DOI: 10.3745/KIPSTD.2011.18D.2.123

이기종 모바일 플랫폼에 애플리케이션 이식성 평가 기법

박 해 윤† · 유 해 영††

요 약

최근 스마트폰과 같은 다양한 애플리케이션을 사용할 수 있는 모바일폰의 유행에 힘입어 모바일 애플리케이션 수요가 폭발적으로 증가하고 있고, 이에 따라 모바일 애플리케이션에서에 품질의 중요성도 점차 커지고 있다. 하지만 모바일 애플리케이션은 다양한 플랫폼이나 모바일폰의 제약사항에 따라 플랫폼이나 디바이스 장치 간의 호환성이 떨어지는 특성을 가짐으로써, 다른 플랫폼에서 애플리케이션을 재사용하기 위해서 해당 플랫폼이나 모바일폰의 제약사항에 따라 다시 개발해야하는 제약을 가진다. 따라서, 본 논문에서는 모바일 애플리케이션을 타 플랫폼에서 재사용하기 위해, 이식할 때 발생할 수 있는 문제를 확인하고 그에 따른 기존 모바일 애플리케이션에 이식의 편이를 확인할 수 있는 이식성 품질 평가 기법을 제안한다. 이에 따라 제안한 이식성의 편이를 나타내는 이식성 점수를 확인함으로써 모바일 애플리케이션이 타 플랫폼에 이식될 때 발생할 수 있는 문제점을 확인하고 이에 따른 재개발 가능 여부와 노력 정도를 확인할 수 있는 근거가 될 수 있을 것으로 기대된다. 또한 이를 모바일 애플리케이션의 설계부터 고려한다면 다양한 모바일 플랫폼에 좀 더 쉽게 이식할 수 있는 품질 좋은 모바일 애플리케이션을 개발할 수 있는 방법이 될 수 있을 것으로 기대된다.

키워드 : 이식성 평가, 이기종 모바일 플랫폼, 모바일 애플리케이션 이식

Portability Evaluation Technique of Application for Heterogeneous Mobile Platform

Park Hae-Yoon† · Yoo Hae-Young††

ABSTRACT

However a mobile application has limited compatibility to other platforms and device. So a mobile application should be rebuilt as particular restrictions of platforms or mobile phones when the mobile application is reused for other platforms. However a mobile application has limited compatibility to other platforms and device. So a mobile application should be rebuilt as particular restrictions of platforms or mobile phones when the mobile application is reused for other platforms. This paper ascertains problems when a mobile application is transplanted for reusing into other platforms, and suggests to evaluation systems of portability's quality that we can check the portability convenience of the existing mobile application. As we confirm its grade that shows convenience of suggested portability, we are able to check problems issued when a mobile application implants to other platforms. Then it is expected that we can check capability of rebuilding and endeavor rate. Also if the method will be considered at the first step of designing a mobile application, it will be the best way to develop the better mobile application that we can easily implant many other mobile platforms.

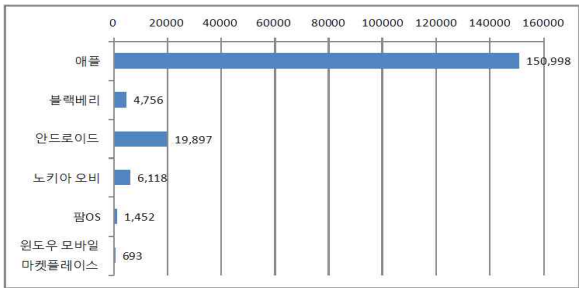
Keywords : Portability Evaluation; Heterogeneous Mobile Platform; Mobile Application Transplant

1. 서 론

최근 스마트폰의 폭발적인 증가에 따라, 모바일 애플리케이션에 대한 관심도 증가하고 있다. 하지만, 모바일 애플리케이션은 플랫폼에 따라 가지게 되는 상이한 여러 가지 특성 때문에 각 플랫폼에 종속적이다[1]. 이에 따라, 각 플랫폼에서 이용할 수 있는 애플리케이션은 상이하며, 이 애플리케이션의 수도 각 플랫폼의 특성이나 이용에 따른 차이를 보인다. 이는 모바일 애플리케이션이 플랫폼에 따라 등록 건수에 차이를 보이는 것에서 확인할 수 있다.

(그림 1)은 2010년 1월을 기준으로 MWC 2010에서 발표된 Distimo[2]의 마켓 별 모바일 애플리케이션 등록건수를 보이고 있다. 여기서 최근, iphone등에서 이용하는 애플社의 iOS 플랫폼 기반에 애플리케이션은 Microsoft社의 윈도우 모바일 플랫폼 기반에 애플리케이션의 217배 이상의 차이를 보인다.

※ 본 과제는 정보통신산업진흥원의 SW공학 요소기술 연구개발사업의 결과물임을 밝힙니다.
† 준 회 원 : 단국대학교 컴퓨터학과 박사과정
†† 정 회 원 : 단국대학교 컴퓨터학부 교수
논문접수 : 2010년 10월 29일
수 정 일 : 1차 2010년 12월 2일, 2차 2010년 12월 17일
심사완료 : 2010년 12월 17일



(그림 1) 애플리케이션 마켓 별 등록건수 비교

또한 모바일 애플리케이션 개발사의 입장에서, 하나의 플랫폼에서 개발된 애플리케이션을 다른 플랫폼에서 출시하기 위해서는 애플리케이션을 재개발해야 하는 문제를 야기한다. 이 때 재개발은 각 플랫폼을 개발할 수 있는 능력을 가진 사람, 각 플랫폼에 의존적인 장비, 재개발에 따른 시간 등 많은 자원을 필요로 한다.

본 논문에서는 모바일 애플리케이션을 타 플랫폼에 이식하기 위해 고려해야 하는 특성들을 확인하고, 이 고려사항을 ISO/IEC 9126 품질 특성의 하나인 이식성(portability)과 매핑 함으로써 모바일 애플리케이션 이식을 위한 이식성 매트릭스를 제안한다. 또한 제안된 매트릭스를 정량적으로 평가하기 위해 이식성 세부 테스트 모듈을 제안한다. 이를 통해, 모바일 애플리케이션을 다른 플랫폼에서 구현 할 때, 발생할 수 있는 문제나 필요한 노력을 확인할 수 있으며, 나아가 재개발을 효율적으로 할 수 있도록 애플리케이션 개발 시 제안된 매트릭스들을 고려함으로써 좀 더 범용적인 애플리케이션을 개발할 수 있을 것으로 기대된다.

본 논문은 다음과 같이 구성된다. 2장에서는 현존하는 모바일 애플리케이션을 평가하는 방법과 이식성 평가들에 대해 조사하고 모바일 애플리케이션의 품질 향상을 위한 연구들을 보이며, 3장에서는 모바일 애플리케이션을 다른 플랫폼에 이식할 때 고려해야 하는 특성을 확인하고, AHP기법을 응용하여 이 고려사항들을 ISO/ICE 9126에 품질특성 중 하나인 이식성에 부특성들과 매핑한다. 이를 기반으로 4장에서 매핑된 이식성 매트릭스들을 정량적으로 평가하기 위한 이식성 세부 테스트 모듈을 제안한다. 5장에서는 제안된 이식성 세부 테스트 모듈이 이식의 편이성을 확인할 수 있음을 보이고, 이를 통해 안드로이드에서 윈도우 모바일로의 이식을 수행한다. 그리고 마지막으로 결론과 향후 연구를 소개한다.

2. 관련연구

ISO/IEC 9126의 품질 모델(Quality Model)은 소프트웨어 품질을 측정·평가하기 위해 소프트웨어의 품질요소와 특성을 정의하고 개발공정에서 품질을 객관적으로 정량화하는데 요구되며, 일반적으로 이러한 품질 모델은 계층구조로 세분화되어 표현된다[3]. ISO/IEC 9126은 (그림 2)와 같이 소프트웨어 품질 특성의 6개 주특성들과 27개의 부특성(subcharacteristics)으로 구성되는 품질모델을 정의한다. 이

(그림 2) ISO/IEC 9126-1에 정의된 주특성과 부특성

품질 모델은 일반적이기 때문에 어떤 소프트웨어 제품에 특정한 용도에 따라 맞춤화하여 적용할 수 있다[4][5].

이 중 이식성이란 한 환경에서 다른 환경으로 전이될 수 있는 소프트웨어 제품의 능력을 말한다[6]. 소프트웨어의 수요가 증가함에 따라 초기에는 소프트웨어 생산성을 향상시키는 최적 방법에 대한 연구가 활발히 진행되었고, 점차 완전히 새로운 프로덕트를 만들기 보다는 현재 가지고 있는 프로덕트를 새로운 컴퓨터에 이식할 수 있다면 노력을 상당히 줄일 수 있기 때문에 재사용 또는 이식에 대한 관심이 고조되었다[7]. 특히 최근 Multi-Vender(다중 사업자)화에 따른 이기종간의 프로그램 호환성 문제가 상당히 부각되어 왔으며, 기존의 프로그램 작성 방법은 프로그램의 기능이나 성능, 품질 등에 중점을 두어 평가함으로써 Multi-Vender화

〈표 1〉 이식성을 측정하기 위한 매트릭스

부특성	매트릭스	계산방법
적응성	호환정보 제공	호환정보에 대한 정보제공 여부를 Yes/No로 측정. 단, Y=1, N=0.
	호환률	호환률(X)=B/A, A는 지원이 명시된 시스템 환경 수, B는 실제 지원 가능한 시스템 환경 수
병행 존재성	공존가능 정보제공	다른 시스템과 동시에 사용할 수 있는지에 대한 정보를 제공하는지 Yes/No로 측정. 단, Y=1, N=0.
	공존 가능률	공존가능률(X) = B/A $B=\sum_{i=1}^{A}\frac{Su\,TC_i}{Total\,TC_i}$ A는 평가할 공존가능한 항목의 수, B는 공존가능 한 항목별 테스트케이스 성공률의 합, Su_TCi는 i번째 성공한 공존가능 항목의 수, TotlaTCi는 i번째 수행한 공존가능 항목의 테스트케이스의 수
대체성	데이터 지속 정보제공	이전버전에서의 데이터를 지속적으로 사용할 수 있는 정보를 제공하는지 Y/N로 측정. 단, Y=1, N=0.
	데이터 지속 가능률	데이터 지속가능률(X) = B/A $B=\sum_{i=1}^{A}\frac{Su\,TC_i}{Total\,TC_i}$ A는 평가할 대체가능한 측정항목 수, B는 각 측정 항목별 테스트케이스 성공률의 합, Su_TCi는 i번째 성공한 대체가능한 항목의 수, Total_TCi는 i번째 수행한 대체가능한 항목의 테스트 케이스의 수
설치성	설치 가능률	설치가능률(X)=B/A, A는 시도한 설치횟수, B는 성공한 시도횟수
	제거 가능률	설치가능률(X)=B/A, A는 시도한 제거횟수, B는 성공한 제거횟수

에 대한 적절한 대응이 없었고, 이런 이유로 프로그램에 이식성이 가미될 수 있는 조건이 필요하게 되었다[8].

이식성을 대한 연구는 국내에서도 활발하게 이루어지고 있다. ISO/IEC 국제표준을 기반으로 하여 국가연구개발사업에 활용할 품질측정모델에 대한 연구에서 이식성을 적응성, 병행존재성, 대체성, 설치성으로 분류하고, 계산은 <표 1>과 같은 방법으로 수행한다[9].

국가연구개발사업의 이식성 품질은 품질특성별 결과값의 합에 의해 결정되며, 전체합의 계산식은 다음과 같다.

$$Q(i) = \frac{\Sigma(MV(i,j))}{N(i)} * W(i) * 100(\%) \qquad \text{(수식 1)}$$
$$Q_t = \Sigma(Q(i))$$

i는 품질특성, j는 품질특성에 포함되어 시험된 매트릭스, MV(i, j)는 품질특성별 매트릭스측정 결과값, N(i)는 품질특성별 시험에 적용된 매트릭스 수, W(i)는 매트릭스별 가중치, Q(i)는 품질특성별 결과값, Qt는 총결과값이다.

ISO/IEC 9126 특성을 직접적으로 참조하는 SW의 이식성 시험 방법에 대한 연구로서, 양해술[10]은 각종 디지털 콘텐츠를 불법복제로부터 보호하는 디지털 저작권 관리 SW의 이식성을 평가하기 위한 시험 매트릭스를 제안했다. 이를 위해 ISO/IEC 9126 이식성의 부특성인 적응성에 권리 표현 지원, 설치가능성에 설치정보 제공, 공존성에 공존 가능 정보 제공, 준수성에 이식 표준 준수 정보 제공을 매트릭스로 정의하고 평가 방법을 제안했다. 이 연구를 통해 기존에 ISO/IEC 9126이나 ISO/IEC 12119기반의 평가에 비해 디지털 저작권 관리 SW의 이식성에 초점을 맞춘 핵심적이고 최적화된 평가를 수행할 수 있다. <표 2>은 이 연구에서 제안한 여러 종류의 이식성 부특성 중 하나인 적응성에 매트릭스로 제안된 권리 표현 지원 매트릭스의 시험 모듈이다.

〈표 2〉 권리 표현 지원 매트릭스의 시험 모듈

적응성	매트릭스명	권리 표현 지원	Right Expression에 관해 콘텐츠의 형태에 따른 다양한 형태의 권한을 제공하는가?
	측정항목	A	콘텐츠 형태에 다른 권리 표현 형태의 수
			문서인 경우 View, Print, 동영상의 경우 Play, SW의 경우 Execute 등
		B	SW에서 제공하는 권리 표현 형태의 수
	계산식		권리 표현 지원 = B/A
	결과영역		0 ≤ 권리 표현 지원 ≤ 1
	적용대상		공통

이 연구에서 측정된 이식성은 디지털 저작권 관리 SW의 이식성에 초점을 맞춰 이에 맞는 최적화된 평가를 수행할 수는 있으나, 모바일 환경이라는 제한된 환경 하에 플랫폼 간의 이식을 고려하기 위해서 이를 적용하기에는 어려움이 있다.

3. 모바일 애플리케이션을 위한 이식성 품질 매트릭스

3.1 모바일 애플리케이션 이식에 고려사항

모바일 애플리케이션을 위한 이식성 품질 평가 모델을 개발하기에 앞서, 모바일 애플리케이션을 다른 모바일 플랫폼에 이식하기 위해 고려해야 하는 특징을 확인해야 한다. 이 절에서는 다음과 같은 고려사항을 도출하였다.

(1) 플랫폼별 개발언어 차이

모바일 애플리케이션들의 개발언어는 플랫폼에 종속적이다. 이런 특성 때문에 플랫폼에 따라 개발언어가 상이할 수밖에 없다. 이는 다양한 플랫폼에서 기존 애플리케이션의 모듈을 재사용할 수 없게 만들며, 재개발을 어렵게 하는 요인이 된다. 이 때문에, 각 언어를 타언어로 변환하는 것에 대한 이식성을 확인하는 것이 필요하다.

(2) 플랫폼에서 지원하는 다양한 API 차이

모바일 애플리케이션을 다른 플랫폼에 이식하기 할 때 가장 어려움을 겪는 부분은 API의 차이이다. A플랫폼의 애플리케이션 개발을 위해 사용되는 API는 B플랫폼에서 다른 API로 대체되어야만 하거나 대체할 수 있는 API가 없다면 다시 구현되어야 한다. 이 때문에 API의 차이를 확인하는 것은 이식을 얼마나 쉽게 할 수 있는가에 대해 영향을 미칠 수밖에 없다. 이런 문제들 때문에, API의 처리 방법에 대한 이식성 문제를 확인할 필요가 있다.

(3) 플랫폼간의 GUI구성의 차이

모바일 디바이스의 디스플레이는 크기가 일반 컴퓨터에 비해 작기 때문에 가시성 및 가독성이 떨어지고 표현할 수 있는 정보의 양이 제한되는 문제를 가진다. 하지만 사용자는 GUI를 통해 애플리케이션과 커뮤니케이션하기 때문에 사용자에게 제일 빠르게 영향을 미치는 부분이라 할 수 있다. 이와 같은 GUI의 고려사항들은 사용자가 직접 경험하는 품질 척도이기 때문에 중요한 테스팅 요소가 된다. 게다가, GUI는 각 플랫폼별로 구성하는 방법이 다르기 때문에 이식 시 반드시 확인해야 하는 고려사항이다. 또한 같은 플랫폼 안에서도 제조사별로 GUI를 다르게 구성할 수 있기 때문에, GUI구성은 재개발하는데 반드시 확인되어야 한다. 이 때문에, GUI를 구성하는 것에 대한 이식성을 확인하는 것이 필요하다.

(4) 사용자 인터렉션에 대한 처리

모바일 디바이스에 종속적인 하드웨어 버튼은 모바일 애플리케이션에도 영향을 미친다. 다양한 인터렉션은 모바일 디바이스에 종속적이지만, 모바일 애플리케이션에 큰 영향을 미친다. 이 때문에 애플리케이션을 이식하는데 있어서

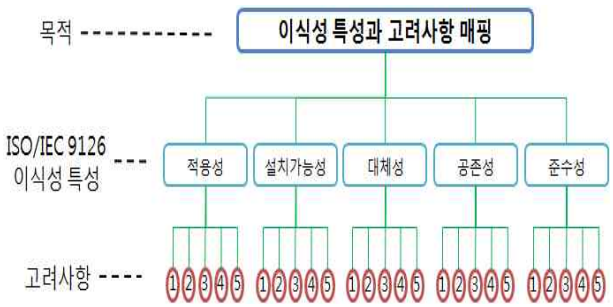
모바일 애플리케이션이 각각의 인터렉션을 어떻게 공유하는지에 대한 지원방법도 고려할 필요가 있다.

(5) 다른 애플리케이션과의 연계 처리

최근 애플리케이션들은 독립적으로 실행되기 보다는 다른 애플리케이션과의 연계를 통해 동작되기도 한다. 이때 연계된 다른 애플리케이션과의 상관관계를 파악하여 이에 해당하는 상관관계를 이식되는 애플리케이션에서도 만들어 줄 필요가 있다. 이 때문에 애플리케이션을 이식하는데 있어서 다른 애플리케이션과의 연계성도 중요한 이식성의 이슈가 된다.

3.2 모바일 애플리케이션의 이식을 위한 고려사항과 ISO/IEC 9126 이식성의 부특성 매핑

위 3.1절에서 고려한 모바일 애플리케이션 이식을 위한 고려사항을 위에 ISO/IEC 9126의 이식성 부특성에 매핑하기 위해 다기준(Multiple-criteria) 의사결정(Decision-making) 문제에 대안을 선택하는 AHP(Analytic Hierarchy Process)[11]에서의 상대비중을 응용하여 매핑하려 한다. 이를 위해 먼저 다음 (그림 3)과 같이 이식성 특성과 고려사항을 매핑하기 위한 계층적 구조를 정의하였다.



(그림 3) 매핑을 위한 계층적 구조

이 계층적 구조를 통해 ISO/IEC 9126의 이식성 부특성들을 기준으로 모바일 애플리케이션을 이식하기 위한 고려사항들을 쌍대비교 함으로써, 상대적 비중이 높은 고려사항을 부특성과 매칭시켜 모바일 애플리케이션 이식을 위한 이식성 매트릭스로 구성하고자 한다. 이를 위해 현재 아이폰 애플리케이션을 개발 완료하고, 이를 안드로이드 플랫폼과 윈도우 모바일 플랫폼으로 각각 이식하고 있는 모바일 애플리케이션 개발자 10명을 대상으로 인터뷰를 통해 쌍대비교를 수행하였다.

쌍대비교를 다섯 가지 부특성을 기준으로 각각 수행하고, 이를 통해 각 고려사항에 가중치를 산정하였다. 그리고 이를 <표 3>와 같이 정리하였다. 또한 AHP기법에서 인터뷰 대상자의 답변에 불일치성에 대한 검증을 위해 수행하는 일치율(Consistency Ratio) 평가를 (수식 2)을 이용하여 확인하였다.

〈표 3〉 부특성별 고려사항의 가중치와 일치율

	적응성	설치 가능성	대체성	공존성	준수성
①	0.061	**0.434**	0.196	0.090	0.200
②	**0.326**	0.193	0.087	0.074	0.200
③	**0.326**	0.191	0.127	0.127	0.200
④	0.180	0.096	0.118	**0.477**	0.200
⑤	0.107	0.086	**0.472**	0.231	0.200
일치율	0.0034	0.0100	0.0718	0.0469	0

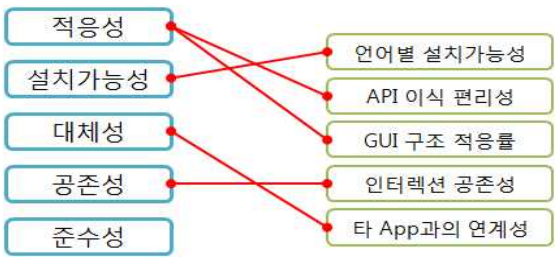
$$\text{Consistency Index} = \frac{\lambda\max - n}{n-1}$$

Random Consistency Index = 1.12 (n=5)

$$\text{Consistency Ratio} = \frac{C.I}{R.C.I} \qquad \text{(수식 2)}$$

이때 일치율(C.R)이 0.1 이내일 경우는 일반적으로 받아들여 지지만 0.2이상일 경우는 자료입력을 다시 할 것을 권하고 있다[12].

산정된 가중치에 따라, 고려사항별로 가장 높은 가중치를 보이는 5가지 점수를 기준으로 ISO/IEC 9126 이식성 부특성과 모바일 애플리케이션 이식 시 고려할 사항을 매핑하도록 한다. 매핑된 결과는 (그림 4)와 같다.



(그림 4) 이식성 부특성과 고려사항 매핑

이 중 준수성에 대해 고려사항이 매핑되지 못한 것은 쌍대비교시 준수성을 기준으로 한 조사에서 모든 고려사항이 동일하게 준수성을 같은 중요도로 판단하여야 한다고 조사되었기 때문에 가중치가 다른 부특성에 비해 낮아졌기 때문이다.

4. 모바일 애플리케이션의 이식성 테스트 모듈

본 장에서는 모바일 애플리케이션의 이식성 테스트를 위해 3.2절에서 매핑된 모바일 애플리케이션 이식을 위한 매트릭스를 기준으로, 모바일 애플리케이션의 이식성을 정량적으로 평가할 수 있는 모바일 애플리케이션 이식성 세부 테스트 모듈을 다음과 같이 설계하였다.

4.1 API 이식 편리성 매트릭스

대상 모바일 애플리케이션에서 API 이식 편리성을 확인하기 위해서는 소스 코드에서 각 API가 대응할 수 있는 API가 있는지를 확인하여야 한다. 이를 위해 API 이식 편리성 매트릭스를 측정할 수 있는 테스트 모듈에서는 <표 4>과 같이 사용되는 API 총 개수를 확인하고, 이를 대체 가능한 API의 개수로 나누어 이식 편리성을 산정한다.

〈표 4〉 API 이식 편리성 매트릭스의 테스트 모듈

적응성	매트릭스명	API 이식 편리성	애플리케이션 구현에 사용된 API와 같은 기능을 하는 API의 존재 유무 확인을 통해 API를 쉽게 구현 할 수 있는 능력
	측정항목		대체 가능한 API 존재 여부
		A	- 각 API별로 타겟 플랫폼에 이식가능한 API가 존재하는지 확인
		B	사용되는 API 총 갯수
	계산식		- API 이식 편리성 = $\sum_{i=1}^{B} \frac{A_i}{B}$ Ai = Y(1) or N(0)
	결과영역		0 ≤ API 이식 편리성 ≤ 1
	적용대상		공통

4.2 GUI 구조 적응률 매트릭스

대상 모바일 애플리케이션에서 GUI 구조 적응률을 확인하기 위해서는 대상 애플리케이션의 GUI가 타겟 기기에서 얼마나 잘 적응할 수 있는지를 확인해야 한다. 이를 위해서 GUI 구조 적응률 매트릭스를 측정할 수 있는 테스트 모듈에서는 다음 <표 5>와 같이 먼저 레이아웃을 표시할 수 있는 영역의 크기를 측정하여 레이아웃 표현 가능 정도를 산정하고, 이 영역 안에 표시되는 사용자 컨트롤이 해당 타겟에서 얼마나 잘 표현될 수 있는지를 측정하며, 마지막으로 사용된 콘텐츠가 타겟 기기에서도 사용할 수 있는 지를 확인해 적응률을 산정한다.

〈표 5〉 GUI 구조 적응률 매트릭스의 테스트 모듈

적응성	매트릭스명	GUI 구조 적응률	GUI가 이식되는 플랫폼에 적응할 수 있는 능력
	측정항목	A	레이아웃 표현 가능 정도
			- 레이아웃 크기 비교를 통해 표현 가능 정도를 확인
		B	사용자 컨트롤 사용 가능 정도
			- 사용자 컨트롤(입력박스, 버튼등)의 사용성 테스트를 통해 사용 가능 정도 확인
		C	사용된 콘텐츠 표현 정도
			- 텍스트, 이미지, 소리등의 콘텐츠의 원본 애플리케이션과의 비교를 통해 표현 가능 정도를 확인
	계산식		- GUI 구조 적응률 = (A + B + C) / 3 - 원본 레이아웃 크기 >= 대상 레이아웃 크기 A = 1-(표현 불가능 영역 넓이 비율) - 원본 레이아웃 크기 >= 대상 레이아웃 크기 A = 1-(낭비되는 영역 넓이 비율) - B = 사용가능한사용자컨트롤갯수 / 사용자컨트롤갯수 - C = 표현가능한콘텐츠갯수 / 표현될콘텐츠총갯수
	결과영역		0 ≤ GUI 구조 적응률 ≤ 1
	적용대상		공통

4.3 언어별 설치가능성 매트릭스

대상 모바일 애플리케이션에서 언어별 설치가능성을 확인하기 위해서는 대상 애플리케이션의 소스코드에서 타겟 플랫폼으로 바로 이식되어 설치 가능한지를 확인해야 한다. 타겟 모바일 플랫폼에서 수정없이 바로 사용될 수 있는 라인이 많다는 것은 언어에 차이에 영향을 받지 않는 라인이 많다는 것으로써, 좀 더 편리한 이식을 수행할 수 있기 때문에 이식성 평가에 있어서 중요하다. 이를 위해서 언어별 설치가능성 매트릭스를 측정할 수 있는 테스트 모듈에서는 다음 <표 6>과 같이 전체 라인 중 타겟 플랫폼에서 그대로 사용할 수 있는, 수정이 필요하지 않은 라인의 수를 확인하여 산정한다.

〈표 6〉 언어별 설치가능성 매트릭스의 테스트 모듈

설치가능성	매트릭스명	언어별 설치가능성	개발언어의 차이에도 불구하고 수정없이 설치될 수 있는 능력
	측정항목	A	수정이 필요하지 않은 라인수
			- 개발 언어의 상이에 따라 영향을 받지 않는 라인의 수
		B	전체 라인 수
	계산식		- 언어별 설치가능성 = A/B
	결과영역		0 ≤ 언어별 설치가능성 ≤ 1
	적용대상		공통

4.4 타 애플리케이션과의 연계성 매트릭스

대상 모바일 애플리케이션에서 타 애플리케이션과의 연계성을 확인하기 위해서는 타겟 환경에서도 연계되어 있는 애플리케이션이 존재하고 제대로 기능하는지를 확인할 필요가 있다. 이를 위해서 타 애플리케이션과의 연계성 매트릭스를 측정할 수 있는 테스트 모듈에서는 다음 <표 7>와 같이 먼저 대상 애플리케이션과 연계되는 타 애플리케이션의 존재의 개수를 확인하고, 타겟 환경에서 연계되는 타 애플리케이션을 보유하는지와 동일한 기능을 하는지를 확인함으로써 산정한다.

〈표 7〉 타 App과의 연계성 매트릭스의 테스트 모듈

대체성	매트릭스명	타 App과의 연계성	원본 애플리케이션과 타 애플리케이션과의 연계를 타겟 플랫폼에서 정확하게 대체하는 능력
	측정항목	A	각 연계점에 대한 대응 여부
			- 다른 애플리케이션과 연결되는 부분 확인 후 이에 대응하는 애플리케이션 확인
		B	원본과 타겟의 연계 App간의 기능 동일 여부
			- 연계 app간의 블랙박스 테스팅을 통하여 애플리케이션 기능 동일성 확인
		C	전체 타 애플리케이션과의 연계점
	계산식		- 타 App과의 연계성 = $\left\{(\sum_{i=1}^{C}\frac{A_i}{C})+(\sum_{i=1}^{C}\frac{B_i}{C})\right\}/2$ Ai = Y(1) or N(0), Bi = Y(1) or N(0)
	결과영역		0 ≤ 타 App과의 연계성 ≤ 1
	적용대상		공통

4.5 인터렉션 공존성 매트릭스

대상 모바일 애플리케이션에서 인터렉션 공존성을 확인하기 위해서는 대상 모바일 애플리케이션의 플랫폼에서 사용할 수 있는 인터렉션이 타겟 환경에서도 애플리케이션과 인터렉션들이 충돌 없이 공존할 수 있는지 확인할 필요가 있다. 이를 위해서 인터렉션 공존성 매트릭스를 측정할 수 있는 테스트 모듈에서는 다음 <표 8>와 같이 먼저 대상 모바일 애플리케이션에서 사용할 수 있는 인터렉션의 개수를 확인하고, 이를 타겟 모바일 환경에서 각각의 사용 가능 여부를 확인함으로써 산정한다.

〈표 8〉 인터렉션 공존성 매트릭스의 테스트 모듈

공존성	매트릭스명	인터렉션 공존성	사용자 인터렉션과 애플리케이션이 충돌 없이 동시에 공존할 수 있는 능력
	측정항목	A	타겟 환경에서 사용할 수 있는 인터렉션 개수
			- 타겟 모바일 애플리케이션에서 사용할 수 있는 인터렉션의 총 개수를 확인 (예) 확인버튼, 1번키, A키, 네비게이션키 등
		B	대상 환경에서 사용할 수 있는 인터렉션 개수
			- 이식 대상 모바일 애플리케이션에서 발생 가능한 인터렉션의 총 개수를 확인
	계산식		- 인터렉션 공존성 = $\sum_{i=1}^{B}\frac{A_i}{B}$ Ai = Y(1) or N(0)
	결과영역		0 ≤ 인터렉션 공존성 ≤ 1
	적용대상		공통

이와 같이 이식성 테스트 세부 모듈은 각 고려사항을 통해 이식성을 테스트할 수 있는 방법을 매트릭스명, 측정항목, 계산식, 결과영역으로 나누어 정리한다.

5. 모바일 애플리케이션 이식 구현과 모듈 검증

이식성 세부 테스트 모듈을 통해 대상 모바일 애플리케이션의 이식성을 확인함으로써 모바일 애플리케이션의 이식성을 확인할 수 있음을 확인하기 위해 다음 (그림 5)과 같은 테스트 방법을 수립하였다.

(그림 5) 테스트 모듈의 이식성 검증 방법

이와 같은 방법을 통해 테스트 모듈로 산정된 이식성 점수가 다른 애플리케이션을 비교하여 구현함으로써 4장에서 정의한 모바일 애플리케이션 이식성 세부 테스트 모듈에서 산정된 점수에 따라 애플리케이션 이식의 용이함을 확인하였다.

5.1 애플리케이션 이식 환경구성

이식성 세부 테스트 모듈을 검증하기 위해서 본 절에서는 소스코드가 공개된 안드로이드 애플리케이션을 대상으로 하여 이를 윈도우 모바일 애플리케이션으로 이식하려 한다. 이를 위해 다음과 같은 환경을 구성하였다.

(1) 안드로이드 애플리케이션 구동환경(이식 대상)
- Java SDK : Java Development kit(JDK) 6
- IDE : Eclipse IDE for Java EE Developers ver. Galileo.
- SDK : Android SDK R05

(2) 윈도우 모바일 애플리케이션 구동환경(이식 타겟)
- IDE : Microsoft Visual studio 2005
- SDK : Windows Mobile 6.5 Professional Edition SDK

5.2 애플리케이션 이식 대상 선정

실험을 위해 선전된 안드로이드 플랫폼을 기반으로 한 애플리케이션은 오픈 API를 기반으로 하고, 개발자 사이트에서 연구나 학습을 위한 오픈소스 애플리케이션을 공개하고 있다. 이 애플리케이션 중에 선정된 안드로이드 애플리케이션은 개발을 용이하게 하고 다른 애플리케이션의 기반이 될 수 있는 예제용 애플리케이션을 대상으로 하였다. 특히, 이식성을 좀 더 객관적으로 비교하기 위해 비슷한 라인 수와 API 수를 가진 5개의 애플리케이션을 선정하였으며, 선정된 애플리케이션은 다음 <표 9>와 같다.

〈표 9〉 애플리케이션 이식 대상

구분	애플리케이션 명	설명
A	GesturesMaker	제스처를 사용하여 해당하는 제스처명을 표시. 제스처를 사용하는 애플리케이션의 구현 예로 사용.
B	SecureKeyborad	핸드폰 키패드를 디스플레이 하여 비밀번호를 생성하고, 입력하는 키보드를 표시. 비밀번호를 입력하는 구현 예로 사용.
C	ImageEditor	이미지를 불러들여, 간단한 수정을 할 수 있도록 패널 표시. 기본적인 이미지 에디터 구현 예.
D	XMLParser	선택된 XML문서를 분석하여 부분으로 분리하여 출력. XML Parser에 대한 구현 예로 사용.
E	TwitterTest	트위터에 username과 password 입력을 통해 트위터에 접근하여 글 입력이 가능. 트위터 관련 애플리케이션 개발에 구현 예.

이와 같이 비슷한 모바일 애플리케이션의 비교를 통해 이식성 테스트 모듈이 이식성의 편이성을 좀 더 객관적으로 확인할 수 있을 것으로 기대된다.

5.3 테스트 모듈을 이용한 이식성 점수 산정

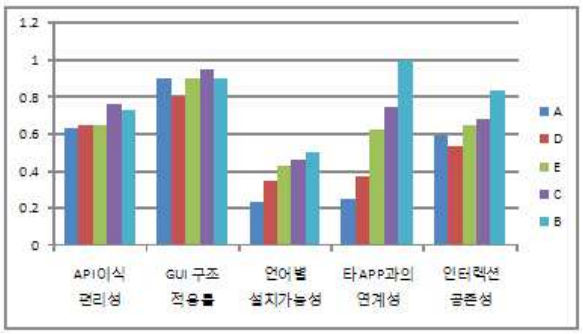
애플리케이션의 이식성 점수를 산정하기 위해 이를 윈도우 모바일 애플리케이션으로 이식한다는 가정 하에 4장에서 정의한 이식성 테스트 세부 모듈을 측정항목으로 정리하여 A 애플리케이션에 적용한 후, 다음 <표 10>과 같이 정리하였다.

〈표 10〉 A애플리케이션의 이식성 테스트 모듈 측정결과

이식성 부특성	세부 모듈명	개수	점수
기본 특성	총 파일 개수	3	
	총 라인 수	108	
API 이식 편리성	윈도우 모바일 API로 대체될 수 있는 API 개수	81	0.63
	사용되는 API 총 갯수	128	
GUI 구조 적응률	레이아웃 표현 가능 정도	1	0.9
	사용자 컨트롤 사용 가능 정도	48/68	
	사용된 콘텐츠 표현 정도	11/11	
언어별 설치가능성	수정이 필요하지 않은 라인 수	25	0.23
	전체 라인 수	108	
타 App과의 연계성	각 연계점에 대한 대응 여부	1	0.25
	원본의 연계app과 타겟의 연계app의 기능 동일 여부	1	
	전체 타 애플리케이션과의 연계점	4	
인터렉션 공존성	원본 기기에서 사용할 수 있는 인터렉션 항목 수	17	0.59
	대상 기기에서 사용할 수 있는 인터렉션 항목 수	29	

- 이식성 점수 = API이식 편리성 점수 + GUI구조 적응률 점수 + 언어별 설치가능성 점수 + 타 App과의 연계성 점수 + 인터렉션 공존성 점수
  = 0.63 + 0.9 + 0.23 + 0.25 + 0.59
  = 2.6

이와 같은 방법으로 5.1절에서 선정한 A, B, C, D, E 5가지의 애플리케이션에서 이식성 점수를 산정하였다. 이식성 점수는 점수에 따라 의미를 가지는 것이 아닌 기준이나 다른 애플리케이션의 이식성 점수에 대한 상대적인 평가이므로, 총 이식성 점수는 각 세부 모듈에서 산정된 이식성 점수의 총합으로 평가한다. 각각의 세부모듈별로 측정된 애플리케이션의 이식성 점수를 그래프로 나타나면 다음 (그림 6)과 같다.



(그림 6) 각 애플리케이션에 세부모듈별 비교

이는 산정된 이식성 점수와 비슷한 추세를 보인다. 단, A와 D 애플리케이션은 몇몇 세부 모듈에서 이식성 점수와 반대되는 결과를 보이는데, 이는 이식성 점수가 큰 차이를 보이지 않기 때문이다.

5.4 수정된 개수와의 비교를 통한 실제 수정과의 연관성 분석

모바일 애플리케이션이 수작업으로 이식되어 개발 시간이나 개발 노력을 확인한다면, 개발자의 능력과 두 플랫폼의 이해정도에 따라 정확하지 않은 결과를 얻을 수 있기 때문에 좀 더 객관적인 비교를 위해 자동화된 변환 툴을 이용하여 이식에 대한 편이성을 확인하는 것이 필요하다. 하지만, 현재 안드로이드 애플리케이션을 윈도우 모바일 애플리케이션으로 이식할 수 있는 자동화된 툴은 존재하지 않는다. 이 때문에 여기에서 사용되는 툴은 모바일 애플리케이션이나 플랫폼을 대상으로 하고 있지는 않지만, 개발언어가 되는 Java와 C#간의 변환을 자동화하는 툴인 Microsoft Java Language Conversion Assistant(JLCA : Java 언어 변환 도구) 3.0을 사용한다. 이 툴은 J2EE 코드를 C# 코드로 변환하는데 유용하게 사용될 수 있다. 하지만, Android는 다른 J2EE와는 다른 SDK를 사용하므로 이 툴을 이용하여도

Android를 Windows Mobile에서 사용할 수 있는 소스로 변환할 수는 없다. 하지만, 이 툴은 변환 보고서를 제공함으로 변환될 수 없는 부분을 확인할 수 있고, 어떤 부분에 수정을 해야 하는지에 대한 정보를 제공한다.

다른 제약으로 안드로이드 애플리케이션 프로젝트의 구성과 윈도우 모바일 애플리케이션 프로젝트의 구성은 다르다. 안드로이드 애플리케이션은 구현부분이 Java파일로 구현되는 것에 반해 애플리케이션의 레이아웃을 XML로 작성하며, AndroidManifest.xml이라는 파일을 통해 대상 애플리케이션 자체뿐만 아니라 애플리케이션에 포함된 컴포넌트(액티비티나 서비스 등)에 어떤 것이 있는지를 정의한다[13]. 하지만, 윈도우 모바일 애플리케이션은 레이아웃도 CS파일로 관리되며, AndroidManifest.xml와 비슷한 역할을 하는 AssemblyInfo.cs파일 또한 CS파일로 관리한다. 이런 프로젝트 구조의 상이는 이식 편이를 확인하는데 제약사항이 될 수 있다. 따라서 수작업으로 구현해야만 하는 레이아웃과 관련된 파일이나 구현시 동적으로 생성되며 프로젝트를 정의하고 환경을 설정하는 파일(AndroidManifest.xml, AssemblyInfo.cs)은 JLCA에 의해 좀 더 객관적으로 이식성을 평가하려는 목적과는 반대로 개발자의 주관적인 능력이나 구현되는 환경에 의해 영향을 받기 때문에 이식성 평가에서 제외한다. 만약 이를 JLCA를 통해 수정하려 할 경우, 파일 내의 대부분이 구문 인식 오류로 인해 변환을 수행하지 못하며, 이를 수작업으로 변환하는 노력은 더욱 커지게 된다. 이는 이식성 평가에 영향을 끼칠 수 있다. 이 때문에 본 절에서는 플랫폼에 의존적인 파일을 제외한 애플리케이션의 알고리즘과 구현부분만을 분리하여 다음과 같이 JLCA를 이용한 변환을 수행하였다.

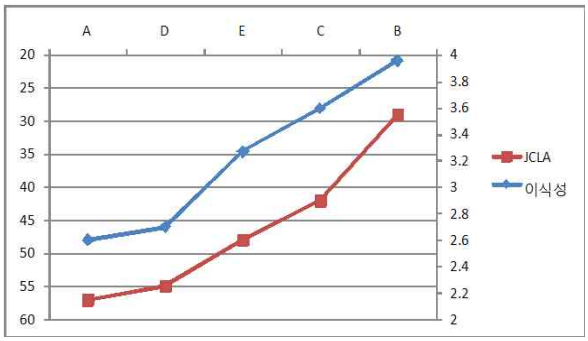
(그림 7) A 애플리케이션에 대한 JLCA변환 보고서

JLCA 변환에서의 문제점은 이식시 확인해야할 문제점이며, 수정할 부분이 되기 때문에, 이런 문제점이 많다는 것은 이식이 쉽지 않다는 것이고, 문제점이 적다는 것은 이식이 좀 더 쉽다는 것을 보인다. 이 때문에, 앞 절에서 선정한 5가지의 애플리케이션의 이식 편이를 확인하기 위해 JLCA의 변환 결과를 비교할 수 있다. 산정된 애플리케이션의 이식성 점수와 JCLA 변환 보고서 상에 수정필요 개수를 비교한 수치는 다음 <표 11>과 같다.

〈표 11〉 애플리케이션들의 이식성 점수와 수정필요 개수 비교

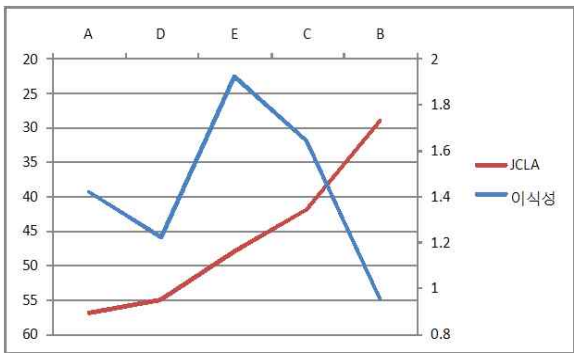
	테스트 모듈로 산정된 이식성 점수	JCLA 변환 보고서 상에 수정필요 개수
A	2.6 점	57 개
B	3.96 점	29 개
C	3.6 점	42 개
D	2.7 점	55 개
E	3.27 점	48 개

이 표의 연관성을 파악하기 위해 그래프로 표현하면 이식성 점수와 수정필요 개수의 연관성은 다음 (그림 8)과 같이 좀 더 확연히 볼 수 있다.



(그림 8) 이식성 점수와 수정 필요 개수의 연관성

위에서 보이는 것처럼, 테스트 모듈에 의해 산정된 이식성 점수와 수정이 필요한 부분에 대한 개수는 비슷한 추세를 보이며, 이는 이식성 점수가 높을수록 수정이 적게 일어나고, 측정된 이식성 점수가 높을수록 이식이 편리하다는 것을 보인다.



(그림 9) 기존 이식성 평가 기법과 수정 필요 개수 비교

특히 기존 품질측정모델에 관한 연구에 의해 산정된 이식성 점수와 수정 개수를 비교한 (그림 9)를 보면, 기존의 연

구에 경우 모바일 특성이 고려되지 않았기 때문에 수정이 필요한 부분의 개수와는 큰 차이를 보이는 것을 볼 수 있다.

이에 따라 모바일 애플리케이션이 타 모바일 플랫폼으로 이식될 때에는 기존 품질측정모델에 이식성 측정보다는 상이한 모바일 플랫폼에서 모바일 애플리케이션을 이식하는 것을 고려한 테스트 모듈에 의해 산정된 이식성 점수 측정이 좀 더 효과적인 이식 편의성을 보일 수 있음을 확인할 수 있다.

5.5 모바일 애플리케이션 이식

모바일 애플리케이션의 타 플랫폼으로의 이식은 현재까지는 자동화된 툴이 존재하지 않기 때문에 수동적인 작업을 통해 이식되어야 한다. 하지만, 본 논문에서는 이식 편이성에 객관적인 비교를 위해 JLCA를 이용하여 JAVA 기반에 안드로이드를 C# 기반의 윈도우 모바일 애플리케이션으로 부분변환 하였다. 그리고 모바일 애플리케이션을 위한 자동화 툴이 아니기 때문에 수동적인 변환 노력이 필요했다.

이식성의 편이성을 위해 이식성 점수를 확인하고 JLCA의 수정개수를 비교하기 위해 선정된 모바일 애플리케이션의 이식된 결과는 다음과 같다.

(그림 10) A 애플리케이션의 이식결과

6. 결론 및 향후 연구

이식성 평가를 위해, 모바일 애플리케이션을 이식하기 위한 고려사항을 확인하고, API 이식 편리성, GUI 구조 적응률, 언어별 설치가능성, 타 App과의 연계성, 인터렉션 공존성이라는 매트릭스를 확인하고 정량적으로 평가하기 위한 테스트 세부 모듈을 매트릭스명, 측정항목, 계산식, 결과영역으로 정리하였다. 또한 이를 검증하는 과정에서 이식성 점수에 따라 수정이 필요한 부분에 대한 차이가 있었으며, 이는 이식성 세부 테스트 모듈이 이식성의 편이를 확인하는데 기여할 수 있음을 볼 수 있었다.

개발자의 능력에 따라 주관적인 평가를 가질 수밖에 없는 한계를 가진 수동적인 이식과정은 개발자의 능력에 의존할 수 있기 때문에, 이를 좀 더 자동화 할 수 있고, 그에 따라 정확한 이식 규모를 확인할 수 있는 연구는 향후 연구되어야 할 과제라 할 수 있다.

참 고 문 헌

[1] Sangwhan Cha, Kurz, J. Bernd, Weichang Du, “Toward a unified framework for mobile applications”, Communication Networks and Services Research Conference, pp.209-216, 2009.

[2] Distimo, "Distimo Rerote", Distimo company, Mobile World Congress 2010 Presentation, 2010.

[3] 오영배, "소프트웨어 제품 품질평가", TTA Journal, 표준기술동향, No.105, 2006.

[4] Ho-Won Jung, Seung-Gweon Kim, Chang-Shin Chung, "Measuring software product quality: a survey of ISO/IEC 9126", Software, IEEE, Vol.21, Issue 5, 2004.

[5] Roger S. Pressman, “Software engineering - A Practitioner's Approach”, Mcgraw Hill International Edition, pp.397-415, 2010.

[6] ISO, “ISO/IEC 9126-1 : Information Technology - Software Quality Characteristics and Metrics - Part 1: Quality characteristics and subcharacteristics”, ISO, 1997.

[7] Stephen R. Schach, 유해영 옮김, “객체지향 소프트웨어공학”, McGraw-Hill Korea, pp.221-254, 2005.

[8] 이재기, 신상권, 남상식, 박권철, “소프트웨어 이식성 분석과 생산성 평가”, 전자통신동향분석, 제14권, 제5호, 1999.

[9] 송병성, 이재성, 류성열, 이남용, “ISO/IEC 국제표준에 기반한 국가연구개발사업 품질측정모델에 관한 연구”, 한국 IT서비스학회지, Vol.7, No.3, 2008.

[10] 양해술, 강배근, 이하용, “디지털 저작권 관리 SW의 이식성 시험 방법”. 한국콘텐츠학회논문지, Vol.9, No.4, 2009.

[11] Ernest H. Forman, Saul I. Gass, “The Analytic Hierarchy Process－An Exposition”, Operations Research, Vol.49, No.4, pp.469-486, 2001.

[12] 윤민석, 박승봉, "AHP기법에 의한 패키지 소프트웨어 품질요구사항 우선순위 차이에 관한 실증 연구", 인터넷전자상거래연구, Vol.8, N0.2, 2008.

[13] 마크 머피, 강철구 옮김, "알짜만 골라 배우는 안드로이드 프로그래밍", 에이콘, pp.41-46, 2009.

박 해 윤

e-mail : park8312@dankook.ac.kr

2002년 단국대학교 정보컴퓨터학과(학사)

2010년 단국대학교 컴퓨터학과(석사)

2010년～현 재 단국대학교 컴퓨터학과 박사과정

관심분야 : 소프트웨어공학, 웹공학, 테스트

유 해 영

e-mail : yoohy@dankook.ac.kr

1979년 단국대학교 수학과(학사)

1981년 단국대학교 수학과(석사)

1994년 아주대학교대학원 컴퓨터공학과 (박사)

1983년~현 재 단국대학교 컴퓨터학부 교수

관심분야 : 소프트웨어공학, 웹공학