http://dx.doi.org/10.14400/JDPM.2013.11.12.439

스크럼 방법에서 백로그 바인더를 이용한 재사용 지원

김지홍*
가천대학교 IT대학 컴퓨터공학과*

Support of Reuse in Scrum Method with Backlog Binder

Ji-Hong Kim*
Dept. of Computer Engineering, College of IT, Gachon University*

요 약 애자일 방법과 소프트웨어 재사용의 장점을 통합하는 여러 연구가 나타나고 있다. 대부분의 연구는 스크럼 방법에 재사용을 도입하기보다는 소프트웨어 프로덕트라인에 애자일 방법을 도입하고 있다. 스크럼은 가장 인기있는 애자일 방법이지만 재사용 결합의 연구는 부족하다. 한편, 백로그와 점증적 산출물과 같은 스크럼 개발의 자산은 재사용이 가능하다. 본 연구는 스크럼 방법에서 재사용이 가능한 통합 자산인 백로그 바인더를 식별하고 이들의 재사용을 지원하는 백로그 바인더 재사용 방안을 제안하였다. 아울러, 제안한 기술을 구인 구직 응용에 적용하여 백로그 바인더와 자산 재사용의 프로토타이핑을 보일 수 있었다.

주제어 : 스크럼, 애자일 소프트웨어 개발, 소프트웨어 재사용, 애자일 방법, 소프트웨어 공학

Abstract There has been a growing amount of research on combining Agile methods and software reuse. Most of it introduces Agile into software product line rather than software reuse into Scrum method. Meanwhile, some assets such as backlogs and incremental artifacts in Scrum development are reusable.
In this paper, we identify a backlog binder that aggregates reusable Scrum assets and proposes a backlog binder reuse technique. In addition, we can apply the proposed technique and show prototyping of backlog binder reuse in job matching applications.

Key Words : Scrum, Agile Software Development, Software Reuse, Agile Method, Software Engineering

1. 서론

오늘날 소프트웨어 개발은 전통적 방식의 계획주도 개발에서 애자일 개발로 패러다임이 빠른 속도로 진행되고 있다[1,2]. 이러한 변화는 새로운 과제와 시도를 낳고 있으며 학계와 산업계에서는 애자일 방법과 소프트웨어 재사용을 결합하는 연구와 성공 사례 보고가 늘어나고 있다[3,4].

대부분의 애자일과 소프트웨어 재사용을 통합하는 연구는 주로 소프트웨어 프로덕트라인에 애자일 방법을 도입하는 것들이다[5,6]. 반대로, 애자일 방법가운데 가장 인기 있는 스크럼에 대한 재사용의 결합 연구는 부족하다.

특히, 애자일 방법에서는 재사용을 위한 사전의 노력들을 급변하는 인터넷 시대에 투자회수가 불확실한 리스크로 인식되어 재사용이 강조되지 않고 있다[7]. 그러나 스크럼에서는 백로그와 산출물이 기본적으로 제공되므

* 이 논문은 2013년도 가천대학교 교내연구비 지원에 의한 결과임.(GCU-2013-R406)
Received 1 December 2013, Revised 20 December 2013
Accepted 20 December 2013
Corresponding Author: Ji-Hong Kim(Gachon University)
Email: wiskjh@gachon.ac.kr

ISSN: 1738-1916

로 이들을 하나로 묶어, 접근과 사용이 용이한 단위체를 통해 재사용의 지원이 가능하다.

본 논문에서는 스크럼 방법에서 재사용을 지원하는 통합 자산인 백로그 바인더를 식별하고 이들의 재사용 방안을 제시한다.

본 논문의 구성은 2장에서 기초 연구에 속하는 계획 주도의 개발, 애자일 방법, 재사용 그리고 백로그 팩토링에 관해 알아본다. 3장에서는 백로그 바인더와 백로그 바인더 재사용 방안을 제안하고, 4장에서는 프로토타이핑을 보인다. 5장에서는 관련 연구를 비교하고, 6장에서 결론을 기술한다.

2. 기초 연구

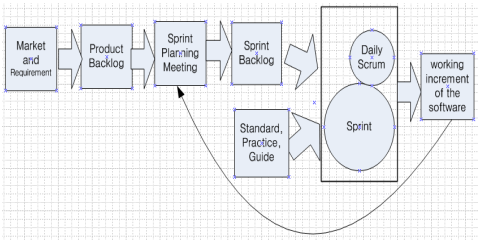
2.1 계획 주도의 개발

소프트웨어 프로세스는 계획주도(plan-driven)와 애자일 개발 프로세스로 분류될 수 있다[8]. 계획 주도는 모든 프로세스 활동이 사전에 계획되고, 이후의 진전은 이 계획을 바탕으로 측정되는 프로세스이다. 이런 방식의 개발은 대량의 문서와 순차적인 소프트웨어 개발 활동의 수행을 강조하고 있으며, 때때로 이러한 방법을 문서 주도 방법 또는 중량급 방법이라 부른다[9]. 이런 방식의 개발은 최종 제품의 모든 속성을 상세히 명시하고 확정된 결과를 명확하게 보여주는 것이 가능한 환경에서 성공적이며 금융 기관, 군사용, 정부 프로젝트에 여전히 많이 사용되고 있다[10].

그러나 시장이 보다 역동적으로 됨에 따라, 기업들은 이런 변화에 빠르게 대응하는 애자일 방법이 필요하게 되었다.

2.2 애자일 방법과 스크럼

애자일 방법(Agile Method : AM)은 정형화된 계획 주도의 전통적 개발방법의 경직성을 개선하여, 잦은 고객 요구사항의 변경을 수용하면서 품질 좋은 소프트웨어를 개발할 수 있도록 나타났다[11]. 이들의 핵심적 개념으로는 동작되는 소프트웨어의 빠른 개발, 시장 출시 시간 단축, 짧은 타임 박스의 반복, 개발팀과 고객간 피드백 단축, 지속적인 통합, 자동화된 회귀 테스트를 들 수 있다. 아울러, 개발에서는 문서의 최소화, 테스트 주도, 짝 프로그래밍의 코드 중심적이다[7].



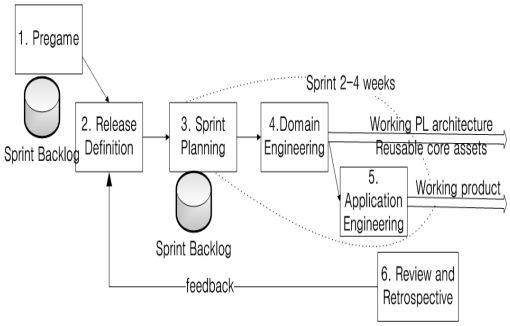
[Fig. 1] Overview of the Scrum process

스크럼은 애자일 방법 가운데 가장 인기 있는 프레임워크이며 그림 1과 같이 스프린트라고 하는 1-4주 단위의 이터레이션을 통해 진행된다. 시장으로부터의 요구사항이 발생되면 제품 백로그에 추가되고 매 이터레이션 초기에 열리는 스프린트 계획회의에서 한 번의 스프린트 동안에 개발할 양의 요구사항을 제품 백로그로부터 선택한다. 이어서, 팀은 선택된 요구사항을 작업 수준의 태스크로 세분화하여 스프린트 백로그로 옮겨 개발을 수행한다. 하나의 스프린트는 반복되는 일일 스크럼(daily scrum)으로 수행되며, 스프린트가 끝나면 동작되는 제품 증분을 갖게 된다.

2.3 재사용과 애자일 프로덕트라인 공학

소프트웨어 재사용은 소프트웨어 시스템의 일부를 새로운 개발에 반복적으로 사용하여 생산성을 향상시키는 기술 및 접근방법이다[11,12]. 이런 접근 가운데 계획 주도 방식인 프로덕트라인 공학(Product Line Engineering : PLE)은 제품군에 속하는 제품들의 공통점과 가변점을 사전에 식별하고 준비하여 새로운 응용에 대규모의 재사용을 지원하는 방법이다. 이는 재사용 가능한 컴포넌트를 개발하는 도메인 공학(Domain Engineering : DE)과 새로운 응용을 개발하는 어플리케이션 공학(Application Engineering : AE)으로 구성된다[13].

애자일 프로덕트라인 공학(Agile Product Line Engineering : APLE)은 소프트웨어 개발 시 PLE나 AM 각각의 기술이 단독으로 적용되었을 때에 나타나는 부족한 점을 대처하기 위해 각 장점의 통합을 혜택 받기위한 시도이다[5,14].



[Fig. 2] Overview of the APLE Scrum Process

APLE는 주로 PLE에 특정 애자일 방법이나 AM의 원칙 또는 프랙티스가 적용되고 있다. 그림 2는 PLE의 DE와 AE 활동을 위해 스크럼의 스프린트 계획과 백로그가 도입된 APLE 스크럼 프로세스를 보여준다. 대부분의 애자일과 재사용의 결합은 애자일에 재사용을 구현하기보다 PLE에 애자일의 도입에 초점이 맞추어져 있다[1,14,15].

2.4 백로그 재사용과 백로그 팩토링

백로그 재사용은 이전에 생성된 제품 백로그와 스프린트 백로그에 있는 항목을 새로운 개발에 다시 사용하여 빠른 개발과 비용절감을 촉진하는 새로운 형태의 소프트웨어 재사용이며 백로그 팩토링을 통해 이루어진다[16].

백로그 팩토링은 각 스프린트에 수행할 요구사항이나 작업을 결정할 때, 이전에 수행된 스토리나 태스크와 동일 또는 유사 유무를 간단하게 발견하여 재사용이 고려된 항목을 제시하는 활동 및 기술이다[16].

3. 재사용을 지원하는 백로그 바인더

3.1 백로그 바인더와 자산

3.1.1 백로그 바인더

백로그 바인더(Backlog Binder : BB)는 동적으로 변화하는 시장의 요구에 능동적으로 대응하기 위해 스크럼 방법으로 소프트웨어 개발 시 생성되는 각종 백로그와 주요 산출물을 하나로 모아놓은 묶음이다. 이는 개발 기간 동안이나 이후 작업에서 필요한 결과물의 손쉬운 접근과 사용을 허락한다. BB는 표 1과 같이 기본 백로그(Elementary Backlog : EB), 프로젝트 규모에 따라 생략 가능한 선택 백로그(Optional Backlog : OB), 그리고 산출물(Artifact : AF)로 구성된다.

〈Table 1〉 Backlog Binder Item

Binder Item	Description
Elementary Backlog	Backlog for Product/Iteration
Optional Backlog	Optional Customizing Backlog
Artifact	Artifacts of Iteration

세부적으로 EB는 요구 백로그(Requirement Backlog : RB)와 반복 백로그(Iteration Backlog : IB)로 이루어진다. RB는 하나의 제품에 필요한 전체적인 요구사항을 위한 백로그이다. 기본적으로, 하나의 제품에는 하나의 RB가 사용되지만 대규모 또는 분산 개발의 경우에는 세분화된 서브 백로그를 가질 수 있다. IB는 스크럼 팀이 매번의 반복에 수행할 작업들로 이루어져 있다. 이는 RB에서 최우선 순위로 구현이 요구되어 선택된 항목들로써 팀 수준의 실질적 작업들이다.

3.1.2 백로그 바인더 자산

BB에 있는 항목은 하나의 제품을 위한 각종 개발의 생성물을 갖고 있기 때문에 후속 개발 활동에서 참조 및 사용될 수 있는 대상이다. 다양한 자산의 관리를 위해 BB는 표 2와 같은 유형을 지원한다.

〈Table 2〉 Asset type for Backlog Binder Item

Binder Item	Asset type
RB	Epic, Story
IB	Task
AF	Analysis, Design, Code, Test

RB에서 재사용 가능한 주된 자산들은 이미 수행된 사용자 스토리 형식의 RB 항목인 요구사항이며 필요에 따라 대형 요구사항을 나타내는 에픽 형식의 요구들도 해당된다.

IB에서 사용 가능한 자산은 요구사항의 구현을 위해 결정된 태스크 형식의 IB 항목이다. 이들은 팀원의 다양한 실행 수준 작업들의 집합이다.

AF의 자산은 각 태스크의 수행으로 나타난 다양한 활동의 결과물이다. 산출물 자산의 관리를 위해, 해당 개발 활동을 대변하는 형이 지원된다.

3.2 백로그 바인더 재사용과 지원

3.2.1 백로그 바인더 재사용

백로그 바인더 재사용은 세부적으로 스토리 재사용, 태스크 재사용, 산출물 재사용을 통해 지원된다.

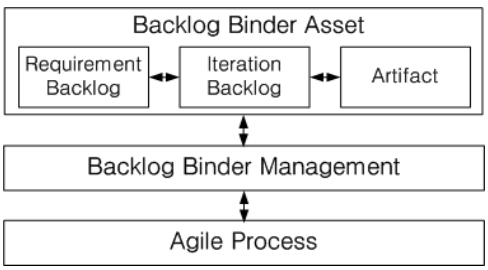
스토리 재사용은 요구 수준에서 지원되는 재사용이다. 이는 패밀리 제품 개발 시 동일 사용법 지원 또는 공통적 기능을 요구하는 스토리들에 대하여 재사용이 발생될 수 있다.

태스크의 재사용은 매 스프린트 동안 요구사항을 구현하는 팀에게 이전 수행 작업과 동일한 요구가 식별되었을 때 발생된다. 이는 재사용 가능한 스토리에 대한 패턴화된 방식의 작업으로 해결하는 경우에 자주 나타난다.

산출물의 재사용은 분석, 설계, 구현, 테스트를 대상으로 다양한 종류의 재사용이 허락된다. 이 가운데 가장 대표적인 것은 코드 재사용이다.

3.2.2 백로그 바인더 재사용 지원

기존의 AM에서 백로그 바인더 자산을 이용한 재사용 처리는 그림 3과 같이 애자일 프로세스, 백로그 바인더 관리, 백로그 바인더 자산의 3부분으로 이루어진다.



[Fig. 3] Structure of Backlog Binder Reuse

애자일 프로세스 부분은 현 프로젝트 개발을 수행하는 스크럼이 해당된다. 스크럼의 반복 점증적 처리와 백로그 바인더 관리를 통해 백로그 접근 및 이용을 한다. 백로그 바인더 관리는 백로그의 생성, 항목 추가/삭제/변경을 주관하며, 백로그 접근이 필요할 때 스토리, 태스크 그리고 산출물에 대한 팩토링을 수행한다.

백로그 바인더 자산 부분은 3.1.2절에서 살펴 본 현 프로젝트에서 산출된 각종 주요 결과물의 집합으로써 스크럼 개발에서의 문서화와 재사용을 지원하는 통합 산출물 단위체이다.

3.3 스크럼 바인더 팩토링과 프로세스

3.3.1 스크럼 백로그 바인더 팩토링

백로그 바인더 팩토링이란 백로그 항목의 우선순위 조정 또는 여러 작은 크기로의 분해시 이전에 수행했던 백로그 항목과의 유사성을 간단히 식별하여 재사용이 고려된 항목을 제시하는 활동 및 기술이다. 이는 BB에 있는 RB와 IB에 대하여 백로그 팩토링의 적용으로 가능하다.

백로그 팩토링을 이용한 바인더 재사용 처리는 그림 4와 같다.

```
BacklogBinderFactoring() {
  CurrentTemporary=GetPriorityTempItem(BacklogBinderItem);
  BacklogFacroring(CurrentTemporary, UpdateItem)
  UpdateBacklog(BacklogBinderItem, UpdateItem);
}
BacklogFactoring(CurrentItem, UpdateItem) {
 FindPreviousAsset(CurrentItem, OldItem);
 DecideCurrentItem(CurrentItem, OldItem, UpdateItem);
}
FindPreviousAsset(Current, OldItem) {
 /* Findout similar OldItem asset with Current one */
}
DecideCurrentItem(Current, Old, Update) {
/* Decide Update from Old and Current */
}
```

[Fig. 4] Overall Backlog Binder Factoring for Reuse

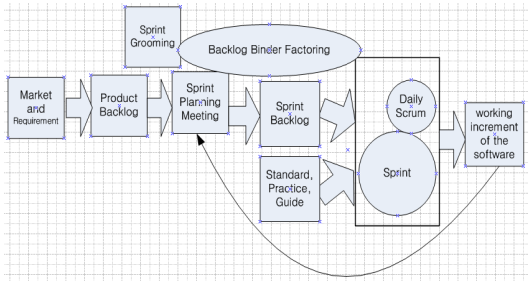
먼저, GetPriorityTempItem(BacklogBinderItem)을 통해 처리하고자하는 최우선 순위로 수행될 후보 백로그 항목(CurrentTemporary)을 백로그에서 얻고, 이에 대해 BacklogFactoring()을 수행한 결과(UpdateItem)를 UpdateBacklog()를 통해 업데이트하고 백로그 바인더의 팩토링을 마친다.

현행 바인더 항목(CurrentItem)에 대하여 백로그 팩토

링을 수행하는 BacklogFactoring()은 FindPreviousAsset()을 통해 이전에 수행된 유사 항목(OldItem) 유무를 간단히 검사하고, DecideCurrentItem()을 통해 재사용이 적용된 항목(UpdateItem)을 할당한다.

3.3.2 스크럼 백로그 바인더 재사용 프로세스

본 논문에서 제안한 백로그 바인더 재사용을 지원하는 확장 스크럼 프로세스는 그림 5와 같다.



[Fig. 5] Extended Scrum Process for Backlog Binder

기존 스크럼 처리에 추가된 백로그 바인더 팩토링은 3.3.1절에 소개된 재사용 활동이다. 이는 스프린트 손질하기, 스프린트 계획회의, 그리고 스프린트 기간 동안에 수행된다. 스크럼에서 스프린트 손질하기는 백로그의 업데이트가 필요할 때 언제든지 수행될 수 있다.

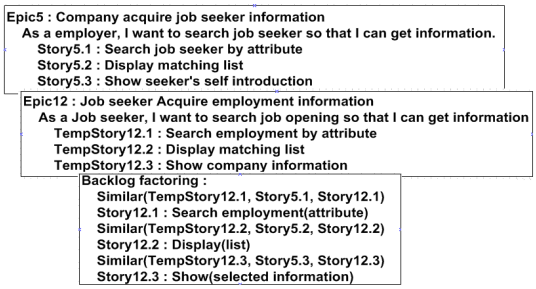
4. 적용과 프로토타이핑

본 장은 3장에서 제안한 백로그 바인더 팩토링 기술을 구인 구직 사이트 응용에 적용하고 요구 백로그 수준에서의 프로토타이핑을 보인다.

• 유사 스토리의 적용

백로그 재사용을 위하여, 한 sprint를 마치고 다음 sprint에서의 유사한 스토리에 대한 팩토링을 먼저 살펴본다.

그림 6의 상위 부분은 기 수행된 여러 스토리 가운데 재사용 대상이 되는 Epic5와 적절한 크기로 분해된 3개의 스토리인 Story5.x를 나타내고 있다.

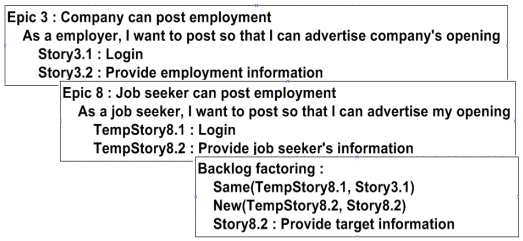


[Fig. 6] Backlog Factoring(Similar Story)

중간 화면은 현 RB에서 최고 우선순위 항목으로 백로그 팩토링 대상이되는 Epic12와 3개로 분해된 임시 스토리TempStory12.x가 있다. 3번째 화면은 3.3.1절에 소개한 백로그 팩토링의 단계를 보인다. 먼저, GetPriorityTempItem()을 통해 우선순위로 처리할 TempStory12.1을 얻어 BacklogFactoring()을 수행한다. 세부적으로 FindPreviousAsset()와 DecideCurrentItem()에 의하여 단순 공통성 검사 결과인 Similar(TempStory12.1, Story5.1, Story12.1)을 수행하고 재사용성있는 Story5.1과 유사한 Story12.1의 할당을 보인다. 나머지 TempStory12.x에도 Similar()가 적용되어 단순화된 Story12.2와 Story12.3의 할당을 보인다.

• 동일 스토리의 적용

여러 스프린트를 끝내고 동일 스토리가 적용되는 새로운 스프린트를 위한 백로그 팩토링을 살펴본다. 그림 7의 상단은 이미 수행된 Epic3와 세분화된 Story3.x를 보여준다. 백로그 팩토링을 위한 Epic8에 대한 임시 스토리인 TempStory8.1과 TempStory8.2가 제시되었다(중앙 화면). TempStory8.1과 Story3.1은 동일성이 식별되어 재사용되고, TempStory8.2은 새로운 스토리로 식별되고 Story8.2의 할당을 나타내고 있다(마지막 화면).



[Fig. 7] Backlog Factoring(Same Story)

5. 관련 연구 비교

일찍이, Tian[17]은 PLE와 AM을 프로젝트관리, 품질 보증 그리고 공학 관점에서 비교하고 애자일 프랙티스를 PLE에 결합을 연구하였다. 그리고 경량급인 APLE로의 테일러링 가능성을 결론지었다. Carbon[7]은 명확하게 정의된 기존의 AM을 직접 적용하기보다는 애자일 원칙을 PLE에 포함시키는 것이 쉽다고 주장하고 PulSE-I을 이용하여 AE에 애자일 원칙의 적용을 보였다. 이들 연구는 애자일 원칙을 PLE에 적용한 연구들이다.

한편 스크럼 프로세스와 PLE를 결합하는 연구들이 있다. Diaz[5]는 스크럼의 스프린트를 2개 서브 프로세스로 나누고 DE와 AE를 각각 지원하는 프로세스를 제안하였다. Kircher[6]는 AM과 PLE에서 공통적으로 사용되는 피처를 중심으로 스크럼과 PLE를 통합하는 프로젝트 성공 사례를 보고하였다. 이들 연구도 스크럼을 이용하였으나 PLE를 위한 접근인데 반해, 본 연구는 스크럼 자산의 재사용을 추구하고 있다.

새로운 관점의 연구로써 Martini[1]는, 대형 소프트웨어 개발에 있어서 재사용과 속도에 영향을 끼치는 인자와 이를 카테고리화한 요인맵(factor map)을 보고하였다. 최근에는 애자일과 PLE를 적용하는 소프트웨어 개발에서 속도와 재사용에 영향을 끼치는 의사소통 요소를 연구하였다[18]. 이들은 속도와 재사용에서의 요인에 집중하는데 비하여 본 연구는 민첩한 방법과 재사용에 중점을 두었다.

6. 결론

본 논문은 AM과 소프트웨어 재사용의 통합을 위해 스크럼 개발에서 BB의 재사용을 연구하였다. 먼저, 애자일의 YAGNI원칙을 준수하며 재사용을 지원하기 위하여 스크럼 개발에서 기본으로 제공되는 백로그와 산출물을 재사용 가능한 자산으로 간주하였다. 그리고 이들 자산을 빠르게 접근, 이해 및 사용할 수 있도록 하나의 단위로 묶는 BB를 식별하고 스크럼 방법에서 재사용을 지원하는 백로그 바인더 팩토링 기술을 제안하였다.

프로토타이핑을 통해, 구직 사이트 응용에 백로그 바인더 팩토링 기술을 적용하고 스크럼에서의 백로그 바인더 자산의 재사용을 보일 수 있었다. 본 기술은 AM을 채택하고 재사용의 장점을 원하면서도, 급변하는 시장 환경에 쓸모가 없을 지도 모르는 재사용의 사전 투자를 피하려는 개발에 혜택을 제공할 수 있다.

앞으로의 연구 과제로, 백로그 바인더 팩토링 기술을 지원하는 소프트웨어 도구의 개발과 애자일과 린의 통합을 통한 대규모 개발에서의 백로그 바인더 팩토링 기술의 이용을 다룰 것이다.

ACKNOWLEDGMENTS

This work was supported by the Gachon University research fund of 2013.(GCU-2013-R406)

REFERENCES

[1] Martini, A., Pareto, L., Bosch, J., Enablers and Inhibitors for Speed with Reuse, Proceedings of the 16th International SPLC, pp. 116-124, 2012.

[2] Ashley Aitken, Vishnu Ilango, A Comparative Analysis of Traditional Software Engineering and Agile Software Development, 2013 46th Hawaii International Conference on System Sciences, IEEE, pp. 4751-4760, 2013.

[3] Diaz, J., Perez, J., Alarcon, P., & Garbajosa, J., Agile product line engineering : a systematic literature review, SP&E, May, pp. 921-941, 2011.

[4] McGregor, J. D., Agile software product lines, deconstructed, Journal of Object Technology, 7(8), November-December. 2008.

[5] Diaz, J., Perez, J., Yague, A., & Garbajosaz, J., Tailoring the Scrum Development Process to Address Agile Product Line Engineering, JISBD, 2011.

[6] Kircher, M., Hofman, P., Combining Systematic Reuse with Agile Development- Experience Report, SPLC, 2012.

[7] Carbon, R., Lindvall, M., Muthig, D., Costa, P., Integrating product line engineering and agile

methods: flexible design up-front vs. incremental design, 2006.
[8] Sommerville, Software Engineering, Pearson, pp. 58-72, 2011.
[9] Hans van Vliet, Software Engineering, Wiley, pp. 50-51, 2008.
[10]http://en.wikiversity.org/wiki/Plan-driven_software_development.
[11] Shari L. Pfleeger, Joanne M. Atlee, Software Engineering, Pearson, pp. 627-633, 2010.
[12] Yuri Chermak, Requirements reuse : the state of the practice, Conference on SWSSTE, IEEE, pp. 46-53, 2012.
[13] Klaus Pohl, van der Linden F., Software Product Line Engineering, Springer, pp. 4-22, 2005.
[14] Noor M., Rabiser R, Gr¨unbacher P., Agile product line planning: A collaborative approach and a case study. Journal of Systems and Software, 81(6), pp. 868 - 882, 2008.
[15] Silva, I., Neto, S., Almeida, D., & Meira, L., Agile software product lines: a systematic mapping study, SP&E, July, pp. 899-920, 2011.
[16] Ji-Hong Kim, Backlog Factoring : Extension of Task Factoring for Reuse in Scrum Method, The Journal of Digital Policy and Management v.10, n.10, pp. 339-345, 2012.
[17] Tian K., Cooper K., Agile and software product line methods: Are they so different? APLE '06, 2006.
[18] Martini, A., Pareto, L., Bosch, Communication factors for speed and reuse in large scale agile software development, Proceedings of the 17th International SPLC, pp. 42-51, 2013.

김 지 홍(Kim, Ji Hong)

- 1974년 2월 : 경희대학교 전자공학과(공학사)
- 1982년 8월 : California State University (Fullerton) Computer Science(이학석사)
- 1995년 8월 : 경희대학교 전자계산공학과(공학박사)
- 1982년 5월 ~ 89년 2월 : 미국 Interpac Software, 소프트웨어 엔지니어
- 1989년 3월~현재 : 가천대 컴퓨터공학과 교수
- 관심분야 : 소프트웨어공학, UML, 소프트웨어 재사용, 애자일 개발
- E-Mail : wiskjh@gachon.ac.kr