JANT 한국항행학회논문지
Journal of Advanced Navigation Technology

항행 및 항법

J. Adv. Navig. Technol. 24(1): 28-36, Feb. 2020

KASS 통합운영국 소프트웨어 품질 보증을 위한 소프트웨어 품질 모델 및 메트릭 적용방안

Application of Software Quality Model and Metric for Software Product Assurance for KASS Control Station

김 연 실[1*] · 이 은 성[2]
[1]한국항공우주연구원 무인기연구부
[2]한국항공우주연구원 SBAS기술팀

Youn-sil Kim[1*] · Eun-sung Lee[2]

[1]Unmanned Aircraft System Research Division, Korea Aerospace Research Institute, Daejeon, 34133, Korea

[2]SBAS Technology Development Team, Korea Aerospace Research Institute, Daejeon, 34133, Korea

[요 약]

KASS(Korea Augmentation Satellite System)는 국토교통부에서 2014년부터 개발 중인 한국형 위성항법보강시스템이다. KASS는 항공기 안전에 영향을 미칠 수 있는 항공용 시스템이기 때문에 KASS의 각 소프트웨어는 안전성 분석을 통해 할당된 DO-178B의 소프트웨어 레벨에 따라 개발이 수행된다. KASS의 하위시스템인 통합운영국의 경우 일부 소프트웨어를 제외하고는 DO-178B 레벨 E를 할당 받았으며 DO-178B 레벨 E 소프트웨어의 경우 제품 보증을 위해 ECSS-Q-ST-80C 카테고리 D를 준수하여 개발하도록 하고 있다. 본 논문에서는 ECSS-Q-ST-80C를 만족하기 위해 ECSS-E-ST-40C, ECSS-Q-HB-80-04A를 분석하여 KASS 통합운영국 소프트웨어의 제품 보증을 위한 소프트웨어 생명 주기 별 활동 및 소프트웨어 품질 모델, 메트릭을 제안한다.

[Abstract]

Korea augmentation satellite system (KASS) is the Korean satellite based augmentation system (SBAS) developed by ministry of land, infrastructure, and transport (MOLIT) since 2014. Since KASS is the safety critical system that can affect to the safety of airplane, the software of KASS is developed according to the DO178B software level induced from safety analysis. In case of KASS control station (KCS), most of the software of KCS get assigned software level E in DO178B. In that case, ECSS-Q-ST-80C category D is assigned as a software product assurance standard. In this paper, the software related standard ECSS-E-ST-40C, ECSS-Q-HB-80-04A are analyzed to satisfy ECSS-Q-ST-80C and as a result the software product assurance activities regarding software life cycle and the software quality model, metric is proposed for the product assurance of the KCS software.

Key words : KASS, KCS, Software product assurance, Software quality model, Software metric.

https://doi.org/10.12673/jant.2020.24.1.28

Received 17 December 2019; **Revised** 5 February 2020
Accepted (Publication) 25 February 2020 (28 February 2020)

***Corresponding Author, Youn-sil Kim**

Tel: +82-42-870-3505
E-mail: younsil@kari.re.kr

Ⅰ. 서 론

SBAS (satellite based augmentation system)는 GNSS (global navigation satellite system)를 보강하는 시스템으로 ICAO (International Civilian Aviation Organization)에서 발행한 SARPs (standards and recommended practices) annex10에 항공기 운항 모드에 따른 SBAS 요구사항이 제시되어 있다[1].

KASS는 한국형 위성항법보강시스템으로 인천 FIR (flight information region) 및 제주도를 포함한 대한민국 내륙 지방에 대해 GPS (global positioning system) 신호를 보강하는 역할을 한다. 그림 1은 KASS의 아키텍처를 나타낸다[2].

KRS(KASS reference station)는 GPS 위성 신호를 수집하는 역할을 하며 이렇게 수집된 정보는 KPS(KASS processing station)로 전달된다. KPS는 수집된 GPS 신호를 이용하여 GPS 위성 궤도 및 시계 오차, 전리층 지연 오차에 대한 보정 정보 및 무결성 정보를 생성하는 역할을 한다. KPS에서 생성된 KASS 메시지는 KUS(KASS uplink station)로 전달되면 KUS에서 KASS의 GEO(geostationary earth orbit) 위성으로 KASS 메시지를 송신한다. GEO 위성에서 방송된 KASS 메시지를 항공기 등의 사용자가 수신하여 위치 계산 및 보호수준을 계산하는데 사용한다. KASS 메시지는 GPS 위성 궤도 및 시계, 전리층 지연에 대한 오차 보상을 위한 보정 정보와 이에 대한 무결성 정보 포함한다.

KASS는 국토교통부 주관으로 2014년 10월 개발이 착수되었으며 APV-1급 SoL(safety of life) 서비스 제공을 목표로 현재 개발 중에 있다. EGNOS 개발 경험이 있는 TASF(thales alenia space France)사가 시스템 레벨 개발자로 참여하고 있으며 일부 지상국은 국내에서 개발이 진행되고 있다.

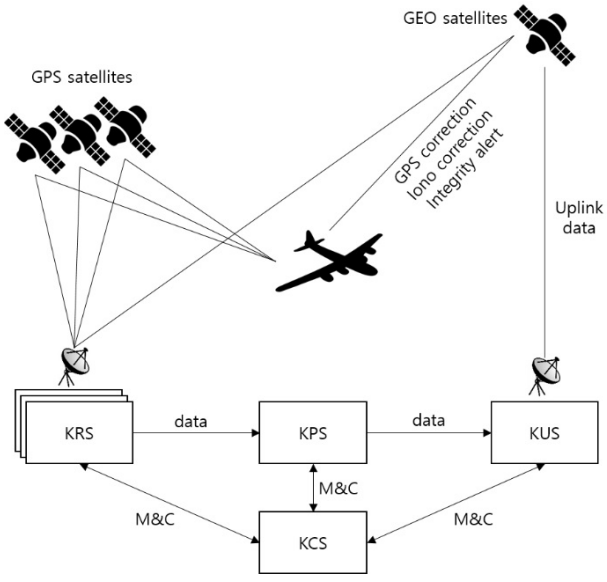

그림 1. KASS 아키텍처[2]
Fig. 1. KASS architecture[2].

KASS는 항공기 이착륙에 활용되는 항공용 시스템이기 때문에 개발의 신뢰성을 보장을 위해 여러 국제표준을 따라 개발된다[3]. 유럽의 TASF사가 시스템 레벨 개발을 담당하고 있기 때문에 이에 따라 사업 전반에 할당된 표준 또한 유럽우주국(ESA; european space agency)에서 발행한 ECSS (european cooperation for space standardization) 표준을 따르고 있다. ECSS 표준 구성은 그림 2와 같다[4].

소프트웨어 제품 보증 표준인 ECSS-Q-ST-80C는 ECSS 표준 중 소프트웨어 엔지니어링 표준인 ECSS-E-ST-40C와 짝을 이루어 내용이 구성되어 있다. ECSS-E-ST-40C는 소프트웨어 생명 주기에 대한 정의 및 생명 주기 별 엔지니어링 활동 내용을 담고 있으며 ECSS-Q-ST-80C에서는 이에 대한 제품 보증 요구사항을 다루고 있다.

KASS의 통합운영국의 경우 일부 소프트웨어를 제외하고는 DO-178B 레벨 E 및 ECSS-Q-ST-80C 카테고리 D를 할당받았으며 ECSS의 소프트웨어 카테고리는 ECSS-Q-ST-80C annex D Table D-1의 criticality에 따라 결정된다[5],[6].

본 논문에서는 ECSS-Q-ST-80C를 준수하기 위해 먼저 ECSS-E-ST-40C에서 제시한 소프트웨어 생명 주기를 소프트웨어 개발자 관점으로 분석하여 ECSS-Q-ST-80C의 적용을 용이하게 하였으며 더불어 ECSS-Q-ST-80C에서 요구하는 소프트웨어 메트릭을 적용하기 위해 ECSS-Q-HB-80-04A를 기반으로 소프트웨어 메트릭 및 계산방법, 목표 값을 제시하였다.

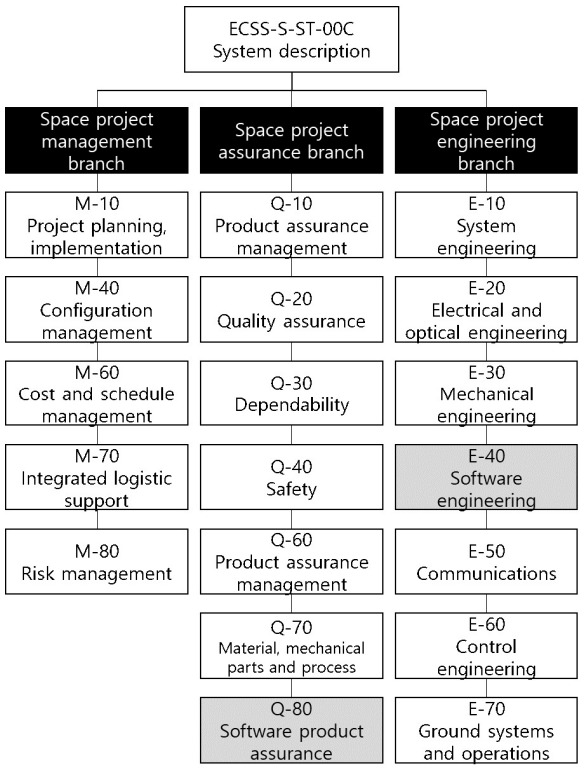

그림 2. ECSS 표준 구성[4]
Fig. 2. ECSS standard organization[4].

특히 소프트웨어 메트릭 적용 관련해서는 ISO/IEC 9126, 25000 시리즈 또는 ECSS-Q-HB-80-04A 등의 표준을 활용할 수 있다. ISO/IEC 9126 시리즈는 가장 널리 쓰이는 소프트웨어 품질 표준이며 ISO/IEC 25000 시리즈는 9126의 업데이트 버전이다[7]. 관련 연구로는 테스트 데이터 기반 품질 모델 분석 연구, ISO/IEC 25023 기반 소프트웨어 품질 평가 방안 연구 등 ISO 표준을 활용한 연구들이 주류를 이루고 있다[8], [9].

본 논문에서는 ECSS-Q-ST-80C 적용을 용이하게 하기 위해 ISO 표준 대신 ECSS 소프트웨어 생명 주기에 최적화된 ECSS-Q-HB-80-04A를 활용하여 메트릭을 제시하였다. 더불어 ECSS-Q-HB-80-04A에서 명확하게 표현되지 않은 항목의 경우 시스템 레벨 개발자의 기준을 바탕으로 구체화하였다.

II. 소프트웨어 표준 분석

그림 3은 ECSS 소프트웨어 관련 프로세스와 해당 프로세스를 담고 있는 표준을 표시하고 있다[10].

ECSS-E-ST-40C는 소프트웨어의 개발, 운영, 유지보수 프로세스에 대한 요구사항을 다루고 있는데 소프트웨어 생명 주기의 각 입출력 산출물을 검토하는 verification 활동 및 소프트웨어 테스트를 통해 소프트웨어 요구사항에 대한 만족여부를 검증하는 validation 활동에 대한 내용을 포함한다.

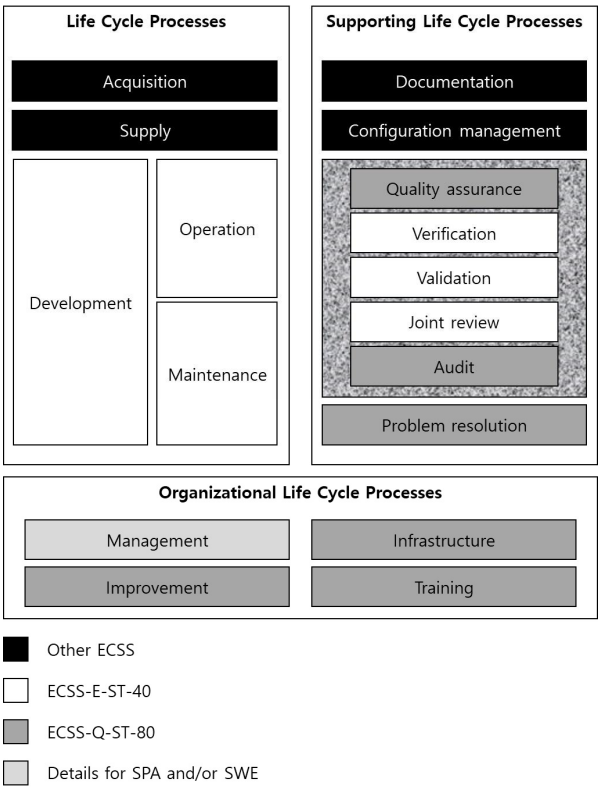

그림 3. ECSS 소프트웨어 관련 프로세스[10]
Fig. 3. ECSS SW(software) related processes[10].

ECSS-Q-ST-80C는 소프트웨어 제품 보증 관련 내용을 포함하고 있으며 소프트웨어 품질 보증, 감사, 문제 해결 등에 대한 내용을 제시하고 있으며 소프트웨어 프로세스 품질 평가 및 소프트웨어 품질 모델, 품질 요구사항, 메트릭에 대한 요구사항을 포함하고 있다.

소프트웨어 메트릭에 대한 내용은 ECSS 소프트웨어 관련 핸드북인 ECSS-Q-HB-80-04A에서 가이드라인을 제시하고 있으며 기본적으로 ECSS에서 사용하는 소프트웨어 품질 관련 내용은 ISO/IEC 9126 및 ISO/IEC 24765의 내용을 참조하고 있다. 본 논문의 3장에서 제시한 소프트웨어 품질 모델 및 메트릭 또한 ECSS-Q-HB-80-04A를 기반으로 하였다.

소프트웨어의 형상 관리에 대한 것은 ECSS-M-ST-40C에서 다루고 있다.

2-1 소프트웨어 엔지니어링 표준 분석

그림 4는 ECSS-E-ST-40C에서 제시하는 소프트웨어 엔지니어링 프로세스 별 활동을 좀 더 이해하기 쉽도록 표현한 그림이다[10]. 그림 4의 하단에는 각 활동을 수행하는 주체를 표시하였다. 활동의 주체는 시스템 개발자 및 소프트웨어 개발자의 역할로 나뉜다. 본 논문에서는 소프트웨어 개발자 중심으로 소프트웨어 수락까지 총 8개의 프로세스를 정의하였다.

(1) 소프트웨어 관련 시스템 요구사항 프로세스는 그림 4에서와 같이 시스템 개발자가 주로 수행하며 시스템 요구사항 중 소프트웨어에 할당되는 요구사항을 도출한다. 이를 requirement baseline(RB)로 정의하며 그림 4의 SRR (system requirement review)의 출력이 된다.

(2) 소프트웨어 요구사항 프로세스에서는 RB를 기반으로 소프트웨어 요구사항을 구체화하는 활동을 수행한다. 소프트웨어 요구사항 프로세스부터 QR(qualification review)까지는 소프트웨어 개발자가 주도적으로 수행하는 활동이다. 소프트웨어 요구사항 프로세스의 출력은 TS (technical specification) 및 인터페이스 요구사항이며 그림 4의 마일스톤 상으로는 SWRR(software requirement review)이 된다.

(3) 소프트웨어 아키텍처 프로세스에서는 소프트웨어 아키텍처 및 아키텍처 간 인터페이스 설계가 진행되고 마일스톤 상으로 PDR (preliminary design review)이 된다. 또한 PDR에서는 verification 및 validation에 대한 계획이 완료되며 단위 소프트웨어의 통합 계획이 작성된다. ECSS-E-ST-40C에서 소프트웨어 verification은 모든 소프트웨어 개발 프로세스의 입력이 적절하며 출력이 정확하고 일관성 있는지 등을 검토하는 활동을 의미하며 소프트웨어 개발 활동과 병렬적으로 수행된다. 소프트웨어 validation은 소프트웨어가 TS 및 RB를 만족함을 증명하기 위한 소프트웨어 테스트 활동을 말한다.

(4) 소프트웨어 설계 프로세스에서는 소프트웨어 상세 설계가 진행되며 모든 소프트웨어 설계 활동이 완료된다. 소프트웨어가 알고리즘을 포함하고 있을 경우 이 시기에 알고리즘 설계서가 도출된다.

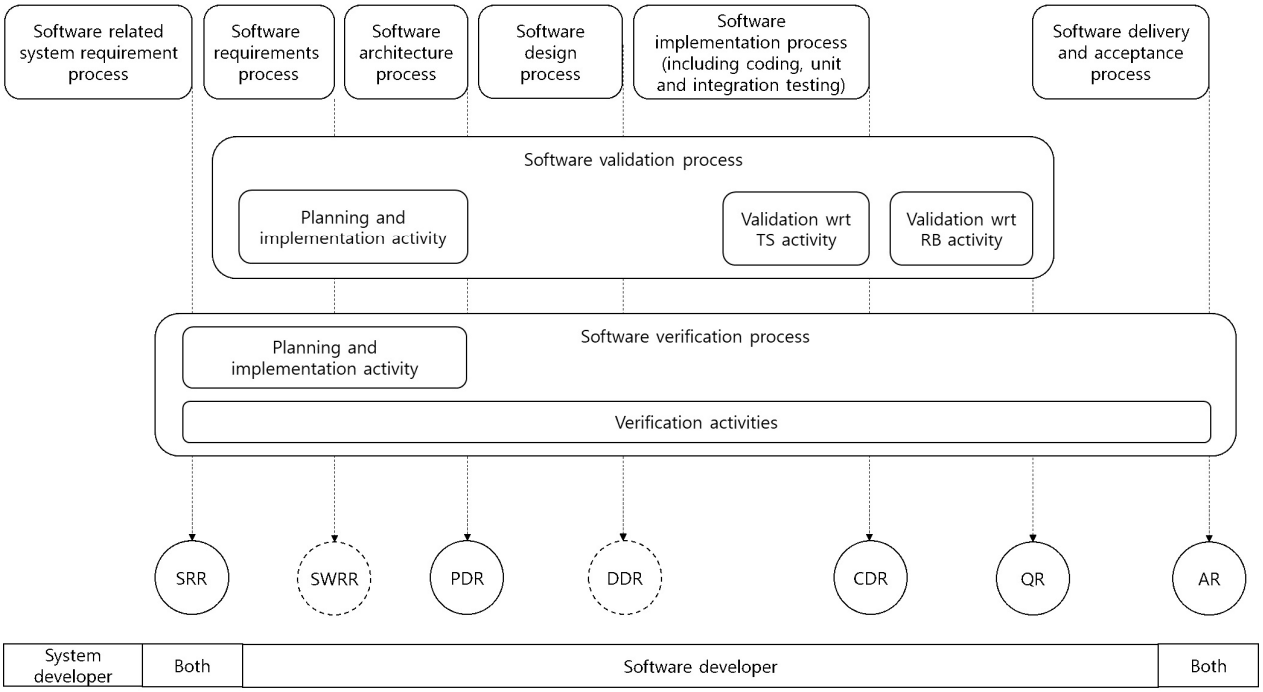

그림 4. ECSS 소프트웨어 생명 주기 프로세스[10]
Fig. 4. ECSS software life cycle process[10].

ECSS 소프트웨어 마일스톤 상으로는 DDR(detailed design review)이 되며 이때 소프트웨어 단위 테스트에 대한 계획이 함께 산출된다.

(5) 소프트웨어 구현 프로세스에서는 소프트웨어 코딩 및 단위 테스트, 통합 테스트가 진행되고 테스트 결과가 도출된다. 또한 이 시기에 TS에 대한 validation 활동이 함께 이루어진다. 소프트웨어 구현 프로세스 및 TS에 대한 validation이 완료된 후에 CDR(critical design review)이 진행되는데, 일반적인 시스템 엔지니어링에서 CDR이 모든 설계 완료 후 시스템 구현 전에 수행되는 것과는 달리 ECSS-E-ST-40C에서는 이 시기가 DDR에 해당되고 CDR은 소프트웨어 단위, 통합 테스트, TS에 대한 validation 테스트가 수행된 후에 진행된다는 점에서 차이가 있다. ECSS-E-ST-40C를 기준으로 하면 소프트웨어 DDR이 진행된 후에 시스템 CDR이 진행된다.

(6) 소프트웨어 validation 프로세스에서는 소프트웨어 제품이 TS 및 RB를 만족함을 검증하는 테스트가 수행되며 DDR과 CDR 사이에 TS에 대한 validation이 수행되고 CDR과 QR사이에 RB에 대한 validation이 수행된다. RB에 대한 validation이 결국 시스템 개발자가 할당한 소프트웨어 요구사항에 대한 만족 여부를 검증하는 것을 의미하므로 이 활동을 수행한 후에 소프트웨어 개발자는 개발된 소프트웨어 제품에 대한 QR을 진행할 수 있게 된다.

(7) 소프트웨어 수락 프로세스에서 QR이 완료된 제품이 시스템 개발자에 전달되어 시스템 개발자가 소프트웨어 운영 환경에서 수락 테스트를 진행하여 소프트웨어 제품이 모든 요구사항을 만족할 경우 AR (acceptance review)를 진행하여 제품 수락을 완료한다.

(8) 소프트웨어 verification 프로세스에서는 위 나열한 모든 개발 프로세스의 입, 출력을 검토하고 분석하는 활동을 수행한다. 요구사항의 추적성, 설계 적합성, 코드 커버리지, 단위 및 통합 테스트 계획/결과의 적합성, validation 결과의 적합성 등을 확인하며 verification 활동의 결과를 verification report에 작성한다. 소프트웨어 validation 및 verification 프로세스는 요구되는 독립성에 따라 같은 조직 내 다른 인력 혹은 아예 다른 조직에서 수행되기도 한다.

ECSS-Q-ST-80C가 할당된 통합운영국 소프트웨어는 위 서술한 개발 프로세스를 따라 소프트웨어 개발이 될 예정이며 다음 2-2장에서 각 개발 프로세스에 대한 제품 보증 요구사항에 대한 내용을 기술한다.

2-2 소프트웨어 제품 보증 표준 분석

ECSS-Q-ST-80C에서 제시하고 있는 소프트웨어 품질 보증에 대한 요구사항 중 통합운영국 소프트웨어와 관련이 높은 항목을 그림 4에 표현된 단계를 기반으로 표 1, 2에 요약하였다. ECSS-Q-ST-80C는 크게 소프트웨어 제품 보증 프로그램에 대한 요구사항, 소프트웨어 프로세스 보증에 대한 요구사항, 소프트웨어 제품 품질 보증에 대한 요구사항을 다루고 있다.

소프트웨어 제품 보증 프로그램 구현에 대한 내용으로는 조직 및 책임, 제품 보증 프로그램 관리, 보고, 감사, 소프트웨어 문제, 부적합 사항 처리, 품질 모델, 위험 관리, 하위 공급자 선정 기준, 조달, 도구 및 환경에 대한 내용을 담고 있으며 구

체적인 내용은 표 1에 요약, 정리하였다.

소프트웨어 제품 보증 프로그램은 그림 4의 모든 프로세스에 관여하며 프로젝트에 적용된 표준 및 승인된 계획에 대한 준수여부를 검토하기 위하여 소프트웨어 개발 활동을 독립적으로 감시하는 역할을 한다. 소프트웨어 제품 보증 프로그램을 통해 프로젝트에 할당된 표준 및 승인된 계획에 맞게 소프트웨어 생명 주기 활동이 수행되는지 감시하며, 프로젝트 수행 도중 발견된 모든 미흡/부적합한 사항들이 적절하게 처리되고 해결되도록 지원하고 관리한다. 소프트웨어 제품 보증 프로그램 구현상의 가장 큰 특징은 소프트웨어 개발 조직과의 독립성이다. 이를 통해 소프트웨어 제품 보증 조직은 소프트웨어 개발 조직에 종속되지 않은 상태에서 적절한 권한을 가지고 소프트웨어 개발 전 과정을 감독할 수 있다.

표 2에서는 ECSS-Q-ST-80C에 제시된 소프트웨어 프로세스 보증, 소프트웨어 제품 품질 보증에 대한 요구사항을 요약, 정리하였다. 소프트웨어 프로세스 보증 요구사항은 다시 소프트웨어 개발 프로세스 전반에 적용되는 요구사항과 각 개발 프로세스 별로 적용되는 요구사항으로 나뉜다.

표 1. 소프트웨어 제품 보증 프로그램 요구사항
Table 1. SW product assurance program requirements.

Contents
● Organization and responsibility ▷The supplier shall identify the software PA(product assurance) manager who has appropriate authority and independence. ▷The supplier shall provide proper training to the personnel for the tasks in the project.
● SW PA program management ▷The supplier shall develop a software product assurance plan.
● SW PA reporting ▷The supplier shall report on a regular basis on the status of the SW PA programme and the SW PA report shall include an assessment of the current quality of the product and processes based on measured metric, verifications undertaken, problems detected/resolved.
● Audits ▷For SW audits, ECSS-Q-ST-10 clause 5.2.3 shall apply.
● Software problems ▷The supplier shall define and implement procedures for the logging, analysis and correction of all software problems encountered during software development.
● Nonconformances ▷For software nonconformance handling, ECSS-Q-ST-10-09 shall apply.
● Quality models ▷Quality model shall be used to specify the software quality requirements. (ISO/IEC 9126 or ECSS-Q-HB-80-04)
● Critical item control ▷For critical item control, ECSS-Q-ST-10-04 shall apply.
● Supplier selection and control ▷For supplier selection, ECSS-Q-ST-20 clause 5.4.1 shall apply.
● Procurement ▷For procurement documents, ECSS-Q-ST-20 clause 5.4.2 shall apply. ▷All the procured software shall be identified and registered by configuration management.
● Tools and supporting environment ▷The correct use of methods and tools shall be verified and reported. ▷The suitability of the software development environment shall be justified.

표 2. 소프트웨어 프로세스 및 제품 요구사항
Table 2. SW process, product assurance requirements.

Contents
Process assurance
● Software development life cycle ▷The software life cycle shall be reviewed against the contractual software engineering and product assurance requirements.
● Requirements appliable to all software engineering processes ▷All plans shall be finalized before the start of the related activities. ▷All plans shall be updated for each milestone to reflect any changes during development. ▷ECSS-M-ST40 shall be applied for software configuration management. ▷Metrics shall be used to manage the development and to assess the quality of the development processes. ▷A summary of the assurance activities concerning the verification process and their findings shall be included in SW PA reports. ▷The completion of actions related to software problem reports generated during verification shall be verified and recorded. ▷Analysis of the advantages to be obtained with the selection of existing software instead of new development shall be carried out. ▷All the reused software shall be kept under configuration control. ▷The requirements on testing applicable to the automatically generated code shall ensure the achievement of the same objectives as those for manually generated code.
● Requirements applicable to individual software engineering processes ▷For the definition of the software related system requirements to be specified in the RB, ECSS-E-ST-40 clause 5.2 shall apply. ▷In addition to the functional requirements, the technical specification shall include all non-functional requirements necessary to satisfy the requirements baseline, including, as minimum, the followings: performance, safety, reliability, robustness, quality, maintainability, configuration management, security, privacy, etc. ▷Adherence to design standards shall be verified. ▷Coding standards shall be specified and observed. ▷Test coverage shall be checked with respect to the stated goals. ▷The supplier shall ensure that nonconformances and software problem reports detected during testing are properly documented and reported to those concerned. ▷Areas affected by any modification shall be identified and re-tested. ▷The customer shall ensure that the acceptance tests are performed in accordance with the approved acceptance test plan. ▷Before the software is presented for customer acceptance, the supplier shall ensure that: the delivered software complies with the contractual requirements, all agreed changes are implemented, all nonconformances are either resolved or declared. ▷During the demonstration that the software conforms to the operational requirements, the following shall be covered as a minimum: availability and maintainability of the host system, human-computer interface, operating procedures, etc. ▷The organization responsible for maintenance shall be identified to allow a smooth transition into the operations and maintenance.
Product quality assurance
● Product quality objectives and metrication ▷Quality requirements shall be expressed in quantitative terms. ▷The supplier shall define assurance activities to ensure that the product meets the quality requirements as specified in the technical specification. ▷In order to verify the implementation of the product quality requirements, the supplier shall define a metrication programme based on the identified quality model. ▷The results of metrics collection and analysis shall be reported to the customer. ▷The software quality requirements shall be documented in the requirements baseline and technical specification. ▷Numerical accuracy shall be estimated and verified.

자세한 내용은 표 2에 제시하였다. 프로세스 보증 요구사항을 보면 '프로세스의 품질을 조사하고 관리하기 위해 프로세스 메트릭이 사용되어야 한다'는 항목이 있는 것을 확인할 수 있다. 또한 소프트웨어 제품 품질 보증 요구사항 파트에서도 제품 품질 요구사항의 증명, 보고를 위해 제품 메트릭을 사용하고 메트릭 프로그램을 정의할 것을 요구하고 있다. 또한 표 1과 표 2의 내용을 비교분석 해보면 소프트웨어 품질 모델을 적용하고 이를 기반으로 소프트웨어 품질 요구사항을 구체화하는 것을 알 수 있다.

이와 같이 소프트웨어 제품 보증을 위해서는 프로세스 및 제품 메트릭이 사용된다. 본 논문에서는 ECSS-Q-ST-80C에서 요구하는 소프트웨어 제품 보증을 위해 ECSS-Q-HB-80-04A를 기반으로 소프트웨어 메트릭을 3장에 제안하였다.

Ⅲ. 소프트웨어 품질 모델 및 메트릭 제안

3-1 소프트웨어 품질 모델

그림 5는 품질 모델을 구성하는 소프트웨어 품질 요구사항, 품질 특성, 메트릭 등의 관계를 나타낸다. 표 1에 요약한 바와 같이 ECSS-Q-ST-80C에서는 소프트웨어 품질 요구사항을 구체적으로 명시하기 위해 품질 모델을 사용할 것을 요구하고 있다. 그림 5는 이와 같은 품질 모델을 정의하기 위한 구성 요소 및 그 관계를 나타낸다[11].

품질 모델은 주 특성 및 부 특성으로 구성되는데 일반적으로 계층적 트리 구조로 표현될 수 있다. 그림 5에서와 같이 주 특성은 프로세스 관련 특성과 제품 관련 특성이 있으며 주 특성은 다시 여러 개의 부 특성으로 세분화된다. ISO 9126, ISO 25000, ECSS-Q-HB-80-04A 등의 표준에 여러 기본 품질 모델이 정의되어 있다[7], [8], [9], [11], [12]. 소프트웨어 개발자는 이를 적절하게 변경하여 해당 소프트웨어에 사용할 품질 모델을 정의할 수 있다. 소프트웨어의 품질 특성을 정량적으로 측정하기 위한 방법으로 메트릭이 사용되는데 이는 3-2장에서 더욱 자세히 설명하기로 한다.

본 논문에서는 ECSS-Q-HB-80-04A에 제시된 기본 품질 모델을 기반으로 통합운영국 소프트웨어 criticality인 카테고리 D에 적용되는 품질 모델을 표 3과 같이 제안하였다. 제안된 품질 모델은 총 7개의 특성, 12개의 부 특성으로 구성되어 있으며 각 부 특성은 다시 이를 정량적으로 측정할 수 있는 메트릭과 연결되어 있다. 소프트웨어 개발자는 이를 활용하여 소프트웨어의 프로세스 및 제품 품질을 측정하고 평가할 수 있다.

3-2 소프트웨어 메트릭

소프트웨어 메트릭은 그림 5에서 나타낸 바와 같이 품질 모델에서 정의된 부 특성들을 정량적으로 측정하기 위해서 사용된다.

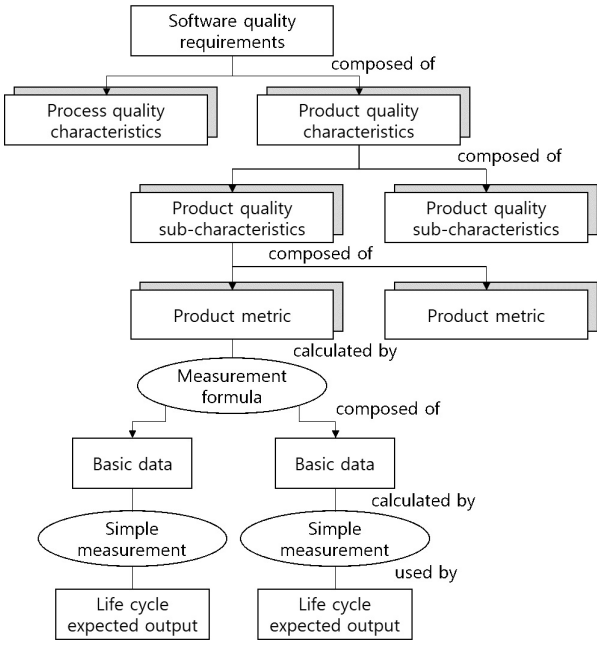

그림 5. 소프트웨어 품질 모델의 요소[11]
Fig. 5. Elements of software quality model[11].

표 3. 제안된 품질 모델
Table 3. Proposed quality model.

Characteristics	Sub-characteristics	Metrics
Functionality	Completeness	Requirement allocation
		Requirement implementation coverage
		V&V(validation & verification) coverage
	Correctness	SPR(software problem report)/NCR(nonconformance report) trend analysis
		Documentation suitability
		Adherence to coding standards
Reliability	Reliability evidence	SPR/NCR status
		Structural coverage
	Correctness	= Functionality correctness
Maintainability	Complexity	Cyclomatic complexity
		Nesting level
		Lines of code
		Comment frequency
	Correctness	= Functionality correctness
	User doc. quality	User manual suitability
Reusability	Reusability documentation	Reusability checklist
Security	Security evidence	Security checklist
Usability	User doc. quality	= Maintainability User doc. quality
	User interface quality	Adherence to MMI(man machine interface) standards
Software Development Effectiveness	Project management effectiveness	Code size stability
		RID(review item discrepancy)/action status
		V&V progress

표 4. 제안된 소프트웨어 메트릭
Table 4. Proposed software metric.

#	Metric name	Formula	Evaluation method	Target value
1	Requirement allocation	X=A/B A=number of system level requirements for software that have one or more trace to SW requirements or SW design components B=number of system level requirements for software	Check traceability matrix.	1 (0<=X<=1, the closer to 1 the better)
2	Requirement implementation coverage	X=A/B A=number of correctly implemented requirements confirmed by verification (including test, inspection, review, and analysis, validation) B=number of requirements	Analyze traceability matrix.	1 (0<=X<=1, the closer to 1 the better)
3	V&V coverage (RB/TS levels)	X=A/B A=number of elements covered by at least one V&V activity (test, inspection, review or analysis) B=total number of element	Analyze traceability matrix	1 (0<=X<=1, the closer to 1 the better)
4	SPR/NCR trend analysis	There is no formula associated to this metric. The graphic exhibit convergence between number of raised and fixed problems. The lower the gap between these numbers the better.	Analyze SPR/NCR database	# of raised problem - # of fixed problem =0
5	Documentation suitability	X=A/B A=sum of Yes or N/A answers to the questions in the checklist B=total number of questions	Documentation suitability checklist	1
6	Adherence to coding standards	X=A/B A=sum of Yes or N/A answers to the questions in the checklist B=total number of questions	Coding standard checklist, to be filled in with the help of static analysis tools	1
7	SPR/NCR status	There is no formula associated to this metric.	Analysis of SPR/NCR database	# of major SPR/NCR remain open at AR=0
8	Structural coverage	X=A/B A=number of executed statements/decisions/conditions B=total number of statements/decisions/conditions	Dynamic analysis of the code with the support of automatic tools.	1 (Statement coverage)
9	Cyclomatic complexity (VG)	Cyclomatic complexity of a single routine (function or procedure) : VG=number of edges − number of nodes + 2 Cyclomatic complexity of a module : X=average VG for all routines in the module	Static code analysis with the support of automatic tools	20
10	Nesting level	Nesting level of a single routine: NL=maximum number of nested statements (simple or multiple choice decisions, loops) in the routine Nesting level of a module: X=maximum NL for all routines in the module	Static code analysis with the support of automatic tools.	7
11	Lines of code	LOC=total number of lines of code − comment and blank lines	Static code analysis with the support of automatic tools.	75
12	Comment frequency	X=A/B A=number of comment lines (excluding headers) B=LOC+number of lines of comments=total number of lines excluding blank lines	Static code analysis with the support of automatic tools.	0.2 (0<=X<=0.3)
13	User manual suitability	X=A/B A=sum of Yes or N/A answers to the questions in the checklist B=total number of questions	User manual checklist	1
14	Reusability checklist	X=A/B A=sum of Yes or N/A answers to the questions in the checklist B=total number of questions	Reusability checklist	1
15	Security checklist	X=A/B A=sum of Yes or N/A answers to the questions in the checklist B=total number of questions	Security checklist	1
16	Adherence to MMI standards	X=A/B A=sum of Yes or N/A answers to the questions in the checklist B=total number of questions	MMI standards checklist, to be filled in with the help of static analysis tools	1
17	Code size stability	X=A-B A=estimated code size B=actual code size	Static code analysis with the support of automatic tools	0
18	RID/action status	There is no formula associated to this metric.	Analysis of RID/action database	# of major RID/action from previous milestone remain open at next milestone=0
19	V&V progress	X=A/B A=number of successfully completed V&V activities B=number of planned V&V activities at this time	Analysis of SW V&V progress data or reports	1

표 3과 같이 품질 모델에서 부 특성은 1개 이상의 메트릭과 연결되어 있으며 하나의 메트릭이 여러 부 특성에 사용되기도 한다. ISO 9126에서는 메트릭 특성에 따라 내부 메트릭, 외부 메트릭, quality in use 메트릭으로 메트릭을 구분하고 있으나 ECSS-Q-HB-80-04A에서는 이를 따로 구분하지 않고 메트릭으로 통칭하여 사용한다.

그림 5에서와 같이 메트릭은 측정 식으로부터 계산되며 측정 식을 구성하는 베이직 데이터는 소프트웨어 생명 주기의 활동의 결과를 간단히 측정함으로써 계산된다. 본 논문에서는 표 3에서 정의된 총 19개의 메트릭에 사용되는 측정 식, 베이직 데이터, 평가 방법, 목표 값을 표 4에 제시하였다. 표 4의 측정 식 열에서 A, B로 표현된 항목이 그림 5에서의 베이직 데이터에 해당하며 관련 생명 주기 결과는 평가 방법 열에 제시되어 있다. 또한 본 논문에서는 각 메트릭의 목표 값을 제안하였는데 ECSS-Q-HB-80-04A 및 시스템 개발자인 TASF의 기준을 기반으로 목표 값을 구체화하였다.

Ⅴ. 결 론

본 논문에서는 ECSS-Q-ST-80C에서 요구하는 제품 보증 요구사항을 만족하기 위해 ECSS-E-ST-40C를 바탕으로 ECSS 소프트웨어 생명 주기에 적용되는 제품 보증 요구사항을 분석하였고 또한 ECSS-Q-HB-80-04A를 분석하여 소프트웨어 개발에 적용되는 소프트웨어 프로세스 및 제품 메트릭을 제안하였다.

먼저 ECSS-Q-ST-80C 제품 보증 요구사항의 적절한 반영을 위해서 ECSS 소프트웨어 생명 주기를 정의하고 있는 ECSS-E-ST-40C 표준을 분석하였고 이에 따라 소프트웨어 생명 주기 별로 적용되는 제품 보증 요구사항을 표 2에 정리하였다. 다음으로 본 논문에서는 ECSS-Q-ST-80C에서 요구하는 소프트웨어 메트릭 프로그램을 정의하기 위해 기존 소프트웨어 품질 관련 표준 중 ECSS 소프트웨어 생명 주기에 최적화된 ECSS-Q-HB-80-04A를 기반으로 소프트웨어 품질 모델 및 프로세스, 제품 메트릭을 제안하였다. 소프트웨어 메트릭 정의, 계산 식, 평가 방법, 목표 값을 표 4에 제시하였으며 이를 통해 KASS 통합운영국 소프트웨어 개발자는 해당 소프트웨어의 품질을 정량적으로 평가할 수 있다. 더불어 ECSS-Q-HB-80-04A에 구체적으로 명시되어 있지 않은 항목은 시스템 레벨 개발자인 탈레스의 프로젝트 수행 기준을 바탕으로 제시하였다.

결과적으로 본 논문에서 제안한 소프트웨어 품질 모델 및 메트릭을 기반으로 소프트웨어 품질을 정량적으로 측정, 평가, 보고함으로써 할당된 소프트웨어 제품 보증 표준을 만족하는 통합운영국 소프트웨어를 개발할 수 있을 것으로 예상한다. 더불어 본 논문의 분석 결과는 향후 ECSS 표준을 기반으로 소프트웨어 제품 보증을 수행하는 관련 프로젝트에 참고자료로 활용될 수 있을 것으로 기대한다.

Acknowledgments

연구는 국토교통부 항공안전기술개발사업의 연구비 지원(19ATRP-A087579-06)에 의해 수행되었습니다.

References

[1] International Standards and Recommended Practices, Annex 10 to the Convention on International Civil Aviation, Volume 1, Radio Navigation Aids, ICAO, 2006.

[2] Y. S. Kim, D. H. Won, and E. S. Lee, “Requirement analysis for KASS control station,” in *Proceedings of Annual Conference of the Korean Navigation Institute 2017*, Seoul: Korea, pp. 179-182, 2017.

[3] Y. S. Kim, and E. S. Lee, “Study for product assurance standard for KASS control station/wide area network service,” in *Proceedings of Annual Autumn Conference of the Society for Aerospace System Engineering Institute 2018*, Gyeongju: Korea, Nov 2018.

[4] European Cooperation for Space Standardization. ECSS Document Tree (ECSS Architecture) [Internet]. Available: https://ecss.nl/standards/ecss-document-tree-and-status/

[5] D. H. Bae, “Allocation of design assurance level for KASS based on international standards,” *Journal of Advanced Navigation Technology*, Vol. 20, No. 1, pp. 1-7, Feb 2016.

[6] Space Product Assurance: Software Product Assurance, ESA-ESTEC Requirements & Standards Division, ECSS-Q-ST-80C, 2009.

[7] H. J. Jung, “The Trend of Software Quality Testing”, *Korea Information Processing Society Review,* Vol. 21 No. 4, pp. 24 - 29, 2014.

[8] H. J. Jung, “The Quality Analysis Model for Software Testing”, *The Journal of Digital Policy & Management,* Vol. 11, No. 3, pp. 293 - 298, 2013.

[9] H. J. Jung, “The Software Quality Testing on the basis of the International Standard ISO/IEC 25023”, *Journal of the Korea Convergence Society*, Vol. 7, No. 6, pp. 35-41, 2016.

[10] Space Engineering: Software, ESA-ESTEC Requirements & Standards Division, ECSS-E-ST-40C, 2009.

[11] Space Product Assurance: Software Metrication Programme Definition and Implementation, ESA-ESTEC Requirements & Standards Division, ECSS-HB-80-04A, 2011.

[12] H. J. Jung, “Standardization trends of software quality evaluation,” *TTA Journal*, No. 128, pp. 101-105, 2010.

김 연 실 (Youn-Sil Kim)
2009년 2월 : 세종대학교 항공우주공학 (공학사)
2011년 2월 : 서울대학교 항공우주공학 (공학석사)
2016년 2월 : 서울대학교 항공우주공학 (공학박사)
2016년 ~ 현재 : 한국항공우주연구원
※관심분야 : GNSS/INS 통합항법, 반송파 기반 정밀 항법, SBAS

이 은 성 (Eun-Sung Lee)
1996년 2월 : 건국대학교 항공우주공학 (공학사)
1998년 2월 : 건국대학교 기계공학 (공학석사)
2005년 2월 : 건국대학교 항공우주공학 (공학박사)
2007년 ~ 현재 : 한국항공우주연구원 선임연구원
※관심분야 : 위성항법 정밀위치결정, 위성항법 시스템 고장검출, 위성항법 보강항법시스템