소프트웨어 프로세스 개선을 위한 점증적인 프로세스 적용 전략

이선아[0] 최순규 최정은
삼성전자 CTO전략실 소프트웨어센타
(salee, soonkew, jeunchoi)@samsung.com

Incremental Process Deployment Strategy for Software Process Improvement

Seon-ah Lee[0] Soon-kew Choi Jeong-eun Choi
Software Center, Corporate Technology Operations, Samsung Electronics Co., Ltd

요 약

프로세스 개선은 프로젝트의 지속적인 성공과 밀접히 연관된다. 따라서 많은 기업들이 프로세스 개선을 위해 노력하고 있으나 프로세스 개선에 대한 투자 비용, 내부적인 저항, 내부적인 참여 등의 문제로 프로세스 개선의 실질적인 효과를 보지 못하는 경우가 적지 않다. 본 논문에서는 조직내의 충돌을 최소화하면서 프로세스 적용을 가속화하기 위한 방안으로 점증적인 프로세스 적용 전략을 제안한다. 점증적인 프로세스 적용 전략은 점증적인 개발 프로세스 채택, 점증적인 프로세스 적용, 점증적인 프로세스 정립을 병행하는 방안이다.

1. 서 론

프로세스의 개선이 프로젝트의 일정, 예산 및 제품 품질에서의 성공과 밀접히 연관됨은 잘 알려진 사실이다. 회사의 경쟁력이 점점 더 Time To Market에 의존하고 있는 현 상황에서 짧은 시간 내에 고품질의 제품을 개발하는데 기반이 되는 프로세스 개선은 필수적이다.

앞선 기업들은 수십 년의 프로세스 개선 활동을 통하여 이미 CMM Level 5의 프로세스를 갖추고 있다[1]. 이러한 상황에서 기존의 전통적인 개발 방식을 고수하고 있는 기업들은 빠른 프로세스 개선을 이루지 않으면 경쟁에서 밀려날 수 밖에 없다.

하지만 프로세스 개선은 조직의 문화를 바꾸는 작업이기 때문에 프로세스를 개선하는 것은 오랜 시간이 걸린다. 전통적인 개발 조직일 경우 이러한 문화 변경은 더욱 어려울 수 있다. 또한 전통적인 개발 조직이 프로세스 개선의 성공 요건을 갖추지 못한 경우 프로세스 개선은 더욱 늦어질 수 밖에 없다.

본 논문은 전통적인 조직에서 진행하는 ad-hoc 개발 프로세스[2,3]와 incremental & iterative 개발 프로세스[3,4]가 비슷한 점에 착안하여 점증적인 프로세스 적용 전략을 제안한다. 점증적인 프로세스 적용 전략은 프로세스 개선의 장애물인 조직내의 반발을 최소화하면서 프로세스 개선을 가속화할 수 있는 방안이다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 연구를 살펴보고 본 연구가 필요한 이유를 설명한다. 3장에서는 전통적인 조직의 프로세스 특성을 분석한다. 4장에서는 점증적인 프로세스 적용 전략을 제시하고 5장에서는 그에 따른 사례를 보여준다. 6장에서는 결론 및 향후 연구 과제를 제시한다.

2. 기존 연구

기존의 프로세스 개선 연구는 프로세스 개선 체제를 중심으로 이루어졌다. 그 결과 CMM[5]과 SPICE[6] 등의 프로세스 성숙도 모형이 제시되어 프로세스 개선의 기본틀로 사용되고 있다.

성숙도 모형은 하향식 접근 방법[7]으로 프로세스 개선을 수행한다. 즉 현재 조직의 프로세스 상태를 성숙도 모형의 프로세스와 비교하여 약점을 발견하고 이를 기반으로 도달해야 할 목표를 설정, 개선하는 방법이다. 하향식 접근 방법은 성과가 크다는 장점이 있지만 그 만큼 사전에 투자가 커야 한다. 따라서 경영층의 적극적인 참여 등의 프로세스 성공 요건[8]을 갖추지 않으면 성공하기 어렵다.

반면 프로세스 개선은 제품 개선을 위한 것이며 모든 소프트웨어는 다르기 때문에 프로세스도 달라야 한다는 원칙을 바탕으로 한 상향식 접근 방법[7]이 있다. NASA의 SEL[9]이 대표적인 예이다. 상향식 접근 방법은 적용 대상의 프로세스, 제품 및 조직 특성에서 중요한 개선 요소를 명시하고 이를 개선한다. 상향식 접근 방법은 개선 요소에 대한 통찰이 정확하고 측정할 수 있어야 프로세스 개선 프로그램이 성공할 수 있다.

한편 조직의 변화를 꾀할 때 사용할 수 있는 모형도 제시되었다. PDCA[8,10], IDEAL[11] 등은 조직에서 프로세스 개선을 일으키기 위한 일반 모형이다. Demming이 설명한 PDCA(Plan-Do-Check-Act) 모형[8,10]은 프로세스 개선을 계획하고 시행한 후 검토하여 일반화하는 것을 골격으로 한다. McFeely가 제시한 IDEAL(Initiating-Diagnosing-Establishing-Acting-Leveraging)모형[11]은 현재의 프로세스를 분석하여 지향해야 할 프로세스를 계획하고 프로세스를 변경하여 새로운 프로세스를 정착하도록 한다.

이처럼 프로세스 개선 체제가 제시되었음에도 불구하고 프로세스 개선은 여전히 어려운 문제로 남아 있다[12,13,14,15,16]. 이에 따라 프로세스 개선 시도가 왜 실패하는지에 대해서 많은 연구가 진행되고 있으며 이러한 연구들은 프로세스 개선 사례를 분석하여 프로세스 개선의 성공 요소와 실패 요소를 밝혀내고 있다. 하지만 성공 요소가 갖추어지지 상황에서 어떻게 개선된 프로세스를 적용할 것인가에 대한 지침에 대한 연구는 일반적인 수준으로 아직 구체적인 전략을 제공하지 못하고 있다.

3. 전통적인 개발 조직의 프로세스 특성

전통적인 개발 조직에서 프로세스 개선을 수행할 경우 많은 장애물을 만나게 된다. Wiegers[14]는 장애물로 1)불충분한 시간 2)지식 결여 3)잘못된 동기 4)독단적인 접근 5)불충분한 참여를 꼽았다. Myers[15]는 1)변화에 대한 저항 2)지식 결여 3)효과 미비 4)시도 실패 5)투자 미비를 말하였다. O' Day[16]는 1)저항 2)이전으로의 복귀 용이성 3)연속성의 결여 등을 이야기 하였다. 이러한 장애물은 잘못된 프로세스 개선 시도를 제외하고 나면 투자의 문제, 저항의 문제, 참여의 문제로 정리될 수 있다.

본 논문은 위의 문제는 전통적인 개발 조직에서 이제까지 갖추어온 프로젝트 성공 요인을 저해하지 않으면서 실패 요인을 성공 요인으로 바꾸어 나갈 때에 해결될 수 있다고 보았다. 이러한 해결 방안을 위하여 전통적인 개발 조직의 프로세스 특성을 살펴보았다. 해당 특성은 표 1과 같이 요약할 수 있다.

표 1. 전통적인 개발 조직의 프로세스 특성

성공에 긍정적인 요소	성공에 부정적인 요소
■ 기술적인 리더	■ 프로젝트 관리 미약
■ 점진적인 개발	■ 계속되는 재작업
■ 빠른 릴리즈	■ 문서 미비
■ 도제 제도	■ 코딩 위주

전통적인 개발 조직에서의 프로세스는 기술 중심의 개발에 릴리즈 시점이 빠른 점진적 프로세스을 채택하여 소수 인원의 빠른 개발 진행에 경쟁력을 보인다. 그러나 요구사항 변경 관리의 어려움, 아키텍처 유지의 어려움, 지속적인 유지 보수의 어려움에 직면한다.

4. 점증적인 소프트웨어 프로세스 적용 전략

본 장에서는 조직내의 충돌을 최소화하면서 프로세스 적용을 가속화하기 위한 방안으로 점증적인 프로세스 적용 전략을 제안한다. 점증적인 프로세스 적용 전략은 점증적인 개발 프로세스 채택, 점증적인 프로세스 적용, 점증적인 프로세스 정립을 병행하는 것이다.

4.1 프로세스 적용 원칙

프로세스 적용 시에 반발을 최소화하기 위하여 다음과 같은 적용 원칙을 수립하였다.

1. 간단한 절차 및 산출물로부터 시작한다
2. 적용 계획을 수립하여 적용팀의 합의를 받는다
3. 점증적인 프로세스 적용 방법을 사용한다.
4. ROI(Return On Investment)를 빨리 보여준다.

4.2 점증적인 개발 프로세스 채택

점증적인 개발 프로세스는 점진적이고 릴리즈(release)가 빠르다는 점에서 전통적인 개발 조직의 프로세스와 비슷하다. 또한 개발 기간동안 코딩 작업이 지속되므로 개발자가 익숙한 개발 습관을 갑자기 바꾸지는 않는다. 그러나 1)프로젝트 관리 2)문서화 3)개발단계별 비중이 다르다. 이러한 개발 습관과 다른 부분을 각 증가분 별로 나누어 개선하여 나감으로써 ad-hoc 프로세스를 잘 정립된 iterative & incremental 프로세스로 바꾸어나갈 수 있다.

4.3 점증적인 프로세스 적용

전통적인 개발 조직에서는 프로젝트 관리, 요구 사항 관리, 구조 설계 등의 문제가 서로 연관되어 있으므로 하나의 프로세스 분야를 성공적으로 개선했다고 해도 그 효과가 잘 보이지 않는다. 반면 많은 프로세스 기법을 한 번에 적용하려고 할 경우 개발자의 습득 시간으로 인한 개발 프로젝트의 지연으로 반발이 심해질 수 있다. 따라서 현재의 문제점을 해결할 수 있을 만큼의 프로세스 분야를 각 증가분 별로 적절하게 할당하여 적용하도록 한다. 이와 함께 각 증가분에서의 프로세스 개선 노력이 실질적으로 어떤 이득을 가져왔는지도 각 증가분이 끝날 때마다 보여주도록 한다.

4.4 점증적인 프로세스 정립

프로세스의 결함을 반영하여 표준 프로세스가 수정 보완되어야 한다. 이와 동시에 수정된 프로세스를 다시 적용 프로젝트에 적용한다면 개발자의 프로세스 결함에 대한 반발을 줄일 수 있다. 점증적인 개발 프로세스에 따라 n차분에서 정립한 프로세스를 n+1차분에서 적용하도록 한다. 이와 함께 SPI (Software Porcess Improvement) 조직은 프로젝트 종료 시 해당 부분을 조직 내 전파함으로써 조직의 프로세스가 지속적으로 개선되도록 한다.

5. 전통적인 개발 조직에서 점증적인 프로세스 적용 전략

본 장은 전통적인 개발 조직에서 점증적인 프로세스 적용 전략이 어떻게 구현되어 문제점을 해결해 나가는지를 보여준다.

프로세스 적용 프로젝트를 시작할 때 전통적인 개발 조직의 상황을 살펴보면, 조직 내에 이미 SPI 조직이 있지만 프로세스 개선에 대한 조직 내의 이해가 부족하여 적용팀의 참여 및 경영층의 지원이 부족한 상태였다. 프로세스 개선 성공 요인[13]을 살펴본 결과 표 2와 같다. 프로세스 적용 시의 구체적인 장애물은 표 3과 같다.

표 2. 프로세스 개선 성공 요인의 상태

프로세스 개선의 성공요소	-	-	0	+	+
1. 경영층의 참여 및 지원			■		
2. 적용팀의 참여		■	■		
3. 프로세스 개선 목적에 대한 이해		■	■		
4. 프로젝트에 맞는 개선 목표 설정			■		
5. 개선 프로젝트의 관리			■	■	
6. 변화를 촉진하고 도와주는 인력			■	■	■
7. 개선된 프로세스를 고정화			■	■	
8. 의사소통 및 협업 촉진			■		
9. 적절하고 현실적인 목적 설정			■		
10.프로세스 변화를 수용하는 조직		■	■		

표 2. 프로세스 적용 시의 장애물

문제점	예
잘못된 동기	경영층에서 CMM Level2의 실제 사례를 요청함
투자의 문제	초기 투자의 문제로 프로세스 심사가 도입되지 않음
저항의 문제	개발팀은 적용하려는 프로세스가 현재 개발 프로젝트에 적합하지 않음을 지적하고 도입 거부함
참여의 문제	프로젝트 진행에 있어 가장 중요한 것은 프로젝트의 성공이며 해당 책임은 개발팀에서 있음

SPI 조직은 프로세스 성숙도 모형 및 프로젝트의 목적, 개발팀의 현재 수준 및 개발팀의 설문조사의 결과를 반영하여 프로세스 개선 항목 및 우선 순위를 설정하였다. 1차는 표준문서양식 및 도구를 도입하였다. 2차는 소프트웨어 아키텍처를 중심으로 프로세스를 적용하였다. 3차는 품질 보증 및 형상관리를 개선항목으로 잡았다. 이러한 활동이 전통적인 개발 조직에서의 약점을 어떻게 극복하는지 전략을 정리하면 다음과 같다.

표 4. 전통적인 개발 조직의 개선 전략

개발조직의 약점	개선 전략
프로젝트 관리 미약	개발 계획서 수립 및 각 차수에서의 문서 보완
계속되는 재작업	문서, 요구관리, 형상관리 및 설계기법 적용
문서 미비	표준 문서양식 적용
코딩 위주	점증적인 개발로 코딩과 문서를 병행

SPI 조직이 프로세스를 적용하는 프로젝트에서 수행한 활동은 그림1과 같다. SPI조직은 각 적용 활동마다 전문담당자를 두어 적극적인 프로젝트 활동을 수행하였다. 예를 들어 표준문서양식 담당자는 양식을 제공하는 외에 문서를 표준 양식에 맞게 수정하고 내용이 각 항목별로 적절하게 기입되어 있는지를 검토하였다. 도구담당자는 해당 도구의 사용법 및 연동된 기법을 교육하고 활동이 제대로 이루어지고 있는지를 모니터링 하였다. 또한 각 차수별 프로세스 적용 전에는 프로세스 적용 방안을 설명하고 개발팀과 적용 항목에 대해서 합의를 하였다. 각 프로세스 적용 후에는 오류원인 분석을 통하여 적용 효과를 분석하였다

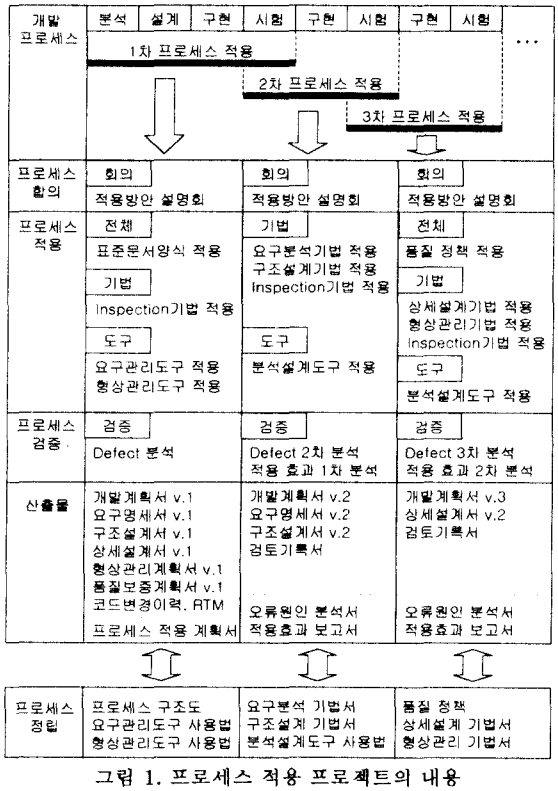

그림 1. 프로세스 적용 프로젝트의 내용

그 결과 개발자의 반발이 줄어들고 개선된 프로세스를 도입하는데 대한 인식이 긍정적으로 변화하였다. 이러한 결과는 프로세스 개선에 따른 반발 및 참여의 문제에 대한 해결책을 보여준다.

6. 결론 및 향후 연구 과제

점증적인 소프트웨어 프로세스 적용 전략은 개선 효과를 빨리 보여주어 개발팀의 적극적인 참여를 유도한다. 또한 한 프로젝트에서 여러 기법을 순차적으로 도입하여 프로젝트의 지연을 염려하는 개발자들의 반발을 해결한다. 점증적인 소프트웨어 프로세스 적용 전략은 한 프로젝트에서 여러 기법의 성공사례를 만들어 냄으로써 프로세스 개선을 가속화할 수 있다. 해당 전략은 최신의 소프트웨어 공학 기법들을 어떻게 실제 프로젝트에 적용해 나갈 것인가에 대한 해결책도 되리라고 생각한다.

해당 전략을 좀더 효과적으로 적용하기 위해서는 적용할 프로세스에 포함될 기법, 활동, 도구와 개선 효과와의 연관성이 적용 전에 제시되어 있어야 한다. 또한 개발 프로젝트의 특성에 따른 도입 기법의 적절성에 대해서도 계속 연구되어야 한다.

참 고 문 헌

[1] Rico, D. F., "Software Process Improvement(Impacting the Bottom Line by using Powerful "Solutions")", http://davidfrico.com/spipaper.pdf, Dec. 1998.

[2] Raymond, E. S., *The Cathedral and the Bazaar*, http://tuxedo.org/~esr/writings/cathedral-bazaar/, Nov. 1998.

[3] Sotirovski, D., "Heuristics for Iterative Software Development", *IEEE Software*, pp.66-73, May/June 2001.

[4] Jacobson, I., Booch, G., "The Unified Process", *IEEE Software*, pp.96-102, May/June 1999.

[5] Paulk, M. C. et al, *The Capability Maturity Model: Guidelines for Improving the Software Process*, Addison-Wesley Pub Co., 1995.

[6] ISO/IEC 15504 TR2, *Software Process Assessment and Capability determination*, ISO/IEC, 1998.

[7] Thomas, M., McGarry F., "Top-down vs. Bottom-up process improvement", *IEEE Software*, pp.12-13, Jul. 1994.

[8] Stelzer, D., and Mellis, W., "Success Factors of Organizational Change in Software Process Improvement", *Software Process Improvement and Practice*, pp.227-250, 1998.

[9] Basili, V. et al, "SEL's Software Process improvement program", *IEEE Software*, pp.83-87, Nov. 1995.

[10] Grady, R. B., *Successful Software Process Improvement*, Hewlett-Packard Company, 1997.

[11] McFeeley, B., *IDEAL℠:A User;s Guide for Software Process Improvement*, CMU/SEI-96-HB-001, Feb. 1996

[12] Rifkin, S., "Why Software Process Innovations Are Not Adopted", *IEEE Software*, Jul./Aug. 2001.

[13] Bach, J., "Enough about process: What we need are heroes", *IEEE Software*, Mar. 1995.

[14] Wiegers, K. E.,"Why is Software Improvement So Hard?", *Software Development* , Feb. 1999.

[15] Myers, W., "Why software developers refuse to improve", *Computer*, pp.110-112, Apr. 1998.

[16] O'Day, D., "This old house [software development]", *IEEE Software*, pp. 72-75, Mar./Apr. 1998.

[17] Pfleeger, S. L., "Realities and Rewards of Software Process Improvement", *IEEE Software*, pp.99-101, Nov. 1996

[18] Jakobsen, A. B., "Bottom-up Process Improvement Tricks", *IEEE Software*, Jan./Feb. 1998.

[19] Appleton, B., "Patterns for Conducting Process Improvement", *Proceedings of the 4th Annual Conference on PLoP*, 1997.