컴포넌트 개발 문서의 품질 평가 및 개선에 관한 경험적 연구

장 윤 정† · 이 경 환††

요 약

최근에 많은 기업들이 재사용 컴포넌트를 이용하여 소프트웨어를 개발하고 있다. 컴포넌트의 효과적인 재사용은 소프트웨어 개발 생산성을 높여주고, 품질 향상을 유도하며, 품질이 우수한 컴포넌트 산출물의 개발은 컴포넌트 기반 소프트웨어 개발의 이슈로 논의되는 유지보수 문제를 감소시킨다. 본 논문에서는 경험적 연구에 의한 컴포넌트 개발 문서의 품질 평가 및 개선에 관한 방안을 제시한다. 컴포넌트 품질 평가 방안은 컴포넌트 품질 참조 모델과 품질 평가 모델로 구성된다. 참조 모델은 컴포넌트 개발자를 위한 품질 지침을 포함하며, 평가 모델은 컴포넌트 유통 및 평가기관을 위한 평가 지침을 포함하고 있다. 본 논문의 타당성을 검증하기 위해, 컴포넌트 개발 기업을 대상으로 품질평가 모델을 적용하고 개선에 관한 방안을 제시하였다. 또한, 컴포넌트 문서의 품질 개선 방법 및 개선 효과에 대한 분석 내용을 제시하였다. 본 논문에서 제시한 컴포넌트 개발 산출물의 품질평가 모델은 품질이 우수한 컴포넌트 개발 산출물의 생산을 유도하며, 합리적인 품질 평가 방안을 제공한다.

An Empirical Study on Quality Evaluation & Improvement of Component Development Documents

Yun-Jeong Jang† · Kyung-Whan Lee††

ABSTRACT

Recently, many IT organizations develop software system with reusable component. Effective reusing of components increases software development productivity and quality. And, development of high quality component documents decrease maintenance problems, which are issues in component-based software development. In this paper, we propose a quality evaluation model of component development documents by empirical research. It consists of component quality reference model (CQRM) and quality evaluation model (CQEM). CQRM contains quality guidelines for component developers. CQEM contains evaluation guidelines for component consumers and distributors. We performed case study to verify this paper. Also, we presented quality improvement methods and improvement effects of component development documents. The quality evaluation model of component development documents proposed in this paper leads component development documents with high quality, and provides a rational quality evaluation model.

키워드 : 컴포넌트(component), 개발 문서(Development Document), 품질 평가(Quality Evaluation), 품질 개선(Quality Improvement)

1. 서 론

오늘날 인터넷 및 하드웨어 기술의 발달로 소프트웨어에 대한 사용자의 요구사항이 매우 다양하게 나타나고 있으며, 이를 반영하기 위한 소프트웨어 시스템의 규모는 더욱 커지고 복잡해지고 있다. 복잡하고 다양한 사용자의 요구사항을 반영하기 위해 많은 소프트웨어 엔지니어들은 최소의 시간에, 최소의 비용으로, 최대의 품질을 가진 소프트웨어를 생산하기 위해 재사용 기술을 많이 이용하고 있다[2]. 최근 들어 재사용 기술의 일환으로 소프트웨어 컴포넌트에 대한 연구가 활발히 이루어지고 있으며, 산업계에서는 재사용 컴포넌트를 기반으로 소프트웨어를 개발하려고 많은 노력을 기울이고 있다. 컴포넌트의 효과적인 재사용은 소프트웨어 개발 생산성을 높여주고 품질을 향상시킬 수 있다[15]. 이러한 컴포넌트 기반 개발은 소프트웨어 시스템의 복잡성을 조정하는데 있어 가장 진보된 방법으로 인식되고 있으나, 표준 컴포넌트 기반 개발 방법론의 부재로 인해 많은 어려움을 겪고 있는 실정이며, 그로 인해 개발한 컴포넌트 문서의 품질 준수 여부도 다양한 형태로 나타나고 있다.

※ 본 연구는 2001년도 중앙대학교 학술비 지원에 의해 연구되었음.
† 준 회 원 : 중앙대학교 대학원 컴퓨터 공학과
†† 정 회 원 : 중앙대학교 컴퓨터 공학과 교수
논문접수 : 2002년 1월 29일, 심사완료 : 2002년 4월 25일

컴포넌트의 품질 수준은 컴포넌트 유통시장에서 사용자들의 선택을 위한 판단 기준으로 사용될 수 있으며, 성공적인 컴포넌트 기반 개발을 유도할 수 있다. 또한, 품질이 우수한 컴포넌트 문서의 개발은 개발 단계의 초기에 결함을 해결해 줌으로서 품질이 우수한 컴포넌트 제품의 생산을 유도하며, 컴포넌트 기반 개발의 큰 이슈로 논의되는 유지보수 문제를 감소시킬 수 있다[2]. 컴포넌트의 개발 및 유통 장점을 최대한 살리기 위해서는 품질이 우수한 컴포넌트가 개발되어야 하며, 그것을 위한 합리적인 품질 평가 모델이 제시되어야 한다.

본 논문에서는 컴포넌트 문서의 품질 준수 사항과 품질 평가 방안에 대한 경험적 연구를 제안한다. 2장에서는 관련 연구로서 컴포넌트 기반 소프트웨어 개발 프로세스와 관련 문서를 설명하며, 소프트웨어 제품품질 표준인 ISO 9126 및 ISO 14598과 소프트웨어 제품 품질 모델 및 품질 평가 요소를 살펴본다. 3장에서는 컴포넌트 문서에 대한 품질 평가 모델을 제시하고, 4장에서는 국내 응용 컴포넌트 개발사들의 컴포넌트 개발 프로젝트를 대상으로 품질 평가에 대한 사례적용 결과를 제시하고, 타 모델과의 비교 평가를 실시하였다. 마지막으로 5장에서는 본 논문의 결론과 향후 연구과제에 대해 제시하였다.

2. 관련 연구

2.1 컴포넌트 기반 개발

컴포넌트 기반 개발 방법은 "컴포넌트"라는 소프트웨어 모듈을 조립하고 합성하여 사용자의 요구에 맞는 소프트웨어를 개발하는 새로운 소프트웨어 개발 패러다임으로, 일반적으로 (그림 1)과 같은 프로세스로 구성된다. 요구사항 정의 단계에서는 여러 어플리케이션 도메인을 분석하여 정규화된 요구사항 정의서를 작성한다. 요구사항 정규화 작업은 컴포넌트의 가변성과 공통성을 판단하는 중요한 수단이 된다. 객체지향 분석 단계에서는 UML(Unified Modeling Language)을 사용하여 기능 모델에 해당하는 사용사례(Use Case) 모델, 정적 모델에 해당하는 클래스다이어그램, 그리고 동적 모델에 해당하는 시퀀스다이어그램 및 상태다이어그램을 작성한다. 컴포넌트 설계 단계에서는 특정 알고리즘에 의해 컴포넌트를 추출하고 컴포넌트 및 인터페이스를 상세 설계한다. 컴포넌트 구축 및 조립 단계에서는 컴포넌트를 구현하고, 조립하여 사용자가 원하는 어플리케이션을 개발한다. 개발된 CBD 어플리케이션은 CBD 테스트 단계에서 검증된다.

<표 1>은 컴포넌트 개발 프로세스에서 생성되는 필수 개발 문서의 리스트를 보여주고 있다. 본 논문은 <표 1>에 제시된 컴포넌트 개발 문서를 기초로 품질 평가 방안을 제시하고자 한다.

〈표 1〉 컴포넌트 개발 문서

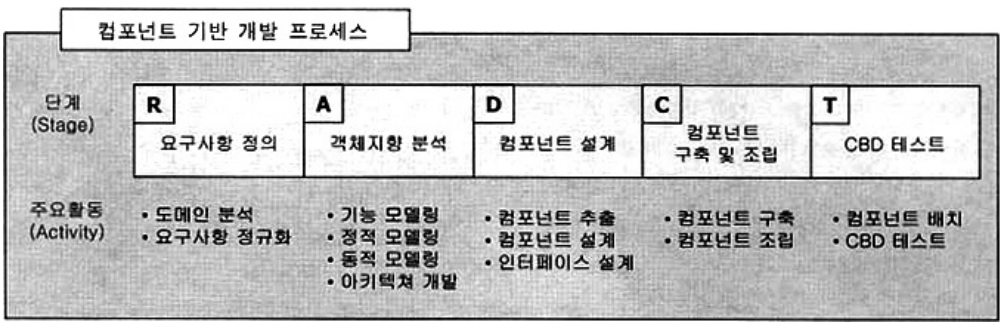
주요 활동	관련 필수 문서
R1. 도메인 분석	d1. 표준 용어 사전
R2. 요구사항 정규화	d1. 표준 용어 사전, d2. 요구사항 명세서
A1. 기능 모델링	d1. 표준 용어 사전, d2. 요구사항 명세서, d3. 유즈케이스 모델
A2. 정적 모델링	d1. 표준 용어 사전, d4. 클래스 다이어그램
A3. 동적 모델링	d5. 시퀀스 다이어그램
A4. 아키텍쳐 개발	d6. 컴포넌트 다이어그램
D1. 컴포넌트 추출	d7. 개략 컴포넌트 명세서
D2. 컴포넌트 설계	d8. 상세 컴포넌트 명세서
D3. 인터페이스 설계	d8. 상세 컴포넌트 명세서
C1. 컴포넌트 구축	d9. 컴포넌트 코드
C2. 컴포넌트 조립	d10. 커스터마이제이션 지침
T1. 컴포넌트 배치	d11. 컴포넌트 시험 보고서
T2. CBD 테스트	d11. 컴포넌트 시험 보고서

2.2 소프트웨어 제품 품질

본 절에서는 소프트웨어 제품 품질에 대해 살펴본다. 소프트웨어 제품 품질은 개발 문서와 깊은 연관이 있으며, 개발 문서의 품질은 소프트웨어 제품의 품질에 영향을 준다[11, 12].

2.2.1 소프트웨어 제품 품질 모델

소프트웨어 품질 연구소의 R.G. Dromey에 의하면 소프트웨어 제품 품질은 프로그래밍 언어의 문장들에서 사용하

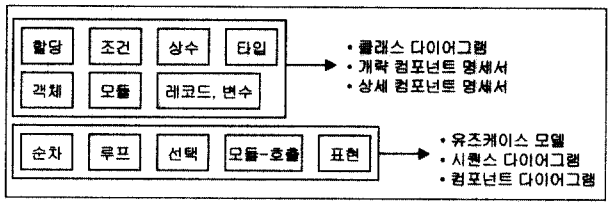


(그림 1) 컴포넌트 기반 개발 프로세스

고 있는 각각의 구조적 형태에 대한 품질-유도 속성의 집합과 관련이 있다고 정의하고 있다[10]. 품질-유도 속성은 ISO-9126의 최상위 제품 품질 요소와 관련이 있다. 소프트웨어 제품 품질 모델(MSPQ)는 소프트웨어 품질, 코딩 표준, 품질 결함을 포함하고 있으며, 코드 문장들의 구조적 형태와 품질-유도 속성과의 관계, 품질-유도 속성과 품질 속성과의 관계를 정의하고 있다.

코드 문장의 구조적 형태는 객체 무결성, 모듈 무결성, 순차 무결성, 루프 무결성, 선택 무결성, 모듈-호출 무결성, 할당 무결성, 조건 무결성, 표현 무결성, 레코드 무결성, 변수 무결성, 상수 무결성, 타입 무결성으로 분류되고 있으며, 13가지 구조적 형태에 따른 품질-유도 속성과 결함을 정의하고 있다. 품질-유도 속성은 모두 26가지로 분류되어 있으며, 품질-유도 속성에 부합될 경우, 이를 결함으로 정의하고 있다. MSPQ는 소프트웨어 제품 품질 모델로서 프로그래밍 언어의 구조적 문장 형태를 품질-유도 속성과 품질 속성 및 품질 결함과의 연관성 분석을 통해 고품질의 코드 생산을 유도하고 있다. MSPQ의 품질 속성은 ISO-9126에서 제시하고 있는 7가지 속성-기능성, 신뢰성, 사용성, 효율성, 유지보수성, 이식성, 재사용성을 사용하고 있으며, 품질-유도 속성이 ISO-9126의 품질 속성을 만족한다고 설명하고 있다.

MSPQ에서 정의하고 있는 코드 문장의 구조적 형태는 컴포넌트 개발 문서의 구성요소로 포함될 수 있으며, (그림 2)와 같은 관련성이 존재한다. 코드 문장의 결함은 잘못된 소프트웨어 개발 문서의 작성으로 인해 발생될 수 있으며, 이를 해결하기 위해 품질이 우수한 개발 문서의 작성이 우선되어야 한다.

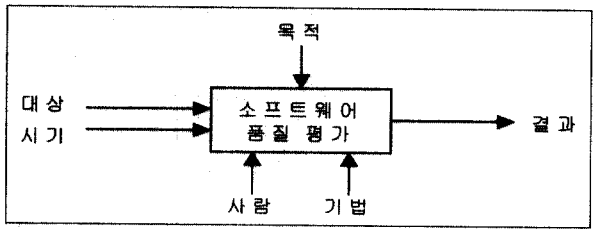


(그림 2) 품질-유도 속성과 컴포넌트 개발 문서와의 관련성

2.2.2 소프트웨어 품질 평가 요소

소프트웨어 품질 평가(SQE)는 프로세스 평가와 제품 평가로 구성되며, 프로세스 평가의 경우, CMM과 같은 프로세스 성숙도 모델을 사용하여 평가할 수 있다. 또한, 제품 평가의 경우 ISO-9126과 같은 국제 표준을 이용하여 제품을 평가할 수 있다[4]. SQE는 소프트웨어 품질 평가와 관련된 요소를 (그림 3)과 같이 정의하고 있다[12].

SQE의 첫 번째 요소인 목적은 'Why'에 해당하는 것으로, 소프트웨어 품질 평가의 이유를 나타낸다. SQE의 목적은 다양한 형태로 나타날 수 있으나, 약 67%가 소프트웨어 품질 개선을 목적으로 하고 있다. 그 외에 소프트웨어 패키지의 선택, 고객의 승인, 개발에 대한 불확실성 제거 등의 목적을 예로 들 수 있다. 다음 요소는 대상으로 'What'에 해당한다. 일반적으로 사용자의 경우, 품질 평가 대상을 운영중인 제품에 가장 큰 비중을 두고 있으며, 개발자의 경우, 요구사항 명세서, 개발 프로세스, 문서, 소스 코드, 테스트 데이터 등을 대상으로 하고 있다. 시기 요소는 'When'에 해당하는 것으로, 소프트웨어 개발 생명주기 상의 시점을 의미한다. 사람 요소는 'Who'를 의미하며, 일반적으로 제품 사용자, 고객, 공급자, 개발자, 관리자, 평가자, 컨설턴트 등이 존재한다. 기법 요소는 도구, 메트릭 등을 포함하는 평가 기법으로 정적 분석, 동적 분석, 체크리스트, 검토 등이 존재한다. SQE는 4가지의 요소를 기본으로 평가를 수행하며, 출력으로 평가에 대한 결과를 산출한다[15].

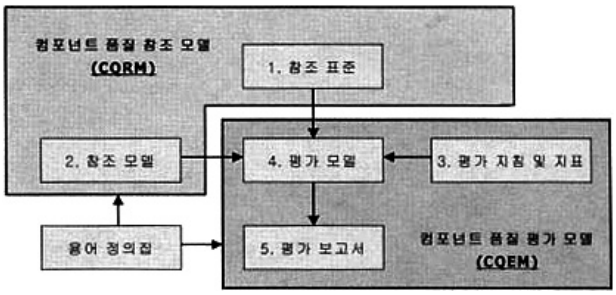


(그림 3) SQE 요소(factor)

3. 컴포넌트 개발 문서 품질평가 모델

3.1 개요

컴포넌트 개발 문서의 품질평가 모델은 크게 컴포넌트 품질 참조모델(Component Quality Reference Model, CQRM)과 컴포넌트 품질 평가모델(Component Quality Evaluation Model, CQEM)로 구성된다. (그림 4)는 컴포넌트 품질 평가 모델간의 연관성을 보여주고 있다.



(그림 4) 품질평가 모델간의 연관성

참조표준은 평가모델을 위한 각종 표준안으로 ISO 9126과 ISO 14598 및 프로세스 품질과 관련된 CMM, SPICE,

생명주기와 관련된 ISO 12207, IS-FCD 15288등을 의미한다. 이는 문서 품질평가 모델을 생성하는데 있어 참조표준으로 사용된다. 참조모델은 평가모델의 평가지침을 위한 품질지침을 포함하고 있으며, 평가지침 및 지표는 평가 지침을 작성하고 평가 방법에 대한 규정을 포함하고 있다.

3.2 컴포넌트 품질 참조 모델(CQRM)

본 절에서는 컴포넌트 참조 모델을 구성하고 있는 개발 문서의 품질지침에 대해 살펴본다. 품질 지침은 모든 개발 문서에 적용이 가능한 일반 지침과 특성 문서별 지침인 상세 지침으로 구성된다.

3.2.1 컴포넌트 개발 문서 일반지침

컴포넌트 개발 문서는 컴포넌트 기반 개발 프로세스의 활동에 의해 산출된다. 일반지침은 모든 컴포넌트 개발 문서에 공통적으로 적용되는 품질지침을 의미하며, 표준 모델링 언어인 UML을 기준으로 제시되었다.

모든 개발 문서는 정해진 표준을 준수해야 하며, 사용자의 요구사항에 대해 누락된 것이 없어야 한다. 또한, 개발 문서로 표현된 요구사항은 사용자의 요구 및 기능 면에서 정확해야 한다. 다음 <표 2>는 컴포넌트 개발 문서에 대한 일반지침과 내용을 보여주고 있다.

〈표 2〉 컴포넌트 개발 문서의 일반 지침

ID	품질 지침 항목	지침 내용
A	완전성 (Correctness)	● 모든 개발 문서는 사용자의 기능/비기능 요구사항을 모두 반영하여야 한다. ● 표준 개발 문서의 필수항목이 모두 작성되어야 한다.
B	정확성 (Completeness)	● 모든 개발 문서는 사용자의 기능/비기능 요구사항을 정확히 반영하여야 한다. ● 표준 개발 문서의 항목이 정확히 작성되어야 한다. ● 모든 개발 문서는 애매모호한 내용이 없어야 한다.
C	추적성 (Traceability)	● 모든 개발 문서의 내용은 추적이 가능해야 한다. ● 개발 문서의 입력/출력이 정확히 정의되어 사용되어야 한다. ● 개발 문서간의 개발 내용이 일관성을 가져야 한다.
D	사용성 (Usability)	● 모든 개발 문서는 이해하기 쉬워야하며, 읽기 쉬워야 한다. ● 모든 개발 문서는 Stakeholder 간의 대화 수단으로 사용되어져야 한다.
E	유지보수성 (Maintainability)	● 모든 개발 문서는 수정사항을 쉽게 반영할 수 있어야 한다. ● 모든 개발 문서는 쉽게 확장될 수 있어야 한다.
F	이식성 (Portability)	● 하드웨어 독립성이 있어야 한다. ● 소프트웨어 독립성이 있어야 한다. ● 재사용성이 있어야 한다.

3.2.2 컴포넌트 개발 문서별 상세 지침

컴포넌트 개발 문서별로 작성되는 방법과 그 의미가 서로 다르므로 각 개발 문서별로 서로 다른 품질지침이 적용된다. 개발 문서별 상세 지침은 각 개발 문서의 필수 항목 요소와 세부 지침이 분리되어 제시되었다. 다음은 컴포넌트 개발 문서별 상세 지침을 나타낸다.

(1) 표준 용어 사전

요구사항 집합에서 공통된 단어를 추출하고, 이를 정리하여 표준 용어집을 만든다. 표준 용어사전은 요구사항의 표준화를 위해 사용되며, 표준 용어, 유사 용어를 포함한다.

표준 용어 사전 세부지침	선택성	관련 지침
표준 용어는 '한글'로 표기한다.	필수	B, D, E
설명 부분에는 영문 표기와 함께 표준 용어의 설명을 기록한다.	필수	A, D
유사 용어는 '한글' 혹은 '영문'으로 표기하며, 표준 용어와 동일하더라도 기록한다.	필수	A, D
2개 이상의 도메인을 분석하여 표준 용어를 결정한다.	선택	D, F
표준 용어 사전은 하나의 최종 버전만을 유지 및 관리한다.	필수	E
고객/사용자로부터 확인을 받는다.	선택	A, B

(2) 요구사항 명세서

각각의 요구사항 명세서에서 공통된 기능성을 추출하여, 공통 요구사항을 만든다. 공통된 기능성 중에서도 가변성이 있으면 테이블에 명시한다. 요구사항 명세서 개발 문서는 요구사항ID, 요구사항 명, 요구사항기술, 가변성 항목을 포함한다.

요구사항 명세서 세부지침	선택성	관련 지침
표준 용어를 사용한다.	필수	B, C, D, F
표준 개발 문서 작성규칙을 준수한다.	필수	B, D, E
각각의 요구사항에 대해 모든 공통 기능을 추출한다.	필수	A
공통된 기능성 중에 가변성이 있으면 요구사항 명세서의 가변성 부분에 기록한다.	선택	A, F
2개 이상의 요구사항 명세서를 분석하여 공통 기능을 추출한다.	필수	F
추출된 비기능(성능, 하드웨어) 요구사항을 기재한다.	선택	A, F
애매 모호한 요구사항은 정확히 기재한다.(예. 성능이 좋아야 한다, 응답 속도가 빨라야 한다.)	필수	B
중복되는 요구사항이 없어야 한다.	필수	D, E
고객/사용자로부터 확인을 받는다.	선택	A, C

(3) 유즈케이스 모델

유즈케이스 다이어그램은 기능성을 보여주고, 유즈케이

스 기술서는 한 유즈케이스의 상세 설명과 흐름을 보여준다. 유즈케이스 다이어그램과 유즈케이스 기술서는 공통된 기능성 뿐 아니라, 가변성도 표현할 수 있으며, 유즈케이스 명, 주흐름, 선택흐름, 예외흐름, 가변성 항목을 포함한다.

유즈케이스 모델 세부지침	선택성	관련 지침
표준 용어를 사용한다.	필수	B, C, D
표준 개발 문서 작성 지침과 UML 표준을 준수한다.	필수	B, D, E, F
추출한 유즈케이스의 범위가 정확한지 점검한다.	필수	B, D, E
온라인 유즈케이스와 배치 유즈케이스 전부 식별한다.	필수	A
누락된 사용자 요구사항이 없는 지 점검한다.	필수	A, C
유즈케이스 기술은 주흐름, 선택흐름, 예외흐름으로 분리하여 작성한다[1][8].	필수	A, B, C, D, E, F
유즈케이스에 대해 우선 순위 작업을 수행한다[1].	선택	E
사용 시나리오를 통해 유즈케이스 모델을 점검한다[1].	선택	A, B
유즈케이스 모델에 UML의 Stereo Type을 이용하여 가변성을 표현한다.	선택	F
고객/사용자로부터 확인을 받는다[1].	선택	A, B

(4) 클래스 다이어그램

공통된 클래스들과 클래스들 사이의 간계, 속성, 오퍼레이션을 보여준다. 클래스의 속성과 오퍼레이션에 가변성이 있으면 가변성을 표시하며, 클래스, 속성, 오퍼레이션, 관계, 가변성 항목을 포함한다.

클래스 다이어그램 세부지침	선택성	관련 지침
표준 개발 문서 작성지침과 UML 표기법을 준수한다.	필수	B, D, E, F
클래스 다이어그램의 클래스명, 속성명, 오퍼레이션명은 영문으로 표기한다.	필수	C, E
클래스의 오퍼레이션명과 시퀀스 다이어그램의 메시지명을 일치 시킨다[8].	필수	B, C, E
클래스의 Stereo Type을 명확히 표현한다.	선택	A, B, D, E, F
클래스는 Stereo Type을 이용하여 가변성을 표현한다.	선택	F
클래스 간 관계를 명확히 정의한다.	필수	A, B
누락된 속성과 오퍼레이션이 없는지 점검한다.	필수	A, D, E

(5) 시퀀스 다이어그램

각 유즈케이스에 대한 메시지의 흐름을 보여준다. 시퀀스 다이어그램은 시스템의 동적인 면을 보여주며, 클래스의 다이어그램의 일관성을 체크한다. 컴포넌트 개발 시에는 인터페이스 디자인, 컴포넌트 워크플로우 디자인 등에 사용되며, 액터, 객체, 메시지흐름, 가변성 항목을 포함한다.

시퀀스 다이어그램 세부지침	선택성	관련 지침
표준 개발 문서 작성지침과 UML 표기법을 준수한다.	필수	B, D, E, F
시퀀스 다이어그램은 유즈케이스 당 하나 이상을 작성한다. 일반적으로, Workflow 당 하나를 작성한다[8].	필수	A, B, C, D, E, F
시퀀스 다이어그램의 객체는 반드시 클래스 인스턴스로 한다[8].	필수	B, C
시퀀스 다이어그램의 메시지 흐름과 유즈케이스의 기술 내용이 일치하도록 한다[8].	필수	A, B, C
시퀀스 다이어그램의 작성 결과로 도출되는 클래스의 속성 및 오퍼레이션은 패밀리 클래스 다이어그램에 반영한다.	필수	A, B, C
가변성 메시지 흐름 위에 Stereo Type과 Guard Condition으로 처리하여 표현한다.	선택	F

(6) 컴포넌트 다이어그램

컴포넌트의 인터페이스, 컴포넌트 내부에 포함되는 클래스와 유즈케이스를 보여준다. 또한 컴포넌트 들 사이의 관계성을 유즈케이스 사이의 관계성과 클래스 사이의 관계성을 통해 보여줄 수 있으며, 컴포넌트 다이어그램은 컴포넌트명, 오퍼레이션 리스트, 클래스 다이어그램, 유즈케이스 다이어그램 항목을 포함한다.

컴포넌트 다이어그램 세부지침	선택성	관련 지침
표준 개발 문서 작성지침을 준수한다.	필수	B, D, E, F
컴포넌트 다이어그램의 클래스 다이어그램과 유즈케이스 다이어그램은 클래스 다이어그램 및 유즈케이스 다이어그램과 일치해야 한다.	필수	A, B, C, E

(7) 개략 컴포넌트 명세서

컴포넌트 다이어그램, 컴포넌트의 이름, 인터페이스에 대한 명세, 워크플로우 등을 통해 컴포넌트 기능과 API를 명

개략 컴포넌트 명세서 세부지침	선택성	관련 지침
표준 개발 문서 작성지침을 준수한다.	필수	B, D, E, F
컴포넌트는 특정 추출 알고리즘(방법)에 의해 추출한다.	선택	B, E, F
컴포넌트 인터페이스는 "Provided" 와 "Required"로 분리하여 추출한다.	선택	A, B, E, F
각 오퍼레이션에 대해 패러매터, 사전 조건, 사후 조건을 명시한다.	필수	A, B, D
인터페이스 당 하나의 컴포넌트 워크플로우를 작성한다.	선택	A, B, D, E, F

세하며, 컴포넌트 명, 기술, 컴포넌트 다이어그램, 인터페이스 명세, 워크플로우 다이어그램 항목을 포함한다.

(8) 커스터마이제이션 지침

커스터마이제이션 지침 세부지침	선택성	관련 지침
표준 개발 문서 작성지침을 준수한다.	필수	B, D, E, F
모든 가변성에 대해 커스터마이제이션 지침을 개발한다.	선택	C, E, F
컴포넌트 재사용 시나리오를 통해 커스터마이제이션 지침을 검증한다.	선택	A, B, F

커스터마이제이션 지침은 가변성에 대한 커스터마이제이션 구현 전략을 명세하며, 컴포넌트 명, 가변성 명, 가변성 유형, 구현 유형, 오퍼레이션 항목을 포함한다.

(9) 상세 컴포넌트 명세서

상세 컴포넌트 명세서는 구현 수준에서 컴포넌트 기능과 인터페이스를 명세하며, 컴포넌트 명, 기술, 컴포넌트 다이어그램, 컴포넌트 인터페이스 명세, 컴포넌트 워크플로우 다이어그램, 구현 정보, 플랫폼 항목을 포함한다.

상세 컴포넌트 명세서 세부지침	선택성	관련 지침
표준 개발 문서 작성지침을 준수한다.	필수	B, D, E, F
컴포넌트 인터페이스는 "Provided" 와 "Required"로 분리하여 추출한다.	선택	A, B, E, F
각 오퍼레이션에 대해 패러매터, 사전 조건, 사후 조건을 명시한다.	필수	A, B, D
인터페이스 당 하나의 컴포넌트 워크플로우를 작성한다.	선택	A, B, D, E, F

(10) 컴포넌트 코드

컴포넌트가 실행되기 위한 컴파일 된 코드와 실행 환경을 명세하며, 코딩 표준, 소스코드 항목을 포함한다.

컴포넌트 코드 세부지침	선택성	관련 지침
조직의 코딩 표준을 준수한다.	필수	D, E, F
컴포넌트 코드와 컴포넌트 중간 개발 문서의 내용을 일치시키고, 상호 추적이 가능하게 작성한다.	필수	A, B, C, D, E
컴포넌트 코드는 일기 쉽게 작성한다.	필수	D, E, F
재사용 및 확장을 위해 코드에 대해 Refactoring 작업을 수행한다.	선택	D, E, F

(11) 컴포넌트 시험 보고서

컴포넌트 시험의 결과를 요약하며, 날짜, 해당 요구사항, 컴포넌트 명, 오퍼레이션 명, 테스트 케이스 항목을 포함한다.

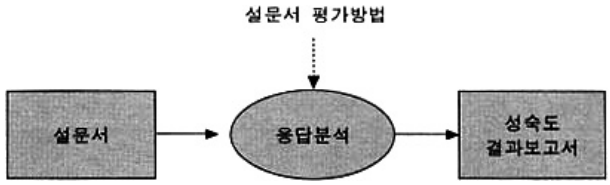
컴포넌트 시험 보고서 세부지침	선택성	관련 지침
표준 개발 문서 작성지침을 준수한다.	필수	B, D, E, F
블랙박스 테스트와 화이트박스 테스트를 분리하여 보고서를 작성한다.	선택	A, B
시험 보고서의 '해당 요구사항' 명세 내용은 요구사항 명세서의 내용과 일치시키고, 추적이 가능하게 한다.	필수	C
시험 보고서의 '테스트 케이스'는 요구사항 명세서를 기반으로 추출하여 작성한다.	선택	A, B, C
고객/사용자로부터 확인을 받는다.	선택	A, B

3.3 컴포넌트 품질 평가 모델(CQEM)

3.3.1 품질 평가 프로세스

컴포넌트 개발 문서 평가는 3단계의 프로세스로 진행한다. 먼저, 컴포넌트 개발자를 대상으로 사전 설문조사를 실시하여, 개발 문서 작성 능력 및 조직의 프로세스 성숙도를 평가한다. 다음으로 개발자의 필수 11가지 개발 문서, 응용/확장 개발 문서 및 관리 문서를 대상으로 문서 심사를 실시한다. 개발 문서에 대한 세부 평가는 11가지 필수 개발 문서를 대상으로 실시하며, 기타 제출 문서는 설문 평가 및 문제점 개선 보고서 작성 시 반영한다. 마지막으로 문서 현장 심사는 개발 조직을 방문하여, 인터뷰를 통해 평가를 실시하며, 문서심사 및 설문심사의 오류를 조정하여 최종 평가 보고서를 작성한다.

사전 설문조사는 CMM/SPICE 기반의 프로세스 성숙도 설문서를 기반으로 실시하며, Level 3을 목표로 설문을 실시한다. 전체 6가지 핵심 프로세스 중, 하청 관리는 컴포넌트 개발 문서 품질평가와 관련이 없으므로 제외한다.



(그림 6) 설문 평가 프로세스

사전 설문 응답 내용을 바탕으로 각 프로세스 영역에 대해 강점과 약점을 파악하고 정리하여, 성숙도 결과 보고서를 작성한다. 부족한 내용은 현장 심사 시 인터뷰를 통해 발견하고, 최종 문서 평가 보고서에 반영한다.

3.3.2 평가 지침 및 지표

(1) 품질 목표

컴포넌트 개발 문서의 품질 목표는 기능성, 신뢰성, 사용성, 효율성, 유지보수, 이식성을 선정하였으며, 아래 <표 3>은 참조모델에서 설명된 컴포넌트 개발 문서 품질 지침이

컴포넌트 개발 문서의 품질 목표를 어느 정도 수용하고 있는지를 보여주고 있다. <표 3>을 통해, 15개의 품질 지침이 6개의 ISO9126 품질 목표를 모두 수용하고 있음을 알 수 있다.

〈표 3〉 품질 목표에 대한 개발 문서 품질 지침의 반영 정도

품질목표 / 개발문서 품질지침	기능성	신뢰성	사용성	효율성	유지 보수성	이식성
A. 완전성						
f1. 기능	○	○	○		○	
f2. 문서	○	○				
B. 정확성						
f3. 기능	○	○	○		○	
f4. 문서	○	○			○	
f5. 내용		○		○	○	
C. 추적성						
f6. 기능추적	○	○			○	
f7. 항목추적		○			○	
f8. 선후관계					○	
D. 사용성						
f9. 판독성			○	○	○	
f10. 대화수단			○	○	○	
E. 유지보수성						
f11. 수정용이			○	○	○	
f12. 확장용이			○	○	○	
F. 이식성						
f13. H/W 독립성						○
f14. S/W 독립성						○
f15. 재사용성			○			○

(2) 평가의 설계 및 수행

평가단계에 맞추어서 평가에 필요한 양식을 편집하고, 등급 결정을 위한 기준을 정한다. 또한 모든 평가원들에게 유사한 적용을 할 수 있도록 준비한다[3].

1. 설문서 분석, 문서검토, 면담 과정을 거쳐서 평가를 수행한다.
2. 평가결과를 분석한다.
3. 평가보고서를 작성한다.

(3) 평가 지표

컴포넌트 개발 문서의 품질은 발견되는 결함수와 관련이 있다. 발견되는 결함 비율에 의해 개발 문서의 품질이 결정되며, 개발 문서의 작성 능력에 대해서는 CMM와 SPICE를 사용한다. 개발 문서의 작성 능력을 알아보는 프로세스 능력 평가의 지표는 CMM 및 SPICE 기준에 따른다.

결함은 품질 요구사항을 만족시키지 못할 때, 모두 결함으로 본다[13]. 따라서 개발 문서의 결함은 개발 문서의 품질 지침에 위배되는 모든 것을 결함으로 본다. 따라서 <표 2>에서 제시된 15가지 품질 지침을 만족하지 못할 경우, 개발 문서의 결함으로 추출한다. 결함을 파악할 때, 개발 문서의 일반 지침과 각 개발 문서별 상세 지침을 참고한다. 지침에 위배되는 모든 요소는 결함으로 분류하고 추출한다.

3.3.3 평가 양식서

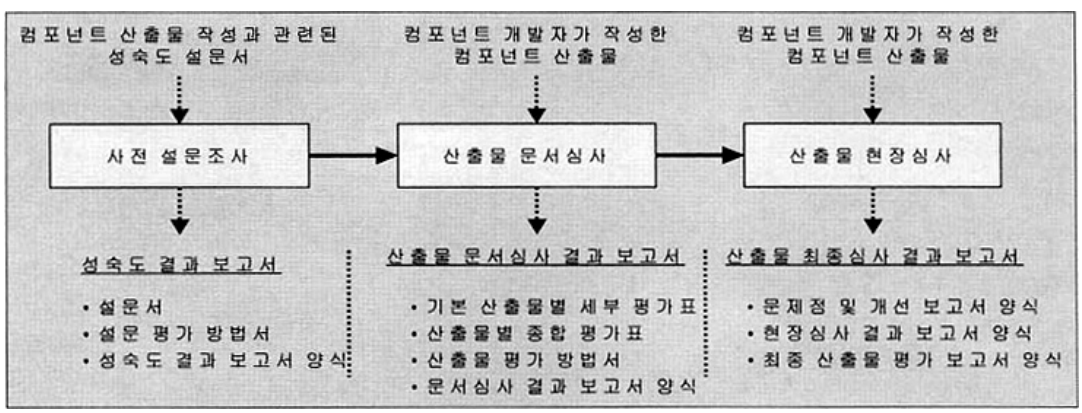
(1) 성숙도 결과 보고서

이 양식은 컴포넌트 개발 문서 작성 능력에 대한 CMM/SPICE 기반의 설문조사 결과를 보여준다.

성숙도 결과 보고서					
평가업체명		작성일자		평가자	
사 업 명				페이지	
관련 프로세서 영역		강 점		약 점	
요구사항 관리					
소프트웨어 프로젝트 계획					
종합의견					

(2) 기본 개발 문서별 세부 평가표

이 양식은 11가지 기본 개발 문서별로 세부 평가 결과를



(그림 5) 컴포넌트 품질 평가 프로세스

반영한다.

- 항목 설명
  - 1~11 : 표준 용어 사전(d1)~컴포넌트 시험 보고서(d11)를 기재한다.
  - 평가항목 : <표 7>의 15가지 개발 문서 품질 지침(f1~f15)을 기재한다.

개발문서별 세부 평가표												
평가업체명		작성일자						평가자				
사 업 명								페이지				
품질지침	평가항목	1	2	3	4	5	6	7	8	9	10	11
완 전 성	요구사항추출	10										
	필수산출물항목	8										
정 확 성	요구사항추출	10										
	필수산출물항목	7										
	요구사항내용	10										
합 계		45										
평 균		9										

- 합계(S) : 각 개발 문서에 대한 평가 항목의 합계를 기재한다.

$$Si = \sum_{1}^{15} fj \quad (i : \text{개발 문서 번호}, j : \text{평가항목 번호})$$

$$Si = \sum_{1}^{15} (wj \times fj) \quad (wj : \text{비중치})$$

(3) 문서심사 결과 보고서

이 양식은 11가지 기본 개발 문서의 문서 심사 결과를 보여준다.

문서심사 결과 보고서						
평가업체명		작성일자		작성자		
사 업 명				페이지		
산출물명	비중	평가자1	평가자2	평가자3	평균점수	총점
표준 용어 사전	5					
요구사항 명세서	7					
종합의견						

- 항목 설명
  - 비중(W) : 평가자들이 모여, 각 개발 문서에 대한 비중을 결정하고, 기재한다.
  - 평가자 점수(P) : 해당 개발 문서에 대해 1점~10점을 부여한다.

$Pij = 10 - (Dj / 10)$ (i : 평가자, j : 개발 문서 번호)

예) 표준 용어 사전에 대한 Defect 발견률이 45%일 경우, 평가자 1의 점수는

$P_{11} = 10 - (45/10) = 10 - 4 = 6$

- 평균점수(A) : 평가자 평가점수에 대한 평균값을 기재한다.

$$Aij = (\sum_{i=1}^{3} Pi)/3$$

- 총점(T) : 해당 개발 문서에 비중에 평균점수를 합한 값을 기재한다.

$$Tij = Wj \times Aij$$

- 종합의견 : 문서심사 결과에 대한 종합 의견으로, 품질이 우수한 개발 문서 및 품질이 나쁜 개발 문서에 대한 의견을 제시하고, 평점을 기재한다.

(4) 문제점 및 개선 보고서

이 양식은 11가지 개발 문서에 대한 현장 심사 내용을 보여준다.

문제점 및 개선 보고서					
평가업체명		작성일자		평가자	
사 업 명				페이지	
순번	문제점		관련산출물	개선요구사항	
1					
2					

(5) 문서 최종평가 보고서

이 양식은 컴포넌트 개발 문서에 대한 최종 평가 내용을 보여준다.

개발문서 최종평가 보고서					
평가업체명		작성일자		평가자	
사업명				페이지	
설문평가결과					
강 점			약 점		
문서심사결과					
총 점	비중치 합계		평점(10점 만점)	등 급	
현장심사결과					
강 점			약 점		

- 항목 설명
  - 설문평가결과 : 문서심사를 통해 발견된 프로세스의 전반적 강점과 약 점을 기재한다. 이때, 성숙도 결과 보고서를 기초로 작성한다.
  - 문서심사결과 :
    1. 총점 : 문서별 총점의 합계를 기재한다.
    2. 비중치 합계 : 문서별 비중의 합계를 기재한다.
    3. 평점 : 총점에 비중치 합계를 나눈 값을 기재한다.
    4. 등급 : A - 평점 9.0점~10점, B - 평점 8.0점~8.9점
       C - 평점 7.0점~7.9점, D - 평점 6.0점~6.9점
       E - 평점 5.0점~5.9점, F - 평점 5.0 미만

4. 품질평가 사례적용

본 논문의 적용 가능성을 검증하기 위해 응용 컴포넌트 개발사 15곳을 선정하여, 품질평가 모델에 의한 개발 문서의 품질을 평가해 보았다. 1차 품질평가 결과 후 참조 모델과 프로세스 개선을 통한 품질 개선 실시 후, 2차 평가를 실시하여 본 논문의 타당성을 검증했다.

4.1 평가업체 개요

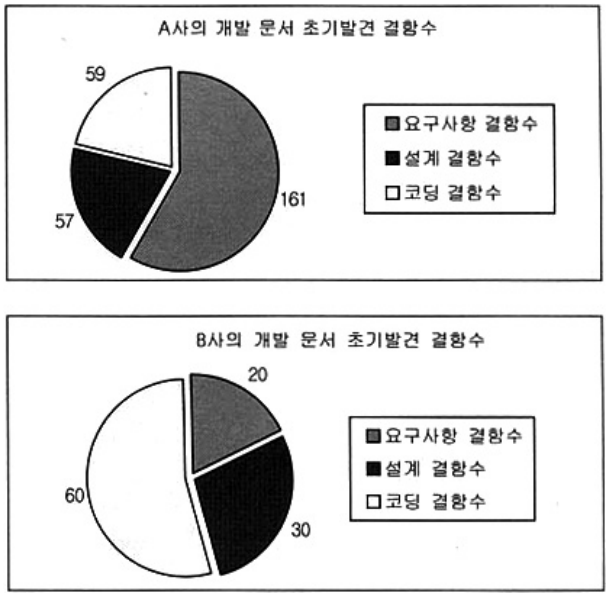
1차 평가는 2001년 7월 5일부터 7월 14일까지 실시되었으며, 7월 15일부터 20일까지 1차 보고서 작성이 이루어졌다. 1차 평가에 참여한 업체 분류는 대기업이 3곳, 중소기업 11곳, 은행 1곳으로 나타났다. 영역 분류는 제조 3곳, 뱅킹 3곳, 신용 2곳, 의료 2곳, 인터넷 5곳으로 조사되었다. 2차 평가는 2001년 12월 10일부터 14일까지 실시되었으며, 12월 17일부터 22일까지 보고서 작성이 이루어졌다. 2차 평가는 프로세스 능력 수준 0인 업체 3곳을 제외한 12곳 업체를 대상으로 평가를 실시하였으며, 프로세스 능력 평가와 품질 평가가 이루어졌다. 1, 2차 평가에 의해 품질 참조 모델에 의한 결함 제거 효과와 프로세스 개선을 통한 개발 문서의 품질 향상 효과를 알 수 있었다.

4.2 평가모델에 의한 평가 결과 및 개선 방안

1차 평가에서는 개발 문서의 품질 평가에 앞서 개발 조직의 프로세스 능력을 평가했다. 평가 모델은 SPICE 모델을 사용하였으며, 평가 결과는 수준 2의 능력을 가진 조직이 10개사, 수준 1이 2개사, 심각한 수준인 수준 0인 업체가 3곳으로 나타났다. 컴포넌트 개발을 위한 표준 프로세스는 존재하지 않았으며, 대부분의 업체가 개발 문서의 표준 양식과 지침이 없었다. 개발 문서의 품질 평가 결과는 C등급이 5곳, D등급이 7곳, E이하가 3곳으로 조사되었다. 1차 평가에서는 본 논문에서 제시한 품질 지침이 제공되지 않은 상태에서 평가가 이루어졌으며, 요구사항, 분석, 설계의 결함으로 분류하여 개발 문서에 대한 품질을 평가하였다. 1차 평가 업체를 대상으로 프로세스 및 개발 문서의 품질 개선을 위한 노력에 있어 다음과 같은 문제점이 조사되었다.

- 개발 문서의 종류와 양식의 부재
- 품질 개선을 위한 체크 항목 및 기준의 부재
- 품질 평가 절차 및 방법에 대한 부재
- 프로세스 및 개발 문서의 품질 관리 방안의 부재

1차 평가이후 개발 문서의 종류 및 양식과 본 논문에서 제시한 품질 지침이 제공되었으며, 이를 기반으로 조직 및 개발 문서에 대한 개선 노력이 이루어졌다. 2차 평가는 조직의 프로세스 능력 수준이 0으로 평가된 3곳을 제외한 12곳을 대상으로 평가가 이루어졌다. 2차 평가에서는 본 논문에서 제시한 품질 평가 모델에 의해 평가가 이루어졌으며, 프로세스 능력 평가의 경우, 1차 평가와 동일하게 SPICE 모델을 사용하였다. 2차 평가 결과 SPICE 수준이 3인 업체가 5곳, 수준 2인 곳이 8곳, 수준 1이 1곳으로 조사되었다. 개발 문서에 대한 품질은 90% 향상된 것으로 조사되었다. SPICE 수준이 3인 업체 5곳을 선정하여 결함 제거 비율을 조사한 결과 초기 발견 총 결함수가 평균 약 200여개에서 품질 개선 작업 이후의 결함수가 20여개로 줄어들었다. 다음 (그림 7)은 평가 업체 2곳의 초기 발견 결함수를 보여주고 있다.

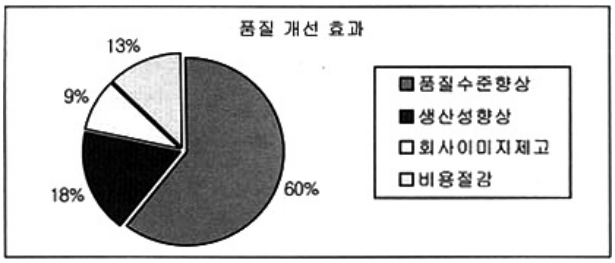


(그림 7) 초기발견 결함수

A사의 경우, 전체 결함수는 277개로 측정되었으며, B사의 경우는 110개로 측정되었다. 약 3개월의 품질 개선 과정

을 통해 A사는 초기 발견된 결함수 대부분을 제거하여 약 99%의 품질 개선 효과를 보였다. B사 또한, 초기에 110개로 발견된 결함수가 개선 활동을 통해 103개를 제거하여, 약 94%의 품질 개선 효과를 보였다.

SPICE 수준 3의 우수 업체 5곳을 대상으로 품질 개선 효과를 조사한 결과, (그림 8)과 같이 조사되었다. 개발 문서의 품질 개선은 개발 문서의 품질 수준을 향상시키고, 생산성과 비용이 절감되는 것으로 조사되었다.



(그림 8) 우수 업체 품질 개선 효과

4.3 결함 제거 효율성

결함 제거 효율성(DRE ; Defect Removal Efficiency)은 품질 보증이나 제어 활동이 모든 프로세스 프레임워크 주요 활동에 적용되어 품질 보증과 제어 활동의 여과 능력을 측정하는 것이다[11]. DRE의 이상적인 값은 1로서 소프트웨어 내에 어떠한 결함도 발생하지 않았음을 의미한다. E가 증가하면 DRE는 1에 접근할 수 있으며, E가 증가하면 D의 마지막 값은 줄어든다. DRE는 다음과 같이 정의된다.

$$DRE_i = E_i/(E_i + E_{i+1})$$

여기서 E_i = 소프트웨어 공학 활동 i에서 발견되는 오류의 수

E_{i+1} = 소프트웨어 공학 활동 i+1 동안 발견되는 오류의 수

A사의 경우 분석 결함이 161개, 설계 결함이 57개, 구축 결함이 59개로 조사되었다. B사의 경우 분석 결함이 20개, 설계 결함이 30개, 구축 결함이 60개로 조사되었다. A4사의 데이터를 기초로 DRE를 측정하였을 때 다음과 같은 결과를 얻을 수 있다.

〈표 4〉 1차 평가에 대한 DRE 측정값

프로세스 단계	A사 DRE	B사의 DRE
분석단계(E_i)	161 / (161 + 57) = 0.74	20 /(20 + 30) = 0.4
설계단계(E_{i+1})	57 /(57 + 59) = 0.49	30 /(30 + 60) = 0.33

A사의 경우 1차 평가에서 발견된 분석 결함 161개를 기반으로 품질 참조 모델을 적용하였을 경우, 161개에 대하여 결함으로 발견되지 않았다. 따라서 DRE의 측정 시, E_i값은 1차 평가의 수치를 입력으로 하며, E_{i+1}은 2차 평가의 수치를 입력으로 계산한다. 품질 참조 모델을 적용한 이후의 DRE의 값이 <표 5>와 같이 측정되었으며, <표 4>에 비해 DRE의 값이 1에 가까움을 알 수 있다.

〈표 5〉 품질 참조 모델 적용 이후의 DRE 측정값

프로세스 단계	A사 DRE	B사의 DRE
분석단계(E_i)	161/(161+0) = 1	20/(20+5) = 0.8
설계단계(E_{i+1})	57/(57+2) = 0.97	30/(30+2) = 0.94

품질 참조 모델은 컴포넌트 기반 개발 프로세스의 생명주기 초기 단계에 결함을 제거하였고 품질의 개발 문서의 생산을 유도한다. 개발 문서에 대한 초기 결함 제거는 결함 증폭을 감소시키며, 최종 제품의 결함을 감소시킨다.

4.4 타 모델과의 비교 평가

소프트웨어 제품 품질 모델은 대부분 소스 코드에 대한 평가 메트릭이 주를 이루고 있다. 요구사항, 분석 및 설계

〈표 6〉 타 모델과의 비교 평가

평가항목	ISO9126	MSPQ	CQRM과 CQEM
분 류	소프트웨어 제품 품질 표준	소프트웨어 제품 품질 모델	컴포넌트 개발 문서 품질 모델
대 상	최종 제품(실행 가능한 형태의 소프트웨어)	소스 코드 중심	컴포넌트 개발 문서
장 점	국제 표준이며, 품질 목표와 요인에 대한 분류가 체계적으로 제시되었다.	ISO9126의 품질 속성을 반영하고 있으며, 소스 코드 중심의 품질-유도 속성을 26가지로 세분하여 정의하고 있다. 또한, 품질-유도 속성별로 결함을 분류하여, 결함 추출을 용이하게 한다.	ISO9126의 품질 속성을 반영하고 있으며, 개발 문서에 관한 품질 속성을 15가지 요소로 분류하여 정의하고 있다. 또한, 결함 발견 및 개선을 위한 품질 지침을 제공하고 있으며, 양식등의 제공으로 실용화하기 쉽다.
단 점	개념적인 정의를 확립하는 수준으로 실용화하기가 어렵다.	소스 코드에 국한된 품질 모델이다.	특정 개발 문서에 국한되어 적용된다.

단계의 개발 문서에 대한 품질 모델은 생명주기 초기 단계에 결함을 제거하여 고 품질의 개발 문서의 산출을 유도한다. 개발 문서에 대한 초기 결함 제거는 소스 코드에서의 결함과 최종 제품의 결함을 감소시킨다. 본 논문에서 제시한 CQRM/CQEM, 국제 표준인 ISO9126, 그리고 소프트웨어 품질 연구소에서 발표한 MSPQ 모델을 비교 평가하였다. 모델 간의 비교는 장점과 단점을 중심으로 평가하였다.

5. 결 론

본 논문에서는 컴포넌트 개발 과정에서 생성되는 개발 문서에 대한 품질 평가 방안을 모색해 보았다. 컴포넌트 개발 문서 품질 평가 방안은 크게 참조 모델과 평가 모델로 구성된다. 참조 모델은 컴포넌트 개발 업체를 위한 개발 가이드라인 모델이며, 평가 모델은 컴포넌트 유통 및 평가 기관을 위한 평가 가이드라인 모델이다. 본 연구의 최종 목표는 개발 업체로 하여금 우수한 품질의 개발 문서를 개발할 수 있도록 유도하고, 개발한 컴포넌트 개발 문서에 대해, 합리적이고 공정한 평가 모델을 제시하는 것이다. 참조 모델 개발을 위해, 여러 국제 표준을 참조하였으며, 11가지 필수 개발 문서에 대한 작성 지침 및 표준 양식을 참조로 하였다. 또한, 평가 모델 개발을 위해, ISO 국제 표준에서 제시된 품질 목표 및 기준을 수용하였으며, 평가자의 편의를 위해 평가 양식을 제시하였다. 본 논문에서 제시한 평가모델의 타당성을 검증하기 위해, 응용 컴포넌트 개발사 15곳을 대상으로 개발 문서에 대한 품질 평가를 실시하였으며, 평가 결과에 따라 결함을 제거한 후, 그 결과를 분석하였다. 본 논문에서 제시한 참조 모델이 개발 문서의 품질 개선에 효과적 이였음을 사례적용을 통해 알 수 있었으며, 품질 평가 모델에 대한 타당성을 검증할 수 있었다. 본 논문의 결과는 국내 기업들에게 컴포넌트 개발 문서에 대한 품질 지침을 제시하여, 품질이 우수한 컴포넌트 개발 문서의 생산을 유도할 수 있으며, 합리적이고 공정한 컴포넌트 개발 문서 품질 평가 모델을 통해, 국내 컴포넌트 유통시장의 안정 및 컴포넌트 산업 활성화를 유도할 수 있을 것으로 보인다. 또한, 개발 초기단계의 결함 제거를 통해 소스코드 및 제품에 대한 결함 증폭을 감소시키고, 컴포넌트 개발비용 및 유지보수 비용을 감소시킨다. 향후 연구과제로는 컴포넌트 개발 문서의 품질 항목을 개발자, 사용자, 유지보수자 등 다양한 관점에서 추출해 보고, 품질 항목간의 연관성을 파악해 볼 필요가 있겠다. 또한, 개발 문서와 더불어 소스코드를 포함한 실행 가능한 컴포넌트 제품에 대한 품질 평가가 요구되며, 품질, 비용 및 생산성을 모두 고려한 동적인 품질 평가 모델이 개발되어야 할 것으로 보인다.

참 고 문 헌

[1] Frank Armour, Advanced Use Case Modeling, Addison Wesley, pp.301-321, 2001.

[2] George T. Heinenman, William T. Councill, Component-Based Software Engineering, Addison Wesley, pp.435-452, 2001.

[3] ISO/IEC 14598-1, "Software Product Evaluation Part1 : General Overview," April, 1999.

[4] ISO/IEC FDIS 9126-1, "Software product quality Part1 : Quality Model," March, 2000.

[5] ISO/IEC 15504, "Software Process Improvement and Capability dEtermination," 1998.

[6] Mark C. Paulk, Charles V. Weber, etc., The Capability Maturity Model, Addison Wesley, pp.15-20, 1994.

[7] Ram Chillarege, "Orthogonal Defect Classification," IBM Watson Research Center, 2000.

[8] Rational Corp., "Unified Modeling Language 1.3," http://www.rational.com, 2002.

[9] R. G. Dromey, "Software Quality and Productivity Improvement," Software Quality Institute, Griffith University, pp.1-3, 1994.

[10] R. G. Dromey, "A Model for Software Product Quality," Australian Software Quality Research Institute, pp.15-23, October, 1994.

[11] Roger S. Pressman, Software Engineering-A Practitioner's Approach, 5th Edition, McGraw-Hill, 2001.

[12] Teade Punter, Giuseppe Lami, "Factors of software quality evaluation," Proceedings ESCOM-ENCRES'98, May, 1998.

[13] Thomas B. Hilburn, Massood Towhidnejad, "Software Quality : A Curriculum Postscript," ACM SIGCSE, pp. 167-171, 2000.

[14] 남기현, 한판암, 양해술, "개발산출물의 신뢰성 측정을 위한 메트릭의 제안과 평가", 정보처리학회논문지, 제8-D권 제3호, pp.247-256, 2001.

[15] 이경환, 최신 소프트웨어공학, pp.179-223, 청문각, 1998.

장 윤 정

e-mail : yjjang@object.cau.ac.kr

1995년 명지대학교 전자계산학과 졸업 (학사)

1998년 중앙대학교 정보산업 대학원(공학석사)

2001년 미 CMU MSE 수료

2002년 중앙대학교 대학원 컴퓨터공학과(공학박사)

1995년~1998년 기아정보시스템 연구원

관심분야 : 소프트웨어공학, 컴포넌트 기반 개발, 프로젝트 관리

이 경 환

e-mail : kwlee@object.cau.ac.kr

1980년 중앙대학교 대학원 응용 수학 (이학박사)

1982년~1983년 미국 Auburn 대학 객원 교수

1992년~현재 ISO/SC7/WG10 한국 LTC 위원장

1971년~현재 중앙대학교 컴퓨터공학과 교수

관심분야 : 소프트웨어공학, 객체 모델링, 소프트웨어 재사용