RUP 의 열 가지 필수사항— 효과적인 개발 프로세스의 필수사항

Leslee Probasco

Rational Software 백서

TP177, 9/00

Rational®
the software development company

목차

요약

"RUP"(Rational Unified Process[1])와 같은 소프트웨어 개발 프로세스를 효과적으로 적용하려면 해당 프로세스의 핵심 목표, 각 목표의 중요성 및 개발 팀이 이해 당사자(stakeholder)의 "실제" 요구를 충족시키는 고품질 제품을 생성할 수 있도록 프로세스를 상호 운용하는 방법을 먼저 이해해야 합니다.

도움 요청

몇일 전 저녁 때 이웃에 사는 Randy 가 도움을 요청하기 위해 우리집에 들렸습니다. 그는 교회 모임 사람들과 주말에 캠핑 겸 하이킹을 가기 위해 준비를 하면서 갖고 갈 기어 [1] 를 결정해야 하는 상황이었습니다. 그는 이미 옷과 장비를 챙겼으며 좀 더 구입하려고 하고 있었습니다. 즉, 그는 내 기어 목록을 빌리러 온 것이었습니다.

그가 내 기어 목록을 기억하고 있는 이유는 이전에 내가 등산 준비를 할 때 우리집을 방문한 적이 있었기 때문입니다. 그는 내가 야영이나 여행을 좋아하고 경험이 많다는 사실을 알고 있으며 여러 가지 장비와 옷 등의 목록을 만들어 놓고 공간이 한정된 배낭에 가져갈 물건을 빠르고 효율적으로 결정하는 것을 인상깊게 보았던 것입니다. 그는 지금 그 목록을 빌리러 온 것입니다.

나는 기꺼이 목록을 빌려주었지만 그다지 도움이 될 것 같지는 않습니다. 이유는 그 목록에는 그에게 실제로 필요한 모든 것이 있을 것 같지 않기 때문입니다.

즉, 그에게 필요한 모든 항목이 목록에 *표시*될 수는 있겠지만 엄밀히 말해 그의 이번 여행에 필요한 정확한 항목은 해당 목록에 전혀 없을 수도 있습니다. 예를 들어, 그의 신발 크기가 내 신발 크기보다 *훨씬* 클뿐만 아니라 내가 즐겨 갖고 가는 음식(및 분량)과 다른 음식을 좋아할 것입니다.

세부 고려사항

실제로 내 야외 활동 기어 목록에는 배낭 여행 및 등산에서 스키, 스노우슈잉, 빙벽타기 및 카약에 이르기까지 다양한 여러 야외 활동에 필요한 항목이 정말로 *수백 가지* 기재되어 있으며 여행 기간 또한 당일 여행에서 장기 탐험 여행까지 매우 다양합니다. 또한 조직을 위한 여행을 인솔할 때 필요한 항목(예: 여행 동의서, 면제 양식)과 친구들과 가까운 야외로 나갈 때 필요한 항목이 다릅니다. 무엇보다, 목록의 수많은 항목 중에서 상대적으로 간단한 여행에 *실제*로 필요한 항목을 구분하기가 어려울 것입니다.

Randy 는 준비를 하다가 화가 나서 두 손을 들고 말았습니다. 그는 아마도 제시간에 하이킹 준비를 마치기 어려울 것 같습니다. 우선 그는 무엇을 가져가야 할지도 알지 못합니다. 물론 그는 지난 주부터 인터넷 검색을 하면서 멋있는 부츠, 쟈켓, 필요한 옷 등을 새로 구입하여 이미 배낭이 가득 찬 상태입니다.

그러나 배낭에는 아직 음식이나 물도 넣지 않은 상태입니다. 이 밖에도 위험한 상황에도 대비해야 하며 필요한 장비 등을 잃어버리거나 부상자가 발생하는 경우에도 대비해야 합니다.

필수 항목 파악

Randy 가 배낭에 넣은 항목을 나열해보니 그가 야외에 나갈 때 반드시 필요한 항목에 대해 잘 알지 못함을 알 수 있었습니다.

나는 "열 가지 필수 항목"을 갖고 있는지 물었습니다. 그는 열 가지 필수 항목의 개념을 전혀 알지 못했습니다.

---

[1] "기어" 여기서 기어란 등산이나 야외 취미 활동을 즐기는 사람들이 전문 장비, 도구, 옷, 신발 등 여러 가지 다양한 스포츠에 사용되는 물건(아티팩트)을 통틀어 표현할 때 사용하는 용어입니다. RUP 의 아티팩트가 다양하다고 생각되는 경우 www.mgear.com 을 참조하십시오.

나는 Randy 에게 필요한 목록을 작성하기 위해 백지를 한 장 꺼내 맨 위에 "열 가지 필수 항목"이라고 적은 후 열 가지 항목을 적었습니다.

1. 지도
2. 나침반
3. 선글라스, 선크림
4. 여벌의 옷
5. 비상 식량, 물
6. 헤드램프
7. 구급 상자
8. 부싯돌
9. 성냥
10. 칼[2]

"기본 목록 작성"

열 가지 필수 항목이 작성되었습니다. 이 열 가지 항목을 정하고 준비를 마친 후 나머지 필요한 항목을 추가하면 됩니다. 물론 이 "필수 항목" 각각은 여행의 특성에 따라 늘리거나 줄일 수 있지만 "기본 목록"에서 시작하여 필요에 따라 항목을 추가하는 방법이 복잡한 목록에서 *필요하지 않은* 항목을 제외시키는 것보다 쉽습니다.

경험이 쌓일수록 자신만의 "세부 목록"을 갖게 되며 시간이 지나면서 이 목록은 자신에게 맞는 목록이 됩니다. 이 때 다른 사람의 목록을 참조하여 자신의 목록을 확장하는 것도 효과적인 방법입니다. 그러나 속이지 않는 한 두 사람의 목록이 결코 정확하게 같을 수 없습니다.

이 목록은 몇년 전 내가 처음 등산을 시작할 때 사용한 목록이며 아직도 여행의 종류와 기간에 관계 없이 이 목록을 참조합니다. 경험이 풍부한 등산가에게 "자신만의 열 가지 필수 항목"이 있는지 물어보면 대부분 질문의 의도를 정확히 파악할 것입니다. 실제로 "열 가지 필수 항목"은 그룹이나 개인에 따라 다를 수 있지만 본질적으로는 동일한 내용입니다. 또한 각각의 여행에 따라 실제로 가져가는 항목이 완전히 다를 수도 있지만 항상 시작은 "열 가지 필수 항목"에서 시작합니다.

상황별 필수 항목

이처럼 여행에 필요한 필수 항목 목록을 RUP 에 적용할 수 있습니다. 즉, 프로젝트 팀과 수많은 RUP 요소를 분류할 때 가장 많이 듣게 되는 질문은 "이 모든 항목을 모두 어떻게 분류하고 또한 프로젝트에 필요한 항목을 어떻게 판별하는지", "이 항목은 필수 항목인지" 또는 "RUP 는 대규모 프로젝트에만 사용되는 것이 아닌지" 등에 대한 내용입니다.

이러한 질문에 대한 응답은 대부분의 경우 "상황에 따라 다르다는 것입니다. "

실제로 필요한 것은 내 친구 Randy 를 도와준 것과 같이 프로세스 안내가 필요한 사람들에게 유용한 "RUP 의 열 가지 필수사항" 목록입니다. 이 목록은 프로젝트에 포함할 올바른 요소를 판별하기 위한 바람직한 시작점이 될 수 있으며 하루 정도 하이킹을 갈 때(소규모 프로젝트), 야간 산행, 스키 또는 카약을 타러 갈 때(다양한 분야의 중간 규모 프로젝트)나 몇 일 동안 장기 산행을 갈 때(중요한 대규모 프로젝트) 모두 적용할 수 있습니다. 핵심은

[2] 이 "열 가지 필수 항목" 목록에 대한 자세한 설명은 "Mountaineering – The Freedom of the Hills", 6th Edition(The Mountaineers, Seattle, WA, 1997), 35 – 41 페이지를 참조하십시오.

RUP 또는 실제로 효과적인 소프트웨어 프로세스의 "필수사항"에 초점을 맞추는 것입니다. 이는 또한 소프트웨어 개발에 있어 기존의 "우수 사례"[3] 개념과도 연결됩니다.

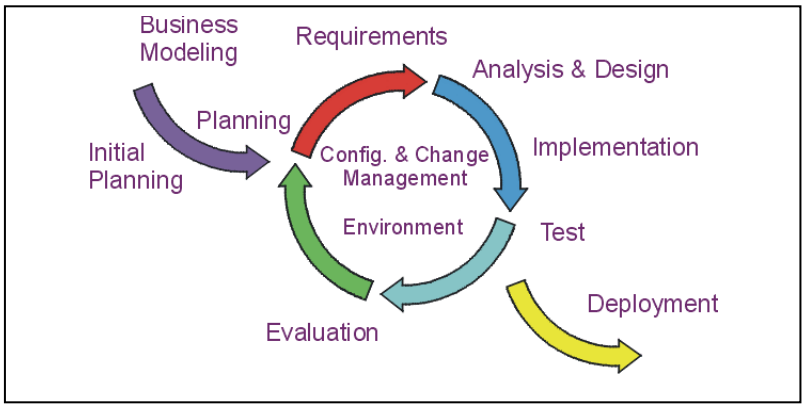
프레임워크 빌드

많은 프로젝트에서 공통된 문제점은 종종 특정 영역에 너무 집중한다는 것입니다. 즉, 고품질 제품을 생성하는 전체 프로세스 라이프사이클과 관련된 "핵심" 요소에 대한 아이디어를 파악하기 전에 해당 특정 영역의 세부사항에만 너무 치우쳐 있는 것입니다.

이러한 경우 프로젝트가 늦어지면 "요구사항 관리로 인해 작업 속도가 늦어진 것"으로 이해하고 불평합니다. 앞 문장에서 "요구 사항 관리"는 "유스 케이스", "프로젝트 메트릭 수집", "형상 관리 사용", "결함 추적 도구 사용", "상태 관련 회의" 또는 사용자에게 친숙한 여러 문구 중 어느 것으로라도 바꿀 수 있습니다.

보다 조직적이고 전체적인 접근 방식을 통해, 특정 문제점 영역에 초점을 맞추는 것으로 결정하기 전에 프로세스의 핵심 요소 구성이 올바른지(예: 아키텍처) 확인하는 것이 보다 효과적인 방법입니다.

올바른 소프트웨어 프로세스에 이 프레임워크(또는 아키텍처)가 적용되면 프로젝트에서 해당 문제점의 주요 원인으로 식별된 특정 영역에만 효과적으로 집중할 수 있게 됩니다. 일반적으로는 요구사항 관리가 가장 주된 원인입니다[4]. 그러나 RUP 접근 방식은 프로젝트의 위험성을 식별하고 우선순위를 결정하며 식별된 해당 위험성의 초기 완화 전략을 결정할 때 이 선택사항을 기반으로 합니다.



RUP 의 열 가지 필수사항

이제 "RUP 의 열 가지 필수사항"에 대해 알아보겠습니다. 다음 목록은 프로젝트에 포함될 최소 세트의 항목을 나타냅니다. 이 항목은 RUP 의 "필수사항"을 따라야 합니다.

1. 비전
2. 계획

---

[3] Rational 에서 지난 몇 년간 권장해온 "우수 사례"에는 (1) 반복 개발, (2) 요구사항 관리, (3) 컴포넌트 아키텍처 사용, (4) 시각적 모델링, (5) 품질 검증 및 (6) 변경사항 제어가 포함됩니다.

[4]요구사항 관리는 CMM(Capability Maturity Model)의 초기 "핵심 프로세스 영역" 중 하나이므로, 이는 소프트웨어 개발 프로세스를 개선하려는 사용자에게 전혀 낯선 시나리오가 아닙니다.

3. 위험성
4. 문제
5. 비즈니스 사례
6. 아키텍처
7. 제품
8. 평가
9. 변경 요청
10. 사용자 지원

이 항목을 각각 개별적으로 검토하여 RUP 에서의 적합한 위치를 파악하고 필수 항목의 "기본 목록"에 이러한 항목이 포함된 이유를 이해합니다.

비전 — 비전 개발

명확한 비전 수립은 이해 당사자(stakeholder)의 *실제* 요구"를 충족시키는 제품을 개발하는 데 있어 중요한 요소입니다. [5]

RUP 에서 비전은 *요구사항 원칙*의 "필수사항"(문제점 분석, 이해 당사자의 요구 이해, 시스템 정의 및 변경 요구사항 관리)을 캡처합니다.

비전은 세부 기술 요구사항에 대한 상위 레벨 기반 또는 경우에 따라 계약 기반을 제공합니다. 비전은 상위 레벨의 요구사항 및 디자인 제한조건을 캡처하여 사용자가 개발 예정 시스템을 이해할 수 있도록 도와 줍니다. 이것은 프로젝트 승인 프로세스에 입력을 제공하므로 비즈니스 사례와 밀접하게 관련되어 있습니다. 또한 이는 프로젝트에 관련된 기본적인 "이유 및 내용"을 커뮤니케이션하고 검증해야 하는 향후 모든 결정에 대한 척도가 됩니다.

비전 내용에는 다음 질문에 대한 응답이 포함되어야 하며 이러한 질문은 세부 아티팩트를 구분하기 위해 필요에 따라 제시될 수 있습니다.

- 주요 용어는 무엇입니까? (용어집)
- 해결하려는 문제점은 무엇입니까? (문제점 설명)
- 이해 당사자(stakeholder)는 누구입니까? 사용자는 누구입니까? 요구사항은 무엇입니까?
- 제품 기능은 무엇입니까?
- 기능적 요구사항은 무엇입니까? (유스 케이스)
- 비기능적 요구사항은 무엇입니까?
- 디자인 제한조건은 무엇입니까?

계획 — 계획 관리

"제품의 품질은 제품 계획만큼만 달성됩니다. "[6]

---

[5] "목적은 정해진 시간과 예산 범위 내에서 이해 당사자(stakeholder)의 실제 요구를 충족시키는 고품질 제품을 개발하는 것입니다." — *Managing Software Requirements,* Dean Leffingwell & Don Widrig, Addison-Wesley Longman, January, 2000.

RUP 에서는 소프트웨어 개발 계획(SDP)에서 프로젝트를 관리하는 데 필요한 모든 정보를 수집합니다. SDP 에는 도입/인식(Inception) 단계에서 개발된 여러 개별 아티팩트가 포함되며 전체 프로젝트 기간에서 유지보수됩니다.

SDP 는 프로젝트 스케줄 및 자원 요구를 계획하고 스케줄에 따라 진행상태를 추적하는 데 사용됩니다. SDP 에서 다루는 영역에는 프로젝트 조직, 스케줄(프로젝트 계획, 반복 계획, 자원, 도구), 요구사항 관리 계획, 형상 관리 계획, 문제점 해결 계획, QA 계획, 테스트 계획, 테스트 케이스, 평가 계획 및 제품 적합성 계획이 포함됩니다.

간단한 프로젝트의 경우 각각 하나 또는 두 개의 문장으로 구성될 수 있습니다. 예를 들어, CM 계획의 경우 "매일 업무 마감 시 프로젝트 디렉토리 구조의 컨텐츠를 압축하여 날짜를 표기하고 레이블을 부착한 zip 디스크에 복사한 후 버전 번호를 표시하여 중앙 파일 캐비넷에 보관합니다"와 같이 간단하게 작성할 수 있습니다.

소프트웨어 개발 계획 자체의 형식은 관련 활동과 사고만큼 중요하지 않습니다. 따라서 계획의 형식이나 계획을 빌드하기 위해 사용하는 도구 역시 중요하지 않습니다. Dwight D. Eisenhower 의 말대로 "계획 자체보다 계획을 수립하는 과정이 중요합니다."

필수 항목 번호 2, 3, 4, 5 및 8 은 RUP 에서 *프로젝트 관리 원칙의 "필수사항"(새 프로젝트 착상, 범위 및 위험성 평가, 프로젝트 모니터링 및 제어, 각 반복 및 단계에 대한 계획과 평가)을 캡처합니다.*

위험성 — 위험성 식별 및 완화

RUP 의 핵심 가치는 위험성이 가장 높은 항목을 프로젝트 초기 단계에서 식별하여 처리하는 것입니다. 위험성 목록은 프로젝트 성공과 관련하여 인지된 위험성을 캡처하기 위해 작성됩니다. 위험성 목록은 상당히 부정적인 결과물을 낼 수 있는 이벤트를 우선순위의 내림차순으로 식별합니다.

각 위험성마다 해당 위험성을 완화시키기 위한 계획이 필요합니다. 이 계획은 프로젝트 활동 계획을 위한 중심점이자 반복 구성의 기반이 됩니다.

문제 — 문제 지정 및 추적

모든 프로젝트에 있어, 지속적인 활동에서 직접 파생되는 객관적 데이터와 계속 개방적인 커뮤니케이션을 수행함으로써 제품 구성을 전개하는 것이 중요합니다. RUP 에서는 이를 위해 관리 문제, 기술적 문제 및 프로젝트 위험성을 처리, 커뮤니케이션 및 해결하는 메커니즘을 제공하는 정기적인 상태 평가를 수행합니다. 상태 평가에서는 문제를 식별하는 것 이외에 해결책을 담당하는 책임자와 납기일이 지정되어야 하며 필요에 따라 정기적으로 추적하고 갱신해야 합니다.

이런 프로젝트 스냅샷은 관리 주의에 대한 핵심을 제공합니다. 기간이 변경되는 동안, 프로젝트 히스토리를 캡처하고 진행을 방해하는 장애 또는 병목 현상을 제거하기 위해 강제 실행 기능이 필요합니다.

비즈니스 사례 — 비즈니스 사례 점검

비즈니스 사례는 비즈니스 관점에서 이 프로젝트의 투자 가치 여부를 판별하는 데 필요한 정보를 제공합니다.

비즈니스 사례의 기본 목적은 프로젝트 비전을 실현하기 위해 경제적 계획을 개발하는 것입니다. 일단 개발한 비즈니스 사례는 프로젝트에 의해 제공된 투자 수익률(ROI)을 정확히 평가하는 데 사용됩니다. 이를 통해 프로젝트에 대한 정당한 이유를 제공하고 경제적 제한조건을 설정합니다. 이것은 프로젝트의 가치에 대해 경제적 의사 결정자에게 정보를 제공하고, 프로젝트가 진행되어야 하는지 여부를 판별하는 데 사용됩니다.

설명은 특정 문제점에 대해 깊이 파고드는 대신 제품이 필요한 이유에 대해 설득력 있는 주장을 제공해야 합니다. 그러나 모든 프로젝트 팀 구성원이 이해하고 기억하기 쉬울만큼 간결해야 합니다. 주요 이정표에서 예상되는

---

[6] Johnson Space Center Shuttle Software Group, "They Write the Right Stuff", Charles Fishman, *Fastcompany,* Issue 6, p. 95, December, 1996.

수익 및 비용의 측정이 아직 정확한지, 프로젝트가 계속되어야 하는지 여부를 알아보기 위해 비즈니스 사례가 다시 점검됩니다.

아키텍처 — 컴포넌트 아키텍처 디자인

RUP 에서, 특정 시점의 소프트웨어 시스템 아키텍처는 인터페이스를 통해 상호작용하는 중요한 시스템 컴포넌트의 조직 또는 구조이며 이 컴포넌트는 연속적으로 더 작은 컴포넌트 및 인터페이스로 구성됩니다. 여기서 기본 요소와 요소 간 상호작용을 이해해야 합니다.

RUP 는 소프트웨어 아키텍처를 디자인, 개발 및 유효성 검증하기 위한 일정 방식의 조직적인 방법을 제공합니다. 이는 RUP *분석 및 디자인 원칙의 "필수사항"(후보 아키텍처 정의, 아키텍처 정제, 동작 분석 및 시스템 컴포넌트 디자인)입니다.*

소프트웨어 아키텍처에 대해 논의하려면 우선 아키텍처 표시, 즉 아키텍처의 주요 측면을 설명하는 방법을 정의해야 합니다. RUP 에서, 이 설명은 소프트웨어 아키텍처 문서에서 캡처되며 다중 보기에 아키텍처를 표시합니다.

각 아키텍처 보기에는 개발 프로세스의 이해 당사자(stakeholder)인 일반 사용자, 디자이너, 관리자, 시스템 엔지니어, 유지보수자 등과 관련된 일부 특정 관심사항 세트가 표시됩니다. 이 보기는 구조적으로 중요한 프로젝트 관련 의사 결정 시 설계자와 다른 프로젝트 팀 구성원 간의 커뮤니케이션 매체로 사용됩니다.

제품 — 점진적인 제품 빌드 및 테스트

RUP 에서 *구현 및 테스트 원칙*의 "필수사항"은 시스템 컴포넌트를 점진적으로 코딩, 빌드 및 테스트하는 것입니다. 도입/인식(Inception) 후 각 반복의 종료 시점에는 실행 가능 릴리스가 생성됩니다.

정제(Elaboration) 단계 종료 시점에는 아키텍처 프로토타입을 평가할 수 있으며 필요한 경우 사용자 인터페이스 프로토타입이 포함됩니다. 구현/구축(Construction) 단계의 각 반복에서는 컴포넌트가 실행 가능하고 테스트 완료된 빌드에 통합됩니다. 또한 이 빌드는 최종 제품으로 발전됩니다.

이 필수 요소의 핵심은 제품 빌드를 수반하는 테스트 활동의 통합 세트와 지속적인 형상 관리 및 검토 활동입니다.

평가 — 정기적인 결과 평가

반복 평가는 반복의 결과, 평가 기준의 충족 정도, 학습 내용 및 구현될 프로세스 변경사항을 캡처합니다.

반복 평가는 반복 방법의 필수 아티팩트입니다. 프로젝트의 범위 및 위험성, 반복적 특징에 따라 단순한 예시 및 결과물 레코드에서 완전한 형식의 테스트 검토 레코드까지 될 수 있습니다.

여기서, 핵심은 제품 문제점뿐 아니라 프로세스 문제점에 초점을 맞춥니다. "빨리 뒤쳐질수록 따라잡는 데 더 오래 걸립니다."

변경 요청 — 변경 관리 및 제어

형상 및 변경 관리 원칙의 "필수사항"은 프로젝트 라이프사이클에서 변경사항이 발생할 때 모든 이해 당사자(stakeholder) 요구를 고려하고 해당 요구를 최대한 충족시키면서 프로젝트 범위를 관리하고 제어하는 것입니다.

첫 번째 프로토타입이 사용자 앞에 놓이자 마자(종종 그 이전) 변경이 요청*됩니다*. (라이프의 확실성 중의 하나). 변경을 제어하고 프로젝트의 범위 및 이해 당사자의 기대를 효율적으로 관리하려면 개발 아티팩트에 대한 모든 변경사항이 변경 요청을 통해 제안되고 일관적인 프로세스로 관리되어야 합니다.

변경 요청은 결함, 개선사항 요청 및 기타 유형의 제품 변경 요청을 문서화하고 추적하는 데 사용됩니다. 변경 요청의 장점은 결정 레코드를 제공하고, 평가 프로세스를 통해 모든 프로젝트 팀 구성원이 잠재적 변경의 영향을 파악했는지 확인한다는 것입니다. 변경 요청은 제안된 변경의 영향 평가와 프로젝트 범위 관리에 필수적입니다.

사용자 지원 — 사용자 지원 제공

사용자 지원의 최소 구성요소는 일반적으로 온라인 도움말을 통해 구현되는 사용자 안내서이며 이 밖에 설치 안내서 및 릴리스 정보도 포함될 수 있습니다. 그러나 제품의 복잡도에 따라, 제품 패키지와 함께 전달되는 명세서와 훈련 자료도 필요할 수 있습니다.

RUP 에서 *배치 원칙*의 "필수사항"은 제품과 함께, 제품의 학습, 사용, 작동 및 유지보수에 있어 일반 사용자를 지원하는 데 필요한 모든 자료를 정리하여 전달하는 것입니다.

요구사항 고려

이 문서에서 제공하는 필수사항 목록을 보고 해당 내용에 동의하지 않는 사람도 있을 수 있습니다. 예를 들어, 이 그림에서 "요구사항"을 적용할 위치에 대해 의문을 가질 수 있으며 일반적으로는 "비전"의 위치가 적합합니다. 또한 요구사항이 필수사항인지 여부에 대해서도 의문이 있을 수 있습니다. 요구사항은 가장 중요한 필수사항이며 목록에 "요구사항"을 표시하려면 이 시점에서 목록에 추가할 수 있습니다. 각 프로젝트 팀은 자체 목록을 작성해야 하며 이 문서에서 제공하는 열 가지 필수사항 목록은 세부 논의를 위한 시작점으로서의 의미가 있습니다.

그러나 경험상으로 볼 때 프로젝트 팀(특히 내부 프로젝트의 경우)을 대상으로 "해당 요구사항"에 대해 물으면 "특별한 요구사항이 없다"고 대답합니다. 처음에는 이런 반응에 놀라기도 했습니다(군사 우주 항공 분야에서). 어떠한 요구사항도 없다는 것은 있을 수 없기 때문입니다. 그러나 사람들은 "반드시" 준수하지 않으면 프로젝트가 거부되는, 외부에서 부과되어 "강요"된 사항을 의미하는 것으로 "요구사항"이라는 단어를 생각하며, 그들에게는 *실제로* 그러한 사항이 없다는 것을 알게 되었습니다. 이들은 프로젝트 기간 동안 제품 요구사항이 발전하는 연구 및 개발과 관련이 있습니다.

이제 이처럼 일반적인 반응을 고려하여 제품의 비전이 무엇인지 정의해야 합니다. 그 다음에는 아무 문제점 없이 위의 A – G 질문이 각각 진행되고 그 다음에 요구사항이 논의될 수 있습니다. 이는 요구사항이 강제로 부과되는 경우보다 "발견"되는 경우가 많은, 계약 프로젝트보다 협업 환경에서 일반적인 패턴입니다.

테스트 고려

또한 이 문서에는 RUP 의 "필수사항"으로 "테스트"가 포함되어 있지 않습니다. 이유는 효과적인 소프트웨어 개발 프로세스에 선택적인 비즈니스 모델링과 달리 테스트는 선택적 요소가 아니라고 생각하기 때문입니다. 테스트는 IEEE 1074 소프트웨어 프로세스 표준과 같이, 지속적인 형상 관리 및 검토 활동과 동일한 방식으로 제품 디자인 및 빌드를 수반하는 활동의 통합 세트로 간주됩니다.

또한 테스트 작업이 제품(필수사항 번호 7) 빌드와 검증 및 평가(필수사항 번호 8)에 중요한 테스트 결과와 함께 포함됩니다. 소프트웨어 개발을 위한 RUP 의 반복적 접근 방식의 본질은 문제점을 처리하고 위험성 및 문제를 최대한 빨리 해결하기 위해 제품의 실행 가능 버전을 지속적으로 빌드, 테스트 및 평가하는 것입니다.

요약: 열 가지 필수사항 적용

"RUP 의 열 가지 필수사항"을 실생활에 활용할 수 있는 방법을 연구해야 합니다.

소규모 프로젝트의 경우

먼저 내 친구 Randy 와 같은 누군가가 단지 프로세스 학습을 목적으로 RUP 를 사용하여 직접 또는 경험이 없는 소규모 팀과 함께 단순한 제품을 빌드하려는 경우, RUP 의 모든 기능과 Rational Suite 도구로 어려움을 겪도록 하기 보다는 내가 갖고 있는 "열 가지 필수사항" 목록을 제공하여 활용하도록 할 수 있습니다.

실제로 이 열 가지 필수사항은 특별한 도구 지원 없이 수행할 수 있습니다. 열 가지 필수사항 각각에 한 섹션이 할당된 프로젝트 노트북은 실제로 프로젝트 관리를 위한 올바른 시작점으로 활용할 수 있습니다. 또한 RUP 의 개인 응용프로그램에 대한 변경 요청을 처음 관리하고 우선순위를 결정하며 추적할 때는 포스트잇 메모를 효과적으로 활용할 수 있습니다.

성장 단계 프로젝트의 경우

이처럼 열 가지 필수사항을 적용하는 간단한 방법은 프로젝트의 규모 또는 복잡도가 증가하면서 그 유효성이 감소하므로, Rational 도구의 잠재 고객을 고려해야 하는 경우 "RUP 의 열 가지 필수사항"을 적용하는 데 있어 적절한 가능성을 발견할 수 있습니다.

또한 각 영역을 보다 심도 있게 다루려는 경우 완벽한 업무 수행을 위해 결국 Rational 도구지원과 함께 전체 RUP 가 필요하게 됩니다. 그러나 "우수 사례" 적용에 있어 도구보다는 "RUP 의 열 가지 필수사항"을 활용하도록 권장합니다.

성숙 단계 프로젝트 팀의 경우

성숙 단계 프로젝트 팀의 경우, "열 가지 필수사항"을 통해 해당 프로세스의 핵심 요소에 대한 밸런스를 신속하게 평가할 수 있는 방법과, 개선으로 최대 효과를 얻을 수 있는 영역을 식별할 수 있습니다.

모든 프로젝트에 있어 이러한 필수사항을 무시할 때 발생할 수 있는 결과를 고려해야 합니다. 즉, 필수사항을 고려하지 않을 때 실패하게 될 결과를 미리 파악해야 합니다. 예를 들어, 다음과 같습니다.

- 비전이 없습니까? 현재 추구하는 방향을 쉽게 잃을 수 있습니다.
- 계획이 없습니까? 진행을 추적할 수 없을 것입니다.
- 위험성 목록이 없습니까? 현재 잘못된 문제에만 집중하며 앞으로 5 개월 후에 예상치 못했던 문제가 발생할 수 있습니다.
- 문제 목록이 없습니까? 책임이 없는 경우 성공에 장애가 됩니다.
- 비즈니스 사례가 없습니까? 프로젝트로 인해 시간과 돈을 잃을 위험성이 있습니다. 즉, 프로젝트 자체가 취소되거나 파산할 수 있습니다.
- 아키텍처가 없습니까? 통신, 동기화 및 데이터 액세스 문제 발생 시 처리하지 못할 수 있습니다. 규모 및 성능 관련 문제점이 발생할 수 있습니다.
- 제품(프로토타입)이 없습니까? 코드를 작성합니다. 서류 작업이 아닌 고객에게 유용한 코드를 작성해야 합니다.
- 평가는 없습니까? 현실을 회피해서는 안됩니다. 사실을 인식해야 합니다. 최종 기한에 얼마나 근접해 있습니까? 품질 또는 예산이 목적에 얼마나 근접해 있습니까?
- 변경 요청은 없습니까? 변경 요청을 추적할 수 있는 방법은 무엇입니까?
- 사용자 지원은 없습니까? 사용자에게 질문이 있거나 제품 사용 방법을 알지 못하는 경우에는 어떤 결과가 발생합니까? 도움을 쉽게 얻을 수 있습니까?

결론

"열 가지" 필수사항을 강조하는 데 특별한 이유가 있는 것은 아닙니다. 단지 열 가지 *이상*을 지정하고 싶지 않았을 뿐입니다. "아홉 가지 필수사항" 또는 그 보다 더 적은 수를 정할 수도 있겠지만 열 가지가 가장 적당하다고 생각됩니다. 적어도 나에게는 그렇습니다. 프로젝트 그룹에서 "열 가지 필수사항"이 무엇인지에 대한 논의의 시작점으로 이를 사용하도록 권장합니다.?

이 문서 내용에 동의하는 사람이라면 "열 가지 필수사항" 목록을 쉽게 기억하기 위한 간단한 인용 문구 또는 두문자어를 지정할 수 있습니다. 예를 들어, 이 목록을 네 개 음절로 요약하려면 V-PRI-BAPE-CU 를 사용할 수 있습니다. 물론 여기서는 필수사항의 수를 10 으로 제한하므로 목록을 참조하지 않고 열 손가락으로도 기억할 수 있습니다.

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