RUP – 보다 높은 프로세스 성숙도를 위한 인에이블러

Annie Kuntzmann-Combelles, Q-Labs France

Philippe Kruchten, Rational Software Canada

Rational Software 및 Q-Labs 백서

TP 178, 2/01

Rational
the software development company

Q-Labs

목차

요약

이 문서는 성숙한 조직(레벨 3 개발 단위)에 필요한 핵심 개념과 Rational Unified Process(RUP) 컴포넌트가 해당 요구사항을 충족시키는 방법에 대해 강조 설명합니다.

높은 프로젝트 성숙도와 조직 성숙도에 대해 모두 설명합니다. 또한 섹션 4 는 RUP 시작 방법에 대한 유용한 정보를 제공하며 얼리어댑터(early adopter)가 다양한 측면에서 관찰한 주요 이점에 대해 보고합니다.

효율성의 가치

오늘날 기업들은 보다 높은 성과를 달성하기 위한 방법을 지속적으로 모색해야 하며 글로벌 환경에서의 경쟁이라는 향후 과제를 안고 있습니다. 전세계 시장을 무대로 경쟁하는 경우 다음과 같은 공통 환경에 대처해야 합니다.

- **성능 요구**: 유효성 검증 단계에서 버그가 너무 많이 발견되었으며 납기를 맞추기 위해 불안정한 제품 버전이 시장에 출시되었습니다.
- **효율성 요구**: 프로젝트 예산을 초과하고 계획이 지연되었습니다.
- **시장 점유율 하락**: 경쟁자가 보다 우수한 성과를 나타내고 있으며 보다 나은 품질을 제공합니다.
- **우수한 인재의 부족**: 이직율 상승 및/또는 신규 엔지니어 인력을 확보하기가 매우 어렵습니다.
- **최신 기술과 최소 위험성의 통합 요구**: 제품 지원 기술이 최신 동향을 따라야 하며 도전 과제를 해결할 수 있는 팀 경쟁력이 부족합니다.

즉, 개발된 제품의 모든 측면을 최적화해야 한다고 말할 수 있습니다.

따라서 제품 개발 프로세스가 결과에 큰 영향을 준다는 일반적인 가정하에 가장 일반적인 시나리오는 소프트웨어 프로세스에 역점을 둔 제품 개선입니다.

이러한 상황에 직면한 조직은 성공을 위해 다음과 같은 솔루션에 도달하게 됩니다.

솔루션의 첫 번째 파트는 시장에서의 소프트웨어 가치에 대한 이해입니다. 소프트웨어가 제품 및 프로젝트의 핵심 경쟁력을 유지하는 경우 해당 제품 및 프로젝트를 지정하고 구현하는 프로세스를 장기간에 걸쳐 주의깊게 정의, 문서화 및 최적화해야 합니다. 회사의 상급 관리자는 아직도 소프트웨어로 인해 문제점이 발생하고 제공 시점도 너무 느리다는 확신을 갖고 있으며 이는 소프트웨어에 대한 부가 가치 분석이 수행되지 않았기 때문입니다.

솔루션의 두 번째 파트는 다음과 같이 프로세스의 주요 문제점을 수정하기 위해 필요한 투자를 수행하는 것입니다.

- 프로세스 평가를 통한 개선 기회 식별
- 소모적인 투자를 피하고 타인의 경험을 통한 학습과 발전 속도 가속화
- 프로젝트에 대한 관리 및 기술적 측면 모두 고려
- 앞으로 필요한 스킬 관리(학습, 공유 및 성장)

조직을 생산성이 높은 팀으로 변환시키기 위한 로드맵에는 여러 가지가 있습니다. 그 중 가장 올바른 결정 중 하나는 변경할 수 있는 것은 변경하고, 변경할 수 없는 것은 허용하고, 성공적인 소프트웨어 커뮤니티에서 유효성을 검증한 가능한 모든 것을 활용하는 것입니다. CMM(Capability Maturity Model)[6]과 Rational Unified Process(RUP)[11]는 제품 개발을 위한 올바른 프로세스로의 진행 속도를 가속화할 수 있는 견고한 도구로서 현재 널리 사용되고 있는 대표적인 도구입니다. 도구라는 용어를 사용하는 이유는 이 두 컴포넌트가 소프트웨어

사례를 변경하고 개선할 수 있는 효과적인 지원 기능을 수행함으로써 조직 관점에서 가장 바람직한 우수 사례를 정의할 수 있기 때문입니다.

다음 섹션은 이 두 가지 컴포넌트(CMM 및 RUP)와, 프로세스 및 전달된 제품을 개선하기 위해 고려해야 하는 몇 가지 기타 핵심 컴포넌트에 대해 설명합니다.

개선 로드맵

지난 10 년 간 소프트웨어 커뮤니티에서는 프로세스의 중요성에 대한 인식이 확산되면서 소프트웨어 제품을 생성하는 프로세스와 독립적으로 소프트웨어 제품을 평가할 수 없다는 점을 인식하게 되었습니다. 결과적으로 여러 조직에서 자사 소프트웨어 개발 프로세스를 개선하기 위해 자체 SPI(Software Process Improvement) 프로그램을 작성했습니다. 대부분의 경우 소프트웨어 개발은 하나 이상의 조직 부서 또는 단위에 집중되며 독립 원칙으로 간주됩니다. 이러한 상황은 SPI 프로그램의 적용에 따라 변화하며 이를 통해 소프트웨어 팀은 조직의 다른 원칙과 보다 원활하게 통합됩니다.

조직에서 SPI 프로그램을 시작하기 위해 필요한 준비

- 프로그램 시작을 위한 자극(stimulus)과 개선 활동을 조정(즉, 비전 설명 및 목표로 개선 프로젝트 설정)하는 핵심 인물

- 프로젝트 및 조직의 사례를 평가하고 성공을 방해하는 누락 파트를 식별하기 위한 기준이 되는 참조 모델. SEI(Software Engineering Institute)에서 개발된 CMM 모델은 수천 개 조직에서 사용하는 실제 참조 모델이며 SPICE 프레임워크(ISO 15504)[9]는 머지 않아 CMM 이 준수하게 될 대체 공식 표준입니다.

- 개선 프로그램이 성공적인 결과로 완료될 수 있도록 관리하는 메소드: SEI 에 정의된 IDEAL [4] 프레임워크의 강점이 입증되었습니다.

- 각 프로젝트 성공을 위해 수정할 수 있는 개발 프로세스: RUP 는 유용한 상용 솔루션입니다.

CMM 모델 [6]

SEI(Software Engineering Institute)의 CMM(Capability Maturity Model)은 효과적인 소프트웨어 프로세스[6]의 요소에 대해 설명하는 프레임워크입니다. CMM 은 성숙되지 않은 임시 프로세스에서 성숙된 정규 프로세스로의 점진적인 개선 경로를 설명합니다. 또한 CMM 은 소프트웨어 개발 및 유지보수 기능을 강화하는 것으로 입증된 여러 핵심 프로세스 영역에 권장 사례 세트를 제시합니다. 개발자는 CMM 을 통해 개발 및 유지보수 프로세스를 제어하는 방법과 올바른 소프트웨어 엔지니어링 및 관리 활동을 수행할 수 있는 방법을 참조할 수 있습니다.

IDEAL 메소드 [4]

SPI(Software Process Improvement)는 소프트웨어 작업을 구성하고 수행하는 방법을 발전시킬 수 있는 시스템적인 장기 협업 메소드입니다. 개선 메소드에는 SEI 에서 정의한 SPI 에 대한 통합 접근 방식인 IDEAL 메소드가 포함됩니다. IDEAL 은 시작, 진단, 확립, 실행 및 활용 등 다섯 단계를 식별합니다. 이 단계는 다음과 같이 각각 특정 활동과 연관됩니다.

- 실현 또는 지원될 비즈니스 목적 및 목표 지정 **(시작)**
- 관련 표준 또는 참조 모델과 관련하여 조직의 현재 상태 식별**(진단)**
- 선택한 접근 방식을 구현하기 위한 계획 개발**(확립)**
- 조직 요구에 대한 "최선의 추측(best guess)" 솔루션(예: 기존 도구, 프로세스, 지식 및 스킬)을 작성하는 데 사용할 수 있는 모든 요소의 통합 및 솔루션 구현**(실행)**
- IDEAL 을 구현하는 데 사용되는 프로세스와 관련하여 얻은 교훈 요약 **(활용)**

프로세스

프로세스가 없는 조직도 있고 프로세스가 해당 경험을 기반으로 하는 조직도 있습니다. OPEN 및 Rational Unified Process(RUP)와 같은 사용 가능한 기존 프로세스도 있습니다. OMG Group 은 곧 일반 프로세스 모델을 제안할 예정입니다. "프로세스"란 용어는 소프트웨어 엔지니어링 커뮤니티에서 널리 사용되는 용어이지만 소프트웨어 프로세스의 컴포넌트에 대한 가정은 일관적이지 않습니다. SEI 에서는 프로세스를 소프트웨어 및 연관 제품을 개발하고 유지보수하기 위해 사용하는 활동, 메소드, 사례 및 변환 세트로 정의합니다. 즉, 프로세스는 프로젝트가 정의된 목표를 달성할 수 있도록 해주는 핵심 요소입니다.

이 정의에 활동, 기술(즉, 메소드 및 도구)과 사람이 포함된다는 점에 유의해야 합니다. 이 세 가지 컴포넌트는 다른 어떤 컴포넌트보다도 중요합니다.

Rational Unified Process

RUP 는 오랜 프로젝트 경험에서 얻은 이점이 반영된 기존 프로세스 프레임워크의 한 예입니다[10, 11].

RUP 에서는 해당 아키텍처를 정의하는 시스템의 초기 버전을 신속하게 개발함으로써 고위험성 영역의 초기 처리를 강조합니다. RUP 에서는 프로젝트 도입/인식(Inception) 단계에서 요구사항 세트가 명확하게 확립되지 않으며 프로젝트 진행에 따라 요구사항을 정제할 수 있는 것으로 가정합니다. 또한 변화를 예상하고 수용합니다. 프로세스는 문서 또는 의식(ceremony)을 지나치게 강조하지 않으며 소프트웨어 개발과 연관된 많은 지루한 작업을 자동화합니다. 기본 초점은 소프트웨어 제품 자체와, 일반 사용자를 만족시키고 투자 수익 목표를 모두 충족시키는 정도로 측정한 해당 품질입니다.

프로세스는 크기 및 응용프로그램 도메인 측면 모두에서 다양한 소프트웨어 제품 및 프로젝트에 맞게 사용자 조정되며 세 가지 영역, 즉 사람, 프로세스 및 도구 또는 메소드에 집중합니다[10,11].

ISO 문서

마지막으로, 개선 로드맵을 완벽하게 설명하려면 ISO 9001, ISO 12207, ISO 15504(SPICE)[9]와 같은 ISO 문서에 대해 언급해야 합니다. 조직은 이러한 문서를 참조하여 자사와 다른 조직의 소프트웨어 개발 노하우를 비교할 수 있습니다. ISO 문서는 앞에서 언급한 다른 컴포넌트가 준수하는 일반 프레임워크입니다.

ISO 15504(SPICE 라고도 함)는 소프트웨어 프로세스를 분석하기 위한 또 다른 참조 모델입니다. 이 모델에서는 완전 실현된 많은 평가 모델(예: CMM)과 많은 평가 메소드(예: SEI 에서 정의한 메소드)를 ISO 15504 의 표준 파트에 맵핑할 수 있는 것으로 가정합니다.

이 문서의 나머지 부분에서는 소프트웨어 커뮤니티의 실제 표준인 CMM 에 초점을 맞춥니다. CMM 은 수백 가지 독창적인 소프트웨어 프로세스 개선 방법을 지원하는 완전 실현 모델입니다. 모든 내용은 SPICE 를 준수합니다.

SPI 에 RUP 사용

이전 섹션에 나열된 핵심 요소(CMM 참조 모델, 개선 프로그램 완료를 관리하는 메소드(IDEAL) 및 경쟁력을 향상시키고 소프트웨어의 글로벌 과제를 해결할 수 있는 수정 개발 프로세스)를 고려하여, RUP 의 개념을 CMM 요구사항으로 맵핑하고 특정 소프트웨어 기능을 달성하는 데 있어 RUP 의 잠재력을 검토합니다.
소프트웨어 기능은 실제로 다음과 같은 조직의 역량을 의미합니다.

- 적정 시점의 프로젝트 성공을 위한 올바른 결정
- 비즈니스 성공 및 생존
- 위험 감수 및 프로젝트 산출물 제어
- 우수한 제품 품질 얻기
- 최신 소프트웨어 기술의 제품 통합

소프트웨어 기능은 위의 역량이 정교하게 혼합된 형태이지만 다음 개념 역시 소프트웨어 기능의 일부로 간주해야 합니다.

- 프로젝트 레벨:
  라이프사이클 문제: 프로젝트 라이프사이클은 제품 구현 및 유지보수만으로 한정되지 않습니다. 비즈니스, 재무 및 프로젝트 전략을 처리하는 관리 측면이 있습니다.

  프로세스 선택 및 사용자 조정: 기존 및/또는 과거 경험을 반영한 표준 프로세스는 다양한 프로젝트의 목표를 충족시킬 수 없습니다. 효율성은 특정 오브젝트에 적합한 개발 프로세스를 정의함으로써 얻을 수 있는 결과입니다.

  위험성 관리: 모든 프로젝트에는 위험성이 있습니다. 유능한 조직은 위험성을 예측하여 발생한 위험성의 영향을 최소화할 수 있도록 프로젝트를 재구성합니다.

  측정: 소프트웨어 조직에서 장기 진행상태를 파악할 수 있는 방법 중 하나는 히스토리 데이터를 수집하여 소프트웨어 품질 및 생산성을 분석하는 것입니다. 각 프로젝트마다 히스토리 데이터를 수집하려면 많은 시간이 소요됩니다. 노력, 스케줄, 프로그램 크기 및 구현 기능에 대한 데이터와 결함 수가 향후 프로젝트 계획 및 정확한 예측을 위한 견고한 기반이 됩니다. 예측 성능은 성숙도를 나타내는 척도이기도 합니다.

- 조질 레벨 – 프로젝트 콜렉션으로 표시:
  프로세스 개선: 기능은 조직이 과거, 특히 다른 조직의 실수로부터 교훈을 얻을 수 있으며 이렇게 얻은 교훈을 프로세스 개선으로 변환할 수 있음을 의미합니다. 개선 반복 주기는 개선을 정량화할 수 있는 척도가 정의된 경우에만 완료됩니다.

  자원 및 스킬: 조직의 기능은 프로세스를 적용하고 제품을 개발하는 담당자의 능력과 밀접한 관련이 있습니다.

RUP 는 프로젝트 레벨에서 CMM 요구사항을 충족시킵니다.

CMM 은 조직에서 특정 성숙도 레벨에 도달하는 프로젝트 상태에 대해 설명합니다.

레벨 2 조직의 프로젝트에서는 기본 소프트웨어 관리 제어가 설치됩니다. 현실적인 프로젝트 약속은 이전 프로젝트에서 관찰한 결과와 현재 프로젝트의 요구사항을 기반으로 합니다. 소프트웨어 프로젝트 관리자는 소프트웨어 비용, 스케줄 및 기능성을 추적합니다. 약속 이행과 관련된 문제점은 실제로 발생할 때 식별됩니다. 소프트웨어 요구사항 및 이러한 요구사항을 충족시키기 위해 개발된 중간 산출물이 기준선이 되므로 해당 무결성은 제어됩니다. 소프트웨어 프로젝트 표준이 정의되며 조직에서 해당 표준을 성실하게 준수하는지 확인합니다. 소프트웨어 프로젝트는 해당 하청업체(있는 경우)와 협력하여 강력한 고객–공급자 관계를 형성합니다.

레벨 3(정의됨 레벨)에서는 조직에서 소프트웨어를 개발하고 유지보수하는 표준 프로세스가 문서화됩니다. 조직은 프로젝트를 통해, 정의된 자체 소프트웨어 프로세스를 개발하기 위한 표준 소프트웨어 프로세스를 사용자 조정하며 자체 소프트웨어 프로세스는 프로젝트의 고유 특성을 나타냅니다. CMM 에서는 이렇게 사용자 조정된 프로세스를 프로젝트의 정의된 소프트웨어 프로세스라고 합니다. 정의된 소프트웨어 프로세스에는 잘 정의된 소프트웨어 엔지니어링 및 관리 프로세스의 일관된 통합 세트가 포함됩니다. 잘 정의된 프로세스는 작업 수행을 위한 준비 완료 기준, 입력, 표준 및 프로시저, 검증 메커니즘(예: 피어 검토), 출력 및 완료 기준을 포함하는 것으로 규정될 수 있습니다. 소프트웨어 프로세스는 잘 정의된 프로세스이므로 조직 경영진은 모든 프로젝트에 대한 기술적 진행상태를 정확하게 파악할 수 있습니다.

라이프사이클 문제

CMM 에서는 라이프사이클 문제를 여러 KPA(Key Process Area)에서 다룹니다. CMM 개념을 처음 접한 조직에서는 일반적으로 "소프트웨어 프로세스"를 잘못 이해하여 "소프트웨어 라이프사이클"과 같은 것으로 생각합니다. 프로젝트 라이프사이클에는 개발 문제만큼 많은 비즈니스 및 재무 문제가 포함됩니다. 프로젝트 라이프사이클은 레벨 2 KPA(Key Process Area) 및 일부 레벨 3 KPA 와 관련이 있습니다. 예를 들어, 다음과 같습니다.

- 요구사항 관리 목적 1: 소프트웨어 엔지니어링 및 관리 사용을 위한 기준선을 설정하기 위해 소프트웨어에 할당된 시스템 요구사항을 제어합니다.
- 소프트웨어 프로젝트 계획 목적 1: 소프트웨어 예상을 문서화함으로써 소프트웨어 프로젝트 계획을 수립하고 추적합니다.
- 소프트웨어 프로젝트 계획 목적 2: 소프트웨어 프로젝트 활동 및 약속 계획을 수립하고 문서화합니다.
- 소프트웨어 프로젝트 추적 목적 1: 소프트웨어 계획에 대한 실제 결과 및 성과를 추적합니다.
- 소프트웨어 프로젝트 추적 목적 2: 관련 그룹 및 개인으로부터 소프트웨어 약속 변경사항에 대한 동의를 얻습니다.
- 소프트웨어 형상 관리 목적 1: 선택한 소프트웨어 중간 산출물을 식별, 제어하고 사용 가능하게 합니다.
- 소프트웨어 형상 관리 목적 2: 식별된 소프트웨어 중간 산출물에 대한 변경사항을 제어합니다.
- 소프트웨어 형상 관리 목적 3: 관련 그룹 및 개인에게 소프트웨어 기준선의 상태와 컨텐츠를 알려줍니다.
- 소프트웨어 제품 엔지니어링 목적 1: 소프트웨어 엔지니어링 타스크를 정의, 통합하고 지속적으로 수행함으로써 소프트웨어를 생성합니다.
- 소프트웨어 제품 엔지니어링 목적 2: 소프트웨어 제품을 서로 일치하도록 유지합니다.

즉, CMM 의 특성은 다음과 같습니다.

1. 개발될 응용프로그램의 비즈니스 환경을 고려하여 암묵적으로 기준선을 정의하며 이를 통해 프로젝트 요구사항의 우선순위를 설정합니다. 또한 프로젝트의 기능적, 비기능적 컨텐츠와 관련된 모든 필수 결정을 내립니다.

2. 프로젝트 계획이 수립되며 활동 목록, 역할 및 책임, 이정표와 현실적이고 신뢰할 수 있는 아티팩트를 설정해야 합니다. 프로젝트 계획은 위험성을 고려합니다.

3. 프로젝트 계획을 모니터하여, 기능 컨텐츠 또는 프로젝트 환경/조직이 계획과 많이 다른 경우 프로젝트 팀이 대응합니다.

4. 모든 프로젝트 변경사항을 식별하고 분석합니다. 추가 의사결정을 수행하고 기준선 및 프로젝트 계획에 보고합니다.

5. 프로젝트의 최대 성능을 나타낼 수 있는 메소드 및 도구를 선택하여 적용합니다.

Rational Unified Process 는 도입/인식(Inception) 단계의 주요 타스크에서 다음과 같이 이러한 요구사항을 이행합니다.

- 이전 목록의 항목 1 에 따라 프로젝트 비전을 가져오고 비즈니스 모델을 개발합니다.
- 프로젝트를 구성하고 해당 계획을 수립합니다.
- 잠재적 위험성을 예상합니다. (이 항목은 이전 목록의 항목 2 와 관련이 있습니다.)
- 이전 목록의 항목 3, 4 와 유사한 통합 변경 관리 정책을 전개합니다.

도입/인식(Inception) 단계의 가장 중요한 목적은 프로젝트의 라이프사이클 목표에 대해 모든 이해 당사자(stakeholder) 간의 의견 일치를 달성하는 것입니다. 도입/인식 단계는 주로 새로운 개발 노력에 중요합니다. 이러한 경우 프로젝트를 진행하려면 중요한 비즈니스 및 요구사항 위험성을 처리해야 합니다. 기존 시스템의 개선에 중점을 두는 프로젝트의 경우 도입/인식 단계가 보다 간단하지만 이 때도 역시 프로젝트의 수행 가치 및 가능성을 확인하는 데 초점을 맞추어야 합니다.
도입/인식 단계 종료 시점에는 프로젝트 라이프사이클 목표를 점검하여 프로젝트를 계속 진행할지 또는 취소할지 여부를 결정합니다.

검토할 필수 아티팩트는 다음과 같습니다.

- 프로젝트 비전
- 비즈니스 사례
- 위험성 목록(위험성 관리 섹션 참조)
- 소프트웨어 개발 계획
- 반복 계획
- 개발 사례(프로세스 선택 및 사용자 조정 섹션 참조)

이러한 아티팩트는 이 섹션 처음에 나열된 여러 가지 CMM KPA(Key Process Area) 목적을 충족시킵니다.

프로젝트 비전 수립 및 비즈니스 모델 개발

비즈니스 모델은 비즈니스 작업자의 내부 관점에서 비즈니스 유스 케이스를 정의합니다. 이 모델은 비즈니스 구성원과 이들이 "비즈니스 클래스 및 오브젝트"를 동적, 정적으로 처리 및 사용하여 예상 결과를 생성하는 방법과의 올바른 관계를 정의합니다. 이 모델은 또한 비즈니스 영역에서 수행되는 역할과 역할별 필요한 책임을 강조합니다. 또한 모델 클래스의 오브젝트가 모든 비즈니스 유스 케이스를 수행할 수 있어야 합니다.

Rational Unified Process 는 비즈니스 모델을 기반으로 비전 개발 타스크를 식별합니다. 이 타스크의 목적은 다음과 같습니다.

- 해결해야 하는 문제점에 대한 동의를 얻습니다.
- 시스템 이해 당사자(stakeholder)를 식별합니다.
- 시스템 경계를 정의합니다.
- 시스템의 기본 기능에 대해 설명합니다.

프로젝트 구성 및 계획

소프트웨어 프로세스가 프로젝트 특성의 영향을 받는 것처럼 프로젝트 조직도 마찬가지입니다. 여기에 표시된 기본 구조(그림 1 참조)를 아래 나열된 요인의 영향을 반영하여 수정해야 합니다.

- 비즈니스 컨텍스트
- 소프트웨어 개발 노력의 규모
- 제품의 참신도
- 응용프로그램 유형
- 현재 개발 프로세스
- 조직 요소
- 기술 및 관리 복잡도

RUP 에서는 이 요인을 프로젝트 구조 선택에 영향을 주는 프로세스 판별 요소로 고려합니다.

프로젝트 구조는 주로 다음과 같은 요소로 정의됩니다.

1. 각 반복의 길이
2. 반복 횟수

반복은 모든 주요 워크플로우를 거쳐 대부분의 경우 아직 불완전하지만 실행 가능한 릴리스 시스템을 생성하는 완전한 미니 프로젝트입니다.

반복은 위험성, 크기 및 복잡도에 따라 여러 가지 변형이 가능하며 그에 따라 반복 횟수도 결정됩니다.

제품이 전혀 새로운 도메인용 제품인 경우 도입/인식(Inception) 단계에서 개념을 정립하거나 고객 또는 일반 사용자에게 다양한 실물 모형을 보여주거나 제안 요청에 대한 명확한 응답을 제공하기 위해 일부 반복을 추가해야 할 수 있습니다.

제품이 크고 복잡하며 장기간에 걸쳐 개발되는 경우 구현/구축(Construction) 단계에서 반복을 세 번 이상 수행하는 것으로 계획해야 합니다.

프로젝트의 수명 중 조직은 프로젝트 계획에서 캡처한 작업분류체계(WBS)를 지원하도록 발전합니다. 이 내용은 그림 1, [7]을 참조하십시오.

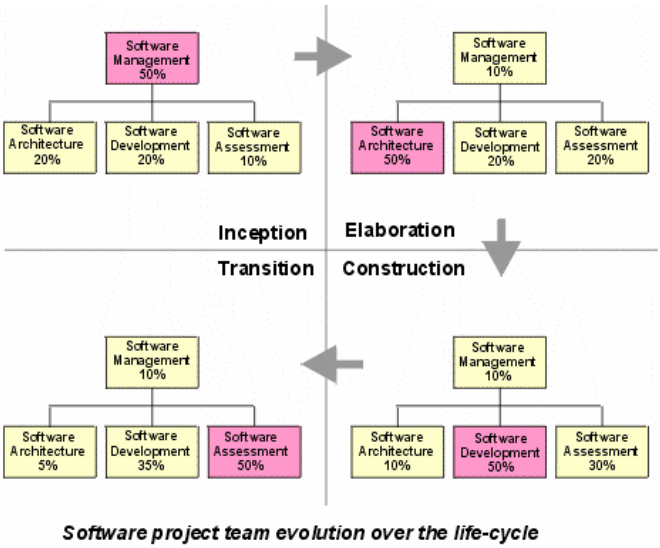

그림 1: 소프트웨어 프로젝트 팀 발전

이 발전은 각 단계에서 다른 활동 세트를 강조합니다.

- 도입/인식(Inception) 팀: 다른 팀의 적극적인 지원을 받아 모든 관점에 대한 동의를 반영한 계획에 초점을 맞추는 조직
- 정제(Elaboration) 팀: 소프트웨어 아키텍처 팀이 프로젝트를 주도하는 아키텍처 중심 조직. 소프트웨어 아키텍처 팀은 안정적인 아키텍처 기준선을 구축하기 위해 소프트웨어 개발 및 소프트웨어 평가 팀의 지원을 받습니다.
- 구현/구축(Construction) 팀: 소프트웨어 개발 및 소프트웨어 평가 팀에서 대부분의 활동을 수행하는 균형 잡힌 조직.
- 전이(Transition) 팀: 사용 피드백으로 배치 활동을 전개하는 고객 중심 조직

이 발전 과정에서 팀 간 이주를 통해 프로젝트 지식과 기능을 보유할 수 있습니다.

통합 변경 관리(UCM) 정책 배치

통합 변경 관리(UCM)는 요구사항에서 릴리스까지 소프트웨어 시스템 개발의 변경을 관리하는 Rational Software 의 접근 방식입니다. UCM 은 요구사항, 디자인 모델, 문서, 컴포넌트, 테스트 케이스 및 소스 코드의 변경 관리 방법을 정의하는 개발 라이프사이클에 적용됩니다.

UCM 모델의 핵심 측면 중 하나는 프로젝트 진행상태를 계획하고 추적하는 데 사용되는 활동과 변경 아티팩트를 통합하는 것입니다. UCM 모델은 프로세스 및 도구에 의해 실현됩니다. UCM 은 Rational ClearCase 및 Rational ClearQuest 기술을 기반으로 합니다. ClearCase 는 시스템 아티팩트 및 프로젝트 관리 아티팩트를 모두 포함하는 소프트웨어 프로젝트에서 생성되는 모든 아티팩트를 관리합니다. ClearQuest 는 프로젝트 타스크, 결함 및 개선사항 요청(일반적으로 활동이라고 함)을 관리하고, 프로젝트 진행상태를 추적하는 데 필요한 차트 및 보고 도구를 제공합니다.

프로세스 선택 및 사용자 조정

성숙도 레벨 3 에는 전체 조직을 대상으로 하는 세 가지 KPA 와 프로젝트 조직, 관리 및 엔지니어링을 대상으로 하는 네 가지 KPA 가 포함됩니다. 관리 및 엔지니어링 활동을 위한 소프트웨어 프로세스는 문서화되고 표준화되어 조직 전체의 소프트웨어 프로세스에 통합됩니다. 모든 프로젝트에서 소프트웨어 개발 및 유지보수를 위해 사용하는 조직 프로세스는 문서화되고 승인된 버전입니다. 조직은 OPF(Organization Process Focus), OPD(Organization Process Definition) 및 TP(Training Program) KPA 를 통해 레벨 2 의 우수 프로젝트 사례를 식별하고 해당 사례를 조직 표준으로 문서화할 수 있습니다. 이러한 KPA 는 또한 프로젝트에서 식별된 요구, 이전 프로젝트에서 얻은 교훈 및 조직의 향후 미션과 비전을 기반으로 하는 스킬 관리에 초점을 맞춥니다. 그러나 특정 프로젝트에서 성공을 달성하고 위험성을 관리하며 성능을 개선하기 위해 표준 프로세스를 수정하는 가이드라인이 없는 경우 성숙도가 레벨 3 요구사항을 충족시키지 않습니다.

따라서 대부분의 앞선 평가자와 CMM 커뮤니티에 있어, 레벨 3 의 특성은 소프트웨어 엔지니어링 및 관리 활동을 일관적이고 정의된 소프트웨어 프로세스로 통합하는 ISM(Integrated Software Management) KPA 입니다. 이 정의된 프로세스는 고객 요구 및 제한조건과 시장 요구를 충족시키고 비즈니스 전략을 수행하는 데 도움을 주는 표준 프로세스를 가장 적절하게 사용자 정의한 프로세스입니다.

ISM 은 CMM 모델의 두 목표로 정의됩니다. 첫 번째 목표에는 사용자 조정이 포함되고 두 번째 목표에는 프로젝트 관리가 필요합니다. ISM 의 활동 10(프로젝트의 소프트웨어 위험성은 문서화된 프로시저에 따라 식별, 평가, 문서화 및 관리됨)은 레벨 3 조직의 두 번째 핵심 지표입니다. 위험성 관리는 레벨 2 에서 필요하지만 레벨 3 조직은 이미 위험성을 보다 정확하게 예측했으며 기업의 의사결정 시스템을 나타낼 수 있습니다. 아래 위험성 관리 섹션은 이러한 측면에 대해 설명합니다.

RUP 에서는 개발 사례 및 환경 원칙의 핵심 개념이 이러한 요구사항을 이행합니다. 개발 사례는 Rational Unified Process 제품의 정적 구성입니다. 즉, 특정 프로젝트, 제품 및 조직에 맞게 사용자 조정된 소프트웨어 엔지니어링을 위한 비즈니스 프로세스입니다. 개발 사례는 수행 내용과 수행 방법에 초점을 맞추며 수행될 프로세스의 개요를 제공 합니다. 이 개요는 모든 프로젝트 관계자가 이해할 수 있습니다.

Rational Unified Process 는 특정 개발 사례에서 수정, 사용자 정의, 추가 또는 억제될 수 있는 프로세스 구성요소를 나열합니다.

- **원칙**
  소프트웨어 프로젝트는 분석 및 디자인, 구현 등과 같은 기본 원칙을 완전히 생략하는 일이 거의 없습니다.

- **아티팩트**
  프로젝트는 프로젝트에서 생성, 갱신 및 전달해야 하는 아티팩트에 따라 달라질 수 있습니다.

- **타스크**
  타스크는 두 가지 이상의 이유로 달라질 수 있습니다. 즉, 아티팩트를 입력으로 사용하고 아티팩트를 출력으로 생성 또는 갱신하는 타스크는 해당 아티팩트 수정에 따라 영향을 받습니다. 특히 일부 아티팩트 또는 아티팩트의 일부 정보 요소가 더 이상 필요하지 않은 경우 해당 단계는 억제되거나 크게 수정될 수 있습니다. 타스크는 특정 응용프로그램 도메인 또는 개발 전문 기술(예: 디자인 단계, 프로그래밍 언어, 자동 코드 생성 도구, 측정 기법 등)과 관련된 특정 기법, 메소드 및 도구를 도입하기 위한 목적으로도 수정됩니다.

프로세스 엔지니어는 프로세스를 구성하고 개발 프로세스 구조를 결정하며 개발 조직(팀, 프로젝트 또는 회사)에 개발 사례를 "설치"하고 개발자에게 개발 사례를 사용하는 방법을 가르쳐줍니다.

프로세스 엔지니어가 개발 사례를 설정하면 프로젝트 관리자가 해당 사례를 인스턴스화하여 해당 프로젝트에 실행합니다. 이것을 보통 프로세스 규정이라고 부릅니다.

프로세스가 전개되면서 해당 프로세스에서도 교훈을 얻을 수 있습니다. 프로세스 엔지니어는 이 교훈을 피드백으로 사용하여 프로세스를 개선합니다.

위험성 관리

위험성 관리는 CMM 에서도 주로 레벨 2 에서 다뤄지며 레벨 3 에서는 보다 깊이있게 다뤄집니다. 소프트웨어 프로젝트 계획, 소프트웨어 프로젝트 추적 및 감독, ISM(Integrated Software Management) KPA(Key Process Area)의 목적을 실현하기 위한 특정 위험성 활동이 나열됩니다. CMM 요구사항은 여러 조직에서 관찰된 사례를 따릅니다. 레벨 2 의 프로젝트는 일반적으로 정기적으로 위험성을 식별하고 평가하지만 그다지 활발하게 수행되지는 않습니다. 즉, 레벨 2 에서는 아직 위험성 관리를 적극 수행/이해하지 않으므로 프로젝트 사후 관리 및 조기 위험성 식별과 관련된 문제점이 있습니다.

반대로 레벨 3 의 조직에서는 위험성을 식별, 평가 및 완화하며 팀에서 정확한 예측을 할 수 있습니다. 대부분의 상위 팀에서는 정량적 척도를 사용하여 의사결정을 수행합니다.

CMM 기반 평가 및 SPI(Software Process Improvement)에 대한 오랜 경험을 통해 위험성 관리에 있어 소프트웨어 단위가 직면한 어려움을 파악할 수 있습니다. 이는 프로젝트 제어 및 측정을 위한 모든 노력에도 불구하고 해결되지 않는 약점이기도 합니다.

RUP 및 반복 접근 방식 사용은 프로젝트 위험성 관리 측면에서 매우 효과적입니다. 전체 Rational Unified Process 는 위험성 관리를 기반으로 합니다.

위험성을 처리하기 위해 반복 계획을 세웁니다. 위험성을 제한하거나 줄이면서 특정 위험성을 처리하도록 반복을 계획합니다. 위험성 목록을 정기적으로 검토하여 위험성 완화 전략의 효과를 평가합니다. 또한 이를 통해 프로젝트 계획에 대한 개정 및 후속 반복 계획이 수행됩니다.

위험성 관리의 핵심은 위험성이 구체화(문제점 또는 장애)되기 전에 해결 방법을 결정하는 것입니다. 대륙을 횡단하는 비행 경로에서는 각도가 약간만 변경되어도 착륙 장소가 크게 달라지는 것과 같이, 위험성이 구체화되기 전에 조기에 발견하여 관리함으로써 비용 및 수고를 덜 수 있습니다.

위험성은 다음과 같이 네 가지 카테고리로 분류할 수 있습니다.

1. 자원 위험성
   - 조직
   - 자금 지원
   - 사람
   - 시간
2. 비즈니스 위험성
3. 기술적 위험성
   - 범위 위험성
   - 기술적 위험성
   - 외부 종속성 위험성
4. 스케줄 위험성

위험성 관리 계획 및 위험성 목록은 모두 RUP 의 프로젝트 관리 원칙으로 식별되는 아티팩트입니다. 두 아티팩트 모두 위험성을 식별하고 평가하도록 권장합니다.

측정

측정은 CMM 의 공통 기능 중 하나입니다. 각 KPA(Key Process Area)에 대해 레벨 2 에서 레벨 5 까지 공통 기능이 존재하며 사례가 규정되는 시점을 나타냅니다. 메트릭은 SW-CMM 의 새 버전(즉, CMMI 프레임워크에 임베드된 버전, 통합 CMM 시스템 엔지니어링 + 소프트웨어 엔지니어링)에 각 KPA(Key Process Area)에 대한 측정 기반 목표가 있을 정도로 중요합니다.

CMM 의 핵심 기능은 프로세스를 측정하여 레벨 2 의 적합성과 레벨 3 의 유효성을 확보하고 있는지 판별하는 것입니다. 메트릭 보고에서 분석 및 조치 수행으로의 전환은 일반적으로 어려운 일이지만 프로젝트가 성숙해지고 있음을 나타내는 명백한 증거입니다. 적은 규모라도 프로젝트를 정량적으로 관리하는 데 성공함으로써 측정의 이점을 수용하고 이해할 수 있습니다.

이러한 일반 요구사항 이외에도 일부 KPA(Key Process Area)에는 소프트웨어 프로젝트 계획, 소프트웨어 프로젝트 추적 및 감독, ISM(Integrated Software Management)과 같은 고유한 측정 요구사항이 있습니다. 이러한 요구사항은 프로젝트 예상 및 프로젝트 제어를 가능하게 하는 프로젝트 데이터와 관련이 있습니다.

RUP 는 메트릭 사용 방법을 안내합니다. 측정 계획 및 측정 아티팩트는 도입/인식(Inception) 단계에서 생성되어야 합니다.

프로젝트 측정 아티팩트는 프로젝트의 메트릭 데이터를 저장합니다. 척도가 작성되거나 사용 가능하게 된 경우 최신 상태로 유지됩니다. 또한 기본 데이터에서 계산되는 파생 메트릭이 포함되며, 파생 메트릭을 얻는 방법에 대한 정보(예: 프로시저 및 알고리즘)를 저장해야 합니다. 프로젝트 상태(예: 목적(기능성 및 품질 등)에 대한 진행상태, 비용 및 기타 자원 이용)에 대한 보고서는 프로젝트 측정을 사용하여 생성됩니다. 예를 들어, 자동 소프트웨어 데이터 콜렉션 에이전트가 프로젝트 상태를 실시간으로 표시하는 Rational Project Dashboard 접근 방식을 사용하여 프로젝트 상태를 자주 또는 지속적으로 표시할 수 있습니다.

Rational 문서에는 간단한 구조로 쉽게 시작할 수 있는 초기 메트릭 세트가 제공됩니다. 프로젝트의 특정 측면에 대한 메트릭에는 다음 내용이 포함됩니다.

- 크기 및 복잡도 진행상태
- 요구사항 또는 구현의 변경 비율, 크기 또는 복잡도의 안정성
- 변경 범위의 모듈화
- 오류 유형 및 그 수와 관련된 품질

- 오류 빈도의 성숙도
- 프로젝트 자원 지출 대 계획된 지출

RUP 는 조직 레벨에서 CMM 요구사항을 충족시킵니다.

레벨 2 조직의 경우 소프트웨어 프로젝트의 계획 및 추적 작업이 안정적이고 이전 성공을 반복할 수 있으므로 해당 소프트웨어 프로세스 기능을 규정된 기능으로 요약할 수 있습니다. 프로젝트 프로세스는 이전 프로젝트 성과를 기반으로 한 현실적인 계획에 따라 프로젝트 관리 시스템에서 효과적으로 제어합니다.

정의됨 레벨(레벨 3)에서는 소프트웨어 엔지니어링 및 관리 프로세스를 모두 포함하여 조직에서 소프트웨어를 개발하고 유지보수하는 표준 프로세스가 문서화되며 해당 프로세스는 일관된 표준으로 통합됩니다. 표준 프로세스는 CMM 전체에서 조직의 표준 소프트웨어 프로세스로서 참조됩니다. 레벨 3 에서 확정된 프로세스는 소프트웨어 관리자 및 기술 담당자의 보다 효과적인 역할 수행을 돕기 위해 사용되고 필요한 경우 변경됩니다. 조직은 해당 소프트웨어 프로세스를 표준화할 때 실질적인 소프트웨어 엔지니어링 사례를 이용합니다. 조직의 소프트웨어 프로세스 활동(예: 소프트웨어 엔지니어링 또는 SEPG)을 담당하는 그룹이 있습니다. 담당자 및 관리자가 지정된 역할을 수행하는 데 필요한 지식과 스킬을 갖추도록 하기 위해 조직 전체를 대상으로 한 훈련 프로그램이 구현됩니다.

레벨 3 조직의 경우 소프트웨어 엔지니어링 및 관리 활동이 모두 안정적이고 반복 가능하므로 해당 소프트웨어 프로세스 기능을 일관적인 표준 기능으로 요약할 수 있습니다. 또한 구축된 제품 라인 범위에서 비용, 스케줄 및 기능성을 제어할 수 있으므로 소프트웨어 품질을 추적할 수 있습니다. 이 프로세스 기능은 정의된 소프트웨어 프로세스에서의 활동, 역할 및 책임에 대한 조직 전체의 공통 이해를 기반으로 합니다.

프로세스 개선

기능은 조직이 과거, 특히 다른 조직으로부터 교훈을 얻을 수 있으며 얻은 교훈을 프로세스 개선으로 변환할 수 있음을 의미합니다. 개선 반복 주기는 개선을 정량화할 수 있는 척도가 정의된 경우에만 완료됩니다.

조직의 성숙도가 향상되면 표준 프로세스가 변경됩니다. 레벨 2에서 레벨 3으로의 이동은 프로젝트의 모든 올바른 사례가 규정되었으며 프로젝트에서 우수 사례를 식별하는 데 도움이 될 평가 프로세스가 있음을 의미합니다. 해당 사례는 OSSP(Organization Standard Software Process)에 문서화됩니다. 이는 OPF(Organization Process Focus) KPA(Key Process Area)의 요구사항입니다. OSSP는 조직 내 프로젝트에서 얻은 교훈을 기반으로 정제됩니다. 여러 프로젝트에서 동일한 표준을 사용하므로 경험과 교훈을 쉽게 축적할 수 있으며 OSSP는 이를 통해 다양한 경험의 이점을 활용할 수 있습니다.

RUP 환경 원칙은 유사한 접근 방식을 전개합니다. "조직 내 프로세스 구현" 개념은 개발 조직에서 프로세스 및 도구를 구현하기 위해 조직 레벨에서 수행되는 작업에 대해 설명합니다.

소프트웨어 개발 조직에서의 새 프로세스 구현은 RUP 의 네 단계인 도입/인식(Inception), 정제(Elaboration), 구현/구축(Construction) 및 전이(Transition) 단계를 사용하여 설명할 수 있습니다.

<u>조직의 프로세스 및 도구를 구현하는 단계</u>

CMM 의 OSSP 는 조직의 새 프로세스이므로 해당 정의는 RUP 의 네 단계를 따를 수 있습니다.

필요한 적응을 위해 각 소프트웨어 개발 프로젝트에서 사용할 수 있는 조직 전체 개발 환경을 개발할지 여부에 대한 결정이 중요하지만 특정 성숙도 레벨이 존재해야 합니다.

조직 전체 환경을 개발하려는 경우, 조직의 개발 환경을 개발하는 프로젝트를 시작해야 합니다. 또한 이러한 프로젝트를 시작하려는 경우, 해당 프로젝트 팀과 소프트웨어 개발 프로젝트 팀의 긴밀한 협조가 필요합니다. RUP 에서는 또한 이를 특정 프로젝트로 간주하도록 권장하며 이를 통해 CMM 요구사항을 다시 이행합니다.

프로세스 구현 프로젝트는 여러 단계로 분할됩니다. 프로젝트 준비가 완료될 때까지 각 단계별로 모두 네 단계가 수행되며, 도구가 배치되어 전체 조직에서 사용합니다.

프로세스 구현 프로젝트는 단계로 분할할 수 있습니다.
네 단계에서 수행되는 작업은 다음과 같습니다.

- 단계 1: 프로세스 구현 프로젝트를 후원자에게 판매합니다.
- 단계 2: 주요 위험성을 처리합니다.
- 단계 3: 모든 작업을 완료합니다. 개발 사례의 템플리트, 가이드라인, 예제 준비가 완료되며 훈련 과정이 완성됩니다.
- 단계 4: 전체 조직에 배치합니다.

이 단계는 Rational Unified Process를 사용하는 소프트웨어 개발 프로젝트에 적용되므로 각각 도입/인식(Inception), 정제(Elaboration), 구현/구축(Construction) 및 전이(Transition) 단계라고 할 수 있습니다.

특정 프로젝트에서 얻은 교훈 또는 특정 기술에 따라 해당 성능을 향상시키기 위해 표준 프로세스를 발전시켜야 할 때마다 동일한 프로젝트 단계가 정의됩니다.

RUP 는 또한 SPI(Software Process Improvement)의 전체 배경이 되는 조직 변경 관리 개념을 정의합니다. 성공적인 프로세스 변경 구현을 위한 권장사항은 다음과 같습니다.

- 조직 내 다양한 레벨에서 변경 에이전트를 식별합니다.
- 합리적이고 측정 가능한 소규모 단계로 변경을 계획합니다.
- 조직 레벨에 적합한 기본 레벨 언어를 사용하여 변경사항을 알립니다.

이러한 권장사항은 모태가 되는 IDEAL 접근 방식의 권장사항과 유사합니다.

마지막으로 RUP 는 "조언자"라는 특정 액터를 정의합니다. 조언자는 프로젝트 팀에 필요한 것이 무엇이고 언제 필요한지를 안내하고 교육하는 사람입니다. 일반적인 조언 방법은 다음과 같습니다.

- 워크샵 리더
  일부 활동은 그룹에서 가장 잘 수행됩니다(예: 유스 케이스 모델링 중 액터 및 유스 케이스 찾기). 이와 같은 활동에서는 프로세스 전문가인 모델링 리더가 있는 것이 좋습니다.

- 프로세스 전문가
  프로세스 전문가는 프로젝트의 현장 지원 인원입니다. 프로세스 전문가의 타스크는 개발자가 프로세스와 가능하면 모델도 사용할 수 있도록 돕는 것입니다.

- 프로젝트 관리자 지원
  프로세스 전문가가 프로젝트 관리자의 프로젝트 계획 및 진행을 도울 수 있습니다. 간혹 프로젝트 관리자는 문제가 있는 프로세스에 대한 경험이 적거나 전혀 없습니다.

- 검토자
  프로세스 전문가가 각 단계의 결과를 검토함으로써 비용 효율이 높은 방식으로 지식을 전달할 수 있습니다. 프로세스 검토자는 또한 특정 프로젝트에서 수행된 프로세스 적응을 검토하는 데 도움이 됩니다.

이 조언자 액터는 CMM 의 SEPG(Software Engineering Process Group)와 같은 유형의 역할을 수행합니다.

자원 및 스킬

CMM 은 조직 레벨에서 "훈련 프로그램" KPA(Key Process Area)를 식별합니다. 여기서는 실제 관심사항을 고려하여 "스킬 관리"로 이름을 지정할 수 있습니다. 성숙한 조직은 스킬 요구를 식별하여 중장기적인 계획을 수립해야 하며 이러한 요구가 정해진 시간 내에 충족될 수 있도록 관리해야 합니다.

훈련 프로그램의 목적은 개인이 해당 역할을 효과적이고 효율적으로 수행하는 데 필요한 스킬과 지식을 개발하는 것입니다. 훈련은 조직의 책임이지만 소프트웨어 프로젝트에서 필요한 스킬을 식별해야 하며 프로젝트가 고유해야 하는 경우 필요한 훈련을 제공해야 합니다. RUP 의 관리 원칙은 프로젝트 인력 충원 시 이러한 문제를 고려합니다.

훈련 프로그램 KPA(Key Process Area)에는 다음과 같은 세 가지 목적이 있습니다.

- 목적 1: 훈련 활동을 계획합니다.
- 목적 2: 소프트웨어 관리 및 기술 역할을 수행하는 데 필요한 스킬과 지식을 개발하기 위한 훈련을 제공합니다.
- 목적 3: 소프트웨어 엔지니어링 그룹 및 소프트웨어 관련 그룹에 속하는 개인은 해당 역할을 수행하는 데 필요한 훈련을 받습니다.

RUP 의 여러 측면이 이러한 요구사항을 이행합니다. 역할을 정확히 식별하고 해당 역량을 정의합니다. 역할은 일반적으로 개인 또는 팀을 이루어 작업을 수행하는 소수의 개인에 의해 실현됩니다. 프로젝트 팀 구성원은 일반적으로 여러 가지 다른 역할을 수행합니다. 즉, 한 사람이 여러 '모자'를 쓸 수 있는 것처럼 여러 가지 다른 역할을 수행할 수 있습니다.

역할은 개인이 아니며, 비즈니스에서 개인이 어떻게 행동해야 하는지와 개인이 수행해야 할 책임을 설명합니다.

대부분의 역할은 조직 내부 인력으로 실현되지만 개발 조직의 외부 인력도 중요한 역할(예: 개발 중인 프로젝트 또는 제품의 이해 당사자(stakeholder)의 역할)을 수행합니다.

각 역할 유형은 팀 구성원이 제공해야 하는 스킬과 지식으로 정의됩니다. 조직은 이러한 정의를 기반으로 이 역할을 효율적으로 수행하는 데 필요한 훈련의 특성과 부족한 스킬을 도출할 수 있습니다.

RUP 가이드라인은 제공될 올바른 규칙 세트를 훈련으로 구성합니다. 과정 개발자는 나열된 역할 카테고리 중 하나입니다.

RUP가 전혀 적용되지 않고 CMM에서 필요한 유일한 측면은 향후 새 스킬을 개발하기 위한 훈련을 식별, 계획 및 전달하는 것입니다. 조직이 앞으로 1년 동안 전략적으로 e-commerce에 조직 활동을 집중하려는 경우를 가정할 수 있습니다. 이 목표는 보다 세부적으로 분해되어야 하며 일반적으로 사용되는 환경 유형 및 수행되는 개발 유형에 영향을 주게 됩니다. 팀에서도 이 변경을 준비해야 하며 필요한 훈련을 정의 또는 개발하거나 외부에서 훈련을 받아야 합니다. 레벨 3의 훈련 KPA(Key Process Area)에서는 이러한 문제를 주의 깊게 관리, 추적 및 기록해야 합니다.

- 이 문서의 섹션 3 에서는 RUP 기능을 포함한 특정 성숙도 프로파일에 대한 세부 맵핑을 제공합니다. 이 문서에서 CMM 참조의 각 목적과 RUP 의 해당 가이드라인을 조직적으로 비교했을 수 있지만 이는 의도한 바가 아니며 높은 성숙도를 나타내는 핵심 지표에만 초점을 맞추었습니다.

- RUP 에서는 테스트 활동 또는 품질 보증 계획 개발 타스크를 통해 소프트웨어 품질을 관리합니다. 그러나 이 문서에서는 높은 성숙도의 실제 의미에 대한 경험과 관찰 내용(레벨 1 또는 하위 레벨 2 조직과 비교하여 레벨 3 조직의 차이점)을 강조합니다.

- 다음 섹션은 이러한 결합 기술을 사용하는 얼리어댑터(early adopter)의 피드백에 근거하여 작성되었습니다.

RUP 시작 방법

가장 일반적인 질문은 이전에 이 일을 시도한 사람이 있는지에 대한 것입니다.

이 질문은 이전 개념을 관리자와 종사자(practitioner)에게 제시할 때 여러 번 나올 수 있는 질문입니다. 보다 높은 성숙도를 나타내고 보다 프로세스 지향적이며 위험성을 관리하고 성공하는 것은 어떤 소프트웨어 비즈니스에 있어서나 매력적인 일입니다. 그러나 변화하는 환경과 빠르게 변경되는 요구사항에서 또한 훈련받지 않은 기술을 적용해야 하는 팀에 있어 실제로 가능한지 생각해볼 필요가 있습니다.

Rational 고객 스토리에는 방위 전자 및 정보 분야의 선두 기업인 Computing Devices International 의 경험이 포함되어 있습니다. 정보는 거의 모든 산업에서 핵심 경쟁력으로 인정받고 있습니다. 그러나 Computing Devices International 의 경우, 단순한 정보가 아닌 요청에 따라 언제 어디서나 업무에 중요한 정보를 제공하는 완벽한 솔루션을 제공해야 하는 미션을 갖고 있습니다.

즉, 고품질 솔루션을 일정에 따라 지속적으로 제공할 수 있어야 하며 해당 경험의 핵심 이점은 다음과 같습니다.

- 반복 디자인 접근 방식을 통한 개발 프로세스의 가속화
- 개발 시간 단축(3 년 → 19 개월)
- 납기 준수에 따른 신뢰성 확보
- 고객 만족도 증가
- 개발 비용 감소(33% 절감)

Computing Devices 가 해당 고객을 위해 개발하는 정교한 시스템은 엄격한 요구사항에 따라 디자인되어야 하는 소프트웨어에 대한 종속도가 높습니다. Computing Devices 는 먼저 안전한 소프트웨어를 개발하기 위한 프로세스에 많은 주의를 기울였습니다.

조직 내 여러 비즈니스 단위에서 구조화된 디자인 또는 "폭포수형" 방법을 기반으로 한 다양한 도구를 사용하고 있었습니다. 결과적으로 개발 프로세스가 지체되고 일관적인 접근 방식도 부족한 상태였으며 이로 인해 납기가 지연되고 많은 비용이 소요되며 소프트웨어 품질이 경우에 따라 엄격한 표준을 충족시키지 못하고 있었습니다. Computing Devices 는 신속한 조치가 필요하다는 사실을 깨달았습니다.

Skandia-IT 는 Rational 포트폴리오의 또 다른 예입니다. 이 조직의 경우, 12 개월 동안 아홉 개의 대규모 보험 시스템이 납기 안에 전달되었습니다.

최신 개발 프로세스에 필요한 개발자와 컨설턴트를 모집하여 새로운 인재를 채용했으며, 요구사항, 모델 시스템 및 직접 개발에 필요한 유스 케이스를 채택하여 고객 만족도를 확보하고, 레거시 시스템을 컴포넌트로 사용함으로써 유연한 3 층 아키텍처를 빠르게 개발했습니다. 이는 Skandia-IT 에서 해결한 도전 과제 중 일부입니다.

다음은 Skandia-IT 관리자의 설명입니다. "전체 프로세스를 변경한 이유는 여러 가지이며 가장 중요한 점은 전체 비즈니스를 재구성했다는 것입니다. 즉, 제품 지향 조직에서 고객 지향 조직으로의 변경을 통해 개발 프로세스를 변경했습니다. 새 시스템에서는 고객이 전화 또는 월드 와이드 웹(WWW)을 통해 많은 일을 스스로 처리할 수 있습니다. 또한 Skandia 직원은 시스템 사용 방법의 학습이 아닌 보험 전문가가 되기 위해 더 많은 시간을 할애할 수 있으며 남은 시간은 고객 관계에 투자하고 있습니다."

Skandia-IT 는 새 프로세스 설치에 많은 비용을 투자했지만 그만한 가치가 있는 투자였습니다. 즉, 경험이 있는 프로젝트 리더와 개발자를 보다 쉽게 채용할 수 있게 된 간접적인 효과와 함께 품질 향상, 납기 예측이 가능하게 되었습니다.

프로세스에 초점을 맞추고 RUP 를 사용하여 배치 속도를 향상시키는 또 다른 회사 예도 있습니다. Q-Labs 는 현재 빠른 성장세를 보이고 있는 인터넷 방송용 소프트웨어 개발 회사인 Lysis 와 프로젝트를 수행하고 있습니다. 시작 환경에서는 높은 수준의 프로세스 성숙도에 초점을 맞추는 프로세스 기술 및 방법론이 효과를 나타내지 못할 수 있습니다. 이러한 유형의 조직은 일반적으로 속도가 빠르고 반응성이 뛰어나며 혁신적인 특성을 갖고 있습니다. 그러나 제품 출시 속도에 대한 요구와 프로세스 제어 극대화에 대한 요구 간의 대립으로 인해 프로세스 주제에서 이처럼 다양한 관점을 어느 정도 수용해야 하는지에 대한 문제가 대두됩니다.

이 회사는 다음과 같은 특성을 갖고 있습니다.

- 성장 속도가 빠른 소프트웨어 단위입니다.
- 개발 속도가 빠릅니다(6 개월 미만).
- 최신 오브젝트 기술을 사용하여 보다 나은 제품을 제공합니다.
- 역동적인 시장 환경에 적응합니다.

이러한 환경은 시간 스케일이 완전히 다르고 예측 가능성이 성공 또는 실패를 나타내거나 팀 작업이 빠르게 변경되어야 한다는 점을 제외하고는 다른 소프트웨어 산업과 크게 다르지 않습니다. 또한 학습 시간이 거의 없으므로 가장 올바른 선택은 입증된 기존 사례에서 시작하는 것입니다. Lysis 는 성장을 위해 CMM 을 채택하고 설치 프로세스의 배경으로서 RUP 를 채택할 때
모두 이 방법을 사용했습니다.

소프트웨어 사례 정의 요구에 대한 이의는 없었으며 이해, 커뮤니케이션, 실행 및 관리를 위해 필요한 작업입니다. 이러한 정의는 또한 다른 조직과의 내적, 외적인 관계 형성의 기반이 됩니다. Lysis 의 경우, 해당 목표는 마케팅 및 판매 부서와의 관계와 잠재적인 소프트웨어 하청업체와의 관계입니다. 정의된 프로세스는 유연성을 유지해야 합니다. 인력, 하부 구조, 릴리스 스케줄 및 채택된 기술이 빠르게 변경될 수 있기 때문입니다. 또한 개발 프로세스에서 이러한 변경사항을 수용해야 하며 신속하면서도 적절한 시점의 응답이 필요합니다. 반복 프로세스는 반복 주기가 짧고 위험성을 고려해야 하며 효과적인 관리가 수행되어야 합니다. 제품 라인 컴포넌트에 프로세스를 반복할 수 있는 기능도 필요합니다. 프로세스 정의는 회사가 성장하면서 새로 합류하는 인력에게 필요한 정보를 제공할 수 있습니다.

CMM 에서는 프로세스 정의(성숙도 레벨 3)를 프로세스 반복성(성숙도 레벨 2) 다음에 수행하지만, RUP 와 같은 표준 프로세스에서 시작하는 회사의 경우 반복성을 확립하는 데 도움이 되며 이로 인해 상당한 이점을 얻을 수 있습니다.

Lysis 에서 채택한 접근 방식은 CMM 요구사항에 대해 개발 사례를 평가함으로써 시작되었으며 그 결과 프로그래밍, 오브젝트 기술 및 도구 환경에서 가장 관심이 있는 팀이 프로세스 방침을 처리할 수 있게 되었습니다. 관찰 내용을 기반으로, 또한 엄격한 시간 제한조건에 따라 프로세스 정의를 위한 조치 계획이 확립되었습니다. 최고 경영진에서 SPI 를 지원하기 위해 설정한 비즈니스 목적은 다음과 같습니다.

- 목표 시장 최초 진입을 위해 개발 프로세스 단축
- 프로젝트 모드에서 제품 모드로의 전이(Transition) 속도 가속화
- 회사 성장 관리
- 구성원 만족
- 시작 대응성 유지
- 예측 가능성 및 가시성 개선(IPO 의 경우, 분기별 결과 필요)
- 제품 품질에 대한 정량적 개선

- 비용 및 잠재적 수익 관리
- 고객 만족 극대화
- 파트너와의 인터페이스 식별 및 관리(작업의 하청업체 파트 관련)

SPI 프로그램은 다음과 같은 RUP 프레임워크를 사용하여 프로젝트로서 확립되었습니다.

- 도입/인식(Inception) = 평가
- 정제(Elaboration) = 프로젝트 계획 + 웹 기반 저장소 정의 및 프로토타입 생성
- 구현/구축(Construction) = 프로세스 정의 + 구현 및 도구 사용
- 전이(Transition) = 파일럿 프로젝트 + 배치

CMM KPA(Key Process Area)를 처리하는 작업 패키지가 정의됩니다. 각 패키지별로 목표, 수행될 활동, 인도물 및 검토 프로세스가 있습니다. 예를 들어, 소프트웨어 계획 목표는 다음과 같습니다.

조치 목표:

- 소프트웨어 계획의 현재 올바른 사례 문서화
- 프로젝트에 대한 일반 조직 세트 정의
- 계획을 위한 역할 및 책임 정의
- 프로젝트 레벨에 대한 약속 프로세스 정의
- 일반 프로젝트 라이프사이클 정의 및 표준 프로젝트 프로세스 세트에 대한 가이드라인 사용자 조정
- 표준 작업분류체계(WBS) 정의
- 인도물 표준 목록 나열
- 개발 계획에 대한 템플리트 및 가이드라인 정의
- 프로젝트 계획 프로세스 작성
- 계획에 필요한 스킬 및 훈련 정의

RUP를 시작하는 경우 이러한 조치 중 일부의 속도가 가속화되며 조직은 유용한 경험을 활용할 수 있습니다. SPI 이니셔티브는 완료되지 않았지만 일부 가시적인 성공 요인이 이미 관찰되었습니다. 신제품 개발이 보다 구조화되며 계획이 존재하고 정기적으로 재검토됩니다. 정정 조치가 수행되며 소프트웨어 팀 외부의 가시성이 크게 향상됩니다.

Rational Software 는 이와 유사한 E-corporation 사례를 보고합니다. E-corporation 은 전세계 회사를 디지털 비즈니스의 최전선으로 안내하는 유럽의 대화식 설계 분야 기업입니다. E-corporation 은 전략적, 전술적, 소프트웨어 스킬을 결합하여 인터넷 및 e-business 벤처를 고안, 작성 및 구현합니다.

RUP 배치에 따른 E-corporation 의 핵심 이점은 다음과 같습니다.

- 팀 구성원 간의 효과적인 커뮤니케이션으로 효율성이 향상됩니다.
- 고객의 예상을 보다 정확하게 충족시킬 수 있습니다.
- 고품질 및 고속 요구를 충족시킬 수 있는 프레임워크가 제공됩니다.
- 반복적 개발에 대한 위험성이 감소하고 전문 지식 및 업계에서 인정한 우수 사례 기반의 입증된 프로세스를 사용합니다.

E-corporation 이 모든 프로젝트에 단일 공통 프로세스를 적용함으로써 얻은 가장 큰 이점은 컴포넌트, 경험 및 활동을 효과적으로 재사용할 수 있다는 것입니다. 조직이 다양한 프로젝트를 통해 경험을 축적하면서 Rational Unified Process 는 지속적으로 세분화되고 조정될 수 있습니다. 모든 활동은 일관된 구조와 언어에 따라 문서화되므로 이전 프로젝트의 많은 경험을 새 프로젝트로 이전할 수 있습니다.

E-corporation 은 모든 사이트의 모든 해당 프로젝트에 Rational Unified Process 를 전개함으로써 해당 팀을 모두 활용할 수 있습니다. Rational Unified Process 는 입증된 원칙을 적용함으로써 모든 노력을 최적화하고 커뮤니케이션을 단순화하며 모든 요구사항을 충족시키는 솔루션 품질을 보장할 수 있는 권한을 해당 개발 팀에 부여합니다.

이는 Rational Unified Process 와 CMM 가이드라인을 결합함으로써 다양한 조직에서 이점을 얻을 수 있는 방법을 나타내는 예입니다. 현재, 적합한 위치에 조직의 프로세스 자산과 함께 사용될 일반 구조로 RUP 를 고려하고 있는 다른 조직과의 프로젝트가 진행되고 있습니다.

결론

RUP 및 CMM 맵핑에 대해 설명한 내용은 ISO 15504 에도 적용할 수 있습니다. 이 두 참조의 기본 개념은 유사하며, 성숙도 프로파일을 설명하기 위해 강조한 핵심 요인은 ISO 15504 의 기능 스케일을 지원합니다.

이 문서에서는 RUP 개념이 성숙한 조직의 목적과 일치되는 정도를 명백하게 설명했습니다. 따라서 RUP 채택은 CMM 요구사항을 성공적으로 이행하기 위한 방법입니다.

Telcordia Technologies 에서 최근 발행한 문서에는 비즈니스의 높은 성숙도 결과가 정량적으로 설명되어 있습니다. 최근 성숙도 레벨 5 로 평가된 Telcordia 는 다음과 같은 내용을 보고했습니다.

- SPI 이니셔티브를 시작한 이래로 현장의 결함(fault) 비율이 94% 감소했습니다.
- 1995 년 이후 98% 이상의 소프트웨어 릴리스가 납기 내에 전달되었습니다.
- 코드 행 테스트 비용이 64% 감소했습니다.

Telcordia 에서 이 성숙도 레벨을 달성하는 데 소요된 시간은 6 년입니다.

e-business 및 e-commerce 분야의 소프트웨어 중심 회사의 경우, 프로세스를 최적화하기 위해 5, 6 년의 시간을 투자할 여력이 없습니다. 따라서 경쟁력을 유지하면서 필요한 성숙도에 도달할 수 있는 시나리오는 다음과 같습니다.

- 소프트웨어 프로세스의 가치 이해
- RUP 와 같은 기존 프레임워크 채택
- 해당 프레임워크를 시장 환경의 특성에 맞게 수정
- 비즈니스에 대한 영향 측정
- 사용자 정의된 프로세스 모델에 대한 학습과 개선

RUP 모델은 일반 모델이므로 현재 상황을 정확히 파악하는 것이 중요합니다. 따라서 Rational 은 모델 채택 시 중점을 둘 측면을 포함하는 "RUP 필수 요소"를 식별합니다[11, 12]. 기본 RUP 를 사용할 수 있지만 처음에는 모든 요소를 구현하지 않아도 됩니다. 중요한 기능은 다음과 같습니다.

- 비즈니스 사례
- 스케줄
- 비전 문서
- 위험성
- 아키텍처
- 변경 요청 및 결함 처리 방법
- 테스트
- 소프트웨어 제품
- 사용자 지원 문서
- 프로젝트 평가

RUP 는 실행이 그 목적이 아니며 그룹 및 팀의 문제를 해결하고 경쟁력 목표를 달성하는 데 도움을 줍니다.

저자 연락처 정보

Annie Kuntzmann-Combelles, Executive VP
Q-Labs France
28 Villa Baudran 94742 Arcueil cedex
전화 +33 (0)1 49 08 58 00
akc@objectif.fr

Philippe Kruchten, Rational Fellow
Rational Software Canada
pbk@rational.com

참조

[1] K. Pulford, A. Kuntzmann-Combelles, and S. Shirlaw, 1995. *A Quantitative Approach to Software Management—The AMI Handbook*. Addison Wesley Longman.

[2] Barry W. Boehm, 1996, "Anchoring the Software Process," *IEEE Software*, July 1996, pp.73-82.

[3] Philippe Kruchten, 1996. "A Rational Development Process," *CrossTalk*, 9 (7), July 1996, p.11-16.

[4] Robert McFeeley, 1996. *IDEAL: A User's Guide for Software Process Improvement*. Software Engineering Institute, Pittsburgh, PA, CMU/SEI-96-HB-001.

[5] Steve McConnell, 1997. *Software Project Survival Guide*. Redmond, WA: Microsoft Press.

[6] Mark Paulk 외 1993. *Capability Maturity Model for Software, Version 1.1*. Software Engineering Institute, Pittsburgh, PA SEI-93-TR-024.

[7] Walker Royce, 1998. *Software Project Management: A Unified Framework*. Addison Wesley Longman.

[8] Jas Madhur et al, 1998. *Reaching CMM Levels 2 and 3 with the Rational Unified Process* white paper, Rational Software.

[9] Alec Dorling 외, 1999. *SPICE, the Theory & Practice of Software Process Improvement*, IEEE Computer Society.

[10] Philippe Kruchten, 2000. *The Rational Unified Process—An Introduction, 2nd ed.* , Addison Wesley Longman.

[11] *Rational Unified Process, version 2000.02.10,* Rational Software Corporation

[12] Leslee Probasco, 2000. "The Ten Essentials of RUP", *The Rational Edge*, December 2000, http://www.therationaledge.com.

Rational®
the software development company

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

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