유스 케이스를 사용한 요구사항 관리를 위한 추적성 전략

Ian Spence 및 Leslee Probasco
Rational Software 백서

TP 166, 1998

Rational®
the software development company

목차

요약

유스 케이스 모델링 기법을 사용하는 대부분의 상용 응용프로그램의 경우, 프로젝트와 관련된 모든 이해 당사자(stakeholder)가 만족할 수 있는 요구사항 관리 프로세스를 제공하려면 유스 케이스 모델과 기존 요구사항 캡처 기법을 결합해야 합니다. 이 문서에서는 요구사항 관리 전략의 일환으로 유스 케이스 모델링 기법을 채택하는 조직에서 사용할 수 있는 추적성 전략에 대해 살펴봅니다.

소개 및 배경

추적성 항목

요구사항 관리에 대해 논의할 때 특히 RequisitePro 와 같은 도구를 사용하는 경우 일반적인 혼란은 "요구사항"이라는 용어의 남용입니다. 일반적으로 "요구사항"으로 정의되는 항목 이외에 많은 다른 항목 유형의 속성과 항목 간 추적성을 캡처하고 추적해야 합니다. 이러한 기타 추적성 항목에는 문제, 가정, 요청, 용어집 용어, 액터, 테스트 등이 포함됩니다.

이러한 유형의 추적성 항목을 캡처하고 추적함으로써 프로젝트 요구사항을 효과적으로 관리할 수 있습니다.

정의: 추적성 항목

> **항목 간 종속성을 지속적으로 추적하기 위해 다른 텍스트 또는 모델 항목에서 명시적으로 추적해야 하는 모든 텍스트 또는 모델 항목**

RequisitePro 의 경우 이 정의는 다음과 같이 변경될 수 있습니다.

> **RequisitePro 에서 RequisitePro 요구사항 유형의 인스턴스로 표시되는 텍스트 또는 모델 항목**

RequisitePro 는 소프트웨어 개발과 관련된 다양한 유형의 추적성 항목 간 값, 속성 및 추적성 링크를 효과적으로 정의, 캡처 및 추적할 수 있는 도구를 제공합니다.

내재적, 명시적 추적성

모든 개발 프로세스에는 어느 정도의 내재적인 추적성이 존재합니다. 이러한 추적성은 일반적으로 프로세스 내 아티팩트 간 정규 관계를 통해 제공됩니다.

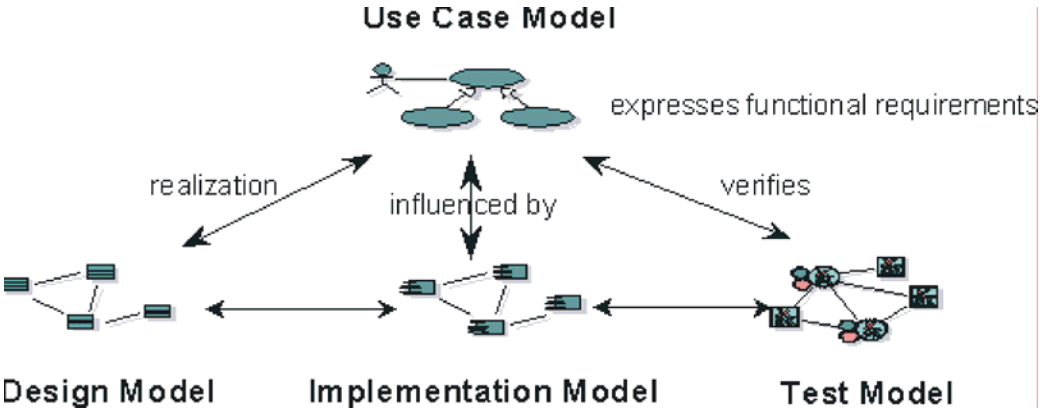
이러한 유형의 내재적 추적성 예제는 다음과 같습니다.

- 이름 지정 규칙
  예를 들어, Fred 디자인 모델의 클래스는 Fred 구현 모델의 클래스로 구현됩니다.
- 모델 간 맵핑 생성
  예를 들어, Rose 의 컴포넌트 보기를 이용하면 Rose 의 논리 보기 내 패키지 및 클래스를 구현 모델의 패키지로 명시적으로 맵핑할 수 있습니다. 이러한 맵핑에는 서로 다른 패키징 전략의 응용프로그램과 모델 간 이름 바꾸기가 포함될 수 있습니다.
- 모델 항목 간 관계
  예를 들어, RUP 에서 디자인 모델의 유스 케이스 실현(realization)은 해당 유스 케이스로 다시 추적됩니다.

- 특정 모델의 요소가 다른 모델의 요소에 내재되어 있는 요구를 충족시키는 방법을 나타내는 다른 관점의 작성
  예를 들어, 디자인 모델의 유스 케이스 실현(realization)은 디자인 모델의 모델 요소가 유스 케이스를 이행하기 위해 협업하는 방법을 나타냅니다. 이 실현은 디자인 모델에서 클래스 및 패키지의 정적 패키징에 대한 유효성을 검증하고 보충하는 유스 케이스 관점을 디자인 모델에 제공합니다.

이러한 모든 예는 추적성 레벨을 제공하며, 개발 모델에 포함된 정보를 사용하여 영향 분석을 수행할 수 있습니다.

아래 그림과 같이 유스 케이스 기반 개발에는 일련의 상호 관련 모델이 포함됩니다.



이 그림은 해당 모델과 이러한 모델 간 내재적 관계를 보여줍니다. 모델 간 관계는 개발 프로세스에 내재적인 추적성 레벨을 제공합니다.

유스 케이스 기반 개발을 수행하는 경우 특정 지원 아티팩트가 유스 케이스 모델을 지원하고 전체 소프트웨어 요구사항 스펙을 정의할 수 있어야 합니다. RUP 의 경우 이 아티팩트는 보충 스펙과 용어집입니다. 또한 프로젝트의 요구, 목적 및 기능에 대한 정의를 포함하는 비즈니스 사례 및 비전 문서도 포함됩니다.

모델 간 관계에는 이러한 지원 아티팩트가 포함되지 않으므로 개발 프로세스에 빌드된 내재적인 추적성이 적용되지 않습니다.

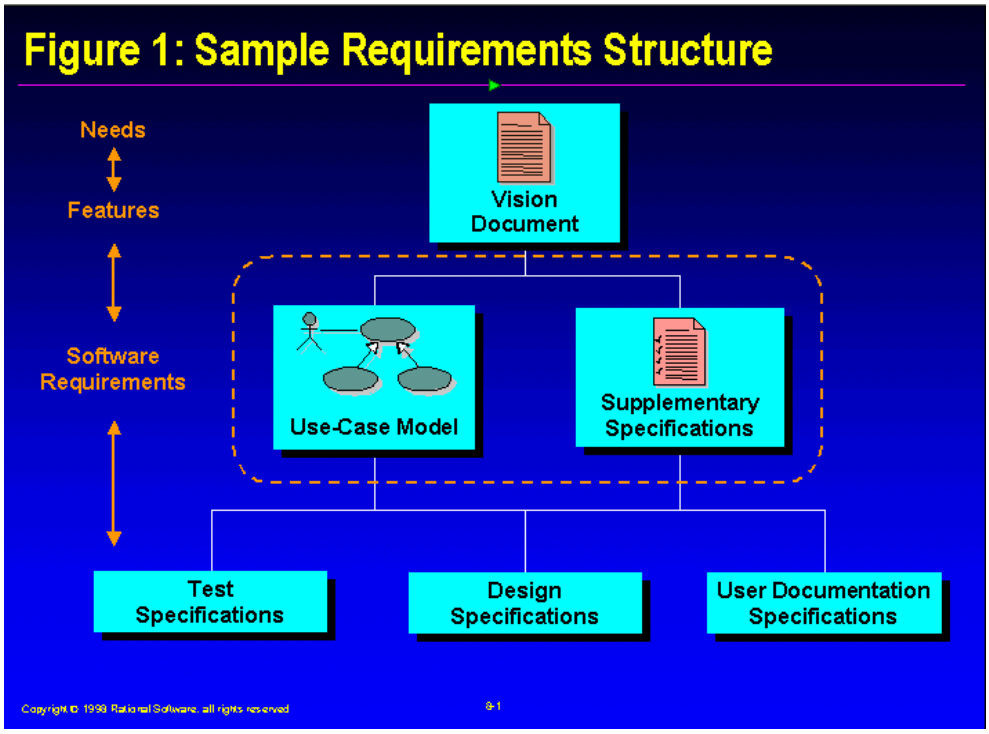
이러한 내재적 관계는 개발 프로세스의 기반이 되며 개발자 작업의 기본 파트로서 빌드되는 데 따른 이점을 얻을 수 있습니다. 이러한 관계는 모델링 프로세스의 핵심으로서, 모델이 성숙됨에 따라 생성 및 유지보수됩니다.

내재적 추적성은 모델링 표기법에서 사용할 수 있는 관계에만 적용됩니다.

그러나 이 밖에 명시적 추적성이 추가로 더 필요합니다.

예를 들어, 요구사항 추적성의 원칙을 채택하려면 요구사항, 모델 항목 및 기타 추적성 항목을 추적성 계층 구조에 통합해야 합니다. 또한 개발 프로세스에 추적성 관계를 더 추가할 수 있습니다.

다음 그림은 비전 문서에 정의된 "기능"과 유스 케이스 모델 및 보충 스펙에 정의된 "소프트웨어 요구사항" 간의 관계를 나타내는 추적성 계층 구조 예제를 보여줍니다. 또한 소프트웨어 요구사항이 테스트 요구사항, 디자인 및 사용자 문서에 추적되는 방법을 보여줍니다.



보충 요구사항(보충 스펙 문서에 정의) 및 디자인과 구현 모델 간의 관계는 모델 간 내재적 추적성으로 나타낼 수 없습니다.

이는 일반적으로 프로젝트에 필요한 명시적 추적성의 추가 레벨를 잘 나타내는 예제입니다. 많은 관계는 유스 케이스 모델과의 내재적 관계로 인해 일련의 모든 모델을 통해 추적됩니다. 보충 요구사항은 보충 스펙의 유스 케이스 모델과 함께 캡처되며, 디자인 모델에서 해당 요구사항을 고려해야 하는 패키지 또는 구현 모델에서 해당 요구사항을 이행해야 하는 컴포넌트와 직접 관련되지 않습니다.

기타 예제에는 다음 요소 간의 관계가 포함됩니다.

- 시스템 기능과 유스 케이스 모델의 기능
- 유스 케이스 모델과 사용자 문서
- 유스 케이스 모델과 테스트 요구사항

요구사항 추적성 프로세스를 설정할 때 필요한 중요한 결정사항 중 하나는 필요한 추적성 레벨과, 이 목적을 충족시키기 위해 필요한 명시적 추적성의 정도입니다. 요구사항, 추적성 및 관리에 대한 접근 방식은 개발 프로세스를 복잡하게 만들거나 제한하기 위해서가 아니라 용이하게 하기 위한 것입니다.

개발 아티팩트에 명시적 추적성을 추가함에 따라 프로젝트의 비용도 현저히 증가하게 됩니다. 이는 특히 이 추가 정보를 제공 및 유지보수하기 위해 장기적으로 비용을 지출해야 하는 경우 특히 중요합니다. 이러한 경우 프로젝트에 적합한 추적성 레벨을 설정해야 하며 유지보수하려는 추가 명시적 추적성에 대한 투자 대비 수익을 얻을 수 있어야 합니다. 개발자는 업무 시간을 추적이 아닌 개발 업무에 투자해야 합니다. 이를 위해서는 프로젝트에 명시적 추적성의 비용을 추가하기 전에 추적성 전략을 확립하고 평가해야 합니다.

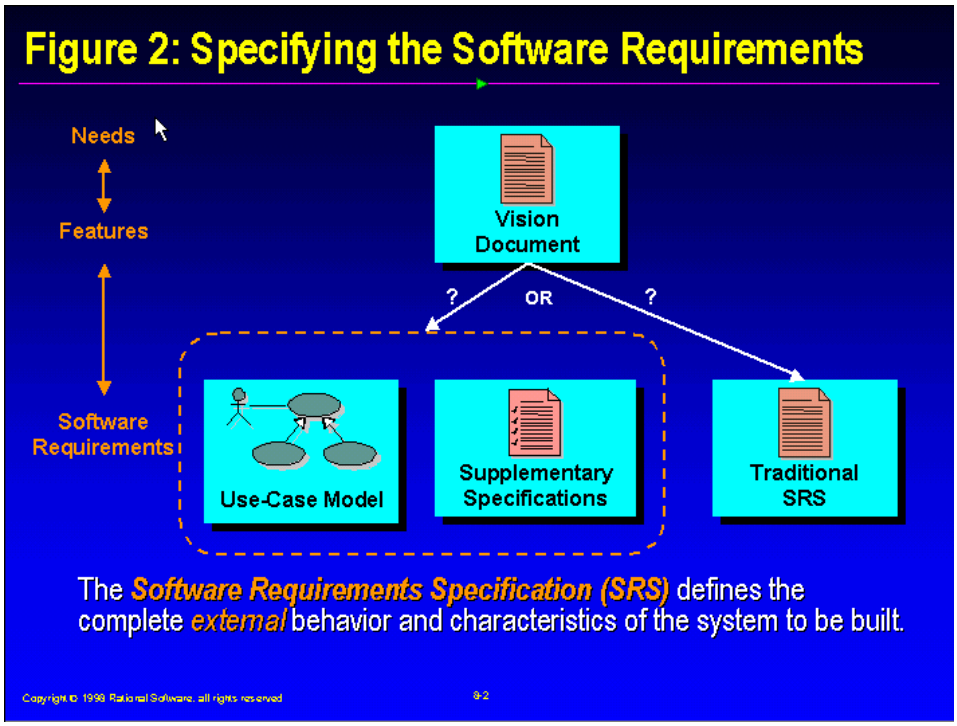
추적성 전략은 소프트웨어 개발 프로세스에 추가할 명시적 추적성 레벨을 정의합니다.

지원 아티팩트 관리

위의 그림 1 은 RUP 의 요구사항 스펙과 관련된 아티팩트를 보여줍니다.

여기서 유의해야 할 사항은 유스 케이스 모델과 보충 스펙이 전체 소프트웨어 요구사항 스펙(SRS)을 구성한다는 것입니다. 즉, 기존 요구사항 관리 기법에서 필요한 정규 소프트웨어 요구사항 스펙 문서가 필요하지 않습니다.

아래 그림 2 는 일반 SRS 문서와 RUP 아티팩트의 일반적인 연관 관계를 보여줍니다. 기존 SRS 는 소프트웨어 요구사항을 문서화하기 위한 대체 방법일 뿐입니다. 두 접근 방식 모두 빌드할 시스템의 전체 외부 동작을 정의하는 소프트웨어 요구사항 스펙을 제공할 수 있음을 이해해야 합니다.



이 관계는 종종 두 가지 요구사항 관리 모델이 공존할 수 없음을 나타내는 것으로 오해됩니다. 사람들은 일반적으로 정규 소프트웨어 요구사항 스펙 문서를 사용하는 기존 요구사항 관리 기법과 유스 케이스 모델 및 보충 스펙을 사용하는 유스 케이스 모델 기반 요구사항 관리 기법 중에서 선택해야 하는 것으로 알고 있습니다. 그러나 특정 상황에서 두 가지 양식의 소프트웨어 요구사항 스펙이 동일한 프로젝트에서 공존해야 하는 경우도 있습니다.

가능한 추적성 전략

다양한 추적성 전략

요구사항 관리 프로세스를 용이하게 하기 위해 여러 추적성 전략을 취할 수 있습니다. RUP 프레임워크에서 작업하는 경우라도 여러 접근 방식을 사용할 수 있습니다. 다음은 가장 일반적으로 사용되는 네 가지 접근 방식이며 모두 작성자가 현재 사용 사실을 확인한 방식입니다.

- 유스 케이스 모델만 사용

이 경우에는 시스템 요구사항을 유스 케이스 모델로만 나타냅니다. 이 접근 방식을 선택하는 프로젝트는 고객과 개발자 간의 밀접한 연관과 신뢰를 바탕으로 합니다.

유스 케이스 모델과 용어집 및 보충 스펙이 시스템의 전체 요구사항을 나타내므로 요구, 제품 기능 또는 소프트웨어 요구사항에 대한 추가 정의는 없습니다.

- 유스 케이스 모델이 기능에 의해 구동

  이 전략은 RUP 에서 권장하는 기본 전략입니다. 유스 케이스 모델과 보충 스펙이 전체 소프트웨어 요구사항 스펙을 나타냅니다. 비전 문서에서 기능을 설명하며 해당 기능은 유스 케이스로 추적됩니다. 유스 케이스 모델에 반영되지 않은 기능은 보충 스펙의 보충 요구사항으로 추적됩니다.

  이러한 경우 주로 유스 케이스 모델에서 기능적 요구사항을 나타냅니다. 유스 케이스 모델 및 보충 요구사항은 요구 및 제품 기능과 용어집 및 보충 요구사항으로 보완됩니다.

- 유스 케이스 모델로 소프트웨어 요구사항 스펙 설명

  이러한 경우 유스 케이스 모델로 기존의 정규 소프트웨어 요구사항 스펙을 나타냅니다. 이 전략은 규정 또는 내부 프로토콜로 인해 기존의 정규 소프트웨어 요구사항 스펙이 적용될 때와 유스 케이스를 통한 개발을 수행하기 위해 유스 케이스 모델이 필요할 때 가장 많이 사용됩니다.

  기능은 기존 요구사항 관리에서와 같이 정규 소프트웨어 요구사항 스펙 문서로 추적되지만 해당 소프트웨어 요구사항은 유스 케이스 모델로 명시적으로 추적됩니다.

- 유스 케이스 모델로 여러 기존 소프트웨어 요구사항 세트 조정

  유스 케이스 모델은 여러 소스의 정규 소프트웨어 요구사항 스펙 세트를 나타내며 단일 공통 시스템의 스펙을 제공합니다.

  이러한 경우 각 이해 당사자(stakeholder)는 고유한 제품 기능 및 소프트웨어 요구사항 세트를 가집니다. 이러한 기능 및 요구사항은 해당 비전 및 소프트웨어 요구사항 스펙 문서에서 자세히 설명합니다. 이러한 여러 시점과 상충되는 요구사항은 시스템 기능을 지정하는 단일 유스 케이스 모델에서 조정됩니다. 이 전략은 독립적인 여러 이해 당사자와의 관계가 필요할 때 매우 효과적입니다.

옵션 1 을 제외한 모든 경우에서 유스 케이스 모델과 기존 요구사항 추적성 프로세스의 요소가 결합됩니다.

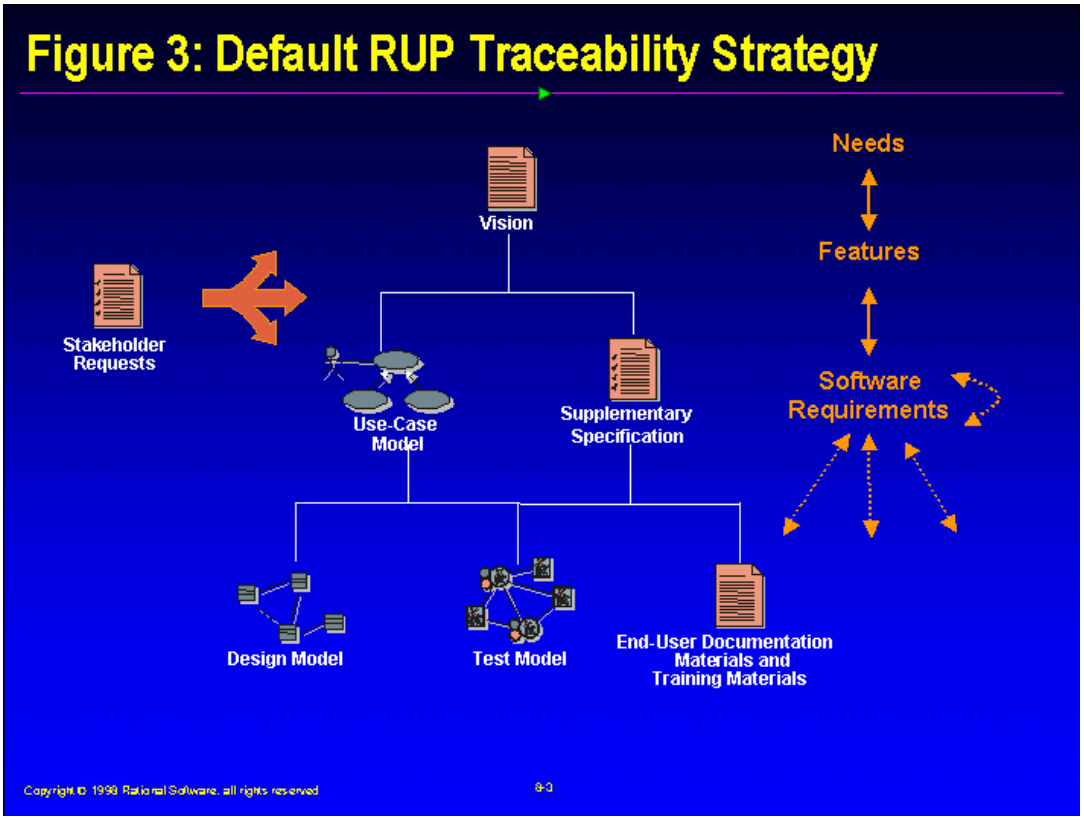
물론 다른 여러 옵션도 가능하며 그 중에는 유스 케이스 모델이 전혀 없는 경우도 있습니다. 이 옵션을 "유스 케이스 모델을 사용하지 않는" 접근 방식이라고 합니다.

이 내용과 기타 접근 방식은 아래 추적성 전략 카탈로그에서 자세히 설명합니다.

혼성 접근 방식 채택

해당 설명에서 알 수 있듯이 위에서 유스 케이스 모델만 사용하거나 유스 케이스 모델을 전혀 사용하지 않는 두 가지 극단적인 솔루션은 한 가지 양식의 요구사항 캡처만 허용하는 순수주의 관점을 나타냅니다. 이 두 가지 솔루션은 모두 모든 프로젝트 및 모든 이해 당사자(stakeholder) 집단에게 일관된 단일 접근 방식을 적용할 수 있는 것으로 가정합니다. 두 가지 접근 방식은 모두 성공적인 결과를 가져왔지만 "실제" 프로젝트에서 발생하는 복잡한 모든 상황과 이해 당사자 간 여러 관계를 올바르게 처리하지 못하고 유연성이 부족하여 평판이 떨어졌습니다.

RUP 에서 권장하는 추적성 계층 구조는 다음과 같습니다.



이 구조는 위의 "유스 케이스 모델이 기능에 의해 구동"되는 옵션이며 일반적으로 가장 효율적인 추적성 전략이지만 RUP 를 채택한 경우라도 이 접근 방식이 항상 가장 큰 효과를 나타내는 것은 아닙니다.

유스 케이스 모델링을 기능적 소프트웨어 요구사항 스펙에 대한 단일 메커니즘으로 사용하여 문제가 발생할 수 있는 경우의 예는 다음과 같습니다.

- 많은 요구사항 소스가 서로 모순되는 경우(예를 들어, 모순되는 여러 요구사항을 추적하고 관리해야 하는 경우)
- 기존의 요구사항 캡처 프로세스를 준수해야 하는 조직에서 프로젝트를 수행하는 경우
- 합의된 단일 요구사항 모델을 만들기 위한 모델링 워크샵에 이해 당사자(stakeholder)를 참석시키는 데 문제점이 있는 경우
- 유스 케이스 모델링을 사용하여 기존 요구사항 캡처 프로세스의 제한조건 범위 내에서 객체 지향 소프트웨어를 개발하려는 경우
- 이해 당사자가 유스 케이스 모델에서 직접 해당 기능적 소프트웨어 요구사항 레벨 기대치를 모두 표현할 수 없거나 표현하지 않으려는 경우
- 고객이 기존 소프트웨어 요구사항 세트의 양식으로 전달할 제품을 정의한 경우 이는 개발이 입찰에 의해 이루어질 때 매우 일반적인 상황이며 이러한 경우 기존 요구사항에 대한 설명은 제공할 계약의 일부로 포함됩니다.

어떤 접근 방식을 채택할 것이가는 각 프로젝트 및 개발 조직의 컨텍스트에서 결정해야 합니다. 이 문제점을 해결할 수 있는 완벽한 단일 솔루션은 없으며 모든 프로젝트를 단일 요구사항 관리 접근 방식에 적용하려는 것은 어리석은 시도입니다.

RUP 는 구성 가능한 프로세스이므로 "유스 케이스 모델이 없는" 접근 방식을 제외한 이 문서의 모든 추적성 전략을 수행할 수 있습니다(이 옵션은 RUP 의 유스 케이스 기반 특성으로 인해 사용할 수 없습니다). 채택할 접근 방식은 RUP 개발 사례 작성 시 결정해야 합니다.

추적성 전략 카탈로그 정보

추적성 전략을 정의하려면 다음과 같은 추적성 항목을 정의하고 분류하기 위한 메커니즘이 필요합니다.

정의: 추적성 유형

공통 특성 및 속성을 기반으로 한 추적성 시간 분류(예: 요구, 제품 기능, 유스 케이스, 소프트웨어 요구사항, 테스트 요구사항, 액터, 용어집 용어 등)

참고: RequisitePro 의 경우 추적성 유형은 요구사항 유형으로 표시됩니다.

따라서 추적성 전략 설정에는 다음과 같이 밀접한 관계의 세 가지 활동이 포함됩니다.

- 추적성 항목을 정의하는 데 필요한 추적성 유형 세트 식별
- 이러한 추적성 유형 간의 올바른 추적성 관계 식별
- 프로젝트 요구사항을 효과적으로 관리하기 위해 추적성 항목에 필요한 속성 식별

추적성 전략 카탈로그는 알려진 추적성 항목 및 해당 추적성 관계의 세트를 문서화함으로써 처음 두 단계를 용이하게 합니다. 세 번째 활동의 경우 추적성 유형에 대한 올바른 속성 정의가 현재 이 문서의 범위에서 벗어나므로 이 카탈로그가 적용되지 않습니다.

카탈로그에서 설명하는 추적성 전략은 모두 동일한 추적성 유형 기본 세트의 서브세트를 이용합니다.

- 요구
- 제품 기능
- 소프트웨어 요구사항(기능 및 비기능적)
- 용어집 항목
- 유스 케이스
- 유스 케이스 섹션
- 액터

참고: 일반적으로 유스 케이스 모델링의 경우 유일한 소프트웨어 요구사항은 보충 스펙에 정의된 보충 요구사항입니다.

유스 케이스 모델의 컴포넌트 파트와 모든 기존 추적성 유형 간을 추적할 수 있는 경우 프로젝트에 사용할 수 있는 추적성 전략 수를 알 수 있습니다.

추적성 항목 간에는 다음과 같은 두 가지 레벨의 추적성이 존재합니다.

- 기본 추적성

  기본 추적성은 선택한 모든 추적성 전략에 적용됩니다. 이 추적성은 추적성 유형의 특성에 내재되어 있으며 유스 케이스와 유스 케이스 섹션 간의 관계 또는 유스 케이스와 액터 간의 관계에 적용됩니다.

  아래 추적성 전략 섹션의 개요 다이어그램에서, 이 기본 추적성은 각 전략에 대해 반복되지는 않지만 기본적으로 해당 추적성 전략에 포함됩니다.

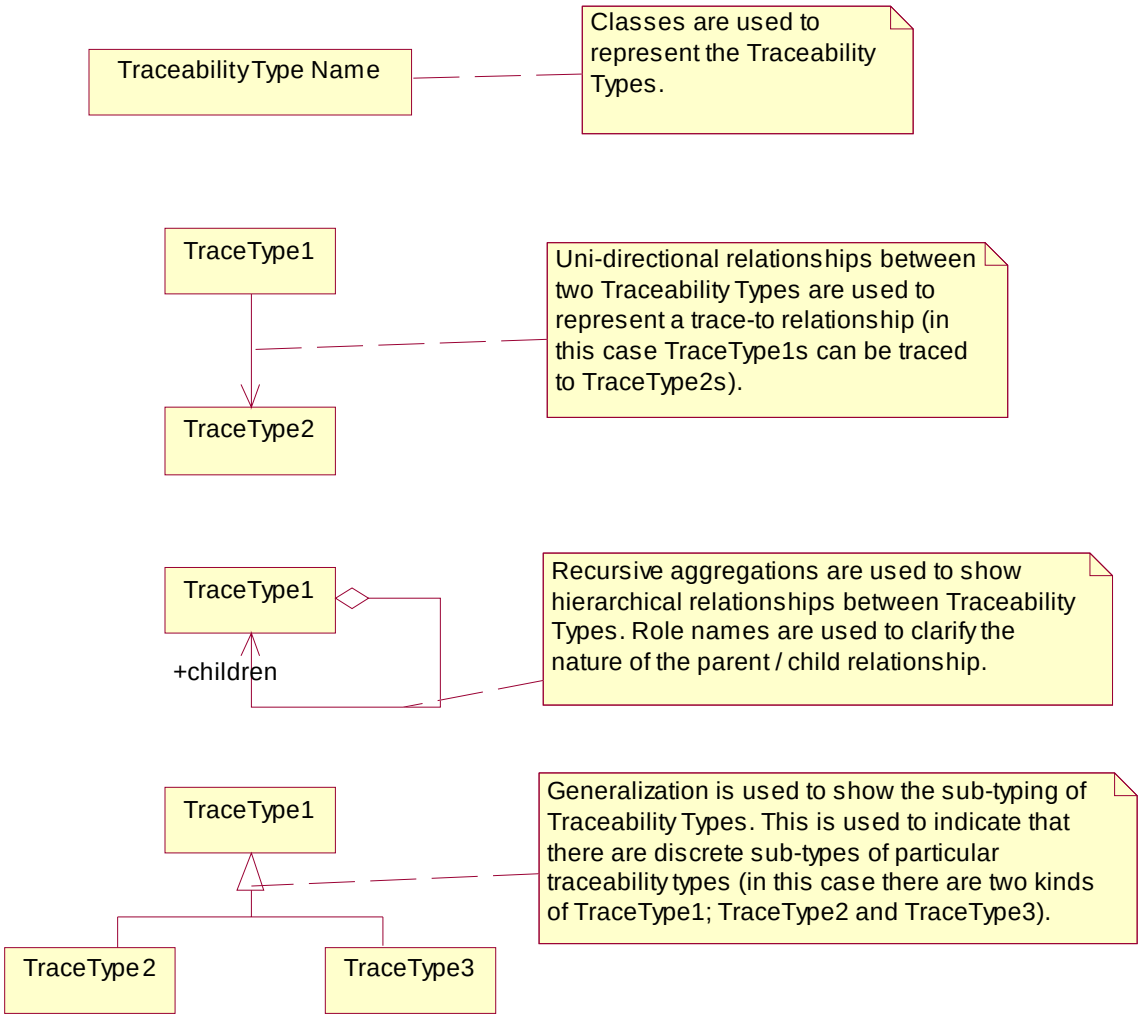
- 확장 추적성

이 추적성은 특정 추적성 전략 중 하나를 지원하기 위해 사용됩니다. 이 추적성은 보다 주관적이며 추적성 전략에 따라 다릅니다.

추적성 전략 카탈로그

다이어그램 표기법

추적성 유형과 해당 추적성 관계는 UML(Unified Modeling Language) 다이어그램으로 표시됩니다. 아래 그림은 UML 사용법을 이러한 컨텍스트에서 해석하는 방법을 보여줍니다.



다이어그램을 완전히 이해하려면 RequisitePro 에서 정의를 구현할 때 사용되는 구현 맵핑을 알고 있어야 합니다. 아래 표는 다이어그램 표기법을 RequisitePro 프로젝트에 맵핑할 수 있는 방법에 대해 설명합니다.

다이어그램 표기법	RequisitePro 맵핑
클래스/추적성 유형	요구사항 유형
관계	RequisitePro "추적" 관계
집계	계층 구조 요구사항
일반화	추가 "하위 분류" 속성을 추가하여 수퍼 요구사항 유형 분류

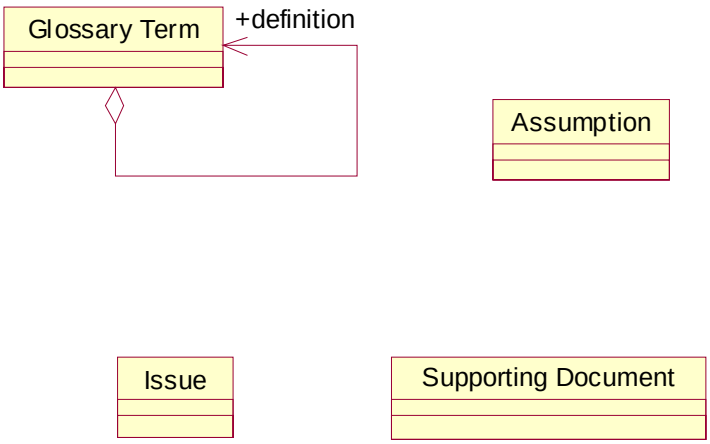
참고: RequisitePro 에서는 모든 추적성 항목을 임의의 기타 항목으로 추적할 수 있습니다. 추적성 전략이 정의하는 내용은 프로젝트의 요구사항 관리 전략에서 가장 중요한 의미있는 추적성 링크입니다.

추적성 유형 지원

설명

이 섹션에서는 선택한 추적성 전략을 지원하는 데 사용할 수 있는 지원 추적성 유형 세트를 정의합니다.

개요



추적성 유형

추적성 유형	설명
용어집 용어	이 추적성 유형은 용어집 용어 및 해당 정의를 나타내는 추적성 항목을 정의합니다. 용어집은 채택한 추적성 전략에 관계 없이 필요하므로 이 유형은 지원 추적성 유형 세트에 포함됩니다.
문제	이 추적성 유형을 사용하면 RequisitePro 에서 추적할 문제를 나타내는 추적성 항목을 추가할 수 있습니다. 이 문제는 영향을 받는 추적성 항목과 연관될 수 있습니다. 문제 추적성 유형을 사용하는 한 예는 용어집 항목과 연관된 문제를 추적하는 것입니다. 정의가 명확하지 않거나 아직 규정되지 않은 경우 RequisitePro 에서 문제가 발생하여 포함될 수 있습니다. 이러한 경우 해당 문제를 간과하지 않고 해결되지 않은 문제와 관련된 모든 용어집 항목에 대해 보고하는 보기가 빌드될 수 있습니다. 이 추적성 유형을 올바르게 사용하는 또 다른 예는 유스 케이스 및 기타 개발 아티팩트를 검토할 때 발생한 문제를 추적하는 것입니다.
가정	이 추적성 유형을 사용하면 자신이 가정한 내용을 추적할 수 있습니다. 이 가정은 영향을 받는 추적성 항목과 연관될 수 있습니다.
지원 문서	이 추적성 유형을 사용하면 원하는 문서를 추적성 계층 구조에 추가할 수 있습니다. 이 유형은 특히 다른 추적성 항목의 의미 또는 목적을 규정하는 기존 예제 또는 문서를 포함할 때 유용합니다. RequisitePro 의 유연한 추적성 메커니즘을 통해 지원 문서와 모든 유형의 모든 추적성 항목을 연관시킬 수 있습니다. 지원 문서 유형을 사용하는 예는 세부 EDI 메시지 스펙을 용어집의 지원 정보로 포함하거나 메시지를 사용할 유스 케이스의 부록으로 포함하는 경우입니다.

기본 추적성

추적성 링크	설명
용어집 용어 대 용어집 용어	이 관계에서는 단일 추적성 유형을 사용하여 용어집 용어의 이름과 해당 정의를 모두 캡처할 수 있습니다.
지원 추적성 유형 대 기타 추적성 유형	이 지원 추적성 유형은 선택한 추적성 전략과 관련된 모든 기타 추적성 유형으로 추적할 수 있습니다.

유스 케이스 모델을 사용하지 않음

설명

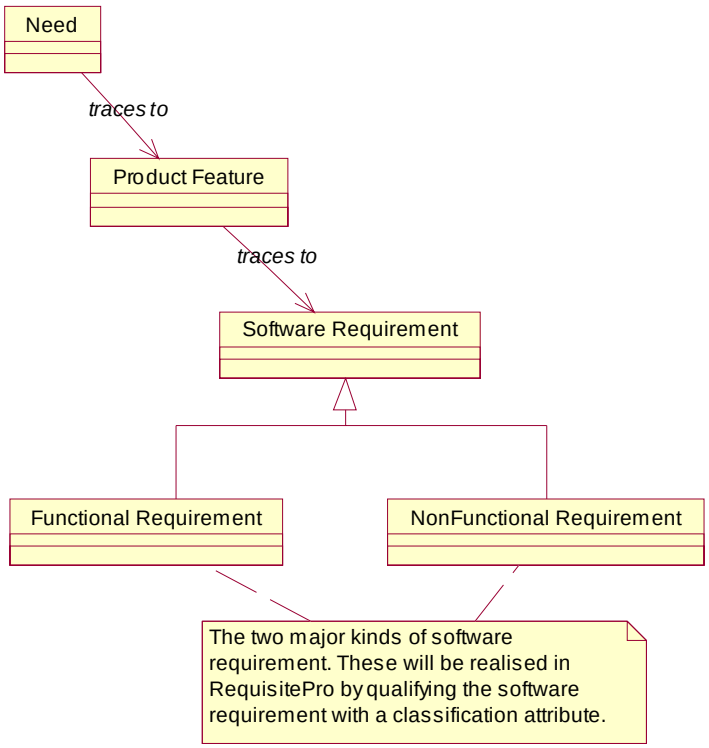
유스 케이스 모델이 없는 경우입니다. 요구는 제품 기능을 만들고 제품 기능은 정규 소프트웨어 요구사항 스펙에 문서화되는 소프트웨어 요구사항을 만듭니다.

일반적으로 "유스 케이스 모델 따위는 필요하지 않다"고 주장하는 프로젝트 관리자가 사용합니다.

특성

특성	값	주석
명시적 추적성	높음	유스 케이스를 사용하지 않고 요구사항 관리 기법을 실행하는 프로젝트에서는 일반적으로 추적성 유형 간에 상위 레벨의 명시적 추적성을 유지보수합니다.
신뢰도	낮음	
책임성	높음	
형식성	높음	
완전성	낮음	기존 소프트웨어 요구사항 세트의 완전성을 평가하는 것은 매우 어렵습니다.
문서 세트	대규모	문서 세트는 일반적으로 인치가 아닌 피트 단위로 측정됩니다.
초점	계약	요구사항 캡처 프로세스의 초점은 해결할 문제점과 제안 솔루션에 대한 이해의 공유가 아닌 고객과 개발자 간에 법적으로 강제할 수 있는 계약을 확립하는 것입니다.
이해 용이성	낮음	요구사항 문서는 사용자 커뮤니티와 개발자가 모두 액세스할 수 없는 경우가 종종 있습니다. 이 문서는 일반적으로 검토자에게 별다른 의미를 부여하지 않는 유형 또는 기능 영역을 기준으로 그룹화되는 여러 개별 품목으로 구성됩니다.
프로세스	일반적으로 폭포수형	기존 요구사항 캡처 기법은 일반적으로 폭포수형 개발 프로세스의 일부로 실행됩니다. 요구사항 서브세트에 대한 컨텍스트가 부족하고 서브세트의 완전성을 평가하기 어려운 경우 반복적이고 점진적인 개발 프로세스를 쉽게 채택할 수 없습니다.
개발 스타일	기능적 분해	유형 또는 기능 영역에 따라 요구사항을 그룹화하는 경우 요구사항을 솔루션으로 변환할 때 지속적인 기능적 분해로 연결될 수 있습니다.

추적성 개요



추적성 유형

요구사항 유형	설명
요구	구매 또는 사용을 입증하기 위해 이행해야 하는 비즈니스 또는 운영상의 문제점(기회). 목적 또는 목표라고도 합니다.
제품 기능	요구를 직접 이행하는 시스템 기능 또는 특성. 일반적으로 시스템의 "핵심 이점"으로 인식됩니다.
소프트웨어 요구사항	빌드 대상 소프트웨어가 준수해야 하는 조건 또는 기능.

추적성 *요약*

추적성 링크	설명
제품 기능으로 요구 추적	각 요구는 기능 세트로 실현됩니다. 이 관계를 사용하면 각 기능의 비즈니스 이점을 추적할 수 있습니다.
소프트웨어 요구사항으로 제품 기능 추적	각 기능은 소프트웨어 요구사항 세트로 실현됩니다. 이 관계를 사용하면 각 소프트웨어 요구사항의 비즈니스 이점을 추적할 수 있으며 제품 기능 레벨에서 소프트웨어 요구사항의 범위를 관리할 수 있습니다.

이점 및 단점

장점:

- 쉽게 이해할 수 있습니다.
- 법률 계약에 적합한 것으로 알려져 있습니다(제공된 소프트웨어가 지정된 요구사항을 충족시킬 수 있는지 여부와 관련되어 현재 진행 중인 모든 재판 참조).
- 많은 표준 프로세스에서 권장합니다.
- 세부적인 정규 하위 레벨 추적성을 사용할 수 있습니다.
- "최신" 아이디어 도입으로 현 상태를 혼란에 빠뜨리지 않습니다.

단점:

- 요구사항 단계에서 쉽게 지체될 수 있어 요구사항 캡처를 완료하기 어렵습니다.
- 이 양식으로 표현된 요구사항을 이해하기 어렵습니다.
- 요구사항 변경에 대한 영향 분석을 평가하기 어렵습니다.
- 개별 요구사항의 컨텍스트가 없습니다.
- 유지보수 비용이 높습니다. 내재적 추적성이 부족하여 프로젝트에서 많은 명시적 추적성 관계를 유지보수하기 위한 비용이 소요됩니다.
- 컨텍스트 부족으로 의미있는 요구사항 서브세트를 식별하기 어렵습니다. 또한 이로 인해 범위 관리와 점진적인 제품 전달이 보다 어려워집니다.

예제

요구사항 추적성과 관련하여 유스 케이스 모델이 없는 접근 방식은 여러 비즈니스 영역의 많은 프로젝트에서 널리 사용됩니다. 많은 조직에서 정규 계약 협상의 기반으로 기존의 정규 소프트웨어 요구사항 스펙을 필요로 합니다. 이러한 경우 해당 조직은 기존 요구사항 관리 접근 방식을 해당 프로젝트에 적합한 유일한 접근 방식으로 인식하게 됩니다.

유스 케이스 모델만 사용

설명

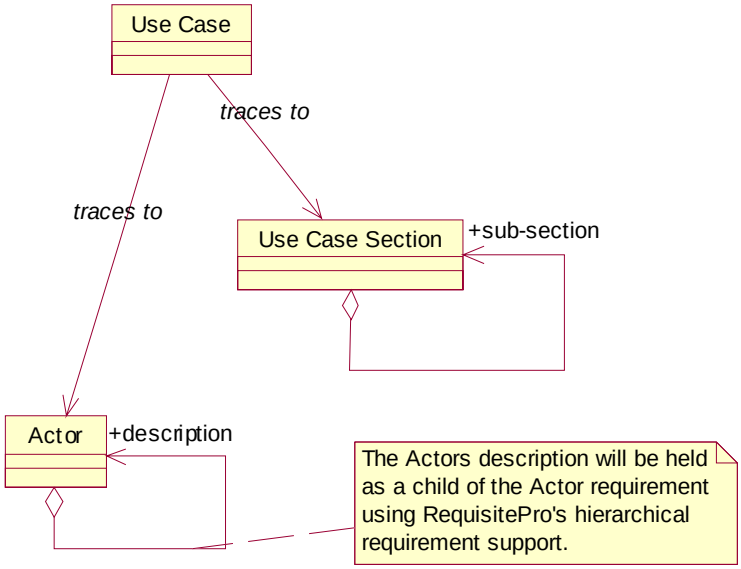
"유스 케이스 모델은 내 요구사항입니다." 고객과 개발자 간의 관계가 밀접하고 신뢰 레벨이 높은 프로젝트의 경우 일반적으로 이 접근 방식을 채택합니다. 이 접근 방식은 보통 개발자가 고객과 협력하거나 고객의 승인을 얻어 유스 케이스 모델을 개발함으로써 요구사항을 명확하게 나타내거나 쉽게 이해할 수 있도록 하려는 프로젝트에 사용됩니다. 이러한 프로젝트의 특징은 별도 설명이 그다지 필요하지 않은 내부용 프로젝트입니다.

이 경우에는 시스템 요구사항을 유스 케이스 모델로만 나타냅니다. 유스 케이스 모델, 용어집 및 보충 스펙이 시스템의 전체 요구사항을 나타냅니다.

특성

특성	값	주석
명시적 추적성	낮음	명시적 추적성이 필요하지 않습니다. 유스 케이스 기반 접근 방식을 채택함으로써 제공되는 내재적 추적성만으로도 충분합니다. 일반적으로 명시적 추적성이 유지보수되지 않으므로 요구사항 관리 도구가 사용되지 않습니다.
신뢰도	높음	요구 또는 기능 레벨의 분석이 적다는 것은 올바른 시스템을 제공하는 데 있어 이해 당사자(stakeholder)가 유스 케이스 모델 개발자를 많이 신뢰함을 의미합니다.
책임성	낮음	
형식성	낮음	
완전성	낮음	유스 케이스 모델만으로도 소프트웨어 요구사항 스펙을 완전하게 나타낼 수 있지만 이해 당사자 요구에 대한 추적성이 부족한 경우 불완전하거나 지나치게 복잡한 시스템이 생성될 수 있습니다.
문서 세트	소규모	이 접근 방식에는 최소 문서 세트가 포함됩니다.
초점	사용자	유스 케이스 모델에 사용자 관점이 적용됩니다.
이해 용이성	높음	모든 프로젝트 이해 당사자가 유스 케이스 모델을 쉽게 이해할 수 있습니다.
프로세스	일반적으로 반복적이고 점진적입니다.	유스 케이스가 반복적이고 점진적인 개발을 용이하게 하는 컨텍스트에서 소프트웨어 요구사항을 처리합니다(유스 케이스는 올바른 전달 단위를 제공합니다). 폭포수형 개발 프로세스에서도 유스 케이스를 사용할 수 있습니다.
개발 스타일	일반적으로 객체 지향적	유스 케이스는 모든 개발 스타일에 적용할 수 있지만 일반적으로 객체 지향 소프트웨어를 개발하는 데 사용됩니다. OO 스타일이 채택되지 않는 경우 높은 수준의 명시적 추적성이 필요합니다.

추적성 개요



추적성 유형

추적성 유형	설명
유스 케이스	이 추적성 유형은 유스 케이스를 나타내는 추적성 항목을 정의합니다.
유스 케이스 섹션	유스 케이스 섹션을 사용하면 유스 케이스 섹션을 추적성 계층 구조에 포함시킬 수 있습니다. 이러한 경우 개별 플로우와, 유스 케이스를 구성하는 다른 특성으로 추적할 수 있습니다. 서브섹션에 대한 계층 구조 관계를 통해 각 섹션의 개별 요소를 캡처할 수 있습니다. 예를 들어, 이를 통해 전제 조건 섹션을 구성하는 개별 전제 조건을 식별할 수 있습니다. 참고: 경우에 따라 유스 케이스 이벤트 플로우에서 개별 소프트웨어 요구사항을 식별해야 하는 경우도 있지만(이 경우 소규모 섹션) 유스 케이스가 안정화될 때까지는 이러한 시도를 해서는 안됩니다.
액터	이 추적성 유형은 액터를 나타내는 추적성 항목을 정의합니다.

추적성 요약

추적성 링크	설명
유스 케이스 대 유스 케이스 섹션	각 유스 케이스는 유스 케이스 섹션 세트로 구성됩니다. 이 관계를 통해 유스 케이스를 구성하는 유스 케이스 섹션과 해당 유스 케이스를 추적할 수 있습니다.
유스 케이스 섹션 대 유스 케이스 섹션	일부 복잡한 유스 케이스 섹션은 여러 서브섹션으로 구성됩니다. 예를 들어, 이벤트 플로우가 여러 서브플로우로 구성되거나 전제 조건 섹션이 많은 전제 조건으로 구성될 수 있습니다.
유스 케이스 대 액터	이 관계를 통해 유스 케이스와 관련된 액터와 해당 유스 케이스를 확인할 수 있습니다.
액터 대 액터	이 관계에서는 단일 추적성 유형을 사용하여 액터의 이름과 간략한 해당 설명을 모두 캡처할 수 있습니다.

이점 및 단점

장점:

- 최소 문서 세트
- 요구사항 관리에 최소 노력 필요
- 범위 관리, 영향 분석 및 점진적 개발을 위한 올바른 지원
- 유스 케이스를 쉽게 이해할 수 있습니다.

단점:

- 이해 당사자(stakeholder) 요구와의 관계가 없습니다. 솔루션 정의를 시작하기 전에 문제점 분석을 위한 실제 시도가 이루어지지 않습니다.
- 유스 케이스 모델만을 기반으로 계약을 승인할 수 없다고 생각하는 사람들이 있습니다.
- 요구 분석을 하지 않으면 유스 케이스 모델로 적합한 솔루션을 설명하기가 어렵습니다. 상상의 나래를 펼치면서 유스 케이스를 작성하기 쉽습니다.
- 정기 릴리스를 수행하는 경우 유스 케이스보다 상위 레벨의 정보 없이는 제품을 관리하고 이해 당사자의 기대를 지속적으로 관리하기 어렵습니다.

예제

이 접근 방식은 일반적으로 개발자와 사용자가 긴밀한 협력 관계를 갖는 소규모 비정규 내부 프로젝트에 사용됩니다.

유스 케이스 모델이 제품 기능 정의

설명

이러한 경우 유스 케이스 모델링은 기본 요구사항 도출 방법으로 사용되며 유스 케이스 모델은 시스템이 제공할 제품 기능의 정의와 소프트웨어 요구사항을 나타냅니다.

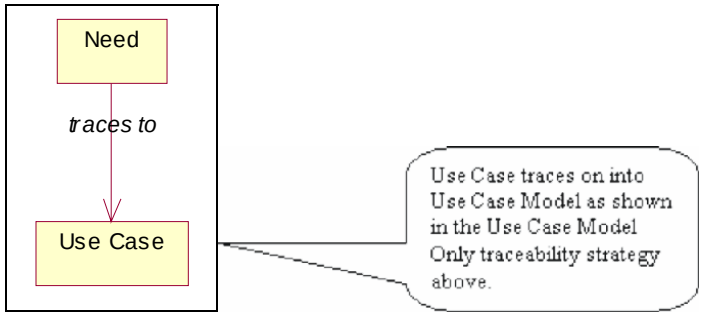
이 옵션은 확장되지 않으므로 라이프사이클이 짧은 소규모 개발 프로젝트에 적합합니다. 각 유스 케이스가 시스템 기능을 나타내더라도 실제로 유스 케이스보다 많은 기능이 존재하며 해당 기능 중 상당수가 많은 유스 케이스에 영향을 줍니다. 시스템이 발전함에 따라 각 릴리스의 새 기능은 새 유스 케이스의 양식을 나타내지 않습니다.

특성

이는 이전의 "유스 케이스 모델만 사용"하는 접근 방식의 변형입니다. 앞에서 이 접근 방식을 설명할 때는 몇 가지 차이점에 대해서만 언급했습니다.

특성	값	주석
명시적 추적성	낮음	유스 케이스 모델만 사용하는 접근 방식과 같습니다. 즉, 요구사항 세트가 적고 그에 따라 추가 추적성도 적으므로 명시적 추적성 또한 그다지 많이 필요하지 않습니다.
신뢰도	중간/높음	유스 케이스 모델에 요구를 추가하면 유스 케이스 모델만 사용하는 경우보다 전략의 신뢰도가 약간 낮아집니다.
책임성	낮음	
형식성	낮음	
완전성	중간	유스 케이스 모델만으로도 소프트웨어 요구사항 스펙을 완전하게 나타낼 수 있지만 이해 당사자(stakeholder) 요구에 대한 추가 추적성을 통해 유스 케이스 모델의 적용 가능성에 대한 유효성을 검증할 수 있습니다.
문서 세트	소규모	이 접근 방식에는 최소 문서 세트가 포함됩니다 (요구를 포함하는 비전 문서와 유스 케이스 모델).
초점	사용자	유스 케이스 모델과 요구 모두 사용자 관점이 적용됩니다.
이해 용이성	높음	모든 프로젝트 이해 당사자가 요구와 유스 케이스 모델을 모두 쉽게 이해할 수 있습니다.
프로세스	일반적으로 반복적이고 점진적입니다.	유스 케이스 모델만 사용하는 접근 방식과 같습니다.
개발 스타일	일반적으로 객체 지향적	유스 케이스 모델만 사용하는 접근 방식과 같습니다.

추적성 개요



추적성 유형

추적성 유형	설명
요구	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
유스 케이스	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.

추적성 요약

추적성 링크	설명
요구 대 유스 케이스	이러한 경우 요구가 직접 유스 케이스로 추적됩니다. 제품 및 범위 관리 시 유스 케이스가 제품 기능의 역할을 수행할 수 있는 것으로 가정합니다.

이점 및 단점

이 접근 방식은 "유스 케이스 모델만 사용"하는 전략과 매우 유사하므로 장단점 또한 다음과 같은 추가사항 및 제한사항을 제외하고는 모두 동일합니다.

장점:

- 이러한 경우 유스 케이스 모델은 이해 당사자(stakeholder) 요구와 관련되므로 유스 케이스 모델의 적합성을 평가하는 데 도움이 됩니다.

단점:

- 초기 프로젝트 단계에서는 유스 케이스가 시스템 기능을 정의하는 것처럼 보이지만 프로젝트가 진행됨에 따라 두 개념은 달라집니다.
- 유스 케이스는 기능이 아닙니다. 따라서 시간과 노력을 줄일 수 있는 것처럼 보이는 전략이 얼마 안 있어 유지보수가 불가능한 골칫거리로 전락하게 됩니다.

예제

이 추적성 전략은 소규모 내부 프로젝트에서 많이 시도되지만 확장성 및 장기적인 제품 발전 문제점으로 인해 권장되는 접근 방식은 아닙니다. 이해 당사자(stakeholder) 요구에 대한 추적성으로 유스 케이스 모델을 보충하려는 경우 "유스 케이스 모델이 기능에 의해 구동"되는 전략을 권장합니다.

유스 케이스 모델이 기능에 의해 구동

설명

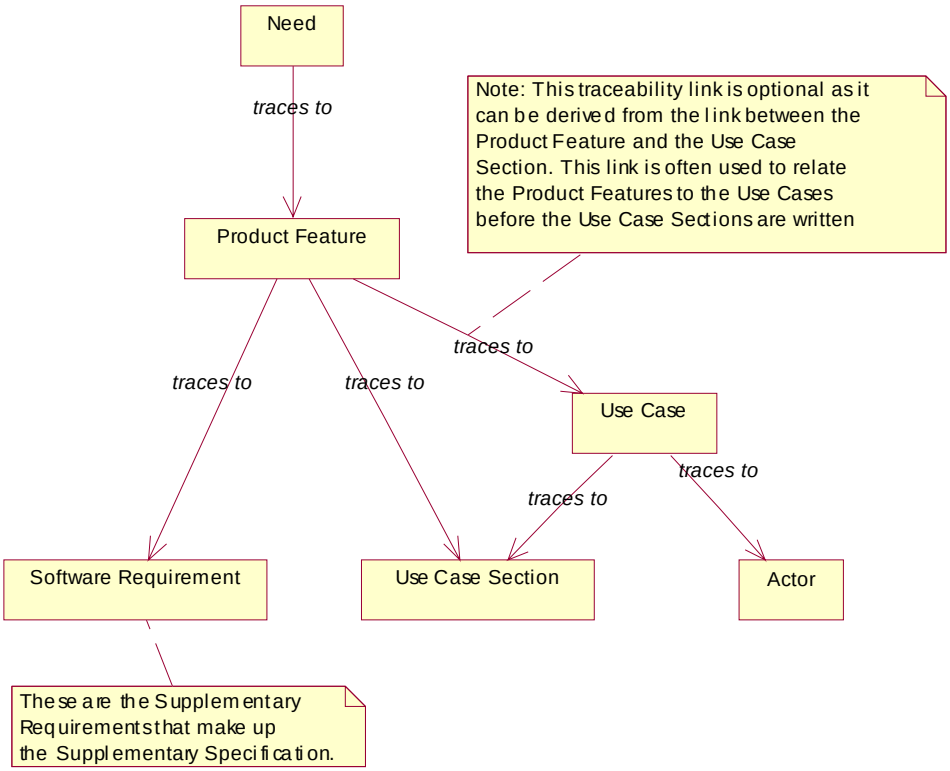
"유스 케이스 모델과 보충 스펙으로 SRS 를 구성합니다." 이는 RUP 에서 아웃라인되고 권장되는 전략입니다. 비전 문서에서 요구 및 제품 기능을 설명하며 유스 케이스로 추적됩니다. 유스 케이스 모델에 반영되지 않은 경우 보충 스펙으로 추적됩니다.

이러한 경우 주로 유스 케이스 모델로 소프트웨어 요구사항을 나타냅니다. 이 접근 방식은 유스 케이스만으로 쉽게 표현할 수 없는 소프트웨어 요구사항을 포함하는 보충 스펙으로 보완됩니다.

카테고리

특성	값	주석
명시적 추적성	중간	이러한 경우 유스 케이스 모델의 내재적 추적성 이외에 요구, 기능 및 유스 케이스 모델 간의 추적성을 명시적으로 유지보수해야 합니다.
신뢰도	중간	
책임성	높음	
형식성	중간	유스 케이스 모델에 요구 및 제품 기능을 추가함으로써 단지 유스 케이스 모델을 유지보수하는 것보다 더 정규적인 요구사항 관리 프로세스가 생성됩니다.
완전성	높음	소프트웨어 요구사항에 기능 및 유스 케이스 관점을 모두 적용함으로써 소프트웨어 요구사항의 캡처 및 우선순위 결정과 관련하여 상위 레벨의 완전성을 달성할 수 있습니다.
문서 세트	중간	요구 및 기능, 유스 케이스 모델과 보충 스펙을 포함하는 비전 문서가 작성됩니다.
초점	사용자, 이해 당사자(stakeholder) 및 프로젝트 관리자	유스 케이스 모델에 요구 및 기능을 추가함으로써 요구사항 활동의 초점을 확장할 수 있으며 이를 통해 제품 관리자와 기타 모든 이해 당사자 및 사용자의 요구를 보다 적극적으로 수용할 수 있습니다. 기능은 이해 당사자의 기대를 효과적으로 관리할 수 있는 도구로서 소프트웨어 요구사항의 유스 케이스 관점을 보완합니다.
이해 용이성	높음	보충 스펙을 적용한 유스 케이스 모델과 함께 요구 및 기능을 정의함으로써 모든 프로젝트 이해 당사자가 쉽게 이해할 수 있는 요구사항 모델을 제공할 수 있습니다.
프로세스	일반적으로 반복적이고 점진적입니다.	유스 케이스 모델만 사용하는 접근 방식과 같습니다.
개발 스타일	일반적으로 객체 지향적	유스 케이스 모델만 사용하는 접근 방식과 같습니다.

추적성 개요



추적성 유형

추적성 유형	설명
요구	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
제품 기능	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
유스 케이스	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.
소프트웨어 요구사항	전체 시스템에 적용되거나 유스 케이스에 쉽게 적용되지 않는 모든 소프트웨어 요구사항. 소프트웨어 요구사항이 빌드 대상 소프트웨어가 준수해야 하는 조건 또는 기능인 경우.
유스 케이스 섹션	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.
액터	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.

추적성 요약

추적성 링크	설명
요구 대 제품 기능	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
제품 기능 대 유스 케이스	선택적 추적성 링크. 제품 기능은 유스 케이스로 직접 추적될 수 있습니다. 이러한 경우 유스 케이스 섹션이 작성되기 전에 유스 케이스에 제품 기능을 지정할 수 있습니다. 또한 제품 기능 레벨에서 유스 케이스 모델에 대한 영향 분석을 수행할 수 있으며 반대의 경우도 가능합니다.
제품 기능 대 유스 케이스 섹션*	제품 기능이 유스 케이스 섹션으로 추적됩니다. 이러한 경우 유스 케이스 모델의 범위를 기능 기반으로 관리할 수 있으며 유스 케이스보다 적합한 레벨로 기능 세트와 유스 케이스 모델 간의 영향 분석을 쉽게 수행할 수 있습니다. 유스 케이스로 추적된 모든 제품 기능은 또한 특정 유스 케이스 섹션으로도 추적되어야 합니다.
제품 기능 대 소프트웨어 요구사항*	제품 기능은 보충 스펙의 소프트웨어 요구사항으로도 추적됩니다. 유스 케이스 모델로 추적되지 않는 모든 제품 기능은 보충 스펙에서 하나 이상의 소프트웨어 요구사항으로 추적되어야 합니다.
유스 케이스 대 액터	유스 케이스 모델만 사용하는 경우에 대한 정의와 같습니다.
유스 케이스 대 유스 케이스 섹션	유스 케이스 모델만 사용하는 경우에 대한 정의와 같습니다.

*각 제품 기능은 하나 이상의 유스 케이스 섹션 또는 보충 소프트웨어 요구사항으로 추적되어야 합니다. 그렇지 않은 경우 소프트웨어에 포함되지 않습니다.

이점 및 단점

이 접근 방식은 유스 케이스 및 기존 요구사항 관리 접근 방식이 제공하는 이점을 극대화하면서 단점은 최소화합니다.

장점:

- 쉽게 이해할 수 있습니다.
- RUP 에서 권장하는 접근 방식입니다.
- 세부적인 정규 하위 레벨 추적성을 사용할 수 있습니다.
- 소프트웨어 요구사항에 제품 기능 및 유스 케이스 관점을 모두 적용함으로써 요구사항 캡처를 쉽게 완료할 수 있습니다. 또한 이를 통해 요구사항 도출 및 캡처 활동에서 지체될 가능성을 최소화할 수 있습니다.
- 소프트웨어 요구사항을 쉽게 이해할 수 있는 양식으로 표현할 수 있습니다.
- 이 추적성 전략으로 요구사항 변경에 따른 영향 분석을 쉽게 수행할 수 있습니다. 즉, 기능 또는 유스 케이스 섹션을 구현하지 않는 데 따른 영향을 정확히 이해할 수 있습니다.

- 개별 요구사항은 유스 케이스 및/또는 제품 기능이 제공하는 컨텍스트를 갖습니다. 이를 통해 의미있는 요구사항 서브세트를 쉽게 식별할 수 있으며 결과적으로 범위 관리와 점진적인 제품 전달을 보다 쉽게 수행할 수 있습니다.
- 완벽한 최소 문서 세트
- 요구사항 관리에 필요한 노력을 최소화합니다.
- 이 솔루션은 확장성이 우수합니다. 정기 릴리스를 수행하는 경우 기능 및 유스 케이스 레벨 모두에서의 범위 관리 기능으로 모든 이해 당사자(stakeholder)가 적합한 세부사항 레벨에서 프로젝트의 진행상태를 추적할 수 있습니다.
- 이러한 경우 유스 케이스 모델은 모든 이해 당사자가 유스 케이스 모델의 적합성을 평가하는 데 도움이 되는 제품 기능을 통해 이해 당사자 요구와 관련됩니다.

단점:

- 모든 조직에 허용할 수 있는 것은 아닙니다.
- 주로 유스 케이스 모델로 표현되는 소프트웨어 요구사항을 기반으로 계약을 작성할 수 없다고 생각하는 사람들이 있습니다. 그러나 실제로는 많은 조직에서 이런 방식을 성공적으로 수행했습니다.

예제

이 접근 방식은 유스 케이스가 대부분의 소프트웨어 요구사항 표현에 적합한 형식으로 허용되는 모든 프로젝트에 적용할 수 있습니다.

유스 케이스 모델로 소프트웨어 요구사항 스펙 설명

설명

"유스 케이스 모델은 정규 SRS 를 나타냅니다." 이 접근 방식은 규정 또는 내부 프로토콜로 인해 정규 SRS 가 적용될 때 가장 많이 사용됩니다.

기존 SRS 는 고정 가격 개발을 수행 또는 아웃소싱하는 경우 일반적으로 계약서 동의를 위한 핵심 파트로 간주되며 다음과 같은 두 가지 일반 상황으로 전개됩니다.

개발 조직에는 기존 SRS 가 제공됩니다. 고객은 이 SRS 를 시스템 개발을 위한 시작점으로 활용합니다.

SRS 문서는 초기 프로젝트 라이프사이클에서 필수 또는 규정 인도물입니다. 모든 프로젝트에는 다른 모든 프로젝트와 동일한 방식으로 시스템 요구사항을 표현하는 기존의 정규 SRS 문서가 필요합니다.

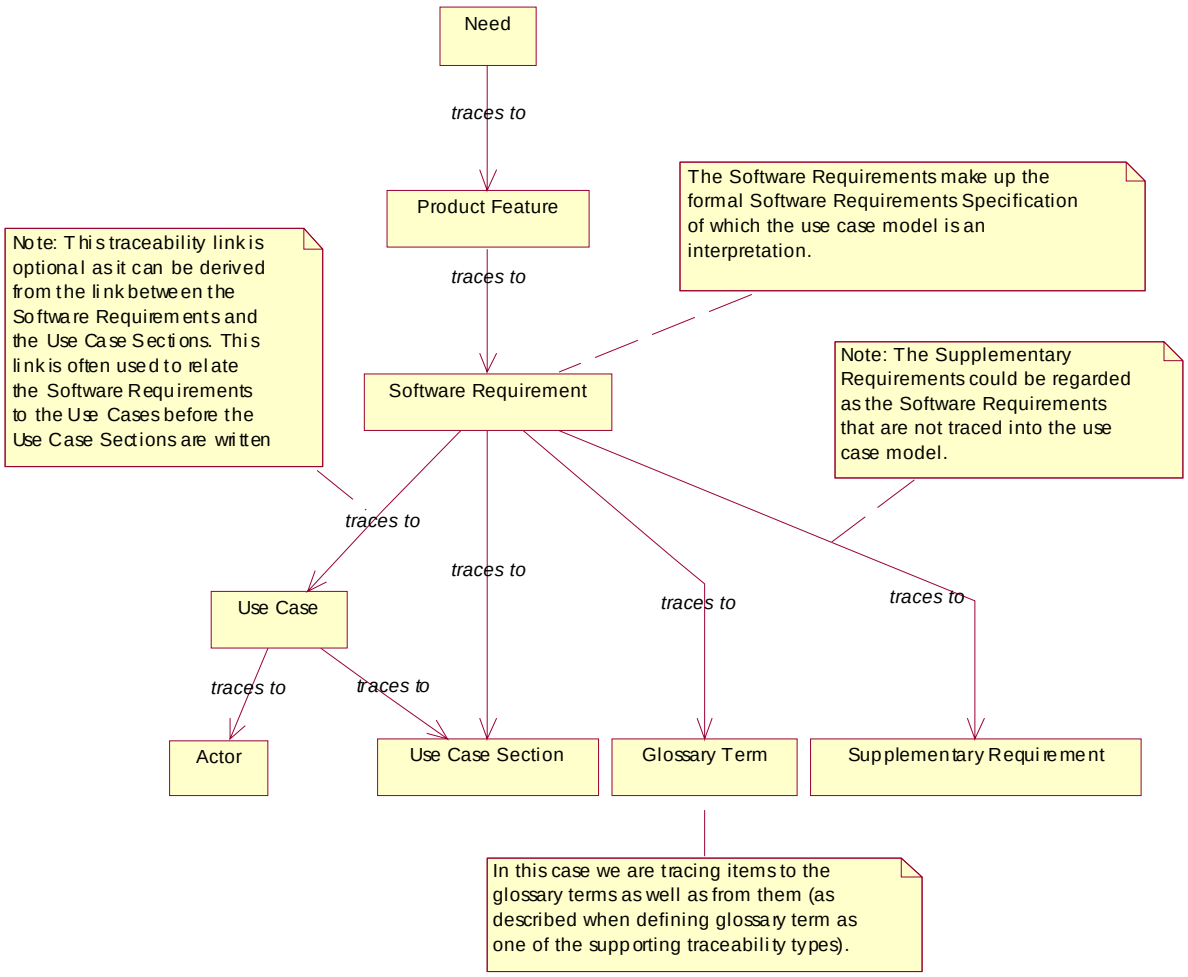
이러한 경우 유스 케이스 모델은 프로젝트 범위의 모든 소프트웨어 요구사항을 모델링하고 재해석하는 데 사용됩니다. 이 접근 방식을 채택하는 경우 일반적으로 SRS 가 먼저 표시됩니다. 특히 "유스 케이스 모델이 기능에 의해 구동" 접근 방식을 채택한 경우 보조 소프트웨어 요구사항 정의를 작성하지 않고, 다른 기법을 사용하여 유스 케이스 모델의 정보를 기존의 정규 SRS 형식으로 렌더링할 수 있습니다.

참고: 이 접근 방식을 채택하는 경우 "기존" 소프트웨어 요구사항 세트가 필요한 전체 기능을 나타내지 않아도 됩니다. 대신 유스 케이스 모델이 완벽한 기능 스펙을 제공하거나 완전성을 보장합니다. "기존" 소프트웨어 요구사항은 이해 당사자(stakeholder)가 직접 식별하거나 제기한 소프트웨어 요구사항을 캡처하는 데 사용할 수 있습니다.

특성

특성	값	주석
명시적 추적성	매우 높음	"유스 케이스 모델을 사용하지 않는" 접근 방식에서 필요한 모든 명시적 추적성은 정규 SRS 를 유지보수하는 데 필요하며 기존 소프트웨어 요구사항을 유스 케이스 모델로 추적하는 데 따른 추가 오버헤드가 발생하는 경우에도 필요합니다.
신뢰도	매우 낮음	신뢰 레벨이 매우 낮음을 나타내는 "안전" 중심의 접근 방식입니다.
책임성	높음	
형식성	매우 높음	두 가지 요구사항 관리 접근 방식을 병렬로 적용하는 정규 접근 방식입니다.
완전성	매우 높음	유스 케이스 모델로 기존 소프트웨어 요구사항 스펙을 보완함으로써 보다 완전한 접근 방식으로 만들 수 있습니다. 참고: 이러한 경우 유스 케이스 모델로 시스템 기능성 스펙을 완전하게 만드는 것으로 가정합니다. 따라서 소프트웨어 요구사항 스펙 세트는 동일한 레벨의 완전성이 필요하지 않습니다.
문서 세트	대규모	이러한 경우에는 기본적으로 시스템을 두 번 지정합니다.
초점	계약	이 접근 방식은 기존 SRS 로 표현되는 기존 계약을 이행하거나 기존 개발 방법에서 SRS 를 개발자와 고객 간의 계약으로 필요로 하는 경우 채택합니다.
이해 용이성	중간	두 가지 소프트웨어 요구사항 정의를 작성하는 경우 처음에는 혼동이 있을 수 있지만 유스 케이스 모델을 마스터 소프트웨어 요구사항 정의로 사용함으로써 시스템 요구사항을 쉽게 이해할 수 있습니다.
프로세스	개방형	반복적이고 점진적인 기법을 사용하기 위해 기존 SRS 에 유스 케이스 모델을 추가해야 하는 경우도 있지만 대부분의 모든 개발 프로세스를 지원할 수 있는 사항이 충분합니다.
개발 스타일	개방형	객체 지향 기법을 사용하기 위해 기존 SRS 에 유스 케이스 모델을 추가해야 하는 경우도 있지만 대부분의 모든 개발 스타일을 지원할 수 있는 사항이 충분합니다.

추적성 개요



추적성 유형

추적성 유형	설명
요구	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
제품 기능	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
소프트웨어 요구사항	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
유스 케이스	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.
액터	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.
유스 케이스 섹션	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.

용어집 용어	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.
보충 요구사항	전체 시스템에 적용되거나 유스 케이스에 쉽게 적용되지 않는 모든 소프트웨어 요구사항. 이러한 요구사항은 원래 소프트웨어 요구사항 세트에서 다시 설명하지 않아도 되지만 유스 케이스 모델로 추적되지 않는 범위 내 소프트웨어 요구사항에 포함될 수 있습니다.

추적성 요약

추적성 링크	설명
요구 대 제품 기능	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
제품 기능 대 소프트웨어 요구사항	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
소프트웨어 요구사항 대 유스 케이스	기능적 소프트웨어 요구사항은 유스 케이스로 추적됩니다. 비기능적 소프트웨어 요구사항의 서브세트 또한 유스 케이스로 추적됩니다. 이 관계에서는 유스 케이스의 요구사항 및 비즈니스 이점 관점에서 유스 케이스 모델을 상위 레벨에서 평가하고 범위를 지정할 수 있습니다. 참고: 범위 내 기능적 요구사항의 모든 요소는 **반드시** 유스 케이스 또는 용어집으로 추적되어야 합니다. 이러한 방식으로 반영되지 않은 요구사항은 구현되지 않습니다.
소프트웨어 요구사항 대 유스 케이스 섹션	유스 케이스로 추적되는 소프트웨어 요구사항은 또한 유스 케이스의 특정 유스 케이스 섹션으로 추적되어야 합니다. 이 관계에서는 유스 케이스에 대한 요구사항과 관련하여 유스 케이스의 유스 케이스 섹션에 대한 유효성을 검증할 수 있습니다. 유스 케이스로 추적된 모든 소프트웨어 요구사항은 특정 유스 케이스 섹션에서 이행되어야 합니다. 이중 추적성을 통해 이를 확인할 수 있으며 필요한 유스 케이스 섹션을 고려하기 전에 유스 케이스에 소프트웨어 요구사항을 할당할 수 있습니다. 일치하는 소프트웨어 요구사항이 없는 섹션도 있습니다. 참고: 유스 케이스로 추적되는 모든 기능적 요구사항은 **반드시** 특정 유스 케이스 섹션으로도 추적되어야 합니다. 이러한 방식으로 반영되지 않은 요구사항은 구현되지 않습니다.
소프트웨어 요구사항 대 용어집 용어	기능적, 비기능적 요구사항은 용어집 항목으로 추적될 수 있습니다. 이는 특히 시스템 관련 엔티티의 속성 및 관계를 식별하는 "정적 요구사항"에 적용됩니다. 용어집 용어가 소프트웨어 요구사항에서 추적되는 경우 해당 용어집 용어는 특정 유스 케이스에서 사용되어야 하며 그렇지 않은 경우 디자인 모델로 전달될 수 없습니다.
유스 케이스 대 액터	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.
유스 케이스 대 유스 케이스	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.

섹션	
소프트웨어 요구사항 대 보충 요구사항	보충 요구사항은 전체 시스템에 적용되거나 유스 케이스 모델에 쉽게 적용되지 않는 소프트웨어 요구사항을 다시 설명할 수 있습니다. 대체 접근 방식은 유스 케이스 모델로 추적되지 않는 범위 내 소프트웨어 요구사항을 모두 보충 요구사항으로 간주하는 것입니다. 이러한 경우 역시 해당 요구사항을 다시 설명하지 않아도 됩니다.

이점 및 단점

이 접근 방식은 지나치게 이상적인 접근 방식입니다. 이 접근 방식은 기존 SRS 로 표시되며 제공된 요구사항을 이해하고 유스 케이스 기반 접근 방식을 용이하게 하기 위해 유스 케이스 모델링을 사용하려는 프로젝트에만 적합합니다.

장점:

- 세부적인 정규 하위 레벨 추적성을 사용할 수 있습니다.
- 소프트웨어 요구사항을 쉽게 이해할 수 있는 양식으로 표현할 수 있습니다.
- 이 추적성 전략으로 요구사항 변경에 따른 영향 분석을 쉽게 수행할 수 있습니다. 즉, 기능, 소프트웨어 요구사항 또는 유스 케이스 섹션을 구현하지 않는 데 따른 영향을 정확히 이해할 수 있습니다.
- 개별 요구사항은 유스 케이스 및/또는 제품 기능이 제공하는 컨텍스트를 갖습니다. 유스 케이스 모델이 존재하므로 의미있는 요구사항 서브세트를 쉽게 식별할 수 있습니다. 결과적으로 범위 관리와 점진적인 제품 전달을 보다 쉽게 수행할 수 있습니다.
- 이러한 경우 유스 케이스 모델은 결과적으로 모든 이해 당사자가 유스 케이스 모델의 적합성을 평가하는 데 도움이 되는 제품 기능 및 소프트웨어 요구사항을 통해 이해 당사자 요구와 관련됩니다.
- 제한사항은 있지만 대부분의 조직에서 허용되며 모든 사용자에게 완벽한 접근 방식입니다. 이 접근 방식은 일반적으로 초기 유스 케이스 프로젝트에서 병렬 요구사항 프로세스의 양식으로 사용되거나(즉, 프로젝트를 이전 방식과 새로운 방식으로 모두 실행) 개발자가 유스 케이스를 사용한다는 사실을 숨기기 위해 채택될 수 있습니다.
- 유스 케이스를 처음 채택하거나 실험할 때 조직 방해를 최소화할 수 있습니다. 외부에는 계속 기존 SRS 가 표시되어 표준 프로시저 및 계약을 사용할 수 있습니다.

단점:

- 쉽게 이해할 수 없습니다. 즉, 기존 요구사항 설명과 유스 케이스 모델이 공존하여 혼동될 수 있습니다.
- 기존 소프트웨어 요구사항 스펙과 유스 케이스 모델이 공존하는 경우 요구사항 활동에 있어서도 두 배로 지연될 수 있습니다. 완전한 소프트웨어 요구사항 스펙을 쉽게 선택할 수 없기 때문입니다.
- 대규모 문서 세트를 유지보수해야 합니다.
- 중복 내용이 많아 요구사항 관리 프로세스가 복잡해집니다. 요구사항 변경에 따라 유스 케이스가 직접 갱신되면 기존 소프트웨어 요구사항을 사용하지 않을 수 있습니다.
- 많은 비용과 유지보수 노력이 필요한 접근 방식입니다.

예제

해당 계약의 일부로 기존 소프트웨어 요구사항 스펙이 제공되는 유스 케이스 기반 개발 기법을 사용하는 개발 회사에 유용한 접근 방식입니다. 이러한 회사는 유스 케이스를 사용함으로써 요구사항에 대한 이해를 나타내고 반복적이고 점진적인 방식으로 소프트웨어를 제공할 수 있습니다.

또한 기존 요구사항 캡처 기법을 사용하고 유스 케이스 기반 접근 방식으로의 변경을 거부하는 회사에 유스 케이스 기법을 도입할 때 유용한 전략입니다. 이러한 경우 유스 케이스가 개발 조직에 해당 가치를 입증해야 하며 유스 케이스에 대한 확신이 증가하면서 기존 소프트웨어 요구사항 스펙은 단계적으로 제거되어야 합니다. 이는 "유스 케이스 모델이 기능에 의해 구동"되는 접근 방식으로 이동하는 첫 번째 단계가 될 수 있습니다.

유스 케이스 모델로 여러 기존 소프트웨어 요구사항 세트 조정

설명

"유스 케이스 모델은 여러 소스의 정규 SRS 를 나타내며 단일 공통 시스템에 대한 스펙을 제공합니다."

이 접근 방식은 독립적인 여러 이해 당사자(stakeholder) 세트에서 여러 가지 기존 SRS 를 제공한다는 점을 제외하고는 "유스 케이스 모델로 소프트웨어 요구사항 스펙 설명" 접근 방식의 변형입니다. 이러한 상황은 일반적으로 연결 관계가 없이 독립적인 다양한 고객의 요구사항을 충족시키는 단일 응용프로그램을 개발하는 소프트웨어 회사에서 자주 발생합니다. 이러한 경우 유스 케이스 모델은 시스템 요구사항에 대한 개발자 통합 보기이며 개별 SRS 는 해당 요구사항을 나타내는 개별 이해 당사자 보기입니다. 다른 이해 당사자의 요구사항은 통합 또는 반영되지 않습니다. 개발자는 요구사항 및 유스 케이스 모델의 여러 개별 세트 간 추적으로 다양한 이해 당사자의 요구를 올바르게 평가하고 있는지 평가할 수 있습니다.

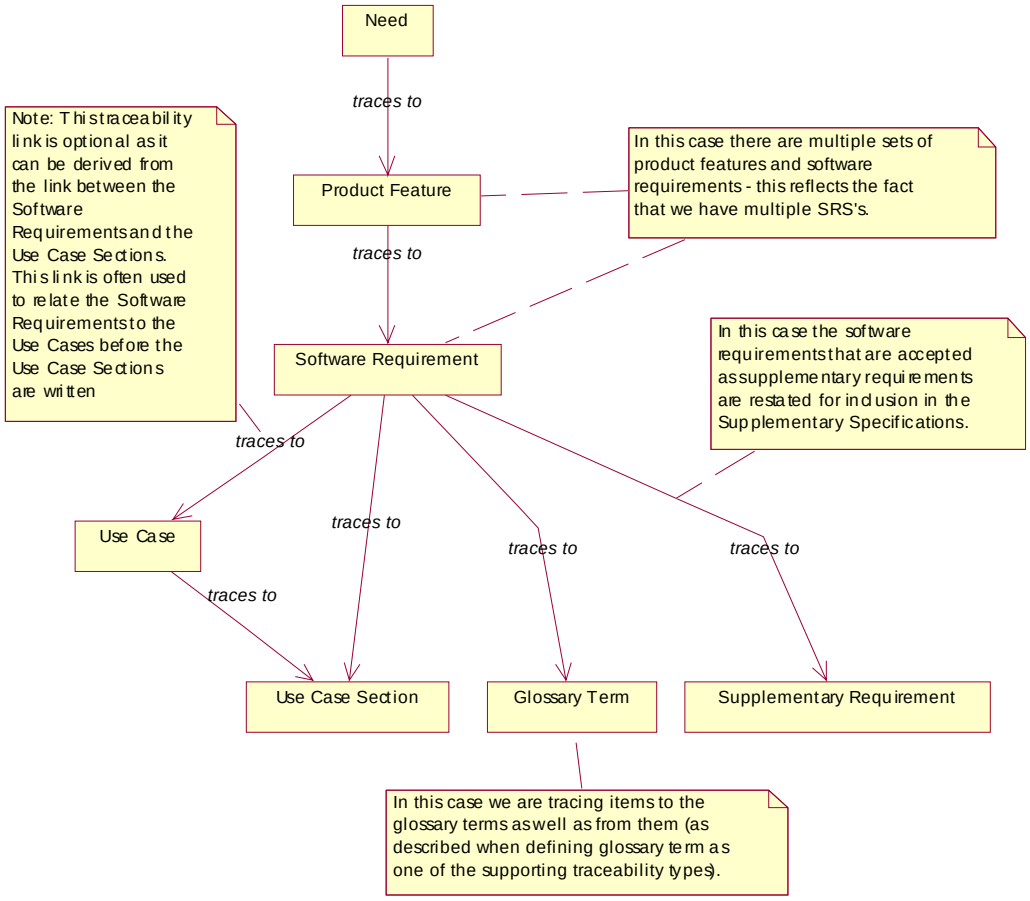
특성

이 전략은 이전의 "유스 케이스 모델로 소프트웨어 요구사항 스펙을 설명"하는 접근 방식의 변형입니다. 앞에서 이 접근 방식을 설명할 때는 몇 가지 차이점에 대해서만 언급했습니다.

특성	값	주석
명시적 추적성	매우 높음	"유스 케이스 모델로 소프트웨어 요구사항 스펙을 설명합니다."
신뢰도	매우 낮음	"유스 케이스 모델로 소프트웨어 요구사항 스펙을 설명합니다."
책임성	매우 높음	이러한 경우 개별 고객의 모든 관점을 해당 SRS 에 보존합니다.
형식성	매우 높음	"유스 케이스 모델로 소프트웨어 요구사항 스펙을 설명합니다."
완전성	매우 높음	"유스 케이스 모델로 소프트웨어 요구사항 스펙을 설명합니다."
문서 세트	대규모	이러한 경우 원하는 시스템의 복수 스펙이 존재하며 이 스펙은 단일 유스 케이스 모델로 조정됩니다.
초점	독립적인 여러 고객 관리	이 접근 방식의 초점은 정치적, 지리적 또는 조직적으로 조화될 수 없는 여러 가지 독립적인 또한 경우에 따라 모순되는 요구사항 소스를 관리하는 데 있습니다.
이해 용이성	중간	"유스 케이스 모델로 소프트웨어 요구사항 스펙을 설명합니다."
프로세스	일반적으로 반복적이고 점진적입니다.	이러한 경우 개발자는 유스 케이스 모델을 SRS 로 사용합니다. "유스 케이스 모델만 사용"하는 접근 방식을 참조하십시오.
개발 스타일	일반적으로 객체 지향적	이러한 경우 개발자는 유스 케이스 모델을 SRS 로 사용하여 소프트웨어 개발을 수행합니다. "유스 케이스 모델만 사용"하는

		접근 방식을 참조하십시오.

추적성 개요



요구사항 유형

요구사항 유형	설명
요구	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
제품 기능	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
소프트웨어 요구사항	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
유스 케이스	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.
유스 케이스 섹션	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.
용어집 용어	"유스 케이스 모델만 사용"하는 경우에 대한 정의와 같습니다.
보충 요구사항	전체 시스템에 적용되거나 유스 케이스에 쉽게 적용되지 않는 모든 소프트웨어 요구사항

추적성 요약

추적성 링크	설명
요구 대 제품 기능	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
제품 기능 대 소프트웨어 요구사항	"유스 케이스 모델을 사용하지 않는" 접근 방식에 대한 정의와 같습니다.
소프트웨어 요구사항 대 유스 케이스	유스 케이스 모델로 SRS 를 나타내는 경우와 같습니다.
소프트웨어 요구사항 대 유스 케이스 섹션	유스 케이스 모델로 SRS 를 나타내는 경우와 같습니다.
소프트웨어 요구사항 대 용어집 용어	유스 케이스 모델로 SRS 를 나타내는 경우와 같습니다.
소프트웨어 요구사항 대 보충 스펙	이러한 경우 보충 스펙 문서에 소프트웨어 요구사항을 다시 설명해야 합니다. 이 보충 문서는 유스 케이스 모델을 통해, 여러 개별 이해 당사자(stakeholder) SRS 에서 공개된 정보를 가져오는 개발에 대한 일관적이고 완벽한 단일 SRS 를 생성할 수 있도록 지원합니다.

이점 및 단점

이 전략은 이전의 "유스 케이스 모델로 소프트웨어 요구사항 스펙을 설명"하는 접근 방식의 변형이며 장단점의 많은 부분도 동일합니다.

다른 전략과 다른 이 접근 방식만의 이점은 독립적인 이해 당사자(stakeholder)의 관점을 정규 개별 SRS 의 양식으로 처리하고 보존할 수 있는 기능입니다.

반면에 대규모 문서 세트가 생성되어 유지보수 및 추적이 어렵다는 단점이 있습니다.

예제

영국의 한 소프트웨어 회사에서 보험 회사가 신상품을 전자 방식으로 분배할 수 있는 보험 브로커 지원 시스템을 개발하고 있었습니다.

이 프로젝트의 이해 당사자 그룹은 약 22 개였으며 그 중 약 2/3 는 보험 중개 회사, 1/3 은 보험 회사였습니다. 이들 이해 당사자는 요구사항에 있어 현격한 차이를 나타냈으며 대부분 보험 회사와 중개 회사가 완전히 모순된 요구사항을 갖고 있었습니다.

이 경우, 각 이해 당사자 회사마다 자사의 특정 요구사항을 자세히 나타내고 개별 관점을 쉽게 유지보수할 수 있는 소프트웨어 요구사항 스펙을 작성하기로 결정했습니다. 유스 케이스 모델은 모든 이해 당사자에게 통합된 시스템 비전을 제공하는 데 사용되었습니다. 원래 SRS 에서 유스 케이스 모델로의 추적성을 통해 이해 당사자가 시스템에서 충족되는 요구사항을 정확히 파악하고 시스템이 해당 요구에 적합한지 여부를 검증할 수 있었습니다. 또한 소프트웨어 회사는 각 이해 당사자별로 모든 이해 당사자 요구사항의 80% 달성을 목표로 진행상태를 추적할 수 있었습니다.

Rational
the software development company

본사 안내:

Rational Software
18880 Homestead Road
Cupertino, CA 95014
전화번호: (408) 863-9900

Rational Software
20 Maguire Road
Lexington, MA 02421
전화번호: (781) 676-2400

수신자 부담 전화번호: (800) 728-1212
전자 우편: info@rational.com
웹: www.rational.com
전세계 지사 안내: www.rational.com/worldwide