임베디드 S/W 재사용을 위한 공통 개발 지침 및 시범 사례 구축

특집 09

목 차

차정은 · 유미선
(한국전자통신연구원)

요 약

산업분야에서 임베디드 S/W의 비중이 크게 확대됨에 따라 소프트웨어 재사용 자산에 대한 가치는 현저하게 증가하고 있다. 따라서 환경적 변화에 따른 유사한 특성을 지닌 임베디드 소프트웨어에 대한 반복적인 수요가 증가하고 있다. 그러나, 임베디드 소프트웨어의 자산화를 위한 구체적인 지침 제공이 전무하여 임베디드 소프트웨어를 단지 개발 과정에서 우연히 발생되는 임시방편적인 산출물로 인식하고 있어 조직 내 임베디드 소프트웨어의 생산은 항상 비용 소모적인 오류를 만들어 내고 있다. 그러므로, 다양한 장비에 다양한 서비스를 급격하게 변화는 고객 요구에 맞도록 지원하고, 점점 단축되는 S/W의 생명주기에 효율적으로 대처하기 위해서는 임베디드 S/W의 공동 자원을 구축하고 이를 적절하게 공유하는 체계 구축은 필수적이다.

본 논문에서는 하드웨어에 의존적이며 외부 환경과 다양하게 상호협력 해야만 하는 임베디드 소프트웨어 자산들의 생성과 활용을 위해 문서화 관점에서 재사용 자산을 정의하고 분류하여 이들간의 상호 관계를 명확히 기술함으로써 임베디드 소프트웨어 재사용을 위한 표준 지침을 제공하고자 한다.

1. 서 론

S/W 개발의 낮은 생산성과 품질의 문제를 혁신적으로 개선(silver bullet) 시켜줄 수 있는 방법으로 S/W 재사용이 제시되었다. 그러나, 현재의 S/W 재사용은 1950년대의 수학이나 군사용 함수와 같은 "silver bullet"이 아니다. 대부분의 S/W들은 아주 작은 부분의 환경 변화나 고객 요구 변경에 유사 기능과 아키텍처를 가진 시스템을 반복적으로 재개발되기 때문에 짧은 생명주기 후에 폐기되어 극히 낮은 생산성을 가진다[1, 2]. 따라서, S/W 개발 조직들이 이론적으로 약속된 체계적인 생산성 향상을 도모를 위해 전사적으로 적용할 수 있는 재사용 지침 구축이 필요하다[3,4].

오늘날 같은 디지털 컨버전스 시대로의 진입을 위한 정보 기술의 핵심은 임베디드 S/W의 생산성이다. 대부분의 임베디드 S/W는 개발 및 탑재되는 외부 환경의 미소한 차이에 따라 동일한 기능을 수행하더라도 상호 호환되지 못하고 다른 S/W 산물로 인식되어 불필요한 비용을 소모하고 있는 실정이다. 특히 임베디드 S/W가 많이 활용되고 있는 휴대단말기나 자동제어 시스템 등의 영역은 비즈니스 프로세스가 거의 동일함에도 불구하고 임베디드 S/W가 가지는 하드웨어 의존성과 실시간성 등의 특성 때문에 영역에서 요구하는 적시성(time-to-market)을 위한 기존 임베디드 S/W의 재사용을 시도하지 못하고 있다. 따라서, 다양한 장비에 다양한 서비스를 급격하게 변화는 고객 요구에 맞도록 지원하고, 점점 단축되는 S/W의 생명주기에 효율적으로 대처하기 위해서는 임베디드 S/W의 공동 자원을 구축하고 이를 적절하게 공유하는 체계 구축은 필수적이다. S/W 재사용, 특히 임베디드 S/W 재사용이 극히 미진한 이유 중 하나는 기존에 생산된 임베디드 S/W에 대한 정확한 이해 정보의 제공이 불충분할뿐더러, 제공되는 S/W의 코드의 판독성을 높일 수 있는 지침이 마련되어 있지 않기 때문이다. 사실, 임베디드 S/W는 작은 규모로 특수한 하드웨어 장비를 대상으로 즉각적으로 생성, 제공되도록 요구되기 때문에 개발자들이 개발 과정에서 만들어야 되는 분석서, 설계서, 혹은 구현 문서 등을 만들지 않을뿐더러, 설사 만들어진 문서들도 체계적인 정보 흐름이나 형태화된 양식을 갖추지 못해 제 3자가 획득하고 이해하는 비용이 재개발 비용 못지 않게 많은 노력이 필요하며 경우에 따라 불가능할 경우도 있다.

더욱이 H/W 개발 원가의 민감성과 비표준화, 임베디드 시스템의 종류 확대 등 예측하기 힘든 외부 요소들을 임베디드 S/W로 해결하고자 하는 개발자의 기대 수준은 점점 높아가고 있다. 그러므로, 다양한 임베디드 시스템의 품질을 고정된 H/W적인 구현보다는 융통성 있게 조정 가능한 S/W적인 구현이 보다 중요시됨은 당연한 일이다.

본 논문에서는 재사용 개발 지침을 통해 임베디드 S/W의 재사용성을 높일 수 표준적인 개발 절차와 산출물 양식을 세부적인 임베디드 S/W 형태에 따라 제공하고자 한다. 이를 통해 이미 개발된 임베디드 S/W에 대한 조직 구성원들 간에 의사소통이 원할 해져 가변 요소가 많은 임베디드 S/W의 중복 개발을 회피하고 한번 개발된 임베디드 S/W는 조직의 자산으로서 지속적인 가치 창출을 위해 확장되고 활용할 수 있도록 지원하고자 한다.

2. 임베디드 S/W의 재사용 현황

국내 임베디드 S/W 개발 업체들은 대부분이 개발자 자신이 만든 코드의 재사용 정도만 수행할 뿐, 하드웨어와 아키텍처 변경에 따른 영향들을 완화시킬 수 있는 추가적인 비용 투자에 매우 소극적이어서 라이브러리나 프레임워크 형태의 재사용은 거의 전무하다. 이는 임베디드 S/W에 대한 반복적인 재사용 요구에도 불구하고, 개발 시 미래의 잠재적인 재사용을 위한 S/W 구조나 인터페이스 등의 계획된 체계를 전혀 고려하지 않기 때문으로 노력에 비해 재사용 효과는 극히 미비한 현실이다.

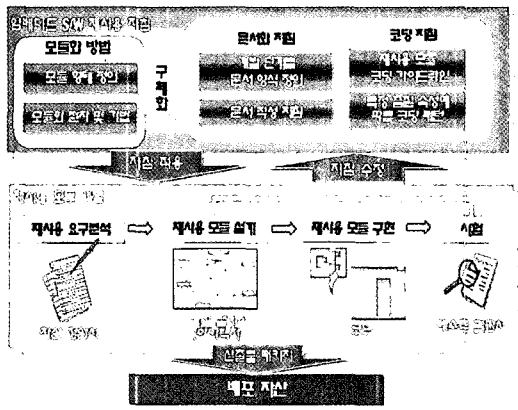
코드의 재사용 못지 않게 문서의 재사용을 중요시하며 설계 단계에서 문서 자산의 재사용 비율이 27% 이상이다. 문서 재사용에 있어서도 그대로 재사용하는 경우는 2%에 지나지 않으며 40% 이상이 문서 자산의 50% 이상을 수정하여 재사용한다[5]. 이는 임베디드 S/W 자산 개발을 위한 조직의 규격화를 통해 쉽게 해결할 수 있는 문제지만 국내 모든 기업에서는 간과하고 있는 실정이다. 실제로 핀란드의 노키아는 전사적인 재사용 체계 구축을 위해 제품계열 기반의 생산 프로세스를 구축하고 이를 기반으로 1년에 20~30개의 새로운 휴대 단말을 출시함으로써 유럽

시장의 time-to-market을 만족시키고 있으며, 삼성전자 디지털 TV S/W 개발시 발생하는 체계적인 프로세스의 부재나 다양한 고객 지향의 옵션 제품을 위한 생산 라인 변경 문제점을 S/W 재사용 및 이를 기반으로 한 전사적 프로세스 개선을 통해 노력하고 있다[6]. 특히, 임베디드 시스템 개발시 발생하는 문제점들은 설계 단계의 표준화된 표기법의 부재로 인한 상호 다른 관점의 H/W와 S/W의 통합시 발생하는 것으로[7], H/W와 S/W의 통합 아키텍처를 미리 계획하고 구조화시킬 수 있는 프로세스를 도입함으로써 해결을 시도하고 있다[8].

3. 임베디드 S/W 재사용 지침

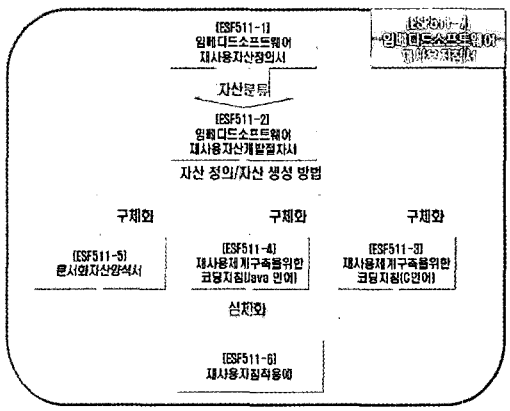
한국전자통신연구원 임베디드 S/W 연구단 선임 연구원들을 면담 설문 조사를 한 결과, 임베디드 S/W 생산성을 위해 가장 필요한 S/W 공학적 기술은 산출물 개발을 위한 표준 프로세스 및 산출물의 공유와 재사용을 위한 체계적인 지원 기법이었다. 즉, 임베디드 S/W 개발 과정의 절차와 기법, 산출물을 조직에 맞게 규격화시켜 불필요한 재개발에 의한 중복 비용을 최소화시키고, 선행 연구 산출물들의 노하우를 최소의 비용으로 전수함으로써 조직의 S/W 자산 가치를 극대화 시킬 수 있기를 요구했다. 이러한 요청을 배경으로, 조직 내 임베디드 S/W 재사용 문화 확산을 통한 S/W 개발 프로젝트의 생산성 향상을 위한 간단하면서도 실질적인 재사용 템플릿 제공을 목적으로 임베디드 S/W 재사용 지침을 구축하였다.

(그림 1)은 임베디드 S/W 재사용 지침의 개념적 구성도이다. 재사용 지침은 임베디드 S/W를 의미있는 재사용 단위로 자산화시키는 방법과 절차를 정의하고 그 과정에서 생성되는 문서자산들의 양식과 작성 기법을 제시하며, 코딩시 가독성을 높일 수 있는 권장 가이드라인과 성능, 메모리관리 등 특정한 임베디드 S/W의 품질 속성을 향상시킬 수 있는 코딩 패턴 제공한다. 임베디드 S/W 자산 개발자들은 이 지침에서 명시하는 템플리트에 따라 산출물들을 작성하고 패키지하여 배포하게 된다.



(그림 1) 임베디드 S/W 지침의 개념적 구성

(그림 1) 내용의 체계적 구축을 임베디드 S/W 재사용 지침서를 개발하였다(그림 2). 본 문서에는 임베디드 시스템 영역에서 재사용하고자 하는 임베디드 S/W 재사용 자산의 정의와 종류를 명확히 정의한 재사용 자산 정의서와, 임베디드 S/W 재사용 자산의 개발을 위한 모듈화 방법과 모듈 개발 절차에 따른 문서 작성 지침 및 코딩을 위한 지침, 지침에 따라 재사용 모듈을 구현하는 적용 예제가 내용으로 포함되어 있다.



(그림 2) 재사용 지침서의 구성

특히 코딩 지침에는 재사용 모듈의 코딩 스타일을 C 언어와 Java 언어 두 가지로 작성하였다. 그리고, 재사용 지침의 검증을 위해 EHS-Embedded HTTP Server및 MP3 플레이어, FCM(Fan Controller) 시스템을 대상으로 적용하였다.

3.1 임베디드 S/W 재사용 자산

본 논문에서 S/W 재사용 자산을 다음과 같이 정의한다.

"재사용 가능한 일정한 형태로 가공된 재사용 모듈과 이와 연관된 문서 및 관련 자료들이다. 재사용 모듈은 코딩된 원시 코드와 실행 가능한 바이너리 코드의 집합을 의미하며, 문서에는 재사용 자산 개발 과정에서 작성된 요구사항 정의서, 설계서 등이 포함된다. 또한 관련 자료는 재사용 자산의 개발 및 운영에 사용되었던 DB, 이미지 파일, 그래픽 라이브러리 등이 있다."

임베디드 SW 자산의 종류와 그에 따른 형태는 자산이 생성되는 시기나 자산의 성격 혹은 자산의 규모에 따라 다르게 나타난다. 본 논문에서는 이러한 개별 기준들을 다 포함할 수 있도록 임베디드 S/W 계층을 1차 기준으로 분류하고, 이를 자산의 개발단계별로 다시 구분하였다.

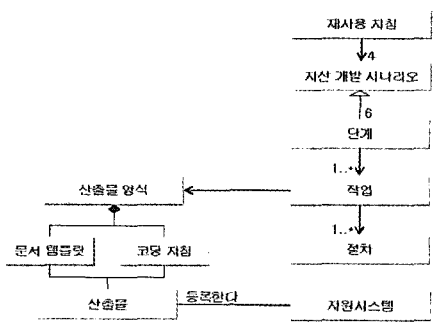
<표 1>은 임베디드 S/W의 개발 단계에 따라 분류된 문서 자산들로 자사용자의 필요성에 따라 필수, 선택 자산으로 구분된다[9].

3.2 임베디드 S/W 재사용 자산 메타 모형

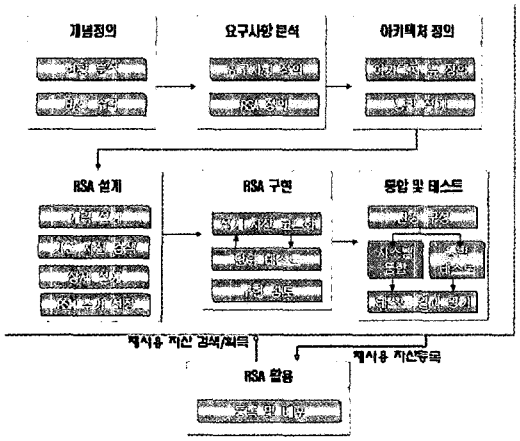
(그림 3)은 재사용 지침에 의해 재사용 자산 생성을 위한 메타모형이다. 재사용 자산의 형태에 생성할 산출물들과 개발 절차를 4가지의 시나리오로 정의하였다. 각 시나리오는 논리적인 작업 그룹인 단계와 수행하는 일의 물리적인 단위인 작업으로 구분되며 작업마다 생성할 산출물을 위한 문서 템플릿과 .코딩 지침을 제공한다.

<표 1> 개발 단계에 따른 문서 자산의 분류

	분류	자산명	내용
요구사항 정의	필수	요구사양 분석서	타겟 시스템에 대한 요구사항들을 수집하여 정형화된 형식으로 문서화한 것임. 자산의 정의, 목적, 요구사항(기능/품질), 활용 범위 등을 나타냄
		시스템 정의서	목표 시스템의 개발 및 운영, 유지보수 정보 및 제약 조건 등을 다양한 측면(즉, 비즈니스 측면 기술적 측면, 환경적 측면)에서 상세히 정의한 문서임
	선택	현황 분석서	현재 대체되는 제품들(시스템)의 특징 및 문제점, 관련 이해당사자들이 제기하는 발전 방향, 그리고 목표 시장서의 경쟁과 효과 등을 정의한 문서임.
		비전 문서	구축할 시스템에 대해 누가 사용하고, 어떤 영역에 적용되며, 어떻게 활용할 수 있는지를 정의한 문서로, 이행 당사자 들이 달성할 목표와 범위를 비즈니스와 기술적 관점에서 나타냄
		품질 정의서	요구사항 정의서에서 품질(Quality) 요구사항에 관한 사항만을 추출하여 품질 분류표 작성을 통해 시스템이 달성하고 평가 받을 품질 요소를 정의함
아키텍처 설계	필수	아키텍처 정의서	목표 제품 개발을 위한 구현 방법, 프로세서, 언어, 필요 요소들의 획득 방법 등을 정의하고 목표 시스템 전체의 연관 구조(블록 다이어그램)를 정의
		기능모듈 정의서	아키텍처 정의서의 전체 시스템의 구조에 포함된 소프트웨어 블록들의 기능적 정의를 나타낸 문서임
		종속성 분석서	환경 종속적 부분에 대한 정보를 제공하는 산출물로, 하드웨어 제약사항이나, 응용 계층 및 시스템 계층 소프트웨어에 대한 제약사항 등을 제시함
	선택	모듈별 정의서	소프트웨어 기능 모듈 정의서에서 설명된 각 소프트웨어 기능 모듈의 상세한 명세를 작성한 산출물
자산 설계	필수	자산 설계서	재사용 모듈의 설계문서
		개발환경 명세서	타겟 OS, 교차 개발 도구, 참조 타겟보드 종류, 하드웨어 제약사항 등의 정보를 제시
	선택	활용자산 명세서	커널, 부트로더, 디바이스 드라이버, 그래픽 라이브러리 및 상용 컴포넌트 등에 대한 정보를 제시
자산 구축	필수	재사용 모듈	원시코드+실행 파일
		테스트 결과서	단일 테스딩 및 통합 테스팅 결과를 제시
	선택	파일구조 정의서	재사용 모듈에 사용된 물리적인 파일의 구조 및 개별 파일들의 주요 정보들을 제시
		하드웨어 명세서	자산 개발에 필요한 하드웨어 환경에 정보(장치별 메모리 맵, 인터럽트, DMA, I/O 등)를 정의함
배포	필수	재사용 자산 등록서	재사용 지원 시스템에 등록을 위해 재사용 지원 시스템 형태에 필요한 정보를 포함하는 산출물
	선택	배치 계획서	자산 설치 및 설정에 관련된 매뉴얼 및 자산의 라이센스 등에 관련한 정보를 정의



(그림 3) 재사용 자산 작성 모형



(그림 4) 재사용 자산의 개발 절차

3.3 임베디드 S/W 재사용 자산 개발 프로세스

재사용 자산을 개발하기 위한 작업의 프로세스는 (그림 4)와 같다. 임베디드 S/W 속성을 보다 잘 반영하고 재사용 자산들 간의 유기적 연계성을 유지할 수 있도록 개발 과정을 체계화 것이다. 각 단계는 정의된 산출물을 통해 상호 연관관계를 유지하며, 필요에 따라 임의의 단계로부터 절차를 시작하거나, 동일 단계를 반복적으로 수행한다. 보편적인 임베디드 S/W에 기준을 맞추어 세부적인 단계의 내용이 구성되어 있으며, H/W에 대한 내용은 제한적으로 포함되어 있다.

개별 작업들은 자산의 형태에 따라 선택적으로 자산 개발 시나리오를 정의할 수 있다. 예를 들어 자산이 단일 메소드의 형태라면 자산 설계의 인터페이스 명시와 코딩, 단위테스트 작업만 수행하게 되고, 만약 여러 모듈의 집합적 자산이라면 통합설계를 통해 자산의 S/W 관점과 H/W 관점을 분리하고 S/W 관점을 중심으로 개별 모듈과의 인터페이스 정의를 위한 통합 설계 작업이 필요하다.

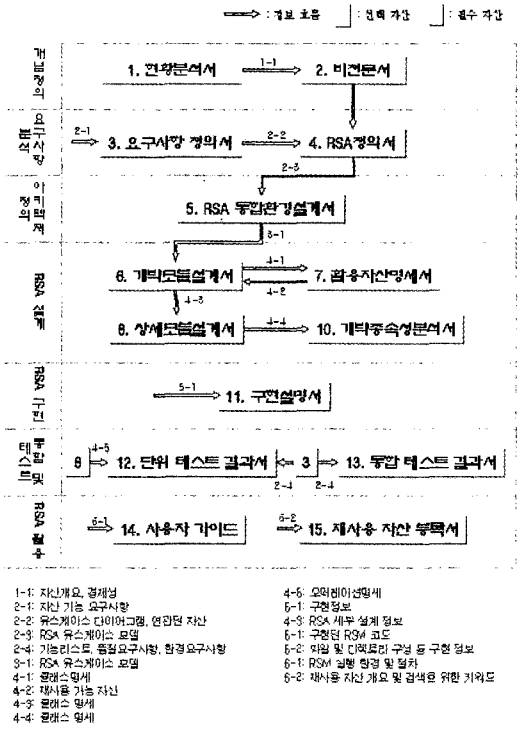
만약 기존 라이브러리에 재사용 자산을 추가할 경우는 개발될 자산의 아키텍처가 미리 정해져 있어 그에 따라 구현하게 된다. 따라서 기존 라이브러리에 포함시킬 수 있는지 결정하는 인수 테스트가 필요하다. 그리고 기존 자산과의 연관성이 없는 독립 자산을 만들 경우는 개발할 자산이 이미 존재하는가에 대한 사전 분석 작업을 수행하고, 단독 자산이므로 아키텍처 생성 작업이 필요 없다. 또한 임베디드 S/W 개발 과정에서 우연히 발생한 재사용 요구사항을 자산화할 경우에는 재사용 자산만을 위한 정확한 명세와 구현 방법을 결정하는 통합 설계 및 통합 테스트 작업들이 필요하고, 도메인 한정적인 재사용 자산의 집합을 생성하기 위한 경우라면, 재사용 자산의 토플로지 설정을 위한 아키텍처 설계 작업과 개발된 개별 자산들 간의 통합 테스트의 작업들이 필수적으로 요구된다.

3.4 임베디드 S/W 재사용 자산들 간의 정보 흐름

임베디드 S/W 재사용 자산 개발 절차 수행 과정에서 생성되는 산출물들은 정제와 상세화 작업을 통해 일관성 있는 정보 흐름을 가진다. 즉, 재사용 자산 요구 고객의 비정형화된 개념적인 요구사항들이 분석, 설계 그리고 구현이라는 일련의 규정된 작업의 규칙에 따라 가치 높은 재사용성의 물리적인 재사용 모듈로 변환된다.

(그림 5)는 임베디드 S/W 재사용 자산 개발과정에 수반되는 다양한 산출물 간의 관계성을 도식화 한 것이다. 진한 색 사각형은 재사용 자산 생성 과정 중에 반드시 만들어야 할 산출물을 의미하고, 옅은 색 사각형은 필수적이지는 않지만 재사용 자산 생성에서 원활한 정보 흐름을 지원하고 재사용 자산의 풍부한 정보 생성을 위한 산출물이다.

특히 이들 산출물의 흐름도는 (그림 4)에서 정의한 개발 절차를 기반으로 하는 재사용 자산 개발 시나리오의 일반화한 경우이다. 따라서, 재사용 고객의 요구에 따라 재사용 개발 절차는 맞춤화 될 수 있으며 이에 따라 생성된 산출물 범위 역시 달라지게 된다.



(그림 5) 임베디드 S/W 자산간의 정보 흐름

3.5 임베디드 S/W 재사용 문서 자산

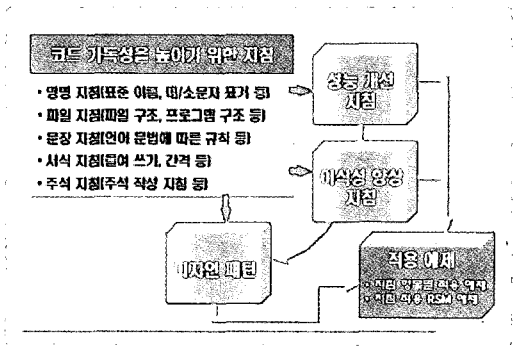
문서 자산은 자산 수요자가 자산을 활용하기 위한 이해 정보 제공이 목적이다. 따라서 자산 이해에 필요한 충분한 정보를 제공해야 한다. 하지만 자산 공급자가 문서 자산 작성에 너무 많은 비용을 소요한다면 자산의 수요, 공급이 원할지 않을뿐더러 궁극적으로 자산 공급원들의 축소를 초래할 수 밖에 없다. 따라서, 문서 자산의 최적화가 필요하다. 즉, 꼭 필요한 정보만을 최소한의 분량으로 체계화시켜 간소하게 제공 해야만 한다.

본 논문에서는 임베디드 S/W자산의 형태와 개발 절차 별로 자산의 종류를 필수/선택 자산으로 구분하고 각 문서 자산들이 포함해야 할 정보들의 속성들을 정의하였다.

<표 2>는 재사용 자산 개발 단계별 산출물과 각 단계별 문서의 필수/선택 작성 속성을 정리하여 간략하게 나타낸 것이다.

<표 2> 재사용 자산 개발 절차 별 산출물 분류

단계별 산출물	RSA 종류	어플리케이션	단일 함수	함수 집합	단일 클래스	클래스 집합	정형 API
개념 정의	현황 분석서	△	△	X	X	X	X
	비전 문서	△	△	X	X	X	X
요구 사항 분석	요구사항 정의서	O	O	△	△	△	△
	자산 정의서	O	O	O	O	O	O
아키텍처 정의	통합 환경 설계서	O	X	△	X	△	O
RSM 분석 및 설계	개략 모듈 설계서	△	△	△	△	△	△
	활용 자산 명세서	△	△	△	△	△	△
	상세 모듈 설계서	O	O	O	O	O	O
	개략종속 성분석서	△	△	△	△	△	△
RSM 구현	RSM 코드	O	O	O	O	O	O
	구현 설명서	O	O	O	O	O	O
통합 및 테스트	단위테스트결과서	O	O	O	O	O	O
	통합테스트결과서	X	△	△	△	△	△
자산 활용	재사용자 산등록서	O	O	O	O	O	O
	사용자 가이드	O	△	O	△	O	O



(그림 6) 임베디드 S/W 재사용 자산 코딩 지침

3.6 코딩 지침

재사용 자산의 가독성을 증대 시키고 자산의 이식성과 재사용성 향상을 위해 명명규칙과 서

실 및 주석 규칙 등의 일반적인 코딩 규칙을 제공을 위한 본 논문의 재사용 지침에서는 일반적인 임베디드 S/W 개발을 위한 코딩 지침을 정의하였다. 또한 효율적인 코드 구조를 디자인 패턴으로 제시하는 코딩 지침을 정의하였으며 지침의 효율적 적용을 위해 예제 코드로 함께 포함하여 제공한다.

본 코딩 지침은 C와 Java 두 가지 언어로 제공되며 일반적인 코딩 원칙과 특정 품질 속성을 위한 프로그램 구조 등으로 구성되어 있다. 특히, 코딩 과정에서 적용할 수 있는 명명규칙, 주석 작성 방법, 코드의 구조 및 서식 등으로 분류하여 설명하였으며 각각의 코딩 방법은 이해를 돕기 위해 예제와 함께 설명한다. 그리고 임베디드 S/W에서 중요한 문제인 이식성과 성능을 향상시키는데 참조할 수 있는 코드 작성 원칙도 포함한다.

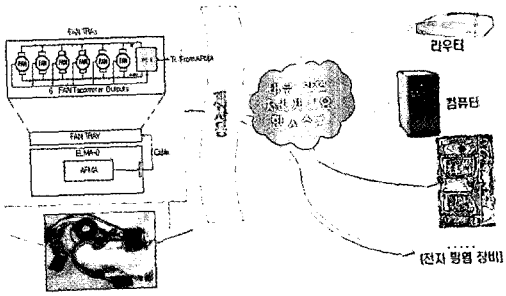
(그림 6)은 본 논문에서 정의한 임베디드 S/W 재사용 자산 개발을 위한 코딩 지침의 전체 구성을 도식화 한 것이다.

4. 임베디드 S/W 자산 개발 시범 사례

본 논문에서 정의한 임베디드 S/W 재사용 지침의 유용성을 검증해보기 위하여 임베디드 시스템에서 재사용 빈도가 높은 FCM (Fan Control Module)을 대상으로 재사용 자산 생성 작업을 수행하였다. FCM은 컴퓨터나 라우터 장비 내부의 온도나 습도를 조절하기 위하여 팬(FAN)의 동작을 관리하는 S/W이다. (그림 7)은 FCM의 내부 구조 및 물리적인 형상, 재사용 분야 등을 보여준다.

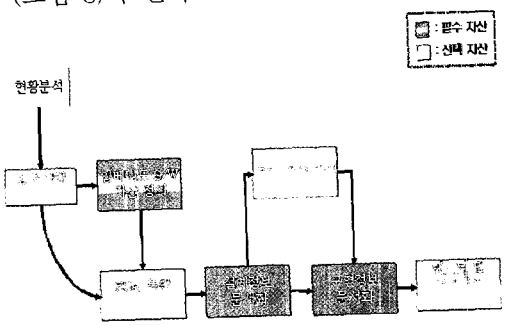
본 장에서는 FCM을 임베디드 S/W 재사용 지침에 따라 FCM을 재사용 자산으로 생성해 나가는 과정 및 결과를 제시한다.

임베디드 S/W 재사용 지침 중 3.5 장에서 정의한 문서자산 생성 지침을 적용하여 FCM을 사용하는데 필요한 핵심적인 문서자산을 생성하였다.



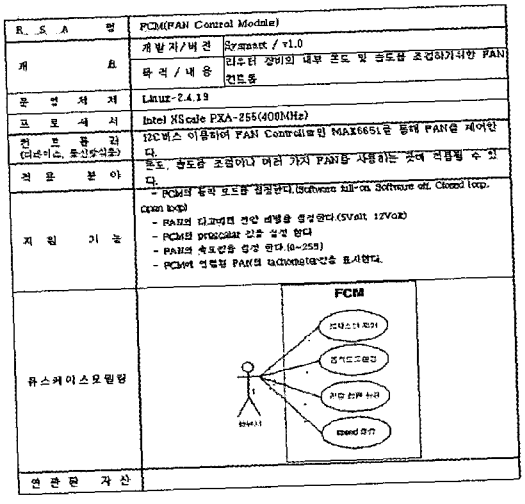
(그림 7) FCM 구조 및 재사용 시나리오

FCM을 재사용 자산으로 생성하기 위하여 적용한 임베디드 S/W 재사용 지침에 정의된 절차는 (그림 8)과 같다.



(그림 8) 핵심 문서 자산의 개발 절차

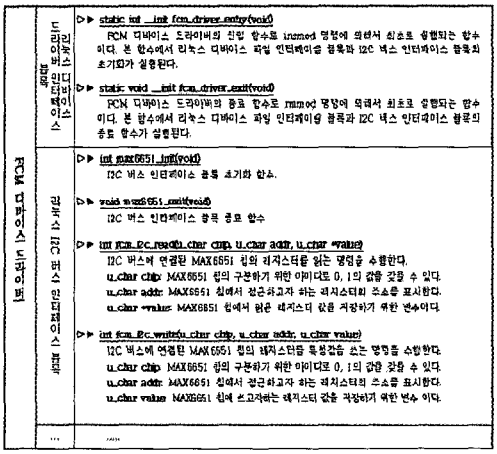
(그림 8)에 따라 임베디드 S/W 자산 정의, 설계 정보 문서화, 구현 정보 문서화 단계에서 생성된 FCM 재사용 자산 관련 문서들의 예는 다음과 같다.



(그림 9) FCM의 재사용 자산 정의서 예

4.1 설계 정보 문서화

이 단계에서는 FCM RSA의 상세 구조와 행위를 명세화 하고, 구성요소간 인터페이스를 상세히 기술한다.



(그림 10) FCM 상세 모듈 설계서 예

4.2 구현 정보 문서화

이 단계에서는 자산 개발에 영향을 준 모든 결정 사항들에 대한 정보를 명세화 한다.

(1) 개발 환경 정보

개발환경정보	운영체제/언어/컴파일러	Linux 2.4.20/C/arm-linux-gcc
	프로세서	MX21(ARM926EJ)
	개발 보드	자체 개발
	메모리	SDRAM 64MB
	Ethernet	10/100Mbps Ethernet
	Serial	UART 1 Slot
	Fan controller	MAX6651
활용 자산 정보	라이브러리	FCM API library

(2) 파일 구조

디렉토리	파일	
	파일명	파일 기능 설명 주요 함수
/fcm/bin	FCM 소프트웨어의 디바이스 드라이버 모듈 및 응용 소프트웨어 이미지가 위치하는 디렉토리	
	driver.o	FCM 디바이스 드라이버 이미지파일
	fcm_app	FCM 시험 소프트웨어 이미지 파일
	up	FCM 디바이스 드라이버 이미지 커널 로딩 스크립트
	down	FCM 디바이스 드라이버 이미지를 커널으로부터 제거하는 스크립트
/fcm/cfg	FCM 응용 소프트웨어에서 설정 파일에서 옵션 항목을 읽을 때 사용하는 라이브러리 소스 디렉토리	
/fcm/dbg	FCM 소프트웨어에서 사용하는 디버깅 용 출력 함수 및 매크로 관련 라이브러리	
/fcm/includes	FCM 소프트웨어 모듈에서 공통으로 사용하는 헤더 파일 디렉토리	
	fcm_api.h	FCM API 라이브러리 함수 인터페이스 헤더 파일
/fcm/driver	FCM 디바이스 드라이버가 위치하는 디렉토리	
	fcm_drv.h fcm_drv.c	리눅스 디바이스 드라이버 등록 구현 함수의 소스파일
	fcm_fops.h fcm_fops.c	리눅스 디바이스 파일 인터페이스 등록 구현 함수의 소스 파일
	fcm_i2c.h fcm_i2c.c	리눅스 I2C 버스 인터페이스 등록 구현 소스 파일 fcm_i2c_read(), fcm_i2c_write()
.....		

(그림 11) FCM의 구현 설명서 예

또한 임베디드 S/W 재사용 지침 중 C 코드의 재사용성, 가독성, 유지보수성을 향상시키기 위하여 작성된 "C 코딩 지침"을 적용하여 FCM 코드를 작성하였다. FCM 코드는 크기가 크진 않지만 다양한 분야에 반복적으로 재사용되므로 가독성을 높기기 위하여 코딩 지짐중 명명규칙과 주석 규칙, 그리고 이식성 향상 지침 부분을 주로 적용하였다.

5. 결론 및 향후 연구

디지털 컨버전스 시대로의 진입을 위한 정보기술의 핵심은 임베디드 S/W의 생산성이다. 다양한 장비에 다양한 서비스를 급격하게 변화는 고객 요구에 맞도록 지원하고, 점점 단축되는 S/W의 생명주기에 효율적으로 대처하기 위해서는 임베디드 S/W의 공동 자원을 구축하고 이를 적절하게 공유하는 체계 구축이 필요하다.

그러나, 큰 규모의 SI 업체 및 연구 조직이라 하더라도 대부분이 일시적으로 특정 목표 시스템 개발에 집중함으로써 임베디드 S/W 공통 기술 개발을 위한 협력이 매우 미흡하다. 따라서 이미 수행된 과제의 개발 경험이나 산출물들은 고작 후속과제에 연계되는 수준에서만 재사용만 가능한 실정이다. 그러므로 적시에 즉각적인 S/W 해결책을 제시해야 하는 임베디드 S/W 개발 시 유사한 목표 시스템 개발을 위해 반복적인 노력과 비용이 소모되는 비생산적 구조가 지속되고 있다.

본 논문은 임베디드 S/W의 재사용성 향상을 위한 실질적인 방법 제시를 목적으로 작성되었다. 즉, S/W의 문서화 기능 최적화함으로써 재사용 자산의 가독성을 최대화 하고, 체계적 재사용 자산 모듈 개발 기법을 정의하여 임베디드 S/W 개발자들을 위한 실질적 지침으로 활용하는 것을 목적으로 한다. 이를 위해 임베디드 S/W 재사용 자산의 개발을 위한 재사용 자산을 정의하고, 이들 자산을 체계적으로 개발하기 위한 프로세스와 산출물들을 자산의 형태에 따라 분류하

여 제시하였으며, 실제 문서 자산들을 FCM 자산 생성을 위해 적용한 예를 제시하였다.

본 논문에서 지시하는 임베디드 S/W 재사용 지침은 S/W 재사용 문화를 확산시킴으로써 조직의 생산성 향상을 위한 작고 실제적인 재사용 템플리트를 보급하고 적용하는 것을 목표하여 개발되었다. 따라서 재사용 지침을 통해 중복 개발 비용 최소화 및 개발 경험 공유로 조직 내 S/W 자산의 가치를 지속적으로 증대시킬 수 있을 뿐 아니라 개발자가 동의하는 재사용 필수 항목들만을 양식화하여 초기 자산 개발 비용은 보다 많이 요구될 수 있는 지침 적용의 부담을 최소화시킬 수 있으리라 기대된다.

향후 특정 영역에 대한 재사용 자산 카탈로그를 작성하고 재사용자들의 효과적인 활용을 지원하기 위한 라이브러리 시스템을 구축함으로써, 재사용 지침에 대한 정량적인 평가 데이터 구축 및 재사용 지침의 best practices 개발하여 재사용 기술 확산 전략을 수립할 계획이다.

참고문헌

[1] McGary, F., G. Page, et al. 1984, 'An Approach to Software Cost Estimation', NASA Software Engineering Laboratory, SEI-83-001(Feb.)

[2]BiggerStaff, Ted, and Alan J. Perlis, 1989, Software Reliability, New York: ACM Press

[3] 나종화, 강순주. 임베디드 시스템 프로그래밍 : 이론 및 실습, 사이텍미디어, 2004년 10월

[4] David E. Simon, 입문자를 위한 임베디드 시스템, 사이텍미디어, 2003년 11월

[5] KOSTA, "S/W 프로젝트 수행 환경 및 생산성에 관한 실태 조사", 20061

[6] 삼성전자, DTV Software Platform, 한국정보과학회 컴퓨터시스템 연구회 워크샵 자료, 2006.2.

[7] 임베디드SW산업협의회 45개회원사 설문 조사 결과
(04.12.15~05.1.15)

[8] Jorge L. D?az-Herrera† and Vijay K. Madisetti, The Yamacraw Embedded Software(YES) Methodology: A Technical Analysis", Yamacraw (YES) Technical Report CSIP-TR-00-01, 1/31/2000., Georgia Tech, 1999.

[9] 차정은, 양영종, "임베디드 소프트웨어 재사용을 위한 문서 자산의 분류", 한국정보처리학회, 추계학술대회, 2005. 11

저자약력

차 정 은

1995년 효성여대 전자계산학과 졸업(이학사)
1997년 대구효성가톨릭대학교 대학원 전자계산학 졸업
2001년 대구효성가톨릭대학교 대학원 전자계산학 졸업(이학박사)
2001년 ~ 현재 ETRI 임베디드 S/W연구단 S/W공학팀 선임연구원.
2006년- 현재 ETRI 임베디드 S/W연구단 S/W공학팀 팀장.
관심분야: 컴포넌트기반 개발, 임베디드 소프트웨어, 재사용, 아키텍처 설계, 재공학

유 미 선

1999년 충남대학교 컴퓨터공학과 졸업
2002년 포항공대 대학원컴퓨터공학과 졸업
2002년- 현재 ETRI 임베디드 S/W연구단 편재형 컴퓨터미들웨어팀 연구원
관심분야: 임베디드 소프트웨어, 재사용, 센서 네트워크, S/W 개발 프로세스