Journal of the Korea Academia-Industrial Cooperation Society
Vol. 12, No. 11 pp. 5252-5259, 2011

http://dx.doi.org/10.5762/KAIS.2011.12.11.5252

임베디드 소프트웨어를 위한 프레임워크의 재사용성 메트릭에 관한 연구

조은숙[1], 김철진[2*], 이숙희[3]
[1]서일대학 컴퓨터 소프트웨어과, [2]인하공업전문대학 컴퓨터 시스템과
[3]서경대학교 컴퓨터과학과

A Study on Reusability Metric of Framework for Embedded Software

Eun-Sook Cho[1], Chul-Jin Kim[2*] and Sook-Hee Lee[3]
[1]**Dept. of Computer Software, Seoil College**
[2]**Dept. of Computer System, Inha Technical College**
[3]**Dept. of Computer Science, SeoKyeong University**

요 약 임베디드 소프트웨어 분야에서는 최적화와 재사용 기술이 상품의 가치를 좌우하는 핵심 요소기술로 간주되고 있다. 최적화와 재사용 기술의 대표적인 형태가 프레임워크 기술이다. 프레임워크를 기반으로 소프트웨어를 개발할 경우, 소프트웨어의 개발 생산성 뿐만 아니라 재사용성의 효과를 향상시킬 수 있다. 그러나 현재 임베디드 소프트웨어 개발에 있어서는 프레임워크를 적용한 개발 형태가 매우 미흡한 상태이다. 뿐만 아니라 임베디드 소프트웨어 개발을 위한 프레임워크의 개발 또한 시작 단계에 불과해서 개발되는 프레임워크가 과연 기대하는 만큼의 재사용성의 효과를 가져올 수 있는지에 대한 의문점이 존재하게 된다. 본 연구에서는 선행 연구로 임베디드 소프트웨어의 재사용성 향상을 위해 설계한 프레임워크의 재사용성을 측정하기 위한 도구로 재사용성 측정 메트릭을 제안한다. 제안한 메트릭을 실제 설계 사례에 적용한 결과 기존의 설계 방식에 비해 프레임워크 기반의 설계가 재사용성을 보다 향상시키는 결과를 도출할 수 있었다.

Abstract Both Optimization and Reuse Technology are considered as core technologies handling the values of products in embedded software. Framework technology is a typical type of optimization and reuse technology. When we develop software based on framework, The effect of reusability as well as development productivity can be improved. However, currently the form of framework-based development is very poor in embedded software development. Furthermore, because framework development is also beginning stage in embedded software development, there are questions whether developing framework can bring reusability effect. In this paper, we propose metrics measuring reusability of framework which is designed for improving reusability of embedded software. As as result of applying proposed metrics into real design cases, we can obtain more effective results in framework-based design than existing design.

Key Words : Optimization, Reuse Technology, Framework, Embedded Software, Reusability, Metirc

1. 서론

1.1 분석결과

1.1.1 신뢰도

임베디드 소프트웨어는 마이크로프로세서 위에 내장되어 산업 및 군사용 제어기기, 디지털 가전기기, 자동센서 장비 등의 기능을 다양화하고 부가가치를 높이는 핵

본 논문은 2010년 서일대학교 학술연구비에 의해 연구되었음.
*교신저자 : 김철진(cjkim777@gmail.com)
접수일 11년 10월 31일 수정일 11년 11월 09일 게재확정일 11년 11월 10일

심 소프트웨어를 의미하며, 임베디드 운영체제, 임베디드 미들웨어, 임베디드 응용 소프트웨어, 임베디드 소프트웨어 개발 도구 등 다양한 영역들이 여기에 해당된다고 볼 수 있다[1,2,3]. 즉, 임베디드 소프트웨어는 우리가 일상에 쉽게 접하는 휴대폰, TV, 세탁기, 기차, 비행기, 엘리베이터 등의 제품 안에 내장된 임베디드 시스템에서 하드웨어를 제외한 나머지 부분이라고 말할 수 있다. 예를 들어, Smart TV에 내장된 인터넷 접속 기능, 멀티미디어 처리 기능, 전자상거래 기능 등을 제공하는 소프트웨어가 임베디드 소프트웨어를 의미한다. 이러한 임베디드 시스템 가운데 현재 대표적으로 많이 개발되어 있는 형태가 홈 네트워크 시스템이라 할 수 있다. 홈 네트워크 시스템이란 가정 내의 가전, PC, 통신 기기 등의 기기간의 네트워크 연결을 통해서 서로 데이터를 주고 받을 수 있는 시스템을 말한다. 즉, 서로 다른 가전기기를 일괄적으로 관리하고 이를 제어하는 시스템을 의미한다[4].

이러한 임베디드 소프트웨어 분야에서는 최적화와 재사용 기술이 상품의 가치를 좌우하는 핵심 요소기술로 간주되고 있다. 최적화와 재사용 기술의 대표적인 형태가 프레임워크 기술이다. 프레임워크를 기반으로 소프트웨어를 개발할 경우, 소프트웨어의 개발 생산성 뿐만 아니라 재사용성의 효과를 향상시킬 수 있다. 그러나 현재 임베디드 소프트웨어 개발에 있어서는 프레임워크를 적용한 개발 형태가 매우 미흡한 상태이다. 뿐만 아니라 임베디드 소프트웨어 개발을 위한 프레임워크의 개발 또한 시작 단계에 불과해서 개발되는 프레임워크가 과연 기대하는 만큼의 재사용성의 효과를 가져올 수 있는지에 대한 의문점이 존재하게 된다. 본 연구에서는 임베디드 소프트웨어를 위한 프레임워크의 재사용성을 측정할 수 있는 메트릭을 제안하고자 한다. 2장에서는 관련 연구로서 재사용성 측정 관련 메트릭에 대해서 제시한다. 3장에서는 선행된 재사용 프레임워크의 재사용성을 파악하기 위해 고려해야 할 사항들과 임베디스 소프트웨어의 재사용성을 측정하기 위해 필요한 메트릭을 제안한다. 4장에서는 본 논문에서 제시한 재사용성 메트릭을 임베디드 소프트웨어 프레임워크에 적용한 사례와 평가 결과를 제시한다. 마지막으로 5장에서 결론 및 향후 연구과제를 제시한다.

2. 관련 연구

컴포넌트의 재사용성을 측정을 위한 기존 방법들에 대해 살펴봄으로서, 기존 연구에서의 특징과 한계점들을 제시한다.

2.1 경험적 방법에 의한 재사용성 측정

소프트웨어의 재사용성을 측정하기 위한 방법으로 크게 경험적 방법과 정성적 방법으로 구분한다[5,6].

Prieto-Diaz와 Freeman[7]은 재사용성의 측정에 사용되는 다섯 가지 요소들을 나열하였다. 프로그램의 사이즈, 프로그램의 구조, 프로그램의 문서화, 프로그래밍 언어, 그리고 재사용 경험이 소프트웨어의 재사용성의 측정에 사용된다고 보았다. Caldiera와 Basili[8]는 소프트웨어의 재사용성은 소프트웨어의 정확성(Correctness), 가독성(Readability), 테스트가능성(Testability), 수정용이성(Ease of Modification), 그리고 성능(Performance)에 의하여 좌우가 된다고 보았다. 그러나, 이러한 요소들은 직접적인 측정이 어렵기 때문에 그들은 다음과 같은 방법으로 측정을 하였다. Halstead의 프로그램의 크기(Program Volume)를 이용하는 방법, McCabe의 Cyclomatic 복잡도를 이용하는 방법, 정규도(Regularity), 그리고 재사용 빈도수에 의하여 소프트웨어의 재사용성을 측정 하였다[9,10]. 그러나 이러한 방법은 소프트웨어 재사용성에는 쉽게 적용되지만, 단위 컴포넌트의 재사용성이나 프레임워크의 재사용성을 평가하는데 있어서는 컴포넌트와 프레임워크의 특징인 가변성, 인터페이스, 메소드, 속성 등과 같은 요소들이 반영되어 있지 않아서 그대로 적용하는데 한계성이 있다.

2.2 정성적 방법에 의한 재사용성 측정

재사용성은 소프트웨어의 품질 측면에서 상당히 중요한 부분을 차지한다. 따라서, 소프트웨어의 품질을 향상시킬 수 있는 요소들이 역시 재사용성을 향상시킬 수 있는 요소가 될 수 있다[4]. 정성적인 측정 방법은 개인적인 기준에 의해서 측정하는 방법을 말한다. 따라서 측정 결과가 다소 주관적인 결과가 나올 수 있다.

ISO 9126[11]은 소프트웨어의 품질 측정에 관한 모델을 제시하고 있다. ISO 9126은 소프트웨어의 품질을 6개의 특징(Characteristics)으로 나누어서 측정하며, 그 각각의 특징에는 하위 특징(Sub Characteristics)들이 존재한다[12]. 이러한 하위 특징들은 내부 측정기준(Internal Metrics)과 외부 측정기준(External Metrics)으로 측정을 한다. ISO 9126은 소프트웨어의 일반적인 품질 측정에 관한 모델은 제시하고 있지만 컴포넌트와 관련된 특징들에 대해서는 반영하지 못하는 부분들이 존재하기 때문에 그대로 적용하기가 어렵다.

In Guen[13]의 연구에서 제시한 컴포넌트 재사용성 측정 방법은 직접적 측정과 간접적 측정 방법을 제시하였다. 직접적 측정은 컴포넌트를 구성하는 클래스들과 컴포

넌트의 인터페이스들을 기반으로해서 얻는 방법으로서, 컴포넌트의 크기, 복잡도, 결합도, 응집도 등을 측정하였다. 간접적 측정은 직접적 측정기준을 토대로 측정이 되는 방법으로서 컴포넌트의 이해도, 적용가능성, 수정가능성, 모듈화 가능성 등을 측정하였다. 이러한 간접적 측정이 궁극적으로 재사용성 측정을 하는데 사용하였다. 그러나 이 방법에서도 프레임워크의 특징인 가변성을 고려한 확장성이나 이식성에 대한 측정은 제시하지 못했다.

3. 임베디드 소프트웨어 프레임워크의 재사용성 측정 메트릭

이 장에서는 선행 연구된 임베디드 소프트웨어를 위한 재사용 프레임워크[14]의 재사용성을 측정하기 위한 요소와 그러한 요소를 기반으로 재사용성 측정 메트릭을 제시한다.

3.1 재사용성 측정을 위한 5가지 요소

본 논문에서는 컴포넌트의 재사용성을 측정하기 위해 이해도(Understandability), 적응성(Adaptability), 이동성(Portability) 측면에서 메트릭을 제안한다.

3.1.1 이해도(Understandability)

이해도는 새로운 요구사항을 만족시킬 수 있는지를 결정하기 위해 컴포넌트의 기능성을 이해하는 정도를 나타낸다. 이해도는 EMI(Existence of Meta-Information)와 RCO(Rate of Component Observability) 메트릭을 통해 측정될 수 있다. EMI는 컴포넌트의 메타정보가 존재하는지를 측정하기 위한 메트릭이다. 만약 컴포넌트의 메타정보가 제공된다면 사용자는 컴포넌트를 쉽게 이해할 수 있다. RCO는 함수, 입력, 출력에 관련된 정보를 쉽게 파악할 수 있는지를 측정하기 위한 메트릭이다. 블랙박스 컴포넌트는 주로 read 함수로 구성되기 때문에 read 함수의 수에 의해 RCO가 평가된다.

3.1.2 적응성(Adaptability)

적응성은 컴포넌트가 새로운 시스템의 특정 기능적 요구사항에 적응될 수 있는지를 나타낸다. 적응성은 RCC(Rate of Component Customizability) 메트릭을 통해 측정될 수 있다. RCC는 컴포넌트에 변경 가능한 속성의 수에 의해 측정된다. 컴포넌트가 변경 가능한 속성을 많이 포함하고 있다고 해서 커스터마이제이션이 좋다고 할 수 없으며 속성이 많으면 잘못 적용될 가능성도 높다고 할 수 있다. 따라서 RCC는 본 연구의 결과로 산출된 신뢰구간 내에 포함될 때 적응성이 높다고 할 수 있다.

3.1.3 이식성(Portability)

이식성은 컴포넌트가 새로운 환경에 포팅(Porting)될 수 있는지를 나타내는 것으로, 컴포넌트가 사용되는 환경으로부터 얼마나 독립적인지를 측정한다. 이식성은 SCCr(Self-Completeness of Component's Return Value)와 SCCp(Self-Completeness of Component's Parameter) 메트릭을 통해 측정될 수 있다. SCCr은 컴포넌트 내 모든 비즈니스 함수 중에 결과값이 없는 비즈니스 함수의 비율로 측정된다. 즉, 결과값이 없는 함수는 다른 부분과의 의존성을 감소시킨다. SCCp는 컴포넌트 내의 모든 비즈니스 함수 중에 입력값을 가지고 있지 않는 비즈니스 함수의 비율로 측정된다. SCCr과 동일하게 입력값이 없는 함수를 포함할수록 독립성이 강하다고 할 수 있다.

컴포넌트의 재사용성 측정 메트릭인 COR(Component Overall Reusability)은 5가지 메트릭 중에 3가지 메트릭을 이용하여 컴포넌트의 재사용성을 측정한다.

3.2 재사용성 측정 메트릭 정의

정의 1. 이해성(EMI: Existence of Meta-Information)

EMI(c)는 한 컴포넌트 내에 메타 정보의 존재 유무를 의미한다.

$$EMI(c) = \begin{cases} 1 & \text{(Meta Info exists)} \\ 0 & \text{(otherwise)} \end{cases}$$

Confidence Interval: [0.5, 1.0]

정의 2. 컴포넌트 적응성(RCC: Rate of Component Adaptability)

RCC(c)는 한 컴포넌트 내의 속성들 가운데 수정 가능한 속성들의 비율을 의미한다.

$$RCC(c) = \begin{cases} \frac{P_w(c)}{A(c)} & (A(c) > 0) \\ 0 & \text{(otherwise)} \end{cases}$$

where,

$P_w(c)$: number of writable properties in Component c

$A(c)$: number of fields in Component c

Confidence Interval: [0.17, 0.34]

정의 3. 컴포넌트 이식성(SCCr: Rate of Component Portability)

SCCr(c)는 한 컴포넌트 내의 비즈니스 메소드들 가운데 리턴 값이 없는 비즈니스 메소드들 수의 비율을 의미한다.

$$SCC_r(c)=\begin{cases}\frac{B_v(c)}{B(c)} & (B(c)>0)\\ 1 & (\text{otherwise})\end{cases}$$

where,
$B_v(c)$: number of business methods without return value in Component c
$B(c)$: number of business methods in Component c
Confidence Interval : [0.61, 1.0]

정의 4. 컴포넌트 변경성(Changeability)

Changeability(c)는 한 컴포넌트 내의 요구 인터페이스들의 메소드들 가운데 변경 가능한 메소드들의 비율을 의미한다.

$$\text{Changeability}(c)=\begin{cases}\frac{R_c(c)}{R(c)} & (R(c)>0)\\ 0 & (\text{otherwise})\end{cases}$$

where,
$R_c(c)$: number of changeable methods of Component c's required interface
$R(c)$: number of methods in Component c's required interface
Confidence Interval : [0.17, 0.34]

정의 5. 컴포넌트 교체성(Replaceability)

Replaceability(c)는 한 컴포넌트 내의 요구 인터페이스의 메소드들 가운데 교체 가능한 메소드들의 비율을 의미한다.

$$\text{Replaceability}(c)=\begin{cases}\frac{R_r(c)}{R(c)} & (R(c)>0)\\ 0 & (\text{otherwise})\end{cases}$$

where,
$R_r(c)$: number of replaceable methods of Component c's required interface
$R(c)$: number of methods in Component c's required interface
Confidence Interval : [0.17, 0.34]

정의 6. 확장성(Extensibility)

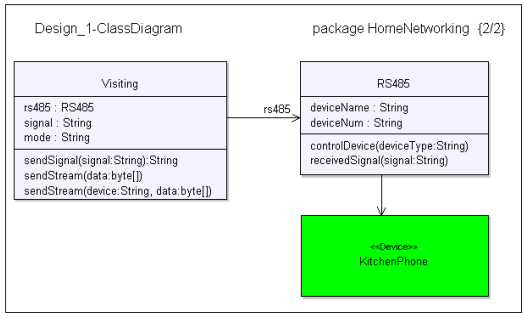
Extensibility(c)는 컴포넌트의 요구 인터페이스의 메소드들 가운데 확장 가능한 메소드들의 비율을 의미한다.

$$\text{Extensibility}(c)=\begin{cases}\frac{R_e(c)}{R(c)} & (R(c)>0)\\ 0 & (\text{otherwise})\end{cases}$$

where,
$R_e(c)$: number of extensible methods of Component c's required interface
$R(c)$: number of methods in Component c's required interface
Confidence Interval : [0.17, 0.34]

4. 사례 연구 및 평가

본 사례 연구에서는 기존의 객체지향 설계 방식과 본 논문에서 제시하는 재사용 프레임워크를 이용한 설계 사례를 제시하여 재사용 평가 메트릭의 이해성(Understand-ability), 적응성(Adaptability), 이식성(Portability), 변경성(Changeability), 교체성(Replaceability), 확장성(Extensibility)을 측정한 결과를 제시한다.



[그림 1] 출입통보 설계 사례 1
[Fig. 1] Case 1 of Entrance Notification

그림 1은 홈 네트워크 솔루션의 출입 방문 시 'Visiting' 클래스와 가전 디바이스 제어 프로토콜인 'RS485' 클래스와 관계를 통해 주방폰에 방문자가 도착했음을 통보한다. 본 사례는 'Visiting' 클래스와 'RS485' 클래스가 직접적으로 결합되어 있다. 본 사례 연구에 대한 재사용 메트릭의 측정 결과는 표 1과 같다.

[표 1] 설계 사례 1의 재사용 측정값
[Table 1] Reusability Measurement of Case 1

Factor	Measured Value
Meta Info	No
Number of Field	5
Number of Writable Properties	3
Number of Business Method	5
Number of Business Method without return value	4
Number of Method of Required Interface	0
Number of Changeable Method of Required Interface	0
Number of Replaceable Method of Required Interface	0
Number of Extensible Method of Required Interface	0

Metric	Measured Value	V_{Metric}	Measured Value
EMI	0	V_{EMI}	0
RCC	3/5=0.6	V_{RCC}	0
SCC_r	4/5=0.8	V_{SCCr}	1
Changeability	0/0=0	$V_{changeability}$	0
Replaceability	0/0=0	$V_{replaceability}$	0
Extensibility	0/0=0	$V_{Extensibility}$	0

$$COR = 1.76 \times \frac{0+0+1+0+0+0}{6} - 1.13 = -0.84$$

이해성을 제공하기 위한 메타정보가 존재하지 않기 때문에 EMI는 0이며 신뢰구간 [0.5, 1.0] 사이에 있지 않으므로 VEMI은 0이다.

적응성을 제공하기 위해 전체 속성 데이터 중에 수정할 수 있는 속성 데이터는 3개이기 때문에 RCC는 0.6이며, 신뢰구간 [0.17, 0.34] 사이에 있지 않으므로 VRCC는 0이다.

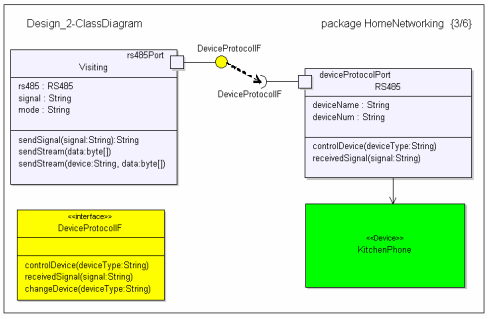
이식성을 제공하기 위해 전체 비즈니스 함수 중에 결과값(Return Value)이 없는 비즈니스 함수는 4개이므로 SCCr는 0.8이며, 신뢰구간 [0.61, 1.0] 사이에 있으므로 VSCCr는 1이다.

변경성을 제공하기 위해 요구 인터페이스의 함수 중에 변경을 위한 함수는 0개이므로 Changeability는 0 이며, 신뢰구간 [0.17, 0.34] 사이에 있지 않으므로 VChangeability는 0 이다.

교체성을 제공하기 위해 요구 인터페이스의 함수 중에 교체를 위한 함수는 0개이므로 Replaceability는 0 이며, 신뢰구간 [0.17, 0.34] 사이에 있지 않으므로 VReplaceability는 0 이다.

확장성을 제공하기 위해 요구 인터페이스의 함수 중에 교체를 위한 함수는 0개이므로 Extensibility는 0 이며, 신뢰구간 [0.17, 0.34] 사이에 있지 않으므로 VExtensibility는 0 이다.

위와 같이 이해성, 적응성, 이식성, 변경성, 교체성, 확장성을 기반으로 재사용 메트릭인 COR는 -0.84 이다. 일반적으로 본 설계는 'Visiting' 클래스와 'RS485' 클래스 간에 강하게 결합되어 있으므로 가변성을 처리하기 위한 미약하며 재사용성이 낮다고 판단할 수 있다.



[그림 2] 출입통보 설계 사례 2
[Fig. 2] Case 2 of Entrance Notification

출입통보 설계 사례 2는 그림 2와 같이 'Visiting' 클래스와 가전 디바이스 제어 프로토콜인 'RS485' 클래스와 직접 연결되지 않으며 인터페이스를 통해 연결한다. 본 사례 연구에 대한 재사용 메트릭의 측정 결과는 표 2와 같다.

[표 2] 설계 사례 2의 재사용 측정값
[Table 2] Reusability Measurement of Case 2

Factor	Measured Value
Meta Info	No
Number of Field	5
Number of Writable Properties	3
Number of Business Method	5
Number of Business Method without return value	4
Number of Method of Required Interface	3
Number of Changeable Method of Required Interface	1
Number of Replaceable Method of Required Interface	0
Number of Extensible Method of Required Interface	0

Metric	Measured Value	V_{Metric}	Measured Value
EMI	0	V_{EMI}	0
RCC	3/5=0.6	V_{RCC}	0
SCC_r	4/5=0.8	V_{SCCr}	1
Changeability	1/3=0.3	$V_{changeability}$	1
Replaceability	0/3=0	$V_{replaceability}$	0
Extensibility	0/3=0	$V_{Extensibility}$	0

$$COR = 1.76 \times \frac{0+0+1+1+0+0}{6} - 1.13 = -0.54$$

본 사례 연구의 설계 사례들은 모두 동일 데이터를 기반으로 하기 때문에 그림 1의 설계 사례1과 같이 이해성, 적응성, 이식성은 동일한 측정 결과를 갖는다.

변경성을 제공하기 위해 요구 인터페이스의 함수 중에 변경을 위한 함수는 1개이므로 Changeability는 0.3 이며, 신뢰구간 [0.17, 0.34] 사이에 있으므로 VChangeability는 1 이다.

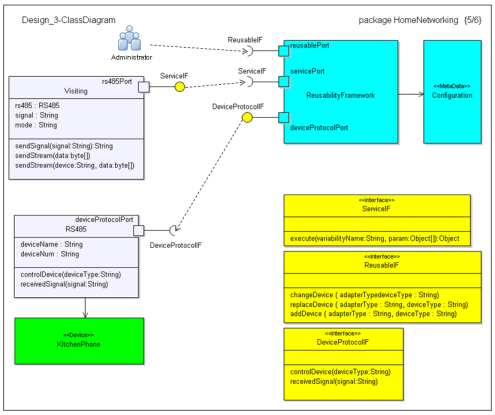
교체성을 제공하기 위해 요구 인터페이스의 함수 중에 교체를 위한 함수는 0개이므로 Replaceability는 0 이며, 신뢰구간 [0.17, 0.34] 사이에 있지 않으므로 VReplaceability는 0 이다.

확장성을 제공하기 위해 요구 인터페이스의 함수 중에 교체를 위한 함수는 0개이므로 Extensibility는 0 이며, 신뢰구간 [0.17, 0.34] 사이에 있지 않으므로 VExtensibility는 0 이다.

위와 같이 이해성, 적응성, 이식성, 변경성, 교체성, 확장성을 기반으로 재사용 메트릭인 COR는 -0.54 이다. 일반적으로 본 설계는 'Visiting' 클래스와 'RS485' 클래스 간에 인터페이스로 결합되어 있으므로 설계 사례 1 보다 재사용성이 높으나 재사용 메트릭 COR은 0 보다 클 경우에만 재사용성이 있다고 할 수 있으므로 일반적으로 재사용하기에는 미흡하다고 할 수 있다.

재사용성 프레임워크를 이용한 설계 사례는 그림 3과 같으며 'Visiting' 클래스와 가전 디바이스 제어 프로토콜인 'RS485' 클래스가 'Reusability Framework'를 통해 연

결된다. ‘Visiting’ 클래스는 디바이스 제어 프로토콜 ‘RS485’ 클래스와 직접 연결되지 않는다.



[그림 3] 프레임워크 기반의 설계 사례 3
[Fig. 3] Case 3 of Design based on Framework

본 설계 사례는 설계 사례 1(그림 1)과 설계 사례 2(그림 2)와 같이 동일 데이터를 기반으로 하기 때문에 이해성, 적응성, 이식성은 동일한 측정 결과를 갖는다.

변경성을 제공하기 위해 요구 인터페이스의 함수 중에 변경을 위한 함수는 1개이므로 Changeability는 0.3 이며, 신뢰구간 [0.17, 0.34] 사이에 있으므로 VChangeability는 1 이다.

[표 3] 프레임워크 기반 설계의 재사용 측정값
[Table 3] Reusability Measurement of Design based on Framework

Factor	Measured Value
Meta Info	Exists
Number of Field	5
Number of Writable Properties	3
Number of Business Method	5
Number of Business Method without return value	4
Number of Method of Required Interface	3
Number of Changeable Method of Required Interface	1
Number of Replaceable Method of Required Interface	1
Number of Extensible Method of Required Interface	1

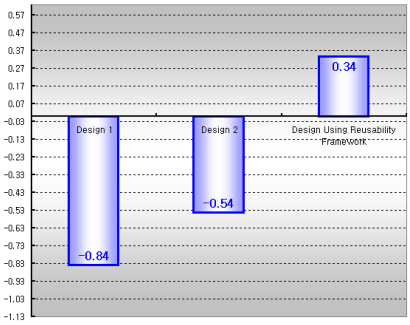
Metric	Measured Value	V_{Metric}	Measured Value
EMI	1	V_{EMI}	1
RCC	3/5=0.6	V_{RCC}	0
SCC_r	4/5=0.8	V_{SCCr}	1
Changeability	1/3=0.3	$V_{changeability}$	1
Replaceability	1/3=0.3	$V_{replaceability}$	1
Extensibility	1/3=0.3	$V_{Extensibility}$	1

$$COR = 1.76 \times \frac{1+0+1+1+1+1}{6} - 1.13 = 0.34$$

교체성을 제공하기 위해 요구 인터페이스의 함수 중에 교체를 위한 함수는 1개이므로 Replaceability는 0.3 이며, 신뢰구간 [0.17, 0.34] 사이에 있으므로 VReplaceability는 1 이다.

확장성을 제공하기 위해 요구 인터페이스의 함수 중에 교체를 위한 함수는 1개이므로 Extensibility는 0.3 이며, 신뢰구간 [0.17, 0.34] 사이에 있지 않으므로 VExtensibility는 1 이다.

위와 같이 이해성, 적응성, 이식성, 변경성, 교체성, 확장성을 기반으로 재사용 메트릭인 COR는 0.34 이다. 일반적으로 본 설계는 ‘Visiting’ 클래스와 ‘RS485’ 클래스가 직접 연결되지 않고 ‘Reusability Framework’를 통해 연결되므로 ‘Visiting’ 클래스는 출입통보를 위해 다양한 디바이스 프로토콜을 제공하거나 다른 기능(다양한 코덱)을 동시에 제공할 수 있다.



[그림 4] 설계 사례별 재사용 측정 결과값
[Fig. 4] Reusability Measurement according to Design Case

그림 4와 같이 기존 설계 방식인 설계 사례 1과 설계 사례 2와 다르게 ‘Reusability Framework’을 이용한 설계 사례는 COR이 0 보다 크므로 재사용성이 있다고 할 수 있다[1]. 구조적 측면에서는 ‘Reusability Framework’이 추가되기 때문에 복잡성이 높아 질 수 있으나 설계 사례 1과 설계 사례 2에 비해 변경성, 교체성, 확장성이 좋아지므로 재사용성이 향상된다고 할 수 있다.

5. 결론 및 향후 연구과제

본 연구를 통해 얻을 수 있는 기대효과는 다음과 같다. 첫째, 임베디드 소프트웨어 개발시 적용할 프레임워크를 선택하는 데 있어서 최적의 프레임워크를 결정할 수 있도록 한다. 둘째, 임베디드 소프트웨어 개발에 적용할 프레임워크의 재사용성에 대한 검증을 지원한다. 셋째, 임

베디드 소프트웨어 개발에 필요한 프레임워크 설계 및 개발에 반영함으로써, 재사용성이 높은 프레임워크를 개발할 수 있도록 한다.

본 연구를 통해 도출된 연구 결과물은 여러 가지 분야에 다음과 같이 활용될 수 있다. 임베디드 소프트웨어 개발 방법론에 적용함으로써, 임베디드 소프트웨어 개발의 생산성 향상을 가져온다. 프레임워크 기반의 임베디드 소프트웨어 개발에 있어서 재사용성의 향상을 도모할 수 있다. 프레임워크 설계 및 개발시 본 연구에서 제안한 측정 메트릭을 적용함으로써 프레임워크의 설계 및 개발을 검증하는데 활용할 수 있다.

References

[1] T. Lewis, "Information appliances: gadget Netopia", IEEE Computer, Vol.31, No.1, pp.59-68, Jan. 1998.

[2] HomePNA, Home Phoneline Networking Alliance: Simple, High-speed Ethernet Technology for Home, A white paper, June 1998.

[3] S. Hong, "Embedded Linux Outlook in the PostPC Industry", IEEE International Symposium on Object-Oriented Real-time distributed Computing, pp. 37-40, May, 2003.

[4] S. Hong, "Coping with Embedded Software Crisis using Real-Time Operating Systems and Embedded Middleware", IEEE Asian Pacific ASIC(AP-ASIC) Conference, pp. 37-40, May, 2003.

[5] Jeffrey S. Poulin, "Measuring Software Reusability," Proceeding of the Third International Conference on Software Reuse, Rio de Janeiro, Brazil, 1-4 November 1994, pp.1-5.

[6] Jeffrey S. P., "Measuring Software Reusability", IEEE Software, 1994.

[7] Prieto-Diaz, Ruben and Peter Freeman, "Classifying Software for Reusability," IEEE Software, Vol. 4, No. 1, January 1987, pp.6-16.

[8] Caldiera, Gianluigi and Victor R. Basili, "Indentifying and Qualifying Reusable Software Components," IEEE Software, Vol. 24, No. 2, Febuary 1991, pp.61-70.

[9] Pressman, R.S., Software Engineering: A Practitioner's Approach, McGraw-Hill, 2002.

[10] STARS, "Repository Guidelines for the Software Technology for Adaptable, Reliable Systems(STARS) Program," CDRL Sequence Number 0460,15 March 1989.

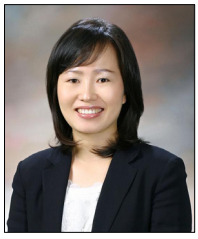
[11] ISO/IEC, FCD 9126-1.2 Information Technology Software product quality-Part 1:Quality model, 1998.

[12] ISO/IEC JTC1/SC7 N2419 "DTR 9126-2: Software Engineering - Product Quality Part 2 - External Metrics", 2001.

[13] In Guen Park, Soo Dong Kim, "Software Component Reusability Metrics", Journal of The Korea Information Science Society: Software and Application, Vol. 31, No. 6, pp.760-772, June 2004.

[14] Chul Jin Kim, Eun Sook Cho, Chee Yang Song, "A Design Technique of Configurable Framework for Home Network Systems", Journal of Korea Academia -Industrial Cooperation Society, Vol. 12, No. 4, pp.1844-866, April 2011.

조 은 숙(Eun-Sook Cho) **[정회원]**



- 1993년 2월 : 동의대학교 전산통계학과(이학사)
- 1996년 2월 : 숭실대학교 대학원 컴퓨터학과(공학석사)
- 2000년 2월 : 숭실대학교 대학원 컴퓨터학과(공학박사)
- 2000년 9월 ~ 2005년 2월 : 동덕여자대학교 강의전임교수
- 2005년 3월 ~ 현재 : 서일대학교 컴퓨터소프트웨어과 부교수

<관심분야>
컴포넌트 기반 개발 방법론, 서비스지향 아키텍처(SOA), 프레임워크 설계 및 개발, 클라우드 컴퓨팅

김 철 진(Chul-Jin Kim) **[정회원]**

- 1996년 2월 : 경기대학교 전자계산학과(학사)
- 1998년 2월 : 숭실대학교 대학원 컴퓨터학과(공학석사)
- 2004년 2월 : 숭실대학교 대학원 컴퓨터학과(공학박사)
- 2004년 3월 : 가톨릭대학교 컴퓨터 정보공학부 강의전담교수
- 2004년 3월 ~ 2009년 2월 : 삼성전자 책임연구원
- 2009년 3월 ~ 현재 : 인하공전 컴퓨터시스템과 조교수

<관심분야>
CBD, Component Customization, Embedded Software

이 숙 희(Sook-Hee Lee) **[정회원]**

- 1979년 2월 : 숙명여자대학교 독문학과(문학사)
- 1982년 2월 : 동국대학교 경영대학원 정보처리학과(경영학석사)
- 1991년 2월 : 성균관대학교 대학원 통계학과(공학박사)
- 1987년 3월 ~ 1993년 2월 : 동신대학교 전자계산학과 교수
- 1993년 3월 ~ 현재 : 서경대학교 인터넷정보학과 교수

<관심분야>
객체지향, 소프트웨어 테스팅, 컴포넌트기반 소프트웨어 공학