DOI: 10.3745/KIPSTD.2009.16D.5.767

의존과 관점 기반 임베디드 시스템의 요구사항 우선순위 프로세스

황 위 용[†] · 강 동 수[††] · 송 치 양[†††] · 성 재 석[††††] · 백 두 권[†††††]

요 약

임베디드 시스템을 위한 릴리스 플랜 수립은 제품 개발 초기 요구사항 분석 단계에서 요구사항의 우선순위 결정을 통해 달성되므로 우선순위 결정은 매우 중요하다. 이때, 우선순위 활동에서는 요구사항간 의존관계와 제품 개발에 관여하는 관점들을 고려해야 한다. 특히 임베디드 시스템에서 하드웨어와 소프트웨어의 특징에 기반한 아키텍처 스타일에 따른 요구사항간 상충관계는 반드시 반영해야 한다. 그러나, 기존 연구에는 하드웨어 및 소프트웨어로 구성된 시스템에 대한 요구사항을 고려하는 우선순위 선정 프로세스가 체계적이지 못하다. 따라서, 본 논문에서는 임베디드 시스템을 위한 요구사항간 의존관계, 관점에 기반한 요구사항 우선순위의 모델과 프로세스를 제안한다. 이를 위해 아키텍처 스타일에 따른 우선순위 요소 또는 요구사항간 상충관계를 분석하고, 임베디드 시스템 제품 개발에 관여하는 관점들을 식별하여 요구사항 우선순위를 정립한다. 그리고 제안된 방법을 핸드폰 개발 사례의 요구사항 우선순위에 적용하여 신뢰성을 보인다. 본 논문의 기법을 적용하여 임베디드 시스템의 특성에 부합하도록 요구사항을 좀 더 명확하게 최적화하여 우선순위화함으로써 제품 릴리스에 대한 위험성을 최소화할 수 있다.

키워드 : 요구사항 우선순위, 임베디드 시스템, 상호의존 관계, 관점, 요구공학

A Requirement Priority Process of Embedded Systems based on the Dependency and Aspect

Wiyong Hwang[†] · Dongsu Kang[††] · Cheeyang Song[†††] · Jaeseok Seong[††††] · Dookwon Baik[†††††]

ABSTRACT

Setting up a priorityfor an embedded system is greatly significant because a release plan at the early stage of product developments can properly be established through right decision making procedures based on the priorities. For instance, both dependencies among requirements and the aspects of product developers should be considered into the priorities to improve the embedded system. Especially, trade-offs among the requirements, which are quite different depending on H/W and S/W architecture styles they use, should be acknowledged without exception. However, the selection process on the priority has hitherto been fairly systematic in the existing environment where hardware and software are not being considered at once. Therefore, this paper suggests an dependency and aspect-based model and process for the requirements of the priority. For this, the paper analyzes the trade-offs between the requirements depending on the disparate Architecture styles of H/W and S/W, and it also reflects the viewpoints of the developers. For thelast thing, the model and process suggested will be applied to the case of the development of both cell phones and cameras to gain authenticity and reliability. In conclusion, the danger occurring when the release plan is constructed can be minimized by screening the priorities that optimizes the embedded system more explicitly.

Keywords : Requirement Priority, Embedded System, Dependency, Aspect, Requirement Engineering

1. 서 론

최근 들어 임베디드 시스템은 그 응용 분야 및 시장 규모가 빠르게 변화되면서, 하드웨어나 임베디드 소프트웨어에 관한 요구의 다양화, 기능의 복잡화, 대규모화는 피할 수 없는 상황이다[1, 2]. 특히 임베디드 시스템 제품 시장에서 유사 제품간 경쟁은 점차 치열해 지는 추세로써 제품 경쟁력

※ 이 연구에 참여한 연구자는 '2단계 BK21 사업'의 지원을 받았음.
본 과제는 한국소프트웨어진흥원의 SW공학 요소기술 개발과 전문인력 양성사업의 결과물임.
† 준 회 원 : 고려대학교 컴퓨터·전파통신공학과 석사과정
†† 준 회 원 : 고려대학교 컴퓨터·전파통신공학과 박사과정
††† 정 회 원 : 경북대학교 소프트웨어공학과 조교수
†††† 정 회 원 : LG전자 MC연구소 책임연구원
††††† 종신회원 : 고려대학교 컴퓨터·전파통신공학과 교수
논문접수 : 2009년 7월 9일
수 정 일 : 1차 2009년 8월 7일, 2차 2009년 8월 31일
심사완료 : 2009년 8월 31일

확보를 위한 릴리스 플랜이 중요하다[3, 4]. 따라서, 임베디드 시스템 제품 개발은 주어진 자원과 시간의 제약으로 인하여 모든 요구사항을 만족시킬 수가 없기 때문에 계획된 일정 안에 제품을 출시하기 위하여 요구사항을 우선순위화하고, 릴리스 플랜(Release plan)을 수립 하여 여러 릴리스로 요구사항을 나누는 활동의 중요성이 증가하고 있다[5-7].

우선순위와 릴리스 플랜 관련하여 다양한 분석 연구[6-9]가 이루어지고 있다. 우선순위에 대한 기존 연구는 우선순위 자체에 초점을 맞춘 연구와 릴리스 플랜을 위한 우선순위 연구로 구분할 수 있다.

전자인 우선순위에 중점을 둔 연구에서 요구사항이 독립적이라는 가정아래 각각 개별적으로 특정한 관점(예를 들어, 가치, 비용, 중요성, 위험성 등)만을 고려한 우선순위 방법론을 제안하고 있어서 다양한 상황을 종합한 요구사항 우선순위 관점을 반영하지 못한다[10-14]. 후자인 릴리스 플랜을 위한 우선순위 연구는 요구사항간 상호 영향 분석에 초점을 두고 상호의존 관계를 고려하여 요구사항의 우선순위를 결정하는 방법을 제안하고 있으나[15-20], 아키텍처 스타일에 따른 상충관계(임베디드 시스템의 구성을 위한 하드웨어와 소프트웨어 요소간 역할분담에 따른 우선순위 요소 또는 아키텍처 요구사항간 상충관계)는 고려하지 않는다.

경쟁력 있는 임베디드 시스템 제품 개발의 핵심은 고객을 만족시키는 제품을 적기에 출시되도록 기능 요구사항 과 아키텍처 요구사항의 의존성을 고려한 우선순위화 및 릴리스 플랜 수립이다. 이를 위해 요구사항간 의존관계를 정의하여 요구사항의 의존성을 분석하고, 임베디드 시스템 개발에서 영향관계에 있는 관점들에 기반하여 요구사항의 우선순위를 결정하는 활동이 요구된다. 그러나, 기존의 기법들은 소프트웨어에 한정된 기능 요구사항과 우선순위 관점에서 소프트웨어에 관련된 중요성, 가치, 위험성 등과 같은 제한적인 관점만을 고려하고 있어 하드웨어와 소프트웨어 요소를 모두 고려한 시스템의 우선순위 프로세스를 제공하지 못하고 있다. 결국, 기존 연구들은 제한적인 관점 그리고 기능 요구사항에 한정된 범위로 요구사항의 우선순위를 다루기 때문에 아키텍처와 관련한 요구사항을 고려하고, 다양한 관점과 의존관계에 대한 분석이 결합된 임베디드 시스템 개발에 적합한 요구사항 우선순위 프로세스를 제공하지 못한다.

본 논문은 임베디드 시스템의 특징을 반영하여 요구사항간 의존성과 이해당사자의 관점에 기반한 요구사항의 우선순위를 결정하는 모델과 프로세스를 제시한다. 먼저, 요구사항 우선순위 모델은 우선순위에 영향을 미치는 요소들을 가지고 계층적 구조로 정의한다. 우선순위화 프로세스는 시스템 요구사항을 정의하고, 요구사항간 의존관계를 분석하고, 관점 기반의 요구사항 우선순위화 과정으로 구축한다. 요구사항간 의존관계는 정적 및 동적 의존, 구현 선후 관계 그리고 상충관계를 가지고 분석한다. 관점은 이해당사자의 비즈니스, 고객, 개발 측면에서 임베디드 시스템 개발에 영향을 주는 적시성, 개발 비용, 개발 시간, 기술적 위험 등에 의해 기준을 정하여 우선순위에 반영한다.

따라서, 요구사항 우선순위화는 제품의 목표에 부합한 관점을 선정하고, 디자인 매트릭을 사용해서 요구사항간 의존관계를 분석하고, 가중치가 부여된 관점에 대한 기준을 정량적으로 정의하며, 분석된 의존관계와 관점별 기준에 의해 요구사항간 우선순위를 결정한다. 즉, 참여 이해당사자는 요구사항간 분석된 의존정보, 선정된 요구사항 우선순위 관점과 그에 대한 가중치에 기반하여 관점별 기준을 세우고 우선순위화를 실시한다. 이러한 임베디드 시스템의 의존성에 초점을 둔 요구사항 우선순위 프로세스의 정립을 통하여 보다 충족스러운 요구사항의 우선순위 결과를 기대할 수 있으며, 제품 개발 목표와 개발 조직의 상황에 맞는 최적화된 요구사항 선택으로 위험이 최소화된 릴리스 플랜의 수립이 가능해 진다.

본 논문은 다음과 같이 구성되어 있다. 2장은 관련연구로 소프트웨어 시스템과 비교를 통하여 임베디드 시스템의 특징 그리고 의존관계 및 관점에서 우선순위와의 관계를 분석하고, 3장은 본 연구 접근의 설계 원리 및 모형을 기술한다. 4장에서는 우선순위에 영향을 주는 요소들로 구성된 요구사항 우선순위 모델을 정의한다. 5장은 의존 관계와 관점 기반의 임베디드 시스템 우선순위 프로세스를 제시한다. 6장에서는 핸드폰 개발에 제안된 방법을 사례 적용하고, 분석 및 평가를 실시한다. 마지막으로 7장에서 결론 및 향후 과제를 기술한다.

2. 관련 연구

요구사항 우선순위화를 위해 요구되는 임베디드 시스템의 특징을 살펴보고 이 특징을 고려하여 요구사항간 의존관계 및 이해당사자의 관점과 요구사항 우선순위의 관계를 기술한다.

2.1 임베디드 시스템의 특징

일반적인 임베디드 시스템의 개발 과정은 제품 명세 및 개념적인 수준의 아키텍처 스타일 분석, 상세 설계 및 구현, 통합 테스팅 및 유지보수의 3단계로 구성되며, 세부적으로 요구사항 분석, 시스템 요구사항 정의, 하드웨어와 소프트웨어의 역할 분담, 소프트웨어와 하드웨어의 상세 설계 및 구현, 통합 시뮬레이션, 통합 테스팅, 제품 릴리스, 유지보수 및 진화와 같이 8단계로 이루어진다[21]. 특히, 제품 명세 및 설계 결정 단계에서는 일차적으로 프로젝트 기획의 연장선상에서 목표 시스템의 주요성능(수행성능, 메모리 사용량, 시스템 크기 및 무게 등) 혹은 시장(제품당 가격, 출시 시기 등) 목표 및 제약사항을 정확히 설정하고, 이차적으로 프로젝트 개발 참여자들간의 이해 및 합의를 동시에 수행하면서 릴리스(출시)를 위한 개발 제품의 요구사항을 선정한다[5].

따라서, 임베디드 시스템은 기본적으로 하드웨어와 소프트웨어 요소를 함께 고려해야 한다. 그래서 이해당사자들은 어떤 기능 요구사항을 하드웨어로 해결해야 되고 어떤 부분을 소프트웨어로 해결해야 하는지를 최우선으로 결정 해야

한다[22]. 다시 말하면, 하드웨어와 소프트웨어의 역할 분담이 임베디드 시스템의 개념적 수준의 아키텍처가 된다. 또한, 하드웨어와 소프트웨어의 역할 분담 문제는 정확한 해법이 있는 것이 아니고 단지 주어진 자원 및 제약사항 내에서 시스템의 최적화 구성을 위한 문제로서 우선순위를 통해 제품 목표 및 상황에 맞추어 결정하는 것이 바람직하다. 이와 같이 임베디드 시스템의 개발은 하드웨어가 가미됨으로써 일반적인 소프트웨어 시스템 개발에서 나타나지 않는 하드웨어와 소프트웨어의 역할 분담 및 동시 설계와 같은 작업활동에 차별성을 가진다. 그래서 임베디드 시스템의 요구사항 우선순위화 활동은 요구사항 분석 과정에서 하드웨어와 소프트웨어의 역할 분담에 관계 한다.

임베디드 시스템의 특징은 <표 1>과 같이 다양한 측면에서 비교할 수 있다[2-5]. <표 1>에서 비교항목들은 요구사항 우선순위화에 영향을 미치는 인자들로Primary Goal, Main Stakeholders, Requirements Conception, Life Cycle, Specific RE issues, Business, User, Development View로 식별하였다[2-5]. <표 1>의 요소들은 4장에서 요구사항 우선순위 모델을 설계하는데 기본적인 요소들로 반영된다. 오픈 시장에서 임베디드 제품이 가지는 특징은 첫째로 제품으로 출시되는 경우 불특정 다수인 임의의 고객을 대상으로 시장에서 유사제품간 치열한 경쟁을 한다는 것이며, 둘째로 제품 개발의 경우 생명 주기(Life cycle)는 여러 연속적인 릴리스를 가진 생명주기를 가지고, 시장이 있는 한 지속된다는 것이며, 셋째로 하드웨어와 소프트웨어 요소를 모두 포함하며 다양한 응용 분야를 가진 시스템이라는 것이다. 이러한 점으로 보아 요구사항의 우선순위화를 통한 전략적인 제품 개발이 중요함을 알 수 있다. 또한 제품 개발을 위한 후보 요구사항은 매우 많을 수 있고, 지속적으로 추출되기 때문에 우선순위, 비용 예측에 좀 더 노력을 기울이며 이러한 활동은 개발 조직에서 대부분 이루어지게 된다[23]. 따라서, 임베디드 시스템을 제품으로 출시하기 위해 우선순위 활동에서 고려해야 할 특징은 다음과 같다.

- 첫째, 아키텍처 관련된 위험, 신 기술과 중요 요구사항의 누락, 그리고 품질과 관계된 위험이다. 이에, 초기에 심각한 위험 요소를 제거 또는 완화시켜야 하기 때문에 위험성과 아키텍처를 고려하여 요구사항을 우선순위화 해야 한다.
- 둘째, 시스템 요구사항 레벨의 기능 요구사항과 기능 요구사항을 이루는 서브 시스템 레벨의 아키텍처 요구사항을 포함하는 요구사항간 존재하는 의존성이다. 즉, 요구사항간 의존관계는 동일 계층 레벨에서의 요구사항간 의존관계와 계층 레벨이 다른 요구사항간 의존관계가 존재한다. 이러한 요구사항간 의존성은 시스템의 설계 및 구현에 상당한 영향을 미치게 되므로 요구사항의 우선순위를 결정할 때 이 같은 사항이 반영되어야 우선순위에 따른 제품의 개발에서 위험을 줄이고 구현이 쉬워진다. 따라서 우선순위 이전에 단위 요구사항의 독립성을 높이기 위해 의존관계를 파악해야 하며, 결합도(Coupling)가 높은 요구사항들은 바인딩(Binding) 시켜야 한다[20, 24].
- 셋째, 아키텍처 스타일에 의한 우선순위 요소 및 아키텍처 요구사항간 존재하는 상충성이다. 이러한 상충성은 하드웨어와 소프트웨어 요소가 기능 요구사항과 매핑이 되고, 역할 분담이 이루어 지면서 발생한다. 이에, 하드웨어와 소프트웨어 요소가 가진 특징이 개발 시간, 개발 비용, 재사용성, 유연성, 성능과 관계되는 처리 속도, 전력소모와 같은 측면에 영향을 미치게 된다[2, 3, 21]. 따

〈표 1〉 소프트웨어 시스템과 임베디드 시스템의 비교

Facet	소프트웨어 시스템	임베디드 시스템
Primary Goal	- 전형적으로 사용자의 요구충족 및 정보처리	- 전형적으로 특정 기능만을 수행
Main Stakeholders	- 주 이해당사자는 고객 조직	- 주 이해당사자는 개발 조직
Requirements Conception	- 요구사항이 고객을 중심으로 추출, 분석, 검증됨	- 요구사항이 개발조직에서 발명됨
Life Cycle	- 다양한 개발 프로세스 존재	- 반복 점증적 개발 프로세스
Specific RE issues	- 요구사항의 추출, 모델링, 검증, 갈등 해결에 초점	- 지속적인 요구사항 수집, 우선순위화, 비용 예측, 릴리스 플랜 수립에 초점
Business View	- 포괄적인 범용 S/W - 개발의 보편화 - 실시간성, 고신뢰성, 자원 제약성 등이 중요하게 요구되지 않음 - 소프트웨어만 개발 (S/W 지식 및 경험 요구) - 운용환경의 보편화 (H/W, OS 등)	- 특정 제품에 종속된 S/W (H/W나 OS에 종속적) - 개발의 보편화가 이루어지지 않음 - 실시간성, 고신뢰성, 자원 제약성 등이 중요하게 요구됨 - 소프트웨어, 하드웨어 모두를 개발 (S/W, H/W 지식 및 경험 요구) - 운용환경의 보편화가 이루어지지 않음 (같은 기능이지만 H/W, OS에 따라 다르며, 이식이 필요)
User View	- 사용자의 상호작용은 GUI를 통해 발생 - 알려져 있거나 식별된 사용자 - 고장 발생시 쉽게 유지 보수	- 사용자의 상호작용은 최종 제품을 통해 발생 - 주로 고객은 불특정 다수로 정해지지 않음 - 고장 발생시 제품 사용이 불가
Development View	- 고객의 요구사항을 만족시키는 것이 주 목표 - 패키지 소프트웨어 시장이 커지는 추세	- 시장 적시성(Time-to-Market)을 만족시키는 것이 주 목표 - 시장에서 유사 제품 경쟁

라서 임베디드 시스템은 상기 특징들에 대해 매우 민감한 특성이 있기 때문에 이해당사자는 우선순위 이전에 분석을 통하여 인지하고, 이를 우선순위화에 고려해야 한다.

- 넷째, 다양한 응용 분야에 따른 개발 목적 및 상황을 고려할 수 있는 우선순위 프로세스의 유연성이다. 임베디드 도메인은 다양한 응용 분야를 포함하고 있기 때문에 응용 분야에 따라 개발 상황과 목적에 차이가 있으며, 개발조직은 상기 차이점들을 인지하고 조직의 역량에 맞추어 우선순위 활동을 수행하는 것이 바람직하다.

2.2 요구사항 우선순위와 상호의존 관계

대부분의 임베디드 시스템 개발 프로젝트는 자원의 제약을 가지고 있다. 결국, 정의된 기능 요구사항을 달성하기 위해서는 요구사항간 의존관계 파악과 아키텍처 스타일에 따른 개발 시간, 개발 비용 등과 같은 민감한 우선순위 요소와 아키텍처 요구사항간 상충의 상대적인 우선순위를 고려하여 후속 개발 프로세스에서 위험을 최소화하는 것이 중요한 이슈이다.

의존관계란 요구사항간 또는 구현된 시스템과 새로운 요구사항간의 기능 측면의 정적, 동적 의존관계를 말한다. 일반적으로 요구사항 중에 20%만이 다른 요구사항에 독립적이라고 한다[15]. 따라서 요구사항은 상호 연관되어 있기 때문에 요구사항을 선택 시에 의존관계를 고려하지 않고 요구사항의 우선순위를 정하는 것은 적합하지 않다. 우선순위화를 위한 의존관계에 대한 관련 연구로 시스템 구조상의 결합(Coupling)에 의한 정적 의존관계, 시스템 구현시의 구현 선후 관계(Precedence), 시스템 운영시의 동적 의존관계를 정의하고 있다[6, 19].

Adaptive requirements prioritizing기법[6]은 소프트웨어를 대상으로 개발 조직의 상황에 따라 적절한 기존의 우선순위 방법[10-14]을 유연하게 선택하여 요구사항을 우선순위화하는 프로세스를 제시했다. 그러나, 요구사항의 도메인을 패키지 소프트웨어로 한정하여 우선순위 측면에서는 하드웨어 요소 및 이와 관련된 요구사항을 다루지 못했으며, 의존관계 측면에서는 임베디드 시스템에서 필수적으로 고려해야 하는 하드웨어와 소프트웨어 요소의 특징에 기반한 아키텍처 스타일 상충관계 분석이 반영되지 않았다. 즉, [6]은 임베디드 시스템이 아닌 일반적 소프트웨어 시스템에 부합하는 우선순위 프로세스를 제시한 것이다. 이러한 문제점은 우선순위화를 위해 하드웨어 요소 및 이와 관련된 요구사항을 고려하고, 아키텍처 스타일에 의한 우선순위 요소 또는 아키텍처 요구사항간 상충관계를 정의하여, 기 제시된 우선순위 모델과 의존관계를 확장하는 방법으로 해결 할 수 있다. 기존 우선순위 및 상호의존에 대한 연구들의 상호 비교 결과[6]는 <표 2>와 같으며, 본 논문에서 [6]은 ARP (Adaptive requirements prioritizing)로 약칭 한다.

일반적으로 요구사항 분석과 설계 사이에는 뚜렷한 경계가 존재하는 것이 아니라 오히려 그 경계는 모호하다고 할 수 있다[25]. 그러나 릴리스 플랜에 중심을 두고 있는 기법들은 우선순위화 및 상호의존 분석에 있어서 단순히 기능 중심의 요구사항만을 다루고 있으며, 초기 위험성이 큰 아키텍처에 관련된 요구사항을 고려하지 않고 있다. 즉, 요구사항간 의존관계에 있어서 하드웨어와 소프트웨어 요소가 가지는 특징에 기반한 아키텍처 요구사항간 상충관계가 반영되지 못했다.

임베디드 시스템은 시장 주도형 제품이라는 점과 소프트웨어뿐만 아니라 하드웨어 요소 및 이와 관련된 요구사항까지 고려해야 하는 특징을 갖는다. 결국, 요구사항간 우선순위 및 상호의존에 대한 많은 연구가 진행되어 왔으나 하드웨어 요소 및 이에 대한 요구사항을 고려하지 않아, 임베디드 시스템의 도메인에는 적합하지 않다. 따라서 임베디드 시스템을 위한 우선순위 프로세스는 하드웨어와 소프트웨어 요소의 특징 중 주요한 우선순위 요소(개발 비용, 시간, 품질, 제약사항 등)에 기반하여 아키텍처 스타일에 따른 아키텍처 요구사항간 상충관계를 분석하는 활동을 고려해야 한다.

2.3 요구사항 우선순위와 관점

요구사항은 요구사항 별로 중요도가 상이하다. 요구사항의 우선순위에 다양한 요소들이 영향요소로 작용하지만 가장 영향을 미치는 핵심적인 요소는 이해당사자와 관점이다. 일반적으로 이해당사자의 관점에 의해 요구사항의 중요성이 평가되어 우선순위가 결정된다. 요구사항 우선순위 관점은 다양할 수 있기 때문에 시스템의 특성을 고려해서 어떤 관점을 가지고 요구사항에 대한 우선순위 결정을 수행해야 하는지가 중요하다. 그러나 이러한 관점들은 동일한 관점에 대해서도 바라보는 측면에 따라서 이해당사자간 서로 다를 수 있다. 따라서 이해당사자간 공통된 기준을 정의하여 관점이 가지는 모호성이나 교차관심사(Crosscutting Concerns) [26, 27]로 인한 충돌가능성을 해결해야 한다.

[6]에서 패키지 소프트웨어 요구사항을 우선순위화 하기 위하여 비즈니스, 고객, 개발에 대한 세 가지 관점으로 체계화했다. 세 가지 관점의 분류를 통해 우선순위에 관련되는 다양한 관점을 제공하며, 목표를 기반으로 관점을 선정한다. 목표 기반 요구사항 분석[28]에서는 목표는 요구사항을 식별하고 구조화하고 타당성을 증명하는 논리적인 메커니즘이라고 정의하고 있다. 또한, 목표 지향 요구공학[29]에서는 목표의 역할을 언급하고 있다. 부적절한 요구사항을 피하는 것이 요구공학의 중요 관심사 중의 하나이며, 목표는 요구사항의 적절성에 대한 분명한 기준을 제공 한다. 따라서, 목표는 요구사항을 우선순위화하는 관점을 선택하는 중요한 기준이 된다. 그러나, [6]은 요구사항의 도메인을 패키지 소프트웨어로 한정하여 우선순위를 위한 관점 측면에서는 임베디드 시스템의 도메인과 관련한 관점에 미흡하다. 이러한 문제점은 임베디드 시스템의 도메인에 관련된 관점을 추가하여 기 제시된 우선순위 모델을 확장하는 것으로 해결 할 수 있다.

〈표 2〉 우선순위 관련 기존 연구 비교

Method / Factor	릴리스 플랜 중심 기법			우선순위 중심 기법					
	IFM	EVOLVE	ARP	$100 Test	Numerical Assignments	Planning Game	Ranking	Cost-value	Wiegers'Method
Stakeholders	All Major Stakeholders	All Major Stakeholders	All Major Stakeholders	All Major Stakeholders	All Major Stakeholders	Customer, Developer	OneStakeholder	Project Manager, Customers, Users	PM,Customer, Development
Aspect	Cost, Time-to-Market	Business Value, Urgency	Business Customer, Development	Importance	Importance	Importance, Cost,Risk	Importance	Cost,Value	Value,Cost,Risk
RequirementsDomain	Software, Functional	Software, Functional	Software, Functional	Software, Functional	Software, Functional	Software, Functional	Software, Functional	Software, Functional	Software, Functional
H/W, S/WArchitecture Trade-off	None	None	None	None	None	None	None	None	None

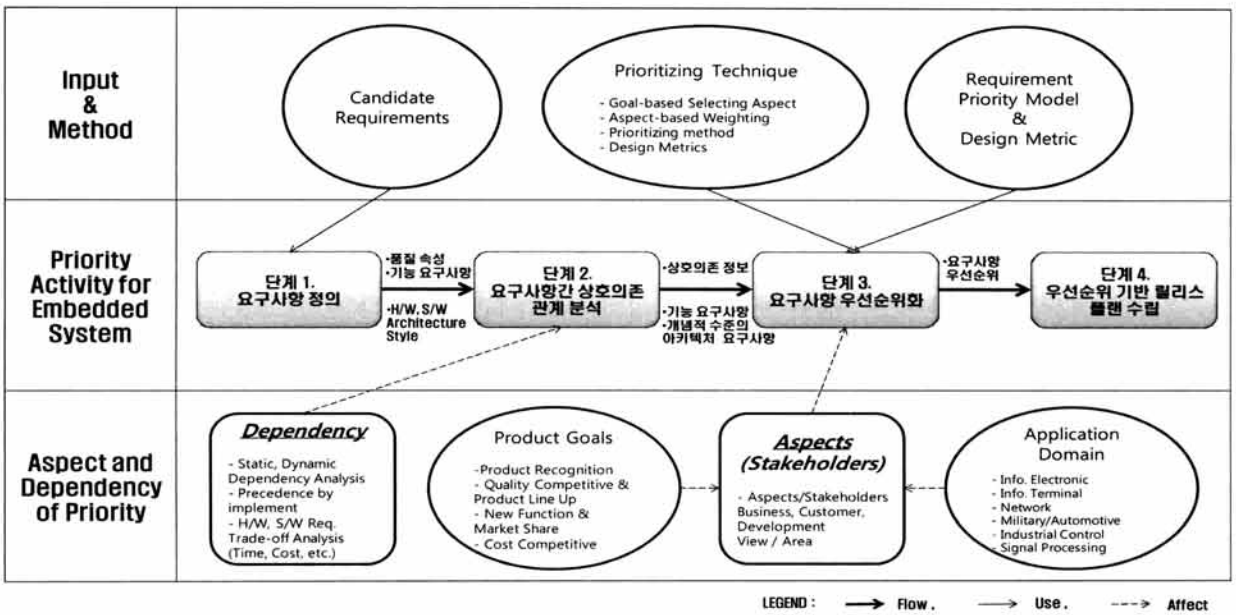
3. 요구사항 우선순위화 접근 모형

본 장에서는 요구사항간 의존관계 및 관점 기반의 임베디드 시스템 요구사항 우선순위를 위한 접근 모형을 기술한다. 임베디드 시스템의 도메인상에서 요구사항 우선순위 프로세스의 핵심적 활동은 기능 요구사항간 의존관계와 아키텍처 스타일에 따른 우선순위 요소 또는 아키텍처 요구사항간 상충관계를 분석하고, 요구사항 우선순위 모델을 활용하여 다양한 목표 및 상황을 고려한 관점을 선택하고, 분석된 의존정보를 반영하여 기능 요구사항 및 개념적 수준의 아키텍처 요구사항까지 고려된 우선순위를 결정하는 것이다. 단원 2.1의 관련연구에서 제시한 임베디드 시스템의 특징을 우선순위 프로세스에 반영하기 위한 요구사항을 다음과 같이 도출한다.

- 임베디드 시스템은 소프트웨어와 하드웨어 요소로 구성되며, 소프트웨어가 하드웨어에 종속된 결합체이다. 문제가 발생시 소프트웨어 개발 프로세스처럼 손쉽게 앞 단계로 되돌아 오지 못하기 때문에 우선순위 활동에서 초기에 아키텍처와 관련된 심각한 위험 요소를 제거 또는 완화시켜야 한다. 따라서 기능 요구사항뿐만 아니라 기능 요구사항에 대한 개념적 수준의 아키텍처 요구사항도 우선순위 결정이 요구된다.
- 임베디드 시스템의 구성요소인 하드웨어와 소프트웨어 요소는 개발 시간과 비용 등과 같은 민감한 우선순위 요소에 있어서 서로 차이를 가지기 때문에, 아키텍처 스타일에 따라 개발 시간이나 비용이 증가하거나 감소하는 상충관계가 발생하므로 우선순위 요소 또는 아키텍처 요구사항간 상충관계에 대한 분석이 요구된다
- 임베디드 시스템에서 기능, 품질, 비용, 적시성, 제약사항과 같은 중요한 요소들은 응용 분야 및 제품 종류, 규모에 따라 중요도가 상이하므로 다양한 응용 분야에 따른 개발 목적 및 상황을 고려할 수 있는 우선순위 모델과 프로세스의 유연성이 요구된다.
- 우선순위 정립은 임베디드 시스템 개발 프로세스 중 하드웨어와 소프트웨어 요소간 역할분담 단계에서 이루어져야 하기 때문에 개발 프로세스에 적용 가능한 체계화된 우선순위 선정 프로세스의 설계가 요구된다.

다음으로 요구사항의 우선순위화를 위한 본 연구의 설계 원리 및 특징은 다음과 같다.

- 다양한 영향 요소를 고려한 관점기반 우선순위 모델을 설계한다. 즉, 요구사항의 중요성 및 요구사항의 우선순위 표기방식에 영향을 미치는 영향 요소들인 요구사항의 규모, 개발 제품의 목표, 이해당사자 그리고 제품의 응용분야 및 종류 등을 고려하여 우선순위 모델을 구축한다.
- 임베디드 시스템에서 요구사항에 대한 품질속성과 관련한 주요 제약사항 또는 최적화 사항을 매트릭(Metric)으로 식별하여 임베디드 시스템 디자인 매트릭으로 정의한다. 상기 디자인 매트릭은 요구사항간 상충관계 파악 및 시뮬레이션, 관점에 대한 정량화된 기준을 정의할 때 요구사항을 측정하고 분석할 수 있는 가이드 라인으로 활용 할 수 있다. 이를 통해 요구사항간 상충의 정도를 보다 가시적으로 쉽게 파악이 가능하며, 우선순위 결정을 위한 객관적이고 정량화된 기준 정의가 가능하다.
- 하드웨어와 소프트웨어의 특징에 기반한 우선순위 요소 또는 아키텍처 요구사항간 의존성을 상충관계 분석을 통해 이해당사자가 우선순위 결정에 반영할 수 있도록 한다. 이를 위해 임베디드 시스템을 구성하는 하드웨어와 소프트웨어 요소가 가지는 특징 중 요구사항의 우선순위에 영향을 줄 수 있는 요소들을 한정적으로 선정하여 분석한다. 즉, 개발 비용, 처리속도, 전력소모, 개발시간, 유연성, 재사용성 등과 같은 중요한 우선순위 관점에 관련되는 특징 요소들에 기반하여 상충 관계를 정의하고, 분석하여 요구사항의 우선순위 결정에 반영한다.
- 임베디드 시스템 개발 프로세스에 적합하고, 체계화된



(그림 1) 임베디드 시스템의 요구사항 우선순위화 접근 모형

우선순위 프로세스를 정립한다. 임베디드 시스템 개발 프로세스의 특징인 하드웨어, 소프트웨어 역할 분담 및 동시 설계 활동을 고려한 우선순위 프로세스를 제안한다. 즉, 임베디드 시스템의 개발 프로세스에 적용 가능하고, 요구사항 우선순위 모델에 기반한 우선순위 프로세스를 구축한다.

상기 임베디드 시스템의 특징을 우선순위 프로세스에 반영하기 위한 요구사항과 설계 원리에 입각하여, (그림 1)은 임베디드 시스템의 의존성을 고려한 요구사항간 의존관계와 관점에 기반하여 요구사항의 우선순위를 정립하는 연구 접근 모형을 나타낸다.

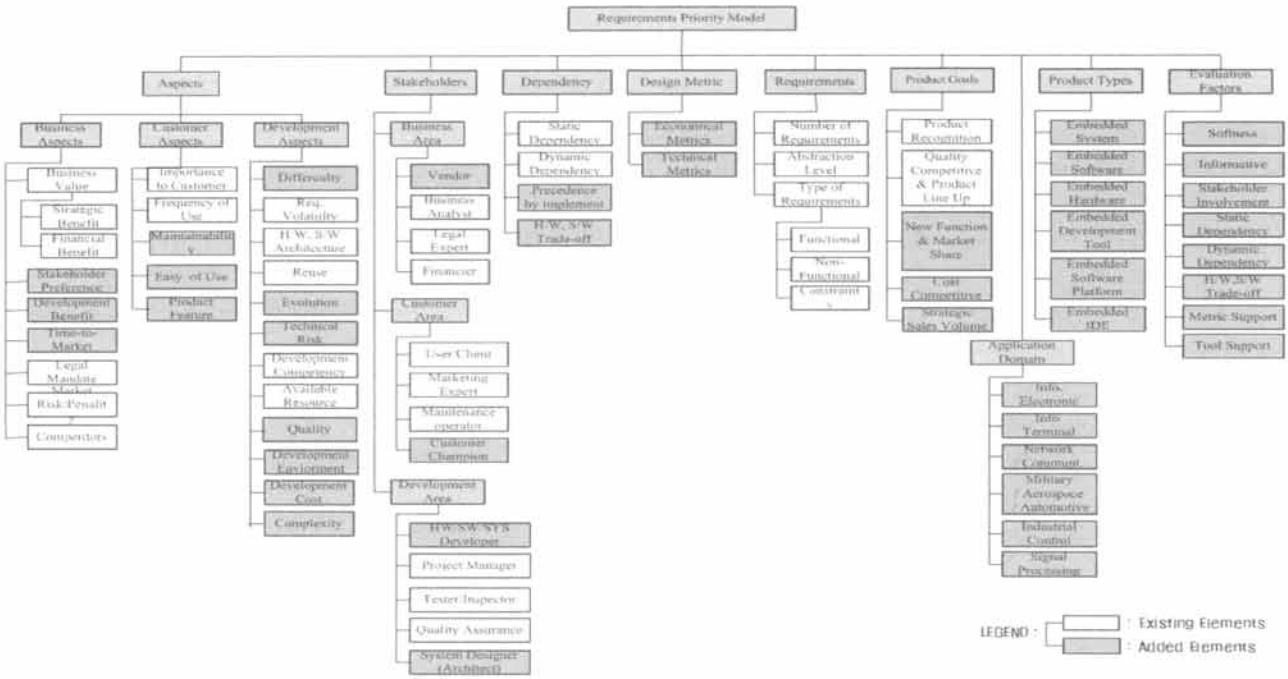
임베디드 시스템의 요구사항 우선순위를 정립하기 위한 연구 접근 과정은 다음과 같다. 단계1은 제품화 대상 임베디드 시스템의 요구사항을 수집 및 추출해서 개발할 제품의 요구사항을 정의하는 단계이다. 단계2는 정의된 요구사항에 대해 요구사항간 의존관계를 분석하여 의존정보를 파악하는 단계이다. 단원2.2에서 분류한 3가지 의존관계에 상충관계를 추가하여 4가지 유형으로 의존관계를 분석한다. 즉, 하드웨어와 소프트웨어의 특징에 기반한 우선순위 요소 또는 아키텍처 요구사항간 의존성을 상충관계로 추가 정의한다. 요구사항간 상충관계 및 중요도 분석은 측정이 필요한 우선순위에 관계되는 주요한 요소들을 선정하고, 단원4.3의 디자인 매트릭을 이용한다. 단계3은 요구사항간에 우선순위를 결정하는 단계이다. 이것은 관점을 선택하고 디자인 매트릭을 이용해서 선택된 관점에 대한 관점별 기준을 정의하고, 우선순위의 표기방식을 선정하여 우선순위화를 수행한다. 단계4는 단계 3의 요구사항 우선순위에 의해 개발 제품의 릴리스 플랜을 수립한다. 본 논문은 임베디드 시스템의 요구사항을 우선순위화하기 위한 2, 3단계에 초점을 둔다.

4. 임베디드 시스템의 요구사항 우선순위 모델

본 논문은 임베디드 시스템 제품 개발에서 요구사항 우선순위에 영향을 주는 이해당사자, 개발 제품의 목표, 제품 응용 분야 및 종류에 따른 맞춤형 우선순위화를 고려할 수 있도록 유연한 요구사항 우선순위 프로세스를 제시한다. 이를 위해 우선순위에 영향을 주는 다양한 요소(관점, 영향 요소, 디자인 매트릭 등)들로 구성된 요구사항 우선순위 모델이 필요하다. 우선순위 모델은 [6]이 제시한 모델에서 임베디드 시스템의 특성을 반영하여 확장, 구축한 것이다. 우선순위 모델은 디자인 매트릭(Design Metric), 응용분야(Application Domain), 제품종류(Product Type) 요소를 새로이 추가하고, 관점(Aspect), 의존관계(Dependency), 이해당사자(Stakeholders), 제품목표(Product Goals)를 재정립, 확장하여 구축한다. 본 모델에서 정의한 디자인 매트릭은 5장의 우선순위화 프로세스의 요구사항간 상호의존 관계 분석 및 정량화된 관점별 기준을 정의하는데 적용된다.

4.1 요구사항 우선순위 모델

임베디드 시스템 요구사항 우선순위 모델은 관점(Aspect), 의존관계(Dependency), 이해당사자(Stakeholders), 디자인 매트릭(Design Metric), 요구사항(Requirements), 제품 목표(Product Goals), 제품 응용 분야(Application Domain), 제품 종류(Product Types) 그리고 평가요소(Evaluation Factors)를 포함한 총 9가지 항목으로 (그림 2)와 같이 구성한다. [6]에서 제시한 일반적인 요구사항 우선순위 모델과 달리 임베디드 시스템에서는 요구사항의 관점과 의존관계, 디자인 매트릭 등이 중요하다. 각 요소가 우선순위에 영향을 주는 특징은 다음과 같다.



(그림 2) 임베디드 시스템의 요구사항 우선순위 모델

4.1.1 관점(Aspect)

우선순위와 관련된 관점은 (그림 2)의 우선순위 모델상에서 비즈니스, 고객, 개발의 세 가지 분류로 나눌 수 있으며, 요구사항, 제품 목표, 제품 응용 분야 및 종류 그리고 조직의 상황에 따라 적절한 관점을 선택하는 것이 필요하다. 요구사항을 우선순위화하기 위해서는 다양한 관점들 중에서 어떤 기준에 의해 관점을 선택할 것인가에 대해 결정을 내려야 한다.

비즈니스 관점(BA: Business Aspects)은 사업의 관점에서 요구사항 우선순위화에 가질 수 있는 관점으로 비즈니스 가치, 긴급도, 개발비용과 시간, 경쟁사 현황 등이 있으며, 고객 관점(CA: Customer Aspects)은 사용자 관점으로 고객의 중요도, 사용 빈도 등을 고려해야 한다. 마지막으로 개발 관점(DA: Development Aspects)은 개발자 관점에서 중요시되는 인자들로 난이도, 요구사항 안정도, 아키텍처 영향도, 재 사용성 등이 있으며, 임베디드 시스템에서는 비즈니스 관점과 더불어서 매우 중요한 관점이 된다. 임베디드 시스템이 가지고 있는 전형적인 문제로는 개발 비용, 제품 단가, 품질에 민감한 점 그리고 하드웨어 요소를 포함한 개발 환경이 가지는 의존성 등이 있으며, 이를 반영하기 위해 추가한 요소로 비즈니스 관점의 개발 이익, 개발 관점의 품질, 개발환경, 개발 비용 등이 있다.

4.1.2 이해당사자(Stakeholders)

우선순위화는 최종적으로 이해당사자들의 만족을 위해 수행하는 활동이며, 개발 제품과 관계가 있는 모든 사람 또는 조직을 이해당사자로 볼 수 있다. 이해당사자는 새로운 시스템 또는 애플리케이션의 구현에 의해 실질적으로 영향을 받는 사람 또는 조직을 말한다[30]. 예를 들어, 임베디드 소프트웨어는 하드웨어에 종속적이기 때문에 하드웨어 벤더에 따라 개발 여건이 크게 영향 받을 수 있으며, 하드웨어 개발자와 소프트웨어 개발자간 중요시하는 요구사항이 서로 다를 수 있다. 이와 같이 임베디드 시스템 개발은 이해당사자 및 개발활동에서 일반적인 패키지 소프트웨어 개발에 비해 차이를 가진다. 따라서 우선순위화를 위한 관점 및 기준 그리고 관점별 가중치는 이해당사자에 따라 달라지기 때문에 이해당사자는 우선순위에 영향을 미치는 가장 중요한 인자 중 하나이다. 이를 반영하기 위해 (그림 2)의 우선순위 모델에서는 관점의 BA, CA, DA에 해당되는 각 영역(Area)의 전문가를 이해당사자로 대응시켜 우선순위화 활동에 참여하도록 설계했다.

4.1.3 의존관계(Dependency)

기존의 많은 연구 결과에 따르면, 완전한 독립이 가능한 요구사항은 불과 5%에 불과하며, 요구사항 각각은 독립적이지 않고, 요구사항 사이에 다양한 종류의 의존관계가 있다고 알려져 있다. 일반적으로 소프트웨어에서의 의존관계는 정적, 동적 또는 구현 선후 관계로 제시한다. 그러나, 개발비용, 제품 단가, 개발 시간, 재사용성, 성능 등에 영향을 주는 하드웨어와 소프트웨어 요소에 의한 우선순위 요소 또는 아키텍처 요구사항간 상충관계는 고려하고 있지 않다[6,9]. 따라서 임베디드 시스템에서의 의존관계를 요구사항간 구현 선후 관계, 동적, 정적 의존관계 그리고 상충관계로 정의한다. 또한, 상충관계를 적용시키는 요구사항의 범위는 기능 요구사항에 대한 아키텍처 스타일을 의미하는 개념적 수준의 아키텍처 요구사항으로 한정한다.

4.1.4 디자인 매트릭(Design Metric)

디자인 매트릭은 요구사항간 의존관계 분석 및 관점별 기준을 정의하는데 사용되는 요소이다. (그림 2)의 우선순위 모델상에서 보듯이 경제적 매트릭(Economical Metric), 기술적 매트릭(Technical Metric)의 두 가지 분류로 나눌 수 있으며, 제약사항 및 최적화사항과 같은 품질 속성과 연관 관계를 가진다. 즉, 디자인 매트릭은 중요시 되는 요구사항이나 관점을 가시적으로 측정할 수 있는 좋은 방법이 되므로 우선순위를 위한 관련된 요소가 된다. 디자인 매트릭은 단원4.3에서 자세하게 기술한다.

4.1.5 요구사항(Requirements)

요구사항의 규모, 추상화 수준 및 종류는 요구사항의 우선순위를 나타내는 방법에 영향을 주는 인자로 볼 수 있다[11]. 요구사항의 추상화 수준을 결정하는 것은 어려운 일이며 요구사항의 숫자와 그 복잡도에 달려있다. 이러한 요구사항의 본질적인 특성을 고려하여 본 논문에서는 요구사항의 용어 정의를 이해당사자의 관점에서 강하게 연결되어 하나의 특성을 나타내는 요구사항의 집합[9]으로 정의하며, 이하 논문에서 사용되는 '요구사항'과 '요구사항의 집합'은 동일한 것으로 간주한다. 또한, 본 모델에서는 요구사항의 분류를 최상위 목표 수준의 이해당사자 요구사항, 시스템 요구사항 수준의 기능 요구사항, 기능 요구사항을 이루는 서브 시스템 요구사항 수준의 아키텍처 요구사항으로 정의한다. 따라서 (그림 2)의 요구사항 요소 중 '기능 요구사항'의 범주는 시스템 요구사항 수준이고, '비기능 요구사항이나 제약사항'의 범주는 서브 시스템 요구사항 수준인 아키텍처 요구사항으로 볼 수 있다.

4.1.6 제품 목표(Product Goals)

오픈 시장에서의 임베디드 시스템은 가격 경쟁, 유사 제품간 경쟁이 매우 심하기 때문에 일반적으로 시장 점유, 전략적인 판매량 달성과 같은 목표도 관점 선정 시에 중요한 기준이 될 수 있다. 따라서 임베디드 시스템 제품 개발의 목표는 다양할 수 있지만 요구사항 우선순위 모델에서와 같이 일반적으로 상품인지도(Product Recognition), 품질 경쟁력과 제품 구색(Quality Competitive & Product Line Up), 신규기능 개발과 시장 점유율(New Function & Market Share) 및 가격 경쟁력(Cost Competitive) 그리고 전략적인 판매량 달성(Strategic Sales Volume)과 같이 5가지로 분류해 볼 수 있다. 이것은 우선순위를 위한 관점, 즉 무엇에 기준을 두고 순위화할 것인가에 영향을 준다[26-29].

4.1.7 응용 분야(Application Domain)

요구사항 우선순위 모델에서는 임베디드 시스템의 응용분야를 정보 가전(Info. Electronic), 정보 단말(Info. Terminal), 네트워크/통신(Network/Communication), 군사/우주항공/자동차(Military/Aerospace/Automotive), 산업 기계(Industrial Control), 신호 처리(Signal Processing)로 구분한다. 정보 가전은 제품 단가와 전력 소모에 민감하고, 정보 단말은 낮은 전력 소모뿐만 아니라 통신 품질(QoS)을 고려해야 한다. 또한 네트워크/통신과 같은 통신 분야는 보안성 및 데이터의 무결성 등이 중요시되고 있으며, 군사/우주항공/자동차와 같은 분야는 안전성이 최우선적으로 고려되어야 한다. 마지막으로 산업 기계 분야와 신호 처리 분야는 각각 중복제어구조와 이상신호 대응 그리고 빠른 컴퓨팅 스피드 능력이 중요시하게 요구된다[2-5]. 위와 같이 중요시되는 주요한 속성들에 대해서 요구사항단계부터 명확히 정의하고, 이해당사자간 이해가 선행되어야 한다.

4.1.8 제품 종류(Product Type)

임베디드 시스템과 관련된 제품 종류는 임베디드 시스템(Embedded System)을 비롯하여, 임베디드 소프트웨어(Embedded Software), 하드웨어(Embedded Hardware), 그리고 개발 지원 툴(Embedded Development Tool), 플랫폼(Embedded Software Platform), 통합 개발 환경(Embedded Integrated Development Environment) 등으로 구분할 수 있다.

임베디드 시스템은 다양한 규모와 분야에서 급속히 늘어나고 있으며, 이를 수용하기 위해서는 각각의 응용분야에 특화되는 방향으로 개발이 되어야 한다. 예를 들어 임베디드 소프트웨어 개발 도구는 단품부터 OS와 H/W를 같이 제공하는 IDE와 같이 통합된 제품이 있다. 이러한 제품을 이용하거나 개발할 경우 개발환경에서 차이가 나게 되며, 각 요소간 의존성, 개발시간과 비용에 영향을 미치게 된다. 그리고 개발 도구는 분야별로 S/W개발도구, 설계자동화, 테스트자동화 도구로 크게 나눌 수가 있는데, 자동차/군사항공 분야에서는 설계 및 테스트 관련 도구 요구사항에 대한 중요성이 더 높으며, 가전분야에서는 S/W개발 관련 도구의 요구사항이 더 중요하다.

4.1.9 평가 요소(Evaluation Factors)

일반적으로 평가 요소는 요구사항 우선순위 결과에 직접적인 영향을 주는 요소와 직접적인 영향을 주지 않는 요소로 나눌 수 있다[6]. 그러나 본 모델의 평가요소는 우선순위 기법에 대한 전반적인 비교 평가를 위해 전자와 후자의 구분 없이 구성하며, 단원 6.3에서 기존 연구와 제시 방법과의 비교분석을 위한 평가요소로 사용한다. 즉, 의존과 관점 기반 임베디드 시스템의 요구사항 우선순위 프로세스라면 임베디드 도메인에 적합한 이해당사자, 우선순위 요소 그리고 다양한 개발 상황을 커버하기 위한 유연성과 의존성이 매우 큰 도메인의 특성상 시스템 요구사항간 의존관계 및 아키텍처 요구사항간 상충관계 분석 지원, 마지막으로 우선순위 프로세스의 시간소모, 정량적인 측면의 효율성 평가를 위해 매트릭의 사용과 자동화 도구의 지원을 비교 평가한다.

4.2 관점과 상호의존 관계

관점 기반 요구사항의 우선순위 결정은 해당 이해당사자가 요구사항간 중요도를 차별화하여 나타내는 것이다. 그러

나 이해당사자들은 각자 추구하는 가치나 관점이 서로 동일하지 않다. 이를테면 이해당사자가 개발자라면 요구사항을 개발하는 개발자 관점에서의 우선순위를 가장 우선시하게 될 것이며, 이것이 개발 우선순위가 된다. 또한, 이해당사자가 사용자라면 요구사항을 제시하는 사용자 관점의 우선순위가 될 것이다. 이것이 업무의 중요도가 된다. 결국, 좀 더 객관적인 우선순위화를 위해 다양한 이해당사자가 요구사항을 동일하게 바라볼 수 있는 기준이 요구된다. 따라서 중요도를 차별화하는 기준으로 관점이 사용되며, 관점은 우선순위화에 중요한 요소가 된다. 본 논문에서는 관점을 비즈니스(Business), 고객(Customer), 개발(Development)의 세가지 측면과 대응되도록 비즈니스(Business Aspects), 고객(Customer Aspects), 개발(Development Aspects)의 관점으로 분류하여 관점을 선택할 수 있도록 한다.

상호의존 관계란 어떤 요구사항에 의해 수행된 동작은 의도하거나 기대하지 않았음에도 다른 요구사항에 영향을 줄 수 있는 관계를 의미한다. 이러한 상호의존 관계는 제약사항이나 전제조건과 같은 의미를 포함하고 있다. 그래서 요구사항은 복잡하게 관계되어 있고 서로 영향을 주기 때문에 독립적으로 다루어질 수 없다. 또한 상충 관계에 놓여있는 요구사항은 개발 제품의 목표 및 상황을 고려하여 적절한 협의가 필요하며, 우선순위화를 통한 결정이 한가지 해결책이 될 수 있다.

요구사항간 의존관계는 우선순위화뿐만 아니라 릴리스 플랜을 위해 고려해야 하는 의존관계로서 시스템 구조상의 결합(Coupling)에 의한 정적 의존관계(Static Coupling), 시스템 운영시의 동적 의존관계(Dynamic Coupling), 시스템 구현시의 구현 선후 관계(Precedence by implement) 및 아키텍처 스타일에 의한 상충관계(H/W, S/W Architecture Trade-off)의 4가지로 분류한다.

4.2.1 정적 상호의존 관계

정적 연결은 구조적으로 부모-자식과 같은 관계로 동일한 릴리스에 함께 구현되어야 하는 관계를 나타낸다. 그 결합 정도에 따라 Mandatory와 Optional로 나눌 수 있으며, Mandatory는 동일한 릴리스에 포함되도록 하지만, Optional의 경우 우선순위와 출시일정에 따라 동일한 릴리스에 포함되지 않아도 된다. 또한 이런 정적인 정보가 아닌 동적인 관계 정보는 요구사항 변경으로 인한 파급효과를 최대한 정확하게 인지하게 위해서도 필요하다.

4.2.2 동적 상호의존 관계

동적 연결은 구현 후 동작 운영할 때 연동되는 관계로 순차적으로 또는 동시에 동작 운영되거나, 하나의 요구사항이 다른 요구사항에 의해 영향을 받아 상태, 행위, 데이터, 코드 등이 변경되는 관계로 그 결합 정도에 따라 Mandatory와 Optional로 나눌 수 있다. 즉, 두 개의 요구사항이 반드시 동일한 릴리스에 포함되어야 하면 Mandatory, 우선순위와 출시일정에 따라 동일한 릴리스로 묶여지지 않아도 된다면 Optional로 한다.

4.2.3 구현 선후 관계

구현 선후 관계는 구현 과정에서의 선후 관계로 즉, 두 개의 요구사항 R(i)와 R(j) 사이의 구현 순서에 대한 관계로써 R(i)가 구현되기 이전에 R(j)가 먼저 구현되어야만 하는 관계를 나타내며, 반드시 동일한 릴리스에 포함되지 않아도 된다. 하지만, 구현 선후 관계는 설계 및 구현에 관계되어 개발 시간 및 일정에 직접적인 영향을 줄 수 있다. 이에, 통합과 테스팅 작업을 포함하여 제품의 출시에까지 영향을 미치므로 중요하다.

4.2.4 상충관계

상충관계의 의미는 일반적으로 요구사항간 갈등(Conflicts) 또는 상충(Trade-off)되는 관계로 요구사항이 동시에 존재할 수 없거나, 하나의 요구사항을 만족시키면 다른 요구사항의 만족도는 떨어지는 관계를 나타낸다. 본 논문에서는 하드웨어와 소프트웨어 요소가 기능 요구사항과 매핑되면서 도출되는 아키텍처 스타일에 따른 우선순위 요소 또는 아키텍처 요구사항간의 상충관계로 재정의 한다. 상충되는 요소들은 소프트웨어와 하드웨어 요소가 가지는 특징들 중 비용(Cost), 개발 시간(Development Time), 유연성(Flexibility),, 재사용성(Reusability), 수행성능(Performance), 전력소모량(Power)으로 한정하며, 선정된 요소들은 임베디드 시스템에서 요구사항의 우선순위에 영향을 줄 수 있는 요소들이다.

하드웨어, 소프트웨어 요소가 지닌 특징에 기반한 상충관계는 일반적인 소프트웨어 시스템에서는 하드웨어 요소를 고려하지 않기 때문에 해당사항이 없지만 하드웨어와 소프트웨어 요소 모두를 포함하고 있는 임베디드 시스템의 개발에서는 반드시 고려해야 하는 핵심 이슈이다. 즉, 임베디드 시스템 개발 프로세스에서는 설계와 구현단계 이전에 하드웨어, 소프트웨어 요소와 기능 요구사항이 매핑되는 역할분담이 최우선적으로 이루어져야 하며, 요구사항의 우선순위화도 설계와 구현단계 이전에 이루어져야 하기 때문에 고려해야 하는 이슈가 된다.

하드웨어와 소프트웨어 요소의 특징이 가지는 상충관계를 (그림 3)에서 우선순위에 관련된 요소들[2, 5, 21, 22]과 대비해서 보여준다. (그림 3)에서 제시된 요소간 상충관계 외에 다양한 요소에 따른 상충관계가 존재하지만, 본 논문에서는 요구사항 우선순위에 관련된 (그림 3)의 6가지 특징에 한정

Matrix / Component	Cost	Time	Performance		Flexibility	Reuse
			Speed	Power		
Software	Low	High	Low	High	High	Low
↕	↓	↑	↓	↑	↑	↓
Hardware	High	Low	High	Low	Low	High

(그림 3) 상충관계 분석의 기반이 되는 하드웨어와 소프트웨어 요소의 특징

하여 이를 기반으로 상충관계 분석을 한다.

임베디드 시스템은 일반적으로 응용에 특화된 하드웨어나 유연하게 프로그래밍이 가능한 소프트웨어 프로세서 그리고 mechanical transducers and actuators로 이루어진다. 또한 응용에 특화된(application-specific) 하드웨어로는 boards, ASICs, FPGAs 등이 있으며 이들이 갖는 장점은 performance, low power등 이며, 유연하게 프로그래밍이 가능한 소프트웨어 프로세서(SW on program processors)로는 DSPs, micro controllers 등이 있으며 이들이 갖는 장점은 flexibility, complexity 등이 있다[2, 3, 5, 22]. (그림 3)에서 화살표가 나타내는 방향성의 의미는 화살표 방향에 따라 기능 요구사항과 소프트웨어, 하드웨어 요소간의 상대적인 할당 정도를 나타낸다. 가령, 유연성의 경우 소프트웨어 요소가 하드웨어 요소보다 유연성이 높기 때문에 요구되는 제품의 특성이 높은 유연성이라면 하드웨어 요소보다 소프트웨어 요소에 중점을 두어야 한다. 반면, 소프트웨어 요소는 개발 시간이나 성능적인 측면에서 만족도가 떨어지게 된다.

Cost와 Time의 경우를 살펴보면 Cost는 제품 단가를 의미하는 Unit Cost 개념이고, Time은 개발 시간을 의미한다. 예컨대, 우선순위를 위한 관점 중 가격 경쟁력이 가장 중요한 경우는 개발 비용 및 유연성 측면에서 소프트웨어 요소가 하드웨어 요소보다 더 뛰어나기 때문에 소프트웨어 요소를 포함한 아키텍처 요구사항의 중요도가 상대적으로 높게 평가될 수 있다. 즉, 하드웨어 요소가 포함되지 않은 소프트웨어 요소로만 이루어진 아키텍처 요구사항이나 하드웨어와 소프트웨어 요소를 모두 포함하는 아키텍처 요구사항 중에서 소프트웨어 요소를 더 많이 포함하는 요구사항의 우선순위가 높아야 목표와 최적화 사항 등에 부합하는 우선순위 결과를 기대할 수 있다.

4.3 임베디드 시스템의 디자인 매트릭

디자인 매트릭(Design Metric)은 요구사항으로부터 식별되며, 제품의 특성 및 개발과 관련 있는 요소로서 시스템 개발에서 고려해야 할 제약사항이나 최적화 요구사항에 대한 측정 가능한 기준을 나타낸다[2]. 시스템의 명세가 올바른지, 구현과 시스템 명세가 부합하는지, 실시간성 특징이나 실제 데이터상에서 어떻게 테스트 해야 하는지 등과 같은 시스템의 올바른 작동에 대한 문제점을 푸는데 활용할 수 있다.

본 논문에서는 <표 3>과 같이 임베디드 시스템과 관련된 디자인 매트릭을 정의한다. 임베디드 시스템의 디자인 매트

〈표 3〉 임베디드 시스템 디자인 매트릭의 예

Metric Name		Description	
Technical Metrics	Size	Definition	시스템에 필요한 물리적인 공간
		Measurement	Quantitative Criterion, Gate
	Performance	Definition	실행 시간 또는 시스템의 처리량
		Measurement	Quantitative Criterion, MIPS, operations/sec, throughput
	Power(Energy Efficiency)	Definition	시스템에 의해 소모되는 전원의 양
		Measurement	Quantitative Criterion, Watts, Joule, Energy
	Flexibility	Definition	재구성 및 재사용의 용이성에 대한 정도
		Measurement	Qualitative Criterion
	Reliability	Definition	시스템이 지속적으로 올바른 서비스를 제공할 수 있는 신뢰의 정도
		Measurement	Quantitative Criterion, MTTF
	Availability	Definition	사용자가 서비스나 시스템 사용을 원할 때 시스템이 제공해줄 수 있는 가용의 정도
		Measurement	Quantitative Criterion, A(t) = MTTF / (MTTF + MTTR)
	Safety	Definition	사용자 또는 환경을 치명적인 위험으로부터 지키는 안전의 정도
		Measurement	Quantitative Criterion
	Security	Definition	비 인가자의 무단 접근을 예방하는 정도(무결성 + 기밀성)
		Measurement	Qualitative Criterion
	Testability	Definition	요구사항에 대한 테스트가 용이한 정도
		Measurement	Quantitative Criterion, TA = Number of testing requirement/Number of overall requirement
		Measurement	Quantitative Criterion, Risk is the expected loss per unit time
Economical Metrics	Unit cost	Definition	NRE 비용을 제외한 시스템의 각 사본을 제조하는 금전적 비용
		Measurement	Quantitative Criterion, per-product cost = total cost / # of units = (NRE cost / # of units) + unit cost
	NRE cost (Non-Recurring Engineering cost)	Definition	시스템을 설계하는데 단 한번만 소비되는 금전적 비용
		Measurement	Quantitative Criterion, Cost
	Time-to-prototype	Definition	동작하는 버전의 시스템을 구축하는데 필요한 시간
		Measurement	Quantitative Criterion, Time, Day, Month, Year
	Time-to-market	Definition	시스템이 릴리스되고 소비자에게 팔리는시점을 위해 요구되는 개발 시간
		Measurement	Quantitative Criterion, Time, Day, Month, Year
	Flexibility	Definition	큰 NRE 비용을 들이지 않고 시스템의 기능을 변경할 수 있는 유연의 정도
		Measurement	Qualitative Criterion
	Maintainability	Definition	첫 릴리스 이후에 시스템을 수정할 수 있는 유지의 정도
		Measurement	Qualitative Criterion

릭은 기술적 매트릭과 경제적 매트릭으로 구성되며, 기술적 매트릭은 임베디드 디바이스들의 기술적 설계 그리고 명세나 기술적 요구사항이 충족되는지 결정을 내리기 위해 주로 사용한다. 또한 경제적 매트릭은 시장에서 상용제품을 구매하거나 판매하기 위한 의사 결정에 주로 사용한다.

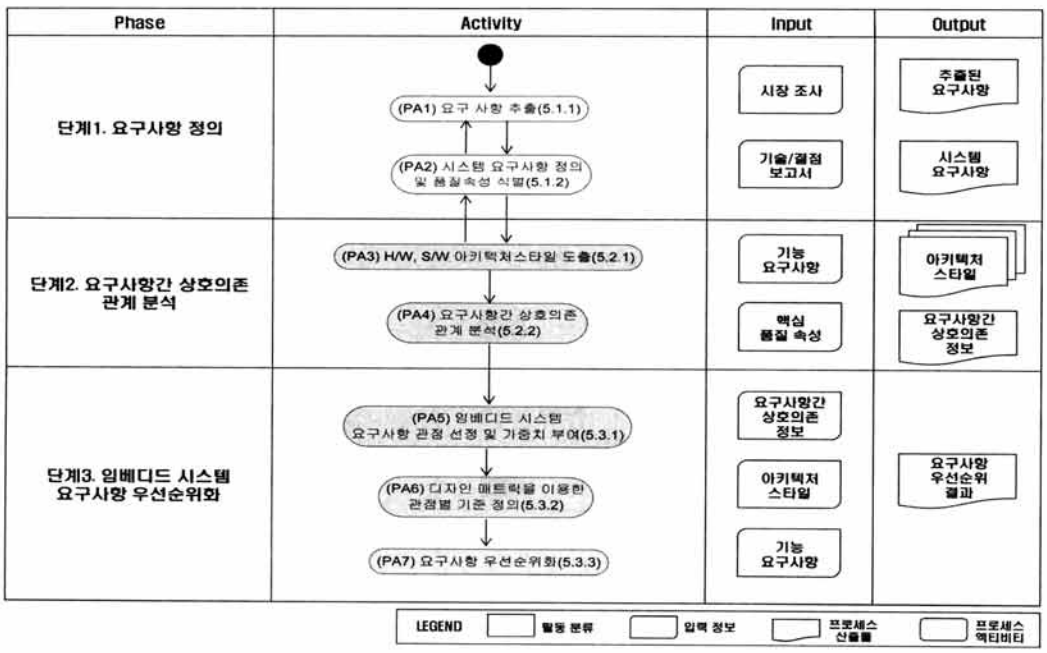
이러한 매트릭은 요구사항에 대한 품질속성이나 관점 등과 대응되는 관계를 가지며, 관점과 관련된 여러 요소 및 요구사항간 관계 분석을 보다 정량적으로 측정 가능케 하는 도구가 된다[34]. 이를 테면 기술적, 경제적 매트릭은 개발 관점과 비즈니스 관점에 관련한 요소와 대응된다. 또한, 고객 측면을 포함한 그 외의 매트릭의 경우 기술적, 경제적 매트릭을 이용하여 어느 정도 보완할 수 있으며, 주관적인 요소의 개입이 많은 사용자의 행복지수를 객관적으로 측정하거나 정량화시키기는 매우 어려우므로 이 부분은 다루지 않는다. 그 외에 상황에 따라 필요한 매트릭은 GQM(Goal Question Metric)기법[33]을 이용하여 적절하게 개발 조직에서 필요한 매트릭을 식별하고, 개발하여 해결할 수 있다.

예컨대, 임베디드 시스템 개발에서 고려가 필요한 주요한 요소들에 대해 보다 더 객관적인 이해를 돕기 위해 임베디드 시스템 디자인 매트릭을 이용한다. 가령, 가용성과 신뢰성이나 안전성이 상대적으로 중요시 된다면, 신뢰성(Reliability), 가용성(Availability), 중복성(Redundancy)과 같은 품질 속성이 중요할 것이다. 따라서 이것들을 측정할 수 있는 MTTF(Mean time to failure), MTTR(Mean time to repair) 그리고 N-Modular-Redundancy 측정을 위한 매트릭을 선정하여 6장의 <표 5>와 같이 관점별 기준을 선정 시에 수치화된 정보를 통해 이해당사자들의 객관적인 공감을 이끌어 낼 수 있다. 또한 5장의 상호의존 관계 분석 활동(PA4: Process Activity 4)에서 예로든 가용성과 신뢰성 사이의 상충관계 분석을 명확하게 할 수 있다.

임베디드 시스템의 디자인 매트릭을 가이드 라인으로 활용하여 요구사항이나 관점을 측정하고 분석하여 기대 되는 이점은 다음과 같다. 비용과 관련된 비즈니스 요소나 품질과 관련된 기술적인 요소를 측정할 수 있도록 가이드 라인이 되며, 프로젝트의 스케줄링 및 계획의 기초가 되고, 보다 더 정확한 계획 수립을 기대할 수 있다. 또한 매트릭이 가지는 가시적인 특성이 핵심 매트릭(Core metric)간의 상호의존 관계 파악에 용이하게 작용하며, 매트릭을 이용한 선정된 관점의 정량화를 통해 이해당사자간 보다 객관적인 의사소통 및 이해가 가능하다.

5. 임베디드 시스템의 요구사항 우선순위 프로세스

임베디드 시스템의 요구사항에 대한 우선순위를 정하기 위해서는 요구사항 우선순위 모델을 기반으로 구성된 체계적인 프로세스가 필요하다. 본 논문에서 제시하는 요구사항 우선순위 프로세스는 (그림 4)에 나타나 있으며, 본 장에서는 요구사항 우선순위 모델에 기반한 우선순위 프로세스를 수행 절차에 따라서 기술한다. 요구사항 우선순위 프로세스는 요구사항 추출부터 우선순위화까지의 과정으로 각 단계는 세부 활동들로 구성되며, 활동 수행에 필요한 입력 정보를 사용해서 해당 결과물을 생성한다. 요구사항에 대한 우선순위 프로세스는 먼저 임베디드 시스템의 요구사항을 정의하고, 요구사항간 상호의존 관계를 분석한다. 이어서 관점과 의존관계를 반영하여 요구사항 우선순위화를 수행하는 3단계로 구성된다. 즉, (그림 4)에서 단계2와 단계3 활동이 임베디드 시스템을 위한 핵심 요구사항 우선순위화 과정이다.



(그림 4) 임베디드 시스템의 요구사항 우선순위 프로세스

5.1 요구사항 정의

요구사항 정의 활동에서는 요구사항이 다양한 경로를 통해 요구사항이 수집되고, 개발조직에 의해 전략적으로 개발이 된다. 이 활동은 요구사항 추출(PA1)과 시스템 요구사항 정의 및 품질속성 식별(PA2)의 하부 활동들로 이루어진다.

5.1.1 요구사항 추출(PA1)

본 활동에서는 시스템 요구사항 정의를 위해 사용자를 이해하고 그들이 무엇을 필요로 하는가를 다양한 방법으로 찾아내어, 요구사항 후보들을 추출한다. 이와 같은 일련의 활동을 요구사항 추출 활동이라 하며, 시장조사와 기존 제품의 기술 및 결점보고서 등 다양한 경로를 통하여 불특정한 고객을 대상으로 일반적인 이해당사자 요구사항을 추출한다.

요구사항 추출에 있어서 시장 주도형 제품 개발에서의 방법과 전통적인 요구사항 추출 방법이 있으며 방법 별로 차이가 있다. 요구사항이 추출 될 때 고객 주도형의 전통적인 방법에서는 분명한 사용자 또는 고객이 존재하지만, 시장 주도형의 제품 개발 시에는 사용자 또는 고객의 대상은 잠재적으로 불특정 다수의 고객을 예상할 뿐이다[23]. 다만 소비자와 관련한 정보는 시장조사를 통해 제품이 출시된 이후부터 제품과 관련한 다양한 요구사항을 지속적으로 피드백 받게 된다.

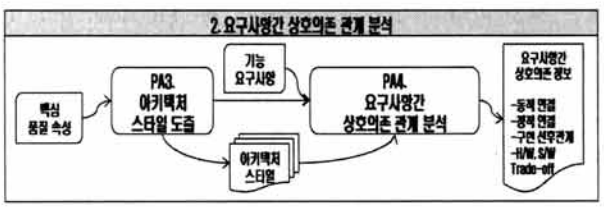
5.1.2 요구사항 정의 및 품질속성 식별(PA2)

본 활동은 요구사항 추출(PA1)활동으로부터 추출된 요구사항을 기반으로 기능 요구사항과 품질 속성을 정의한다. 여기서 품질 속성은 기능 요구사항에 대한 제약사항 또는 최적화 사항과 같은 비기능 요구사항에 관계되는 시스템이 지녀야 할 특성을 의미한다. 임베디드 시스템과 같은 시장 주도형 제품은 목표와 시나리오를 기반으로 개발 조직에서 제품 요구사항을 작성한다. 다시 말하면 기능 요구사항은 시나리오를 기반으로 추출, 정의하며, 주로 사용자 시나리오가 이용된다. 그리고 품질 속성은 기능 요구사항에 대한 제약사항 또는 최적화 사항을 기반으로 식별, 정의한다.

따라서 이 활동은 제품으로 개발할 임베디드 시스템의 전체 사양 및 기능성을 결정하여 품질 속성을 포함한 시스템 요구사항을 개발하는 목표를 가진다. 또한 요구사항이 가지는 문제점으로 요구사항이 상당히 많을 수 있다는 것과 추상화 수준에 따라 우선순위화, 의존관계 분석 등을 수행하는데 어려움이 발생할 수 있다. 이에, 목표 및 이해당사자와 관련이 있는 요구사항들을 그룹화하여 단위 요구사항 집합으로 식별해 낸다. 이러한 활동으로 생성되는 산출물은 시스템 요구사항과 품질 속성이 되며, 다음 활동인 PA3의 입력으로 이용된다.

5.2 요구사항간 상호의존 관계 분석

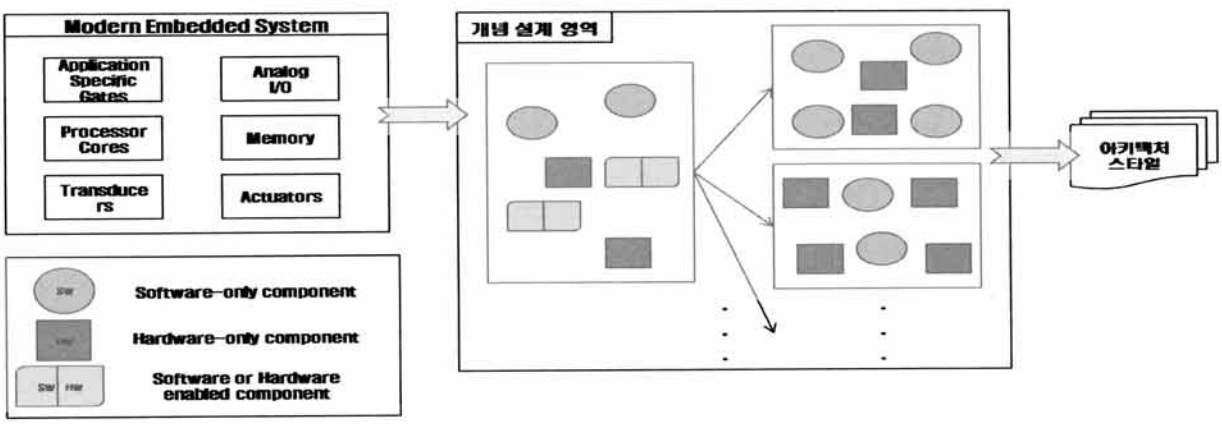
임베디드 시스템 요구사항간 의존관계 분석은 (그림 5)와 같이 시스템 요구사항과 아키텍처 스타일 후보 도출(PA3) 활동에서 도출된 아키텍처 스타일을 입력으로 단원 4.2에서 정의된 요구사항간 의존관계에 준거하여 분석(PA4)을 수행하는 것이다. 즉, PA3에서는 PA2에서 정의된 시스템 요구사항과 관련된 중요한 품질 속성을 기반으로 아키텍처 요구사항이 되는 아키텍처 스타일을 도출한다. 그리고 PA4에서는 PA2에서 정의된 기능 요구사항을 대상으로 요구사항간 의존관계인 정적, 동적 관계, 구현 선후 관계 그리고 PA3의 산출물인 아키텍처 스타일을 대상으로 아키텍처 스타일에 따른 상충관계에 대하여 분석을 수행한다.



(그림 5) 요구사항간 의존관계 정의 및 분석

5.2.1 H/W, S/W 아키텍처 스타일 도출(PA3)

시스템 요구사항에 대한 아키텍처 스타일 도출의 세부흐름은 (그림 5)와 같다. PA2에서 정의된 기능 요구사항과 식별된 핵심 품질속성을 고려하여 (그림 6)과 같이 아키텍처



(그림 6) 하드웨어 및 소프트웨어 요소간 역할 분담에 따른 아키텍처 스타일 도출

스타일을 정의한다. 즉, 임베디드 시스템을 개발할 때 제품의 특성이 드러나는 요소가 품질 속성이기 때문에 요구사항에 대한 품질 속성을 만족시킬 수 있는 여러 가지 솔루션(기능 요구사항에 대한 하드웨어, 소프트웨어 요소의 역할 분담)을 파악하여 아키텍처 스타일을 도출한다. 아키텍처 스타일은 개발 프로세스에서 설계 및 구현단계로 넘어가기 이전에 시스템의 요구사항을 달성하기 위해 기능 요구사항을 하드웨어와 소프트웨어 요소에 할당하는 역할 분담 단계에서 고려되며, 기본적인 설계 솔루션이 될 수 있는 개념적 수준의 아키텍처 요구사항을 의미한다. 예를 들어 대규모로 생산될 제품인 경우에 가격 경쟁력을 중요시한다면, 생산 비용이 적게 들어가도록 (그림 6)과 같이 개념 설계 영역에서 하드웨어나 소프트웨어 요소를 선택함에 있어서 하드웨어 요소보다 소프트웨어 요소에 중점을 두어야 한다. 반면, 빠른 릴리스를 중요시한다면 소프트웨어 요소보다 재사용이용이한 하드웨어 요소에 중점을 두는 아키텍처 스타일을 고려해야 하며, 아키텍처 스타일 도출에 관한 구체적인 방법은 [2, 22]에서 제시한 방법을 적용한다.

따라서 이 활동에서는 기능 요구사항에 대한 아키텍처 스타일과 통합을 위한 인터페이스 등과 같은 개념적 수준의 아키텍처와 관련된 요구사항들이 추가적으로 도출된다[3].

5.2.2 요구사항간 상호의존 관계 분석(PA4)

본 활동은 (그림 2) 우선순위 모델의 상호의존 분류를 활용하여, 시스템 요구사항이 가지는 의존관계인 정적 연결 및 동적 연결 그리고 구현 선후 관계뿐만 아니라 아키텍처 스타일에 따른 우선순위 요소 또는 아키텍처 요구사항간의 상충관계까지 분석하는 것이다. 우선순위에서 가장 핵심적인 요소는 우선순위 요소나 요구사항간 의존 및 상충관계를 파악하여 요구사항의 우선순위에 반영하는 것이다.

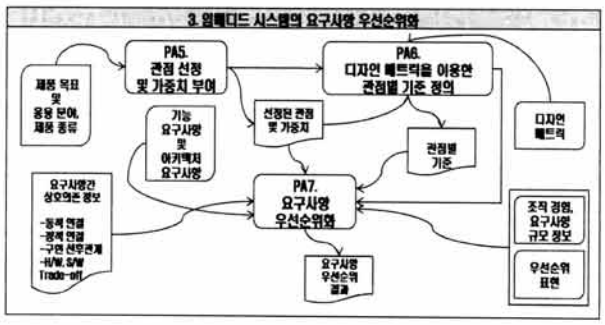
의존관계를 분석하기 위해서는 관계 종류를 정의하고 요구사항간 어떤 관계가 있는지 상호 비교를 통하여 식별하는 것이 일반적인 방법이다[6]. 따라서 이 활동에서는 PA2에서 정의된 기능 요구사항을 대상으로 요구사항간 의존관계인 시스템 구조상의 결합(Coupling)에 의한 정적 의존관계(Static Coupling), 시스템 구현시의 구현 선후 관계(Precedence by implement)[19], 시스템 운영시의 동적 의존관계(Dynamic Coupling) 및 아키텍처 스타일에 따른 상충관계(H/W, S/W Architecture Trade-off)에 준거하여 상호 비교를 통해 의존관계를 파악한다. <표 4>와 같이 기능 요구사항을 대상으로 요구사항간 전체적인 연결관계를 확인하고 런타임시에 연결의 유무를 구분하여 동적, 정적 연결을 식별한다. 이어서 요구사항간 구현 선후 관계를 식별한 뒤, 그 다음 PA3의 산출물인 아키텍처 요구사항을 대상으로 우선순위에 관련되는 (그림 3)의 요소들에 기반하여 상충관계를 분석한다. 분석된 의존 관계에 대한 정보는 산출물로 생성되어 다음 우선순위 활동(PA5)의 입력으로 이용되며, 우선순위화를 위한 정보를 제공한다.

본 논문에서는 [2]의 방법을 적용하며, [2]에서는 VHDL (VHSIC Hardware Description Language)을 이용하여 하드웨어 요소를 명세하고, 테스트 시뮬레이터를 통하여 분석을 수행한다. 아키텍처 요구사항에 대한 상충관계의 예로서 제품의 개발 목표가 가격 경쟁력 이라면 생산 비용이 적게 들어가도록 하드웨어 요소보다 소프트웨어 요소에 중점을 둔 요구사항이 목표에 부합한다. 이러한 경우 비용 절감을 통한 가격 경쟁력 향상은 만족시킬 수 있지만 일반적으로 개발 시간이 지연되고, 재사용성은 낮아지게 되는 단점을 가진다. 반면에 제품의 개발 목표가 빠른 출시를 통한 제품 인지도 확보라면 재사용성을 높여 개발 시간을 단축하기 위해 소프트웨어 요소보다 하드웨어 요소에 중점을 둔 요구사항이 목표에 부합한다. 이러한 경우 개발 시간이 단축되어 빠른 출시를 기대할 수 있지만 생산 비용이 전자보다 커지기 되고, 가격 경쟁력은 떨어지는 단점이 있으므로 적절한 절충이 필요하다.

5.3 임베디드 시스템의 요구사항 우선순위화

본 단계는 임베디드 시스템의 요구사항 우선순위 프로세스를 나타내는 (그림 4)의 요구사항 우선순위화 단계로써 임베디드 시스템의 요구사항에 대하여 우선순위를 결정하는 것이 목적이다. 일반적으로 제품에 대한 요구사항은 방대하기 때문에 모든 후보 요구사항을 구현하여 출시하는 것은 불가능하다. 따라서 요구사항 우선순위의 목적은 요구사항의 중요도를 분석하여 가용한 자원을 바탕으로 다음 릴리스로 출시될 제품의 구현을 위한 최적화된 요구사항 집합을 선정하여 정해진 출시 일정 안에 목표를 달성하고 고객의 만족도를 충족시켜 가치를 높이는 데에 있다.

4장의 우선순위 모델을 활용하여 우선순위 활동에 참여할 이해당사자와 우선순위를 위한 관점을 서로 대응되도록 선택하고, 관점별로 가중치를 부여한 후 점수 부여 기준을 정립한다. 따라서 우선순위를 수행할 때 제안된 우선순위 모델을 활용하여 요구사항의 의존성, 제품 개발 목표, 조직의 역량 등에 적합한 맞춤형 우선순위 기법을 설계하여 수행할 수 있다. 또한 요구사항간 우선순위 결정은 동일한 계층 수준의 요구사항들간에 수행되어야 하기 때문에 기능 요구사항과 아키텍처 요구사항으로 구분하여 수행한다. 요구사항 우선순위화 활동에 대한 세부 흐름은 (그림 7)과 같다. 구체적인 기술은 본 단원에서 각 세부 활동 별로 나누어 기술한다.



(그림 7) 임베디드 시스템 요구사항 우선순위화

5.3.1 관점 선정 및 가중치 부여(PA5)

본 활동에서는 이해당사자가 어떠한 관점을 기반으로 요구사항간 우선순위를 결정할 것인지에 대하여 우선순위를 위한 관점과 이에 대한 가중치를 부여하는 것이 목표가 된다. 우선순위 관점은 다양할 수 있으므로 어떤 관점을 선정하여 요구사항의 우선순위를 수행해야 하는지 결정이 필요하다. 이러한 관점 선정이 쉬운 작업은 아니지만 목표를 기반으로 적절한 관점을 선택하여 가중치를 부여할 수 있으며, 우선순위에 대한 비용과 난이도 문제를 고려하여 관점은 5개 이하로 선택한다[11, 12].

먼저 관점 선택의 기준은 첫째, 제품의 목표를 기반으로 선정하는 것이다. 요구공학에서 목표가 차지하는 비중은 크다고 할 수 있다. 목표는 요구사항을 식별하고 구조화하고 타당성을 증명하는 논리적인 메커니즘이며[28], 요구사항의 적절성에 대한 분명한 기준을 제공할 수 있다[29].

관점 선택의 두 번째 기준은 릴리스 플랜 수립을 위한 위험의 최소화에 초점을 두고 관점을 선정하는 것이다. 단원 2.1에서 언급한 바와 같이 아키텍처 관련된 위험, 신 기술과 중요 요구사항의 누락, 그리고 품질과 관계된 위험이 핵심 사항이기 때문이다.

관점의 중요도는 [6]에서와 같이 일반적인 목표에 대하여 Case로 나누어 상대적인 관점간에 가중치를 부여하며, 본 논문에서는 요구사항 우선순위 모델의 목표 요소를 활용하여 5가지 Case로 나누어 목표에 따라 상대적인 관점의 중요도를 제시하여 가중치를 부여할 수 있도록 한다. 선택된 관점에 대한 가중치 부여는 일반적으로 관점간의 상대적 중요도를 가중치의 합이 1이 되도록 부여 하며, 상황에 따라 가중치 부여에 대한 기준과 아래의 케이스별 관점의 중요도에 의거하여 가중치를 부여한다.

또한 선정된 관점의 하위 요소들간 가중치 부여에 대한 기준은 제품의 응용 분야 및 종류를 기반으로 부여하는 것이다.

왜냐하면 임베디드 시스템은 제품의 응용 분야 및 종류에 따라 요구되는 기능과 품질 특성이 각기 차이가 있기 때문이다. 예를 들어, 개발 관점에 속해있는 하위 관점들간 가중치 부여 문제를 생각해보자. 하위 관점이 개발 비용과 품질로 선정되었을 때, 제품의 응용 분야가 가격 경쟁력을 중요시하는 정보 단말이라면 개발 비용에 대한 측면이 품질에 대한 측면보다 가중치가 상대적으로 높을 수 있다. 반면에 제품의 응용 분야가 안전성을 중요시하는 중대한 시스템에 속한다면 품질적 측면이 비용적 측면보다 가중치가 상대적으로 높을 수 있다.

- (1st Case) 시장에서 상품의 인지도를 높이는 것이 중요한 목표일 경우에는 비즈니스 가치, 전략적 이점, 출시 일정, 고객 중요도와 같이 비즈니스와 고객의 관점이 중요하다. (관점의 중요도 순서: BA>CA>DA)
- (2nd Case) 품질 경쟁력과 제품 구색이 중요한 목표일 경우에는 고객 중요도, 재 사용성, 기술적 위험 최소화, 경쟁사와 같은 고객과 개발의 관점이 중요하다. (관점의 중요도 순서 CA>DA>BA)
- (3rd Case) 새로운 기능 개발 및 시장 점유율 유지가 중요한 목표일 경우에는 고객 중요도, 출시 일정, 비즈니스 가치와 같은 고객과 비즈니스의 관점이 중요하다. (관점의 중요도 순서: CA>BA>DA)
- (4rd Case) 전략적인 시장 점유를 통한 제품 판매량 달성이 목표일 경우에는 비즈니스 가치, 출시 일정, 개발 비용, 기술적 위험 최소화와 같은 개발과 비즈니스 관점이 중요하다. (관점의 중요도 순서: BA>DA>CA)
- (5rd Case) 유사 경쟁 제품간 가격경쟁력을 높이는 것이 중요한 목표일 경우에는 고객 중요도, 출시 일정, 비즈니스 가치와 같은 고객과 비즈니스의 관점이 중요하다. (관점의 중요도 순서: DA>BA>CA)

5.3.2 디자인 매트릭을 이용한 관점별 기준 정의(PA6)

본 활동은 PA5에서 우선순위화를 위해 선정된 관점들에 대하여 기준을 정의한다. 왜냐하면 일관성 있는 이해당사자 사이의 의사소통 및 요구사항의 우선순위 결정을 위해 관점별로 정량화된 기준이 필요하다. 즉, 객관적으로 정의된 기준이 있어야 이에 기반하여 참여 이해당사자가 요구사항의 중요도를 차별화하여 나타낼 수 있다.

임베디드 시스템은 그 응용 분야, 규모, 제품 종류에 따라 각기 다른 다양한 목표를 가지고 개발될 수 있으며, 그에 따라 요구되는 기능성이나 중요시되는 품질에 대한 특성도 달라질 수 밖에 없다. 따라서 어떠한 관점을 선정하고 가중치를 부여한 뒤, 이에 대한 기준을 정의할 때 개발 제품의 응용 분야 및 종류에 대한 이해가 선행되어야 하며, 이를 바탕으로 객관적인 평가를 위한 기준을 세워야 한다. 본 논문에서는 단원4.2에서 정의한 디자인 매트릭을 이용하여 가시적으로 측정할 수 있는 기준 선정에 집중하며, 이를 통해 보다 명확한 기준으로 이해당사자간 일관성 있는 평가가 가능하도록 한다.

디자인 매트릭은 시스템 개발에서 고려해야 할 제약사항이나 최적화 요구사항에 대하여 측정 가능한 기준을 제시해 주기 때문에 일반적으로 명확한 기준이 가장 필요한 관점을 순서대로 나열하면 개발 관점이나 비즈니스 관점 그리고 소비자 관점 순이 될 수 있다.

왜냐하면 개발 관점은 제품 개발에 직접적으로 관여되는 관점으로써 제품의 설계 및 구현 그리고 품질과 같은 측면이 속하기 때문이다. 또한 비즈니스 관점은 마케팅, 적시성, 제품 개발을 통한 손익 등의 측면이 관계되기 때문에 정량적인 기준이 요구된다. 가령, 개발 측면에서 제품의 성능과 대응되는 Performance, 신뢰성과 대응되는 Reliability와 Availability, 위험성과 대응되는 Testability 등은 <표 3>의 예시처럼 측정 가이드로 고려하여 정량적인 기준으로 정의할 수 있다. 본 활동을 통하여 관점별 기준 정의가 완료되면, 각 영역별로 전문가에 해당하는 이해당사자들이 기능 요구사항과 아키텍처 요구사항에 대하여 우선순위화를 수행한다.

5.3.3 요구사항 우선순위화(PA7)

본 활동에서는 (그림 7)에 준거하여 기능 요구사항과 아키텍처 요구사항을 대상으로 요구사항간 우선순위를 결정한다. 입력 정보는 요구사항 우선순위 결정의 대상이 되는 기능 요구사항과 아키텍처 요구사항이 기본이 된다. 다음으로 PA5에서 선정된 관점 및 관점별 가중치와 PA6에 준거한 관점별 기준으로 이해당사자는 대응되는 각 영역에 대하여 우선순위 결정을 수행할 수 있다. 이때, 우선순위 결정에서 반드시 고려해야 하는 요구사항간 의존성 및 상충성에 대한 정보를 참여 이해당사자는 의존성 분석 활동을 통해 숙지해야 한다.

즉, 기능 요구사항의 우선순위화에서 의존관계 정보 중에 동적, 정적 의존관계는 요구사항간 바인딩을 위해 파악이 필요하며, 구현 선후 관계는 요구사항간 우선순위 역전 문제를 방지하기 위해 필요하다. 상충관계는 기능 요구사항이 아키텍처 관련된 위험 그리고 품질과 관계된 위험을 수반하는지 여부를 아키텍처 요구사항과의 관련성을 통해 인지하기 위해 필요하다.

또한 특히, 상충관계는 우선순위에 관련한 하드웨어, 소프트웨어 요소의 특징에 기반하여 아키텍처 요구사항간 상충되는 관계를 이해하기 위해 필요하다.

이어서 우선순위를 결정하기에 앞서 요구사항의 우선순위를 어떠한 표기법으로 나타낼 것인지를 개발 상황에 맞추어 고려해야 한다. 우선순위 표기를 고려하기 위해 다음과 같은 선정기준을 사용한다.

- 기존 우선순위 방법의 특징을 이해하고 결정한다.
- 개발 조직의 요구사항 우선순위 활동에 대한 경험을 고려하여 결정한다.
- 요구사항의 규모(Requirement Scalability)를 고려하여 결정한다.

방법 선정에 앞서 첫째, 기존 우선순위 방법의 특징을 이해하는 것이 중요하다. 그 이유는 우선순위 표기법들간의 차이점은 어떤 척도(Scale)를 사용하는가 이다. 그 척도에는 명목(ordinal)과 비율(Ratio) 척도로 나눌 수 있으며, 명목 척도는 숫자 간격의 크고 작음에는 의미가 없으며 숫자의 순서에만 의미가 있고, 비율척도는 숫자의 간격에 의미를 부여할 수 있게 되지만, 복잡한 계산을 필요로 한다. 그 외 우선순위를 나타내는 표기법을 결정할 때 고려되는 항목은 [8, 10, 31-33]에서 참고하였으며, 해당 항목들은 다음과 같다.

- Time Consuming: 우선순위화에 소요되는 시간
- Ease of Use: 사용 편의성
- Informative: 우선순위의 분석 정보량
- Scalability: 요구사항의 규모가 클 때 적용이 용이한 정도
- Stakeholder Involvement: 개발 제품에 관련된 이해당사자들의 체계적 식별 유무
- Fault Tolerance: 우선순위가 제대로 수행되었는지 자체 점검 방법 유무
- Tool Support: 우선순위를 위한 Tool지원 유무

해당 항목을 기준으로 살펴보자면, 기존의 우선순위 표기법으로 우선순위를 숫자로 나타내고 평균으로 결정하는 Numerical Assignment[33] 및 Planning Game[32], 특정한 등급으로 나타내는 Ranking[10]이 전반적으로 확장성, 편의성, 시간소모 면에서 강점을 가지며, 분석 정보량은 미약한 것으로 나타났다. 또한 AHP(Analytic Hierarchy Process) 기법을 이용한 비용-가치 다이어그램[32]은 분석 정보량에서는 상대적으로 강점을 가지지만, 그 외 다른 항목에서는 좋지 못했다.

둘째, 개발 조직의 요구사항 우선순위 활동에 대한 경험을 고려해야 한다. 경험이 있다면 사용했던 방법을 그대로 적용할 수 있고, 보다 정보량이 많은 방법을 선택할 수도 있다. 하지만 이러한 경험이 없다면 사용하기 쉬운 방법을 선택하는 것이 바람직할 것이다.

셋째, 요구사항의 규모를 고려해야 한다. 요구사항의 규모가 매우 크다면 확장성(Scalability)이 좋은 방법을 선택하는 것이 적절하다.

상기 언급한 기준들에 대해 살펴본 바와 같이 요구사항의 우선순위를 어떻게 표현하는 가는 여러 가지 항목간에 trade-off가 있기 때문에 쉽지 않은 의사 결정이 될 수 있다. 즉, 분석의 정보량을 높이면서 시간 소모나 적용의 어려움을 희생할 것인지, 빠르고 간편한 우선순위화 수행을 위해 분석 정보량을 줄일 것인지를 고려해야 한다.

최종적으로 우선순위화가 완료되면 결과를 토대로 후보 요구사항을 목적에 부합하게 선택할 수 있다. 이 때 각 영역에 다수의 이해당사자가 참여할 수 있기 때문에 일관성 결여 문제를 고려하여 익명으로 우선순위 활동을 수행한 후 통합하는 것이 바람직하다. 또한 복수 개의 값을 처리하는 방법으로 델파이 방법 또는 중앙값을 활용할 수 있다.

6. 사례 연구 및 평가

본 논문에서 제시하는 상호의존 및 관점 기반의 임베디드 시스템 요구사항 우선순위 프로세스의 타당성을 검증하기 위해 핸드폰 개발의 일부 파트를 대상으로 사례를 적용하여 이에 대한 분석을 포함하여 기존 기법과 비교 평가를 하였다. 사례 적용 대상 핸드폰은 기본적인 기능에 500만 화소의 카메라를 탑재하여 카메라 기능이 핵심이 되는 제품이다.

6.1 사례 적용

제안된 프로세스의 적용 범위는 핸드폰 개발 일부 파트의 요구사항을 기초로 하여 요구사항을 추출하는 단계(PA1)를 시작으로 임베디드 시스템 요구사항 우선순위화(PA7)까지 사례를 적용하였다. 본 사례 연구의 대상이 되는 핸드폰 제품에 대한 요구사항 추출(PA1) 및 요구사항 정의(PA2)가 수행되어 핵심 기능을 중심으로 요구사항의 그룹화가 이루어져 기능 요구사항 9개(R1~R9), 아키텍처 요구사항 4개(R10~R13)의 요구사항을 식별하였다. 본 사례에서 제품의

개발은 카메라를 강점으로 가격경쟁력을 가지는 저가형(Low-end) 핸드폰 제품으로, 4장에서 제시된 일반적인 목표 Case중 전략적인 시장 점유를 통한 제품 판매량 달성으로 4번째 Case에 해당된다. 본 사례의 제품은 시장 분석을 통하여 200,000개 이상의 판매량을 기대하고 있다면 시장 진입은 6개월 이내로, 100,000개의 판매량을 기대하고 있다면 시장 진입은 6~12개월 사이로, 12개월을 넘어가는 시장 진입은 제품 출시가 무의미한 것으로 분석되었다. 또한 제품 종류는 임베디드 시스템에 포함되는 정보 단말 기기로 소프트웨어뿐만 아니라 하드웨어 요구사항 및 개발 일정이 고려되어야 하며, 개발 조직은 이와 같은 프로세스를 통해 우선순위화 및 릴리스 플랜을 수립해 본 경험이 없는 상태라고 가정한다.

6.1.1 요구사항 추출(PA1)

본 활동에서는 카메라 핸드폰에 대한 시장의 요구를 파악하여, 후보 요구사항을 추출한다. 요구사항 추출은 시장의 요구사항을 중심으로 추출하며, 개발조직은 마케팅 부서로부터 시장의 요구사항을 받는다. 시장의 요구사항은 고수준의 추상적 요구사항이 된다.

예를 들어, 카메라 핸드폰의 요구사항은 "적어도 50장 이상의 이미지를 찍고, 저장할 수 있어야 하며 컴퓨터에 업로드 가능해야 한다." 와 같은 요구사항이 된다. 또한 개발 비용과 개발 기간 등의 윤곽이 드러난다.

6.1.2 요구사항 정의 및 품질 속성 식별(PA2)

본 활동에서는 카메라 핸드폰에 대한 요구사항을 정의하고, 중요한 품질 속성을 식별한다. 시스템이 해야 하는 기능을 명세하는 단계로서 PA1에서 추출된 시장의 요구사항으로부터 구체적인 기능, 비기능(품질, 제약사항) 요구사항을 명세 한다. 사례 적용의 대상은 정의된 요구사항 중 핸드폰 개발 일부 파트에 해당하는 <표 4>의 13개 요구사항으로 선정했다. 선정된 요구사항은 기능 요구사항 및 아키텍처 요구사항을 포함한다. 요구사항으로부터 선정한 중요 품질 속성은 카메라 기능과 핸드폰에 관련된 성능(이미지 처리에 필요로 하는 시간), 사이즈(IC에 탑재되어 있는 기본 로직 게이트의 개수), 파워(시스템이 동작하는데 소모되는 전력), 에너지(배터리의 지속시간)이다.

6.1.3 H/W, S/W 아키텍처 스타일 도출(PA3)

본 활동에서는 카메라 요구사항(R2)에 대해 아키텍처 스타일 후보를 도출한다. 카메라 요구사항은 H/W 및 S/W 요소로 구성이 가능한 기능 요구사항으로서 아키텍처 스타일 도출이 필요한 기능 요구사항이다. 아키텍처 스타일 도출을 위한 첫 시작은 [2,5]의 방법을 적용하여, 요구되는 기능을 단일 목적 프로세서와 범용 목적 프로세서로 할당 하는 작업으로 출발한다. 본 사례에서는 저렴한 범용 목적 프로세서와 플레쉬 메모리의 연동을 시작으로 범용 프로세서에 기능을 모두 소프트웨어로 할당하고, 품질 및 제약사항에 관

〈표 4〉 요구사항간 상호의존 관계

Requirements		Function Requirements									Architecture Requirements			
		R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13
		SD Card	Camera	Phonebook	PC Sync	SMS	Capture Operation	MMS	Product Case	Upload/ Download	Microcontroller alone	M.C and CCDPP	M.C and CCDPP /Fixed-Point DCT	M.C and CCDPP/DCT
R1	SD Card	-	DC Mandatory				DC Optional		SC Mandatory					
R2	Camera	DC Mandatory	-		DC Optional				SC Mandatory		Trade-off	Trade-off	Trade-off	Trade-off
R3	Phonebook			-						DC Optional				
R4	PC Sync		DC Optional		-									
R5	SMS					-		P→						
R6	Capture Operation	DC Optional					-							
R7	MMS					P←		-						
R8	Product Case	SC Mandatory	SC Mandatory						-					
R9	Upload/Download			DC Optional						-				
R10	Microcontroller alone		Trade-off								-			
R11	M.C and CCDPP		Trade-off									-		
R12	M.C and CCDPP /Fixed-Point DCT		Trade-off										-	
R13	M.C and CCDPP/DCT		Trade-off											-

범례(상호의존 관계 명칭 / 정의)	
SC Mandatory	Static Coupling Mandatory
SC Optional	Static Coupling Optional
DC Mandatory	Dynamic Coupling Mandatory
DC Optional	Dynamic Coupling Optional
P→	Precedence(r1 → r2 ; r1 선 구현)
P←	Precedence(r1 ← r2 ; r2 선 구현)
Trade-off	If r1 increase then r2 decrease vice versa

련한 비기능 요구사항(성능, 사이즈, 파워, 에너지)을 고려하여 단일 목적 프로세서를 추가 할당하는 방식으로 접근 했다. 특정한 여러 아키텍처 매핑에 따른 모든 구현 집합이 후보가 된다. 아키텍처 스타일은 H/W와 S/W간 역할 분담을 통하여4개의 아키텍처 요구사항을 식별했다. 식별된 아키텍처 요구사항은 각각 H/W와 S/W의 할당 정도에 차이를 가지며, 이것들이 가지는 특징은 다음과 같다.

- 첫 번째 Microcontroller alone(R10)는 범용 프로세서로 모든 기능을 처리하는 방법으로 비용, 전력소모, 개발기간 등은 평이하지만, 처리속도 면에서 매우 취약하다.
- 두 번째 Microcontroller and CCDPP(R11)는 이미지 촬영에 특화된 센서를 가진 프로세서를 추가하여 첫 번째에 비해 처리속도가 개선되었으며, 개발이 용이하여 위험성이 낮다.
- 세 번째 Microcontroller and CCDPP/Fixed-Point DCT (R12)는 하드웨어의 구현 비용이 비교적 크지만 빠른 시간 내에 개발이 가능하며, 처리속도와 배터리의 효율에서 강점을 가진다.
- 네 번째 Microcontroller and CCDPP/DCT(R13)는 개발 비용과 개발 시간에 대한 위험성이 매우 크지만 처리속도, 전력소모 면에서 가장 좋은 성능을 가진다.

상기 아키텍처 요구사항은 우선순위를 위한 관점에 중요한 영향을 미치는 개발 및 생산 비용, 개발 시간, 품질(성능, 사이즈, 파워, 에너지)과 같은 특성을 가지고 있으며, 각각 여러 요소간 차이를 가진다. 따라서 PA4에서 각 아키텍처 요구사항에 따른 요소간 상충분석이 필요하다.

6.1.4 요구사항간 상호의존 관계 분석(PA4)

요구사항간의 상호 비교를 통해 4가지 상호의존 관계를 식별한다. <표 4>의 결과에 따라서 SD Card(R1)와 Camera (R2)는 사진 촬영 후 저장 공간이 필요로 하기 때문에 Dynamic Coupling Mandatory(정적 의존)가 되고, Camera(R2)와 Product Case(R8)는 물리적인 사이즈 제약과 강하게 연결되어 있기 때문에 Static Coupling Mandatory(동적 의존)이 된다.

따라서 R1과 R2 그리고 R2와 R8은 동일 릴리스로 묶여야 하며, R2의 연결성 때문에 R1, R2, R8이 동일 릴리스로 묶여야 함을 알 수 있다. 또한 SMS(R5)와 MMS(R7)간에는 Precedence by implement(선구현) 관계가 있음을 식별 했으며, R2와 아키텍처 요구사항인 R10~R13은 상충관계를 가지는 것으로 파악되었다.

즉, PA3에서 R2에 대하여 도출된 아키텍처 요구사항들은 각각 개발 비용, 개발 시간 그리고 품질 특성(성능, 사이즈, 파워, 에너지)면에서 차이를 지니고 있으므로, 어떠한 아키텍처 요구사항을 선택하느냐에 따라서 카메라 요구사항에 대하여 개발 비용, 기발 시간 그리고 비기능적인 요소들에서 trade-off가 발생하므로 상충관계에 있다고 할 수 있다.

따라서 R1, R2, R8가 바인딩 되고, R5는 R7 이전에 구현되어야 하며 그리고 R2와 R10~R13은 상충관계에 있으므로 이해당사자간 우선순위 협의를 통한 결정이 필요하다.

6.1.5 관점 선정 및 가중치 부여(PA5)

본 활동의 목표는 우선순위 결정을 위한 관점과 이해당사자 그리고 관점에 대한 가중치를 부여하는 것이다. 요구사항 우선순위를 위해 먼저 개발의 목표와 우선순위 모델을 통해 우선순위 방법과 관점을 선택한다. 이로써, 제품의 특성(응용 분야와 제품 종류)에 따라 다양한 관점을 고려한 방법과 관점을 선택하고 가중치를 부여함으로써 합리적인 요구사항의 우선순위화를 달성할 수 있다. 본 사례 연구 제품의 개발 목표는 전략적인 시장점유를 통한 제품 판매량을 달성하는 것이며, 카메라를 강점으로 가지는 저가형(Low-end) 핸드폰 제품이다.

관점 선정에 있어서 사례 적용 제품은 4장에서 제시된 일반적인 목표 Case중 4번째에 해당되므로, 비즈니스 가치, 출시 일정, 개발 비용, 기술적 위험 최소화와 같은 비즈니스와 개발 관점이 중요하다(관점의 중요도 순서 BA>DA>CA). 따라서 전략적인 출시를 통한 시장점유가 중요하므로 BA에서는 빠른 개발이 중요한 우선순위 관점이 되며, 저가형 제품이므로 DA에서는 제품 개발 비용과 위험성이 중요시 된다. 그래서 BA와 DA에서 각각 Time-to-market, Development Cost, Technical Risk관점을 선정 하였다.

또한 가중치 부여는 목표 Case 4번째에 의거, BA에 60%, DA에 40%를 부여하기로 했다. 참여 이해당사자로는 선정한 관점에 대응하도록 BA관점의 Time-to-market에 대하여 마케팅 전문가 2명, DA 관점의 Development Cost 및 Technical Risk에 대하여 개발자 2명과 프로젝트 매니저를 선정하였다.

6.1.6 디자인 매트릭을 이용한 관점별 기준 정의(PA6)

본 활동의 목표는 PA5에서 선정된 관점에 대하여 이해당사자들이 우선순위 결정을 위한 공통된 기준을 정의 하는 것이다. 본 사례에서는 상충 관계에 있는 아키텍처 관련 요구사항간 trade-off를 이해당사자간 충분한 이해를 통해 기준을 협의하는 것이 중요하다.

관점별 기준 정의에 앞서, 디자인 매트릭은 요구되는 핵심 품질 속성에 수반되는 기술적 위험성을 고려하기 위해 핵심 품질 속성에 대응하도록 한정했다. 즉, <표 3>와 같이 이미지 처리하는데 필요로 하는 시간(Performance), IC에 탑재되어 있는 기본 로직 게이트의 개수(Size), 이미지 처리 중 소모되는 전기 에너지의 평균 측정값(Power), 배터리의 지속시간(Energy)에 대하여 측정할 수 있는 디자인 매트릭을 식별하였다.

우선순위를 위한 마지막 준비 과정으로 핵심 품질 속성간 trade-off 및 관점별 명확한 기준 정의를 위해 이해당사자들이 함께 <표 5>와 같이 협의하여 정의하였다.

〈표 5〉 관점별 정량화 기준 정의

Scale	Time-to-Market	Development Cost	Technical Risk
5	Very High (미미한 일정 지연 초래 예상)	Very High (미미한 개발 비용 상승을 예상)	Mandatory (제품의 품질 저하는 눈에 띄지 않아야 함)
4	High (전체 일정의 5% 미만의 지연 초래 예상)	High (전체 개발 비용의 10% 미만 상승 예상)	Very Important (품질 저하는 제품 경쟁력에 막대한 영향을 미침)
3	Moderate (전체 일정의 5~10%의 지연 초래 예상)	Moderate (전체 개발 비용의 10~20% 상승 예상)	Rather Important (품질 저하는 제품 경쟁력에 영향을 미침)
2	Low (전체 일정의 10~20%의 지연 초래 예상)	Low (전체 개발 비용의 20~40% 상승 예상)	Not Important (품질 저하는 제품 사용에 미미한 영향을 미침)
1	Very Low (전체 일정의 20% 이상의 지연 초래 예상)	Very Low (전체 개발 비용의 40% 이상 상승 예상)	Does not matter (품질 저하는 제품 사용과 관련이 없음)

[출처 : PMBOK Guide 2004 Figure 11-2]

6.1.7 요구사항 우선순위화(PA7)

본 활동의 목표는 최종적으로 요구사항의 우선순위를 결정하는 것이다. 즉, PA1~PA6까지의 우선순위화를 위한 준비 활동들을 거쳐 마지막으로 우선순위화(PA7) 활동에서 이해당사자가 요구사항의 우선순위를 결정한다.

본 사례는 요구사항의 우선순위 표기법을 결정하는 기준으로 우선순위 프로세스 경험이 없고, 요구사항의 숫자 많지 않은 점에 착안하여 상대적으로 쉬운 방법을 선택하는 것이 바람직하다고 판단하였다. 그래서 Numerical Assignment 방법을 사용한다.

요구사항의 우선순위화는 기능 요구사항간 우선순위와 아키텍처 스타일의 우선순위를 구분하여 수행하였으며, 각각의 우선순위 결과는 <표 6>, <표 7>과 같다. <표 6>에서와 같이 상호의존 관계를 통해 바인딩 되어 식별된 요구사항에 대해 본 논문에서 제시한 우선순위 가이드를 따라 선정된 관점으로 각 이해당사자들이 익명으로 점수를 부여하

〈표 6〉 기능 요구사항의 우선순위 결과

구분	비즈니스 관점			개발 관점								Priority	Rank
	Time-to-market			Development Cost				Technical Risk					
Weight	0.6			0.2				0.2					
Stakeholders	S1	S2	Median	D1	D2	PM	Median	D1	D2	PM	Median		
R1_R2_R8	5	5	5	5	5	5	5	5	5	5	5	5	1
R3	4	5	4.5	5	4	5	5	1	1	1	1	3.9	4
R4	3	4	3.5	3	4	3	3	1	2	2	1	3.1	6
R5	5	5	5	5	5	5	5	2	2	2	2	4.4	3
R6	5	4	4.5	3	4	4	4	4	5	5	5	4.5	2
R7	3	2	2.5	4	4	5	4	3	3	4	3	2.9	7
R9	4	4	4	4	3	4	4	2	2	3	2	3.6	5

〈표 7〉 아키텍처 요구사항의 우선순위 결과

구분		비즈니스 관점			개발 관점								Priority	Rank
		Time-to-Market			Development Cost				Technical Risk					
Weight		0.4			0.3				0.3					
Stakeholders		S1	S2	Median	D1	D2	PM	Median	D1	D2	PM	Median		
R2(Camera)	Requirement 10(Microcontroller Alone)	3	2.5	2.75	4	4	5	4	1	1	1	1	2.65	3
	Requirement 11 (M.C and CCDPP)	2.5	2.5	2.5	3	4	4	4	3	3	3.5	3	2.9	2
	Requirement 12(M.C and CCDPP/Fixed-Point DCT)	3	3.5	3.25	4	3.5	3.5	3.5	4	4	5	4	3.65	1
	Requirement 13 (M.C and CCDPP/DCT)	2	2	2	2	2.5	2	2	5	5	5	5	2.6	4

였으며, 일관성 결여 문제를 고려하여 복수 개의 응답에 대해 중앙값을 취하여 나타내었다. 우선순위화는 각 요구사항에 대한 관점별로 대응하는 중앙값과 가중치를 곱한 후에 더한 값으로 점수를 도출하여 요구사항의 순위를 구했다. 우선순위 결과 R1_R2_R8이 1순위로, R7은 7순위로 최하 순위가 되었으며, 각 요구사항간 우선순위를 결정할 수 있었다.

카메라 요구사항에 대한 아키텍처 스타일인 4개의 아키텍처 요구사항을 본 논문의 방법을 따라서 <표 7>과 같이 우선순위화 하였고, 우선순위 결과 R12가 1순위로, R13은 가장 낮은 4순위로 나타났다.

6.2 사례 적용에 대한 분석 평가

본 단원에서는 제시한 임베디드 시스템의 우선순위 프로세스를 적용 사례를 중심으로 세 가지 측면에서 적용하지 않은 경우에 대해 문제점을 분석한다.

- 첫째, 관점의 측면이다. 프로젝트의 목표를 기반으로 관점을 선택하지 않는 다면, 관점의 가중치에 대한 일관성이 결여 되는 문제점이 발생한다. 후보 요구사항은 이해당사자간 협의를 통해 선택되어야 하며, 이러한 협의는 명확하고 객관적인 기준하에 이루어 지는 것이 바람직하다. 만약 그렇지 않다면 궁극적으로 요구사항을 우선순위화하는 의미가 축소되며, 이해당사자간 상이한 중요도 및 전문성 때문에 갈등이 유발될 수 있다. 또한, 임베디드 시스템은 다양한 분야를 가지기 때문에 응용분야 및 제품 종류에 따라 다양한 관점을 가지고 우선순위화 되어야 한다. 본 논문에서는 우선순위 모델을 활용하여 프로젝트 목표 및 개발 상황에 부합하도록 맞춤형 우선순위를 위한 가이드 라인을 제시하고 있다.
- 둘째, 요구사항간 상호의존성의 측면이다. 동적 상호의존 관계인 R1과 R2를 고려하지 않았을 경우를 고려해 보자. R1의 기술적 위험성이 낮기 때문에 요구사항간 우선순위가 7위로 결정된다면, 동일 릴리스로 묶이지 않을 수 있고, 추후에 테스트시 문제가 생길 수 있으며, 출시가 되었을 경우 제품에 대한 고객의 만족도는 좋지 않을 것이다. 또한, H/W 및 S/W 요소로 구성이 가능한 요구사항인 R2는 R10~R13과 상충관계에 있다. 만약 상충관계를 고려하지 않았다면, 아키텍처 요구사항인 R10~R13과의 관계의 미확인으로 인해 기술적 위험도, 개발 일정, 개발 비용과 같이 프로젝트에서 민감한 요소들이 증가하고 감소하는 정보를 우선순위 관점의 고려대상에서 제외되는 문제가 발생한다. 그리고 R2의 우선순위가 낮아져 최우선적으로 구현이 되지 못했을 것이다. 결국, 요구사항의 대부분이 제약사항이나 최적화 사항에 해당되는 하드웨어 요구사항과 관계되는 의존성을 최우선적으로 고려하지 않고, 개발이 이루어진다면 예상치 못한 문제가 발생시 되돌리기 힘들며, 해결하는데 많은 비용이 들 수 있다.
- 셋째, 일관성 있는 요구사항 우선순위 프로세스 측면이다. 특히, 제품 명세 및 설계 결정 단계에서는 1차적으로 프로젝트 기획의 연장선상에서 목표 시스템의 주요 성능(수행성능, 메모리 사용량, 시스템 크기 및 무게 등) 혹은 시장(제품당 가격, 출시 시기 등) 목표 및 제약사항을 정확히 설정하고, 2차적으로 프로젝트 개발 참여자들 간의 합의 및 이해를 동시에 수행하면서 릴리스를 위한 개발 제품의 요구사항을 선택해야 한다[24]. 결국, 상기 단계에 적용이 가능하고, 이해당사자간 협의가 고려되어 있는 체계적인 우선순위 프로세스가 요구된다. 따라서 본 논문에서는 상기 단계들을 반영하여 우선순위 프로세스를 체계화 시켰다. 만약, 소프트웨어와 하드웨어의 동시 설계 단계나 이해당사자간 협의가 고려되어 있지 않은 기존의 우선순위 기법을 그대로 적용시킨다면 구현이나 릴리스가 되는 시점에서 일정 연기, 비용 초과, 제품 가치의 하락과 같은 위험성을 최소화 시킬 수 없게 된다.

6.3 비교 평가

기존의 요구사항 우선순위에 대한 연구들은 제안 프로세스에 비해 상이한 적용 도메인과 목표를 지니므로 직접적인 정량적 평가는 어렵기 때문에 비교를 통해 정성적으로 평가한다. 관련연구 2.2 단원에서 살펴본9개의 우선순위 기법과의 비교 평가를 위해 대상 도메인을 임베디드 시스템으로 선정하고, 우선순위 모델에서 제안한 8가지 평가 요소를 반영하여 비교 평가한다. 다음 <표 8>은 기존 연구와의 비교 평가 기준의 항목으로써 우선순위 모델의 8가지 평가 항목에 대하여 우선순위화 활동에서 지녀야 할 기본 특징을 포함하고 있는가를 평가하는 요소와 하드웨어와 소프트웨어를 포함하여 복잡한 의존성을 지닌 임베디드 시스템의 특징을 고려했는가를 평가하는 요소 그리고 우선순위 기법에서 일반적으로 지원한다면 유용한 특징인 매트릭의 사용과 우선순위 활동 지원 도구를 제공하는가에 대한 평가 요소로 분류하여 기술한다.

의존과 관점 기반 임베디드 시스템의 요구사항 우선순위 프로세스라면 임베디드 도메인에 적합한 이해당사자, 우선순위 요소 그리고 강한 의존성과 다양한 개발 상황이 존재하는 임베디드 도메인의 특징을 고려하기 위해 유연성, 의존성 분석 등을 반영하고 있어야 한다.

각 연구에서 평가 요소에 대한 평가 값은 아래에 언급한 평가 기준에 맞추어 우수, 보통, 미흡의 3가지로 나누었으며, 기존 연구된 방법론에 대해 상호 비교한 연구를[7,10,31-33] 참고하여 본 논문에서 상대적 평가를 통해 부여하였다. 또한 본 논문에서 제안한 프로세스에 대한 평가는 단원6.1의 적용 사례를 기준으로 평가하였다.

유연성(Softness)의 측면은 다양한 개발 상황 및 목적에 부합하도록 우선순위를 위한 관점과 우선순위 표기방식 등의 선택에 유연성이 있는지를 판단하기 위한 기준이다. 임베디드 도메인은 다양한 응용 도메인을 가지고 있기 때문에 시스템의 규모나 요구사항이 다양할 수 밖에 없으며 이에 따라 개발 상황이나 목적이 차이를 지닐 수 있다. 유연성에

〈표 8〉 기존 연구와의 비교 평가 기준

비교 항목		내 용
Prioritization	Softness	우선순위를 위한 관점과 우선순위 표기방식 등의 선택에 유연성이 있는가?
	Informative	우선순위화 이전/이후에 분석 정보량이 많은가?
	Stakeholder Involvement	개발 제품에 관련된 이해당사자들이 체계적으로 식별되는가?
Considering Dependency	Static Dependency	요구사항 간의 정적 상호의존 관계 파악을 위한 의존 관계 종류가 정의되어 있는가?
	Dynamic Dependency	요구사항 간의 동적 상호의존 관계 파악을 위한 의존 관계 종류가 정의되어 있는가?
	HW/SW ArchitectureTrade-off	아키텍처 요구사항에 의해 발생하는 우선순위 요소 및 요구사항 간의 상충 관계 파악을 위한 활동이 정의되어 있는가?
Common	Metric Support	우선순위를 위한 전반적인 활동에서 주요한 요구사항에 대해 이용할 수 있는 측정 가능한 척도를 제공하는가?
	Tool Support	우선순위 활동을 위한 Tool이 지원 되는가?

대한 기준은 우선순위를 위한 관점과 표기방식을 상황에 맞게 선택할 수 있다면 우수함, 한정된 일부 관점만을 제공한다면 보통, 특정 관점과 표기방식만을 사용한다면 미흡으로 판단한다.

분석 정보량(Informative)의 측면은 우선순위화를 위해 이해당사자에게 제공되는 정보량과 우선순위화 이후 우선순위 결과가 지니고 있는 정보량의 차이를 판단하기 위한 기준이다. 분석 정보량은 우선순위화 이전에 이해당사자에게 우선순위에 관계되는 의존요소 등과 같은 분석정보를 제공하거나 우선순위화 이후에 요구사항간 우선선위 순서뿐만 아니라 중요도의 차이를 수치적인 정보로 유효하게 제공하는 것을 의미한다. 분석 정보량에 대한 기준은 우선순위 이전과 이후에 제공되는 정보가 의미 있다면 우수함, 우선순위 이후에 나타나는 정보가 의미 있다면 보통, 정보가 의미 없다면 미흡으로 판단한다.

이해당사자 참여(Stakeholder Involvement)에 대한 측면은 개발 제품에 관련된 다양한 영역의 이해당사자들이 체계적으로 식별되는지의 여부를 판단하기 위한 기준이다. 이해당사자 참여에 대한 기준은 이해당사자가 비즈니스, 고객, 개발 영역에서 빠짐 없이 식별되었다면 우수함, 한 가지 영역이 제외되어 식별되었다면 보통, 특정한 영역에서만 식별되었다면 미흡으로 판단한다.

정적 의존(Static Dependency)에 대한 측면은 요구사항간의 정적 상호의존 관계파악을 위한 의존관계 종류와 파악을 위한 활동이 정의되어있는지를 판단하기 위한 기준이다. 정적 의존에 대한 기준은 관계파악을 위한 관계 종류와 프로세스상의 활동이 정의되었다면 우수, 관계 종류는 정의되었지만 프로세스상에 구체적 활동이 정의되지 않았다면 보통, 관계 종류가 정의되지 않았다면 미흡으로 판단한다.

동적 의존(Dynamic Dependency)에 대한 측면은 요구사항간의 동적 상호의존 관계파악을 위한 의존관계 종류와 파악을 위한 활동이 정의되어있는지를 판단하기 위한 기준이다. 동적 의존에 대한 기준은 관계파악을 위한 관계 종류와 프로세스상의 활동이 정의되었다면 우수, 관계 종류는 정의되었지만 프로세스상에 구체적 활동이 정의되지 않았다면 보통, 관계 종류가 정의되지 않았다면 미흡으로 판단한다.

우선순위 요소 및 요구사항간 상충(HW/SW Architecture Trade-off)에 대한 측면은 시스템 요구사항이 하드웨어와 소프트웨어 요소와 매핑이 되면서 도출되는 아키텍처 요구사항에 의해 발생하는 요구사항간 또는 우선순위 요소간 상충되는 관계를 의미하며, 이러한 관계를 파악하기 위해 상충관계와 파악을 위한 활동이 정의되어있는지를 판단하기 위한 기준이다. 우선순위 요소 및 요구사항간 상충에 대한 기준은 상충관계와 프로세스상의 분석 활동이 정의되었다면 우수, 상충관계는 정의되었지만 프로세스상에 분석 활동이 정의되지 않았다면 보통, 상충관계가 정의되지 않았다면 미흡으로 판단한다.

매트릭 사용(Metric Support)의 측면은 우선순위 프로세스 주요한 요구사항에 대해 측정 가능한 척도를 사용하여 요구사항간 또는 우선순위 요소간 관계를 분석하거나 선정된 관점에 대한 정량적인 기준을 정의하는지의 여부를 판단하기 위한 기준이다. 매트릭 사용에 대한 기준은 우선순위 프로세스 전반에 걸쳐 매트릭을 식별, 정의하여 체계적으로 관계 분석 및 관점별 기준의 정량화에 사용한다면 우수, 매트릭을 정의하지 않고 직관적인 기준으로 대체하여 사용한다면 보통, 어떠한 기준을 제시하지 않는다면 미흡으로 판단한다.

마지막으로 우선순위 도구 지원(Tool Support)의 측면은 우선순위 활동을 위한 도구가 지원되는지의 여부를 판단하기 위한 기준이다. 도구 지원에 대한 기준은 시간소모량 측면에서 우선순위 활동을 위한 도구가 지원된다면 우수, 도구 지원이 필요 없을 만큼 기법이 간단하다면 보통, 시간소모량이 크면서 도구 지원이 안되면 미흡으로 판단한다.

제안 기법과의 비교에 사용된 기존 연구들은 단원 2.3의 <표 2>에 나타나 있는 9개의 기법들이다. <표 9>는 <표 8>의 평가 기준에 따라서 기존 연구와의 비교 평가한 결과를 보여준다.

<표 9>의 평가 결과를 보면, 평가 항목 중 임베디드 도메인의 특징과 관련되어 넓은 응용 분야에 따른 다양한 개발 상황 및 목적을 커버하는 유연성, 소프트웨어 요소뿐만

〈표 9〉 기존 연구와의 비교평가

Method		IFM	EVOLVE	ARP	$100 test	Numerical Assignments	Planning Game	Ranking	Cost-value Diagram	Wiegers' Method	제안기법
Prioritization	Softness	1	1	3	1	1	1	1	1	1	3
	Informative	2	2	2	2	2	2	2	3	3	3
	Stakeholder Involvement	3	3	3	3	3	3	1	2	2	3
Considering dependency	Static dependency	2	2	2	1	1	1	1	1	1	2
	Dynamic dependency	1	1	2	1	1	1	1	1	1	2
	HW/SW Trade-off	1	1	1	1	1	1	1	1	1	2
Common	Metric Support	1	1	1	1	1	1	1	2	1	2
	Tool Support	1	3	1	1	1	1	1	1	1	1
Total Summation		12	15	15	11	11	11	10	12	11	18

[범례] 3: 우수, 2: 보통, 1: 미흡

아니라 하드웨어 요소가 가미되어 개발 단계의 복잡성에 따른 강한 의존성을 반영하기 위한 의존성 분석 부문에서는 기존의 우선순위 기법들은 낮은 점수를 보인다. 기존 연구들은 릴리스 플랜 수립을 고려한 일부의 기법들을 제외하고는 요구사항간 의존성을 전혀 고려하지 않았으므로 요구사항간 의존성을 반영하는 것에 대단히 미흡하며, 전체적으로는 소프트웨어 도메인을 전제로 삼고 있기 때문에 임베디드 시스템의 요구사항에 대한 의존성 분석은 반영되지 않았다. 반면 기본적으로 요구사항 자체의 우선순위를 위한 연구들이므로 이해당사자 참여나 분석 정보량에서는 비교적 우수함을 보인다. 특히 분석 정보량에 있어서 Cost-value Diagram과 Wieger's Method가 요구사항의 우선순위 결과가 나타내는 분석 정보량에서 뛰어남을 보인다. 그러나 단순히 소프트웨어의 포괄적인 위험성, 중요도, 가치와 같은 비즈니스 관점에 치중되어 있으며 개발 관점에서의 우선순위 정보는 매우 빈약하다. 반면, 기존 기법과는 달리 제안 기법은 우선순위화를 위해 시스템 요구사항을 이루는 아키텍처 요구사항에 대한 중요도와 의존성을 고려하므로 분석 정보를 근거로 하여 개발 조직내의 분석가와 설계자간 의사소통에 기여할 수 있다고 판단된다. 또한, 기존 연구 중 100$ Test, Wieger's Method, ARP는 우선순위를 위한 관점 자체가 모호하거나 관점에 대한 기준이 모호한 문제점이 존재하지만 본 논문에서는 디자인 매트릭을 이용하여 관점별 기준을 정량화시켜 이해당사자간 우선순위 기준의 모호함을 보완하였다고 평가된다.

전체적으로 보았을 때에 총점이18점으로 기존의 연구들에 비하여 우수한 편으로 볼 수 있다. 임베디드 도메인의 특징인 응용 도메인의 다양성으로 인해 개발 조직은 다양한 목표와 상황에 놓일 수 있으므로 제품의 목표와 개발 조직의 경험 등에 맞추어 관점과 표기방식을 선택 또는 응용할 수 있다. 특히 소프트웨어와 하드웨어 요소의 구조에 관계되는 아키텍처 요구사항에 따른 우선순위 요소 또는 요구사항간 상충관계를 분석하여 이해당사자가 비용, 시간 등과 관계되는 최적화 사항 및 위험성을 이해하고, 이를 우선순위 결정에 반영할 수 있다. 따라서 제안 프로세스는 임베디드 시스템에 초점을 맞춘 우선순위 모델에 기반한 요구사항 우선순위 프로세스로 임베디드 시스템이라는 도메인에 우수한 프로세스로 볼 수 있다.

본 논문의 한계점은 적시성, 개발 비용 등과 같이 민감한 우선순위 요소는 고려하지만 분석 수준이 단순한 요소에 기반하기 때문에 아직 미흡한 단계로 평가된다. 본 논문에서는 하드웨어 요소와 소프트웨어 요소가 가지는 개발 비용, 개발 시간, 처리속도, 전력소비, 유연성과 같이 단순화시킨 일부 요소들만을 고려하여 우선순위 요소 또는 요구사항간 상충관계를 분석하기 때문에 임베디드 시스템이 가지는 다양한 특성을 모두 반영시키기에는 어려운 문제점을 가진다. 또한 본 프로세스의 적용 절차가 길어 편의성이 떨어지고 우선순위 활동 지원 Tool 구현이 미흡해 아직 개선의 여지가 남았다고 볼 수 있다.

7. 결론 및 향후 연구

임베디드 시스템에서 요구사항에 대한 우선순위는 요구공학에서 특히, 오픈 시장을 고객으로 유사제품간 경쟁이 심한 제품 개발에 있어서 핵심적 활동이다. 기존 기법들은 요구사항간 상충되는 의존관계를 반영하지 못했고, 하드웨어 요소를 제외한 소프트웨어에 도메인에 중심을 둔 우선순위 기법을 제안했기 때문에 임베디드 시스템의 우선순위화에 적합하지 않다.

본 논문은 임베디드 시스템을 위해 의존관계 및 다양한 관점에 기반한 요구사항간 우선순위 모델과 프로세스를 제시했다. 특히, 우선순위 모델에서는 임베디드 시스템에서 요구사항 우선순위 결정에 영향을 주는 우선순위 요소들을 관점별로 계층화해서 정의하여, 이를 우선순위 프로세스에서 활용한다. 요구사항의 우선순위 프로세스는 요구사항을 정의하고, 하드웨어와 소프트웨어 요소의 역할 분담을 통해 아키텍처 스타일을 도출하며, 요구사항간 의존성을 4가지의 의존관계로 정의하고 분석한다. 그 다음 이해당사자의 관점을 선정하고, 관점에 대한 가중치를 부여하며, 관점별 정량화된 기준을 정립한다. 이어서 우선순위 표기법을 선정하고, 기능과 아키텍처 요구사항을 구분하여 우선순위를 결정하는 과정으로 구했으며, 평가를 위해 핸드폰 개발 적용사례를 통해 그 신뢰성과 적용성을 보였다.

본 기법을 적용하여 우선순위화에 참여하는 이해당사자는 분석된 요구사항간 의존성과 아키텍처 스타일에 따른 우선순위 요소 또는 아키텍처 요구사항간 상충관계를 요구사항의 우선순위 결정에 반영할 수 있다. 또한 임베디드 시스템에 초점을 둔 요구사항 우선순위 프로세스의 정립을 통하여 보다 충족스러운 요구사항의 우선순위 결과를 기대할 수 있으며, 제품 개발 목표와 개발 조직의 상황에 맞는 요구사항 선택으로 최적화 및 위험의 축소 등을 반영한 릴리스 플랜 수립이 가능해 진다.

향후 과제로는 본 연구에서 제안한 프로세스를 다수의 실사례에 적용하여 발견되는 문제점을 보완하고, 자동화시키는 방법이 필요하다. 또한 우선순위화 이후에 릴리스 플랜을 수립하는 방법까지 확장하여 연구를 진행하고자 한다.

참 고 문 헌

[1] CMP Media. 2006 State of Embedded Market Survey. April, 2006.

[2] Vahid, F. and Givargis, T., Embedded System Design: A Unified Hardware / Software Introduction, John Wiley, 2002.

[3] Philip Koopman, "Embedded System Design Issues(the Rest of the Story),"Proceedings of the 1996 International Conference on Computer Design(ICCD96), October, 1996.

[4] 황위용, 강동수, 송치양, 백두권, "임베디드 시스템을 위한 요구사항 우선순위 프로세스", 제31회 한국정보처리학회 춘계학술발표대회 논문집, 제16권 제1호, pp.444-447, 2009.4

[5] Wolf, Wayne, Computers as Components - Principles of Embedded Computing System Design, Morgan-Kaufmann, 2005.

[6] 성재석, 강동수, 송치양, 백두권, "릴리즈 플랜의 적응적 요구사항 우선순위," 정보처리학회논문지D, 제15-D권, 제6호, pp. 841-856, 2008. 12.

[7] Joachkim K., Claes W., Bjorn R., "An Evaluation of methods for prioritizing software requirements," Information and Software Technology, 39(14-15), pp.939-947, 1998.

[8] Firesmith, D. G., "Prioritizing Requirements," Journal of Object Technology (JOT), 3(8), Swiss Federal Institute of Technology(ETH), Zurich, Switzerland, pp.35-47, September/October, 2004.

[9] Aybüke A., Claes W., Engineering and Managing Software Requirements, Springer Berlin Heidelberg, August, 2005.

[10] Berander, P. and Andrews, A., "Requirements Prioritization," in Engineering and Managing Software Requirements, ed. Aurum, A., and Wohlin, C., Springer Verlag, Berlin, Germany, pp.69-94, 2005.

[11] Karlsson, J., and Ryan, K. "A Cost-Value Approach for Prioritizing Requirements," IEEE Software 14(5), pp.67-74, 1997.

[12] Karl E. Wiegers, Software Requirements, second edition, pp.247-258, Microsoft Press, 2003.

[13] Berander P, "Using students as subjects in requirements prioritization,"Proceedings of the 2004 International Symposium on Empirical Software Engineering(ISESE'04), IEEE Computer Society, Los Alamitos, pp.167-176, 2004.

[14] K. Beck, "Extreme Programming Explained: Embrace Change", Addison-Wesley, Reading, Mass., 2000.

[15] Carlshamre P, Sandahl K, Lindvall M, Regnell B, Natt och Dag J., "An industrial survey of requirements interdependencies in software Release Planning,"Proceedings of the 5th IEEE international symposium on requirements engineering, pp.84-91, 2001.

[16] Penny D. A., "An Estimation-Based Management Framework for Enhancive Maintenance in Commercial Software Products," In Proceedings of International Conference on Software Maintenance (ICSM), pp.122-130, 2002.

[17] Denne, M. and Cleland-Huang, J., "The Incremental Funding Method: Data Driven Software Development,"21(3), pp.39-47, 2004.

[18] Ruhe, G., Saliu, M. O., "The Science and Practice of Software Release Planning," IEEE Software, 2005.

[19] Greer, D., and Ruhe, G., "Software Release Planning: an Evolutionary and Iterative Approach,"Information and Software Technology, 46(4), pp.243-253, 2004.

[20] Saliu O, Ruhe G., "Supporting software Release Planning decisions for evolving systems," Proceedings of 29th IEEE/NASA software engineering workshop, Greenbelt, MD, USA, 6-7 April, 2005.

[21] Lee, Dong-hyun, In, Hoh Peter, Lee, Keun, Park, Sooyong, Hinchey, Mike,"A Survival Kit Adaptive Hardware Software Codesign Life-Cycle Model," IEEE COMPUTER SOC, Volume: 42, Issue: 2, Ppp.100-102, FEB., 2009.

[22] Kalavade, A. and Lee, E., "A Hardware-software codesign methodology for DSP applications", IEEE Design & Test of

Comp. 10, 3, pp.16-28, 1993.
[23] B. Regnell and S. Brinkkemper, "Market-Driven Requirements Engineering for Software Products,"Engineering and Managing Software Requirements, A. Aurum and C., Wohlin (eds.), Berlin, Germany, Springer Verlag, pp.287-308, 2005.
[24] Jacobson, I., Booch, G.,Rumbaugh, J., The Unified Software Development Process, pp.94-98, Addison Wesley, Object Technology Series, 1999.
[25] Karl E. Wiegers, "More about software requirements: Thorny Issues and Practical Advice", Microsoft Press, 2006.
[26] A. Rashid, "Website: Early Aspects: Aspect-Oriented Requirements Engineering and Architecture Design," URL: Early Aspects: Aspect-Oriented Requirements Engineering and Architecture Design, 2005.
[27] Y. Yu, J. C. S. P. Leite, J. Mylopoulos, "From Goals to Aspects: Discovering Aspects from Requirements Goal Models," Proceedings of Requirements Engineering Conference, Kyoto, Japan, pp.38-47, 2004.
[28] A. I. Antón, "Goal-Based Requirements Analysis," Proceedings of ICRE'96, pp. 136-144, 1996.
[29] A. Lamsweerde, "Goal-Oriented Requirements Engineering: A Guided Tour," 5th International Symposium on Requirements Engineering, IEEE Computer Society Press, pp.249-261, 2001.
[30] Dean Leffingwell, Don Widrig, Managing Software Requirements ; A Use Case Approach, Second Edition, pp.10 - 11, pp.95-100, pp.50, Addison Wesley, 2003.
[31] Lehtola, L., and Kauppinen, M. "Empirical Evaluation of Two Requirements Prioritization Methods in Product Development Projects," Proceedings of the European Software Process Improvement Conference (EuroSPI 2004),Trondheim, Norway, pp.161-170, 2004.
[32] Karlsson, L., Thelin, T., Regnell, B., Berander, P., and Wohlin, C., "Pair-Wise Comparisons versus Planning Game Partitioning - Experiments on Requirements Prioritisation Techniques," Journal of Empirical Software Engineering, Vol.1.11 Nr2, 2006.
[33] Ahl, Viggo. "An Experimental Comparison of Five Prioritization Techniques - Investigating Ease of Use, Accuracy, and Scalability,"Master Thesis No. MSE-2005-11, School of Engineering, Blekinge Institute of Technology, 2005.
[34] Taehee Gwak and Yoonjung JangAn, "Empirical Study on SW Metrics for Embedded System." LNCS 3966, pp.302 - 313, 2006.

황 위 용

e-mail : wyhwang@korea.ac.kr
2008년 상명대학교 소프트웨어학부(학사)
2008년~현 재 고려대학교 컴퓨터·전파통신공학과 석사과정
관심분야 : 소프트웨어공학, 요구공학, Embedded System

강 동 수

e-mail : greatkoko@hotmail.com
1997년 해군사관학교 전기공학(학사)
2006년 국방대학교 전산학과(이학석사)
2008년~현 재 고려대학교 컴퓨터·전파통신공학과 박사과정
관심분야 : 소프트웨어공학, SOA, System Engineering, 요구공학

송 치 양

e-mail : cysong@knu.ac.kr
1985년 한남대학교 전산학과(학사)
1987년 중앙대학교 전산학과(이학석사)
2003년 고려대학교 컴퓨터학과(이학박사)
1990년~2005년 한국통신 중앙연구소 책임연구원
2005년~2007년 상주대학교 소프트웨어공학과 조교수
2007년~현 재 경북대학교 소프트웨어공학과 조교수
관심분야 : UML 모델링 기술, 소프트웨어 개발방법, IP-TV 서비스

성 재 석

e-mail : damuljss@gmail.com

1995년 인하대학교 전자공학과(학사)

2008년 고려대학교 컴퓨터 · 전파통신공학과(공학석사)

1995년~1998년 SKC 근무

2001년~현 재 LG전자 MC연구소 책임연구원

관심분야 : 소프트웨어공학, 요구공학, Project Management, Six Sigma

백 두 권

e-mail : baikdk@korea.ac.kr

1974년 고려대학교 수학과(학사)

1977년 고려대학교 산업공학과(공학석사)

1983년 Wayne State Univ. 전산학과 (이학석사)

1985년 Wayne State Univ. 전산학과 (이학박사)

2002년 고려대학교 정보통신대학 학장

1986년~현 재 고려대학교 컴퓨터 · 전파통신공학과 교수

1989년~현 재 한국 정보처리학회 부회장

1992년~현 재 ISO/IEC JTC1/SC32 국내위원회 위원장

1999년~현 재 정보통신진흥협회 데이터 기술위원회 의장

2005년~현 재 한국 시뮬레이션학회 고문

2009년~현 재 고려대학교 정보통신대학 학장

관심분야 : 데이터베이스, 소프트웨어공학, 데이터공학, 컴포넌트 기반 시스템, 메타데이터 레지스트리, 정보 통합, 프로젝트 매니지먼트