8. 제품 책임자, 스크럼 마스터, 스크럼 팀의 역할 p. 90

표 2-8 제품 책임자, 스크럼 마스터, 스크럼 팀의 역할

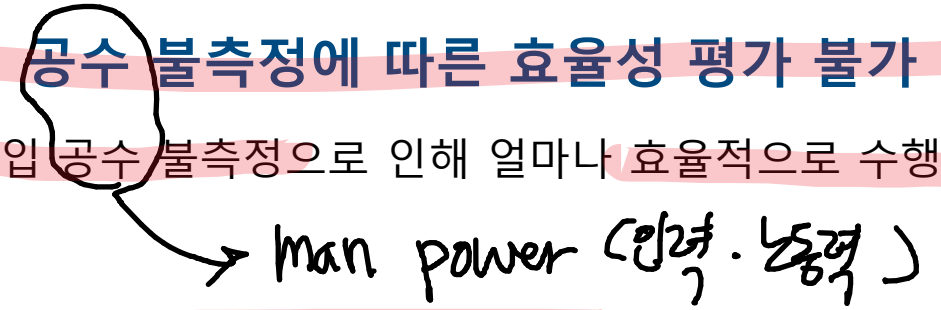
담당자	역할
제품 책임자product owner (사용자의 대표)	• 제품 기능 목록을 만듦. • 비즈니스 관점에서 우선순위와 중요도를 매기고 새로운 항목을 추가함. • 스프린트 계획 수립 시까지만 역할을 수행하고, 스프린트가 시작되면 팀 운영에 관여하지 않음. 스크럼.
스크럼 마스터scrum master	• 제품 책임자를 돕는 조력자 • 업무를 배분만 하고, 일은 강요하지는 않음. • 스크럼 팀이 스스로 조직하고 관리하도록 지원함. • 개발 과정에서 스크럼의 원칙과 가치를 지키도록 지원함. • 개발 과정에 방해될 만한 요소를 찾아 제거함.
스크럼 팀scrum team	• 팀원은 보통 5~9명으로 구성되며, 사용자 요구 사항을 사용자 스토리로 도출하고 이를 구현함. • 기능을 작업 단위로 나누고, 일정이나 속도를 추정해서 제품 책임자에게 알려줌. • 하나의 스프린트에서 생산된 결과물을 제품 책임자에게 시연함. • 매일 스크럼 회의에 참여하여 진척 상황을 점검함. 의의회의.

9. 스크럼 방식의 장점

- 실행 가능한 제품을 통해 사용자와의 충분한 의견 조율 가능
- 일일 회의를 통한 팀원들 간의 신속한 협조와 조율 가능
- 일일 회의 시 직접 자신의 일정 발표를 통한 업무 집중 환경 조성
- 다른 개발 방법론들에 비해 단순하고 실천 지향적
- 팀의 문제를 해결할 수 있는 스크럼 마스터의 능력(역할)
- 프로젝트 진행 현황을 통한 신속하게 목표와 결과 추정 가능, 목표에 맞는 변화 시도 가능

10. 스크럼 방식의 단점

- **추가 작업 시간 필요**
  - 반복 주기가 끝날 때마다 실행 가능하거나 테스트할 수 있는 제품을 만들어야 하기 때문
- **일일 스크럼 회의를 15분 안에 마쳐야 함**
  - 길어지는 회의 시간으로 인한 작업의 방해
- **투입 공수 불측정에 따른 효율성 평가 불가**
  - 투입 공수 불측정으로 인해 얼마나 효율적으로 수행되었는지 모름



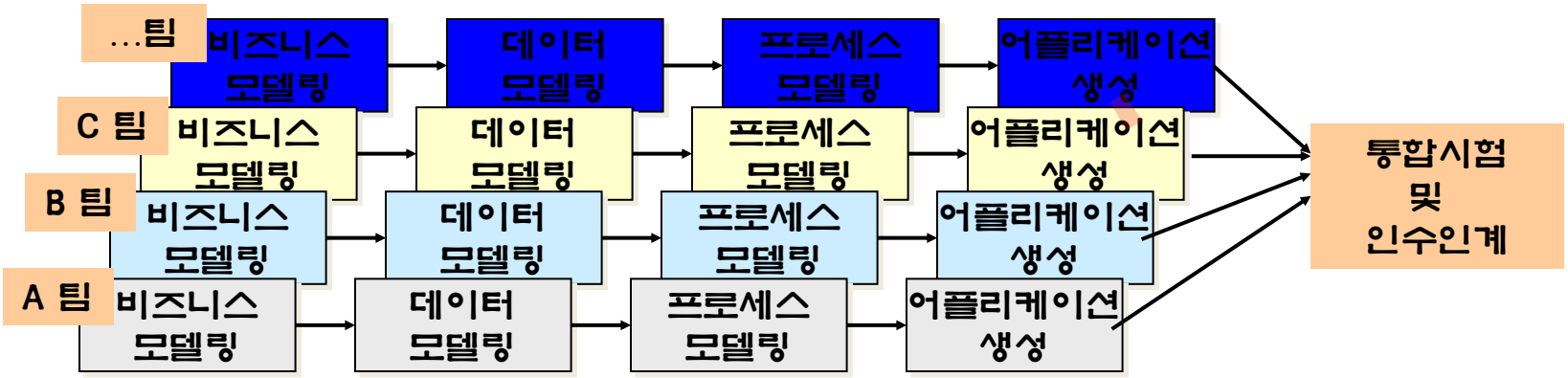
- **프로세스 품질 평가 불가**
  - 프로세스 품질 미평가로 인한 품질 관련 활동이 미약하고 품질의 정도를 알 수 없음

:

Section 11 RAD 모델

* RAD

- **RAD(Rapid Application Development) 모델**
  - 짧은 개발주기 동안 소프트웨어를 개발하기 위한 순차적 프로세스 모델
  - 폭포수 모델의 응용 형태로서 컴포넌트 기반 소프트웨어 개발을 지원
  - 전형적인 RAD 모델의 단계 : 각 주요 기능들을 분리된 팀들에 의해 개발된 후 통합



RAD 모델의 단계

- **비즈니스 모델링(Business Modeling)**
  - 비즈니스 기능(business function) 간의 정보 흐름을 모델링
  - 고려사항
    - 비즈니스 프로세스를 유도하는 정보
    - 처리 결과 생성되는 정보
    - 정보 생성자 및 처리자
    - 정보의 활용

- **데이터 모델링(Data Modeling)**
  - 정보 흐름을 데이터 객체의 집합으로 정제
  - 고려사항
    - 시스템에 의해 처리되는 가장 우선적인 데이터 객체
    - 각 데이터 객체의 구성요소와 객체를 표현하는 속성
    - 각 객체간의 관계와 객체를 변환시키는 프로세스와의 관계

RAD 모델의 단계 (2)

- **프로세스 모델링(Process Modeling)**
  - 데이터 객체가 비즈니스 기능을 담당할 수 있도록 데이터 객체를 변환시키는 프로세스를 모델링
    - 데이터 객체의 추가, 수정, 삭제 및 검색
- **어플리케이션 생성(Application Generation)**
  - 실제 모듈을 구현
    - 신속한 구현을 위해 자동화 도구를 활용
  - 기존 컴포넌트를 재사용 또는 재사용 가능한 컴포넌트를 생성
- **시험 및 인수 단계(Testing and Turnover)**
  - 새롭게 추가한 컴포넌트를 시험
    - 상당수의 기존 컴포넌트들은 시험되어 있음
  - 인터페이스 시험

SW개발. (기능적) module = component (부품.)

부품

Section 12 Design-to-Tools 모델

* 도구맞춤 설계(Design-to-Tools)

- **매우 응급한 경우에만 사용해온 극단적인 방법이다.**
  - 기존 소프트웨어 도구에서 지원하는 기능만 제품에 넣는 방법.
  - 도구에서 지원하지 않는 기능은 제품에서 제외.
    → 여기서 도구는 코드클래스 라이브러리, 코드자동 생성기 쾌속개발언어… 기타 구현시간을 줄이는 소프트웨어 도구
  - 주요단점
    - 제품에 대한 통제력을 잃는다.
    - 원하는 기능 모두를 구현할 수 없을지 모른다.
    - 원하는 기능을 정확하게 구현하지 못할 수도 있다.
    - 상용 소프트웨어 생산업체의 제품전략과 재정전략에 의존하게 된다.
    - 각 도구 개발업체가 제품 사슬을 끊을 수 있다.
      * 제품사슬 (product chain) : 제품이 나오기 위해 차례로 직전 제품을 의존하는 개발 방식

- **도구맞춤 설계로 우수한 개발 속력을 얻을 수는 있지만, 다른 생명주기 모델에 비해 제품 기능을 제어하기가 어렵다.**