한국사물인터넷학회논문지 Vol. 6, No. 2, pp. 87-92, 2020 https://doi.org/10.20465/KIOTS.2020.6.2.087

사물인터넷 환경에서 소프트웨어 공학 기반의 소프트웨어 개발 진척 분석

이성훈[1], 이동우[2*]
[1]백석대학교 ICT학부 교수, [2]우송대학교 컴퓨터정보학과 교수

A Software Engineering-Based Software Development Progress Analysis in IoT Environment

Seong-Hoon Lee[1], Dong-Woo Lee[2*]
[1]Professor, Division of ICT, Baekseok University
[2]Professor, Department of Computer Information, Woosong University

요 약 우리가 생활하고 있는 주변 환경들은 정보통신 기술(ICT)의 영향으로 인하여 시시각각 변화하고 있다. 이러한 변화의 중심에는 산업분야뿐만 아니라 대부분의 생활 영역에서 나타나고 있다. 정보통신 기술의 중심에는 소프트웨어를 비롯한 지능화, 센싱 기술 등이 있다. 정부 및 관련기관에서는 다양한 소프트웨어 육성을 위한 정책들을 추진하고 있으며, 이러한 정책으로 꾸준히 소프트웨어 관련 산업은 발전하고 있다. 소프트웨어 육성과 관련한 긍정적인 면이 존재하지만 또한 부정적인 내용도 나타나고 있다. 소프트웨어 개발에 따른 복제 문제, 진척도 문제등이 소프트웨어 산업이 양적으로 증가하면서 이들 문제들도 증가하는 모습을 보인다. 본 연구에서는 소프트웨어 개발 과정에서 나타나는 개발 진척도와 관련한 문제가 발생할 경우 문제 해결 방안으로서 소프트웨어 공학을 기반으로 한 좀 더 객관성을 지닌 방안을 제시하였다.

주제어 : 사물인터넷, 지능, 정보통신기술, 소프트웨어, 소프트웨어 개발 주기

Abstract The surrounding environments in which we live are changing from time to time due to the influence of ICT. At the heart of this change is not only the industrial sector, but it appears in most areas of everyday life. At the center of information and communication technology are software, intelligence, and sensing technology. The government and related organizations are promoting policies to foster various software, and with these policies, the software-related industry is steadily developing. There are positive aspects about software development, but also negative ones. The problems of duplication and progress due to software development have been increasing as the software industry has increased in quantity. In this study, we proposed a more objective method based on software engineering as a solution to problems when problems related to development progress occurred during the software development process.

Key Words : IoT, Intelligence, ICT, Software, Software Development Life Cycle

*교신저자 : 이동우(dwlee@wsu.ac.kr)
접수일 2020년 3월 20일 수정일 2020년 4월 10일 심사완료일 2020년 4월 23일

1. 서론

현재 전 세계적으로 나타나는 산업 구조 변화 과정에서의 핵심적인 내용은 4차 산업혁명이라 할 수 있다. 4차 산업혁명을 통해 현재 사회 및 산업 구조는 기계 중심적인 구조에서 센싱, 지능적인 산업구조로의 변화를 꾀하고 있다. 따라서 4차 산업혁명 시대의 주요 핵심 기술은 센싱 기반의 사물인터넷 기술, 지능적인 기술, 5세대 통신 등을 포함하고 있다. 이러한 모든 기술적인 면에는 전자적인 요소들 뿐 아니라 소프트웨어가 기본적으로 필요하다.

우리나라에서도 소프트웨어가 기반이 되고 있는 ICT 기반의 산업들이 전체 산업의 견인 역할을 하면서 내수 및 수출등을 주도하고 있다고 볼 수 있다. 정부에서도 소프트웨어 산업 발전을 위해 다양한 제도 개선 및 소프트웨어 육성 프로그램등을 통해 소프트웨어 산업 분야를 지원하고 있다. 일례로 대학의 소프트웨어 인력 양성을 위해 지원해오고 있는 '소프트웨어중심대학'지원사업을 들 수 있다. 이러한 다양한 대학 및 산업계에 대한 지원은 바람직하다고 볼 수 있다. 다양한 자원이 절대적으로 부족한 우리입장에서는 미래 사회를 위한 먹거리를 해결할 수 있는 대안 중의 하나로 소프트웨어 산업이 될 수 있기 때문이다. 소프트웨어 산업 지원에 대해 전체적으로는 긍정적인 결과가 있을 것으로 보이나 한편으로는 부정적인 측면이 있을 것으로도 보인다.

컴퓨터 소프트웨어 혹은 소프트웨어는 컴퓨터 시스템의 일부분이다[1-5]. 이러한 소프트웨어를 개발하는 소프트웨어 산업은 무에서 유를 창출한다는 측면에서 보면 오랜 시간과 노력을 투자해야만 한다. 따라서 일부 기업들에서는 자의적으로, 혹은 타의적으로 오랜 시간과 노력을 투자하지 않고 결과를 얻으려 하는 경우들이 발생하고 있다. 이를 반증하는 상황들이 소프트웨어 보호로 나타나고 있다. 대다수의 성실한 기업들이 많은 시간과 재정, 노력들을 하면서 소프트웨어 산업을 발전시키고 있지만 일부 기업들에서는 시간 및 재정 투자등의 노력 없이 기존 기업들의 소프트웨어를 복제하거나 도용하는 경우가 발생하면서, 소프트웨어 산업의 건전한 발전을 저해하고 있는 것이다.

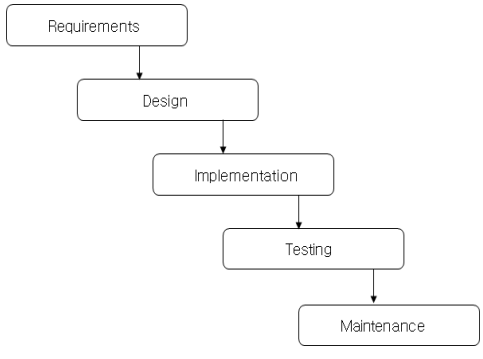
본 연구에서는 이러한 소프트웨어 산업의 건전한 발전을 지원하기 위해 소프트웨어 공학적 측면을 기본으로 하는 객관성을 포함한 소프트웨어에 대한 개발 진척도를 판단하는 방안을 제시하고자 한다. 2장에서는 본 연구의 대상이 되는 소프트웨어 개발 진척 문제와 연관된 내용들의 상황 및 소프트웨어 공학에서의 폭포수 모델에 대해 살펴보았다. 3장에서는 폭포수 모델을 기반하는 소프트웨어 개발 진척도 개선 방안에 대해 기술하였다. 마지막으로 4장에서 결론을 기술하였다.

2. 소프트웨어 개발 주기 및 관련 문제

2.1 소프트웨어 개발 전주기

소프트웨어 개발을 위해서는 소프트웨어 개발 전주기(Software Development Life Cycle)를 다루는 있는 소프트웨어 공학이 기본으로 이용된다. 소프트웨어 공학은 소프트웨어의 개발, 운용, 유지보수 등의 생명 주기 전반을 체계적으로 다루는 학문으로서 관련한 의미 해석들이 다양하다[6-9]. 소프트웨어 공학에서의 소프트웨어 개발 전 주기는 소프트웨어를 개발하기 위해 행해지는 전체 과정에 대해 기술한 것이다. 이러한 전체 과정을 통하여 소프트웨어 개발에 대한 내용 및 과정들을 이해하고, 공유할 수 있는 도구라 할 수 있다. 이러한 모델 중 가장 보편화된 모델이 폭포수 모델이라 할 수 있다.

폭포수 모델은 순차적으로 개발 과정이 이루어지는 모델로서 순차적인 소프트웨어 개발 전주기라 한다[1-3]. 개발 주기 과정이 요구사항 분석단계로부터 시작하여 소프트웨어 설계, 소프트웨어 구현, 소프트웨어 테스트(시험), 소프트웨어 통합단계, 소프트웨어 유지보수 단계로 마무리되는 구조이다. 이러한 폭포수 모델의 기본은 소프트웨어 전 주기에서 행해지는 과정들이 100% 순차적인 구조라는 점이다. 따라서 한 단계가 완료되어야 다음 단계 과정이 수행되게 된다. 폭포수 모델이 다른 모델들과 근본적으로 다른 점은 피드백의 의미가 포함되지 않은 100% 순차적인 구조라는 점이다. 아래 그림 1은 전형적인 폭포수 모델을 나타낸 것이다.



[Fig. 1] Waterfall Model

소프트웨어 개발 기업들은 개발 프로그램에 대해 각자의 소프트웨어 개발 주기를 문서화하고 있다. 물론 기업마다 조금씩의 변형들이 있을 수 있지만 소프트웨어 개발의 전 과정을 포함하는 내용들로 구성되어 있는 것이 보편적이다. 따라서 문서화된 개발 주기를 기반으로 하여 소프트웨어 개발 진척과 관련한 문제가 제기 될 시에 폭포수 모델에 근거한 소프트웨어 공학 기반의 소프트웨어 개발 진척도 산정 방안을 제시하고자 한다.

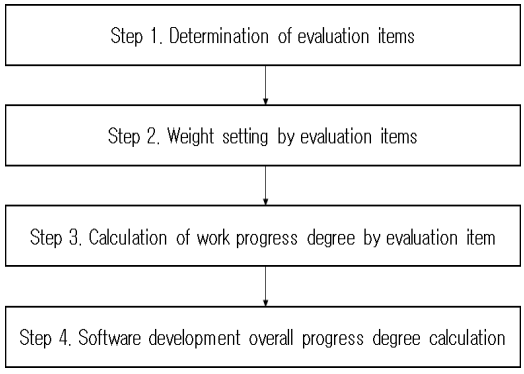
2.2 개발 전주기 관련 판단 문제

소프트웨어 개발과 관련하여 다양한 문제들이 발생하여 왔다. 소프트웨어 개발 결과물에 대한 복제 여부와 관련한 문제, 소프트웨어 개발 과정에서 발생하는 당사자들 간의 개발 진척도와 관련한 문제, 소프트웨어 개발 비용의 적정성 문제 등이 해당된다. 이러한 문제들 중에서 본 연구에서는 소프트웨어 개발 과정에서 발생할 수 있는 개발 의뢰 기업과 개발사간의 개발 진척도와 관련한 분쟁이 발생했을 때 객관성을 향상하면서 소프트웨어 공학 기반의 해결방안에 대해 다루었다.

소프트웨어 개발 주기 문제에 대한 판단의 특성은 크게 3가지로 요약할 수 있다. 먼저, 다양한 형태로 이루어지는 소프트웨어 개발 주기로 인하여 개발 주기에 대한 판단 방법도 다양하다는 점이다. 다음으로 개발 프로세스에 대한 진척도 혹은 완성도등에 대한 판별에 대한 문제 제기등이 다양하게 나타나고 있다. 이러한 특성으로 인하여 소프트웨어 개발 주기와 관련된 제기된 문제에 대해 판단 유무를 수행하는데 야기될 수 있는 문제점 및 고려사항들은 다음과 같다. 먼저 평가항목 설정의 다양성을 들 수 있다. 각 기업마다 소프트웨어 개발 전주기 내용이 다양하므로 이에 따른 평가 항목들이 다양하게 나타남으로서 표준화된 안으로의 수행이 어려워진다. 또한 항목 요소에 대한 가중치를 산정하는데 사용될 수 있는 기준이 있어야 한다. 일례로 개발 주기에서의 각 단계별로 이루어지는 소요기간을 들 수 있다. 하지만 개발 주기 안에 각 단계별로 소요기간이 기술되어 있지 못한 경우에는 다른 요소가 필요하다. 따라서 평가 요소들에 대한 객관적인 가중치 산정을 위해 이용할 수 있는 기준들이 준비, 혹은 확보되어야 한다.

3. 소프트웨어 공학 기반의 소프트웨어 개발 진척 판단 방안

기존의 소프트웨어 개발 전주기와 관련한 사례들에서 나타난 개선점 및 위에서 기술된 문제점들을 개선하여 좀 더 객관성을 확보하기 위한 방안에 대해 제시하고자 한다. 소프트웨어 개발 주기와 관련한 문제를 판단하기 위한 수행 프로세스는 4단계를 거치는 것으로 다음 그림 2와 같다.



[Fig. 2] Software Development Progress Judgement Procedure

가장 먼저 수행되는 내용은 평가 항목 도출 단계이다. 현재 다루고 있는 소프트웨어 개발 주기는 기본적으로 소프트웨어를 개발하는데 필요한 전체 과정속에서 단계적으로 추진되어지는 일련의 작업을 의미한다. 단계별 과정 및 이러한 과정에서 발생되는 결과물은 다음 표 1과 같다. 개발 주기와 관련한 문제를 판단하기 위한 평가 항목들은 아래 표에 기술되어 있는 각 단계별 산출물들이 명확하고 객관적인 내용이므로 객관성있는 평가 항목들로 사용될 수 있다.

〈Table 1〉 Development Process Steps and Outputs

Step Name		Output
1	Basic design	Basic Design report
2	Detailed design	Detailed design report
3	Program design	Program design report
4	Implementation	Source program
5	Testing (unit, integration)	Test result report
6	Maintenance	Operation plan report

다음으로 2번째 단계에서는 전 단계에서 결정된 평가 항목에 대하여 가중치를 부여한다. 이 부분에서는 당사자간에 합의된 내용인 계약서에 첨부된 '소프트웨어 개발 계획서'나 혹은 '개발계획 명세서'등과 같은 문건의 내용

을 기반으로 한다. 이러한 문건들에는 보통 소프트웨어 개발 주기에 따른 각 단계별 작업 수행을 위한 개발 소요 기간 및 투입인력에 대한 정보를 기술하고 있다. 따라서 이러한 정보들을 이용하여 각 항목에 대한 가중치를 결정하는데 기본적인 자료로 활용함으로서 문제를 판단하는데 개입될 수 있는 주관적인 사항들을 예방함으로서 보다 객관성을 확보할 수 있다.

먼저 가중치 산정을 위해 소요 기간(투입 시간)을 기반으로 하는 방안은 다음과 같다. 당사자 간 합의된 문건인 계약서등에 프로그램 개발 기간만이 명시되어 있는 경우에는 프로그램 개발에 따른 각 단계별 개발 소요기간을 기준으로 하여 가중치를 설정하며, 이를 기준으로 한 식은 아래 식(1)과 같다.

$$W_i(\%) = \frac{T_i}{\sum_{i=1}^{n} T_i} \quad \cdots\cdots\cdots (1)$$

여기서 W_i는 i번째 평가항목에 대한 가중치를 의미하며, T_i는 i번째 평가항목의 개발 소요 기간을 의미한다. 또한 n은 평가항목의 개수를 의미한다. 식 1의 의미는 프로그램 개발에 따른 전체 소요기간에서 각 항목에 해당하는 기능을 개발하는데 소요되는 기간이 차지하는 비율을 의미한다.

다음으로 투입 인력 정보를 기반으로 하는 가중치 산정 방안은 아래 식(2)로 산정할 수 있다. 당사자 간 합의된 문건인 계약서등에 프로그램 개발 기간은 명시되어 있지 않으나, 기능별 투입인력 정보가 있는 경우 이를 반영한 가중치 산출은 주관성을 배제한 객관적인 내용이 될 수 있다.

$$W_i(\%) = \frac{I_i}{\sum_{i=1}^{n} I_i} \quad \cdots\cdots\cdots (2)$$

여기서 W_i는 i번째 평가항목에 대한 가중치를 의미하며, I_i는 i번째 평가항목의 투입 인력을 의미한다. 또한 n은 평가항목의 개수를 의미한다. 식(2)의 의미는 프로그램 개발에 따른 전체 투입인력에서 각 항목에 해당하는 기능을 개발하는데 투입되는 인력이 차지하는 비율을 의미한다.

각 평가 항목에 대한 중요도를 고려하여 소프트웨어 개발 진척도를 계산하고자 하면 각 항목에 대해 중요도를 부여할 때 주관성이 배제된 객관적인 자료로부터 중요도를 부여하는 것이 중요하다. 이를 위해 소프트웨어 개발시에 나타나는 일반적인 특성을 이용하여 중요도를 부여할 수 있다. 일반적인 특성으로는 소프트웨어 개발시 간단한 작업들에 대해서는 소요 기간 및 투입인력이 상대적으로 적으며, 대부분이 중급이하의 인력에서 행해진다. 반면에 중요하거나 복잡한 기능에 대해서는 소요기간 및 고급인력의 투입이 상대적으로 증가하는 것이 일반적이라 할 수 있다.

소프트웨어 및 프로그램을 개발하는 과정에서는 각 단계에서 나타나는 작업의 특성을 고려하여 투입 인력이 결정하게 된다. 이러한 투입인력에 대한 전문성은 보통 초급, 중급, 고급으로 구분되어 시행되고 있다. 따라서 이러한 인력의 전문성을 고려하여 각 항목에 대한 중요도를 반영하는 것이 필요하다. 예를 들면 초급인력의 중요도는 1, 중급인력은 2, 고급인력은 3으로 인력의 전문성에 따라 중요도를 달리 반영하는 것이다. 하지만 인력의 전문성에 따른 중요도 격차의 적정성은 좀 더 많은 논의가 필요해 보인다. 위에서 기술한 내용을 요약 정리한 한 가지 방안으로는 아래 표 2와 같다.

〈Table 2〉 Importance by Input Manpower

Classification of input personnel	Importance	Remarks
Beginner	1	The importance difference according to the expertise of the input manpower needs to be discussed.
Middle class	2	
Advanced class	3	

3단계로서 각 평가 항목에 대해 작업의 진척도를 판단하는 과정으로, 이 과정에 주관성이 개입될 가능성이 있으므로 가급적 객관적인 자료를 중심으로 진척도를 산출하는 노력이 중요하다. 이러한 점에서 몇 가지 방안을 기술하면 다음과 같다.

기본설계 항목에 대한 진척도 평가는 각 시스템에서 개발하고자 하는 기능들에 대해서 당사자 간에 합의된 요구분석서 혹은 기본 설계서의 자료로 이용하며 개발 시스템이 포함하여야 하는 내용 및 네트워크 구성도, 필요로 하는 데이터베이스 등이 포함되어 있는지를 확인하여 진척도를 산출한다. 상세설계 단계에 대한 진척도는 계획서상의 데이터베이스 개수와 실제 이루어진 DB 설계서상의 DB 테이블 개수를 비교하여 수행한다. 또한 계획서상의 해당 시스템에 포함되어야 하는 프로그램 개수와 실제로 이루어진 프로그램 설계서 및 소스 코드 등의 개수를 참조하여 산정한다. 구현 단계에 대한 진척도 산

정은 시스템에서 포함해야 하는 프로그램의 개수와 실제 프로그램 언어로 작성, 개발된 소스코드 개수를 비교하여 산출한다. 마지막으로 테스트 및 운영과 관련한 진척도는 개발된 프로그램들이 사용자측면에서 활용가능한지에 대해 테스트 작업의 유, 무로 판단하며, 실제 시스템 운영에 필요한 교육 등이 계획대로 시행되었는지를 확인하여 산출한다.

본 연구의 내용이 기존의 방법과 다른 주요 특징은 다음과 같다. 기존의 방법에서 소프트웨어 개발 과정에서 나타나는 단계별 투입 시간 및 인력을 고려함 없이 각 단계의 중요도가 결정되었던 반면에 본 연구에서는 이러한 부분들이 반영되었다는 점이다. 이러한 결과로서 이전의 방법보다는 좀 더 객관적인 자료를 근거로 하기 때문에 문제 발생시 해결방안으로서 신뢰성을 확보할 수 있는 있을 것이다.

4. 결론

4차 산업혁명의 시대가 강조되면서 더욱 관심을 갖는 분야가 정보통신 관련 산업 분야라 할 수 있다. 이러한 정보통신산업 중에서도 기본적인 내용은 소프트웨어라 할 수 있을 것이다. 정부 및 관련기관에서는 소프트웨어 산업 육성을 위해 많은 노력들을 지속적으로 추진하고 있다. 이러한 결과로 국내 관련 산업들은 좀 더 전문성을 지니며 성장하고 있으며, 새로운 기업들이 만들어지고 있다. 반면에 소프트웨어 개발을 위해 많은 시간 및 노력을 기울이지 않고 손쉽게 소프트웨어를 복제 혹은 도용하여 사업을 하는 바람직하지 못한 문제점도 보이고 있다.

본 연구에서는 건전한 소프트웨어 산업 육성에 필요한 소프트웨어 개발 진척과 관련한 문제가 발생시에 문제 해결을 위한 방안으로 좀 더 객관성이 내포된 소프트웨어 공학 관점을 이용한 개발 진척도 문제 해결 방안에 대해 기술하였다. 소프트웨어 개발 프로세스애서 다루어지는 내용도 시간에 따라 점점 다르므로 이에 대한 개선 및 보완이 지속적으로 필요할 것이다.

REFERENCES

[1] R. S. Pressman, Software Engineering - A Practitioner's Approach. 3th edn. McGraw-Hill, Inc: New York, 1992.

[2] I. Sommerville, Software Engineering, Addison-Wesley Publishing Co, 2001.

[3] J. Whitten and L. Bentley, System Analysis & Design methods, IRWIN, 1994.

[4] T. Gildersleeve, Successful Data Processing Systems Analysis, 2nd edn, Englewood Cliffs, New York, Prentice Hall, 1985.

[5] E. Sebastian, The Economic Properties of Software, Jena Economic Research papers, 2008.

[6] https://terms.naver.com/entry.nhn?docId=2073346&cid=44414&categoryId=44414.

[7] https://terms.naver.com/entry.nhn?docId=852368&cid=42346&categoryId=42346.

[8] P. Naur and B. Randel, "SOFTWARE ENGINEERING", Report on a conference sponsored by the NATO SCIENCE COMMITTEE, 1969.

[9] J. C. Satish, Software Documentation Management Issues and practices: A Survey. Indian Journal of Science and Technology. 2016 May, 9 (20), pp. 1-7.

[10] Program Copying Appraisal method, Program Deliberation and Mediation Committee, 2002.

[11] K. Dan, Why Open Source is the Optimum Economic Paradigm for Software. 1999.

[12] Source Code Pragiarism in UK HE computing school, issues, attitudes and tools, Technical report, 2001

[13] F, Culwin. Pragmatic Anti-pragiarism in Action. Proceeding of 3th al Ireland conference on the teaching of computing, Dublin, 1995.

[14] S. P. Trash, N. K. Naresh and V. Shrich. An Empirical Analysis on Reducing Open Source Software Development Tasks using Stack Overflow. Indian Journal of Science and Technology. 9 (21), pp. 1-7. 2016.

[15] W.H.Lee, "Computer Programs and Laws," Binary Publishing Company, 2000.

[16] B. Marr, "Why Everyone Must Get Ready For The 4th Industrial Revolution", Forbes, 2016.

[17] K.M.Cho, "Design and Diagnosis Case of Energy Efficiency Diagnostic Solution based on IoT", Journal of The Korea Internet of Things Society, Vol.6, N0.1, pp.23-30, 2020.

[18] J.H.Hong and K.H.Lee, "A Scheme on Object Tracking Techniques in Multiple CCTV IoT Environments," Journal of The Korea Internet of Things Society, Vol.5, N0.1, 2019.

이 성 훈(Seong-Hoon Lee) [종신회원]

- 1995년 2월 : 고려대학교 일반대학원 컴퓨터학과 (이학석사)
- 1998년 2월 : 고려대학교 일반대학원 컴퓨터학과 (이학박사)
- 1998년 3월 ~ 현재 : 백석대학교 ICT학부 교수

〈관심분야〉
분산시스템, 웹서비스, 지능정보, 컨버전스, 융합산업등

이 동 우(Dong-Woo Lee) [정회원]

- 1984년 8월 : 고려대학교 일반대학원 컴퓨터공학 (공학석사)
- 2005년 2월 : 고려대학교 일반대학원 전산과학과 (이학박사)
- 1995년 3월 ~ 현재 : 우송대학교 컴퓨터정보학과 교수

〈관심분야〉
웹기반분산시스템, 능동시스템, 데이터베이스, 컨버전스등