소규모 프로젝트를 위한 애자일 프레임워크 설계 및 평가

(Design and Evaluation of Agile Framework for Small Projects)

이 세 영 † 용 환 승 ††

(Seiyoung Lee) (Hwan-Seung Yong)

요 약 본 논문에서는 애자일 방법론을 기반으로 한 소프트웨어 개발 프레임워크(AFSP)를 설계하였다. AFSP는 확장된 스크럼 프로세스와 소규모 프로젝트에 최적화된 애자일 프랙티스로 구성된다. AFSP 프랙티스는 스크럼, XP, FDD, DSDM, 크리스탈 클리어로부터 민첩도가 높은 프랙티스를 접목함으로써 소규모 프로젝트 개발 및 관리에 보다 최적화될 수 있도록 하였으며, 소프트웨어 개발 생명 주기에 따라 6대 애자일 프로젝트 성공요소를 반영하여 효과적인 적용이 가능하도록 했다. 또한, AFSP를 소규모 웹 어플리케이션 프로젝트에 적용하고 종합적인 평가를 수행함으로써 그 효율성을 입증하였다.

키워드 : 애자일 방법론, 스크럼, 소규모 프로젝트, 애자일 프레임워크

Abstract In this paper, the Agile Framework for Small Projects (AFSP) was applied to four industry cases. The AFSP provides a structured way for software organizations to adopt agile practices and evaluate the results. The framework includes an extended scrum process and agile practices, based on agility and critical success factors in agile software projects that are selected from Scrum, XP, FDD, DSDM and Crystal Clear. The case projects were evaluated, and the analysis of the results clearly showed that the framework used in the aforementioned cases displayed a high degree of efficiency.

Key words : Agile methods, Scrum, Small projects, Agile framework

1. 서 론

IT산업이 발달한 대부분의 국가에서 직원 수 50명 이하의 소규모 기업(small enterprises)과 25명 이하의 극소규모 기업(VSEs: very small enterprises)은 전체 소프트웨어 회사의 85%에 달한다[1,2]. 이러한 소규모 회사들은 일반적으로 환경변화에 유연하고 대응이 빠르며 직간접적으로 많은 소규모 프로젝트를 수행한다. 소규모 프로젝트는 6개월 이하의 개발 기간, 10명 이내의 개발 인력, 파트타임 작업 수반, 소수의 기술 영역 투입, 단일 목표와 해결책, 제한된 개발 범위, 단순한 제품 출시, 여러 기술 영역과의 상호 의존 성향 등의 특성을 가진다. 본 논문에서는 소규모 프로젝트에서 발생하기 쉬운 다섯 가지 문제점[3]-1) 불충분한 계획, 2) 낮은 우선순위, 3) 경험이 부족한 프로젝트 팀, 4) 동시에 여러 가지 역할을 담당하는 프로젝트 매니저, 5) 소규모 프로젝트의 특성을 고려하지 않은 표준 방법론과 도구의 사용--에 대한 대안으로써 애자일 방법론을 기반으로 하는 소규모 프로젝트용 애자일 프레임워크(Agile Framework for Small Projects: AFSP)를 제안하였다.

애자일 방법론은 짧은 주기의 반복적이고 지속적인 제품 출시를 통해 시장의 반응을 빠르게 적용하며, 전문성이 높은 소규모의 팀을 구성하여 효과적인 대화와 협력으로 생산성을 극대화한다. 즉, 변화무쌍한 시장과 고객의 요구 변화에 유연하게 대응함으로써 고객의 만족과 비즈니스의 성공을 이끌어내는 데에 그 핵심이 있다[4]. 포레스트 리서치의 2008년 2월 보고서[5]에 따르면, 북미와 유럽 지역 기업의 25% 이상이 공식적으로 애자일 방법론을 도입했으며, 나머지 기업 가운데 50%가 이미 도입 계획을 가지고 있거나 검토 중에 있다. 이처럼 애자일 방법론은 빠른 속도로 전 세계에 확산되고 있다[6,7].

본 연구에서는 소규모 프로젝트가 직면하고 있는 현실적인 문제점을 극복하기 위한 기존의 소프트웨어 공학적 접근방법을 분석, 검토하였으며 주로 애자일 방법론을 연구하였다. 대표적인 애자일 방법론인 스크럼(Scrum), XP(Extreme Programming), FDD(Feature-Driven Development), DSDM(Dynamic Systems Development Method), 크리스탈 클리어(Crystal Clear) 등에 대한 기존 연구를 분석한 결과, 스크럼과 XP의 프로세스와 프랙티스를 현업의 요구에 따라 다양한 조합으로 최적화하여 사용하는 것이 최근의 적용 사례임을 발견[8-11]하였다. 이에 본 연구에서는 스크럼과 XP의

· 이 논문은 2009 한국컴퓨터종합학술대회에서 '소규모 프로젝트를 위한 애자일 프레임워크 설계 및 평가'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 이화여자대학교 컴퓨터공학과
sarahlee230@yahoo.com
†† 종신회원 : 이화여자대학교 컴퓨터공학과 교수
hsyong@ewha.ac.kr

논문접수 : 2009년 8월 13일
심사완료 : 2009년 10월 1일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제15권 제11호(2009.11)

결합 모델을 기반으로 하고 그 밖의 유용한 애자일 프랙티스를 접목시켜 소규모 프로젝트에 최적화된 애자일 프레임워크(AFSP)를 설계하였다. 또한, 미국 야후사의 소규모 웹 어플리케이션 프로젝트 4건에 AFSP를 실제 적용하고 다양한 관점에서 종합적으로 평가함으로써 그 효율성을 입증하였다.

2. 관련 연구

시간과 비용을 많이 들여 대규모 프로젝트에 적합하도록 설계된 소프트웨어 프로세스와 프랙티스를 소규모 프로젝트에 그대로 적용하기는 어렵다[3]. 본 논문에서는 소규모 프로젝트 개발 및 관리 방안과 소규모 소프트웨어 업체에 적합한 소프트웨어 프로세스 개선(SPI)에 관한 연구를 바탕으로 소규모 프로젝트 관리를 위한 관점을 다음과 같은 세 가지로 분류하였다.

첫째, 기존의 표준 방법론을 소규모 프로젝트 혹은 소규모 기업에 적합하도록 간소화한다. 예를 들어, Rowe [3]는 PMBOK(Project Body of Knowledge)의 주요 프로세스와 문서화 기법을 사용하여 축약된 버전의 도구와 프로세스를 제안하였으며, Trengove와 Dwolatzky [13]는 ISO 9001과 Unified Process를 바탕으로 역시 축소된 버전의 소프트웨어 공학 프로세스를 개발하였다. 소프트웨어 프로세스 평가 및 개선 관점에서 보면, 최소한의 평가 비용과 시간을 들여 CMMI 프레임워크를 사용하기 위해 일부 소규모 업체들은 프로세스 개선을 위한 표준 CMMI 견적 방법(SCAMPI)인 클래스 A 대신 클래스 B나 C 견적을 사용하여 그 과정을 간소화 하고 있다[1]. 또한 Harba 등[14]은 극소규모 기업 내에서 SPI를 처음 적용하기 위한 방안으로써 OWPL 점진 SPI 방식을 제안했다. 일련의 마이크로 평가, OWPL 평가와 SPICE, CMMI 또는 CMMI 평가로 이루어진 이 스키마는 조직과 프로젝트의 상황에 맞추어 순차적으로 선택하여 적용될 수 있도록 설계되었다.

둘째, 독자적인 애자일 접근 방식을 사용한다. 예를 들어, Wang[15]은 전통적인 방법론이 대규모, 혹은 분산 소프트웨어 개발에 적절한 반면, 애자일 방법론은 중소 규모의 소프트웨어 개발에 적합하다고 단언했다. Boehm[16] 역시 애자일 방법론이 요구사항이 명확치 않거나 자주 변화하는 소규모 프로젝트에 유리한 반면, 전통적인 계획 중심의 개발 방법론은 고도의 확실성이 요구되는 복잡한 소프트웨어 프로젝트에 필요하다고 주장했다. Rising과 Janoff[12]는 요구사항의 변화가 잦은 소규모 팀 중심의 프로젝트를 위한 해법으로 스크럼을 제안하며 세 가지 유형의 적용 사례를 소개하였다.

셋째, 기존의 표준 방법론에 애자일 기법을 접목한다. 예를 들어 Boehm과 Turner[17]는 기존의 전통적인 방법론과 애자일 방법론은 각기 다른 홈 그라운드가 있으므로 프로젝트 상황에 맞게 위험 수준을 분석하고 그 결과에 따라 이 두 가지 방법론을 적절하게 조합하여 이용하는 전략이 요구된다고 주장했다. Cohen[18] 역시 이 두 가지 방법론은 프로젝트 참여 인원, 응용 분야, 중요성, 혁신성 등의 프로젝트 환경에 따라 공생하는 관계에 있다고 주장했다. 또한, Caffery[2] 등은 소규모 소프트웨어 업체를 위한 SPI 모델로써 Adept를 개발 및 제안하였다. 이는 CMMI 클래스 C 평가 지침을 포함한 계획 중심 기법과 변형된 애자일 기법을 하나로 통합한 경량의 평가 프로세스이다.

3. 소규모 프로젝트용 애자일 프레임워크(AFSP) 설계

3.1 AFSP 개요

본 논문에서 제안한 AFSP는 애자일 소프트웨어 개발 프로세스와 프랙티스 풀로 구성되며, 이를 소프트웨어 프로젝트에 적용하기 위해 그림 1과 같은 5가지 단계를 거친다. 이 가운데 실선으로 표시된 3, 4단계가 핵심이다. 즉, AFSP 프랙티스 풀로부터 각 개발 환경에 대한 분석을 기반으로 결정된 애자일 프랙티스가 내포된 AFSP 프로세스에 근거하여 프로젝트를 수행한다.

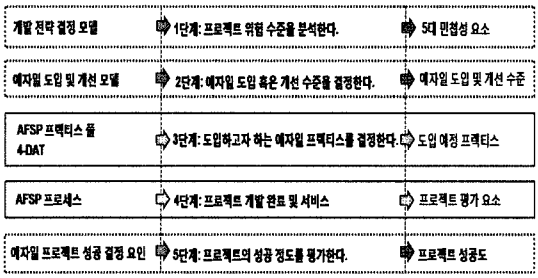

그림 1 AFSP의 적용을 위한 5단계 (점선은 생략 가능, 실선은 필수 단계)

3.2 AFSP 프로세스

AFSP 프로세스는 스크럼을 기반으로 전통적인 계획 중심 방법론의 강점인 프로젝트 준비 단계를 "스프린트(sprint) 0"이라는 이름 아래 선택적으로 수용하였다(그림 2). 스크럼 프로세스는 2-4주마다 반복되는 일련의 "스프린트" 단위로 그 산출물이 점증적으로 출시된다. 매 스프린트 시작 전 전체 프로젝트 팀이 참석하는 스프린트 계획 미팅을 통해 사용자의 요구사항을 기술한 프로덕트 백로그(product backlog)를 바탕으로 해당 스프린트의 목표를 설정한다. 그리고 이에 따른 작업 범위, 개발 담당자, 작업 예상 시간 등 구체적인 개발 계획이 포함된 스프린트 백로그가 작성된다. 새로운 스프

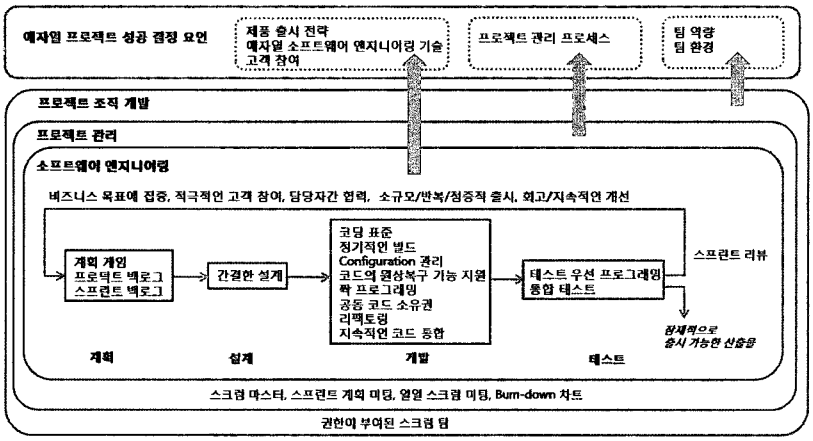

그림 3 민첩도와 애자일 프로젝트 성공요소를 반영한 AFSP 프랙티스 풀

린트가 시작되면 스크럼 마스터가 주축이 되어 일일 스크럼을 운영한다. 이를 통해 프로젝트 진행 사항, 협력이 필요한 부분 등을 하루 단위로 점검하여 스프린트 목표 달성을 위해 팀 전체가 단합한다. 매 스프린트 종료 시점에 전체 팀이 참여하는 스프린트 리뷰 미팅에서는 짧은 데모를 통해 해당 스프린트의 산출물을 공유하고 새로이 발견된 문제점이나 개선점 등을 논의한다. 애자일 회고라고 불리는 이러한 과정을 거쳐 팀과 고객의 피드백을 다음 스프린트에 신속하게 반영하고 프로젝트 수행 과정을 점진적으로 개선해 나간다.

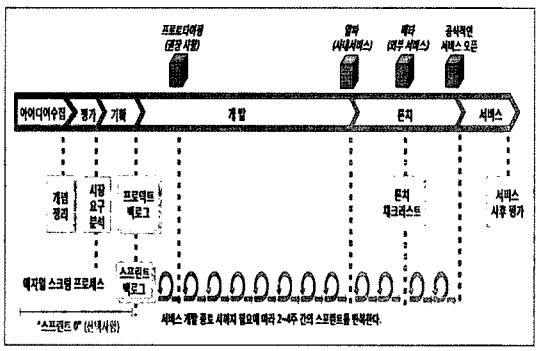

그림 2 계획 중심 방법론의 장점과 스크럼을 결합한 AFSP 프로세스

3.3 AFSP 프랙티스 풀

AFSP 프랙티스 풀은 다음과 같이 3가지 기준에 의해 설계되었다. 첫째, 대표적인 애자일 방법론인 스크럼, XP, FDD, DSDM, 크리스탈 클리어의 각 프랙티스 가운데 민첩도(Degree of Agility)[19]가 높은 것을 그 대상으로 하였다. 둘째, 6대 애자일 프로젝트 성공 요소[20]-출시 전략, 애자일 소프트웨어 엔지니어링 기술, 팀 역량, 프로젝트 관리 프로세스, 팀 환경, 고객 참여-를 반영하여 성공적인 프로젝트 수행에 필수적인 사항을 고려하였다. 셋째, 소프트웨어 개발 생명 주기를 반영하여 프로젝트 팀 구성, 프로젝트 관리 및 실제 개발에 유용한 소프트웨어 엔지니어링 프랙티스를 구분하였다. 또한, 소프트웨어 엔지니어링 프랙티스를 계획, 설계, 코딩, 테스트, 출시 단계 별로 세분하여 현업에서 적용이 용이하도록 하였다.

3.4 AFSP의 민첩도 평가

본 절에서는 4-DAT를 이용하여 AFSP 프로세스와 프랙티스의 민첩도를 평가하였다. 4-DAT는 소프트웨어 개발 방법론에 대한 민첩도를 평가하기 위해 방법론 공학에 근거하여 개발된 4차원 분석 프레임워크이다. 이를 이용하여 특정 방법론에 대한 민첩도*(DA(Method))*를 평가하기 위해 해당 방법론에서 사용된 프로세스의 구성 단계 및 프랙티스의 민첩도인 *DA(Method, Phases)*와 *DA(Method, Practices)*를 각각 산출한다. 즉, 각 프로세스 구성 단계 및 프랙티스의 유연성(flexibility: FY), 신속성(speed: SD), 경제성(leanness: LS), 학습효과(learning: LG), 대응성(responsive: RS) 여부를 이진법으로 측정하여 모두 합산하고 전체 항목 수(m)로 나누어 구한 값을 식 (1)과 같이 계산한다[23].

$$DA(Method)=\frac{1}{m}\sum mDA(Method, Phases\ or\ Practice) \quad (1)$$

민첩도는 0부터 1 사이의 값으로 숫자가 높을수록 민첩성(agility)의 정도가 높으며, 프로세스 혹은 프랙티스 등이 민첩한지 여부를 판단하기 위한 threshold는 0.5에서 0.6 사이이다. 본 논문에서 *DA(AFSP, Phases)*와 *DA(AFSP, Practices)*를 계산한 결과, 각각 0.8과 0.817이 나왔다(표 1). 이를 기존의 애자일 방법론에 대한 평

가 결과[19]와 비교하면, 프로세스의 경우 가장 민첩도가 높은 크리스탈 패밀리(0.8)와 동일하고, 프랙티스의 경우는 가장 민첩도가 높은 스크럼(0.8)보다 높은 수치이다. 여기서 산출된 각 프로세스와 프랙티스 값을 이용하여 *DA(AFSP)*를 구하면 0.809로 기존의 애자일 방법론 가운데 민첩도가 높은 순서인 크리스탈 패밀리, XP, 스크럼에 비해 각각 5%, 11%, 15% 향상되었음을 알 수 있다.

표 1 기존 애자일 방법론과 AFSP의 민첩도 비교

방법론	*DA(Phases)*	*DA(Practices)*	*DA(Method)*
XP	0.702	0.732	0.717
스크럼	0.603	0.800	0.702
FDD	0.485	0.703	0.594
DSDM	0.466	0.684	0.575
크리스탈	0.800	0.732	0.766
AFSP	0.800	0.817	0.809

4. 소규모 프로젝트용 애자일 프레임워크(AFSP) 구현 및 평가

AFSP의 효율성을 입증하기 위하여 3.1절의 그림 1에서 소개한 5가지 적용 단계를 거쳐 미국 야후사의 소규모 프로젝트 4건에 AFSP를 그림 4와 같이 실제 구현하였다. 또한, 관련 기존 연구를 바탕으로 다음과 같이 4가지 척도를 이용하여 적용 사례에 대한 평가를 수행하였다.

첫째, 개발 전략 결정 모델로써 Boehm과 Turner의 5대 민첩성 요소[17]-개발인력의 전문성 정도, 프로젝트 중요도, 팀 크기, 팀 문화, 사용자 요구사항의 변화율-를 평가하여 본 적용사례가 애자일 프로젝트에 적합한지 여부를 판단했다. 분석 결과, 4건의 적용사례 모두 전통적인 계획 중심의 방법론 보다 애자일 방법론의 적용이 적합하였다.

둘째, 앞서 수행한 각 프로젝트 개발 환경에 대한 분석 결과를 기반으로 AFSP 프랙티스 풀로부터 적합한 애자일 프랙티스를 선정하였다. 이렇게 선정된 프랙티스를 내포한 AFSP 프로세스에 따라 스프린트 단위로 각 개발 산출물을 반복 점증적으로 출시하였다.

셋째, Qumer와 Handerson-Sellers의 애자일 도입 및 개선 모델[21]을 바탕으로 각 적용 사례의 애자일 도입 및 개선 수준(Agile Adoption and Improvement Model Level: AAIML)을 평가하여 AFSP의 적응성을 보였다. AAIM은 다음과 같이 총 3단계에 포함되는 여섯 레벨로 구성된다. AAIML1은 애자일 도입의 착수단계로써 신속, 유연한 대응성을 지향한다. 그 상위 단계인 AAIML2부터 AAIML4는 핵심 단계로써 각각 커뮤니케이션, 실행 가능 산출물, 그리고 인적 자원에 주요 가치를 둔다. 끝으로 애자일 도입의 절정단계인 AAIML5는 학습 지향, AAIML6은 저비용 고효율 지향 및 유지를 목표로 프로젝트의 효율성을 극대화해 나간다. 본 사례 연구의 평가 결과 애자일 방법론에 대한 교육 및 코칭 시스템이 상대적으로 잘 갖추어진 환경에서 수행된 사례 1의 경우 AAIML3, 상대적으로 제반 환경이 부족하지만 애자일 기법에 경험이 있는 팀이 주축이 된 사례 2의 경우 AAIML2, 유사한 환경에서 처음으로 애자일을 도입했던 사례 3과 4의 경우 애자일 도입의 기초단계인 AAIML1 수준의 속성을 보여 AFSP가 다양한 환경에서 무리 없이 적용됨을 보였다. 역으로, 그림 1에서 제안한 바와 같이 AAIM을 애자일 프로젝트 수행 전 도입 혹은 개선 목표 및 전략을 수립하는데 활용할 수도 있다.

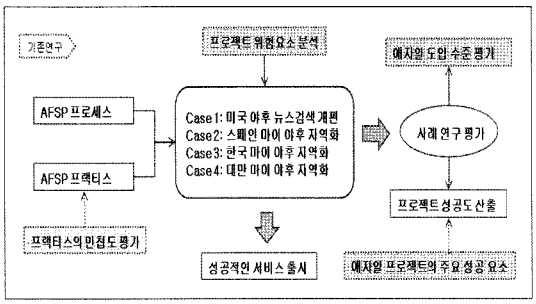

그림 4 본 논문에서 수행한 AFSP 기반의 사례 연구 및 평가

넷째, Chow와 Cao가 제안한 애자일 프로젝트 성공요소 및 그 속성[20]에 대한 연구를 기반으로 본 연구에서는 애자일 프로젝트 성공도를 다음과 같은 공식을 개발하여 정량적으로 산출하였다. 즉, 프로젝트 성공을 결정하는 품질, 범위, 시간, 비용의 차원에서 각 적용 사례의 프로젝트 성공 정도를 진단하였다. 애자일 프로젝트 성공도(Degree of Agile Project Success: DAPS)는 0부터 1 사이의 값으로 숫자가 높을수록 프로젝트의 주요 성공요소를 잘 수행한 것이며, 프로젝트의 성공 여부를 판단하기 위한 threshold는 대략 0.7에서 0.8 사이가 적합하다. m은 프로젝트 성공 요소의 개수, W는 각 성공요소의 가중치, N은 각 성공요소 별 속성의 개수, Sum은 각 속성값의 총합이라 할 때 $DAPS$ 값은 다음과 같이 산출된다.

$$DAPS(Project) = \sum_{i=1}^{m} W_i \times \frac{Sum_i}{N_i} \quad (2)$$

식 (2)의 적용 결과 각 사례 연구의 $DAPS$값은 사례 1부터 순서대로 0.95, 0.835, 0.755, 0.877이 나왔으며, 이는 각 프로젝트 팀이 진단한 프로젝트 결과에 대한 만족도와도 대체적으로 일치하였다. 결과적으로 AFSP

를 적용한 4개 프로젝트 모두 각 스프린트 목표를 점증적으로 달성함으로써 납기 내에 성공적으로 완료되었다.

5. 결 론

본 논문에서는 기존의 적용사례와 연구 결과를 토대로 소규모 프로젝트 개발에 최적화된 애자일 방법론 기반의 프레임워크를 설계하였다. 또한, 이를 4건의 소규모 웹 어플리케이션 프로젝트에 실제 적용하고 다양한 관점에서 평가함으로써 애자일 프로젝트에 대한 새로운 평가 모델을 제시하였다. AFSP의 성공 요인은 소규모 프로젝트의 특성을 고려하여 민첩도가 높은 프로세스와 프랙티스를 선별하여 적용한 데에 있다. 민첩도가 높은 소프트웨어 개발 방법론은 인간 중심적, 커뮤니케이션 지향적이며, 시장의 변화에 유연하게 대응한다. 또한, 짧은 주기의 반복 점증적인 제품 출시를 통해 저비용 고효율과 지속적인 개선을 추구함으로써 비즈니스의 성공을 지향한다. 본 논문에서 제안한 소규모용 애자일 프레임워크 및 적용 사례는 실제 산업 현장에서 중소규모 프로젝트를 위한 애자일 도입 및 평가 모델로 활용될 수 있다고 판단된다.

참 고 문 헌

[1] I. Richardson, C.G. von Wangenheim, "Why Are Small Software Organizations Different?," *IEEE Softw.*, vol.24, no.1, pp.18-22, 2007.

[2] F. M. Caffery, P. S. Taylor, and G. Coleman, Adept: A Unified Assessment Method for Small Software Companies, *IEEE Softw.*, vol.24, no.1, pp.24-31, 2007.

[3] S. Rowe, Project Management for Small Projects, Management Concepts, 2006.

[4] Agile Manifesto, Manifesto for Agile Software Development Retrieved September 2008, 2001, available online at http://www.agilemanifesto.org.

[5] C. Schwaber, Enterprise Agile Adoption in 2007, Forrester Research, 2008.

[6] O. Salo, P. Abrahamsson, "Agile Methods in European Embedded Development Organizations: a survey study of Extreme Programming and Scrum," *IET Softw.*, vol.2, pp.58-64, 2008.

[7] T. Dyba°, T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Information and Software Technology*, vol.50, no.9-10, pp.833-859, 2008.

[8] A. Begel, N. Nagappan, "Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study," 1st Int. Symposium on Empirical Software Engineering and Metrics, 2007.

[9] B. Fitzgerald, G. Hartnett, K. Conboy, "Customizing agile methods to software practices at intel Shannon," *European Journal of Information Systems*, vol.15, no.2, pp.200-213, 2006.

[10] J. Sutherland, K. Schwaber, The scrum papers: nuts, bolts, and origins of an agile method, Scrum Inc., 2007.

[11] J. Sutherland, G. Schoonheim, E. Rustenburg, and M. Rijk, "Fully Distributed Scrum: The Secret Sauce for Hyperproductive Offshored Development Teams," *Proc. of Agile Conference 2008*, pp.339-344, 2008.

[12] L. Rising L, N.S. Janoff, "The Scrum Software Development Process for Small Teams," *IEEE Softw*, vol.17, no.4, pp.26-32, 2000.

[13] E. Trengove, B. Dwolatzky, "A software development process for small projects," *International Journal of Electrical Engineering Education*, vol.41, no.1, pp.10-27, 2004.

[14] N. Habra, S. Alexandre, J-M Desharnais, C.Y. Laporte, "A. Renault, Initiating software process improvement in very small enterprises," *Information and Software Technology*, vol.50, no.7-8, pp.763-771, 2008.

[15] L. Wang, "Agility counts in developing small-size software," *IEEE Potentials*, vol. 26, no.6, pp.16-23, 2007.

[16] B. Boehm, Get Ready for Agile Methods, with Care, *IEEE Comput*, vol.35, no.1, pp.64-69, 2002.

[17] B. Boehm, R. Turner, "Using Risk to Balance Agile and Plan-Driven Methods," *IEEE Comput.*, vol.36, no.6, pp.57-66, 2003.

[18] D. Cohen, M. Lindvall, P. Costa, "An introduction to agile methods, Advances in Computers," vol.62, pp.1-66, 2004.

[19] A. Qumer, B. Henderson-Sellers, "An evaluation of the degree of agility in six agile methods and its applicability for method engineering," *Information and Software Technology*, vol.50, pp.280-295, 2008.

[20] T. Chow, D. Cao, "A survey study of critical success factors in agile software projects," The Journal of Systems and Software, vol.81, pp.961-971, 2008.

[21] A. Qumer, B. Henderson-Sellers, "A framework to support the evaluation, adoption and improvement of agile methods in practice," The Journal of Systems and Software, vol.81, pp. 1899-1919, 2008.