테스트나 작업자 간 서비스 이해를 위해 작성하는 샘플이다.
- 프로토타입은 작성 방법에 따라 페이퍼 프로토타입과 디지털 프로토타입으로 나뉜다.
- **프로토타입 툴** : HTML/CSS, Axure, Flinto, 네이버 프로토나우, 카카오 오븐 등

페이퍼 프로토타입

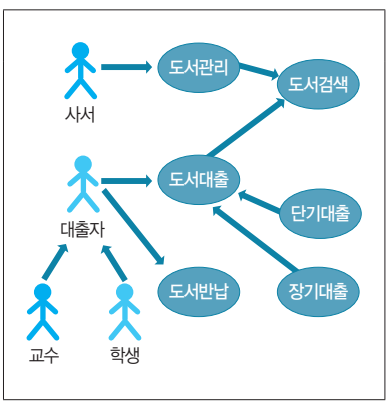


디지털 프로토타입

6 유스케이스(Use Case)

유스케이스는 사용자 측면에서의 요구사항으로, 사용자가 원하는 목표를 달성하기 위해 수행할 내용을 기술한다.

- 사용자의 요구사항을 빠르게 파악함으로써 프로젝트의 초기에 시스템의 기능적인 요구를 결정하고 그 결과를 문서화할 수 있다.
- 유스케이스는 자연어로 작성된 사용자의 요구사항을 구조적으로 표현한 것으로, 일반적으로 다이어그램 형식으로 묘사된다.
- 유스케이스 다이어그램이 완성되면, 각각의 유스케이스에 대해 유스케이스 명세서를 작성한다.



유스케이스 다이어그램

도서 대출 예약 시스템의 도서 반납 유스케이스(Use CASE)

1. 개요
   사용자는 대출반납 기기를 통하여 원하는 도서를 반납한다.
2. 액터
   사용자
3. 이벤트 흐름
   (1) 기본 사항
   (가) 사용자는 '도서대출/반납메인화면'에서 반납 버튼을 누른다.
   (나) 시스템은 반납할 도서의 인식을 요청하는 '반납도서인식요청화면'을 출력한다.
   (다) 사용자는 반납하고자 하는 도서를 대출반납 기기에 인식시킨다.
   (라) 시스템은 데이터베이스에서 대출 정보를 수정한다.
   (마) 시스템은 반납 결과를 보여주는 '반납결과화면'을 출력한다.
   (바) 사용자는 확인 버튼을 누른다.
   (2) 추가 사항
   (가) 사용자가 '반납도서인식요청화면'에서 취소를 누를 경우
   (나) 시스템은 '도서대출/반납메인화면'을 출력한다.
4. 처리내용
   데이터베이스에 대출 정보가 수정된다.

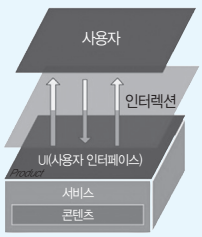
유스케이스 명세서

전문가의 조언

프로토타입은 Section 015에서 자세히 공부하니 이번 섹션에서는 다른 UI 설계 도구와의 차이점 정도만 알아두세요.

인터랙션(Interaction)

사용자와 시스템을 연결하는 것이 UI라면 인터랙션은 UI를 통해 시스템을 사용하는 일련의 상호 작용을 의미합니다. 쉽게 말해 마우스로 화면의 어떤 아이콘을 클릭하면 화면이 그에 맞게 반응하는 것을 말합니다.



출제예상

1. 다음 중 사용자 인터페이스를 설계할 때 사용하는 툴이 아닌 것은?

① 파워포인트
② 파워 목업
③ 드림위버
④ 액슈어(Axcure)

드림위버는 웹페이지 개발에 사용하는 응용 프로그램입니다.

출제예상

2. UI를 설계할 때 화면 단위로 전개될 가상 경로를 예상하여 기획하는 것으로, 화면 설계도이며 구체적인 작업 지침서 역할을 하는 것은?

① 유스케이스 ② 레이아웃
③ 내비게이션 ④ 스토리보드

구체적인 작업 지침서는 무엇일까요?

출제예상

3. 유스케이스에 관한 설명으로 잘못된 것은?

① 사용자의 요구사항을 정리하고 기록하기 위한 도구이다.
② 와이어프레임에 인터랙션을 적용한 모형이다.
③ 유스케이스는 일반적으로 다이어그램 형식으로 작성된다.
④ 완성된 유스케이스에 대해 유스케이스 명세서를 작성한다.

와이어프레임에 인터랙션을 적용한 동적인 형태의 모형은 프로토타입입니다.

출제예상

4. 다음 중 사용자 인터페이스(User Interface)의 설계 도구에 대한 설명으로 틀린 것은?

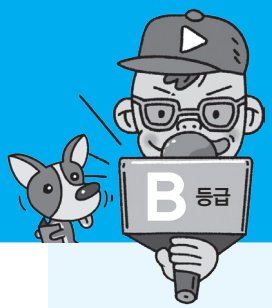
① 화면 설계 도구에는 파워포인트, 스토리보드, 와이어프레임, 목업 등이 있다.
② 와이어프레임(Wireframe)은 기획 단계에서 페이지 레이아웃이나 구성 요소 등 뼈대를 설계하는 단계이다.
③ 목업(Mockup)은 와이어프레임의 내용에 디스크립션을 추가한 문서이다.
④ 프로토타입(Prototype)은 테스트가 가능하도록 만든 일종의 샘플이다.

와이어프레임에 디스크립션을 추가한 것은 스토리보드입니다. 목업은 와이어프레임에 비해 실제 화면과 좀 더 유사하지만 디스크립션을 표시하지는 않습니다.

▶ 정답 : 1. ③ 2. ④ 3. ② 4. ③

SECTION 013 LIVE

UI 요구사항 확인



1 UI 요구사항 확인

UI 요구사항 확인은 새로 개발할 시스템에 적용할 UI 관련 요구사항을 조사해서 작성하는 단계로, 다양한 경로를 통해 사용자의 요구사항을 조사하고 분석한 후 작성해야 한다.

- UI 요구사항 확인 순서는 다음과 같다.

목표 정의 → 활동 사항 정의 → UI 요구사항 작성

전문가의 조언

사용자가 요구하는 UI를 설계하기 위해 가장 중요한 것은 사용자의 요구사항을 엄밀히 분석하고 정확히 이해한 후 설계 작업을 수행하는 것입니다. 사용자의 요구사항을 조사하여 분석하고 작성하기까지의 내용을 숙지해 두세요.

2 목표 정의

목표 정의 단계에서는 사용자들을 대상으로 인터뷰를 진행한 후 사용자들의 의견이 수렴된 비즈니스 요구사항을 정의한다.

- 인터뷰를 통해 사업적, 기술적인 요구사항을 명확히 이해한다.
- **인터뷰 진행 시 유의사항**
  - 인터뷰는 가능하면 개별적으로 진행한다.
  - 가능한 많은 사람을 인터뷰하여 다양한 의견을 수렴하되, 다수의 의견으로 인해 개인의 중요한 의견을 놓치지 않도록 주의한다.
  - 인터뷰는 한 시간을 넘지 않도록 한다.
  - 인터뷰 진행은 반드시 사용자 리서치*를 시작하기 전에 해야 한다.

리서치(Research)

리서치는 사용자들의 요구사항이나 불편사항 등을 파악하기 위해 진행되는 것으로, 리서치 전에 인터뷰를 진행함으로써 효과적인 리서치를 계획할 수 있습니다. 리서치는 설문조사, 개인 인터뷰 등으로 진행됩니다.

3 활동 사항 정의

활동 사항 정의 단계에서는 조사한 요구사항을 토대로 앞으로 해야 할 활동 사항을 정의한다.

- 사용자와 회사의 비전을 일치시키는 작업을 진행한다.
- 리서치 규모, 디자인 목표 등을 결정할 수 있도록 각각에 필요한 예산과 일정을 결정한다.
- 기술의 발전 가능성을 파악하고 UI 디자인의 방향을 제시한다.
- 인터뷰한 내용을 기반으로 경영진마다 다르게 이해하고 있는 프로젝트에 대해 정확히 이해하고 협의하도록 돕는다.
- 사업 전략 및 목표, 프로세스의 책임자 선정, 회의 일정 및 계획 작성, 우선순위의 선정, 개별적인 단위 업무를 구분한다.

4 UI 요구사항 작성

UI 요구사항을 작성할 때는 여러 경로를 통해 수집된 사용자들의 요구사항을 검토하고 분석하여 UI 개발 목적에 맞게 작성해야 한다.

- UI 요구사항은 반드시 실사용자 중심으로 작성되어야 한다.
- UI 요구사항은 여러 사람의 인터뷰를 통해 다양한 의견을 수렴해서 작성해야 한다.
- UI 요구사항을 바탕으로 UI의 전체적인 구조를 파악 및 검토해야 한다.
- UI 요구사항 작성 순서는 다음과 같다.

요구사항 요소 확인 → 정황 시나리오 작성 → 요구사항 작성

5 요구사항 요소 확인

파악된 요구사항 요소의 종류와 각각의 표현 방식 등을 검토한다.

- 요구사항 요소

데이터 요구	• 사용자가 요구하는 모델과 객체들의 주요 특성을 기반으로 하여 데이터 객체들을 정리한다. • 인터페이스 구성에 영향을 미치므로 반드시 초기에 확인해야 한다. 예 이메일의 메시지 속성은 제목, 발신일, 발신인, 참조인, 답변 등이다.
기능 요구	• 사용자의 목적 달성을 위해 무엇을 실행해야 하는지를 동사형으로 설명한다. • 기능 요구 리스트는 최대한 철저하게 정리해야 한다. 예 사용자는 이메일의 메시지를 읽거나 삭제하며, 일정한 양식으로 다른 메시지와 함께 보관한다.
제품/서비스의 품질	데이터 및 기능 요구 외에 제품의 품질, 서비스, 여기에 감성적인 품질 등을 고려하여 작성한다. 예 시스템이 파일을 얼마나 빠르게 처리할 수 있는지 여부 등 정량화가 가능한 요구사항들을 확인한다.
제약 사항	• 제품 완료 데드라인, 전체 개발 및 제작에 필요한 비용, 시스템 준수에 필요한 규제가 포함된다. • 사전에 제약사항의 변경 가능 여부를 확인한다.

전문가의 조언

정황 시나리오는 사용자의 어떤 요구사항이 있을 때 이것을 만족하기 위해 사용자가 수행하는 과정을 이야기 형식으로 표현한 것입니다.

예 책을 구입하기 위해 핸드폰으로 구입할 책이 있는 서점을 검색했다. 서점 위치와 교통편을 확인한 후 서점으로 이동했다. 책이 있는 위치를 검색하여 책을 찾아 계산대로 이동해 결제했다.

6 정황 시나리오 작성

정황 시나리오는 사용자의 요구사항을 도출하기 위해 작성하는 것으로, 사용자가 목표를 달성하기 위해 수행하는 방법을 순차적으로 묘사한 것이다.

- 정황 시나리오는 요구사항 정의에 사용되는 초기 시나리오이다.
- 정황 시나리오는 개발하는 서비스의 모습을 상상하는 첫 번째 단계로 사용자 관점에서 시나리오를 작성해야 한다.
- 사용자가 주로 사용하는 기능 위주로 작성해야 하며, 함께 발생되는 기능들은 하나의 시나리오에 통합한다.

- 육하원칙에 따라 간결하고 명확하게 작성한다.
- 작성된 시나리오는 외부 전문가 또는 경험이 풍부한 사람에게 검토를 의뢰한다.

7 요구사항 작성

요구사항은 정황 시나리오를 토대로 작성한다.

예 요구사항 작성 예

정황 시나리오	요구사항
윤희는 회의가 끝난 후 핸드폰을 켰다. 주요 회의 내용을 메모하고 다음 약속을 확인하는 한편, 회의 동안 중요한 전화가 있었는지 확인했다.	• 문자를 입력할 수 있어야 한다. • 약속을 추적할 수 있어야 한다. • 메시지 리스트를 확인할 수 있어야 한다. • 핸드폰으로 구현이 가능해야 한다.

기출문제 따라잡기 Section 013

출제예상

1. 사용자의 요구사항을 조사하기 위해 인터뷰를 진행하려고 한다. 인터뷰 진행 시 유의사항이 아닌 것은?

① 될 수 있으면 많은 사람을 인터뷰하기 위해 그룹별 인터뷰를 진행해야 한다.
② 인터뷰를 통해 사업적, 기술적인 요구사항을 명확히 이해해야 한다.
③ 인터뷰는 될 수 있으면 한 시간을 넘지 않도록 해야 한다.
④ 인터뷰는 사용자 리서치를 하기 전에 먼저 진행해야 한다.

인터뷰는 개별적으로 진행해야 합니다.

출제예상

2. 사용자들의 UI 요구사항을 작성하려고 한다. 다음 중 틀린 것은?

① 요구사항은 사용자 인터뷰를 통해서만 수집해야 한다.
② 요구사항 작성 관점은 반드시 실사용자 중심이어야 한다.
③ 요구사항은 여러 사람의 인터뷰를 통해 다양한 의견을 수렴하여 작성하는 것이 매우 중요하다.
④ 정황 시나리오를 작성한 후 이를 토대로 요구사항을 작성한다.

UI 요구사항은 인터뷰 외에 리서치 등 여러 경로를 통해 수집해야 합니다.

출제예상

3. 다음 중 정황 시나리오에 대한 내용으로 틀린 것은?

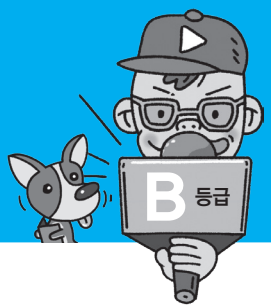
① 정황 시나리오는 육하원칙에 따라 간결하고 명확하게 작성해야 한다.
② 정황 시나리오는 요구사항 정의에 사용되는 시나리오 초안이다.
③ 정황 시나리오는 개발할 프로그램 관점에서 기능 위주로 작성해야 한다.
④ 정황 시나리오를 작성한 이후에는 경험이 풍부한 사람에게 검토를 의뢰하는 것이 좋다.

정황 시나리오는 사용자의 요구사항을 도출하기 위한 것으로 사용자 관점에서 기능 위주로 작성해야 합니다.

▶ 정답 : 1. ① 2. ① 3. ③

SECTION 014 ● LIVE

품질 요구사항



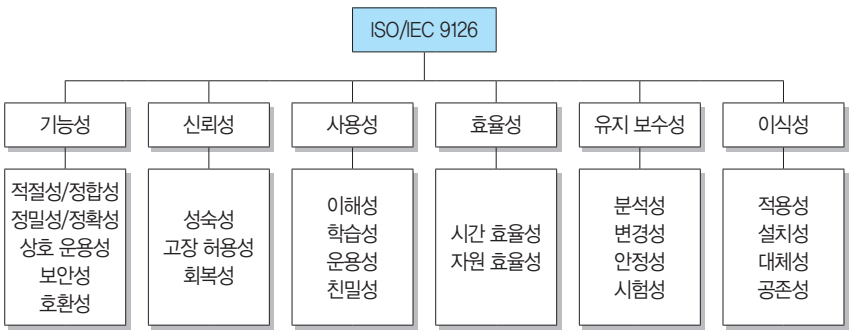
전문가의 조언

ISO/IEC 9126은 사용자 관점에서 본 소프트웨어 품질의 특성에 대한 표준으로, 총 6개의 특성과 각 특성에 대한 부특성으로 이뤄져 있습니다. 먼저 품질 특성의 종류 6가지를 기억하고, 그에 해당하는 부특성을 파악해 두세요.

1 품질 요구사항

소프트웨어의 품질은 소프트웨어의 기능, 성능, 만족도 등 소프트웨어에 대한 요구사항이 얼마나 충족하는가를 나타내는 소프트웨어 특성의 총체이다.

- 소프트웨어의 품질은 사용자의 요구사항을 충족시킴으로써 확립된다.
- **ISO/IEC 9126**
  - ISO/IEC 9126은 소프트웨어의 품질 특성과 평가를 위한 표준 지침으로서 국제 표준으로 널리 사용된다.
  - ISO/IEC 9126은 소프트웨어의 품질에 대한 요구사항을 기술하거나 개발중인 또는 개발이 완료된 소프트웨어의 품질 평가 등에 사용된다.
  - ISO/IEC 9126은 2011년에 호환성과 보안성을 강화하여 ISO/IEC 25010으로 개정되었다.
  - ISO/IEC 9126에서 제시한 소프트웨어의 품질 특성



ISO/IEC 25010

ISO/IEC 25010은 2011년 ISO/IEC 9126을 개정하여 만든 소프트웨어 제품에 대한 국제 표준입니다.

- ISO/IEC 25010은 ISO/IEC 9126에 비해 호환성과 보안성 부분이 강화되었습니다.
- **ISO/IEC 9126의 품질 특성** : 기능성, 신뢰성, 사용성, 효율성, 유지보수성, 이식성
- **ISO/IEC 25010의 품질 특성** : 신뢰성, 사용성, 이식성, 유지보수성, 기능 적합성, 실행 효율성, 호환성, 보완성

2 기능성(Functionality)

기능성은 소프트웨어가 사용자의 요구사항을 정확하게 만족하는 기능을 제공하는지 여부를 나타낸다.

상세 품질 요구사항	설명
적절성/정합성(Suitality)	지정된 작업과 사용자의 목적 달성을 위해 적절한 기능을 제공할 수 있는 능력
정밀성/정확성(Accuracy)	사용자가 요구하는 결과를 정확하게 산출할 수 있는 능력
상호 운용성(Interoperability)	다른 시스템들과 서로 어울려 작업할 수 있는 능력
보안성(Security)	정보에 대한 접근을 권한에 따라 허용하거나 차단할 수 있는 능력
호환성(Compliance)	기능과 관련된 표준, 관례 및 규정을 준수할 수 있는 능력

3 신뢰성(Reliability)

신뢰성은 소프트웨어가 요구된 기능을 정확하고 일관되게 오류 없이 수행할 수 있는 정도를 나타낸다.

상세 품질 요구사항	설명
성숙성(Maturity)	결함으로 인한 고장을 피해갈 수 있는 능력
고장 허용성(Fault Tolerance)	결함 또는 인터페이스 결여 시에도 규정된 성능 수준을 유지할 수 있는 능력
회복성(Recoverability)	고장 시 규정된 성능 수준까지 다시 회복하고 직접적으로 영향 받은 데이터를 복구할 수 있는 능력

4 사용성(Usability)

사용성은 사용자와 컴퓨터 사이에 발생하는 어떠한 행위에 대하여 사용자가 정확하게 이해하고 사용하며, 향후 다시 사용하고 싶은 정도를 나타낸다.

상세 품질 요구사항	설명
이해성(Understandability)	소프트웨어의 적합성, 사용 방법 등을 사용자가 이해할 수 있는 능력
학습성(Learnability)	소프트웨어 애플리케이션을 학습할 수 있도록 하는 능력
운용성(Operability)	사용자가 소프트웨어를 운용하고 제어할 수 있도록 하는 능력
친밀성(Attractiveness)	사용자가 소프트웨어를 다시 사용하고 싶어 하도록 하는 능력

5 효율성(Efficiency)

효율성은 사용자가 요구하는 기능을 할당된 시간 동안 한정된 자원으로 얼마나 빨리 처리할 수 있는지 정도를 나타낸다.

상세 품질 요구사항	설명
시간 효율성 (Time Behaviour)	특정 기능을 수행할 때 적절한 반응 시간 및 처리 시간, 처리율을 제공할 수 있는 능력
자원 효율성 (Resource Behaviour)	특정 기능을 수행할 때 적절한 자원의 양과 종류를 제공할 수 있는 능력

전문가의 조언

각 품질 특성에는 품질 특성과 관련된 표준, 관례, 및 규정을 준수할 수 있는 능력인 '호환성(Compliance)'이 포함되어 있으나, 동일한 내용이므로 기능성에만 포함하였습니다.

6 유지 보수성(Maintainability)

유지 보수성은 환경의 변화 또는 새로운 요구사항이 발생했을 때 소프트웨어를 개선하거나 확장할 수 있는 정도를 나타낸다.

상세 품질 요구사항	설명
분석성(Analyzability)	결함이나 고장의 원인, 수정될 부분들의 식별을 가능하게 하는 능력
변경성(Changeability)	결함 제거 또는 환경 변화로 인한 수정 등을 쉽게 구현할 수 있는 능력
안정성(Stability)	변경으로 인한 예상치 못한 결과를 최소화할 수 있는 능력
시험성(Testability)	소프트웨어의 변경이 검증될 수 있는 능력

7 이식성(Portability)

이식성은 소프트웨어가 다른 환경에서도 얼마나 쉽게 적용할 수 있는지 정도를 나타낸다.

상세 품질 요구사항	설명
적용성(Adaptability)	원래의 목적으로 제공되는 것 외에 다른 환경으로 변경될 수 있는 능력
설치성(Installability)	임의의 환경에 소프트웨어를 설치할 수 있는 능력
대체성(Replaceability)	동일한 환경에서 동일한 목적을 위해 다른 소프트웨어를 대신하여 사용될 수 있는 능력
공존성(Co-existence)	자원을 공유하는 환경에서 다른 소프트웨어와 공존할 수 있는 능력

기출문제 따라잡기 Section 014

출제예상

1. ISO/IEC 9126에서 정의하고 있는 6가지의 품질 특성이 아닌 것은?

① 기능성 ② 신뢰성
③ 재사용성 ④ 유지 보수성

ISO/IEC 9126의 품질 특성 6가지는 기능성, 신뢰성, 사용성, 효율성, 유지 보수성, 이식성! 꼭 기억하세요.

출제예상

2. 소프트웨어 품질 체계인 ISO/IEC 9126의 품질 특성 중 신뢰성(Reliability)에 속하지 않는 특성은?

① 성숙성(Maturity)
② 결함 허용성(Fault Tolerance)
③ 회복성(Recoverability)
④ 정확성(Accuracy)

정확성은 ISO 9126의 품질 특성 중 기능성에 속합니다.

출제예상

3. ISO 9126에서 정의한 소프트웨어가 갖추어야 할 품질의 특성 중 정확하고 일관된 결과를 오류 없이 수행할 수 있는 정도를 나타내는 척도는?

① 효율성 ② 사용 용이성
③ 신뢰성 ④ 이식성

얼마나 믿고 사용할 수 있는지를 나타내는 것은 무엇일까요?

출제예상

4. ISO/IEC 9126의 품질 특성 중 시스템 운영 환경의 변화나 사용자의 새로운 요구사항에 따라 소프트웨어를 쉽게 변경할 수 있는 특성으로 가장 옳은 것은?

① 재사용성 ② 사용성
③ 유지 보수성 ④ 신뢰성

소프트웨어를 사용하다보면 소프트웨어를 수정해야 할 일이 발생합니다. 이를 수월하게 할 수 있는 정도를 나타내는 것은 무엇일까요?

▶ 정답 : 1. ③ 2. ④ 3. ③ 4. ③

SECTION
015
LIVE

UI 프로토타입 제작 및 검토

1 UI 프로토타입의 개요

프로토타입*은 사용자 요구사항을 기반으로 실제 동작하는 것처럼 만든 동적인 형태의 모형으로, 테스트가 가능하다.

- 프로토타입은 사용자의 요구사항을 개발자가 맞게 해석했는지 검증하기 위한 것으로, 최대한 간단하게 만들어야 한다.
- 프로토타입은 일부 핵심적인 기능만을 제공하지만 최종 제품의 작동 방식을 이해시키는데 필요한 기능은 반드시 포함되어야 한다.
- 사용자의 요구사항이 모두 반영될 때까지 프로토타입을 계속하여 개선하고 보완해야 한다.
- 프로토타이핑* 및 테스트를 거치지 않고는 실제 사용자와 제품 간의 상호 작용 방식을 예측하기 어려우므로 실제 사용자를 대상으로 테스트하는 것이 좋다.

2 UI 프로토타입의 장 · 단점

장점	• 사용자를 설득하고 이해시키기 쉽다. • 요구사항과 기능의 불일치 등으로 인한 혼선을 예방할 수 있어 개발 시간을 줄일 수 있다. • 사전에 오류를 발견할 수 있다.
단점	• 프로토타입에 사용자의 모든 요구사항을 반영하기 위한 반복적인 개선 및 보완 작업 때문에 작업 시간을 증가 시킬 수 있고, 필요 이상으로 자원을 소모할 수 있다. • 부분적으로 프로토타이핑을 진행하다보면 중요한 작업이 생략될 수 있다.

3 프로토타이핑의 종류

페이퍼 프로토타입 (Paper Prototype)	• 아날로그적인 방법으로, 스케치, 그림, 글 등을 이용하여 손으로 직접 작성하는 방법이다. • 제작 기간이 짧은 경우, 제작 비용이 적을 경우, 업무 협의가 빠를 경우 사용한다. • 장점 – 비용이 저렴하다. – 회의 중 대화하면서 생성이 가능하다. – 즉시 변경이 가능하다. – 고객이 과다한 기대를 하지 않는다. • 단점 – 테스트하기에 부적당하다. – 상호 관계가 많은 경우 나타내기 복잡하다. – 여러 사람들에게 나눠주거나 공유하기 어렵다.

전문가의 조언

UI 프로토타입(Prototype)은 완전한 UI를 만들기 전에 사용자와의 의사소통을 위해 만든 모형입니다. 백문이 불여일견이라고 했습니다. 전문 지식이 없는 사용자에게 백번 말로 설명하는 것 보다 실제로 동작하는 것처럼 보이는 모형을 한 번 보여주는 것이 훨씬 더 이해가 빠릅니다. UI 프로토타입은 쉽게 말해 최종적으로 구현할 화면의 주요 기능을 직접 실행해 볼 수 있도록 미리 만들어본 임시 화면이라고 생각하면 됩니다. UI 프로토타입은 중요한 개념입니다. UI 프로토타입의 개념을 중심으로 장/단점, 종류 등을 모두 기억해 두세요.

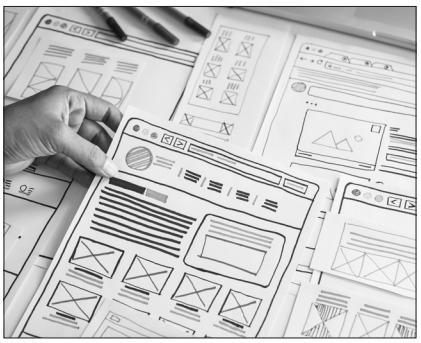
프로토타입

이 섹션에서 언급되는 프로토타입은 모두 UI 프로토타입을 의미합니다.

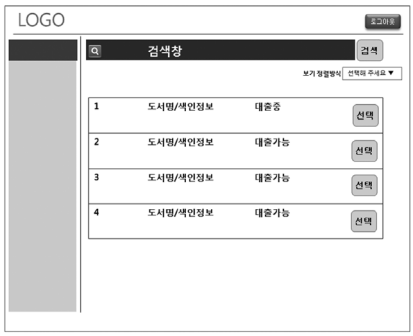
프로토타이핑

프로토타이핑은 프로토타입을 만드는 과정으로, 사용자의 요구사항 검토부터 최종적인 프로토타입을 완성하기 까지 반복적으로 수행되는 전 과정을 의미합니다.

디지털 프로토타입 (Digital Prototype)	• 파워포인트, 아크로뱃, 비지오, 옴니그래플 등과 같은 프로그램을 사용하여 작성하는 방법이다. • 재사용이 필요한 경우, 산출물과 비슷한 효과가 필요한 경우, 숙련된 전문가가 있을 경우 사용한다. • 장점 – 최종 제품과 비슷하게 테스트할 수 있다. – 수정하기 쉽다. – 재사용이 가능하다. • 단점 – 프로토타입을 작성할 프로그램의 사용법을 알아야 한다.



페이퍼 프로토타입



디지털 프로토타입

4 UI 프로토타입 계획 및 작성 시 고려 사항

프로토타입은 일반적으로 프로토타입의 개발 계획을 수립하는 과정과 프로토타입을 개발한 후 결과를 보고하는 과정으로 진행된다.

계획 시 고려 사항	• 프로토타입의 개발 목적을 확인한다. • 소프트웨어, 하드웨어 등 프로토타입 개발에 필요한 환경을 마련한다. • 프로토타이핑 일정*은 일반적으로 아키텍처가 확정된 이후 프로젝트의 실제 분석 작업이 완료되기 이전에 진행해야 한다. • 아키텍처의 핵심이 되는 UI 요소를 프로토타입의 범위로 잡는다. • 리더, 솔루션 담당자, 인프라 담당자, 개발 환경 리더, 공통 모듈 개발자, 프로토타입 개발자 등 프로토타입의 개발 인원*을 확인한다. • 주어진 비즈니스 요구사항을 모두 만족하는지 프로토타입 아키텍처를 검증한다. • 프로토타입을 통해서 발생하는 이슈*를 모두 취합하고 해결 방법을 제시한다. • 프로토타이핑을 진행하면서 분석, 설계, 개발, 테스트 등의 표준 가이드를 확정한다. • 프로토타이핑을 진행하면서 가장 많은 시간이 소요된 구간을 찾고 그 원인을 분석하여 해결 방법을 제시한다. • 고객과 프로젝트 매니저, 프로젝트 리더 등에게 완성된 프로토타입을 시연한다.
작성 시 고려 사항	• 프로토타입의 작성 계획을 세운다. • 프로젝트의 범위나 리스크 상황 등 주변 여건을 감안해서 프로토타입의 범위를 정한다. • 프로토타입을 통해서 얻고자 하는 목표를 확인한다. • 프로토타입의 개발 목표를 달성하기 위해 필요한 최소한의 기간과 비용을 확인한다. • 완성된 프로토타입이 실제 개발에 참조될 수 있는지 확인한다. • 프로토타입으로 검증할 범위가 너무 넓거나 기간이 길면 목표가 커져서 문제가 될 수 있으니 주의한다.

대형 프로젝트의 프로토타이핑 일정

- 프로토타입의 개발 목적이 아키텍처 검증인 경우 : 프로젝트 개발 이전에 완료하거나 대략 1개월 정도로 잡는 것이 좋음
- 프로토타입의 개발 목적이 분석, 설계 가이드에 대한 검증인 경우 : 2개월을 추가할 수 있음

대형 프로젝트의 프로토타입 개발 인원

프로토타입 개발 인원은 대형 프로젝트를 기준으로 하면 리더 1인, 솔루션 담당자 파트타임 2인 이상, 인프라 담당자 파트타임 1인, 개발 환경 리더 겸 공통 모듈 개발자 1인, 프로토타입 개발자 3~5인으로 구성됩니다.

프로토타입 이슈

프로토타입을 통해서 발생하는 이슈는 대부분 소프트웨어 아키텍처의 요소를 검증하는 과정에서 발생합니다. 이슈는 많이 발생할수록 좋은 것으로 프로토타입 리더는 발생한 이슈를 종류별로 취합하고 해결 방법을 제시해야 하며 이것을 모두 정리해서 보고해야 합니다.

5 UI 프로토타입 제작 단계

1단계	사용자의 요구사항을 분석하는 단계로, 사용자 관점에서 기본적인 요구사항이 확정될 때까지 수행한다.
2단계	• 요구사항을 충족하는 프로토타입을 종이에 손으로 직접 그리거나 편집 도구 등을 이용하여 작성한다. • 프로토타입은 개발할 시스템의 핵심적인 기능을 중심으로 개발한다.
3단계	• 작성된 프로토타입이 요구사항을 잘 수행하고 있는지 사용자가 직접 확인하는 단계이다. • 프로토타입에 대해 다양한 추가 및 수정 의견을 제안할 수 있다.
4단계	• 작성된 프로토타입을 기반으로 수정과 합의가 이뤄지는 단계이다. • 개발자는 사용자가 요청한 제안 사항을 수용하여 보완 작업을 한다. • 작업이 완료된 후 3단계로 되돌아간다. • 사용자가 최종적으로 승인을 완료할 때까지 3단계와 4단계가 반복된다.

기출문제 따라잡기 Section 015

출제예상

1. 다음은 무엇에 대한 설명인가?

> • 사용자의 요구사항을 취합하여 만드는 모형이다.
> • 가장 단순한 형태로는 종이에 화면 순서를 기술하여 사용자에게 무엇이 어떻게 일어나는지를 보여주는 것이다.

① 관찰 ② 유스케이스
③ 브레인스토밍 ④ 프로토타입

이 문제의 핵심은 '사용자의 요구사항을 취합해서 만든 모형'입니다.

출제예상

2. UI 프로토타입에 대한 설명으로 틀린 것은?

① 프로토타입은 사용자의 요구사항을 기반으로 만든 모형으로 테스트가 가능하다.
② 일부 핵심적인 기능만을 제공하는 프로토타입을 작성하면 중요한 기능이 생략될 수 있으므로 반드시 전체를 대상으로 프로토타입을 작성해야 한다.
③ 프로토타입을 통해 UI의 개발 시간을 단축시키고 개발 전에 오류를 발견할 수 있다.
④ 프로토타입은 사용자의 요구사항을 개발자가 맞게 해석했는지 검증하기 위한 것으로, 최대한 간단하게 만드는 것이 좋다.

프로토타입은 최소한의 기능만을 부분적으로 제공할 수 있습니다.

출제예상

3. 사용자의 요구사항을 기반으로 하여 UI 프로토타이핑을 진행하려고 한다. 다음 설명 중 틀린 것은?

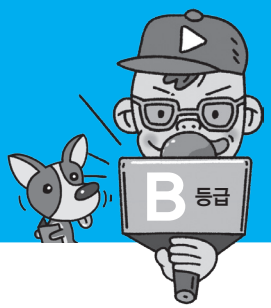
① 프로토타이핑은 프로젝트의 실제 분석 작업이 완료된 이후 진행해야 한다.
② 프로토타입을 통해 발생하는 이슈는 모두 취합하고 해결 방법을 제시해야 한다.
③ 프로토타이핑을 진행하면서 가장 많은 시간이 소요된 구간을 찾고 그 원인을 분석하여 해결 방법을 제시한다.
④ 프로토타이핑을 진행하면서 표준 가이드를 확정한다.

프로토타이핑은 일반적으로 아키텍처가 확정된 이후 프로젝트의 실제 분석 작업이 완료되기 이전에 진행해야 합니다.

▶ 정답 : 1. ④ 2. ② 3. ①

SECTION 016 ● LIVE

UI 설계서 작성



전문가의 조언

UI 설계서와 UI 상세 설계 시 작성하는 UI 화면 설계가 명칭도 비슷하고 작업 형태도 비슷하기 때문에 혼동할 수 있습니다. UI 화면 설계는 실제로 사용할 UI를 설계하는 것이고, UI 설계서는 실제 사용할 UI 화면을 설계하기 전에 사용자의 요구사항을 가시화하고 검증하기 위해 작성하는 것입니다. 이런 차이점을 염두에 두고 UI 설계서를 구성하는 요소에는 어떤 것들이 있는지 기억해 두세요.

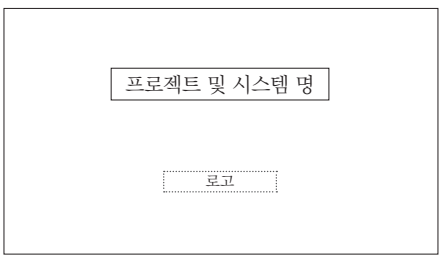
1 UI 설계서의 개요

UI 설계서는 사용자의 요구사항을 바탕으로 UI 설계를 구체화하여 작성하는 문서로, 상세 설계 전에 대표적인 화면들을 설계한다.

- UI 설계서는 기획자, 개발자, 디자이너 등과의 원활한 의사소통을 위해 작성한다.
- UI 설계서는 UI 설계서 표지, UI 설계서 개정 이력, UI 요구사항 정의서, 시스템 구조, 사이트 맵, 프로세스 정의서, 화면 설계 순으로 작성한다.

2 UI 설계서 표지 작성

UI 설계서 표지는 다른 문서와 혼동되지 않도록 프로젝트명 또는 시스템명을 포함시켜 작성한다.



UI 설계서 : 표지 예

3 UI 설계서 개정 이력 작성

UI 설계서 개정 이력은 UI 설계서가 수정될 때마다 어떤 부분이 어떻게 수정되었는지를 정리해 놓은 문서이다.

- 처음 작성 시 첫 번째 항목을 '초안 작성', 버전(Version)을 1.0으로 설정한다.
- UI 설계서에 변경 사항이 있을 때마다 변경 내용을 적고 버전을 0.1씩 높인다.

NO	내용	Version	수정일	작성자
1	초안 작성	V1.0	2018-10-03	김유나
2	보완	V1.1	2018-10-09	윤석호
3	2018-10-10 회의 내용 반영	V1.2	2018-10-11	윤석호
4	2018-10-15 회의 내용 반영	V1.3	2018-10-17	김유나
5	2018-10-21 회의 내용 반영	V1.4	2018-10-22	윤석호
6	2018-10-29 회의 내용 반영	V1.5	2018-10-30	김유나

UI 설계서 : 개정 이력 사례

4 UI 요구사항 정의서 작성

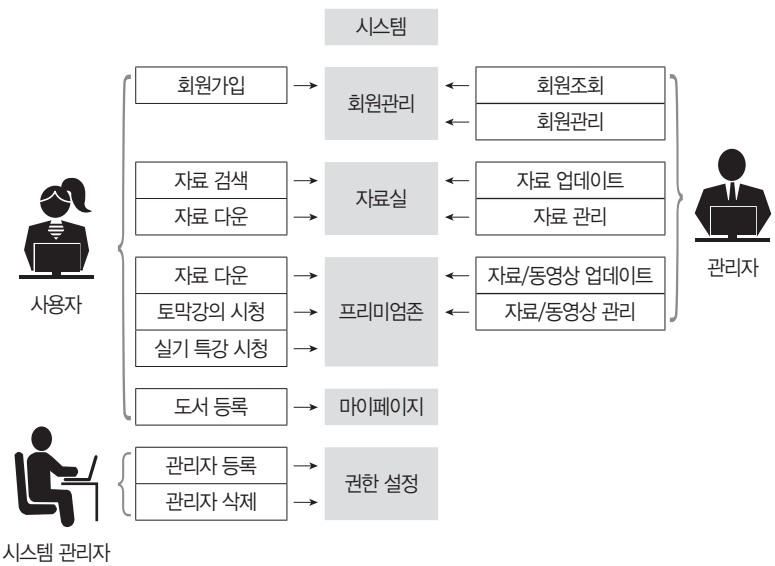
UI 요구사항 정의서는 사용자의 요구사항을 확인하고 정리한 문서로, 사용자 요구사항의 UI 적용 여부를 요구사항별로 표시한다.

No	RFP(Request For Proposal)	확정 여부	비고
1	요구사항1 – 화면에 표현될 기능	확정	화면 설계 적용
2	요구사항2 – 화면 구성 요소	확정	화면 설계 적용
3	요구사항3 – 추가적으로 필요한 구성 요소	확정	화면 설계 적용(1안)
4	요구사항4 – 기능을 표현하기 위한 페이지	확정	화면 설계 적용
5	요구사항5 – 각 화면 간 이동	확정	2018-10-15 회의 반영
6	요구사항6 – 공고 및 이벤트 협업 창	확정	2018-10-30 협의 결정

UI 설계서 : 요구사항 정의서(RFP) 사례

5 시스템 구조 작성

시스템 구조는 UI 요구사항과 UI 프로토타입에 기초하여 전체 시스템의 구조를 설계한 것으로 사용자의 요구사항이 어떻게 시스템에 적용되는지 알 수 있다.



UI 설계서 : UI 시스템 구조 예시

6 사이트 맵(Site Map) 작성

사이트 맵은 시스템 구조를 바탕으로 사이트에 표시할 콘텐츠를 한 눈에 알아 볼 수 있도록 메뉴별로 구분하여 설계한 것이다.

- 사이트 맵을 작성한 후 사이트 맵의 상세 내용(Site Map Detail)을 표 형태로 작성한다.

시나공IT

시험정보/후기
시험정보 합격전략/후기

자료실
실습예제 학습자료 기출문제 정답지 정오표

프리미엄존
시험대비자료 토막강의 실기특강

묻고답하기

고객센터
공지사항 시험공지 이벤트 오류 제보 FAQ

마이길벗
등록도서 나의 동영상 강좌 나의 문의/문답 회원정보관리

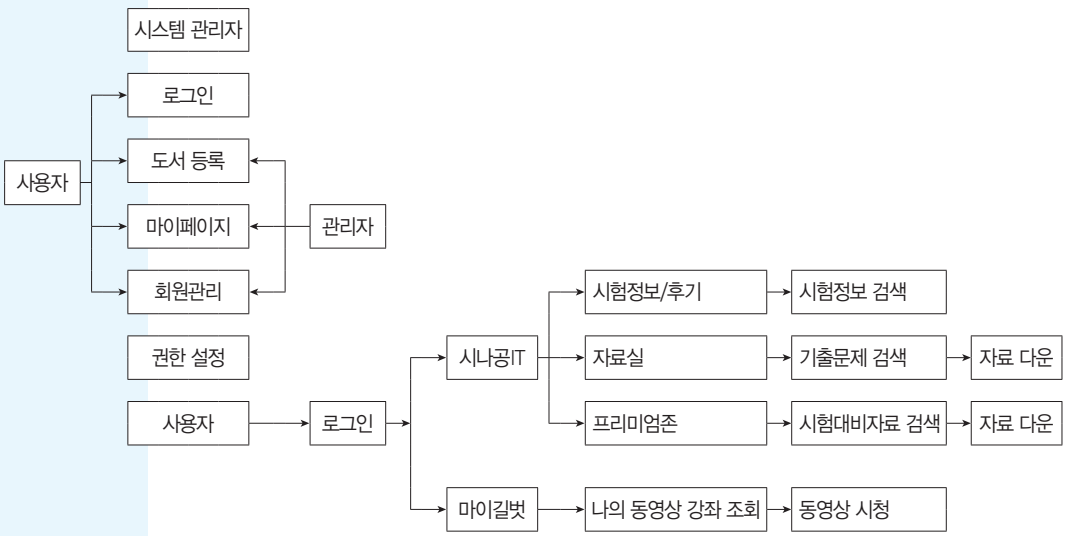
UI 설계서 : 사이트 맵 구조 예시

메뉴1	메뉴2	Description
시나공IT	시험정보/후기	시험정보
		합격전략/후기
	자료실	실습예제
		학습예제
		기출문제
		정답지
		정오표
	프리미엄존	시험대비자료
		토막강의
		실기특강

사이트 맵 상세 내용(Site Map Detail) 사례

7 프로세스(Process) 정의서 작성

프로세스 정의서는 사용자 관점에서 사용자가 요구하는 프로세스들을 작업 진행 순서에 맞춰 정리한 것으로 UI의 전체적인 흐름을 파악할 수 있다.



UI 설계서 : 프로세스 정의서 예시

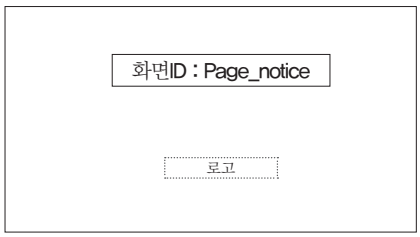
8 화면 설계

화면 설계는 UI 프로토타입과 UI 프로세스를 참고하여 필요한 화면을 페이지별로 설계한 것이다.

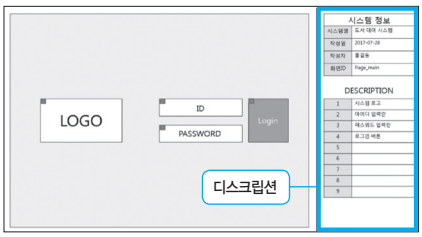
- 화면을 구분하기 위해 화면별 고유ID를 부여하고 별도 표지를 작성한다.

❶ 대표적인 화면들에 대해 포함될 정보, 인터페이스 요소, 레이아웃 등이 표현된 와이어프레임*을 대략적으로 스케치한다.

❷ 주요 흐름을 스토리보드* 형태로 작성한다. 디스크립션에는 시스템 정보, 인터랙션, 로직, 정책 등 디자인하거나 설계할 때 필요한 사항을 기록한다.



UI 설계서 : 공지사항 페이지 타이틀 예시

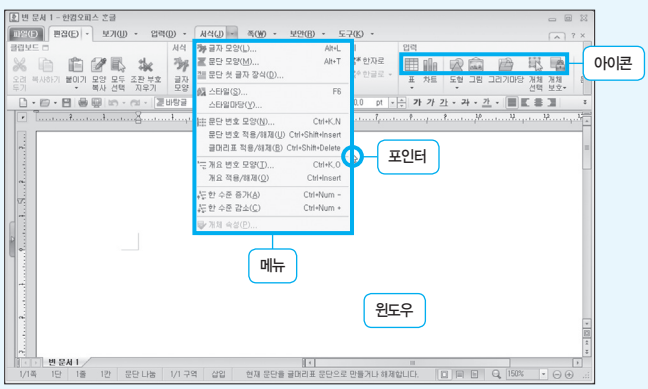


UI 설계서 : 로그인 페이지 화면 설계 예시

잠깐만요

UI 화면 설계의 기본 구성 요소

윈도우(Window)	키보드나 마우스 등을 통해 데이터 입력 및 결과를 보여주는 화면상의 표시 영역입니다.
메뉴(Menu)	• 화면에서 수행할 기능들을 일정한 형태로 모아놓은 인터페이스입니다. • 사용자로 하여금 기능 선택을 수월하게 합니다.
아이콘(Icon)	• 수행하고자 하는 동작, 동작의 대상 등을 조그마한 그림 형태로 표현한 인터페이스입니다. • 동일한 사용 환경 안에서는 아이콘의 크기는 동일하거나 규칙적인 크기 안에서 제공해야 합니다.
포인터(Pointer)	• 입력이 이뤄지는 지점을 알려주는 화면상의 커서입니다. • 시스템의 상태를 포인터의 모양으로도 표시합니다.



와이어프레임(Wire Frame)
와이어프레임은 기획 단계의 초기에 제작하는 것으로, 페이지에 대한 개략적인 레이아웃이나 UI 요소 등에 대한 뼈대를 설계하는 단계입니다.

스토리보드(Story Board)
스토리보드는 와이어프레임에 콘텐츠에 대한 설명, 페이지 간 이동 흐름 등을 추가한 문서입니다.

출제예상

1. UI 설계서는 초안을 작성한 후 내용을 수정할 때마다 개정 이력을 정리해야 한다. 이에 대한 설명으로 틀린 것은?

① UI 설계서가 수정될 때마다 어떤 부분이 어떻게 수정되었는지 명시해야 한다.
② 처음 작성 시에는 '초안 작성'으로 기입하고 버전(version)을 1.0으로 설정한다.
③ 버전은 설계서를 수정 또는 보완할 때마다 1씩 더한다.
④ 개정 이력은 UI 설계서에 반드시 포함되어야 한다.

초안 문서의 버전을 1.0과 같이 소수점 이하를 표시한데는 이유가 있겠죠.

출제예상

2. 다음 내용이 설명하는 것은 무엇인가?

- 화면의 정보를 한눈에 파악하기 위한 시각적인 콘텐츠 모형을 말한다.
- 일반적으로 테이블 형태로 되어 있고, 위에서부터 아래로 내려가며 정보를 찾을 수 있는 계층형으로 되어 있는 것이 보통이다.

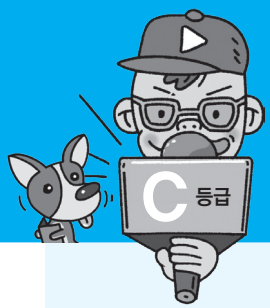
① 스토리보드(Story Board)
② 사이트 맵(Site Map)
③ 레이아웃(Layout)
④ 내비게이션(Navigation)

사이트의 정보를 시각적으로 표현한 것! 무엇일까요?

▶ 정답 : 1. ③ 2. ②

SECTION
017
LIVE

유용성 평가



1 UI의 유용성 평가

- 유용성(Usability)은 사용자가 시스템을 통해 원하는 목표를 얼마나 효과적으로 달성할 수 있는가에 대한 척도로, UI의 주된 목적은 유용성이 뛰어난 UI를 제작하는 것이다.
- 유용성 평가는 사용자 측면에서 복잡한 시스템을 얼마나 편리하게 사용할 수 있는지를 평가하는 것으로, 시스템의 문제점을 찾아내고 개선 방향을 제시하기 위한 조사 과정이다.
- 유용한 UI를 설계하기 위해서는 UI의 구조, 기능, 가치 등에 대해 사용자가 생각하는 사용자 모형과 시스템 설계자가 만들려고 하는 개발자 모형 간의 차이를 최소화해야 한다.
- 사용자 모형과 개발자 모형 간의 차이가 발생하는 원인
  - 실행 차 : 사용자가 원하는 목적과 실행 기능이 다르기 때문에 발생한다.
  - 평가 차 : 사용자가 원하는 목적과 실행 결과가 다르기 때문에 발생한다.

2 실행 차를 줄이기 위한 UI 설계 원리 검토

1. 사용 의도 파악

사용자의 목적을 명확히 파악한 후 불필요한 기능이나 중복되는 기능이 있는지 확인한다.

2. 행위 순서 규정

- 사용자가 특정 기능을 사용하기 위한 행위 순서를 세분화시켜 순서대로 제시하고, 사용자가 임의로 행위 순서를 변경할 수 있도록 한다.
- 특정 작업을 수행하기 위한 단계는 최소화하고, 다양한 방법을 통해 수행할 수 있도록 설계하며, 사용자의 기존 경험에 비추어 가능한 한 친숙하도록 설계한다.

3. 행위의 순서대로 실행

- 프로세스의 흐름을 직접적으로 파악할 수 있도록 제공함으로써 사용자가 행위 순서대로 실행할 때 어려움이 없어야 한다.
- 작업이 원활하게 진행되도록 과도한 상호 작용은 피한다.
- 사용자가 의도한 행위를 효율적으로 실행할 수 있도록 피드백, 취소 기능, 디폴트 값 등을 적절하게 설정한다.

전문가의 조언

좋은 UI는 유용성이 뛰어난 UI이고, 유용성이 뛰어난 UI를 설계하려면 사용자와 개발자가 생각하는 완성된 모형의 차이가 작아야 합니다. 유용성의 개념과 함께 사용자와 개발자가 생각하는 완성된 모형 간의 차이를 줄이기 위한 UI의 설계 원리를 파악해 두세요.

3 평가 차를 줄이기 위한 UI 설계 원리 검토

1. 수행한 키 조작의 결과를 사용자가 빠르게 지각하도록 유도

사용자가 수행한 행위에 대해 최대한 빨리 반응하도록 설계하고, 사용자가 수행한 행위로 인해 현재 시스템의 변화를 직접적으로 파악 할 수 있도록 피드백해야 한다.

2. 키 조작으로 변화된 시스템의 상태를 사용자가 쉽게 인지하도록 유도

시스템의 상태 정보를 가능한 한 단순하고 이해하기 쉽게 제시해야 한다.

3. 사용자가 가진 원래 의도와 시스템 결과 간의 유사 정도를 사용자가 쉽게 파악하도록 유도

사용자의 의도가 시스템을 통해 충족되었는지, 충족될 수 있는지를 사용자가 쉽게 파악할 수 있도록 설계해야 한다.

기출문제 따라잡기 Section 017

출제예상

1. UI를 설계하다 보면 개발자가 설계한 UI가 사용자가 생각한 것과 다르게 실행되는 경우 있다. 이런 일을 예방하기 위한 방법으로 틀린 것은?

① 개발자는 사용자의 목적을 명확히 파악한 후 불필요한 기능이나 중복되는 기능이 있는지 확인한다.

② 개발자는 특정 기능을 사용하기 위한 행위 순서를 세분화시킨 뒤 순서대로 제시해야 한다.

③ 특정 작업을 수행하기 위한 단계는 최소화시켜야 하고, 혼동되지 않게 한 가지의 방법만을 제공해야 한다.

④ 특정 기능을 수행하는 순서는 사용자의 기존 경험에 비추어 가능한 한 친숙하게 설계한다.

특정 작업을 수행하는 방법이 하나일 경우와 여러 개일 경우 어느 것이 사용하기 편리할까요?

출제예상

2. 다음 중 사용자가 원하는 목적과 UI의 실행 결과가 최대한 비슷하게 UI를 설계하려고 할 때의 방법으로 틀린 것은?

① 사용자가 특정 작업을 수행하면 최대한 빨리 반응하도록 설계한다.

② 특정 작업으로 인한 현재 시스템의 변화는 다른 작업에 방해되지 않도록 간접적으로 표시해야 한다.

③ 키 조작으로 변화된 시스템의 상태 정보를 가능한 한 단순하고 이해하기 쉽게 제시해야 한다.

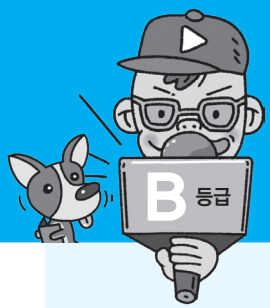
④ 사용자의 의도가 시스템을 통해 충족되었는지 사용자가 쉽게 파악할 수 있도록 설계해야 한다.

현재 시스템의 변화는 사용자가 직관적으로 알 수 있도록 표시해야 합니다.

▶ 정답 : 1. ③ 2. ②

SECTION
018
LIVE

UI 상세 설계



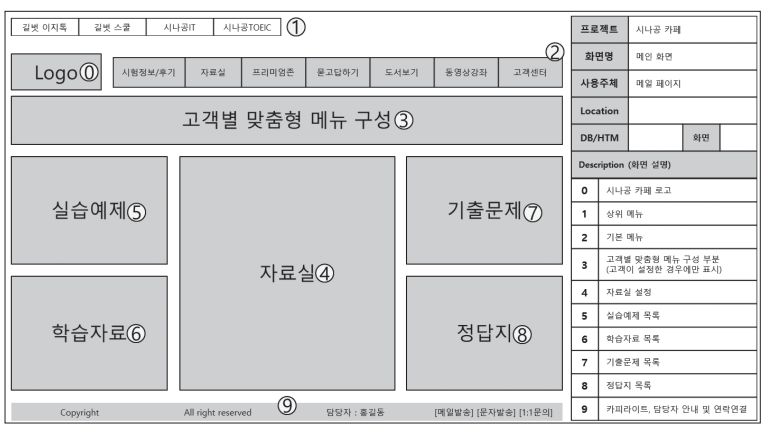
1 UI 시나리오 문서 개요

UI 상세 설계는 UI 설계서*를 바탕으로 실제 설계 및 구현을 위해 모든 화면에 대한 자세한 설계를 진행하는 단계로, UI 상세 설계를 할 때는 반드시 시나리오를 작성해야 한다.

- UI 시나리오 문서는 사용자 인터페이스의 기능 구조, 대표 화면, 화면 간 인터랙션*의 흐름, 다양한 상황에서의 예외 처리 등을 문서로 정리한 것이다.
- UI 시나리오 문서에는 사용자가 최종 목표를 달성하기 위한 방법이 순차적으로 묘사되어 있다.
- UI 설계자 또는 인터랙션 디자이너*가 UI 시나리오 문서를 작성하면 그래픽 디자이너가 시나리오를 바탕으로 디자인을 하고 개발자가 UI를 구현한다.

2 UI 시나리오 문서 작성 원칙

- 개발자가 전체적인 UI의 기능과 작동 방식을 한눈에 이해할 수 있도록 구체적으로 작성한다. 보통 계층(Tree) 구조 또는 플로차트(Flow Chart) 표기법으로 작성한다.
- 모든 기능에 공통적으로 적용될 UI 요소와 인터랙션을 일반 규칙으로 정의한다.
- 대표 화면의 레이아웃과 그 화면에 속할 기능을 정의한다.
- 인터랙션의 흐름을 정의하며, 화면 간 인터랙션의 순서(Sequence), 분기(Branch), 조건(Condition), 루프(Loop) 등을 명시한다.
- 예외 상황에 대비한 다양한 케이스를 정의한다.
- UI 일반 규칙을 지키면서 기능별 상세 기능 시나리오를 정의한다.
- UI 시나리오 규칙을 지정한다.



UI 시나리오 작성 예

전문가의 조언

UI 상세 설계는 흐름 설계에서 작성한 UI 설계서를 토대로 실제 구현을 위한 전체 화면의 세부적인 설계를 하는 부분입니다. UI 상세 설계에서 가장 먼저 하는 작업은 화면 간의 흐름, 다양한 상황에서의 예외 처리 등을 시나리오 문서로 작성하는 것입니다. UI 시나리오 문서를 작성하는 원칙이나 작성 요건 등을 정리해 두세요.

UI 설계서

UI 설계서는 UI 흐름 설계와 UI 상세 설계에서 모두 작성합니다. UI 흐름 설계에서 UI 설계서의 기본적인 토대를 작성한다면 UI 상세 설계에서는 흐름 설계에서 작성한 UI 설계서를 다시 한번 확인하고 추가 또는 수정하여 완성합니다.

인터랙션(Interaction)

사용자와 시스템을 연결하는 것이 UI라면 인터랙션은 UI를 통해 시스템을 사용하는 일련의 상호작용입니다. 쉽게 말해 마우스로 화면의 어떤 아이콘을 클릭하면 화면이 그에 맞게 반응하는 것을 말합니다.

인터랙션 디자이너

인터랙션 디자이너는 제품, 시스템, 서비스에 대한 사용자의 행동과 그에 반응하는 절차를 디자인하는 사람입니다.

전문가의 조언

UI 시나리오의 작성 원칙, 규칙, 요건 등은 정보통신산업진흥원 부설 SW공학센터의 '소프트웨어 개발 UI/UX 참조모델 가이드'(2014)에서 설명한 내용을 토대로 구성하였습니다.

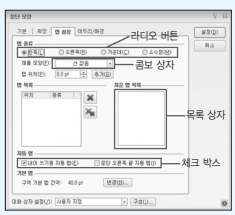
3 UI 시나리오 문서 작성을 위한 일반 규칙

UI 시나리오 문서를 작성하면서 적용할 일반적인 규칙은 다음과 같다.

구분	설명
주요 키의 위치와 기능	모든 화면에 공통적으로 배치되는 주요 키의 위치와 기능을 설명한 것으로, 여러 화면 간의 일관성을 보장한다.
공통 UI 요소	체크 박스, 라디오 버튼, 텍스트 박스 등의 UI 요소*를 언제, 어떤 형태로 사용할지를 정의하고, 사용자가 조작하면 어떻게 반응하는지 그 흐름을 설명한다.
기본 스크린 레이아웃 (Basic Screen Layouts)	모든 화면에 공통적으로 나타나는 Titles, Ok/Back, Soft Key, Option, Functional Buttons 등의 위치와 속성을 정의한다.
기본 인터랙션 규칙 (Basic Interaction Rules)	터치 제스처 등에 공통적으로 사용되는 조작 방법과 실행, 이전, 다음, 삭제, 이동 등의 화면 전환 효과 등을 기술한다.
공통 단위 태스크 흐름 (Task Flows)	많은 기능들에 공통적으로 사용되는 삭제, 검색, 매너 모드 상태 등에 대한 인터랙션 흐름을 설명한다.
케이스 문서	다양한 상황에서 공통적으로 적용되는 시스템의 동작을 정의한 문서이다. 예 사운드, 조명, 이벤트 케이스 등

UI 요소

- **체크 박스(Check Box)** : 여러 개의 선택 상황에서 1개 이상의 값을 선택할 수 있는 버튼
- **라디오 버튼(Radio Button)** : 여러 항목 중 하나만 선택할 수 있는 버튼
- **텍스트 박스(Text Box)** : 사용자가 데이터를 입력하고 수정할 수 있는 상자
- **콤보 상자(Combo Box)** : 이미 지정된 목록 상자에 내용을 표시하여 선택하거나 새로 입력할 수 있는 상자
- **목록 상자(List Box)** : 콤보 상자와 같이 목록을 표시하지만 새로운 내용을 입력할 수 없는 상자



4 UI 시나리오 문서의 요건

완전성(Complete)	• 누락되지 않도록 최대한 상세하게 기술해야 한다. • 해당 시스템의 기능보다는 사용자의 태스크에 초점을 맞춰 기술한다.
일관성(Consistent)	서비스 목표, 시스템 및 사용자의 요구사항, UI 스타일 등이 모두 일관성을 유지해야 한다.
이해성(Understandable)	• 누구나 쉽게 이해할 수 있도록 설명한다. • 불분명하거나 추상적인 표현은 피한다.
가독성(Readable)	• 표준화된 템플릿* 등을 활용하여 문서를 쉽게 읽을 수 있도록 해야 한다. • v1.0, v2.0 등과 같이 문서 인덱스에 대한 규칙이나 목차를 제공한다. • 읽기 쉽도록 줄 간격, 단락, 들여쓰기 등의 기준을 마련한다. • 시각적인 효과를 위해 여백이나 빈 페이지, 하이라이팅을 일관성 있게 지정한다. • 하이퍼링크 등을 지정하여 문서들이 서로 참조될 수 있도록 지정한다.
수정 용이성(Modifiable)	시나리오의 수정이나 개선이 쉬어야 한다.
추적 용이성(Traceable)	변경 사항은 언제, 어떤 부분이, 왜 발생했는지 쉽게 추적할 수 있어야 한다.

템플릿(Template)
템플릿은 형판, 형틀이라는 뜻으로, 화면의 기본 레이아웃 형태를 의미합니다.

5 UI 시나리오 문서로 인한 기대 효과

- 요구사항이나 의사소통에 대한 오류가 감소한다.
- 개발 과정에서의 재작업이 감소하고, 혼선이 최소화된다.
- 불필요한 기능을 최소화한다.
- 소프트웨어 개발 비용을 절감한다.
- 개발 속도를 향상시킨다.

기출문제 따라잡기 Section 018

출제예상
1. 다음 중 사용자 인터페이스의 시나리오 문서에 포함되는 내용이 아닌 것은?

① 다양한 상황에서의 예외 처리 방식
② 대표 화면 간 인터랙션 흐름
③ GUI
④ 기능 구조

UI 시나리오 문서가 완성되면 이것을 토대로 GUI를 설계합니다.

출제예상
2. 완성된 UI 시나리오 문서를 가지고 다음 작업을 진행하는 담당자가 아닌 것은?

① 인터랙션 디자이너 ② 개발자
③ 품질 관리자 ④ GUI 디자이너

인터랙션 디자이너는 UI 시나리오 문서를 작성하는 사람입니다.

출제예상
3. 다음 중 형판, 형틀이라는 뜻으로 화면의 기본적인 레이아웃 형태를 의미하는 것은?

① 텍스트(Text) ② 인터페이스(Interface)
③ 프레임(Frame) ④ 템플릿(Template)

'기본적인 레이아웃 형태' 하면 '템플릿'! 기억해 두세요.

출제예상
4. 다음 중 UI 시나리오 문서에 대한 설명으로 틀린 것은?

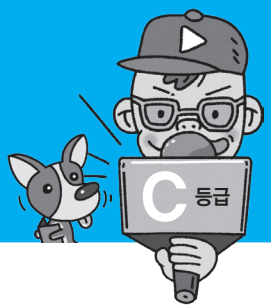
① 해당 시스템의 기능에 초점을 맞춰 작성한다.
② 시각적인 효과를 위해 여백이나 빈 페이지, 하이라이팅을 일관성 있게 지정한다.
③ 시나리오의 수정 또는 개선이 쉬어야 한다.
④ 변경 사항이 있을 경우 언제, 어떤 부분이, 왜 발생했는지 쉽게 추적할 수 있어야 한다.

UI 시나리오 문서는 해당 시스템의 기능보다는 사용자의 태스크에 초점을 맞춰 작성해야 합니다.

▶ 정답 : 1. ③ 2. ① 3. ④ 4. ①

SECTION 019 ● LIVE

HCI / UX / 감성공학



전문가의 조언

UX가 제품을 사용하고 느낀 총제적인 경험이라면 감성공학은 이런 경험을 통해 얻은 복합적인 감각을 의미합니다. UI를 기준으로 HCI, UX, 감성공학의 개념이 구분되도록 알아두세요.

1 HCI(Human Computer Interaction or Interface)

HCI는 사람이 시스템을 보다 편리하고 안전하게 사용할 수 있도록 연구하고 개발하는 학문으로, 최종 목표는 시스템을 사용하는데 있어 최적의 사용자 경험(UX)을 만드는 것이다.

- 원래 HCI는 사람과 컴퓨터의 상호작용을 연구해서 사람이 컴퓨터를 편리하게 사용하도록 만드는 학문이었으나, 대상이 컴퓨터뿐만 아니라 서비스, 디지털 콘텐츠 등으로, 사람도 개인뿐만 아니라 사회나 집단으로 확대되었다.
- HCI는 어떤 제품이 좋은 제품인지, 어떻게 하면 좋은 제품을 만들 수 있는지 등을 연구한다.

2 UX(User Experience)

UX는 사용자가 시스템이나 서비스를 이용하면서 느끼고 생각하게 되는 총체적인 경험을 말한다. 단순히 기능이나 절차상의 만족뿐만 아니라 사용자가 참여, 사용, 관찰하고, 상호 교감을 통해서 알 수 있는 가치 있는 경험을 말한다.

- UX는 기술을 효용성 측면에서만 보는 것이 아니라 사용자의 삶의 질을 향상시키는 하나의 방향으로 보는 새로운 개념이다.
- UI가 사용성, 접근성, 편의성을 중시한다면 UX는 이러한 UI를 통해 사용자가 느끼는 만족이나 감정을 중시한다.
- UX의 특징
  - 주관성(Subjectivity) : 사람들의 개인적, 신체적, 인지적 특성에 따라 다르므로 주관적이다.
  - 정황성(Contextuality) : 경험이 일어나는 상황 또는 주변 환경에 영향을 받는다.
  - 총체성(Holistic) : 개인이 느끼는 총체적인 심리적, 감성적인 결과이다.

감성

여기서의 '감성'은 사용자가 제품을 사용한 경험을 통해 얻은 복합적인 감각을 의미합니다.

3 감성*공학

감성공학은 제품이나 작업환경을 사용자의 감성에 알맞도록 설계 및 제작하는 기술로, 인문사회과학, 공학, 의학 등 여러 분야의 학문이 공존하는 종합과학이다.

- '감성'을 과학적으로 측정하기 위해서는 생체계측 기술, 감각계측 기술, 센서, 인공지능, 생체제어 기술 등이 요구된다.
- 감성공학의 목적은 인간의 삶을 편리하고 안전하며 쾌적하게 만드는 것이다.
- 감성공학은 인간의 감성을 구체적으로 제품 설계에 적용하기 위해 공학적인 접근방법을 사용한다.

- 감성공학은 인간의 신체적, 정신적 특성을 배려한 제품 설계에서 더 나아가 인간의 감성까지 고려한다.
- 감성공학은 인간과 컴퓨터의 상호작용을 나타내는 HCI(Human Computer Interaction or Interface) 설계에 인간의 특성과 감성을 반영하였다.
- 감성공학의 요소기술
  - 기반 기술 : 제품 설계에 적용할 인간의 특성을 파악한다.
  - 구현 기술 : 인간의 특성에 맞는 인터페이스를 구현한다.
  - 응용 기술 : 인간에 맞는지 파악하여 새로운 감성을 만든다.

기출문제 따라잡기 Section 019

출제예상

1. 감성공학을 디자인의 기능적인 측면에서 봤을 경우 해당하는 기능은?

① 물리적 기능
② 생리적 기능
③ 심리적 기능
④ 사회적 기능

상식적인 수준에서 생각해도 맞힐 수 있는 문제네요.

출제예상

2. 다음 설명에 가장 근접한 것은 무엇인가?

> 사람과 컴퓨터 시스템 간의 상호작용을 연구하고 설계하는 것으로, 사람이 컴퓨터를 편리하게 사용할 수 있도록 한다.

① UX ② UI
③ HCI ④ IA

보기에 제시된 단어들이 무엇의 약자인지 생각해 보세요.

▶ 정답 : 1. ③ 2. ③

예·상·문·제·은·행

1. 사용자 인터페이스 설계 시 고려 사항으로 가장 거리가 먼 것은?

① 가시성 ② 다양성
③ 단순성 ④ 일관성

2. 사용자 인터페이스를 설정할 때의 고려 사항이 아닌 것은?

① 누가 보더라도 이해할 수 있도록 직관성을 제공한다.
② 사용자의 목적을 정확하게 달성할 수 있도록 유효성을 제공한다.
③ 사용자의 요구사항을 최대한 수용하고 오류를 최소화할 수 있도록 유연성을 제공한다.
④ 버튼이나 조작 방법 등을 사용자가 쉽게 습득할 수 있도록 다양성을 제공한다.

3. 다음 중 내비게이션에 대한 설명으로 틀린 것은?

① 웹 콘텐츠를 분류하고 체계화시킨 후 이들을 연결시켜 방문자로 하여금 웹사이트를 이용할 수 있게 하는 체계를 말한다.
② 아이콘이나 그래픽 등을 일관성 있게 사용하여 홈페이지를 사용하는 사용자가 어디서라도 필요한 정보를 쉽게 찾을 수 있도록 하는 것이다.
③ 원하는 정보를 쉽게 찾을 수 있도록 시스템 위주로 설계되어야 한다.
④ 사이트의 이동 경로나 방법, 이동을 돕는 구조와 인터페이스 등을 모두 포함하는 개념이다.

4. 다음 중 웹 콘텐츠 접근성을 높이기 위한 고려 사항이 아닌 것은?

① 모든 콘텐츠는 사용자가 그 내용을 이해할 수 있도록 비선형 구조로 작성되어야 한다.
② 사용자가 맥락을 이해할 수 있도록 논리적인 순서를 제공해야 한다.
③ 사용자가 의도하지 않은 기능은 실행되지 않도록 해야 한다.
④ 사용자의 실수로 오류가 발생할 경우 이를 정정할 수 있는 방법을 제공해야 한다.

5. 다음 중 정부기관의 홈페이지 구축 시 반영해야 할 지침으로 가장 옳지 않은 것은?

① 스크립트 비사용자를 위한 대체 정보를 제공해야 한다.
② 호환성이 보장되지 않는 플러그인의 사용은 지양한다.
③ 웹사이트에서 제공하는 미디어는 특정 운영체제에 종속적인 포맷을 사용해야 한다.
④ 다양한 인터페이스로 웹 사이트를 이용할 수 있어야 한다.

6. 다음 중 웹사이트의 내비게이션 요소가 아닌 것은?

① 디렉터리
② 메뉴
③ 템플릿
④ 사이트 맵

7. 다음은 중 화면 단위로 삽입될 구성 요소나 콘텐츠, 기능 등을 구체적으로 정리해 놓은 문서는 무엇인가?

① 레이아웃 ② 내비게이션
③ 스토리보드 ④ 동영상

8. 다음 중 사용자 인터페이스의 설계 도구에 대한 설명으로 틀린 것은?

① 목업은 실제 화면과 유사한 동적인 형태의 모형이다.
② 와이어프레임을 통해 정보의 시각적인 구조를 이해할 수 있다면 프로토타입을 통해 인터페이스의 상호 작용을 시뮬레이션 할 수 있다.
③ 프로토타입에는 페이퍼 프로토타입과 디지털 프로토타입이 있다.
④ 와이어프레임은 기획 초기 단계에 사용된다.

9. 사용자 인터페이스(UI)는 사용자와 시스템 간의 상호 작용이 원활하게 수행되도록 도와주는 연계 작업이다. 다음 중 사용자 인터페이스의 설계 툴이 아닌 것은?

① 스케치
② 프로토나우
③ 아파치
④ 발사믹 목업

10. 사용자의 요구사항을 조사하기 위한 인터뷰 진행 시 유의사항으로 틀린 것은?

① 인터뷰는 실사용자들을 대상으로 진행해야 한다.
② 사용자들의 요구사항이나 불편사항 등을 파악하기 위해 사용자 리서치를 먼저 수행한 후 사용자 인터뷰를 진행한다.
③ 가능한 많은 사람을 인터뷰하여 다양한 의견을 수렴하되, 다수의 의견으로 인해 개인의 중요한 의견을 놓치지 않도록 주의해야 한다.
④ 인터뷰를 진행한 후 사용자들의 의견이 수렴된 비즈니스 요구사항을 정의해야 한다.

해설은 96쪽에 있습니다.

11. 사용자 인터페이스 설계 시 주의할 사항이 아닌 것은?

① 컴퓨터의 사용 환경을 고려하여 설계한다.
② 오류 발생 시 사용자가 쉽게 인지할 수 있도록 설계한다.
③ 초보자가 쉽게 사용할 수 있도록 CLI 방식으로 설계한다.
④ 실사용자에 대한 이해를 바탕으로 설계한다.

12. ISO/IEC 9126은 국제적으로 소프트웨어 제품의 품질을 측정하는데 이용되고 있다. 다음 중 ISO/IEC 9126에서 정의한 소프트웨어의 품질 특성이 아닌 것은?

① 유지 보수성(Maintainability)
② 사용성(Usability)
③ 기능성(Functionality)
④ 가시성(Visibility)

13. 다음의 사용자 인터뷰는 ISO/IEC 9126의 품질 특성 중 어떤 특성에 가장 적합한가?

> 컴퓨터가 다운되면 20초 내에 정상적으로 작동되도록 해주시고, 제가 자리를 비울 경우 다른 사람이 이 컴퓨터를 사용할 수 없게 해주세요.

① 효율성, 신뢰성
② 기능성, 신뢰성
③ 기능성, 유지보수성
④ 사용성, 이식성

14. ISO/IEC 9126에 규정된 품질의 주 특성과 부 특성 간의 연결이 잘못된 것은?

① Maintainability – Textability
② Usability – Stability
③ Functionality – Accuracy
④ Reliability – Fault Tolerance

15. UI 프로토타입에 대한 설명으로 가장 옳지 않은 것은?

① 실제 구현된 것처럼 만든 동적인 형태의 모형이다.
② 프로토타입을 작성하면 사용자를 설득하고 이해시키기 쉽다.
③ 프로토타입은 사용자의 요구사항이 모두 반영될 때까지 계속하여 개선하고 보완해야 한다.
④ 프로토타입은 될 수 있으면 정교하게 만들어야 한다.

16. 사용자를 이해시키기 위해 UI 프로토타입을 작성하려고 한다. 다음 설명 중 틀린 것은?

① 개발 툴, 테스트 툴, 빌드 및 배포 툴, 형상 관리 등 프로토타입 개발에 필요한 환경을 마련해야 한다.
② 프로토타입을 작성하면서 가장 많은 시간이 소요된 구간을 찾고 그 해결 방법을 실제 프로젝트에 적용하면 많은 시간을 절약할 수 있다.
③ 프로토타입으로 검증할 범위는 많은 내용을 포함할 수 있도록 가급적 넓게 잡는 것이 좋다.
④ 프로토타입을 시연할 때는 프로토타입의 개발 목적을 구체적으로 설명해야 한다.

17. 다음 중 UI 설계서에 대한 설명으로 틀린 것은?

① UI 설계서는 사용자의 요구사항을 바탕으로 UI 설계를 구체화한 문서이다.
② UI 설계서는 상세 설계 이후에 작성한다.
③ UI 설계서는 개정 이력, 요구사항 정의서, 시스템 구조, 사이트 맵, 프로세스 정의서, 화면 설계 등으로 구성된다.
④ 사이트 맵(Site Map)을 통해 웹 사이트의 전체 구조를 한 눈에 알아볼 수 있다.

▶ **정답 :** 1. ② 2. ④ 3. ③ 4. ① 5. ③ 6. ③ 7. ③ 8. ① 9. ③ 10. ② 11. ③ 12. ④ 13. ② 14. ② 15. ④ 16. ③ 17. ②

예·상·문·제·은·행·해·설

1. Section 010

사용자 인터페이스는 다양성이 아닌 표준화를 고려해야 한다.

2. Section 010

사용자가 조작 방법 등을 쉽게 습득하게 하려면 다양성을 제공하는 것이 아니라, 기능 구조와 디자인을 표준화하여 한 번 학습한 이후에는 쉽게 사용할 수 있어야 한다.

3. Section 011

내비게이션은 시스템이 아니라 사용자 중심으로 설계되어야 한다.

4. Section 011

모든 콘텐츠는 비선형이 아니라 선형 구조로 작성되어야 한다.

5. Section 011

홈페이지에서 제공하는 미디어는 운영체제의 종류에 상관없이 사용할 수 있도록 특정 운영체제에 종속적인 포맷은 최소화하여 사용해야 한다.

6. Section 011

템플릿은 '형판, 보기 판'이라는 의미로 일정한 모양으로 만들어진 틀을 의미한다. 템플릿은 웹사이트의 내비게이션 요소가 아니다.

8. Section 012

목업은 실제 화면과 유사하지만 실제로는 구현되지 않은 정적인 형태의 도형이다.

9. Section 012

아파치(Apache)는 UI 설계 도구가 아니라 서버 프로그램이다.

10. Section 013

사용자 인터뷰를 진행한 후 파악된 요구사항을 토대로 사용자 리서치를 계획하고 진행해야 한다.

11. Section 010

초보자가 쉽게 사용할 수 있도록 설계하려면 GUI(Graphical User Interface) 방식으로 설계해야 한다.

- CLI(Command Line Interface) : 명령과 출력이 텍스트 형태로 이뤄지는 인터페이스
- GUI(Graphical User Interface) : 아이콘이나 메뉴를 마우스로 선택하여 작업을 수행하는 그래픽 환경의 인터페이스

12. Section 014

ISO/IEC 9126의 품질 특성

기능성(Functionality), 신뢰성(Reliability), 사용성(Usability), 효율성(Efficiency), 유지 보수성(Maintainability), 이식성(Portability)

13. Section 014

- 기능성(Functionality) : 소프트웨어가 사용자의 요구사항을 정확하게 만족하는 기능을 제공하는지 여부를 나타냄
- 신뢰성(Reliability) : 소프트웨어가 요구된 기능을 정확하고 일관되게 오류 없이 수행할 수 있는 정도를 나타냄

14. Section 014

안정성(Stability)은 유지 보수성(Maintainability)에 해당한다.

15. Section 015

프로토타입은 사용자의 요구사항을 개발자가 맞게 해석했는지 검증하기 위한 것으로, 최대한 간단하게 만드는 것이 좋다.

16. Section 015

프로토타입으로 검증할 범위를 너무 넓게 잡거나 기간을 길게 잡으면 원하는 목표가 너무 켜져 오히려 문제가 될 수 있다. 검증 범위는 작은 범위와 적은 인원으로 최소한의 기간 내에 검증할 수 있도록 지정하는 것이 좋다.

17. Section 016

UI 설계서는 상세 설계 이전에 작성한다.

3장 애플리케이션 설계

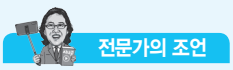
이 장에서 꼭 알아야 할 키워드 Best 10

1. 모듈화 **2.** 추상화 **3.** 정보은닉 **4.** 레이어 패턴 **5.** 클라이언트-서버 패턴 **6.** 모델-뷰-컨트롤러 패턴
7. 클래스 **8.** 캡슐화 **9.** 결합도 **10.** 응집도

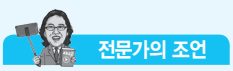
SECTION 020

LIVE

소프트웨어 아키텍처



소프트웨어 아키텍처의 특징과 역할, 설계의 기본 원리들을 확실히 파악하고 넘어가세요.



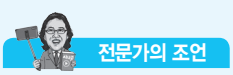
소프트웨어 아키텍처의 설계는 건축과 비교하면 쉽게 이해할 수 있습니다. 먼저 의뢰자의 요구사항에 맞추어 건물의 용도와 형태를 정하고, 땅을 어떻게 다질지, 골조는 어떻게 세울 것인지와 같이 대략적인 것을 정하는 과정이죠.

기능적/비기능적 요구사항

시스템이 갖춰야할 필수적인 기능에 대한 요구항목들을 기능적 요구사항이라고 하며, 그 외의 품질이나 제약사항에 관한 것을 비기능적 요구사항이라고 합니다. 요구사항에 대한 자세한 설명은 Section 006을 참조하세요.

모듈(Module)

모듈은 전체 프로그램의 기능 중에서 특정 기능을 처리할 수 있는 소스 코드를 의미합니다. 모듈에 대한 자세한 설명은 Section 023을 참조하세요.



불필요한 부분을 생략하고 필요한 부분을 강조하여 모델화 하는 것이 추상화입니다. 기억해 두세요.

1 소프트웨어 아키텍처의 설계

소프트웨어 아키텍처는 소프트웨어의 골격이 되는 기본 구조이자, 소프트웨어를 구성하는 요소들 간의 관계를 표현하는 시스템의 구조 또는 구조체이다.

- 소프트웨어 개발 시 적용되는 원칙과 지침이며, 이해 관계자들의 의사소통 도구로 활용된다.
- 소프트웨어 아키텍처의 설계는 기본적으로 좋은 품질을 유지하면서 사용자의 비기능적 요구사항*으로 나타난 제약을 반영하고, 기능적 요구사항*을 구현하는 방법을 찾는 해결 과정이다.
- 애플리케이션의 분할 방법과 분할된 모듈에 할당될 기능, 모듈 간의 인터페이스 등을 결정한다.
- 소프트웨어 아키텍처 설계의 기본 원리로는 모듈화, 추상화, 단계적 분해, 정보은닉이 있다.

2 모듈화(Modularity)

모듈화란 소프트웨어의 성능을 향상시키거나 시스템의 수정 및 재사용, 유지 관리 등이 용이하도록 시스템의 기능들을 모듈* 단위로 나누는 것을 의미한다.

- 자주 사용되는 계산식이나 사용자 인증과 같은 기능들을 공통 모듈로 구성하여 프로젝트의 재사용성을 향상시킬 수 있다.
- 모듈의 크기를 너무 작게 나누면 개수가 많아져 모듈 간의 통합 비용이 많이 들고, 너무 크게 나누면 개수가 적어 통합 비용은 적게 들지만 모듈 하나의 개발 비용이 많이 든다.

3 추상화(Abstraction)

추상화는 문제의 전체적이고 포괄적인 개념을 설계한 후 차례로 세분화하여 구체화시켜 나가는 것이다.

- 인간이 복잡한 문제를 다룰 때 가장 기본적으로 사용하는 방법으로, 완전한 시스템을 구축하기 전에 그 시스템과 유사한 모델을 만들어서 여러 가지 요인들을 테스트할 수 있다.
- 추상화는 최소의 비용으로 실제 상황에 대처할 수 있고, 시스템의 구조 및 구성을 대략적으로 파악할 수 있게 해준다.

- **추상화의 유형**

과정 추상화	자세한 수행 과정을 정의하지 않고, 전반적인 흐름만 파악할 수 있게 설계하는 방법
데이터 추상화	데이터의 세부적인 속성이나 용도를 정의하지 않고, 데이터 구조를 대표할 수 있는 표현으로 대체하는 방법
제어 추상화	이벤트 발생의 정확한 절차나 방법을 정의하지 않고, 대표할 수 있는 표현으로 대체하는 방법

4 단계적 분해(Stepwise Refinement)

단계적 분해는 Niklaus Wirth에 의해 제안된 하향식 설계 전략으로, 문제를 상위의 중요 개념으로부터 하위의 개념으로 구체화시키는 분할 기법이다.

- 추상화의 반복에 의해 세분화된다.
- 소프트웨어의 기능에서부터 시작하여 점차적으로 구체화하고, 알고리즘, 자료 구조 등 상세한 내역은 가능한 한 뒤로 미루어 진행한다.

전문가의 조언

건축을 예로 들면, 먼저 건물의 골조를 설계한 다음, 건물 내 층과 각 방의 경계를 정하고, 그 다음 방들의 인테리어를 구상하는 것과 같이 대략적인 설계에서 점차 세부인 설계로 넘어가는 것과 같다고 할 수 있습니다.

5 정보 은닉(Information Hiding)

정보 은닉은 한 모듈 내부에 포함된 절차와 자료들의 정보가 감추어져 다른 모듈이 접근하거나 변경하지 못하도록 하는 기법이다.

- 어떤 모듈이 소프트웨어 기능을 수행하는데 반드시 필요한 기능이 있어 정보 은닉된 모듈과 커뮤니케이션할 필요가 있을 때는 필요한 정보만 인터페이스를 통해 주고 받는다.
- 정보 은닉을 통해 모듈을 독립적으로 수행할 수 있고, 하나의 모듈이 변경되더라도 다른 모듈에 영향을 주지 않으므로 수정, 시험, 유지보수가 용이하다.

전문가의 조언

캡슐로 된 감기약을 예로 들면, 정보 은닉은 감기약 캡슐에 어떤 재료가 들어 있는지 몰라도 감기 걸렸을 때 먹는 약이라는 것만 알고 복용하는 것과 같은 의미입니다.

6 소프트웨어 아키텍처의 품질 속성

소프트웨어 아키텍처의 품질 속성은 소프트웨어 아키텍처가 이해 관계자들이 요구하는 수준의 품질을 유지 및 보장할 수 있게 설계되었는지를 확인하기 위해 품질 평가 요소들을 시스템 측면, 비즈니스 측면, 아키텍처 측면으로 구분하여 구체화시켜 놓은 것이다.

- **시스템 측면**

품질 속성	내용
성능	사용자의 요청과 같은 이벤트가 발생했을 때, 이를 적절하고 빠르게 처리하는 것이다.
보안	허용되지 않은 접근을 막고, 허용된 접근에는 적절한 서비스를 제공하는 것이다.
가용성	장애 없이 정상적으로 서비스를 제공하는 것이다.
기능성	사용자가 요구한 기능을 만족스럽게 구현하는 것이다.
사용성	사용자가 소프트웨어를 사용하는데 헤매지 않도록 명확하고 편리하게 구현하는 것이다.

전문가의 조언

문제에서 말하는 품질 속성들이 어떤 측면에 해당하는 것인지 가려낼 수 있어야 하며, 설명하는 내용이 어떤 속성을 의미하는지 파악할 수 있어야 합니다.

변경 용이성	소프트웨어가 처음 설계 목표와 다른 하드웨어나 플랫폼에서도 동작할 수 있도록 구현하는 것이다.
확장성	시스템의 용량, 처리능력 등을 확장시켰을 때 이를 효과적으로 활용할 수 있도록 구현하는 것이다.
기타 속성	테스트 용이성, 배치성, 안정성 등이 있다.

- **비즈니스 측면**

품질 속성	내용
시장 적시성	정해진 시간에 맞춰 프로그램을 출시하는 것이다.
비용과 혜택	• 개발 비용을 더 투자하여 유연성이 높은 아키텍처를 만들 것인지를 결정하는 것이다. • 유연성이 떨어지는 경우 유지보수에 많은 비용이 소모될 수 있다는 것을 고려해야 한다.
예상 시스템 수명	• 시스템을 얼마나 오랫동안 사용할 것인지를 고려하는 것이다. • 수명이 길어야 한다면 시스템 품질의 '변경 용이성', '확장성'을 중요하게 고려해야 한다.
기타 속성	목표 시장, 공개 일정, 기존 시스템과의 통합 등이 있다.

- **아키텍처 측면**

품질 속성	내용
개념적 무결성	전체 시스템과 시스템을 이루는 구성요소들 간의 일관성을 유지하는 것이다.
정확성, 완결성	요구사항과 요구사항을 구현하기 위해 발생하는 제약사항들을 모두 충족시키는 것이다.
구축 가능성	모듈 단위로 구분된 시스템을 적절하게 분배하여 유연하게 일정을 변경할 수 있도록 하는 것이다.
기타 속성	변경성, 시험성, 적응성, 일치성, 대체성 등이 있다.

7 소프트웨어 아키텍처의 설계 과정

아키텍처의 설계 과정은 설계 목표 설정, 시스템 타입 결정, 아키텍처 패턴* 적용, 서브시스템 구체화, 검토 순으로 진행된다.

❶ **설계 목표 설정** : 시스템의 개발 방향을 명확히 하기 위해 설계에 영향을 주는 비즈니스 목표, 우선순위 등의 요구사항을 분석하여 전체 시스템의 설계 목표를 설정한다.

❷ **시스템 타입 결정** : 시스템과 서브시스템의 타입을 결정하고, 설계 목표와 함께 고려하여 아키텍처 패턴을 선택한다.

❸ **아키텍처 패턴 적용** : 아키텍처 패턴을 참조하여 시스템의 표준 아키텍처를 설계한다.

❹ **서브시스템 구체화** : 서브시스템의 기능 및 서브시스템 간의 상호작용을 위한 동작과 인터페이스를 정의한다.

❺ **검토** : 아키텍처가 설계 목표에 부합하는지, 요구사항이 잘 반영되었는지, 설계의 기본 원리를 만족하는지 등을 검토한다.

아키텍처 패턴
아키텍처 패턴은 여러 다양한 상황에서 아키텍처를 설계하는데 발생하는 문제들을 해결하기 위해 미리 만들어 놓은 전형적인 해결 방식 또는 예제를 의미합니다. 아키텍처 패턴을 선택하고 이를 참조하여 표준 아키텍처를 설계한다는 말은 건물을 지을 때 전원주택 기본 구조도, 아파트 기본 구조도, 오피스텔 기본 구조도 등 이미 용도에 맞게 설계되어 있는 구조도 중 자신의 용도에 맞는 구조도를 하나 선택하고, 선택한 구조도를 수정하여 자신만의 구조도를 만드는 과정과 같습니다. 아키텍처 패턴에 대한 자세한 설명은 Section 021을 참조하세요.

시스템 타입

시스템 타입은 일반적으로 다음 네 가지 타입으로 나눌 수 있습니다.

- **대화형 시스템** : 사용자의 요구가 발생하면 시스템이 이를 처리하고 반응하는 시스템
  예 온라인 쇼핑몰과 같은 대부분의 웹 애플리케이션
- **이벤트 중심 시스템** : 외부의 상태 변화에 따라 동작하는 시스템
  예 전화, 비상벨 등의 내장 소프트웨어
- **변환형 시스템** : 데이터가 입력되면 정해진 작업들을 수행하여 결과를 출력하는 시스템
  예 컴파일러, 네트워크 프로토콜 등
- **객체 영속형 시스템** : 데이터베이스를 사용하여 파일을 효과적으로 저장·검색·갱신할 수 있는 시스템
  예 서버 관리 소프트웨어

기출문제 따라잡기 Section 020

출제예상

1. 소프트웨어 아키텍처 설계의 기본 원리 중 다음 설명에 해당하는 것은?

> • 다른 객체에게 자신의 정보를 숨기고 자신의 연산만을 통하여 접근한다.
> • 유지보수와 소프트웨어 확장 시 오류를 최소화할 수 있다.

① Abstraction ② Information Hiding
③ Inheritance ④ Polymorphism

문제의 지문을 잘 보세요. '~ 자신의 정보를 숨기고 ~'란 말과 관련된 설계의 기본 원리를 찾아야 합니다.

출제예상

2. 소프트웨어 아키텍처 설계의 기본 원리 중 다음 설명이 의미하는 것은?

> 객체의 성질을 분해하고 공통된 성질을 추출하여 슈퍼 클래스를 선정하는 것이다. 즉 불필요한 부분을 생략하고 객체의 속성 중 가장 중요한 것에만 중점을 두어 개략화, 모델화하는 것이다. 예를 들면, 자동차와 자전거란 클래스에서 "타는 것"이란 클래스를 만드는 것이다.

① Inheritance ② Abstraction
③ Polymorphism ④ Encapsulation

개략화·모델화는 추상화를 의미하는 핵심 단어입니다. 이제 정답을 찾아보세요.

출제예상

3. 소프트웨어 아키텍처 설계의 기본 원리에 해당하지 않는 것은?

① 모듈화 ② 캡슐화
③ 단계적 분해 ④ 추상화

소프트웨어 아키텍처 설계의 기본 원리는 4가지로, 모듈화, 추상화, 단계적 분해, 정보 은닉입니다.

출제예상

4. 소프트웨어 아키텍처 설계의 모듈화에 대한 설명으로 옳지 않은 것은?

① 시스템의 수정 및 재사용, 유지 관리 등을 위해 기능별로 분해하는 것을 의미한다.
② 자주 사용되는 기능을 공통 모듈로 구성할 수 있다.
③ 모듈의 개수가 많으면 모듈 하나의 개발 비용이 적어진다.
④ 모듈의 크기가 크면 모듈 간의 통합 비용이 많이 든다.

모듈의 크기가 크면 모듈 간의 통합 비용이 적어지는 대신 모듈 하나의 개발 비용이 커지게 됩니다.

출제예상

5. 소프트웨어 아키텍처의 품질 속성에 대한 설명으로 옳지 않은 것은?

① 가용성은 장애 없이 정상적으로 서비스를 제공하는 것을 의미한다.
② 사용성은 사용자가 불편 없이 소프트웨어를 사용할 수 있도록 구현하는 것이다.
③ 구축 가능성은 적절한 업무 분배를 통해 유연한 일정 변경을 가능하도록 하는 것이다.
④ 성능은 사용자가 요구한 기능을 만족스럽게 구현하는 것을 의미한다.

'성능'은 사용자의 요청과 같은 이벤트가 발생했을 때, 이를 적절하고 빠르게 처리하는 것을 의미합니다.

▶ 정답 : 1. ② 2. ② 3. ② 4. ④ 5. ④

SECTION 021 ● LIVE

아키텍처 패턴

전문가의 조언

아키텍처의 각 패턴들의 특징을 확실히 파악하고 넘어가세요.

전문가의 조언

아키텍처 패턴은 건축과 비교하면 이해가 쉽습니다. 예를 들어 오피스텔을 짓는다고 가정할 때, "오피스텔을 지을 때는 이런 재질의 골조가, 복도의 넓이는 이정도, 층간 높이는 이만큼이 가장 적절하더라."라는 오피스텔 설계에 대한 가이드라인이 존재한다면 이를 참조해 손쉽게 설계가 가능합니다. 물론 사용자의 요구사항에 따라 세부적인 설계가 변경될 수는 있겠지만 아무런 자료 없이 처음부터 모든 것을 설계하는 것 보다는 훨씬 쉽겠죠? 이 가이드라인이 바로 소프트웨어 설계에서는 아키텍처 패턴에 해당한다고 할 수 있습니다.

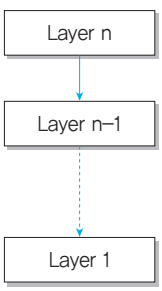
1 아키텍처 패턴(Patterns)의 개요

아키텍처 패턴은 아키텍처를 설계할 때 참조할 수 있는 전형적인 해결 방식 또는 예제를 의미한다.

- 아키텍처 패턴은 소프트웨어 시스템의 구조를 구성하기 위한 기본적인 윤곽을 제시한다.
- 아키텍처 패턴에는 서브시스템들과 그 역할이 정의되어 있으며, 서브시스템 사이의 관계와 여러 규칙 · 지침 등이 포함되어 있다.
- 아키텍처 패턴을 아키텍처 스타일 또는 표준 아키텍처라고도 한다.
- **아키텍처 패턴의 장점**
  - 시행착오를 줄여 개발 시간을 단축시키고, 고품질의 소프트웨어를 생산할 수 있다.
  - 검증된 구조로 개발하기 때문에 안정적인 개발이 가능하다.
  - 이해관계자들이 공통된 아키텍처를 공유할 수 있어 의사소통이 간편해진다.
  - 시스템의 구조를 이해하는 것이 쉬워 개발에 참여하지 않은 사람도 손쉽게 유지보수를 수행할 수 있다.
  - 시스템의 특성을 개발 전에 예측하는 것이 가능해진다.
- 아키텍처 패턴의 종류에는 레이어 패턴, 클라이언트-서버 패턴, 파이프-필터 패턴, 모델-뷰-컨트롤러 패턴 등이 있다.

2 레이어 패턴(Layers pattern)

레이어 패턴은 시스템을 계층(Layer)으로 구분하여 구성하는 고전적인 방법 중의 하나다.



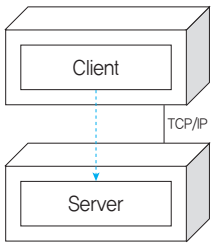
- 레이어 패턴은 각각의 서브시스템들이 계층 구조를 이루며, 상위 계층은 하위 계층에 대한 서비스 제공자가 되고, 하위 계층은 상위 계층의 클라이언트가 된다.
- 레이어 패턴은 서로 마주보는 두 개의 계층 사이에서만 상호작용이 이루어지며, 변경 사항을 적용할 때도 서로 마주보는 두 개의 계층에만 영향을 미치므로 변경 작업이 용이하다.
- 레이어 패턴은 특정 계층만을 교체해 시스템을 개선하는 것이 가능하다.
- 대표적으로 OSI 참조 모델*이 있다.

OSI 참조 모델

OSI 참조 모델은 국제표준화기구(ISO)에서 네트워크 프로토콜을 계층별로 구분한 모델로 물리 계층, 데이터 링크 계층, 네트워크 계층, 전송 계층, 세션 계층, 표현 계층, 응용 계층으로 구성되어 있습니다.

3 클라이언트-서버 패턴(Client-Server Pattern)

클라이언트-서버 패턴은 하나의 서버 컴포넌트*와 다수의 클라이언트 컴포넌트로 구성되는 패턴이다.



- 클라이언트-서버 패턴에서 사용자는 클라이언트와만 의사소통을 한다. 즉 사용자가 클라이언트를 통해 서버에 요청하고 클라이언트가 응답을 받아 사용자에게 제공하는 방식으로 서비스를 제공한다.
- 서버는 클라이언트의 요청에 대비해 항상 대기 상태를 유지해야 한다.
- 클라이언트나 서버는 요청과 응답을 받기 위해 동기화되는 경우를 제외하고는 서로 독립적이다.

컴포넌트(Component)
컴포넌트는 독립적인 업무 또는 기능을 수행하는 실행코드 기반으로 작성된 모듈입니다.

4 파이프-필터 패턴(Pipe-Filter Pattern)

파이프-필터 패턴은 데이터 스트림* 절차의 각 단계를 필터(Filter) 컴포넌트로 캡슐화하여 파이프(Pipe)를 통해 데이터를 전송하는 패턴이다.

- 필터 컴포넌트는 재사용성이 좋고, 추가가 쉬워 확장이 용이하다.
- 필터 컴포넌트들을 재배치하여 다양한 파이프라인*을 구축하는 것이 가능하다.
- 파이프-필터 패턴은 데이터 변환, 버퍼링, 동기화 등에 주로 사용된다.
- 대표적으로 UNIX의 쉘(Shell)이 있다.

데이터 스트림(Data Stream)
데이터 스트림은 데이터가 송·수신되거나 처리되는 일련의 연속적인 흐름입니다.

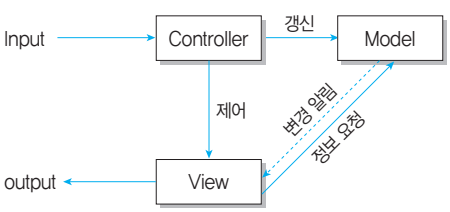
파이프라인(Pipeline)
파이프라인은 필터와 파이프를 통해 처리되는 일련의 처리 과정입니다.

5 모델-뷰-컨트롤러 패턴(Model-View-Controller Pattern)

모델-뷰-컨트롤러 패턴은 서브시스템을 3개의 부분으로 구조화하는 패턴이며, 각 부분의 역할은 다음과 같다.

- 모델(Model) : 서브시스템의 핵심 기능과 데이터를 보관한다.
- 뷰(View) : 사용자에게 정보를 표시한다.
- 컨트롤러(Controller) : 사용자로부터 받은 입력을 처리한다.

- 모델-뷰-컨트롤러 패턴의 각 부분은 별도의 컴포넌트로 분리되어 있으므로 서로 영향을 받지 않고 개발 작업을 수행할 수 있다.
- 모델-뷰-컨트롤러 패턴에서는 여러 개의 뷰를 만들 수 있으므로 한 개의 모델에 대해 여러 개의 뷰를 필요로 하는 대화형 애플리케이션*에 적합하다.



대화형 애플리케이션
대화형 애플리케이션은 온라인 쇼핑몰 사이트나 스마트폰 앱과 같이 사용자의 요구가 발생하면 시스템이 이를 처리하고 반응하는 소프트웨어를 의미합니다.

6 기타 패턴

마스터-슬레이브 패턴 (Master-Slave Pattern)	• 마스터 컴포넌트에서 슬레이브 컴포넌트로 작업을 분할한 후, 슬레이브 컴포넌트에서 처리된 결과물을 다시 돌려받는 방식으로 작업을 수행하는 패턴이다. • 마스터 컴포넌트는 모든 작업의 주체이고, 슬레이브 컴포넌트는 마스터 컴포넌트의 지시에 따라 작업을 수행하여 결과를 반환한다. • 장애 허용 시스템*과 병렬 컴퓨팅 시스템에서 주로 활용된다.
브로커 패턴 (Broker Pattern)	• 사용자가 원하는 서비스와 특성을 브로커 컴포넌트에 요청하면 브로커 컴포넌트가 요청에 맞는 컴포넌트와 사용자를 연결해준다. • 원격 서비스 호출에 응답하는 컴포넌트들이 여러 개 있을 때 적합한 패턴이다. • 분산 환경 시스템에서 주로 활용된다.
피어-투-피어 패턴 (Peer-To-Peer Pattern)	• 피어(Peer)를 하나의 컴포넌트로 간주하며, 각 피어는 서비스를 호출하는 클라이언트가 될 수도, 서비스를 제공하는 서버가 될 수도 있는 패턴이다. • 피어-투-피어 패턴에서 클라이언트와 서버는 전형적인 멀티스레딩* 방식을 사용한다.
이벤트-버스 패턴 (Event-Bus Pattern)	• 소스가 특정 채널에 이벤트 메시지*를 발행(Publish)하면, 해당 채널을 구독(Subscribe)한 리스너들이 메시지를 받아 이벤트를 처리하는 방식이다. • 4가지 주요 컴포넌트 – 이벤트를 생성하는 소스(Source) – 이벤트를 수행하는 리스너(Listener) – 이벤트의 통로인 채널(Channel) – 채널들을 관리하는 버스(Bus)
블랙보드 패턴 (Blackboard Pattern)	• 모든 컴포넌트들이 공유 데이터 저장소와 블랙보드 컴포넌트에 접근이 가능한 형태로, 컴포넌트들은 검색을 통해 블랙보드에서 원하는 데이터를 찾을 수 있다. • 해결책이 명확하지 않은 문제를 처리하는데 유용한 패턴이다. • 음성 인식, 차량 식별, 신호 해석 등에 주로 활용된다.
인터프리터 패턴 (Interpreter Pattern)	• 프로그램 코드의 각 라인을 수행하는 방법을 지정하고, 기호마다 클래스를 갖도록 구성된다. • 특정 언어로 작성된 프로그램 코드를 해석하는 컴포넌트를 설계할 때 사용되어진다.

장애 허용 시스템(Fault Tolerance System)
장애 허용 시스템은 시스템의 일부가 결함 또는 고장으로 기능이 정지되더라도 해당 부분의 기능만 수행이 불가능할 뿐 전체 시스템은 정상적으로 수행이 가능한 시스템을 말합니다.

멀티스레딩(Multi Threading)
멀티스레딩은 프로세스를 두 개 이상의 실행 단위로 구분하여 자원을 공유하며 병렬로 수행하는 기능합니다.

메시지(Message)
메시지는 객체들 간에 상호작용을 하는 데 사용되는 수단으로, 객체에게 어떤 행위를 하도록 지시하는 명령 또는 요구사항 입니다.

기출문제 따라잡기 Section 021

출제예상

1. 소프트웨어 아키텍처의 패턴 중 Pipe-Filter에 대한 설명으로 옳은 것은?

① 컴포넌트 간의 인터페이스가 복잡한 시스템에 적합하다.
② 유저가 직접 데이터 흐름에 간섭할 필요가 있는 경우에 사용된다.
③ 모든 컴포넌트가 동시에 처리되어야 하는 병렬처리 시스템에 적합하다.
④ 하나의 컴포넌트에서 처리가 끝나면 다음 컴포넌트가 결과물을 받아 처리하는 과정이 반복된다.

Pipe-Filter Pattern은 각 단계를 필터(Filter) 컴포넌트로 캡슐화하여 앞의 컴포넌트에서 처리한 결과를 파이프(Pipe)를 통해서 전송하면 뒤의 컴포넌트에서 결과를 받아 다음 작업을 수행하는 구조로 구성되어 있습니다.

출제예상

2. 네트워크 프로토콜의 OSI 참조 모델과 가장 관련이 깊은 아키텍처 모델은?

① Peer-To-Peer Model
② Mvc Model
③ Layers Model
④ Client-Server Model

OSI 참조 모델은 네트워크 프로토콜을 계층(Layer) 별로 구분한 모델입니다.

출제예상

3. 아키텍처 패턴에서 사용되는 MVC(Model-View-Controller) 모델은 서브시스템을 모델, 뷰, 컨트롤러로 구조화한다. 이중 Model 부분의 기능은?

① 사용자에게 정보 표시
② 데이터 처리 및 보관
③ 사용자 인터페이스를 담당
④ 뷰의 제어

①번은 뷰(View)의 기능, ③ · ④번은 컨트롤러(Controller)의 기능에 해당합니다. 각 부분의 기능을 확실히 파악해 두세요.

출제예상

4. 아키텍처 패턴(Architecture Pattern)에 대한 설명 중 가장 옳지 않은 것은?

① 소프트웨어 초기 설계에서 발생하는 문제들을 해결하기 위한 전형적인 해결 방식들을 의미한다.
② 검증된 구조로 개발하기 때문에 오류가 적어 개발시간을 단축할 수 있다.
③ 서브시스템들에 대한 역할을 정의하고 있지만, 그들 간의 인터페이스에 대한 지침은 없다.
④ 시스템에 대한 이해가 쉬워지고, 특성을 예측할 수 있게 된다.

아키텍처 패턴은 서브시스템들의 역할을 사전에 정의할 뿐만 아니라, 시스템 간의 관계(Interface)를 정리하기 위한 규칙과 지침이 포함되어 있습니다.

출제예상

5. 다음 중 클라이언트-서버(Client-Server) 모델에 대한 설명으로 가장 거리가 먼 것은?

① 사용자는 클라이언트를 통해서 요청을 전달하며, 서버는 이에 응답하는 방식이다.
② 서버는 클라이언트의 요청에 대비하여 항상 대기 상태를 유지한다.
③ 서버와 클라이언트는 서로 독립적이다.
④ 다수의 서버와 하나의 클라이언트로 구성되는 패턴으로 분산 환경 시스템에 적합하다.

클라이언트-서버(Client-Server) 패턴은 하나의 서버와 다수의 클라이언트로 구성되는 패턴입니다.

출제예상

6. 여러 컴포넌트들 중 각 컴포넌트들이 서비스를 제공하는 서버가 될 수도 있고, 서비스를 요청하는 클라이언트도 될 수 있는 패턴으로 전형적인 멀티스레딩을 사용하는 방식의 패턴을 무엇이라 하는가?

① 클라이언트-서버
② 블랙보드
③ 이벤트-버스
④ 피어-투-피어

패턴의 종류 중 멀티스레드 방식하면 피어-투-피어(Peer-to-Peer)라는 것을 기억하세요.

▶ 정답 : 1. ④ 2. ③ 3. ② 4. ③ 5. ④ 6. ④

SECTION 022 ● LIVE

객체지향(Object-Oriented)

전문가의 조언

객체지향의 특징과 객체지향과 관련된 용어들을 확실히 파악하고 넘어가세요.

현실 세계의 개체

현실 세계의 개체는 사람, 자동차, 컴퓨터, 고양이 등과 같이 우리 주위에서 사용되는 물질적이거나 개념적인 것으로, 명사로 사용됩니다.

구조적 기법

구조적 기법은 프로시저에 근간을 두고 하나의 커다란 작업을 여러 개의 작은 작업으로 분할하고, 분할된 각각의 소작업을 수행하는 모듈을 작성한 다음 이들을 한 곳에 모아 큰 작업을 수행하는 하나의 완벽한 프로그램으로 작성하는 기법입니다.

구조적 기법의 문제점

- 유지보수는 고려하지 않고 개발 공정에만 너무 집중합니다.
- 개발이 시작된 이후 추가적인 요구사항에 대응하기 어렵습니다.
- 재사용이 어려워 이전에 개발한 소프트웨어와 유사한 소프트웨어를 다시 개발할 때도 시간과 인력이 동일하게 소모됩니다.

1 객체지향의 개요

객체지향은 현실 세계의 개체(Entity)*를 기계의 부품처럼 하나의 객체(Object)로 만들어, 기계적인 부품들을 조립하여 제품을 만들 듯이 소프트웨어를 개발할 때에도 객체들을 조립해서 작성할 수 있는 기법을 말한다.

- 객체지향 기법은 구조적 기법의 문제점*으로 인한 소프트웨어 위기의 해결책으로 채택되어 사용되고 있다.
- 객체지향은 소프트웨어의 재사용 및 확장이 용이하여 고품질의 소프트웨어를 빠르게 개발할 수 있고 유지보수가 쉽다.
- 객체지향은 복잡한 구조를 단계적 · 계층적으로 표현하고, 멀티미디어 데이터 및 병렬 처리를 지원한다.
- 객체지향은 현실 세계를 모형화하므로 사용자와 개발자가 쉽게 이해할 수 있다.
- 객체지향의 주요 구성 요소와 개념에는 객체(Object), 클래스(Class), 캡슐화(Encapsulation), 상속(Inheritance), 다형성(Polymorphism)이 있다.

2 객체(Object)

객체는 데이터와 데이터를 처리하는 함수를 묶어 놓은(캡슐화한) 하나의 소프트웨어 모듈이다.

데이터	• 객체가 가지고 있는 정보로 속성이나 상태, 분류 등을 나타낸다. • 속성(Attribute), 상태, 변수, 상수, 자료 구조라고도 한다.
함수	• 객체가 수행하는 기능으로 객체가 갖는 데이터(속성, 상태)를 처리하는 알고리즘이다. • 객체의 상태를 참조하거나 변경하는 수단이 되는 것으로 메소드(Method, 행위), 서비스(Service), 동작(Operation), 연산이라고도 한다.

- **객체의 특성**
  - 객체는 독립적으로 식별 가능한 이름을 가지고 있다.
    - 예 자동차는 번호판으로 다른 자동차 객체와 구별된다.
  - 객체가 가질 수 있는 조건을 상태(State)라고 하는데, 일반적으로 상태는 시간에 따라 변한다.
    - 예 자동차는 '정지', '이동' 등의 상태가 존재하며, 이러한 '정지'와 '이동'의 상태는 고정된 것이 아니라 시간에 따라 변한다.
  - 객체와 객체는 상호 연관성에 의한 관계가 형성된다.

예 화재 발생 시 소방차, 구급차, 경찰차는 긴밀하게 협조하여 화재를 진압하고 환자를 이송하며, 교통을 정리하는 관계가 형성된다.

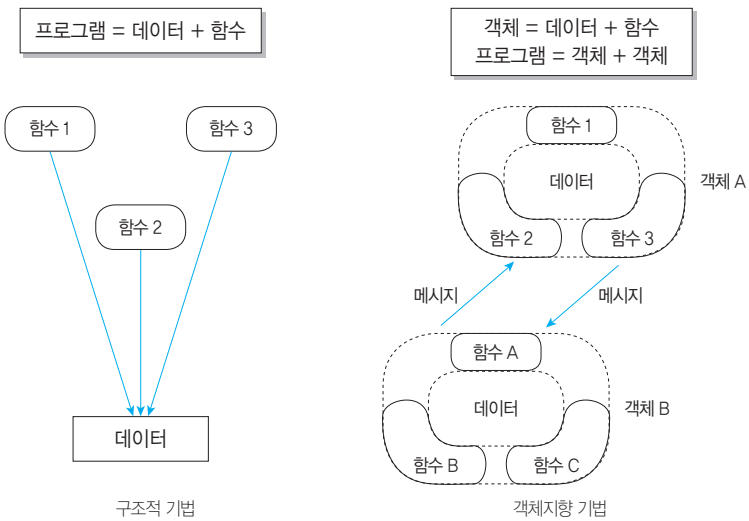
– 객체가 반응할 수 있는 메시지(Message)의 집합을 행위라고 하며, 객체는 행위의 특징을 나타낼 수 있다.

예 자동차 객체는 '가속 페달을 밟는' 행위를 하면 '가속'하는 특징을 나타내고, '브레이크를 밟는' 행위를 하면 '감속'하는 특징을 나타낸다.

– 객체는 일정한 기억장소를 가지고 있다.

예 자동차는 주차장에 있거나 도로 위에 있거나, 일정한 물리적 공간을 점유한다.

- 객체의 메소드는 다른 객체로부터 메시지를 받았을 때 정해진 기능을 수행한다.



3 클래스(Class)

클래스는 공통된 속성과 연산(행위)을 갖는 객체의 집합으로, 객체의 일반적인 타입(Type)을 의미한다.

- 클래스는 각각의 객체들이 갖는 속성과 연산을 정의하고 있는 틀이다.
- 클래스에 속한 각각의 객체를 인스턴스(Instance)라 하며, 클래스로부터 새로운 객체를 생성하는 것을 인스턴스화(Instantiation)라고 한다.
- 동일 클래스에 속한 각각의 객체(인스턴스)들은 공통된 속성과 행위를 가지고 있으면서, 그 속성에 대한 정보가 서로 달라서 동일 기능을 하는 여러 가지 객체를 나타내게 된다.
- 최상위 클래스는 상위 클래스를 갖지 않는 클래스를 의미한다.
- 슈퍼 클래스(Super Class)는 특정 클래스의 상위(부모) 클래스이고, 서브 클래스(Sub Class)는 특정 클래스의 하위(자식) 클래스를 의미한다.

전문가의 조언

캡슐로 된 알약과 비교하면 이해가 쉽습니다. 특정 질환을 치료하기 위해 서로 다른 약들을 조합하여 캡슐에 담아놓는 것과 같이 데이터와 함수들을 묶었다고 생각하면 됩니다.

4 캡슐화(Encapsulation)

캡슐화는 데이터(속성)와 데이터를 처리하는 함수를 하나로 묶는 것을 의미한다.

- 캡슐화된 객체는 인터페이스를 제외한 세부 내용이 은폐(정보 은닉)되어 외부에서의 접근이 제한적이기 때문에 외부 모듈의 변경으로 인한 파급 효과가 적다.
- 캡슐화된 객체들은 재사용이 용이하다.
- 객체들 간의 메시지를 주고받을 때 상대 객체의 세부 내용은 알 필요가 없으므로 인터페이스가 단순해지고, 객체 간의 결합도가 낮아진다.

전문가의 조언

상속은 '학생'을 정의하는 상위 클래스를 하위 클래스가 물려받아 '남자'라는 속성을 첨가하면 상위 클래스에 비해 좀 더 구체적인 '남학생'이라는 클래스가 구성되는 것이라고 생각하면 됩니다.

다중 상속

다중 상속은 클래스 계층을 복잡하게 만들어 상속 순서 추적이 어렵고, 상위 클래스의 변경이 하위 클래스에 의도하지 않은 영향을 미칠 수도 있어 다중 상속을 허용하지 않는 프로그래밍 언어들도 있습니다. 다중 상속이 가능한 프로그래밍 언어에서도 다중 상속을 이용할 때는 이를 고려하여 신중히 사용합니다.

5 상속(Inheritance)

상속은 이미 정의된 상위 클래스(부모 클래스)의 모든 속성과 연산을 하위 클래스(자식 클래스)가 물려받는 것이다.

- 상속을 이용하면 하위 클래스는 상위 클래스의 모든 속성과 연산을 자신의 클래스 내에서 다시 정의하지 않고서도 즉시 자신의 속성으로 사용할 수 있다.
- 하위 클래스는 상위 클래스로부터 상속받은 속성과 연산 외에 새로운 속성과 연산을 첨가하여 사용할 수 있다.
- 상위 클래스의 속성과 연산을 하위 클래스가 사용할 수 있기 때문에 객체와 클래스의 재사용, 즉 소프트웨어의 재사용(Reuse)을 높이는 중요한 개념이다.
- **다중 상속(Multiple Inheritance)*** : 한 개의 클래스가 두 개 이상의 상위 클래스로부터 속성과 연산을 상속받는 것이다.

전문가의 조언

다형성은 여러 가지 형태를 가지고 있다는 의미로 하나의 메시지에 대해 여러 가지 형태의 응답이 있다는 것을 의미합니다. 다형성의 의미를 정확히 기억하고, 특징을 파악해 두세요.

6 다형성(Polymorphism)

다형성은 메시지에 의해 객체(클래스)가 연산을 수행하게 될 때 하나의 메시지에 대해 각각의 객체(클래스)가 가지고 있는 고유한 방법(특성)으로 응답할 수 있는 능력을 의미한다.

- 객체(클래스)들은 동일한 메소드명을 사용하며 같은 의미의 응답을 한다.
- 응용 프로그램 상에서 하나의 함수나 연산자가 두 개 이상의 서로 다른 클래스의 인스턴스들을 같은 클래스에 속한 인스턴스처럼 수행할 수 있도록 하는 것이다.

예 '+' 연산자의 경우 숫자 클래스에서는 덧셈, 문자 클래스에서는 문자열의 연결 기능으로 사용된다.

기출문제 따라잡기

이전기출

1. 객체지향 소프트웨어 공학의 상속에 대해 바르게 설명한 것은?

① 상위 클래스의 메서드와 속성을 하위 클래스가 물려받는 것
② 다른 객체에게 자신의 정보를 숨기는 것
③ 클래스로부터 만들어진 하나의 인스턴스
④ 객체가 연산을 수행할 때 하나의 메시지에 대해 각 객체가 가지고 있는 고유한 방법으로 응답할 수 있는 것

상속은 단어만으로도 그 의미를 파악할 수 있죠? 상속한다. 상속세 등의 의미를 생각하면 쉽게 답을 찾을 수 있습니다.

출제예상

2. 기존의 소프트웨어 공학 기법들과 차별화될 수 있는 객체지향 개념이 아닌 것은?

① 캡슐화(Encapsulation)
② 상속(Inheritance)
③ 다형성(Polymorphism)
④ 모듈화(Modularity)

보기 중 이번 섹션에서 공부하지 않은 것이 하나있네요. 모르겠다면 왼쪽 페이지의 필드명을 확인하세요.

이전기출

3. 객체에 대한 설명으로 가장 옳지 않은 것은?

① 객체는 실세계 또는 개념적으로 존재하는 세계의 사물들이다.
② 객체는 공통적인 특징을 갖는 클래스들을 모아둔 것이다.
③ 객체는 데이터를 가지며 이 데이터의 값을 변경하는 함수를 가지고 있는 경우도 있다.
④ 객체들 사이에서 통신을 할 때는 메시지를 전송한다.

클래스를 모아둔 것이 객체가 아니라, 객체를 모아둔 것이 클래스입니다.

이전기출

4. 객체지향 기법에서 어떤 클래스에 속하는 구체적인 객체를 의미하는 것은?

① Instance ② Message
③ Method ④ Operation

클래스에 속하는 각각의 객체를 인스턴스라 하고, 클래스로부터 객체를 생성하는 것은 인스턴스화라고 합니다.

이전기출

5. 객체지향 기법에서 캡슐화(Encapsulation)에 대한 설명으로 옳지 않은 것은?

① 캡슐화를 하면 객체 간의 결합도가 높아진다.
② 캡슐화된 객체들은 재사용이 용이하다.
③ 프로그램 변경에 대한 오류의 파급효과가 적다.
④ 인터페이스가 단순해진다.

캡슐화된 객체는 객체 내의 응집도는 높아지고, 객체 간의 결합도는 낮아집니다.

▶ 정답 : 1. ① 2. ④ 3. ② 4. ① 5. ①

SECTION 023 ● LIVE

모듈

전문가의 조언

모듈의 개념을 이해하고 응집도, 결합도의 종류들에 대해 확실히 파악하고 넘어가세요.

모듈화(Modularity)

모듈화는 소프트웨어의 성능을 향상시키거나 시스템의 수정 및 재사용, 유지 관리 등이 용이하도록 시스템의 기능들을 모듈 단위로 분해하는 것을 의미합니다.

- **루틴(Routine)** : 기능을 가진 명령들의 모임
- **메인 루틴(Main Routine)** : 프로그램 실행의 큰 줄기가 되는 것
- **서브 루틴(Subroutine)** : 메인 루틴에 의해 필요할 때 마다 호출되는 루틴

서브시스템(Subsystem)

서브시스템은 시스템을 구성하는 요소의 하나로, '단위시스템'이라고도 불리며, 서브시스템 자체로도 하나의 시스템에 필요한 요소들을 갖추고 있습니다. 예를 들어 메인 시스템이 '통합 경영정보 시스템'이라면 여기에 속하는 서브시스템으로 '영업관리 시스템', '생산관리 시스템', '인사관리 시스템' 등이 있을 수 있습니다.

전문가의 조언

결합도는 두 사람이 붙어있는 것과 비교하면 이해가 쉽습니다. 손만 잡고 있는지, 팔짱을 끼고 있는지, 포옹을 하고 있는지에 따라 서로 떨어뜨리기가 어려워지겠죠.

1 모듈(Module)의 개요

모듈은 모듈화*를 통해 분리된 시스템의 각 기능들로, 서브루틴*, 서브시스템*, 소프트웨어 내의 프로그램, 작업 단위 등과 같은 의미로 사용된다.

- 모듈은 단독으로 컴파일이 가능하며, 재사용 할 수 있다.
- 모듈의 기능적 독립성은 소프트웨어를 구성하는 각 모듈의 기능이 서로 독립됨을 의미하는 것으로, 모듈이 하나의 기능만을 수행하고 다른 모듈과의 과도한 상호작용을 배제함으로써 이루어진다.
- 독립성이 높은 모듈일수록 모듈을 수정하더라도 다른 모듈들에게는 거의 영향을 미치지 않으며, 오류가 발생해도 쉽게 발견하고 해결할 수 있다.
- 모듈의 독립성은 결합도(Coupling)와 응집도(Cohesion)에 의해 측정되며, 독립성을 높이려면 모듈의 결합도는 약하게, 응집도는 강하게, 모듈의 크기는 작게 만들어야 한다.

2 결합도(Coupling)

결합도는 모듈 간에 상호 의존하는 정도 또는 두 모듈 사이의 연관 관계를 의미한다.

- 다양한 결합으로 모듈을 구성할 수 있으나 결합도가 약할수록 품질이 높고, 강할수록 품질이 낮다.
- 결합도가 강하면 시스템 구현 및 유지보수 작업이 어렵다.
- 결합도의 종류에는 자료 결합도, 스탬프 결합도, 제어 결합도, 외부 결합도, 공통 결합도, 내용 결합도가 있으며 결합도의 정도는 다음과 같다.

자료 결합도	스탬프 결합도	제어 결합도	외부 결합도	공통 결합도	내용 결합도

결합도 약함 ◀——————————▶ 결합도 강함

자료 결합도 (Data Coupling)	• 모듈 간의 인터페이스가 자료 요소로만 구성될 때의 결합도이다. • 어떤 모듈이 다른 모듈을 호출하면서 매개 변수나 인수로 데이터를 넘겨주고, 호출 받은 모듈은 받은 데이터에 대한 처리 결과를 다시 돌려주는 방식이다. • 모듈 간의 내용을 전혀 알 필요가 없는 상태로서 한 모듈의 내용을 변경하더라도 다른 모듈에는 전혀 영향을 미치지 않는 가장 바람직한 결합도이다.
스탬프(검인) 결합도 (Stamp Coupling)	• 모듈 간의 인터페이스로 배열이나 레코드 등의 자료 구조가 전달될 때의 결합도이다. • 두 모듈이 동일한 자료 구조를 조회하는 경우의 결합도이며, 자료 구조의 어떠한 변화, 즉 포맷이나 구조의 변화는 그것을 조회하는 모든 모듈 및 변화되는 필드를 실제로 조회하지 않는 모듈에까지도 영향을 미치게 된다.

제어 결합도 (Control Coupling)	• 어떤 모듈이 다른 모듈 내부의 논리적인 흐름을 제어하기 위해 제어 신호를 이용하여 통신하거나 제어 요소(Function Code, Switch, Tag, Flag)를 전달하는 결합도이다. • 한 모듈이 다른 모듈의 상세한 처리 절차를 알고 있어 이를 통제하는 경우나 처리 기능이 두 모듈에 분리되어 설계된 경우에 발생한다. • 하위 모듈에서 상위 모듈로 제어 신호가 이동하여 하위 모듈이 상위 모듈에게 처리 명령을 내리는 권리 전도현상이 발생하게 된다.
외부 결합도 (External Coupling)	• 어떤 모듈에서 선언한 데이터(변수)를 외부의 다른 모듈에서 참조할 때의 결합도이다. • 참조되는 데이터의 범위를 각 모듈에서 제한할 수 있다.
공통(공유) 결합도 (Common Coupling)	• 공유되는 공통 데이터 영역을 여러 모듈이 사용할 때의 결합도이다. • 공통 데이터 영역의 내용을 조금만 변경하더라도 이를 사용하는 모든 모듈에 영향을 미치므로 모듈의 독립성을 약하게 만든다.
내용 결합도 (Content Coupling)	• 한 모듈이 다른 모듈의 내부 기능 및 그 내부 자료를 직접 참조하거나 수정할 때의 결합도이다. • 한 모듈에서 다른 모듈의 내부로 제어가 이동하는 경우에도 내용 결합도에 해당된다.

3 응집도(Cohesion)

응집도는 정보 은닉* 개념을 확장한 것으로, 명령어나 호출문 등 모듈의 내부 요소들의 서로 관련되어 있는 정도, 즉 모듈이 독립적인 기능으로 정의되어 있는 정도를 의미한다.

- 다양한 기준으로 모듈을 구성할 수 있으나 응집도가 강할수록 품질이 높고, 약할수록 품질이 낮다.
- 응집도의 종류에는 기능적 응집도, 순차적 응집도, 교환(통신)적 응집도, 절차적 응집도, 시간적 응집도, 논리적 응집도, 우연적 응집도가 있으며 응집도의 정도는 다음과 같다.

기능적 응집도	순차적 응집도	교환적 응집도	절차적 응집도	시간적 응집도	논리적 응집도	우연적 응집도

응집도 강함 ← → 응집도 약함

기능적 응집도 (Functional Cohesion)	모듈 내부의 모든 기능 요소들이 단일 문제와 연관되어 수행될 경우의 응집도
순차적 응집도 (Sequential Cohesion)	모듈 내 하나의 활동으로부터 나온 출력 데이터를 그 다음 활동의 입력 데이터로 사용할 경우의 응집도
교환(통신)적 응집도 (Communication Cohesion)	동일한 입력과 출력을 사용하여 서로 다른 기능을 수행하는 구성 요소들이 모였을 경우의 응집도
절차적 응집도 (Procedural Cohesion)	모듈이 다수의 관련 기능을 가질 때 모듈 안의 구성 요소들이 그 기능을 순차적으로 수행할 경우의 응집도
시간적 응집도 (Temporal Cohesion)	특정 시간에 처리되는 몇 개의 기능을 모아 하나의 모듈로 작성할 경우의 응집도
논리적 응집도 (Logical Cohesion)	유사한 성격을 갖거나 특정 형태로 분류되는 처리 요소들로 하나의 모듈이 형성되는 경우의 응집도

전문가의 조언

응집도는 어질러진 방의 물건들을 상자에 정리하는 것과 비교하면 이해가 쉽습니다. 용도나 종류에 따라 구분하여 박스에 정리했다면 응집도가 강하다고 할 수 있고, 구분없이 박스에 집어넣었다면 응집도가 약하다고 할 수 있습니다.

정보 은닉(Information Hiding)

정보 은닉은 한 모듈 내부에 포함된 절차와 자료들의 정보가 감추어져 다른 모듈이 접근하거나 변경하지 못하도록 하는 기법입니다.

우연적 응집도 (Coincidental Cohesion)	모듈 내부의 각 구성 요소들이 서로 관련 없는 요소로만 구성된 경우의 응집도

전문가의 조언

팬인, 팬아웃은 단순하게 생각하면 이해가 쉬워요. 모듈에 들어오면(In) 팬인, 모듈에서 나가면(out) 팬아웃입니다.

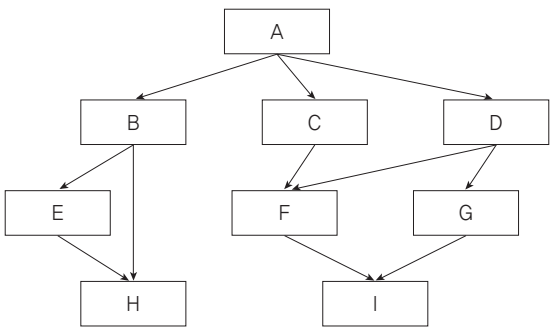
단일 장애점(SPOF, Single Point Of Failure)

단일 장애점은 시스템의 구성 요소 중 동작하지 않으면 전체 시스템이 중단되어 버리는 요소를 의미하며, 단일 실패점이라고도 합니다.

4 팬인(Fan-In) / 팬아웃(Fan-Out)

- 팬인은 어떤 모듈을 제어(호출)하는 모듈의 수를 나타낸다.
- 팬아웃은 어떤 모듈에 의해 제어(호출)되는 모듈의 수를 나타낸다.
- 팬인과 팬아웃을 분석하여 시스템의 복잡도를 알 수 있다.
- 팬인이 높다는 것은 재사용 측면에서 설계가 잘 되어있다고 볼 수 있으나, 단일 장애점*이 발생할 수 있으므로 중점적인 관리 및 테스트가 필요하다.
- 팬아웃이 높은 경우 불필요하게 다른 모듈을 호출하고 있는지 검토하고, 단순화시킬 수 있는지 여부에 대한 검토가 필요하다.
- 시스템의 복잡도를 최적화하려면 팬인은 높게, 팬아웃은 낮게 설계해야 한다.

예제 다음의 시스템 구조도에서 각 모듈의 팬인(Fan-In)과 팬아웃(Fan-Out)을 구하시오.



해설

- 팬인(Fan-In) : A는 0, B · C · D · E · G는 1, F · H · I는 2
- 팬아웃(Fan-Out) : H · I는 0, C · E · F · G는 1, B · D는 2, A는 3

기출문제 따라잡기 Section 023

이전기출

1. 시스템을 설계할 때 필요한 설계 지침으로 두 모듈 간의 상호 의존도 또는 두 모듈 사이의 연관 관계를 나타내는 것은?

① 결합도 ② 응집도
③ 신뢰도 ④ 종합도

여러 모듈 간의 상호 의존도를 나타내는 것은 결합도, 하나의 모듈 안에서 구성 요소 간의 관련 정도를 나타내는 것은 응집도입니다. 혼동하지 마세요.

이전기출

2. 모듈이 파라미터나 인수로 다른 모듈에게 데이터를 넘겨주고 호출 받은 모듈은 받은 데이터에 대한 처리 결과를 다시 돌려주는 유형의 모듈 결합도(Coupling)를 무엇이라고 하는가?

① 내용 결합도 ② 외부 결합도
③ 제어 결합도 ④ 데이터 결합도

모듈이 파리미터(매개변수)나 인수로 데이터를 주고받는 것은 데이터 결합도입니다.

이전기출

3. 결합도(Coupling)가 강한 순서대로 옳게 나열된 것은?

① 내용 결합도 〉 공통 결합도 〉 제어 결합도 〉 스탬프 결합도 〉 데이터 결합도

② 공통 결합도 〉 내용 결합도 〉 제어 결합도 〉 데이터 결합도 〉 스탬프 결합도

③ 데이터 결합도 〉 내용 결합도 〉 제어 결합도 〉 공통 결합도 〉 스탬프 결합도

④ 공통 결합도 〉 내용 결합도 〉 제어 결합도 〉 스탬프 결합도 〉 데이터 결합도

결합도가 강하다는 것은 서로 의존하는 정도가 높다는 것입니다. 즉 각 모듈들 간에 중요하거나 많은 부분을 함께 사용한다는 것을 의미합니다.

이전기출

4. 다음 설명의 ( ) 내용으로 옳은 것은?

( )는(은) 한 모듈 내부의 처리 요소들 간의 기능적 연관도를 나타내며, 모듈 내부 요소는 명령어, 명령어의 모임, 호출문, 특정 작업수행 코드 등이다.

① Validation ② Coupling

③ Interface ④ Cohesion

응집도(Cohesion)는 하나의 모듈을 구성하는 구성 요소 간의 관계, 결합도(Coupling)는 다른 모듈과의 의존도를 의미한다는 것! 기억하세요.

이전기출

5. 모듈의 구성 요소가 하나의 활동으로부터 나온 출력 자료를 그 다음 활동의 입력 자료로 사용하는 같은 모듈 내에서의 응집의 정도를 나타내는 것은?

① 절차적(Procedural) 응집도

② 논리적(Logical) 응집도

③ 기능적(Functional) 응집도

④ 순차적(Sequential) 응집도

절차적 응집도는 모듈 내부의 요소들이 여러 관련 기능이 있을 경우 순차적으로 수행될 때, 논리적 응집도는 유사한 성격을 갖거나 특정 형태로 분류되는 처리 요소들로 하나의 모듈이 형성될 때, 기능적 응집도는 모듈의 모든 기능 요소가 단일 문제와 연관될 때의 응집도를 의미합니다.

이전기출

6. 응집도가 강한 것부터 약한 순서로 옳게 나열된 것은?

① Sequential → Functional → Procedural → Coincidental → Logical

② Procedural → Coincidental → Functional → Sequential → Logical

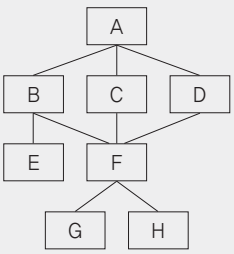
③ Functional → Sequential → Procedural → Logical → Coincidental

④ Logical → Coincidental → Functional → Sequential → Procedural

모듈 요소들 간의 관련성이 많으면 응집도가 높은 것입니다. 각 응집도에 대한 영문 표현도 정확히 기억해 두세요.

이전기출

7. 다음은 프로그램 구조를 나타낸다. 모듈 F에서의 Fan-In과 Fan-Out의 수는 얼마인가?



① Fan-In : 2, Fan-Out : 3

② Fan-In : 3, Fan-Out : 2

③ Fan-In : 1, Fan-Out : 2

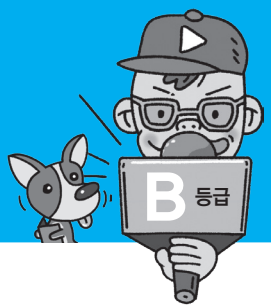
④ Fan-In : 2, Fan-Out : 1

Fan-In은 위에서 특정 모듈로 들어오는(In) 선(Line)의 수, Fan-Out은 특정 모듈에서 아래로 나가는(Out) 선(Line)의 수라고 생각하면 됩니다. 그러므로 F 모듈의 Fan-In은 3개(B, C, D), Fan-Out은 2개(G, H)입니다.

▶ 정답 : 1. ① 2. ④ 3. ① 4. ④ 5. ④ 6. ③ 7. ②

SECTION 024 ● LIVE

공통 모듈



전문가의 조언

공통 모듈과 재사용의 개념에 대해 정확히 이해하고, 효과적인 모듈 설계의 방안은 상식적인 내용이니 가볍게 읽어보고 넘어가세요.

전문가의 조언

인터넷 쇼핑몰 사이트의 '로그인' 기능이 공통 모듈의 대표적인 예입니다. 사이트의 첫 페이지에서 '로그인'을 할 수도 있고, 상품 구매 버튼을 누른 후에 나오는 '로그인' 창에서 할 수도 있습니다. 두개의 '로그인'은 발생하는 위치는 다르지만 동일한 기능을 갖고 있어 공통 모듈로 구성하기에 적합합니다.

1 공통 모듈의 개요

공통 모듈은 여러 프로그램에서 공통적으로 사용할 수 있는 모듈을 의미한다.

- 자주 사용되는 계산식이나 매번 필요한 사용자 인증과 같은 기능들이 공통 모듈로 구성될 수 있다.
- 모듈의 재사용성 확보와 중복 개발 회피를 위해 설계 과정에서 공통 부분을 식별하고 명세를 작성할 필요가 있다.
- 공통 모듈을 구현할 때는 다른 개발자들이 해당 기능을 명확히 이해할 수 있도록 다음의 명세 기법을 준수해야 한다.

정확성(Correctness)	시스템 구현 시 해당 기능이 필요하다는 것을 알 수 있도록 정확히 작성한다.
명확성(Clarity)	해당 기능을 이해할 때 중의적으로 해석되지 않도록 명확하게 작성한다.
완전성(Completeness)	시스템 구현을 위해 필요한 모든 것을 기술한다.
일관성(Consistency)	공통 기능들 간 상호 충돌이 발생하지 않도록 작성한다.
추적성(Traceability)	기능에 대한 요구사항의 출처, 관련 시스템 등의 관계를 파악할 수 있도록 작성한다.

2 재사용(Reuse)

재사용은 비용과 개발 시간을 절약하기 위해 이미 개발된 기능들을 파악하고 재구성하여 새로운 시스템 또는 기능 개발에 사용하기 적합하도록 최적화 시키는 작업이다.

- 재사용을 위해서는 누구나 이해할 수 있고 사용이 가능하도록 사용법을 공개해야 한다.
- 재사용되는 대상은 외부 모듈과의 결합도는 낮고, 응집도는 높아야 한다.
- **재사용 규모에 따른 분류**

함수와 객체*	클래스*나 메소드 단위의 소스 코드를 재사용한다.
컴포넌트*	컴포넌트 자체에 대한 수정 없이 인터페이스를 통해 통신하는 방식으로 재사용한다.
애플리케이션*	공통된 기능들을 제공하는 애플리케이션을 공유하는 방식으로 재사용한다.

- **함수=메소드** : 객체의 데이터를 처리하는 알고리즘
- **객체** : 데이터와 함수를 캡슐화한 소프트웨어 모듈
- **클래스** : 객체를 정의하는 틀
- **컴포넌트** : 하나 이상의 클래스로 작성되는 실행코드 기반의 모듈
- **애플리케이션** : 어떠한 목적을 갖고 개발된 소프트웨어

3 효과적인 모듈 설계 방안

- 결합도*는 줄이고 응집도*는 높여서 모듈의 독립성과 재사용성을 높인다.
- 모듈의 제어 영역* 안에서 그 모듈의 영향 영역*을 유지시킨다.
- 복잡도와 중복성을 줄이고 일관성을 유지시킨다.
- 모듈의 기능은 예측이 가능해야 하며 지나치게 제한적이어서는 안 된다.
- 유지보수가 용이해야 한다.
- 모듈 크기는 시스템의 전반적인 기능과 구조를 이해하기 쉬운 크기로 분해한다.
- 하나의 입구와 하나의 출구를 갖도록 해야 한다.
- 인덱스 번호나 기능 코드들이 전반적인 처리 논리 구조에 예기치 못한 영향을 끼치지 않도록 모듈 인터페이스를 설계해야 한다.

전문가의 조언

공통 모듈뿐만 아니라 일반 모듈 설계에 있어서도 동일한 내용입니다. 효과적인 모듈 설계는 결합도는 줄이고, 응집도를 높이는 것입니다.

결합도 / 응집도

- **결합도** : 모듈 간에 상호 의존하는 정도 또는 두 모듈 사이의 연관 관계
- **응집도** : 모듈의 내부 요소들의 서로 관련되어 있는 정도

모듈의 제어 / 영향 영역

- **모듈의 제어 영역** : 프로그램의 계층 구조 내에서 어떤 특정 모듈이 제어하는 하위 모듈
- **모듈의 영향 영역** : 특정 모듈이 다른 모듈들에게 미치게 되는 영향의 범위

기출문제 따라잡기 Section 024

이전기출

1. 효과적인 모듈 설계를 위한 유의사항으로 옳지 않은 것은?

① 모듈의 기능을 예측할 수 있도록 정의한다.
② 모듈은 단일 입구와 단일 출구를 갖도록 설계한다.
③ 결합도는 강하게, 응집도는 약하게 설계하여 모듈의 독립성을 확보할 수 있도록 한다.
④ 유지보수가 용이해야 한다.

효과적인 모듈 설계는 결합도는 약하게, 응집도는 강하게!

이전기출

2. 소프트웨어 재사용을 통한 장점이 아닌 것은?

① 개발 시간과 비용을 감소시킨다.
② 소프트웨어 품질을 향상시킨다.
③ 생산성을 증가시킨다.
④ 고급 프로그래머 배출이 용이하다.

이미 개발된 기능들을 가져다가 쓰는 것이 고급 프로그래머의 배출과 상관이 있을까요?

출제예상

3. 공통 모듈에 대한 설명으로 틀린 것은?

① 시스템 내에서 공통적으로 사용되는 모듈이다.
② 소프트웨어의 재사용성을 향상시키기 위해 구현할 필요가 있다.
③ 자주 사용되는 계산식이나 로그인 기능이 공통 모듈로 구성하기에 적합하다.
④ 외부 모듈과의 결합도가 높아야 한다.

공통 모듈의 재사용을 위해서는 외부 모듈과의 결합도는 낮고, 응집도는 높아야 합니다.

출제예상

4. 공통 모듈의 명세 기법 중 공통 기능들 간에 상호 충돌이 발생하지 않도록 작성해야 한다는 기법은?

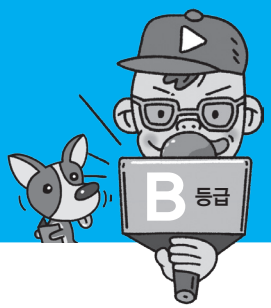
① Correctness
② Completeness
③ Consistency
④ Clarity

①번은 정확성, ②번은 완전성, ③번은 일관성, ④번은 명확성입니다.

▶ 정답 : 1. ③ 2. ④ 3. ④ 4. ③

SECTION
025
LIVE

코드



전문가의 조언

코드의 정의와 기능에 대해 간단히 기억하고, 코드의 종류와 코드 부여 체계에 대해 확실히 파악하고 넘어가세요.

1 코드(Code)의 개요

코드는 컴퓨터를 이용하여 자료를 처리하는 과정에서 분류 · 조합 및 집계를 용이하게 하고, 특정 자료의 추출을 쉽게 하기 위해서 사용하는 기호이다.

- 코드는 정보를 신속 · 정확 · 명료하게 전달할 수 있게 한다.
- 코드는 일정한 규칙에 따라 작성되며, 정보 처리의 효율과 처리된 정보의 가치에 많은 영향을 미친다.
- 일반적인 코드의 예로 주민등록번호, 학번, 전화번호 등이 있다.
- 코드의 주요 기능에는 식별 기능, 분류 기능, 배열 기능이 있다.

식별 기능	데이터의 간의 성격에 따라 구분이 가능하다.
분류 기능	특정 기준이나 동일한 유형에 해당하는 데이터를 그룹화 할 수 있다.
배열 기능	의미를 부여하여 나열할 수 있다.

2 코드의 종류

코드의 종류에는 다음과 같은 것들이 있다.

순차 코드 (Sequence Code)	자료의 발생 순서, 크기 순서 등 일정 기준에 따라서 최초의 자료부터 차례로 일련번호를 부여하는 방법으로, 순서 코드 또는 일련 번호 코드라고도 한다. 예 1, 2, 3, 4, …
블록 코드 (Block Code)	코드화 대상 항목 중에서 공통성이 있는 것끼리 블록으로 구분하고, 각 블록 내에서 일련번호를 부여하는 방법으로, 구분 코드라고도 한다. 예 1001~1100 : 총무부, 1101~1200 : 영업부
10진 코드 (Decimal Code)	코드화 대상 항목을 0~9까지 10진 분할하고, 다시 그 각각에 대하여 10진 분할하는 방법을 필요한 만큼 반복하는 방법으로, 도서 분류식 코드라고도 한다. 예 1000 : 공학, 1100 : 소프트웨어 공학, 1110 : 소프트웨어 설계
그룹 분류 코드 (Group Classification Code)	코드화 대상 항목을 일정 기준에 따라 대분류, 중분류, 소분류 등으로 구분하고, 각 그룹 안에서 일련번호를 부여하는 방법이다. 예 1–01–001 : 본사–총무부–인사계, 2–01–001 : 지사–총무부–인사계
연상 코드 (Mnemonic Code)	코드화 대상 항목의 명칭이나 약호*와 관계있는 숫자나 문자, 기호를 이용하여 코드를 부여하는 방법이다. 예 TV–40 : 40인치 TV, L–15–220 : 15W 220V의 램프
표의 숫자 코드 (Significant Digit Code)	코드화 대상 항목의 성질, 즉 길이, 넓이, 부피, 지름, 높이 등의 물리적 수치를 그대로 코드에 적용시키는 방법으로, 유효 숫자 코드라고도 한다. 예 120–720–1500 : 두께×폭×길이가 120×720×1500인 강판

약호
약호는 간단하고 알기 쉽게 만든 부호를 의미합니다.

합성 코드 **(Combined Code)**	필요한 기능을 하나의 코드로 수행하기 어려운 경우 2개 이상의 코드를 조합하여 만드는 방법이다. 예 연상 코드 + 순차 코드 KE–711 : 대한항공 711기, AC–253 : 에어캐나다 253기

3 코드 부여 체계

코드 부여 체계는 이름만으로 개체*의 용도와 적용 범위를 알 수 있도록 코드를 부여하는 방식을 말한다.

- 코드 부여 체계는 각 개체에 유일한 코드를 부여하여 개체들의 식별 및 추출을 용이하게 한다.
- 코드를 부여하기 전에 각 단위 시스템의 고유한 코드와 개체를 나타내는 코드 등이 정의되어야 한다.
- 코드 부여 체계를 담당하는 자는 코드의 자릿수와 구분자, 구조 등을 상세하게 명시해야 한다.

예 모듈 식별을 위한 코드 부여 체계

자리수	구분자를 포함한 11자리
기본 구조	AAA–MOD–000
상세 구조	AAA • 영문 및 숫자 3자리 • 단위 시스템의 코드 3자리 • 전체 시스템의 경우 'PJC' 고정 MOD • 영문 3자리 • 모듈은 MOD, 공통 모듈은 COM을 사용 000 • 숫자 3자리 • 순차적 일련번호 001~999

예시

- PJC–COM–003 : 전체 시스템 단위의 3번째 공통 모듈
- PY3–MOD–010 : PY3이라는 단위 시스템의 10번째 모듈

개체
소프트웨어 개발에서 코드를 부여할 대상이 되는 개체에는 모듈, 컴포넌트, 인터페이스 등이 있습니다.

기출문제 따라잡기 Section 025

이전기출

1. 코드의 주요 기능으로 거리가 먼 것은?

① 식별 기능 ② 분류 기능
③ 배열 기능 ④ 호환 기능

코드의 기능을 다시 한 번 짚어봅시다. 식별, 분류, 배열 기능!

이전기출

2. 사원 번호의 발급 과정에서 둘 이상의 서로 다른 사람에게 동일한 번호가 부여된 경우에 코드의 어떤 기능을 만족시키지 못한 것인가?

① 표준화 기능 ② 식별 기능
③ 배열 기능 ④ 연상 기능

동일한 번호가 부여되면 식별할 수 있을까요?

이전기출

3. 회사에서 각 부서의 명칭을 코드화하기 위하여 대분류, 중분류, 소분류 등으로 나누어 나타내고자 한다. 이 때 가장 적합한 코드의 종류는?

① 구분 코드(Block Code)
② 그룹 분류 코드(Group Classification Code)
③ 연상 기호 코드(Mnemonic Code)
④ 순차 코드(Sequence Code)

대분류, 중분류, 소분류로 나누어 코드를 분류하는 방법은 뭘까요? 그룹으로 분류하고 있네요!

이전기출

4. 코드화 대상 항목의 길이, 넓이, 부피, 무게 등을 나타내는 문자, 숫자 혹은 기호를 그대로 코드로 사용하는 코드는?

① 그룹 분류식 코드(Group Classification Code)
② 기호 코드(Mnemonic Code)
③ 표의 숫자 코드(Significant Digit Code)
④ 합성 코드(Combined Code)

표의 숫자 코드는 대상체의 성질을 그대로 표시하므로 기억하기 쉽습니다.

이전기출

5. 코드화 대상 자료 전체를 계산하여 이를 필요로 하는 분류 단위로 블록을 구분하고, 각 블록 내에서 순서대로 번호를 부여하는 방식으로, 적은 자릿수로 많은 항목의 표시가 가능하고 예비 코드를 사용할 수 있어 추가가 용이하다. 구분 순차 코드라고도 하는 이것을 무엇이라 하는가?

① 순차(Sequence) 코드
② 표의 숫자(Significant Digit) 코드
③ 블록(Block) 코드
④ 연상(Mnemonic) 코드

문제에서 핵심 단어를 찾아보세요. '블록을 구분~', '블록 내~' 어느 코드를 의미하는지를 찾을 수 있겠죠!

출제예상

6. 코드 부여 체계에 대한 설명으로 옳지 않은 것은?

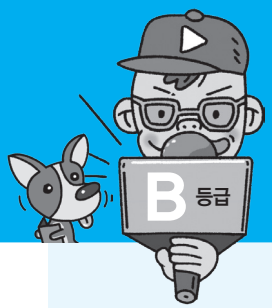
① 모듈이나 컴포넌트에 식별할 수 있는 코드를 부여하는 것을 말한다.
② 프로그래머가 모듈을 개발할 때 마다 임의로 코드를 부여한다.
③ 하나 이상의 코드를 조합하여 사용한다.
④ 코드 부여 체계의 담당자는 코드 규칙을 상세히 정의해야 한다.

코드 부여 체계는 모듈이나 컴포넌트 등에 코드를 부여하는 방식을 의미하기 때문에 코드는 프로그래머가 임의로 부여하는 것이 아닌 코드 부여 체계에 맞게 부여해야 합니다.

▶ 정답 : 1. ④ 2. ② 3. ② 4. ③ 5. ③ 6. ②

SECTION 026 ● LIVE

디자인 패턴



1 디자인 패턴(Design Pattern)의 개요

디자인 패턴은 각 모듈의 세분화된 역할이나 모듈들 간의 인터페이스와 같은 코드를 작성하는 수준의 세부적인 구현 방안을 설계할 때 참조할 수 있는 전형적인 해결 방식 또는 예제를 의미한다.

- 디자인 패턴은 재사용할 수 있는 기본형 코드들이 포함되어 있다.
- '바퀴를 다시 발명하지 마라(Don't reinvent the wheel)'*라는 말과 같이, 개발 과정 중에 문제가 발생하면 새로 해결책을 구상하는 것보다 문제에 해당하는 디자인 패턴을 참고하여 적용하는 것이 더 효율적이다.
- 디자인 패턴은 한 패턴에 변형을 가하거나 특정 요구사항을 반영하면 유사한 형태의 다른 패턴으로 변화되는 특징*이 있다.
- 디자인 패턴은 1995년 GoF(Gang of Four)라고 불리는 에릭 감마(Erich Gamma), 리차드 헬름(Richard Helm), 랄프 존슨(Ralph Johnson), 존 블리시디스(John Vissides)가 처음으로 구체화 및 체계화하였다.
- GoF의 디자인 패턴은 수많은 디자인 패턴들 중 가장 일반적인 사례에 적용될 수 있는 패턴들을 분류하여 정리함으로써, 지금까지도 소프트웨어 공학이나 현업에서 가장 많이 사용되는 디자인 패턴이다.
- GoF의 디자인 패턴은 유형에 따라 생성 패턴 5개, 구조 패턴 7개, 행위 패턴 11개 총 23개의 패턴으로 구성된다.

참깐만요

아키텍처 패턴 vs 디자인 패턴

아키텍처 패턴과 디자인 패턴은 모두 소프트웨어 설계를 위한 참조 모델이지만 다음과 같은 차이가 있습니다.

- 아키텍처 패턴은 디자인 패턴보다 상위 수준의 설계에 사용됩니다.
- 아키텍처 패턴이 전체 시스템의 구조를 설계하기 위한 참조 모델이라면, 디자인 패턴은 서브시스템에 속하는 컴포넌트*들과 그 관계를 설계하기 위한 참조 모델입니다.
- 몇몇 디자인 패턴은 특정 아키텍처 패턴을 구현하는데 유용하게 사용됩니다.

2 생성 패턴(Creational Pattern)

생성 패턴은 객체의 생성과 관련된 패턴으로 총 5개의 패턴이 있다.

- 생성 패턴은 객체의 생성과 참조 과정을 캡슐화 하여 객체가 생성되거나 변경되어도 프로그램의 구조에 영향을 크게 받지 않도록 하여 프로그램에 유연성을 더해준다.

전문가의 조언

디자인 패턴이 무엇인지, 아키텍처 패턴과는 어떤 차이가 있는지 확인해 보세요. 그리고 유형별로 어떤 디자인 패턴들이 있고 각 디자인 패턴들의 대표적인 특징은 무엇인지 잘 알아두세요.

전문가의 조언

아키텍처 패턴에서 건물의 윤곽을 잡는 가이드라인을 제시했다면, 디자인 패턴은 그보다 더 세밀한 부분인 건물의 각 방들을 인테리어 하는 과정이라고 보면 됩니다.

바퀴를 다시 발명하지 마라

'바퀴를 다시 발명하지 마라'는 이미 존재하는 기술이나 제품을 굳이 다시 만들기 위해 시간과 노동력을 소모하지 말라는 의미의 관용구입니다.

디자인 패턴이 변화되는 특징

디자인 패턴이 변화되는 특징은 건축과 비교하면 이해가 쉽습니다. 예를 들어 설계자가 처음에는 '원룸'이라는 패턴을 적용하여 설계하였으나, '주방을 분리'하라는 요청이 있어 이를 반영하고 보니 '투룸'이라는 유사한 형태의 다른 패턴이 되어버린 것과 비교할 수 있습니다.

컴포넌트(Component)

컴포넌트는 독립적인 업무 또는 기능을 수행하는 실행 코드 기반으로 작성된 모듈입니다.

전문가의 조언

각 패턴의 이름과 의미를 연결 지어 기억해 두세요. 추상 팩토리는 서로 다른 부품을 조립만 하는 조립 공장(Factory), 빌더는 건축가(Builder)가 블록을 조립하는 모습, 팩토리 메소드는 부품부터 완성품까지 통째로 찍어내는 공장(Factory), 프로토타입은 원형(Prototype)을 두고 복제품을 만드는 것, 싱글톤은 식당에서 누구나 사용할 수 있지만 하나뿐(Singleton)인 정수기를 염두에 두고 기억해 보세요.

인스턴스(Instance)
인스턴스는 클래스에 속한 각각의 객체를 의미합니다.

추상 팩토리 (Abstract Factory)	• 구체적인 클래스에 의존하지 않고, 인터페이스를 통해 서로 연관 · 의존하는 객체들의 그룹으로 생성하여 추상적으로 표현한다. • 연관된 서브 클래스를 묶어 한 번에 교체하는 것이 가능하다.
빌더(Builder)	• 작게 분리된 인스턴스*를 건축 하듯이 조합하여 객체를 생성한다. • 객체의 생성 과정과 표현 방법을 분리하고 있어, 동일한 객체 생성에서도 서로 다른 결과를 만들어 낼 수 있다.
팩토리 메소드 (Factory Method)	• 객체 생성을 서브 클래스에서 처리하도록 분리하여 캡슐화한 패턴이다. • 상위 클래스에서 인터페이스만 정의하고 실제 생성은 서브 클래스가 담당한다.
프로토타입(Prototype)	• 원본 객체를 복제하는 방법으로 객체를 생성하는 패턴이다. • 일반적인 방법으로 객체를 생성하며, 비용이 큰 경우 주로 이용한다.
싱글톤(Singleton)	• 하나의 객체를 생성하면 생성된 객체를 어디서든 참조할 수 있지만, 여러 프로세스가 동시에 참조할 수는 없다. • 클래스 내에서 인스턴스가 하나뿐임을 보장하며, 불필요한 메모리 낭비를 최소화 할 수 있다.

3 구조 패턴(Structural Pattern)

구조 패턴은 클래스나 객체들을 조합하여 더 큰 구조로 만들 수 있게 해주는 패턴으로 총 7개의 패턴이 있다.

- 구조 패턴은 구조가 복잡한 시스템을 개발하기 쉽게 도와준다.

전문가의 조언

어댑터는 전압을 맞춰주는 변압기(Adapter), 브리지는 두 섬을 연결하는 다리(Bridge), 컴포지트는 폴더와 파일을 합성(Composite)한 것, 데코레이터는 온갖 것으로 장식된(Decorator) 눈사람, 퍼싸드는 외부(Facade)의 리모컨 버튼만으로 복잡한 명령들을 간편하게 수행하는 것, 플라이웨이트는 부담을 가볍게(Flyweight) 하기 위해 물품을 공유하는 것, 프록시는 내가 하기 어려운 법률업무를 대리(Proxy)해서 처리해주는 변호사라고 생각하면서 암기해 보세요.

어댑터(Adapter)	• 호환성이 없는 클래스들의 인터페이스를 다른 클래스가 이용할 수 있도록 변환해주는 패턴이다. • 기존의 클래스를 이용하고 싶지만 인터페이스가 일치하지 않을 때 이용한다.
브리지(Bridge)	• 구현부에서 추상층을 분리하여, 서로가 독립적으로 확장할 수 있도록 구성한 패턴이다. • 기능과 구현을 두 개의 별도 클래스로 구현한다.
컴포지트(Composite)	• 여러 객체를 가진 복합 객체와 단일 객체를 구분 없이 다루고자 할 때 사용하는 패턴이다. • 객체들을 트리 구조로 구성하여 디렉터리 안에 디렉터리가 있듯이 복합 객체 안에 복합 객체가 포함되는 구조를 구현할 수 있다.
데코레이터(Decorator)	• 객체 간의 결합을 통해 능동적으로 기능들을 확장할 수 있는 패턴이다. • 임의의 객체에 부가적인 기능을 추가하기 위해 다른 객체들을 덧붙이는 방식으로 구현한다.
퍼싸드(Facade)	• 복잡한 서브 클래스들을 피해 더 상위에 인터페이스를 구성함으로써 서브 클래스들의 기능을 간편하게 사용할 수 있도록 하는 패턴이다. • 서브 클래스들 사이의 통합 인터페이스를 제공하는 Wrapper 객체가 필요하다.
플라이웨이트 (Flyweight)	• 인스턴스가 필요할 때마다 매번 생성하는 것이 아니고 가능한 한 공유해서 사용함으로써 메모리를 절약하는 패턴이다. • 다수의 유사 객체를 생성하거나 조작할 때 유용하게 사용할 수 있다.
프록시(Proxy)	• 접근이 어려운 객체와 여기에 연결하려는 객체 사이에서 인터페이스 역할을 수행하는 패턴이다. • 네트워크 연결, 메모리의 대용량 객체로의 접근 등에 주로 이용한다.

4 행위 패턴(Behavioral Pattern)

행위 패턴은 클래스나 객체들이 서로 상호작용하는 방법이나 책임 분배 방법을 정의하는 패턴으로 총 11개의 패턴이 있다.

- 행위 패턴은 하나의 객체로 수행할 수 없는 작업을 여러 객체로 분배하면서 결합도를 최소화 할 수 있도록 도와준다.

책임 연쇄 (Chain of Responsibility)	• 요청을 처리할 수 있는 객체가 둘 이상 존재하여 한 객체가 처리하지 못하면 다음 객체로 넘어가는 형태의 패턴이다. • 요청을 처리할 수 있는 각 객체들이 고리(Chain)로 묶여 있어 요청이 해결될 때까지 고리를 따라 책임이 넘어간다.
커맨드(Command)	• 요청을 객체의 형태로 캡슐화하여 재이용하거나 취소할 수 있도록 요청에 필요한 정보를 저장하거나 로그에 남기는 패턴이다. • 요청에 사용되는 각종 명령어들을 추상 클래스*와 구체 클래스*로 분리하여 단순화한다.
인터프리터(Interpreter)	• 언어에 문법 표현을 정의하는 패턴이다. • SQL이나 통신 프로토콜과 같은 것을 개발할 때 사용한다.
반복자(Iterator)	• 자료 구조와 같이 접근이 잦은 객체에 대해 동일한 인터페이스를 사용하도록 하는 패턴이다. • 내부 표현 방법의 노출 없이 순차적인 접근이 가능하다.
중재자(Mediator)	• 수많은 객체들 간의 복잡한 상호작용(Interface)을 캡슐화하여 객체로 정의하는 패턴이다. • 객체 사이의 의존성을 줄여 결합도를 감소시킬 수 있다.
메멘토(Memento)	• 특정 시점에서의 객체 내부 상태를 객체화함으로써 이후 요청에 따라 객체를 해당 시점의 상태로 돌릴 수 있는 기능을 제공하는 패턴이다. • Ctrl+Z와 같은 되돌리기 기능을 개발할 때 주로 이용한다.
옵서버(Observer)	• 한 객체의 상태가 변화하면 객체에 상속되어 있는 다른 객체들에게 변화된 상태를 전달하는 패턴이다. • 주로 분산된 시스템 간에 이벤트를 생성 · 발행(Publish)하고, 이를 수신(Subscribe)해야 할 때 이용한다.
상태(State)	• 객체의 상태에 따라 동일한 동작을 다르게 처리해야 할 때 사용하는 패턴이다. • 객체 상태를 캡슐화하고 이를 참조하는 방식으로 처리한다.
전략(Strategy)	• 동일한 계열의 알고리즘들을 개별적으로 캡슐화하여 상호 교환할 수 있게 정의하는 패턴이다. • 클라이언트는 독립적으로 원하는 알고리즘을 선택하여 사용할 수 있으며, 클라이언트에 영향 없이 알고리즘의 변경이 가능하다.
템플릿 메소드 (Template Method)	• 상위 클래스에서 골격을 정의하고, 하위 클래스에서 세부 처리를 구체화하는 구조의 패턴이다. • 유사한 서브 클래스를 묶어 공통된 내용을 상위 클래스에서 정의함으로써 코드의 양을 줄이고 유지보수를 용이하게 해준다.
방문자(Visitor)	• 각 클래스들의 데이터 구조에서 처리 기능을 분리하여 별도의 클래스로 구성하는 패턴이다. • 분리된 처리 기능은 각 클래스를 방문(Visit)하여 수행한다.

전문가의 조언

책임 연쇄는 위에서 쏟아지는 물을 여러 물받이가 연속(Chain)해서 나눠 받는(Responsibility) 물레방아, 커맨드는 각종 명령어(Command)를 하나로 합쳐둔 것, 인터프리터는 언어 번역가(Interpreter), 반복자는 음악파일의 다음 곡 재생처럼 같은 명령의 반복(Iterator), 중재자는 물품 매매를 중개해주는(Mediator) 인터넷 사이트, 메멘토는 기억 속의 그 때(Memento)로 돌아가는 것, 옵서버는 변화를 지켜보고(Observer) 알려주는 것, 상태는 환자의 상태(State)에 따라 치료방법이 다른 것, 전략은 여러 전략들을 A · b · c 등으로 정하고 필요할 때 원하는 전략(Strategy)을 선택하여 쓰는 것, 템플릿 메소드는 세모 · 네모 · 동그라미를 그리는 방법(Method)들을 도형이라는 하나의 큰 틀(Template)로 묶는 것, 비지터는 책을 만들기 위해 저자 · 편집자 · 홍보팀을 번갈아가며 방문자(Visitor)하는 것과 비교할 수 있습니다.

추상 클래스(Abstract Class)

추상 클래스는 구체 클래스에서 구현하려는 기능들의 공통점만을 모아 추상화한 클래스로, 인스턴스 생성이 불가능하여 구체 클래스가 추상 클래스를 상속받아 구체화한 후 구체 클래스의 인스턴스를 생성하는 방식으로 사용합니다.

구체 클래스(Concrete Class)

구체 클래스는 인스턴스 생성이 가능한 일반적인 클래스를 의미하는 용어로, 추상 클래스와 구분하기 위해 사용됩니다. 구상 클래스 또는 구현 클래스라고도 합니다.

출제예상
1. 〈보기1〉의 디자인 패턴 분류와 〈보기2〉의 디자인 패턴을 바르게 연결한 것은?

〈보기1〉

ㄱ. 생성 패턴	ㄴ. 구조 패턴	ㄷ. 행위 패턴

〈보기2〉

a. Prototype	b. Facade	c. Command

① ㄱ–a, ㄴ–b, ㄷ–c ② ㄱ–b, ㄴ–a, ㄷ–c
③ ㄱ–c, ㄴ–a, ㄷ–b ④ ㄱ–a, ㄴ–c, ㄷ–b

프로토타입(Prototype)은 객체를 생성하는 패턴이고, 퍼싸드(Facade)는 복잡한 구조를 간편하게 해주는 패턴, 커맨드(Command)는 요청이 들어왔을 때 이를 처리하는 방법을 구현한 패턴입니다.

출제예상
2. Iterator 패턴에 대한 설명으로 옳지 않은 것은?

① 내부 표현 방법의 노출 없이 접근이 가능하다.
② 클라이언트는 독립적으로 원하는 알고리즘을 선택하여 사용할 수 있다.
③ 배열이나 리스트 같은 자료 구조를 처리하는 데 사용된다.
④ 집합 객체의 각 요소들에 대해 순차 접근하는 방식이다.

반복자(Iterator) 패턴은 음악을 들을 때 누르는 '다음 곡 재생' 버튼을 생각하세요!

출제예상
3. 다음에서 설명하는 디자인 패턴에 해당하는 것은?

> 기존에 만들어진 클래스를 사용하려고 하는데 인터페이스가 일치하지 않거나, 관련이 없어 사용하지 못했던 클래스들을 다시 사용할 수 있게 만들고자 한다.

① Command ② Factory Method
③ Adapter ④ Memento

호환성이 없는 클래스의 인터페이스를 다른 클래스가 이용할 수 있도록 변환 해주는 패턴은 어댑터(Adapter)입니다.

출제예상
4. 다음 중 디자인 패턴에 대한 설명으로 가장 거리가 먼 것은?

① 객체의 생성과 표현이 분리되어 있어, 동일한 객체 생성에도 다른 결과를 내야 할 때는 빌더 패턴이 권장된다.
② 메모리를 절약하기 위해 인스턴스를 공유해서 사용할 때는 플라이웨이트 패턴이 권장된다.
③ 관련된 객체들의 묶음을 스타일 선택에 의해서 생성할 때는 프로토타입 패턴이 권장된다.
④ 각 클래스의 처리 기능을 분리하여 캡슐화 한 후 필요할 때마다 해당 클래스에 방문해서 처리할 때는 비지터 패턴이 권장된다.

프로토타입 패턴은 원본 객체를 복제하는 방법으로 객체를 생성하는 패턴입니다.

출제예상
5. 다음 중 유형이 다른 디자인 패턴끼리 연결된 것은?

① Builder Pattern – Prototype Pattern
② Singleton Pattern – Abstract Factory Pattern
③ Bridge Pattern – Proxy Pattern
④ Chain of Responsibility Pattern – Facade Pattern

책임 연쇄 패턴(Chain of Responsibility Pattern)은 행위 패턴에 속하고, 퍼싸드 패턴(Facade Pattern)은 구조 패턴에 속합니다.

출제예상
6. 다음 설명에 해당하는 디자인 패턴은?

문제	Java에서 new를 통해 인스턴스를 매번 생성하다 보니 메모리 사용량이 급격하게 증가하였다. 이로 인해 프로그램이 무거워져서 원활한 사용을 위해 수정이 불가피해졌다. 어떻게 해결해야 하는가?
해결책	동일한 객체가 존재할 경우 해당 객체를 공유하여 사용함으로써 메모리를 절약하고, 존재하지 않을 경우만 새로 객체를 생성하여 사용하도록 한다.

① Adapter 패턴
② Bridge 패턴
③ Flyweight 패턴
④ Chain of Responsibility 패턴

'공유', '메모리 절약'이라는 단어가 이 문제를 해결하기 위한 핵심 키워드 입니다.

▶ 정답 : 1. ① 2. ② 3. ③ 4. ③ 5. ④ 6. ③

예·상·문·제·은·행

해설은 127쪽에 있습니다.

1. 소프트웨어 아키텍처 설계에 있어서 정보 은폐(Information Hiding)의 근본적인 목적은?

① 코드를 개선하기 위하여
② 코드의 길이를 짧게 하기 위하여
③ 고려되지 않은 영향(Side Effect)들을 최소화하기 위하여
④ 인터페이스를 최소화하기 위하여

2. 추상화의 유형에서 이벤트 발생의 구체적인 절차 또는 방법을 정의하지 않고 대표할 수 있는 표현으로 대체하는 것을 무엇이라고 하는가?

① 과정 추상화 ② 데이터 추상화
③ 제어 추상화 ④ 이벤트 추상화

3. 소프트웨어 아키텍처 설계에 대한 설명으로 옳지 않은 것은?

① 이해 관계자들의 의사소통 도구로 활용된다.
② 설계된 모듈을 프로그래밍 언어를 통해 구현한다.
③ 애플리케이션을 모듈로 분할하고, 모듈 간 인터페이스를 결정하는 과정이다.
④ 기본 원리에는 모듈화, 추상화, 단계적 분해, 정보은닉이 있다.

4. 소프트웨어 아키텍처의 품질 속성 중 다음 설명이 의미하는 것은?

> • 개발 비용을 더 투자하여 유연성이 높은 아키텍처를 만들 것인지를 결정하는 것이다.
> • 유연성이 떨어지는 경우 이후 유지보수에 많은 비용이 소모될 수 있다는 것을 고려해야 한다.

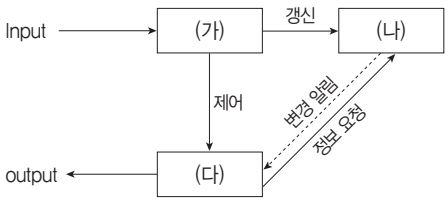
① 비용과 혜택 ② 시장 적시성
③ 가용성 ④ 구축 가능성

5. 소프트웨어 아키텍처의 설계 과정을 순서대로 가장 옳게 나열한 것은?

> 가. 시스템과 서브시스템의 타입을 결정한다.
> 나. 서브시스템의 기능과 서브시스템 간의 인터페이스를 정의한다.
> 다. 표준 아키텍처를 설계한다.
> 라. 요구사항을 분석하여 전체 시스템의 설계 목표를 설정한다.

① 라 → 가 → 나 → 다 ② 라 → 나 → 가 → 다
③ 라 → 다 → 가 → 나 ④ 라 → 가 → 다 → 나

6. 다음 그림은 MVC(Model-View-Controller) 아키텍처 패턴을 나타낸 것이며, 세 가지 요소가 별도의 컴포넌트로 구성된다. 각 (가), (나), (다)에 해당되는 요소를 바르게 나열한 것은?



① (가) 컨트롤러, (나) 모델, (다) 뷰
② (가) 모델, (나) 뷰, (다) 컨트롤러
③ (가) 뷰, (나) 모델, (다) 컨트롤러
④ (가) 컨트롤러, (나) 뷰, (다) 모델

7. 아키텍처의 패턴에 관한 설명 중 가장 옳지 않은 것은?

① 레이어 패턴은 하위 계층이 제공하는 서비스를 상위 계층의 서브시스템이 사용하도록 구성된다.
② 브로커 패턴은 사용자가 원하는 서비스와 특성을 요청하면, 적합한 컴포넌트를 연결해준다.
③ 이벤트-버스 패턴은 이벤트 메시지를 발행하고 구독하는 형태로 이루어진다.
④ 피어-투-피어 패턴에서 각 서브시스템은 서비스를 요청하고 제공하는 능력이 있다.

8. 다음은 아키텍처 스타일과 이를 기반으로 하는 시스템의 관계를 짝지은 것으로 가장 옳지 않은 것은?

① Layer Pattern – OSI 참조 모델
② Master-Slave Pattern – 장애 허용 시스템
③ Blackboard Pattern – 차량 식별 시스템
④ MVC Pattern – 병렬 컴퓨팅 시스템

9. 아키텍처 패턴(Architecture Pattern)의 장점에 해당하지 않는 것은?

① 시행착오를 줄여 개발시간이 최대화되고, 고품질의 소프트웨어를 생산할 수 있다.
② 전형적인 아키텍처를 기반으로 이해를 공유하여 개발자들 간의 의사소통이 간편해진다.
③ 개발에 참여하지 않아도 시스템의 이해가 쉬어 유지보수가 용이하다.
④ 시스템 개발 전에 시스템의 특성을 예측할 수 있다.

▶ **정답 :** 1. ③ 2. ③ 3. ② 4. ① 5. ④ 6. ① 7. ① 8. ④ 9. ①

예·상·문·제·은·행

10. 다음이 설명하고 있는 아키텍처 패턴은?

- 데이터 변환 시스템과 같이 데이터 스트림을 처리해야 하는 소프트웨어에서 주로 사용된다.
- 한 모듈이 데이터 스트림을 입력받아 처리하면, 다음 모듈이 결과를 이어받아 처리하는 형식이다.
- 모듈의 재사용과 추가가 간편하여 확장성이 뛰어나다.

① MVC Pattern
② Pipe-Filter Pattern
③ Peer-To-Peer Pattern
④ Broker Pattern

11. 객체지향 프로그램의 장점으로 거리가 먼 것은?
① 자연적인 모델링이 가능하다.
② 실행 속도가 빨라진다.
③ 소프트웨어의 재사용률이 높아진다.
④ 소프트웨어의 유지보수성이 향상된다.

12. 객체에 대한 특성을 설명한 것으로 가장 옳지 않은 것은?
① 객체마다 각각의 상태를 갖고 있다.
② 식별성을 가진다.
③ 행위에 대하여 그 특징을 나타낼 수 있다.
④ 일정한 기억장소를 가지고 있지 않다.

13. 객체지향 개념 중 하나 이상의 유사한 객체들을 묶어 공통된 특성을 표현한 데이터 추상화를 의미하는 것은?
① 메소드(Method) ② 클래스(Class)
③ 상속성(Inheritance) ④ 추상화(Abstraction)

14. 객체지향 개념에서 연관된 데이터와 함수를 함께 묶어 외부와 경계를 만들고 필요한 인터페이스만을 밖으로 드러내는 과정을 무엇이라고 하는가?
① 정보 은닉(Information Hiding)
② 클래스(Class)
③ 캡슐화(Encapsulation)
④ 통합(Integration)

15. 객체지향 기법에서 상속(Inheritance)의 결과로 얻을 수 있는 가장 주요한 이점은?
① 모듈 라이브러리의 재이용
② 객체지향 DB를 사용할 수 있는 능력
③ 클래스와 오브젝트들을 재사용할 수 있는 능력
④ 프로젝트들을 보다 효과적으로 관리할 수 있는 능력

16. ASCII File을 print하는 method를 갖고 있는 object, Binary File을 print하는 method를 갖고 있는 object, Picture File을 print하는 method를 갖고 있는 object들은 모두 "print"라는 method를 갖고 있으므로, "print"란 메시지를 받으면 수행하게 된다. 그러나 각각의 method에서 print를 수행하는 방법은 모두 다를 것이다. 객체지향 시스템에서 이와 같이 서로 다른 Class들이 같은 의미의 응답을 하는 특성을 무엇이라고 하는가?
① 캡슐화(Encapsulation)
② 상속성(Inheritance)
③ 다형성(Polymorphism)
④ 추상화(Abstraction)

17. 한 모듈이 다른 모듈의 내부 기능 및 그 내부 자료를 참조하는 경우, 이를 무슨 결합이라고 하는가?
① 내용 결합 ② 제어 결합
③ 공통 결합 ④ 스탬프 결합

18. 두 모듈이 동일한 자료 구조를 조회하는 경우의 결합성이며 자료 구조의 어떠한 변화, 즉 포맷이나 구조의 변화는 그것을 조회하는 모든 모듈 및 변화되는 필드를 실제로 조회하지 않는 모듈에까지도 영향을 미치게 되는 결합성은?
① Data Coupling ② Stamp Coupling
③ Control Coupling ④ Content Coupling

19. 결합도(Coupling)에 대한 설명으로 틀린 것은?
① 데이터 결합도(Data Coupling)는 두 모듈이 매개변수로 자료를 전달할 때 자료 구조 형태로 전달되어 이용될 때 데이터가 결합되어 있다고 한다.
② 내용 결합도(Content Coupling)는 하나의 모듈이 직접적으로 다른 모듈의 내용을 참조할 때 두 모듈은 내용적으로 결합되어 있다고 한다.
③ 공통 결합도(Common Coupling)는 두 모듈이 동일한 전역 데이터를 접근한다면 공통 결합되어 있다고 한다.
④ 결합도(Coupling)는 두 모듈간의 상호작용, 또는 의존도 정도를 나타내는 것이다.

20. 모듈의 응집도(Cohesion)에 대한 설명 중 틀린 것은?
① 모듈의 응집도란 모듈 안의 요소들이 서로 관련되어 있는 정도를 말한다.
② 기능적 응집도(Functional Cohesion)는 한 모듈 내부의 한 기능 요소에 의한 출력 자료가 다음 기능 요소의 입력 자료로서 제공되는 형태이다.

해설은 127쪽에 있습니다.

③ 교환적 응집도(Communication Cohesion)는 동일한 입력과 출력을 사용하는 소작업들이 모인 모듈에서 볼 수 있다.

④ 논리적 응집도(Logical Cohesion)는 유사한 성격을 갖거나 특정 형태로 분류되는 처리 요소들로 하나의 모듈이 형성되는 경우이다.

21. 응집도의 종류 중 서로 간에 어떠한 의미 있는 연관관계도 지니지 않은 기능 요소로 구성되는 경우이며, 서로 다른 상위 모듈에 의해 호출되어 처리상의 연관성이 없는 서로 다른 기능을 수행하는 경우의 응집도는?

① Coincidental Cohesion

② Functional Cohesion

③ Sequential Cohesion

④ Logical Cohesion

22. 모듈을 설계하기 위해서 바람직한 응집도(Cohesion)와 결합도(Coupling)의 관계는?

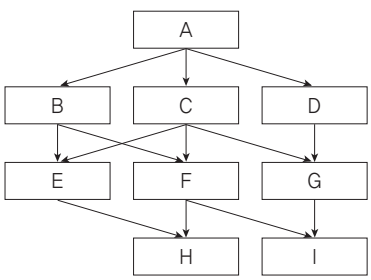
① 응집도는 약하고 결합도는 강해야 한다.

② 응집도는 강하고 결합도는 약해야 한다.

③ 응집도도 약하고 결합도도 약해야 한다.

④ 응집도도 강하고 결합도도 강해야 한다.

23. 다음은 소프트웨어의 구성 요소인 모듈의 계층적 구성을 나타내는 프로그램 구조도이다. 모듈 G에서의 팬인(Fan In)과 팬아웃(Fan Out)은?



① 팬인 : 1, 팬아웃 : 2 ② 팬인 : 2, 팬아웃 : 1

③ 팬인 : 2, 팬아웃 : 3 ④ 팬인 : 3, 팬아웃 : 1

24. 효과적인 모듈 설계 방법으로 가장 거리가 먼 것은?

① Coupling은 약하게 Cohesion는 강하게 설계한다.

② Complexity는 줄이고 Redundancy를 최대한 늘릴 수 있도록 설계한다.

③ Maintenance가 용이하도록 설계한다.

④ Module 크기는 시스템의 전반적인 기능과 구조를 이해하기 쉬운 크기로 설계한다.

25. 소프트웨어 재사용에 대한 내용 중 옳지 않은 것은?

① 비용을 절감하고 개발 시간을 단축하여 생산성이 증가한다.

② 재사용되는 대상은 외부 모듈과의 결합도가 낮아야 한다.

③ 규모에 따라 변수, 함수, 객체, 컴포넌트 단위로 재사용된다.

④ 재사용을 위해서는 사용법이 공개되어야 한다.

26. 공통 모듈을 설계할 때 공통 부분을 명세하기 위한 기법에 해당하지 않는 것은?

① 독립성 ② 명확성

③ 일관성 ④ 정확성

27. 코드화 대상 항목을 10진 분할하고, 다시 그 각각에 대하여 10진 분할하는 방법을 필요한 만큼 반복하는 코드로서, 코드 대상 항목의 추가가 용이하며 무제한적으로 확대할 수 있으나 자릿수가 길어질 수 있고 기계 처리에는 적합하지 않은 코드는?

① Sequence Code

② Group Classification Code

③ Block Code

④ Decimal Code

28. 다음과 같이 주로 도서 분류 코드에 사용되는 코드는?

도서 목록	부여 코드
국문학	100
철학	200
정보학	300

① 10진 코드

② 순서 코드

③ 문자 코드

④ 분류 코드

29. 코드의 종류 중 코드화 대상 항목을 자료의 발생순서, 크기 순서, 가나다라 순서 등과 같이 어떤 일정한 기준에 따라 일련번호를 부여하는 것은?

① Block Code ② Group Code

③ Sequence Code ④ Decimal Code

▶ **정답 :** 10. ② 11. ② 12. ④ 13. ② 14. ③ 15. ③ 16. ③ 17. ① 18. ② 19. ① 20. ② 21. ① 22. ② 23. ② 24. ② 25. ③ 26. ① 27. ④ 28. ① 29. ③

예·상·문·제·은·행

해설은 128쪽에 있습니다.

30. 다음과 같은 표현 방법으로 부여하는 코드는?

제품	부여 코드
냉장고(235 ℓ)	RF-235
형광등(30W 220V 흰색)	K-30-220-W
텔레비전(17인치 흑백)	T-17
텔레비전(25인치 컬러)	T-25-C

① Sequence Code
② Mnemonic Code
③ Block Code
④ Group Classification Code

31. 관찰대상의 데이터가 변화하면 이를 탐지하여 여러 가지 방식으로 사용자에게 출력하는 프로그램을 작성하고자 한다. 이 프로그램에 적용 가능한 디자인 패턴은?

① Composite 패턴
② Facade 패턴
③ Bridge 패턴
④ Observer 패턴

32. 다음 중 복잡한 서브 클래스들을 피해 더 상위에 인터페이스를 구성함으로써 서브 클래스들의 기능을 간편하게 사용할 수 있도록 하는 패턴은?

① Abstract Factory 패턴
② Facade 패턴
③ Singleton 패턴
④ Interpreter 패턴

33. 다음 중 디자인 패턴에 대한 설명으로 가장 옳지 않은 것은?

① Singleton : 하나의 객체를 생성하여 해당 객체를 어디서든 참조할 수 있도록 한다.
② Bridge : 구현에서 추상을 분리하여, 독립적으로 다양성을 가질 수 있다.
③ Chain of Responsibility : 한 객체가 처리하지 못한 요청을 다음 객체가 처리하는 형태이다.
④ Strategy : 알고리즘을 하나로 캡슐화하여 원하는 알고리즘의 선택이 가능하지만, 알고리즘의 변경이 미치는 영향이 크다.

34. 차량 네비게이션 소프트웨어에서 GPS를 수신하는 경우와 수신하지 못하는 경우에 따라 차량의 위치를 구하는 다른 알고리즘을 선택하고자 할 때 가장 적합한 설계 패턴은?

① 데코레이터 패턴
② 전략 패턴
③ 어댑터 패턴
④ 플라이웨이트 패턴

35. 다음 중 성격이 다른 설계 패턴은?

① 상태(State) 패턴
② 전략(Strategy) 패턴
③ 메멘토(Memento) 패턴
④ 컴포지트(Composite) 패턴

36. 상속을 사용하지 않고도 객체의 기능을 동적으로 확장할 수 있도록 해주는 설계 패턴은?

① 데코레이터(Decorator) 패턴
② 프록시(Proxy) 패턴
③ 빌더(Builder) 패턴
④ 커맨드(Command) 패턴

37. 여러 객체들이 서로 메시지를 주고받는 상호작용을 특정 객체 안에 캡슐화하여 서로의 존재를 모르는 상태에서도 메시지를 주고받으며 협력할 수 있도록 하는 설계 패턴은?

① Template Method
② Mediator
③ Visitor
④ Bridge

▶ **정답 :** 30. ② 31. ④ 32. ② 33. ④ 34. ② 35. ④ 36. ① 37. ②

예·상·문·제·은·행·해·설

2. Section 020

- 과정 추상화 : 자세한 수행 과정을 정의하지 않고, 전반적인 흐름만 파악할 수 있게 설계하는 방법
- 데이터 추상화 : 데이터의 세부적인 속성이나 용도를 정의하지 않고, 데이터 구조를 대표할 수 있는 표현으로 대체하는 방법

3. Section 020

②번의 내용은 아키텍처 설계가 아닌 아키텍처 구현에 대한 설명이다.

4. Section 020

- 시장 적시성 : 정해진 시간에 맞춰 프로그램을 출시하는 것
- 가용성 : 장애 없이 정상적으로 서비스를 제공하는 것
- 구축 가능성 : 모듈 단위로 구분된 시스템을 적절하게 분배하여 유연하게 일정을 변경할 수 있도록 하는 것

5. Section 020

소프트웨어 아키텍처의 설계 과정은 설계 목표 설정, 시스템 타입 결정, 아키텍처 패턴 적용, 서브시스템 구체화, 검토 순이다.

6. Section 021

MVC((Model-View-Controller) 패턴의 구성 요소

- 컨트롤러(Controller) : 사용자로부터 받은 입력을 처리함
- 모델(Model) : 서브 시스템의 핵심 기능과 데이터를 보관함
- 뷰(View) : 사용자에게 정보를 표시함

7. Section 021

레이어 패턴은 상위 계층이 제공하는 서비스를 하위 계층의 서브시스템이 사용하도록 구성되어 있는 패턴이다.

8. Section 021

MVC 패턴은 대화형 애플리케이션에 적합하며, 병렬 컴퓨팅 시스템에 적합한 패턴은 마스터-슬레이브 패턴이다.

9. Section 021

아키텍처 패턴은 시행착오를 줄여주기 때문에 개발 시간이 최소화 되고, 고품질의 소프트웨어를 생산할 수 있다.

10. Section 021

- MVC 패턴 (Model-View-Controller Pattern) : 서브시스템을 3개의 부분으로 구조화는 패턴
- 피어-투-피어 패턴(Peer-to-Peer Pattern) : 피어(peer)를 하나의 컴포넌트로 간주하며, 각 피어는 서비스를 호출하는 클라이언트가 될 수도, 서비스를 제공하는 서버가 될 수도 있는 패턴
- 브로커 패턴(Broker Pattern) : 사용자가 원하는 서비스와 특성을 브로커 컴포넌트에 요청하면 브로커 컴포넌트가 요청에 맞는 컴포넌트와 사용자를 연결줌

11. Section 022

객체지향은 기존의 절차지향 방식에 비해 실행 속도가 느리다는 단점이 있다.

12. Section 022

객체는 일정한 기억장소를 가지고 있다.

13. Section 022

- 메소드(Method) : 객체가 수행하는 기능으로 객체가 갖는 데이터(속성, 상태)를 처리하는 알고리즘이며, 객체의 상태를 참조하거나 변경하는 수단이 되는 것
- 상속(Inheritance) : 이미 정의된 상위 클래스(부모 클래스)의 모든 속성과 연산을 하위 클래스가 물려받는 것
- 추상화(Abstraction) : 문제의 전체적이고 포괄적인 개념을 설계한 후 차례로 세분화하여 구체화 시켜 나가는 것

14. Section 022

- 정보 은닉(Information Hiding) : 정보 은닉은 한 모듈 내부에 포함된 절차와 자료들의 정보가 감추어져 다른 모듈이 접근하거나 변경하지 못하도록 하는 기법
- 클래스(Class) : 공통된 속성과 연산(행위)을 갖는 객체의 집합으로, 객체의 일반적인 타입(Type)을 의미

15. Section 022

상속은 상위 클래스의 속성과 연산을 하위 클래스가 사용할 수 있기 때문에 객체와 클래스의 재사용, 즉 소프트웨어의 재사용(Reuse)을 높이는 중요한 개념이다.

예·상·문·제·은·행·해·설

17. Section 023

- 제어 결합도(Control Coupling) : 어떤 모듈이 다른 모듈 내부의 논리적인 흐름을 제어하기 위해 제어 신호를 이용하여 통신하거나 제어 요소를 전달하는 결합도
- 공통 결합도(Common Coupling) : 공유되는 공통 데이터 영역을 여러 모듈이 사용할 때의 결합도
- 스탬프 결합도(Stamp Coupling) : 모듈 간의 인터페이스로 배열이나 레코드 등의 자료 구조가 전달될 때의 결합도

19. Section 023

데이터 결합도(Data Coupling)는 매개 변수나 인수로 데이터를 전달할 때의 결합도이며, 자료 구조 형태로 전달하는 결합도는 스탬프 결합도(Stamp Coupling)이다.

20. Section 023

기능적 응집도(Functional Cohesion)는 모듈 내부의 모든 기능 요소들이 단일 문제와 연관되어 수행될 경우의 응집도이며, ②번의 내용은 순차적 응집도(Sequential Cohesion)에 대한 설명이다.

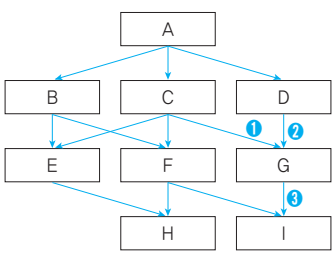
21. Section 023

- 기능적 응집도(Functional Cohesion) : 모듈 내부의 모든 기능 요소들이 단일 문제와 연관되어 수행될 경우의 응집도
- 순차적 응집도(Sequential Cohesion) : 모듈 내 하나의 활동으로부터 나온 출력 데이터를 그 다음 활동의 입력 데이터로 사용할 경우의 응집도
- 논리적 응집도(Logical Cohesion) : 유사한 성격을 갖거나 특정 형태로 분류되는 처리 요소들로 하나의 모듈이 형성되는 경우의 응집도

22. Section 023

모듈은 독립성이 높을수록 좋은 모듈이라 할 수 있으며, 독립성을 높이려면 응집도는 강하고 결합도는 약해야 한다.

23. Section 023



❶, ❷번은 팬인(Fan In), ❸번은 팬아웃(Fan Out)에 해당한다.

24. Section 024

모듈의 중복성(Redundancy)은 가능한 최소화 하는 것이 좋다.

25. Section 024

규모에 따른 재사용 대상들에는 함수, 객체(클래스, 메소드), 컴포넌트, 애플리케이션이 있다.

26. Section 024

공통 모듈을 설계할 때 공통 부분을 명세하기 위한 기법에는 정확성(Correctness), 명확성(Clarity), 완전성(Completeness), 일관성(Consistency), 추적성(Traceability)이 있다.

27. Section 025

- 순차 코드(Sequence Code) : 일정 기준에 따라 순차대로 일련번호를 부여하는 방법
- 그룹 분류 코드(Group Classification Code) : 기준에 따라 분류하여, 각 분류 내에서 일련번호를 부여하는 방법
- 블록 코드(Block Code) : 공통성 있는 것들을 블록으로 분류하여, 블록 내에서 일련번호를 부여하는 방법

30. Section 025

코드화 대상 항목의 명칭이나 약호와 관계있는 숫자나 문자, 기호를 이용하여 코드를 부여하는 방법은 연상 코드(Mnemonic Code)이다.

31. Section 026

- 컴포지트(Composite) 패턴 : 여러 객체를 가진 복합 객체와 단일 객체를 구분 없이 다루고자 할 때 사용하는 패턴
- 퍼싸드(Facade) 패턴 : 복잡한 서브 클래스들을 피해 더 상위에 인터페이스를 구성함으로써 서브 클래스들의 기능을 간편하게 사용할 수 있도록 하는 패턴
- 브리지(Bridge) 패턴 : 구현부에서 추상층을 분리하여, 서로가 독립적으로 확장할 수 있도록 구성한 패턴

32. Section 026

- 추상 팩토리(Abstract Factory) 패턴 : 구체적인 클래스에 의존하지 않고, 인터페이스를 통해 서로 연관 · 의존하는 객체들의 그룹으로 생성하여 추상적으로 표현하는 패턴

- **싱글톤(Singleton) 패턴** : 하나의 객체를 생성하면 생성된 객체를 어디서든 참조할 수 있지만, 여러 프로세스가 동시에 참조할 수는 없는 패턴
- **인터프리터(Interpreter) 패턴** : 언어에 문법 표현을 정의하는 패턴

33. Section 026

전략(Strategy) 패턴은 알고리즘들을 개별적으로 캡슐화하여 필요할 때 마다 원하는 알고리즘을 선택하여 사용할 수 있는 패턴이다.

34. Section 026

- **데코레이터(Decorator) 패턴** : 객체 간의 결합으로 능동적으로 기능들을 확장할 수 있는 패턴
- **어댑터(Adapter) 패턴** : 호환성이 없는 클래스들의 인터페이스를 다른 클래스가 이용할 수 있도록 변환해주는 패턴
- **플라이웨이트(Flyweight)** : 인스턴스가 필요할 때마다 매번 생성하는 것이 아니고 가능한 공유해서 사용함으로써 메모리를 절약하는 패턴

35. Section 026

GoF 패턴의 분류

- **생성 패턴** : 추상 팩토리, 빌더, 팩토리 메소드, 포로토타입, 싱글톤
- **구조 패턴** : 어댑터, 브리지, 컴포지트, 데코레이터, 퍼싸드, 플라이웨이트, 프록시
- **행위 패턴** : 책임 연쇄, 커맨드, 인터프리터, 반복자, 중재자, 메멘토, 옵서버, 상태, 전략, 템플릿 메소드, 방문자

36. Section 026

- **프록시(Proxy) 패턴** : 접근이 어려운 객체와 여기에 연결하려는 객체 사이에서 인터페이스 역할을 수행하는 패턴
- **빌더(Builder) 패턴** : 작게 분리된 인스턴스를 건축 하듯이 조합하여 객체를 생성하는 패턴
- **커맨드(Command) 패턴** : 요청을 객체의 형태로 캡슐화하여 재이용하거나 취소할 수 있도록 요청에 필요한 정보를 저장하거나 로그에 남기는 패턴

37. Section 026

- **템플릿 메소드(Template Method) 패턴** : 상위 클래스에서 골격을 정의하고, 하위 클래스에서 세부 처리를 구체화하는 구조의 패턴
- **방문자(Visitor) 패턴** : 자료구조와 같이 접근이 잦은 객체에 대해 동일한 인터페이스를 사용하도록 하는 패턴
- **브리지(Bridge) 패턴** : 구현부에서 추상층을 분리하여, 서로가 독립적으로 확장할 수 있도록 구성한 패턴

합격 수기

합격수기 코너는 시나공으로 공부하신 독자분들이 시험에 합격하신 후에 직접 시나공 카페(sinagong.gilbut.co.kr)의 〈합격전략/수기〉에 올려주신 자료를 토대로 구성됩니다.

한상헌 • asolid

바쁜 직장인의 정보처리기사 합격기!

1. 필기

평일에는 정말 정신이 없어서 퇴근하고 나면 녹초가 되어 도무지 의욕이 생기지 않았습니다. 그래서 그냥 별 기대 없이 시나공 책에서 분류해 준 'A 등급'의 페이지들을 뒤적거리다가 잠들었습니다.

그리고 주말이 되면 A 등급의 페이지를 다시 보고, B 등급까지 모두 보았습니다. 많이들 어려워하는 2과목 전자계산기는 그냥 책만 읽어서는 이해가 안 가는 부분이 많았습니다. 그렇다고 돈을 내고 강의를 수강하기에는 부담스럽고... 그런데 책 중간중간에 수상한 QR코드가 있어 스캔해 보았더니... 이게 정말 많은 도움이 되었습니다. 다른 과목은 책을 읽고 암기를 하면 그럭저럭 넘어갈 수 있는데, 전자계산기는 정말이지 누군가의 도움이 필요했습니다! 그 때 저에게 떡하니 내려온 토막강의! 시간도 10~15분으로 딱 집중할 수 있는 분량에다가, 모르는 부분만 어찌 그리 잘 알고 시원시원하게 알려 주던지요!

필기 합격은 정말이지 토막강의 + 시나공 책의 A, B 등급이 핵심이었던 것 같습니다.

2. 실기

실기 때는 사정이 더욱 형편 없었습니다. 마치 시험을 포기하라는 듯 과도한 회사 업무량과 함께 시험에 응시했던 동료들의 포기 등으로 인해 마음을 다잡기가 힘들었습니다. 하지만 필기를 공부했던 노력과 다음 시험까지 기다려야 하는 시간이 아까워 일단 시나공 실기 책을 주문해 버렸습니다.

데이터베이스와 애플리케이션은 도서관에 앉아서 심도 있게 공부하고, 3, 4, 5과목은 출퇴근 시간을 활용해서 공부해야겠다고 판단하고 계획을 세웠습니다.

퇴근 후 지친 몸을 이끌고 도서관에서 데이터베이스와 애플리케이션 부분을 독학했습니다. 이번에도 책 속의 토막강의가 큰 도움이 되었습니다. 문과 출신인 저에게 생소한 내용이었음에도 토막강의가 있었기에 부족한 부분을 보충할 수 있었습니다. 3, 4, 5과목은 책 읽듯이, 상식 용어 공부하듯이 틈틈이 읽었습니다. 출퇴근 버스 안에서, 화장실 갈 때, 잠들기 전 10분씩, 그리고 마지막으로 시험 당일 시험장에 일찍 가서 다시 한 번 읽은 다음 마무리로 시나공 기출문제집을 풀었습니다.

3. 그리고 5월 30일

합격자 발표날이었습니다. "합격!"의 기쁜 메시지를 받고, 즐겁게 자격증 신청을 하였습니다. 중요한 부분이 어딘지, 꼭 공부해야 하는 부분이 어딘지 가이드를 잘 해준 시나공의 책 구성과, 토막강의가 저를 합격으로 이끌어 주었습니다. 그리고 한 가지 덧붙인다면, 끝까지 포기하지 않았던 저의 결심에 하늘도 도와준 것 같습니다.

시나공 책과 스스로의 의지만 있다면 '비전공자 + 바쁜 직장인'도

정보처리기사 합격증을 손에 거머쥘 수 있습니다! 모두 파이팅하세요.

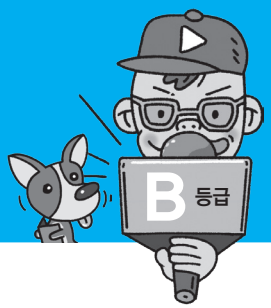
4장 인터페이스 설계

이 장에서 꼭 알아야 할 키워드 Best 10

1. 요구사항 명세서 **2.** 시스템 인터페이스 요구사항 분석 **3.** 요구사항 검토 **4.** 인터페이스 요구사항 검증 항목
5. API/Open API **6.** Socket **7.** Web Service **8.** MOM **9.** TP-Monitor **10.** WAS

SECTION 027 ● LIVE

시스템 인터페이스 요구사항 분석



전문가의 조언

시스템 인터페이스와 관련된 요구사항 분석을 위해 요구사항에 명시되어야 할 구성 요소에는 어떤 것들이 있는지를 알아야 하고, 요구사항을 기능적인 요구사항과 비기능적인 요구사항으로 구분할 수 있어야 합니다. 이를 바탕으로 요구사항 명세서에서 시스템 인터페이스 관련 요구사항을 분석하는 방법을 알아두세요.

전문가의 조언

오른쪽의 요구사항 명세서는 '소프트웨어 사업 관리감독에 관한 일반기준'에 명시되어 있는 '요구사항 내용 작성표' 양식에 맞춰 작성된 것입니다. 공공 소프트웨어 사업 제안요청서는 이 양식에 맞춰 요구사항을 작성해야 하며, 일반 기업의 소프트웨어 사업 제안요청서는 이 양식을 사용해도 되지만 대부분 기업 내에서 정한 표준 양식에 맞춰 작성합니다.

❶ 인터페이스 이름
❷ 연계 대상 시스템
❸ 연계 범위 및 내용
❹ 송신 데이터
❺ 연계 방식
❻ 인터페이스 주기
❼ 기타 고려사항

1 시스템 인터페이스 요구사항 구성

시스템 인터페이스는 독립적으로 떨어져 있는 시스템들끼리 서로 연동하여 상호 작용하기 위한 접속 방법이나 규칙을 의미한다.

- 시스템 인터페이스 요구사항은 개발을 목표로 하는 시스템과 외부 시스템을 연동하는데 필요한 시스템 인터페이스에 대한 요구사항을 기술한 것이다.
- 시스템 인터페이스 요구사항 명세서에는 인터페이스 이름, 연계 대상 시스템, 연계 범위 및 내용, 연계 방식, 송신 데이터, 인터페이스 주기, 기타 고려사항 등이 포함되어야 한다.

요구사항 명세서

요구사항 명세서는 프로젝트 개발 시 기업이나 업체가 요구하는 사항들을 구체화하여 명세화한 문서로, 시스템 기능, 데이터, 인터페이스, 품질 등의 요구사항 단위별로 작성합니다.
다음은 고객 관리 시스템의 시스템 인터페이스 요구사항 명세서에 대한 작성 예입니다.

요구사항 분류		시스템 인터페이스 요구사항
요구사항 고유번호		SIR-001 ※ 요구사항 분류와 일련번호를 조합하여 작성한다.
요구사항 명칭		❶ VOC와 CRM 연계
요구사항 상세설명	**정의**	❷ VOC 시스템과 CRM 시스템 간의 인터페이스
	세부내용	❸ • VOC 시스템을 통해 들어온 ❹ 고객의 의견 및 불만사항 등에 관한 데이터를 ❺ EAI를 통해 CRM 시스템에 ❻ 매일 1회 전달함 • 예상되는 데이터 전달 건수는 1일 1000개임 ❼ [고려사항] • VOC 시스템 담당자는 송신 시스템인 VOC 시스템에서 수신 시스템인 CRM 시스템으로 고객의 의견 및 불만사항 등에 관한 데이터를 송신할 수 있도록 프로그램을 작성함 • 목표 시스템 구축 업체는 VOC 시스템에서 송신한 데이터를 CRM 시스템에서 수신할 수 있도록 프로그램을 작성하고, VOC 시스템과의 연동 테스트를 실시함 ※ 프로젝트 이해관계자가 요구사항을 명확하게 이해할 수 있도록 구체적으로 작성한다.
산출 정보		고객 불만사항 보고서
관련 요구사항		SIR-002 인터페이스 보안 ※ 해당 기능과 연관된 요구사항 고유번호와 명칭을 조합하여 작성한다.
요구사항 출처		CRM 업무 담당자

2 시스템 인터페이스 요구사항 분석

시스템 인터페이스 요구사항 분석은 요구사항 명세서에서 요구사항을 기능적 요구사항*과 비기능적 요구사항*으로 분류하고 조직화하여 요구사항 명세*를 구체화하고 이를 이해관계자에게 전달하는 일련의 과정이다.

- 요구사항 분석은 소프트웨어 요구사항 분석 기법*을 적절히 이용한다.
- 요구사항의 분해가 필요한 경우 적절한 수준으로 세분화한다.
- 요구사항 분석 시 누락된 요구사항이나 제한조건을 추가한다.
- 요구사항에 대한 상대적 중요도를 평가하여 우선순위를 부여한다.

3 시스템 인터페이스 요구사항 분석 절차

시스템 인터페이스 요구사항 분석 절차는 다음과 같다.

❶ 소프트웨어 요구사항 목록*에서 시스템 인터페이스 관련 요구사항을 선별하여 별도로 시스템 인터페이스 요구사항 목록을 만든다.

❷ 시스템 인터페이스와 관련된 요구사항 및 아키텍처 정의서, 현행 시스템의 대 · 내외 연계 시스템 현황 자료 등 시스템 인터페이스 요구사항과 관련된 자료를 준비한다.

❸ 시스템 인터페이스에 대한 요구사항 명세서를 확인하여 기능적인 요구사항과 비기능적인 요구사항으로 분류한다.

예 다음은 회계 관련 시스템의 요구사항 명세서에서 시스템 인터페이스 관련 요구사항을 기능적/비기능적 요구사항으로 분류한 것이다.

요구사항 유형	요구사항 번호	요구사항 명칭
기능	SIR-001	고객 대상 채널 업무 연계 기능 구현
	SIR-002	은행 자금 관리 서비스(CMS) 연동
	SIR-003	인사 및 조직 정보 연계
비기능	SIR-004	대 · 내외 인터페이스 구현 방안

❹ 시스템 인터페이스 요구사항 명세서와 시스템 인터페이스 요구사항 목록 및 기타 관련 자료들을 비교하여 요구사항을 분석하고 내용을 추가하거나 수정한다.

❺ 추가 · 수정한 시스템 인터페이스 요구사항 명세서와 시스템 인터페이스 요구사항 목록을 관련 이해관계자에게 전달한다.

기능적/비기능적 요구사항
요구사항이란 어떠한 문제를 해결하기 위해 필요한 조건이나 제약사항을 의미하며, 크게 기능적 요구사항과 비기능적 요구사항으로 분류합니다. 기능적 요구사항은 시스템이 무엇을 하는지, 어떤 기능을 하는지에 대한 것이고, 비기능적 요구사항은 기능적 요구사항을 제외한 시스템이나 프로젝트 개발 과정 등에서 지켜야 할 제약사항을 의미합니다. 자세한 내용은 Section 006을 참조하세요.

요구사항 명세
여기서 말하는 요구사항 명세는 시스템 인터페이스 요구사항과 관련된 자료를 이용하여 시스템 인터페이스 요구사항 명세서와 목록을 추가 · 수정하는 작업을 의미합니다.

소프트웨어 요구사항 분석 기법
소프트웨어 요구사항 분석 기법에는 요구사항 분류, 개념 모델링, 요구사항 할당, 요구사항 협상, 정형 분석 등이 있습니다. 자세한 내용은 Section 007을 참조하세요.

소프트웨어 요구사항 목록
소프트웨어 요구사항 목록은 소프트웨어가 사용자 요구사항에 따른 문제 해결을 위해 제공하는 서비스에 대한 설명과 정상적으로 운영되기 위해 필요한 제약조건 등을 나타내는 소프트웨어 요구사항을 정리한 목록을 의미합니다.

출제예상

1. 다음 중 시스템 인터페이스 요구사항 명세서의 구성에 포함되지 않는 것은?

① 연계 대상 시스템
② 수신 데이터
③ 인터페이스 이름
④ 인터페이스 주기

시스템 인터페이스 요구사항 명세서의 구성에는 송신 데이터만 포함된다는 것! 잊지마세요.

출제예상

2. 다음 중 시스템 인터페이스 요구사항 분석에 대한 설명으로 옳지 않은 것은?

① 시스템 인터페이스 요구사항 분석은 요구사항 명세서를 통해 요구사항을 기능 · 비기능적 요구사항으로 분류하고 명세화 하는 것이다.
② 시스템 인터페이스 요구사항 분석은 소프트웨어 요구사항 분석 기법을 적절히 이용한다.
③ 시스템 인터페이스 요구사항 분석 시 정의된 인터페이스 요구사항은 분해할 수 없다.
④ 시스템 인터페이스 요구사항 분석 시 요구사항의 중요도에 따라 우선순위를 부여할 수 있다.

필요한 경우 인터페이스 요구사항을 분해할 수 있습니다.

출제예상

3. 다음 중 시스템 인터페이스 요구사항 분석 절차를 올바르게 나열한 것은?

㉠ 요구사항 분석 및 명세서 구체화
㉡ 요구사항 분류
㉢ 요구사항 관련 자료 준비
㉣ 요구사항 선별
㉤ 요구사항 명세서 공유

① ㉢ → ㉡ → ㉣ → ㉠ → ㉤
② ㉣ → ㉡ → ㉢ → ㉠ → ㉤
③ ㉢ → ㉡ → ㉣ → ㉠ → ㉤
④ ㉣ → ㉢ → ㉡ → ㉠ → ㉤

시스템 인터페이스 요구사항 분석 절차를 정확히 기억해 두세요.

출제예상

4. 시스템 인터페이스 관련 요구사항 중 비기능적 요구사항에 해당하지 않는 것은?

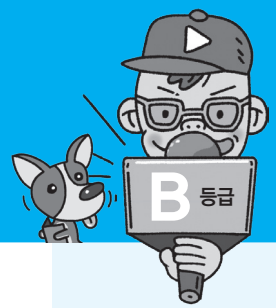
① 인사 및 조직 정보 등록
② 운영 접근 통제
③ 처리 속도 및 시간
④ 시스템 장애 대응

비기능적 요구사항은 시스템이나 프로젝트 개발 과정 등에서 지켜야 할 제약사항을 의미합니다.

▶ 정답 : 1. ② 2. ③ 3. ④ 4. ①

SECTION
028
LIVE

인터페이스 요구사항 검증



1 요구사항 검증(Requirements Verification)

요구사항 검증은 인터페이스의 설계 및 구현 전에 사용자들의 요구사항이 요구사항 명세서에 정확하고 완전하게 기술되었는지 검토하고 개발 범위의 기준인 베이스라인을 설정하는 것이다.

- 인터페이스의 설계 및 구현 중에 요구사항 명세서의 오류가 발견되어 이를 수정할 경우 많은 비용이 소요되므로 프로젝트에서 요구사항 검증은 매우 중요하다.
- 인터페이스 요구사항 검증은 '요구사항 검토 계획 수립 → 검토 및 오류 수정 → 베이스라인 설정' 순으로 수행한다.

2 인터페이스 요구사항 검토 계획 수립

프로젝트 이해관계자*들이 프로젝트 품질 관리 계획을 참조하여 다음과 같이 인터페이스 요구사항 검토 계획을 수립한다.

검토 기준 및 방법	프로젝트의 규모와 참여 인력, 검토 기간 등을 고려하여 검토 기준 및 방법을 정한다.
참여자	프로젝트 규모에 따라 이해관계자들을 파악하여 프로젝트 관리자, 품질 관리자, 인터페이스 분석가, 소프트웨어 아키텍트*, 시스템 사용자, 테스트 관리자 등 요구사항 검토 참여자를 선정한다.
체크리스트	완전성, 일관성, 명확성 등의 항목을 점검할 수 있는 요구사항 검토 체크리스트를 작성한다.
관련 자료	인터페이스 요구사항 목록, 인터페이스 요구사항 명세서, 현행 및 표준 시스템 구성도 등 인터페이스 요구사항 검토에 필요한 자료들을 준비한다.
일정	인터페이스 요구사항 검토 일정을 정한다.

- 검토 계획이 수립되면 인터페이스 요구사항 검토 참여자들에게 검토 관련 자료와 일정 등을 전달한다.

3 인터페이스 요구사항 검토 및 오류 수정

인터페이스 요구사항 검토는 검토 체크리스트의 항목에 따라 인터페이스 요구사항 명세서를 검토하는 것이다.

- 요구사항 검토 시 오류가 발견되면 오류를 수정할 수 있도록 오류 목록과 시정 조치서를 작성한다.
- 오류 수정과 요구사항 승인 절차를 진행할 수 있도록 요구사항 검토 결과를 검토 관련자들에게 전달한다.

전문가의 조언

요구사항 검증은 요구사항 명세서에 명시되어 있는 사용자의 요구사항들이 실제로 실현 가능한지를 확인하는 단계라고 생각하면 됩니다. 요구사항 검증 절차의 각 단계별 특징에 대해 알아두고 요구사항의 검증 항목과 검증 방법에는 어떤 것들이 있는지 숙지하세요.

프로젝트 이해관계자

프로젝트 이해관계자에는 품질 관리자, 프로젝트 관리자, 기술 아키텍처 전문가, 인터페이스 전문가 등이 있습니다.

소프트웨어 아키텍트(Software Architect)

- 소프트웨어 아키텍트는 아키텍처를 설계 및 구축하는 사람으로 TA, SA 등이 있습니다.
- TA(Technical Architect) : 기술 아키텍처의 설계 및 구축을 담당하는 아키텍트
- SA(Solution Architect) : 소프트웨어 아키텍처의 설계 및 구축을 담당하는 아키텍트

전문가의 조언

베이스라인을 설정한 후 인터페이스 요구사항의 변경은 공식적인 변경 통제 절차로만 가능합니다.

- 시정 조치서를 작성한 경우 시정 조치가 완료되었는지 확인하여 시정 조치가 완료되면 인터페이스 요구사항 검토 작업을 완료한다.

4 인터페이스 요구사항 베이스라인 설정

인터페이스 요구사항 검토를 통해 검증된 인터페이스 요구사항은 프로젝트 관리자와 주요 의사 결정자에게 공식적으로 승인 받는다.

- 소프트웨어 설계 및 구현을 위해 요구사항 명세서의 베이스라인을 설정한다.

5 요구사항 검증 방법

인터페이스 요구사항 검증은 다음과 같은 검증 방법을 적절하게 이용한다.

- **요구사항 검토(Requirements Review)** : 요구사항 명세서의 오류 확인 및 표준 준수 여부 등의 결함 여부를 검토 담당자들이 수작업으로 분석하는 방법으로, 동료검토, 워크스루, 인스펙션 등이 있다.

동료검토(Peer Review)	요구사항 명세서 작성자가 명세서 내용을 직접 설명하고 동료들이 이를 들으면서 결함을 발견하는 형태의 검토 방법이다.
워크스루(Walk Through)	검토 회의 전에 요구사항 명세서를 미리 배포하여 사전 검토한 후에 짧은 검토 회의를 통해 결함을 발견하는 형태의 검토 방법이다.
인스펙션(Inspection)	요구사항 명세서 작성자를 제외한 다른 검토 전문가들이 요구사항 명세서를 확인하면서 결함을 발견하는 형태의 검토 방법이다.

- **프로토타이핑(Prototyping)** : 사용자의 요구사항을 정확히 파악하기 위해 실제 개발될 소프트웨어에 대한 견본품(Prototype)을 만들어 최종 결과물을 예측한다.
- **테스트 설계** : 요구사항은 테스트할 수 있도록 작성되어야 하며, 이를 위해 테스트 케이스(Test Case)를 생성하여 이후에 요구사항이 현실적으로 테스트 가능한지를 검토한다.
- **CASE(Computer Aided Software Engineering) 도구 활용** : 일관성 분석(Consistency Analysis)을 통해 요구사항 변경사항의 추적 및 분석, 관리하고, 표준 준수 여부를 확인한다.

6 인터페이스 요구사항 검증의 주요 항목

인터페이스 요구사항 검증은 다음과 같은 항목들을 중심으로 수행한다.

- **완전성(Completeness)** : 사용자의 모든 요구사항이 누락되지 않고 완전하게 반영되어 있는가?
- **일관성(Consistency)** : 요구사항이 모순되거나 충돌되는 점 없이 일관성을 유지하고 있는가?
- **명확성(Unambiguity)** : 모든 참여자가 요구사항을 명확히 이해할 수 있는가?

- **기능성(Functionality)** : 요구사항이 '어떻게(How to)' 보다 '무엇을(What)'에 중점을 두고 있는가?
- **검증 가능성(Verifiability)** : 요구사항이 사용자의 요구를 모두 만족하고, 개발된 소프트웨어가 사용자의 요구 내용과 일치하는지를 검증할 수 있는가?
- **추적 가능성(Traceability)** : 요구사항 명세서와 설계서를 추적할 수 있는가?
- **변경 용이성(Easily Changeable)** : 요구사항 명세서의 변경이 쉽도록 작성되었는가?

기출문제 따라잡기

Section 028

출제예상

1. 다음 중 인터페이스 요구사항 검증에 대한 설명으로 틀린 것은?

① 인터페이스 요구사항 검증은 요구사항 명세서의 요구사항에 오류가 없는지 확인하는 작업이다.

② 인터페이스 요구사항 검증 방법에는 프로토타이핑, 동료 검토, 테스트 설계 등이 있다.

③ 인터페이스 요구사항 검증은 '계획 수립 → 베이스라인 설정 → 요구사항 검토 및 오류 수정'의 순으로 수행된다.

④ 인터페이스 요구사항 검토 시 오류를 발견하면 오류 목록과 시정 조치서를 작성한다.

인터페이스 요구사항 검증은 '계획 수립 → 요구사항 검토 및 오류 수정 → 베이스라인 설정'의 순으로 수행됩니다.

출제예상

2. 다음 중 요구사항 검증 방법 중 요구사항 검토(Requirements Review)에 대한 설명으로 틀린 것은?

① 요구사항 검토는 검토 담당자들이 수작업으로 분석하는 방법이다.

② 요구사항 검토 중 워크스루는 짧게는 일주일 길게는 한 달 동안 요구사항을 확인하여 결함을 발견하는 방법이다.

③ 요구사항 검토 중 동료 검토는 요구사항 명세서 작성자가 직접 설명하면서 결함을 발견하는 방법이다.

④ 요구사항 검토 중 인스펙션은 요구사항 명세서 작성자를 제외한 다른 검토 전문가나 그룹이 결함을 발견하는 방법이다.

워크스루는 검토 회의 전에 요구사항 명세서를 미리 배포하여 사전 검토하므로 회의 시간이 1~2시간으로 짧은 편입니다.

출제예상

3. 다음 중 인터페이스 요구사항 검증의 주요 항목으로 가장 먼 것은?

① 보안성(Security)

② 완전성(Completeness)

③ 기능성(Functionality)

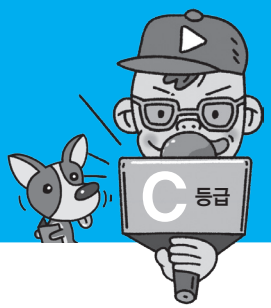
④ 추적 가능성(Traceability)

인터페이스 요구사항 검증의 주요 항목에는 완전성, 일관성, 명확성, 기능성, 검증 및 추적 가능성, 변경 용이성 등이 있습니다.

▶ 정답 : 1. ③ 2. ② 3. ①

SECTION 029 ● LIVE

인터페이스 시스템 식별



전문가의 조언

인터페이스 시스템을 식별한다는 것은 개발할 시스템과 연계할 시스템 사이의 인터페이스를 식별한 다음 각 인터페이스별로 사용되는 송 · 수신 시스템을 구분하는 것을 말합니다. 이를 위한 선행 작업으로 개발할 시스템과 이와 연계할 내 · 외부 시스템을 식별하고, 각 시스템들의 환경 및 관리 주체, 네트워크 연결 정보 등을 확인해야 합니다.

시스템 영문명

'시스템 영문명' 대신 '시스템 영문 코드'를 사용해도 됩니다.

시스템 레벨

시스템 목록 작성 시 내부 시스템의 경우에는 기업에서 사용하고 있는 시스템 분류 체계를 기반으로 시스템 레벨과 상위 시스템을 입력해야 합니다. 시스템 레벨은 일반적으로 각 단위 업무 시스템이 포함되는 상위 시스템에 따라 레벨로 구분하여 표시합니다. 예를 들어 상 · 중 · 하의 3단계로 나누어지는 시스템이라면 총 3레벨로 구분하여, 1레벨은 1, 2, …, 2레벨은 1.1, 1.2, 2.1, 2.2, …, 3레벨은 1.1.1, 1.1.2, 2.1.1, 2.1.2, …으로 각 단계마다 계층을 두어 구분합니다.

전문가의 조언

외부 시스템의 경우 타 기업의 시스템 분류 체계를 확인하기 어렵기 때문에 시스템 레벨과 상위 시스템, 설치 위치는 입력하지 않습니다.

1 개발 시스템 식별

개발 시스템을 식별하는 것은 인터페이스 관련 자료들을 기반으로 개발하고자 하는 시스템의 상세 식별 정보를 정의하고 목록을 작성하는 것이다.

예 개발할 시스템 목록

구분	시스템 한글명	시스템 영문명※	시스템 설명	시스템 레벨※	상위 시스템	설치 위치
내부	고객	CUST	고객 정보를 통합 관리하는 시스템	1		IN1
	고객관계관리	CRM	고객 정보를 분석하여 마케팅 등에 활용하는 시스템	1.1	CUST	IN2
	홈페이지	HOME	고객의 계약 조회 등 온라인 업무 지원	2		IN3

인터페이스 관련 자료

인터페이스 관련 자료에는 개발할 시스템에 대한 업무 정의서, 시스템 아키텍처 정의서, 유스케이스 정의서, 현행 시스템에 대한 인터페이스 요구사항 명세서 및 목록, 대내외 연계 시스템 목록, 연계 대상 시스템에 대한 정의서 및 인터페이스 목록 등이 있습니다.

- **시스템 아키텍처**(System Architecture) : 시스템 내부에서 각각의 하위 시스템들이 어떠한 관계로 상호 작용하는지 파악할 수 있도록 구성이나 동작 원리를 나타내는 것입니다.
- **유스케이스**(Use Case) : 사용자의 요구사항을 기능 단위로 표현하는 것입니다.

2 내 · 외부 시스템 식별

내 · 외부 시스템을 식별하는 것은 인터페이스 관련 자료들을 기반으로 개발할 시스템과 연계할 내 · 외부 시스템들의 상세 식별 정보를 정의하고 목록을 작성하는 것이다.

예 개발할 시스템과 연계할 내 · 외부 시스템의 목록

구분	시스템 한글명	시스템 영문명	시스템 설명	시스템 레벨	상위 시스템	설치 위치
내부	회계	ACC	고객의 회계를 관리하는 대내 연계 시스템	3		IN4
외부	길벗은행	ABK	길벗은행의 대외 연계 시스템			
	길벗카드	BCD	길벗카드사의 대외 연계 시스템			

3 내 · 외부 시스템 환경 및 관리 주체 식별

- 내 · 외부 시스템 환경은 연계할 시스템 접속에 필요한 IP 또는 URL, Port 정보 등 시스템의 실제 운용 환경을 의미한다.
- 내 · 외부 시스템 관리 주체는 하드웨어를 실제적으로 관리하는 담당자를 의미한다.
- 인터페이스 관련 자료들을 기반으로 내 · 외부 시스템의 실제 운용 환경과 하드웨어 관리 주체를 확인한다.

예 내 · 외부 연계 시스템 운용 환경 정보 및 하드웨어 관리 주체

구분	시스템 ID*	시스템 한글명	시스템 영문명	시스템 설명	… 중략	IP/ URL	Port	담당자
내부	ACS-001	고객	CUST	고객 정보를 통합 관리하는 시스템	…	IP1	40001	김상욱
	ACS-002	고객관계 관리	CRM	고객 정보를 분석하여 마케팅 등에 활용하는 시스템	…	IP2	40001	임선호
	ACS-003	홈페이지	HOME	고객의 계약 조회 등 온라인 업무 지원	…	IP3	40001	황진주
	ACS-004	회계	ACC	고객의 회계를 관리하는 대내 연계 시스템	…	IP4	40001	김선길
외부	ACS-005	길벗은행	ABK	길벗은행의 대외 연계 시스템	…	IP5		이가영
	ACS-006	길벗카드	BCD	길벗카드사의 대외 연계 시스템	…	IP6		신중희

4 내 · 외부 시스템 네트워크 연결 정보 식별

내 · 외부 시스템 네트워크 연결 정보는 시스템 로그인 및 DB 정보*를 의미한다.

- 인터페이스 관련 자료들을 기반으로 내 · 외부 시스템을 연계하는데 필요한 네트워크 연결 정보를 확인한다.

예 내 · 외부 연계 시스템 네트워크 연결 정보

구분	시스템 ID	Hostname	IP	Port	Login ID	Password	DB Type	DB User ID	DB User PW
내부	ACS-001	HOST1	IP1	40001	LID1	PW1	Oracle	UID1	UPW1
	ACS-002	HOST2	IP2	40001	LID2	PW2	Oracle	UID2	UPW2
	ACS-003	HOST3	IP3	40001	LID3	PW3	Oracle	UID3	UPW3
	ACS-004	HOST4	IP4	40001	LID4	PW4	DB1	UID4	UPW4
외부	ACS-005	HOST5	IP5		LID5	PW5	DB2	UID5	UPW5
	ACS-006	HOST6	IP6		LID6	PW6	DB2	UID6	UPW6

전문가의 조언

대 · 내외 시스템의 연계를 위해서는 실제 운용 환경과 하드웨어 관리 주체, 네트워크 연결 정보 등 시스템에 대한 상세 식별 정보가 필요합니다.

시스템 ID

시스템 ID는 기업에서 사용하고 있는 시스템 분류 체계에 따라 부여된 시스템 식별번호를 사용합니다.

DB 정보

DB 정보는 데이터베이스 연계 시 필요한 DBMS 유형, DBMS 로그인 정보 등을 의미합니다.

전문가의 조언

현행 시스템에서 사용하는 인터페이스를 재사용할 경우 신규와 기존 인터페이스를 구분하는 정보를 인터페이스 목록에 추가합니다.

인터페이스 ID

인터페이스 ID는 인터페이스를 구분하기 위한 식별자로, 인터페이스의 식별성 강화를 위해 업무 분류 코드와 연속 번호를 같이 사용하는 것이 일반적입니다.

인터페이스명

인터페이스명은 인터페이스 목록 작성 시 개발할 시스템과 다른 시스템 사이의 인터페이스를 식별할 수 있도록 시스템 간 인터페이스를 대표하는 이름을 부여합니다.

관련 요구사항

관련 요구사항은 인터페이스 요구사항 명세서나 목록의 요구사항 번호를 입력합니다.

전문가의 조언

인터페이스 시스템 식별은 인터페이스를 식별하고 작성한 인터페이스 목록에 있는 인터페이스들의 송 · 수신 시스템을 식별하는 것입니다.

인터페이스 구분

인터페이스가 내부 시스템 또는 내 · 외부 시스템 사이 중 어디에서 발생하는지에 따라 대 · 내외 여부를 구분하여 입력합니다.

5 인터페이스 식별

인터페이스를 식별하는 것은 인터페이스 요구사항 명세서와 인터페이스 요구사항 목록을 기반으로 개발할 시스템과 이와 연계할 내 · 외부 시스템 사이의 인터페이스를 식별하고 인터페이스 목록을 작성하는 것이다.

예 인터페이스 목록

인터페이스 ID※	인터페이스명※	설명	관련 요구사항※
IFID-001	지급 정보 전송	매일 1회 지급 대상 건에 대한 지급 데이터를 은행에 전송	IRE-001
IFID-002	은행 수금 내역 수신	매일 은행으로부터 입금 내역을 전송 받아 자동으로 가수금 전표를 생성	IRE-002
IFID-003	은행 계좌 잔액 수신	통장 잔액 확인	IRE-003
IFID-004	예금주 조회	예금주 정보 조회	IRE-004
IFID-005	법인 카드 사용 내역 조회	매일 1회 카드사로부터 법인 카드 사용 내역을 수신받아 회계시스템 DB에 저장	IRE-005
IFID-006	고객 로그인	고객이 입력한 로그인 정보의 일치여부 확인	IRE-006
IFID-007	고객 계약 조회	고객 계약에 대한 내용 조회	IRE-007

6 인터페이스 시스템 식별

인터페이스 시스템을 식별하는 것은 인터페이스별로 인터페이스에 참여하는 시스템들을 송신 시스템과 수신 시스템으로 구분하여 작성하는 것이다.

예 인터페이스 송 · 수신 시스템 식별

인터페이스 ID	인터페이스명	인터페이스 구분※	송신 시스템	수신 시스템
IFID-001	지급 정보 전송	대외	회계	길벗은행
IFID-002	은행 수금 내역 수신	대외	회계	길벗은행
IFID-003	은행 계좌 잔액 수신	대외	회계	길벗은행
IFID-004	예금주 조회	대외	회계	길벗은행
IFID-005	법인 카드 사용 내역 조회	대외	회계	길벗카드
IFID-006	고객 로그인	대내	홈페이지	고객
IFID-007	고객 계약 조회	대내	홈페이지	고객

기출문제 따라잡기

출제예상

1. 다음 중 인터페이스 시스템 식별에 대한 설명으로 틀린 것은?

① 인터페이스 요구사항 명세서와 인터페이스 요구사항 목록을 기반으로 인터페이스 시스템을 식별한다.

② 식별된 내 · 외부 시스템 간 인터페이스에 참여하는 시스템들을 송신 측과 수신 측으로 구분한다.

③ 인터페이스가 내부 시스템 간 또는 내 · 외부 시스템 간에 발생하는지를 파악하여 대 · 내외 여부를 기재한다.

④ 개발하고자 하는 시스템과 타 시스템 사이의 인터페이스를 유일하게 식별할 수 있도록 인터페이스명을 부여한다.

인터페이스 요구사항 명세서와 인터페이스 요구사항 목록을 기반으로 식별하는 것은 개발할 시스템과 이와 연계할 내 · 외부 시스템 사이의 인터페이스입니다.

출제예상

2. 다음 중 개발할 시스템과 내 · 외부 인터페이스 시스템 식별에 대한 설명으로 가장 옳지 않은 것은?

① 인터페이스 관련 자료들을 기반으로 개발하고자 하는 시스템과 연계할 내 · 외부 시스템을 정의한다.

② 시스템 목록 작성 시 대내 시스템의 경우 시스템 레벨은 한국정보화진흥원의 'CBD SW개발 표준 산출물 관리가이드'에 따라 정의한다.

③ 내 · 외부 시스템의 연계를 위한 네트워크 연결 정보를 확인한다.

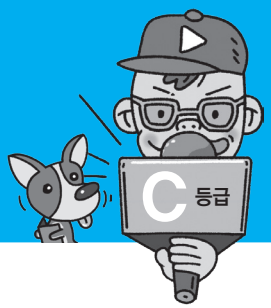
④ 내 · 외부 시스템의 실제 운용 환경과 하드웨어 관리 주체를 확인한다.

대내 시스템의 경우 시스템 레벨은 기업의 분류 체계를 기반으로 정의합니다.

▶ 정답 : 1. ① 2. ②

SECTION 030 ● LIVE

송 · 수신 데이터 식별



전문가의 조언

개발할 시스템의 내 · 외부 인터페이스를 위해 필요한 송 · 수신 데이터를 정확히 식별해야 시스템 내부 및 다른 시스템과의 인터페이스에 의해 전달되는 정보들의 변조 및 손실을 예방할 수 있습니다. 송 · 수신 데이터의 식별은 개발할 시스템과 연계할 내 · 외부 시스템 사이의 정보 흐름과 데이터베이스 산출물들을 기반으로 하며, 식별해야 하는 송 · 수신 데이터에는 인터페이스 표준 항목에 대한 데이터 항목과 코드성 데이터 항목이 있습니다. 송 · 수신 데이터를 식별하는 방법을 알아두세요.

전송 시스템 정보
전송 시스템 정보에는 IP 주소, 시스템 코드, 포트 번호 등이 있습니다.

서비스 코드 정보
서비스 코드 정보에는 수신 시스템에서 호출할 서비스 ID, 송신 시스템의 서비스 ID 등이 있습니다.

정보 흐름
정보 흐름은 필요 시 표나 다이어그램 형태로 작성합니다.

방향성
방향성은 송 · 수신 시스템간 데이터를 송 · 수신하는 형태를 의미하는 것으로 단방향과 양방향(동기, 비동기)이 있습니다. 자세한 내용은 145쪽을 참조하세요.

1 식별 대상 데이터

식별 대상 데이터는 송 · 수신 시스템 사이에서 교환되는 데이터로, 규격화된 표준 형식에 따라 전송된다.

- 교환되는 데이터의 종류에는 인터페이스 표준 항목, 송 · 수신 데이터 항목, 공통 코드가 있다.
- **인터페이스 표준 항목**
  - 인터페이스 표준 항목은 송 · 수신 시스템을 연계하는데 표준적으로 필요한 데이터를 의미한다.
  - 인터페이스 표준 항목은 시스템 공통부와 거래 공통부로 나뉜다.

시스템 공통부	• 시스템 간 연동 시 필요한 공통 정보이다. • 구성 정보에는 인터페이스 ID, 전송 시스템 정보*, 서비스 코드 정보*, 응답 결과 정보, 장애 정보 등이 있다.
거래 공통부	• 시스템들이 연동된 후 송 · 수신 되는 데이터를 처리할 때 필요한 정보이다. • 구성 정보에는 직원 정보, 승인자 정보, 기기 정보, 매체 정보 등이 있다.

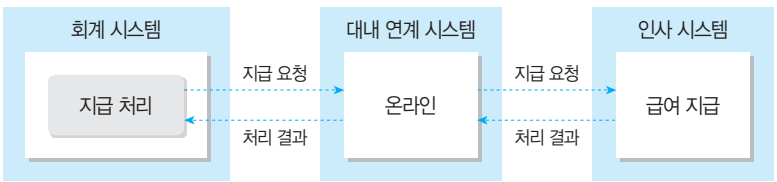
- **송 · 수신 데이터 항목**
  - 송 · 수신 데이터 항목은 송 · 수신 시스템이 업무를 수행하는 데 사용하는 데이터이다.
  - 전송되는 데이터 항목과 순서는 인터페이스별로 다르다.
- **공통 코드**
  - 공통 코드는 시스템들에서 공통적으로 사용하는 코드이다.
  - 연계 시스템이나 연계 소프트웨어에서 사용하는 상태 및 오류 코드 등과 같은 항목에 대해 코드값과 코드명, 코드 설명 등을 공통 코드로 관리한다.

2 정보 흐름 식별

정보 흐름*을 식별하는 것은 개발할 시스템과 내 · 외부 시스템 사이에서 전송되는 정보들의 방향성*을 식별하는 것이다.

- 개발할 시스템과 내 · 외부 시스템에 대한 각각의 인터페이스 목록을 확인하여 정보 흐름을 식별한다.
- 식별한 정보 흐름을 기반으로 송 · 수신 시스템 사이에서 교환되는 주요 데이터 항목이나 정보 그룹을 도출한다.

예 시스템 간 연계 흐름



3 송 · 수신 데이터 식별

개발할 시스템과 연계할 내 · 외부 시스템 사이의 정보 흐름과 데이터베이스 산출물*을 기반으로 송 · 수신 데이터를 식별한다.

- 송 · 수신 데이터의 종류에는 인터페이스 표준 항목에 대한 데이터 항목과 코드성 데이터 항목이 있다.
- **인터페이스 표준 항목과 송 · 수신 데이터 항목 식별** : 송 · 수신 시스템 사이의 교환 범위를 확인하고 인터페이스 표준 항목에 대해 송 · 수신 데이터 항목을 식별한다.

예 송 · 수신 데이터 항목 식별

인터페이스 ID	인터페이스명	정보 그룹	관련 DB	데이터 항목
IFID-008	사원 정보 저장	사원 정보	인사 DB	사원번호, 사원명, 소속, 직급, 담당업무, 연락처, 이메일
IFID-009	조직 정보 저장	조직 정보	인사 DB	조직코드, 조직명, 조직장명, 조직장 사원번호

- **코드성 데이터 항목 식별** : 코드성 데이터 항목에 대해 코드, 코드명, 코드 설명 등의 코드 정보를 식별한다.

예 코드성 데이터 항목 식별

코드분류명	코드	코드명	코드 설명
조직 코드	001	인사부	인사부의 조직 코드
조직 코드	002	영업부	영업부의 조직 코드
가맹점 업종 코드	111	음식점	음식점 업종 코드
가맹점 업종 코드	222	의류판매점	의류판매점 업종 코드

- 코드성 데이터 항목에 대해 송신 시스템에서 사용하는 코드 정보와 수신 시스템에서 사용하는 코드 정보가 동일한 경우 공통 코드 정보를 확보하고, 다른 경우 매핑 필요 대상으로 분류하여 양쪽 시스템에서 사용하는 코드 정보를 확보한다.

데이터베이스 산출물
송 · 수신 데이터를 식별하기 위해 필요한 데이터베이스 산출물에는 인터페이스 요구사항 명세서, 데이터베이스 설계 산출물(ERD), 테이블 정의서, 코드 정의서, 연계 대상 시스템의 테이블 정의서, 파일 레이아웃 등이 있습니다.

※ **테이블 정의서** : 테이블에서 관리되는 컬럼들의 특징, 인덱스, 업무 규칙 등을 정의한 문서

※ **코드 정의서** : 데이터베이스에서 코드성 속성들을 정의한 문서로, 코드성 속성은 속성명 뒤에 '코드'를 붙여 구별하고 알파벳과 문자열을 조합하여 일정한 길이로 구성함

출제예상

1. 다음 중 송 · 수신 시스템 사이에서 교환되는 데이터에 대한 설명으로 가장 옳지 않은 것은?

① 송 · 수신 데이터 항목은 인터페이스별로 전송되는 데이터 항목과 순서가 다르다.

② 인터페이스 표준 항목 중 시스템 공통부는 개념 모델링 과정에서 도출한 개체 타입과 관련 속성, 식별자 등에 대한 개괄적인 정보이다.

③ 인터페이스 표준 항목 중 거래 공통부는 연동 처리 시 필요한 직원, 승인자, 기기, 매체 등으로 구성된다.

④ 대 · 내외 연계 시 사용하는 공통 코드와 연계 시스템 등에서 사용하는 상태 또는 오류 코드 등과 같은 공통 코드 항목에 대해 코드값과 코드명, 코드 설명 정보 등을 공통 코드로 관리해야 한다.

시스템 공통부는 시스템 간 연동 시 필요한 공통 정보를 의미합니다.

출제예상

2. 다음 중 송 · 수신 데이터 식별에 대한 설명으로 가장 옳지 않은 것은?

① 개발하고자 하는 시스템과 연계할 내 · 외부 시스템 사이의 정보 흐름과 데이터베이스 산출물들을 기반으로 송 · 수신 데이터를 식별하고 정의한다.

② 송 · 수신 데이터를 식별하는데 필요한 데이터베이스 산출물에는 테이블 정의서, 코드 정의서, ERD 등이 있다.

③ 데이터베이스 산출물들을 기반으로 송 · 수신 시스템 사이에서 교환되는 범위를 파악하고 데이터 항목을 식별한다.

④ 코드성 데이터 항목에 대해 연계 시스템에서 사용하는 코드명과 코드값이 다를 경우 공통 코드 정보를 확보한다.

코드성 데이터 항목에 대해 연계 시스템에서 사용하는 코드명과 코드값이 다를 경우 코드 매핑 대상으로 식별하고 양쪽 시스템에서 사용하는 코드 정보를 확보합니다.

출제예상

3. 다음 중 개발할 시스템과 내 · 외부 시스템 사이의 정보 흐름에 대한 설명으로 가장 옳지 않은 것은?

① 인터페이스 목록에 있는 인터페이스 각각을 검토하여 송신 시스템과 수신 시스템 사이의 정보 흐름을 식별한다.

② 송 · 수신 시스템 간 정보 흐름을 기반으로 송 · 수신 시스템 사이에서 교환되는 주요 데이터 항목 또는 정보 그룹을 도출한다.

③ 인터페이스에 참여하는 송 · 수신 시스템 사이에서 단방향으로만 이동하는 정보 또는 정보들을 파악한다.

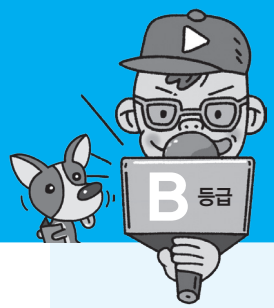
④ 정보 흐름은 필요 시 표나 다이어그램 형태로 작성할 수도 있다.

정보 흐름은 송 · 수신 시스템 사이에서 단방향 또는 양방향의 일정한 방향으로 이동하는 정보를 확인합니다.

▶ 정답 : 1. ② 2. ④ 3. ③

SECTION 031 ● LIVE

인터페이스 방법 명세화



1 인터페이스 방법 명세화의 개념

인터페이스 방법 명세화는 내 · 외부 시스템이 연계하여 작동할 때 인터페이스별 송 · 수신 방법, 송 · 수신 데이터, 오류 식별 및 처리 방안에 대한 내용을 문서로 명확하게 정리하는 것이다.

- 인터페이스별로 송 · 수신 방법을 명세화하기 위해서는 시스템 연계 기술, 인터페이스 통신 유형, 처리 유형, 발생 주기 등에 대한 정보가 필요하다.

2 시스템 연계 기술

시스템 연계 기술은 개발할 시스템과 내 · 외부 시스템을 연계할 때 사용되는 기술을 의미한다.

- 주요 시스템 연계 기술에는 DB Link, API/Open API, 연계 솔루션, Socket, Web Service 등이 있다.
  - DB Link : DB에서 제공하는 DB Link 객체를 이용하는 방식이다.
  - API/Open API* : 송신 시스템의 데이터베이스(DB)에서 데이터를 읽어 와 제공하는 애플리케이션 프로그래밍 인터페이스 프로그램이다.
  - 연계 솔루션 : EAI* 서버와 송 · 수신 시스템에 설치되는 클라이언트(Client)를 이용하는 방식이다.
  - Socket : 서버는 통신을 위한 소켓(Socket)을 생성하여 포트를 할당하고 클라이언트의 통신 요청 시 클라이언트와 연결하여 통신하는 네트워크 기술이다.
  - Web Service : 웹 서비스(Web Service)에서 WSDL*과 UDDI*, SOAP* 프로토콜을 이용하여 연계하는 서비스이다.

3 인터페이스 통신 유형

인터페이스 통신 유형은 개발할 시스템과 내 · 외부 시스템 간 데이터를 송 · 수신하는 형태를 의미한다.

- 인터페이스 통신 유형에는 단방향, 동기, 비동기 방식 등이 있다.
  - 단방향 : 시스템에서 거래를 요청만 하고 응답이 없는 방식이다.
  - 동기 : 시스템에서 거래를 요청하고 응답이 올 때까지 대기(Request-Reply)하는 방식이다.
  - 비동기 : 시스템에서 거래를 요청하고 다른 작업을 수행하다 응답이 오면 처리하는 방식(Send-Receive, Send-Receive-Acknowledge, Publish-Subscribe)이다.

전문가의 조언

인터페이스 방법 명세화란 내 · 외부 시스템이 연계하여 작동할 때 데이터를 주고받는 방법, 주고받는 데이터의 종류, 에러 발생 시 처리해야 할 내용들을 문서로 명확하게 정리하는 것을 말합니다. 송 · 수신 방법에 대한 명세화를 위해 필요한 시스템 연계 기술, 인터페이스 통신 유형, 처리 유형, 발생 주기 등을 먼저 확인한 후 각각의 명세화 방법을 정리하세요.

API(Application Programming Interface)/Open API

API는 운영체제나 프로그래밍 언어 등에 있는 라이브러리를 응용 프로그램 개발 시 이용할 수 있도록 규칙 등에 대해 정의해 놓은 인터페이스를 말하고, Open API는 이러한 기능을 누구나 무료로 사용하여 프로그램을 개발하거나 Open API에 새로운 API를 추가할 수 있도록 공개된 API를 말합니다.

EAI(Enterprise Application Integration)

EAI는 송 · 수신 데이터를 식별하기 위해 송 · 수신 처리 및 진행 현황을 모니터링하고 통제하는 시스템입니다.

- WSDL(Web Services Description Language) : 웹 서비스와 관련된 서식이나 프로토콜 등을 표준적인 방법으로 기술하고 게시하기 위한 언어
- UDDI(Universal Description, Discovery and Integration) : 인터넷에서 전 세계의 비즈니스 업체 목록에 자신의 목록을 등록하기 위한 확장성 생성 언어(XML) 기반의 규격
- SOAP(Simple Object Access Protocol) : 웹 서비스를 실제로 이용하기 위한 객체 간의 통신 규약

4 인터페이스 처리 유형

인터페이스 처리 유형은 송 · 수신 데이터를 어떤 형태로 처리할 것인지에 대한 방식을 의미한다.

- 업무의 성격과 송 · 수신 데이터 전송량을 고려하여 실시간, 지연 처리, 배치 방식 등으로 구분한다.
  - 실시간 방식 : 사용자가 요청한 내용을 바로 처리해야 할 때 사용하는 방식이다.
  - 지연 처리 방식 : 데이터를 매건 단위로 처리할 경우 비용이 많이 발생할 때 사용하는 방식이다.
  - 배치 방식 : 대량의 데이터를 처리할 때 사용하는 방식이다.

5 인터페이스 발생 주기

인터페이스 발생 주기는 개발할 시스템과 내 · 외부 시스템 간 송 · 수신 데이터가 전송되어 인터페이스가 사용되는 주기를 의미한다.

- 인터페이스 발생 주기는 업무의 성격과 송 · 수신 데이터 전송량을 고려하여 매일, 수시, 주 1회 등으로 구분한다.

6 송 · 수신 방법 명세화

송 · 수신 방법 명세화는 내 · 외부 인터페이스 목록에 있는 각각의 인터페이스에 대해 연계 방식, 통신 및 처리 유형, 발생 주기 등의 송 · 수신 방법을 정의하고 명세를 작성하는 것이다.

- 연계 방식, 통신 유형, 연계 처리 형태는 시스템 인터페이스 설계 시 작성한 아키텍처 정의서를 기반으로 하여 업무 및 데이터의 성격, 연계 데이터 발생 건수, 연계 시스템의 기술 구조, 시스템 간의 성능 등을 고려하여 작성한다.

예 인터페이스 송 · 수신 방법 명세화

인터페이스 ID	인터페이스명	송신 시스템	수신 시스템	연계 방식	통신 유형	연계 처리 형태	연계 주기
IFID-001	지급 정보 전송	회계	길벗은행	EAI	요청/응답	실시간	매일
IFID-002	은행 수금 내역 수신	회계	길벗은행	EAI	요청/응답	실시간	매일
IFID-003	은행 계좌 잔액 수신	회계	길벗은행	Soket	요청/응답	실시간	수시
IFID-004	예금주 조회	회계	길벗은행	Soket	요청/응답	실시간	수시
IFID-005	법인 카드 사용 내역 조회	회계	길벗카드	Web Service	요청/응답	실시간	매일
IFID-006	고객 로그인	홈페이지	고객	EAI	단방향	배치	매일
IFID-007	고객 계약 조회	홈페이지	고객	DB Link	단방향	배치	매일

7 송 · 수신 데이터 명세화

송 · 수신 데이터 명세화는 내 · 외부 인터페이스 목록에 있는 각각의 인터페이스에 대해 인터페이스 시 필요한 송 · 수신 데이터에 대한 명세를 작성하는 것이다.

- 인터페이스별로 테이블 정의서*와 파일 레이아웃에서 연계하고자 하는 테이블 또는 파일 단위로 송 · 수신 데이터에 대한 명세를 작성한다.

예 송 · 수신 데이터 명세화

인터페이스 ID	IFID-006
송 · 수신 구분	수신
시스템명	고객 시스템
업무	고객 정보 관리
서비스명	고객 정보 저장

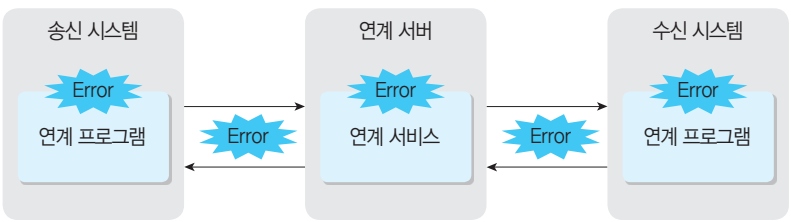
데이터 항목

Seq	Field*	Key*	Type	Size*	Null 허용	Description*	Condition*
1	CON_NO	Y	varchar	13	N	고객번호	
2	REQ_CNT	Y	varchar	2	N	반복회차	
3	SEQ		varchar	2	N	순번	
4	SSN		char	13	N	주민번호	암호화*
5	NAME		char	10	N	이름	암호화
6	LICN_CODE		char	16	N	자격코드	코드

8 오류 식별 및 처리 방안 명세화

오류 식별 및 처리 방안 명세화는 내 · 외부 인터페이스 목록에 있는 각각의 인터페이스에 대해 인터페이스 시 발생할 수 있는 오류를 식별하고 오류 처리 방안에 대한 명세를 작성하는 것이다.

- 시스템 및 전송 오류, 연계 프로그램 등에서 정의한 예외 상황 등 대 · 내외 시스템 연계 시 발생할 수 있는 다양한 오류 상황을 식별하고 분류한다.



인터페이스 오류 발생 영역

- 오류 상황에 대해 오류 코드, 오류 메시지, 오류 설명, 해결 방법 등을 명세화 한다.

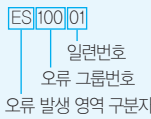
테이블 정의서
테이블 정의서는 테이블에서 관리되는 컬럼들의 특징, 인덱스, 업무 규칙 등을 정의한 문서입니다.

- Field : 테이블 정의서의 필드 ID를 기재합니다.
- Key : 필드가 키(Key)인 경우에만 기재합니다.
- Size :필드의 길이는 바이트(Byte) 단위로 기재합니다.
- Description : 필드의 간단한 설명을 기재합니다.
- Condition : 암호화 적용 여부 또는 공통 코드 여부를 기재합니다.
- **암호화** : 보안이 중요한 데이터 항목의 경우 법률적 근거와 기업의 개인정보 규정 등을 참고하여 암호화 대상을 선택하고 암호화 적용 여부를 기재합니다. 법률에서 정한 암호화 필수 적용 대상 항목에는 주민등록번호, 패스워드, 계좌번호, 공개에 동의하지 않은 개인정보 등이 있습니다.

오류 코드

오류 코드는 표준화된 오류 코드 작성 규칙을 준수해야 하지만 표준 작성 규칙이 없을 경우 이해관계자들의 합의를 통해 지정합니다.

예 오류 코드 명명 규칙



예 오류 식별 및 처리 방안 명세화

오류 코드*	오류 메시지	설명	해결 방법
ES10001	연계 서버에 접속할 수 없음	연계 서버의 네트워크 회선 오류 등으로 인해 연계 서버에 접속할 수 없음	연계 서버의 네트워크 회선 오류 여부를 확인 후 조치
ES50001	연계 서버에서 데이터 변환 에러가 발생함	연계 서버에서 데이터 변환 과정에서 유효하지 않은 코드값으로 인해 매핑 오류가 발생함	미등록 코드를 코드 테이블과 매핑 정의서에 등록한 후 재실행
SD40001	송신 시스템에서 데이터 조회에 실패함	송신 시스템의 인터페이스 프로그램에서 전송할 데이터를 DB에서 읽어 오지 못함	데이터베이스 접근 권한 문제, 작동 여부, 데이터 테이블 삭제 등을 확인 후 재실행

기출문제 따라잡기 Section 031

출제예상

1. 다음 중 인터페이스 송 · 수신 방법을 명세화 하는데 포함되는 항목이 아닌 것은?

① 인터페이스 발생 시점
② 인터페이스 통신 유형
③ 인터페이스 처리 유형
④ 인터페이스 연계 방식

인터페이스 송 · 수신 방법 명세화 하면 인터페이스 '연계 방식, 처리 유형, 통신 유형, 발생 주기'라는 것을 기억해 두세요.

출제예상

2. 다음 보기에 해당하는 시스템 연계 표준 기술은 무엇인가?

〈보기〉
서버는 통신을 위한 소켓을 생성하여 포트를 할당하고 클라이언트의 통신 요청 시 클라이언트와 연결하여 통신하는 네트워크 기술이다.

① DB Link ② Socket
③ Web Service ④ DB Connection

〈보기〉 안에 정답이 숨어 있으니 다시 한 번 잘 읽어보세요.

출제예상

3. 다음 중 인터페이스 통신 유형으로 올바르게 구성된 것은?

① 단방향, 실시간, 동기
② 단방향, 동기, 비동기
③ 동기, 비동기, 배치
④ 실시간, 지연, 배치

인터페이스 통신 유형하면 '단방향, 동기, 비동기'라는 것을 기억해 두세요.

출제예상

4. 다음 중 인터페이스 처리 유형과 발생 주기에 대한 설명으로 가장 옳지 않은 것은?

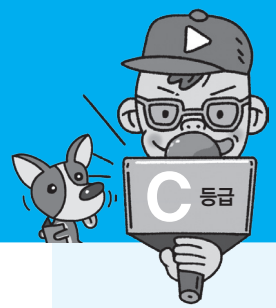
① 인터페이스 처리 유형은 업무의 성격 및 전송량을 고려하여 정의한다.
② 인터페이스 발생 주기는 업무 성격과 송 · 수신 데이터 양을 고려하여 매일, 수시 등으로 정의한다.
③ 인터페이스 처리 유형 중 지연 처리 방식은 데이터를 매건 단위로 처리할 경우 비용이 많이 발생할 때 사용한다.
④ 인터페이스 처리 유형 중 배치 방식은 소량의 데이터를 처리할 경우 사용한다.

배치 방식은 소량이 아니라 대량의 데이터를 처리할 경우 사용합니다.

▶ 정답 : 1. ① 2. ② 3. ② 4. ④

SECTION 032 LIVE

시스템 인터페이스 설계서 작성



1 시스템 인터페이스 설계서의 개요

시스템 인터페이스 설계서는 시스템의 인터페이스 현황을 확인하기 위해 시스템이 갖는 인터페이스 목록과 각 인터페이스의 상세 데이터 명세를 정의한 문서이다.

- 시스템 인터페이스 설계서는 시스템 인터페이스 목록과 시스템 인터페이스 정의서로 구성된다.
- 시스템 인터페이스 설계서는 인터페이스 송 · 수신 방법*과 인터페이스 송 · 수신 데이터* 명세화 과정에서 작성한 산출물을 기반으로 작성한다.
- 시스템 인터페이스 설계서를 작성한 후에는 시스템 인터페이스 목록에 있는 각각의 인터페이스를 시스템 인터페이스 정의서의 내용과 비교하여 누락되거나 보완이 필요한 경우 내용을 수정한다.
- 시스템 인터페이스 설계서는 내 · 외부 모듈 간 공통적으로 제공되는 기능과 각 데이터의 인터페이스를 확인하는데 사용된다.

2 시스템 인터페이스 목록 작성

시스템 인터페이스 목록은 업무 시스템과 내 · 외부 시스템 간 데이터를 주고받는 경우에 사용하는 인터페이스에 대해 기술한 것이다.

- 시스템 인터페이스 목록에는 연계 업무와 연계에 참여하는 송 · 수신 시스템의 정보, 연계 방식과 통신 유형 등에 대한 정보를 기록한다.

예 시스템 인터페이스 목록

인터페이스 ID		경영관리		업무대분류		기획총괄		작성일	2019.05.25		작성자	김선길	
인터페이스 ID	인터페이스명	송신 기관*	송신 시스템	수신 기관*	수신 시스템	대내외 구분*	연계 방식*	통신 유형*	처리 형태*	연계 주기*	데이터 형식	담당자	요구사항 ID*
IFID-001	지급 정보 전송	당사	회계	길벗은행	수신	대외	EAI	요청/응답	실시간	매일	고정 길이	김상욱	IRE-001
IFID-002	은행 수금 내역 수신	당사	회계	길벗은행	수신	대외	EAI	요청/응답	실시간	매일	고정 길이	이가영	IRE-002
IFID-003	은행 계좌 잔액 수신	당사	회계	길벗은행	수신	대외	Soket	요청/응답	실시간	수시	고정 길이	임선호	IRE-003
IFID-004	예금주 조회	당사	회계	길벗은행	수신	대외	Soket	요청/응답	실시간	수시	고정 길이	이상희	IRE-004

연계 방식, 통신 유형, 처리 형태, 연계 주기는 인터페이스 송 · 수신 방법 명세화를 참고하여 기재합니다.

전문가의 조언

시스템 인터페이스 설계서는 인터페이스 방법 명세화 과정에서 작성한 산출물들을 토대로 모듈 간의 공통 기능과 데이터 인터페이스의 식별을 위해 시스템 인터페이스 목록과 정의서를 작성하는 것입니다. 시스템 인터페이스 설계서의 개념 및 시스템 인터페이스 목록과 정의서를 작성하는 방법에 대해 정리하세요.

인터페이스 송 · 수신 방법과 인터페이스 송 · 수신 데이터 명세화에 대한 내용은 147쪽을 참조하세요.

- **송 · 수신 기관** : 내부 시스템이 송 · 수신하는 인터페이스인 경우 송 · 수신 기관은 '당사'로 기재합니다.
- **대 · 내외 구분** : 인터페이스가 기업 내부 시스템 간 발생하는 경우 '대내', 내 · 외부 시스템 간에 발생하는 경우 '대외'로 구분합니다.
- **데이터 형식(포맷)** : 고정길이, XML 등 인터페이스 항목의 데이터 형식(포맷)을 기재합니다.
- **요구사항 ID** : 해당 인터페이스와 관련된 요구사항 분석 단계에서 작성한 인터페이스 정의서의 요구사항 ID를 기재합니다.

IFID-005	법인 카드 사용 내역 조회	당사	회계	길벗카드	법인 카드	대외	Web Service	요청/응답	실시간	매일	고정 길이	고희식	IRE-005
IFID-006	고객 로그인	당사	홈페이지	당사	고객	대내	EAI	단방향	배치	매일		김한순	IRE-006
IFID-007	고객 계약 조회	당사	홈페이지	당사	고객	대내	DB Link	단방향	배치	매일		황진주	IRE-007

3 시스템 인터페이스 정의서 작성

시스템 인터페이스 정의서는 인터페이스별로 시스템 간의 연계를 위해 필요한 데이터 항목 및 구현 요건 등을 기술하는 것이다.

- 시스템 인터페이스 정의서에는 데이터 송 · 수신 시스템 간 데이터 저장소와 속성 등 상세 정보를 기록한다.

예 시스템 인터페이스 정의서

Inferface ID	IF-006	Inferface Name	사원 정보 저장	Description	매일 1회 변경된 사원 기본정보를 인사관리시스템에서 조회해서	작성일	2019-05-25		
처리유형	배치	통신유형	단방향	주기	매일	인터페이스 구분	대내	데이터 포맷	고정길이 ❶
최대 처리 횟수*	10000	데이터크기*	100M						

송신 시스템		수신 시스템		Mapping Rule
시스템명	전자 결재 시스템	시스템명	인사 관리 시스템	
업무	사원 정보 관리	업무	사원 정보 갱신	
서비스명	사원 갱신 정보 조회(IF_svc06)	서비스명	사원 갱신 정보 저장(IF_svc07)	
인터페이스 방식	EAI	인터페이스 방식	EAI	
담당자/연락처	김모모	담당자/연락처	박모모	

Input								Output								
Seq	Field	Key	Type	SIZE	null허용	Description	Condition	Seq	Field	Key	Type	SIZE	null허용	Description	Condition	
1	CON_NO	Y	varchar	13	N	사원번호		1	SNO	Y	varchar	13	N	사원번호		00001~99999 허용
2	REQ_CNT	Y	varchar	2	N	반복회차		2	REQ_CNT	Y	varchar	2	N	반복회차		
3	SEQ	Y	varchar	2	N	순번		3	SEQ	Y	varchar	2	N	순번		차주의 계좌 번호
4	SSN		char	13	N	주민번호	암호화	4	SSN		char	13	N	주민번호	암호화	
5	NAME		char	10	N	이름	암호화	5	NAME		char	10	N	이름	암호화	
6	LICN_CODE		char	16	N	자격코드	코드	6	QAL_CODE		char	16	N	자격코드	코드	LICN_CODE(자격코드) 매핑 규칙 준수
~							❷	28								
	Total Length			56					Total Length			56				

❸

- **최대 처리 횟수** : 단위 시간당 처리될 수 있는 해당 인터페이스 최대 수행 건수를 기재합니다.
- **데이터 크기** : 해당 인터페이스 1회 처리 시 소요되는 데이터의 평균 및 최대 크기를 기재합니다.

❶ 인터페이스 ID, 인터페이스명, 처리 유형, 통신 유형, 주기, 인터페이스 구분, 데이터 포맷은 인터페이스 목록을 참고하여 기재한다.

❷ 인터페이스 송 · 수신 데이터 명세를 참조하여 기재한다.

❸ 코드 매핑 규칙을 작성한다. 매핑 규칙은 송 · 수신 시스템의 항목이 단순 1:1이 아니기 때문에 병합, 변환 등 별도의 처리 로직이 필요한 경우의 규칙을 기재한다.

기출문제 따라잡기

Section 032

출제예상

1. 시스템의 내 · 외부 인터페이스를 식별하고 인터페이스의 명세를 기술하기 위해 작성하는 문서는?

① 시스템 인터페이스 제안 요청서
② 시스템 인터페이스 설계서
③ 시스템 인터페이스 요구사항 정의서
④ 시스템 인터페이스 요구사항 목록

'인터페이스 식별 및 명세 기술'하면 시스템 인터페이스 설계서라는 것을 잊지마세요.

출제예상

2. 다음 중 시스템 인터페이스 정의서에 대한 설명으로 가장 옳지 않은 것은?

① 시스템 인터페이스 정의서에는 데이터 송 · 수신 시스템 간의 데이터 저장소와 속성 등을 작성한다.
② 인터페이스 ID와 인터페이스명 등 기본 정보는 송 · 수신 데이터 명세를 참고하여 기재한다.
③ 최대 처리 횟수는 단위 시간당 처리될 수 있는 해당 인터페이스 최대 수행 건수를 기재한다.
④ 데이터 크기는 해당 인터페이스 1회 처리 시 소요되는 데이터의 평균 및 최대 크기를 기재한다.

인터페이스 ID, 인터페이스명, 처리 유형, 통신 유형 등은 시스템 인터페이스 목록에 기록하는 내용입니다.

출제예상

3. 다음 중 시스템 인터페이스 설계서의 구성으로만 올바르게 묶은 것은?

① 시스템 인터페이스 정의서, 시스템 인터페이스 제안 요청서
② 시스템 인터페이스 정의서, 시스템 인터페이스 업무 정의서
③ 시스템 인터페이스 목록, 시스템 인터페이스 제안 요청서
④ 시스템 인터페이스 목록, 시스템 인터페이스 정의서

시스템 인터페이스 설계서는 시스템 인터페이스 목록과 이를 참고하여 작성한 시스템 인터페이스 정의서로 구성됩니다.

▶ 정답 : 1. ② 2. ② 3. ④

SECTION
033
LIVE

미들웨어 솔루션 명세

전문가의 조언

미들웨어는 클라이언트가 서버 측에 어떠한 처리를 요구하고, 또 서버가 그 처리한 결과를 클라이언트에게 돌려주는 과정을 효율적으로 수행하도록 도와주는 소프트웨어입니다. 예를 들어 미들웨어는 웹 서버와 DB 서버 사이에서 웹 서버가 요구하는 다양한 요청사항들을 DB 서버에 적합한 인터페이스로 변환하여 요청하고 그 결과를 다시 웹 서버에 반환함으로써 원활하게 데이터가 오갈 수 있도록 도와주는 중계자의 역할을 수행합니다. 미들웨어의 개념과 종류별 특징을 구분할 수 있도록 알아두세요.

1 미들웨어의 개념 및 종류

미들웨어(Middleware)는 미들(Middle)과 소프트웨어(Software)의 합성어로, 운영체제와 해당 운영체제에서 실행되는 응용 프로그램 사이에서 운영체제가 제공하는 서비스 이외에 추가적인 서비스를 제공하는 소프트웨어이다.

- 미들웨어는 표준화된 인터페이스를 제공함으로써 시스템 간의 데이터 교환에 일관성을 보장한다.
- 미들웨어는 통신 제공 방법이나 기능에 따라 DB, RPC, MOM, TP-Monitor, ORB, WAS 등으로 구분한다.

2 DB(DataBase)

DB는 데이터베이스 벤더에서 제공하는 클라이언트에서 원격의 데이터베이스와 연결하기 위한 미들웨어이다.

- DB를 사용하여 시스템을 구축하는 경우 보통 2-Tier 아키텍처라고 한다.
- 대표적인 DB의 종류에는 마이크로소프트의 ODBC, 볼랜드의 IDAPI, 오라클의 Glue 등이 있다.

3 RPC(Remote Procedure Call)

RPC(원격 프로시저 호출)는 응용 프로그램의 프로시저를 사용하여 원격 프로시저를 마치 로컬 프로시저처럼 호출하는 방식의 미들웨어이다.

- 대표적인 RPC의 종류에는 이큐브시스템스의 Entera, OSF의 ONC/RPC 등이 있다.

4 MOM(Message Oriented Middleware)

MOM(메시지 지향 미들웨어)은 메시지 기반의 비동기형 메시지를 전달하는 방식의 미들웨어이다.

- 온라인 업무보다는 이기종 분산 데이터 시스템의 데이터 동기를 위해 많이 사용된다.
- 대표적인 MOM의 종류에는 IBM의 MQ, 오라클의 Message Q, JCP의 JMS 등이 있다.

5 TP-Monitor(Transaction Processing Monitor)

TP-Monitor(트랜잭션 처리* 모니터)는 항공기나 철도 예약 업무 등과 같은 온라인 트랜잭션 업무에서 트랜잭션을 처리 및 감시하는 미들웨어이다.

- 사용자 수가 증가해도 빠른 응답 속도를 유지해야 하는 업무에 주로 사용된다.
- 대표적인 TP-Monitor의 종류에는 오라클의 tuxedo, 티맥스소프트의 tmax 등이 있다.

6 ORB(Object Request Broker)

ORB(객체 요청 브로커)는 객체 지향 미들웨어로 코바(CORBA)* 표준 스펙을 구현한 미들웨어이다.

- 최근에는 TP-Monitor의 장점인 트랜잭션 처리와 모니터링 등을 추가로 구현한 제품도 있다.
- 대표적인 ORB의 종류에는 Micro Focus의 Orbix, OMG의 CORBA 등이 있다.

7 WAS(Web Application Server)

WAS(앱 애플리케이션 서버)는 정적인 콘텐츠를 처리하는 웹 서버와 달리 사용자의 요구에 따라 변하는 동적인 콘텐츠를 처리하기 위해 사용되는 미들웨어이다.

- 클라이언트/서버 환경보다는 웹 환경을 구현하기 위한 미들웨어이다.
- HTTP 세션 처리를 위한 웹 서버 기능뿐만 아니라 미션-크리티컬*한 기업 업무까지 JAVA, EJB* 컴포넌트 기반으로 구현이 가능하다.
- 대표적인 WAS의 종류에는 오라클의 WebLogic, IBM의 WebSphere 등이 있다.

8 미들웨어 솔루션 식별

미들웨어 솔루션 식별은 개발 및 운영 환경에 사용될 미들웨어 솔루션을 확인하고 목록을 작성하는 것이다.

- 소프트웨어 아키텍처에서 정의한 아키텍처 구성 정보와 프로젝트에서 구매가 진행 중이거나 구매 예정인 소프트웨어 내역을 확인하여 개발 및 운영 환경에서 사용될 미들웨어 솔루션을 식별한다.
- 식별한 미들웨어 솔루션들에 대해 솔루션의 시스템, 구분, 솔루션명, 버전, 제조사 등의 정보를 정리한 미들웨어 솔루션 목록을 작성한다.
- 작성된 미들웨어 솔루션 목록은 이해관계자 등에게 전달하여 오류 및 누락을 확인하고 수정한다.

트랜잭션 처리
트랜잭션 처리는 온라인 업무 처리 형태의 하나로 네트워크 상의 여러 이용자가 실시간으로 데이터베이스의 데이터를 갱신하거나 검색하는 등의 단위 작업을 처리하는 방식을 말합니다. 작업이 온라인으로 처리되기 때문에 온라인 트랜잭션 처리(OLTP; Online Transaction Processing)라고도 부릅니다.

코바(CORBA; Common Object Request Broker Architecture)
코바는 네트워크에서 분산 프로그램 객체를 생성, 배포, 관리하기 위한 규격을 의미합니다.

미션-크리티컬
미션-크리티컬이란 업무를 수행하는 데 있어 가장 중요한 요소를 의미합니다.

EJB(Enterprise JavaBeans)
EJB는 클라이언트/서버 모델의 서버 부분에서 운영되는 자바 프로그램 컴포넌트들을 설정하기 위한 아키텍처로, 대규모의 분산 객체 환경을 쉽게 구현할 수 있도록 도와줍니다.

예 미들웨어 솔루션 목록

시스템	구분	솔루션명	버전	제조사
사용자 관리 시스템	WAS	nginx	ver 4.1	nginxsoft
데이터 관리 시스템	TP-Monitor	tuxedo	ver 2.0	oracle
결제 관리 시스템	WAS	jeus	ver 3.0	tmaxsoft
콘텐츠 관리 시스템	MOM	titan	ver 1.2	h2o

9 미들웨어 솔루션 명세서 작성

미들웨어 솔루션 명세서는 미들웨어 솔루션 목록의 미들웨어 솔루션별로 관련 정보들을 상세하게 기술하는 것이다.

- 미들웨어 솔루션 제품 명칭 및 버전, 제품 사용 목적 등을 솔루션에 대한 제품안내서 및 설명 자료 등을 통해 검토한다.
- 미들웨서 솔루션 제품에 대한 사용 환경과 특징 등을 솔루션 설명 자료나 관련 담당자를 통해 검토한다.
- 미들웨어 솔루션이 지원하는 시스템 범위와 정상적인 서비스 제공을 위한 환경 구성, 제공 기능 등에 대한 제약사항이 존재하는지 제품안내서 및 기술 지원 담당자를 통해 검토한다.
- 미들웨어 솔루션에 대한 상세 정보 및 제공 기능, 특징, 시스템 구성 환경 등에 대한 제약사항을 정리하여 솔루션에 대한 명세서를 작성한다.

기출문제 따라잡기 Section 033

출제예상

1. 다음 중 미들웨어 솔루션의 유형에 대한 설명으로 가장 옳지 않은 것은?

① ORB는 객체 지향 미들웨어로 코바(CORBA) 표준 스펙을 구현한 미들웨어이다.

② DB는 데이터베이스 벤더에서 제공하는 클라이언트에서 데이터베이스와 연결하기 위한 미들웨어이다.

③ WAS는 웹 환경보다는 클라이언트/서버 환경을 구현하기 위한 미들웨어이다.

④ MOM은 메시지 기반의 비동기형 메시지를 전달하는 방식의 미들웨어이다.

WAS는 클라이언트/서버 환경보다는 웹 환경을 구현하기 위한 미들웨어입니다.

출제예상

2. 다음 중 개발 및 운영 환경에서 사용될 미들웨어 솔루션을 확인하는 방법에 대한 설명으로 가장 옳지 않은 것은?

① 아키텍처 구성 정보를 검토한다.

② 프로젝트의 구매 SW를 검토한다.

③ 프로젝트에서 사용될 미들웨어 솔루션에 대해 정리한다.

④ 솔루션의 제약사항에 대해 검토한다.

솔루션 제약사항의 검토는 미들웨어 솔루션의 명세서를 작성할 때 수행합니다.

기출문제 따라잡기

출제예상

3. 다음 중 미들웨어 솔루션에 대한 명세서를 작성하는 방법에 대한 설명으로 가장 옳지 않은 것은?

① 누락된 솔루션에 대해 검토한다.
② 제조사의 제품 소개서 내용을 검토한다.
③ 솔루션의 제공 기능 및 특징 등에 대해 검토한다.
④ 솔루션의 제약사항에 대해 검토한다.

누락된 솔루션에 대한 검토는 미들웨어 솔루션을 확인할 때 수행합니다.

출제예상

4. 다음이 설명하는 미들웨어는?

- 은행 계정, 항공기와 버스 예약 업무 등 온라인 트랜잭션 업무에서 트랜잭션을 처리하고 감시하는 미들웨어이다.
- 사용자 수가 증가하여도 빠른 응답 속도를 유지해야 하는 업무에 적합한 미들웨어이다.

① TP-Monitor ② MOM
③ ORB ④ WAS

트랜잭션(Transaction)을 처리(Processing)하고 감시(Monitoring)하는 미들웨어는 무엇일까요?

▶ 정답 : 1. ③ 2. ④ 3. ① 4. ①

예·상·문·제·은·행

1. 다음 중 인터페이스 요구사항에 대한 설명으로 틀린 것은?

① 인터페이스 요구사항은 목표 시스템과 외부 시스템을 연동하는데 필요한 인터페이스에 대한 요구사항을 기술한 것이다.

② 인터페이스 요구사항의 인터페이스 이름은 보안을 위해 사람들이 이해하기 어려운 이름으로 지정한다.

③ 인터페이스 요구사항의 인터페이스 전송 주기는 수시, 매일, 주 1회 등으로 지정한다.

④ 인터페이스 요구사항에 인터페이스 관련 담당자도 명시해야 한다.

2. 다음 중 시스템 인터페이스 요구사항 명세서 작성 시 포함되는 항목이 아닌 것은?

① 인터페이스 이름 ② 연계 범위 및 내용

③ 송·수신 데이터 ④ 인터페이스 주기

3. 시스템 인터페이스 관련 요구사항 중 기능적 요구사항이 아닌 것은?

① 은행 자금 관리 서비스 연동

② 대·내외 인터페이스 구현 방안

③ 인사 및 조직 정보 연계

④ 고객 대상 채널 업무 연계 기능 구현

4. 다음 중 요구사항 검토 방법이 아닌 것은?

① 개발자 검토 ② 인스펙션

③ 워크스루 ④ 동료 검토

5. 다음 중 인터페이스 요구사항 검증 항목에 포함되지 않는 것은?

① 명확성 ② 일관성

③ 변경 용이성 ④ 가용성

6. 요구사항 명세서 작성자가 명세서의 내용을 직접 설명하고 이해관계자들이 이를 들으면서 결함을 발견하는 형태로 진행하는 요구사항 검토 방법은?

① 워크스루 ② 인스펙션

③ 동료 검토 ④ 개발자 검토

7. 다음 중 송·수신 데이터 식별 시 해당하는 데이터 종류가 아닌 것은?

① 오류 식별 항목 ② 공통 코드

③ 인터페이스 표준 항목 ④ 송·수신 데이터 항목

8. 인터페이스 처리 유형 중 데이터를 매건 단위로 처리할 경우 비용이 많이 발생할 때 사용하는 방식은?

① 실시간 방식

② 지연 처리 방식

③ 일괄 처리 방식

④ 배치 방식

9. 교환되는 데이터의 종류인 인터페이스 표준 항목 중 다음이 설명하는 것은 무엇인가?

> • 시스템 간 연동 시 필요한 공통 정보를 말하며, 인터페이스 ID, 전송 시스템 정보, 서비스 코드 정보, 응답 결과 정보, 장애 정보 등으로 구성한다.
> • 공통적으로 사용되는 코드 정보는 공통 코드로 추출하고 시스템에서 공통으로 관리한다.

① 거래 공통부 ② 전문 종료부

③ 전문 개별부 ④ 시스템 공통부

10. 다음 중 시스템 연계 기술에 대한 설명으로 가장 옳지 않은 것은?

① DB Link는 수신 시스템에서 DB Link를 생성하고 송신 시스템에서 해당 DB Link를 직접 참조하는 방식이다.

② 연계 솔루션은 EAI 서버와 송·수신 시스템에 설치되는 클라이언트(Client)를 이용하는 방식이다.

③ Socket은 클라이언트는 통신을 위한 소켓(Socket)을 생성하여 포트를 할당하고 서버의 통신 요청 시 서버와 연결하고 통신하는 네트워크 기술이다.

④ Web Service는 웹 서비스에서 WSDL과 UDDI, SOAP 프로토콜을 이용하여 연계하는 방식이다.

11. 다음 중 인터페이스 송·수신 데이터 명세화에 대한 설명으로 가장 옳지 않은 것은?

① 인터페이스 시 필요한 데이터를 확인하고 인터페이스 송·수신 데이터 명세를 작성하는 것이다.

② 보안이 중요한 데이터 항목은 인터페이스 설계자 및 운영자 등이 상의하여 암호화 대상을 선택한다.

③ 송·수신 데이터 항목의 데이터 타입, 길이, 필수 입력 여부(Not Null), 식별자 여부를 정의한다.

④ 코드성 데이터 항목에 대해서는 공통 코드 여부, 코드 값 범위를 정의한다.

해설은 158쪽에 있습니다.

12. 다음 중 인터페이스 오류 식별 및 처리 방안 명세화에 대한 설명으로 옳지 않은 것은?

① 오류가 발생하는 영역을 구분할 수 있는 발생 영역 구분자를 정의한다.

② 오류 발생 시 오류 코드, 오류 메시지, 오류 설명 및 해결 방법 등을 정의한다.

③ 오류를 식별하는 오류 코드는 오류 발생 시기와 위치를 참조하여 정의한다.

④ 오류 처리에 참고할 수 있도록 오류 발생 원인 등 상세한 설명을 포함해서 정의한다.

13. 다음 중 시스템 인터페이스 설계서에 대한 설명으로 가장 옳지 않은 것은?

① 시스템 인터페이스 설계서는 시스템의 내 · 외부 인터페이스를 식별하고 인터페이스의 명세를 기술하기 위해 작성한다.

② 시스템 인터페이스 설계서는 시스템 인터페이스 목록과 시스템 인터페이스 정의서로 구성된다.

③ 시스템 인터페이스 목록은 개념 모델링 과정에서 도출한 개체 타입과 관련 속성, 식별자 등에 대한 전반적인 정보를 포함한다.

④ 시스템 인터페이스 정의서는 데이터 송신 시스템과 수신 시스템 간의 데이터 저장소와 속성 등의 상세 내역을 포함한다.

14. 메시지 기반의 비동기형 메시지를 전달하는 방식의 미들웨어로, 이기종 분산 데이터 시스템의 데이터 동기를 위해 많이 사용되는 것은?

① MOM ② DB

③ ORB ④ RPC

15. 코바(CORBA) 표준 스펙을 구현한 객체 지향 미들웨어로, 최근에는 TP-모니터가 가지고 있는 장점인 트랜잭션 처리, 모니터링 등을 추가로 구현한 미들웨어는?

① DB ② MOM

③ WAS ④ ORB

16. 인터페이스 요구사항 검증의 설명으로 가장 옳지 않은 것은?

① 요구사항 검토 기준과 검토 방법, 검토 일정과 참여자 등은 요구사항 검토 계획 단계에서 선정한다.

② 인터페이스 요구사항 검토 방법과 기준 등의 선정 시 프로젝트 규모와 참여 인력, 검토 기간 등을 고려해야 한다.

③ 인터페이스 요구사항 검토는 검토할 인터페이스 요구사항 명세서와 체크리스트 내용을 확인한다.

④ 인터페이스 요구사항 검증 후 검증에 참여했던 사람인 경우에 한해 요구사항을 변경할 수 있다.

▶ **정답 :** 1. ② 2. ③ 3. ② 4. ① 5. ④ 6. ③ 7. ① 8. ② 9. ④ 10. ③ 11. ② 12. ③ 13. ③ 14. ① 15. ④ 16. ④

예·상·문·제·은·행·해·설

1. Section 027

인터페이스 이름은 사업 고유 식별자나 표준 자료 요소 이름 등 사람들이 협의하기 쉽고 간단하며 의미 있는 이름으로 지정해야 한다.

2. Section 027

시스템 인터페이스 요구사항 명세서의 항목에는 인터페이스 이름, 연계 대상 시스템, 연계 범위 및 내용, 연계 방식, 송신 데이터, 인터페이스 주기, 기타 고려사항 등이 있다.

3. Section 027

기능적 요구사항은 시스템이 무엇을 하는지, 어떤 기능을 하는지에 대한 것이고, 비기능적 요구사항은 기능적 요구사항을 제외한 시스템이나 프로젝트 개발과정 등에서 지켜야 할 제약사항을 의미한다.

4. Section 028

요구사항 검토 방법에는 동료 검토(Peer Review), 워크스루(Walk Through), 인스펙션(Inspection) 등이 있다.

5. Section 028

인터페이스 요구사항 검증의 주요 항목에는 완전성(Completeness), 일관성(Consistency), 명확성(Unambiguity), 기능성(Functionality), 검증 가능성(Verifiability), 추적 가능성(Traceability), 변경 용이성(Easily Changeable) 등이 있다.

6. Section 028

- **워크스루** : 검토 회의 전에 요구사항 명세서를 미리 배포하여 사전 검토한 후에 짧은 검토 회의를 통해 결함을 발견하는 형태의 검토 방법
- **인스펙션** : 요구사항 명세서 작성자를 제외한 다른 검토 전문가들이 요구사항 명세서를 확인하면서 결함을 발견하는 형태의 검토 방법

7. Section 030

- **공통 코드** : 시스템들에서 공통적으로 사용하는 코드
- **인터페이스 표준 항목** : 송 · 수신 시스템을 연계하는데 표준적으로 필요한 데이터
- **송 · 수신 데이터 항목** : 송 · 수신 시스템이 업무를 수행하는데 사용하는 데이터

10. Section 031

Socket은 서버가 통신을 위한 소켓(Socket)을 생성하여 포트를 할당하고 클라이언트의 통신 요청 시 클라이언트와 연결하여 통신하는 네트워크 기술이다.

11. Section 031

보안이 중요한 데이터 항목은 법률적 근거와 기업의 개인정보 규정 등을 참고하여 암호화 대상을 선택한다.

12. Section 031

오류 코드는 오류 발생 영역 구분자와 오류 그룹 번호 등을 참조하여 정의한다.

13. Section 032

- 시스템 인터페이스 목록은 연계 업무와 연계에 참여하는 송 · 수신 시스템의 정보, 연계 방식과 통신 유형 등에 대한 정보를 포함한다.
- 개념 모델링 과정에서 도출한 개체 타입과 관련 속성, 식별자 등에 대한 개괄적인 정보를 포함하는 것은 개체 정의서이다.

14. Section 033

- DB(DataBase) : 데이터베이스 벤더에서 제공하는 클라이언트에서 원격의 데이터베이스와 연결하기 위한 미들웨어
- ORB(Object Request Broker, **객체 요청 브로커**) : 객체 지향 미들웨어로 코바(CORBA) 표준 스펙을 구현한 미들웨어
- RPC(Remote Procedure Call, **원격 프로시저 호출**) : 응용 프로그램의 프로시저를 사용하여 원격 프로시저를 마치 로컬 프로시저처럼 호출하는 방식의 미들웨어

15. Section 033

- DB(DataBase) : 데이터베이스 벤더에서 제공하는 클라이언트에서 원격의 데이터베이스와 연결하기 위한 미들웨어
- MOM(Message Oriented Middleware) : 메시지 기반의 비동기형 메시지를 전달하는 방식의 미들웨어
- WAS(Web Application Server) : 정적인 콘텐츠 처리를 하는 웹 서버와 달리 사용자의 요구에 따라 변하는 동적인 콘텐츠를 처리하기 위해 사용되는 미들웨어

16. Section 028

인터페이스 요구사항 검증 후 인터페이스 요구사항의 변경은 공식적인 변경 통제 절차로만 가능하다.