CALS/EC 개발 프로세스 표준화 발전방향

CALS/EC Development Process Standardization Progress Direction

최헌준 . 이윤희
국방정보체계연구소

<요 약 문>

본고에서는 CALS/EC 개발전략의 일환으로 소프트웨어 프로세스 수명주기 중 국제표준인 ISO/IEC 12207과 현재 미 국가 표준으로 채택된 IEEE/EIA 12207에 관하여 고찰해보고, ISO/IEC 12207의 모체라 할 수 있는 미국방성의 소프트웨어 개발 프로세스인 MIL-STD-498과 비교분석 하였다. 또한 현재 국내 국방분야에서 적용하고 있는 소프트웨어 수명주기 프로세스에 조명해보았다.

MIL-STD-498을 근간으로한 국방분야의 소프트웨어 수명주기 프로세스는 그 내용이 아직도 반복개발의 개념이나 프로젝트 특성에 따른 프로세스의 관리 등 새로운 개발 철학들을 수용하고 있지 못한 실정이다. 본 연구에서는 현 국방분야의 소프트웨어 수명주기 프로세스의 개선안으로 프로세스를 재정립한 후, 점진적 개발접근방법, 진화적 개발접근 방법의 수용과 프로세스의 특성에 따라 프로세스들을 선택.적용할 수 있는 MIL-STD-498을 기반으로 한 테일러링 방법을 제시하였다.

For the purpose of developing CALS/EC, it was surveyed ISO/IEC 12207 that was international standard and IEEE/EIA 12207 that was national standard in U.S.A and then performed analysis comparing with MIL-STD-498 it was called origin of DoD software development process. Also it was surveyed software life cycle process using doemstic defense area now.

Software life cycle process based on MIL-STD-498 in defense area was not yet includ the concept of new development philosophy like iterative development and process management according to the characteristics of project. For the purpose of improving software life cycle in defense area, first refined the proess. And then on this study it was recommended accomodation of incremental and evolutionary development approach and the method of tailoring based on MIL-STD-498 was able to select and apply the process on the characteristics of project.

I. 개 요

기존의 단순히 프로그램 개발만 하던 시대를 지나 소프트웨어 개발규모가 대형화되면서, 소프트웨어의 체계적인 개발을 위해 구조적방법론, 정보공학방법론, 객체지향 방법론 등과 같은 소프트웨어 개발방법론이 등장하여 널리 사용되어 왔다. 이에 더불어 근래에 와서는 소프트웨어 개발 범위 및 규모가 더욱 방대해지면서 소프트웨어 개발 활동 자체를 프로세스로 인식하여 관리하는 소프트웨어 수명주기 프로세스의 개념이 상당히 중요한 개념으로 부각되고 있으며, 소프트웨어 개발 프로젝트의 관리 수단으로 매우 중요한 기술적 관리 구조를 제공하고 있다.

본고에서는 CALS/EC 개발전략의 일환으로 소프트웨어 프로세스 수명주기 중 국제 표준인 ISO/IEC 12207과 현재 미 국가 표준으로 채택된 IEEE/EIA 12207에 관하여 고찰해보고, ISO/IEC 12207의 모체라 할 수 있는 미국방성의 소프트웨어 개발 프로세스인 MIL-STD-498과 비교분석 하였다.

미 국방성은 MIL-STD-498을 제정할 당시부터 국방표준을 국제표준과 국가표준으로 전환하겠다는 장기 계획하에 소프트웨어 개발 프로세스인 MIL-STD-498을 개발하였다. MIL-STD-498은 반복개발 접근방법, 자동화 도구의 사용 등 기존의 개발절차와는 달리 정보기술의 변화에 따른 새로운 개념들을 포함하고 있다.

MIL-STD-498은 소프트웨어 프로세스와 22개의 데이터 항목기술서(DIDs)로 구성되어 있다. 이 내용들의 거의 대부분이 IEEE/EIA 12207에 흡수되었으며, 그 구체적인 내용은 MIL-STD-498의 데이터 항목 기술서가 IEEE/EIA 12207의 수명주기 데이터 부분인 IEEE/EIA 12207.1의 모체를 이루고 있고, 소프트웨어 프로세스가 프로세스 구현 지침 부분인 IEEE/EIA 12207.2에 흡수되었다. 또한 IEEE/EIA 12207은 국제 표준인 ISO/IEC 12207의 프레임하에서 구현되었다.

CALS/EC 정보시스템을 개발하기 위한 전략으로서 소프트웨어 수명주기 프로세스인 IEEE/EIA 12207은 이런 의미에서 매우 중요한 의미를 갖는다. 현재 한국 국방분야에서도 이미 MIL-STD-498에 입각한 국방분야의 소프트웨어 수명주기 프로세스를 정립하여 국방 소프트웨어 개발 분야에 적용 중에 있다. 그러나 그 내용이 아직도 반복개발의 개념이나 자동화 도구의 사용 등 MIL-STD-498의 새로운 개발 철학들을 수용하고 있지 못한 실정이다. 본 연구에서는 현 국방분야의 소프트웨어 수명주기 프로세스를 조명해보고 MIL-STD-498을 근간으로 국방분야의 소프트웨어 수명주기 프로세스의 발전방향에 대한 개선방안을 제시하였다.

국내 정보산업계에서도 CALS/EC 정보시스템을 구현하기 위해서 이러한 국제적인 표준 동향이나 선진동향을 면밀히 고찰하고 국제표준과 호환되는 IEEE/EIA 12207와 같은 국가 표준의 제정이 시급할 것으로 사료된다.

II. 소프트웨어 수명주기 프로세스 동향 분석

본 장에서는 CALS/EC 정보시스템의 개발 전략의 일환으로 현재까지 미국방성에서 적용되어 오던 MIL-STD-498 및 이에 관련된 소프트웨어 수명주기 프로세스들의 종류와 각각의 차이점에 대하여 비교 분석하고 발전동향에 관하여 고찰해 보았다.

2.1 MIL-STD-498

미국방성의 새로운 소프트웨어 개발표준인 MIL-STD-498, "소프트웨어 개발 및 문서화"는 1994년 11월에 승인되었다. MIL-STD-498의 출현 배경은 첫째, 당시 소프트웨어 분야별로 구분되어 있던 소프트웨어 개발표준을 하나로 통합시키고 둘째, 과거의 표준을 사용하는 과정에서 인식되었던 많은 문제점들을 해결하고 셋째, 변화된 소프트웨어 개발기술 및 개발환경과의 호환성을 확립하고 마지막으로 ISO/IEC 12207에 대한 미국 버전을 구현하기 위한 토대를 제공하기 위한 것이었다.

MIL-STD-498은 기술적으로 성숙되어 있고, 소프트웨어 개발에 관련된 활동을 자체적으로 포함하기 때문에 MIL-STD-498 이외의 다른 표준을 별도로 요구하지 않으며 다양한 유형의 소프트웨어 개발에 적용할 수 있도록 확장성을 가지고 있다. 또한 MIL-STD-498은 시스템 수명주기의 모든 단계에서 적용할 수 있으며, 소프트웨어 개발과정의 효율성을 보장하기 위하여 필요한 경우에는 간소화하여 사용할 수도 있다. 그러나 MIL-STD-498은 별도의 소프트웨어 개발 프로세스를 제공하지 않기 때문에 이를 사용하기 위해서는 과거의 표준을 사용하는 것보다 높은 수준의 기술을 필요로 한다. 즉, MIL-STD-498은 개발자가 자체적인 소프트웨어 개발 프로세스를 보유하고 있는 것으로 가정한다. 따라서 개발자는 자신에게 익숙한 프로세스를 사용할 수 있으며, 이러한 과정에서 MIL-STD-498을 융통성 있게 적용할 수 있다. 그러나 효과적인 개발 프로세스를 갖고 있지 못할 경우에는 MIL-STD-498의 사용에 어려움을 느낄 수 있다.

IEEE/EIA 12207의 여러 가지 프로세스를 구현하기 위한 구현 지침 및 데이터 지침은 MIL-STD-498의 내용과 데이터 요구사항을 재배열하고 재포장한 것이다. 따라서 MIL-STD-498은 IEEE/EIA 12207을 성공적으로 채택하기 위한 전략이라고 할 수 있다.

2.2 EIA/IEEE J-STD-016

EIA/IEEE J-STD-016은 IEEE와 EIA가 개발한 MIL-STD-498의 상용 버전이다. IEEE/EIA는 1996년 1월에 시험적인 사용 목적으로 J-STD-016, "정보기술-소프트웨어 수명주기 프로세스-소프트웨어 개발-획득자/공급자 계약"을 발표하였다.

J-STD-016은 ISO/IEC 12207을 구현하는 첫 번째 단계이며, J-STD-016의 향후 버전은 운영 혹은 유지보수 등과 같은 추가적인 소프트웨어 수명주기 프로세스를 구현하는 것이다. 따라서 J-STD-016은 시험적인 사용기간이 종료되는 대로 완전한 상태의 표준으로 전환될 것이며, 2001년까지는 존재할 것으로 예상된다. 또한 J-STD-016은 소프트웨어 개발 관리 분야의 국제 표준을 정립하기 위한 기본적인 메카니즘으로 활용될 것이다.

MIL-STD-498과 J-STD-016은 용어의 사용에서 다음 <표 II-1>과 같은 차이점을 보이고 있다.

MIL-STD-498	J-STD-016
computer software configuration item	software item
hardware configuration item	hard item
support concept	support strategy
fielding	distribution
software support	software maintenance
privacy	privacy protection
requirements analysis	requirements definition
data item descriptions	software product descriptions

<표 II-1> MIL-STD-498과 J-STD-016의 용어 차이점

J-STD-016은 MIL-STD-498과 달리 계약에 대한 부과가 아닌 자발적인 사용으로 적용된다.

2.3 ISO/IEC 12207

ISO/IEC 12207은 소프트웨어를 획득, 공급, 개발, 운영, 유지보수하기 위한 공통적인 프레임 워크를 제공하기 위하여 개발된 국제 표준이다. ISO/IEC 12207은 1988년에 제안되었으며, 1995년 8월에 국제 표준으로서 발표되었다.

ISO/IEC 12207은 소프트웨어 개발 프로세스만을 언급하고 있는 MIL-STD-498과는 달리 획득, 공급, 운영, 유지보수 프로세스 등과 같은 추가적인 프로세스를 제공한다. 또한 ISO/IEC 12207은 주요 프로세스 간의 인터페이스와 인터페이스 간의 상호작용을 관장하는 상위 수준의 관계를 기술한다.

ISO/IEC 12207과 MIL-STD-498을 비교해보면 <표 II-2>와 같이 MIL-STD-498의 주요 활동이 ISO/IEC 12207의 프로세스와 일치한다는 것을 알 수 있다.

MIL-STD-498	ISO/IEC 12207
소프트웨어 구성관리	구성관리 및 감리
소프트웨어 품질보증	소프트웨어 품질보증
소프트웨어 산출물 평가	검증 및 확인
합동 기술검토 및 관리검토	합동 기술검토 및 관리검토
교정 활동	문제 해결
프로젝트 계획 및 관리	관리
소프트웨어 개발환경 설정	기반구조

<표 II-2> MIL-STD-498과의 일치 프로세스

또한 MIL-STD-498의 활동 중에는 ISO/IEC 12207의 문서화 프로세스 및 개선 프로세스와 일치하는 것이 있다. MIL-STD-498의 다음과 같은 활동은 ISO/IEC 12207의 한 프로세스 즉, 개발 프로세스와 일치한다.

- 시스템 요구사항 분석
- 시스템 설계
- 소프트웨어 요구사항 분석
- 소프트웨어 설계
- 소프트웨어 구현 및 단위 테스팅
- 단위 통합 및 테스팅
- 소프트웨어 구성항목 자격 테스팅(qualification testing)
- 소프트웨어 구성항목/하드웨어 구성항목 자격 테스팅
- 시스템 자격 테스팅
- 소프트웨어 사용 준비
- 소프트웨어 인계 준비

ISO/IEC 12207의 획득 프로세스, 공급 프로세스, 운영 프로세스, 유지보수 프로세스, 혹은 훈련 프로세스 등에 대응되는 활동은 MIL-STD-498에 존재하지 않는다. 이 것이 두 표준 간의 커다란 차이점이다.

두 표준간의 또다른 차이는 ISO/IEC 12207은 MIL-STD-498보다 상당히 일반적인 수준으로 구성되어 있다. MIL-STD-498은 각 활동들과 관련된 엔지니어링 및 데이터 요구사항을 기술하기 위하여 본문과 데이터 항목 기술서에 구체적인 요구사항을 기술하고 있지만 이에 대응되는 ISO/IEC 12207 프로세스는 요구사항을 일반적인 수준에서 기술하고 있다. 또한 ISO/IEC 12207은 데이터 항목 기술서를 갖지 못하고 있으며 MIL-STD-498의 요구사항들은 보다 상세한 수준에서 ISO/IEC 12207의 대응되는 요구사항과 호환성을 갖는다.

2.4 IEEE/EIA 12207

IEEE/EIA 12207은 ISO/IEC 12207의 미국 버전으로써 ISO/IEC 12207의 내용에 MIL-STD-498(혹은 EIA/IEEE J-STD-016)의 기술적인 내용을 포함시킨 것이다. IEEE/EIA 12207은 ISO/IEC 12207의 본문과 부록을 그대로 수정없이 적용하고, 이에 추가하여 MIL-STD-498과 MIL-STD-498의 데이터 항목 기술서 및 IEEE 소프트웨어 공학 표준으로부터 유도한 세부지침을 포함하고 있다. 따라서 IEEE/EIA 12207은 MIL-STD-498보다 그 범위가 넓다. 즉 IEEE/EIA 12207의 범위는 소프트웨어 수명주기 전체를 포함한다.

IEEE/EIA 12207과 ISO/IEC 12207의 가장 커다란 차이는 ISO/IEC 12207은 소프트웨어 수명주기 동안에 개발하고 기록해야 하는 데이터 항목의 유형만을 언급하고 잇는 반면에 IEEE/EIA 12207은 MIL-STD-498로부터 편입해 온 데이터 항목에 대한 상세한 설명을 제공한다.

IEEE/EIA 12207은 개별적인 프로젝트의 수준보다는 조직 수준의 일관성을 유지하는데 초점을 두고 있다. 따라서 IEEE/EIA 12207을 효과적으로 활용하는 방법은 IEEE/EIA 12207의 요구사항을 따르는 일련의 프로세스와 절차를 조직 수준에서 개발하여, 이를 조직 전체에 활용하는 것이다. 개별적인 프로젝트는 조직 수준의 프로세스와 절차 중에서 적절한 부분을 선택하여 프로젝트에 적용시킴으로써 수행할 수 있다.

IEEE/EIA 12207은 12207.0, 12207.1 그리고 12207.2로 지정된 세부분으로 구성되어 있다. 12207.0은 ISO/IEC 12207의 전문을 그대로 수용한 것이며, 새로이 추가된 부분은 다음과 같

다.

1) ISO/IEC 12207의 본문에서 발견한 오류의 목록을 제공하는 부분

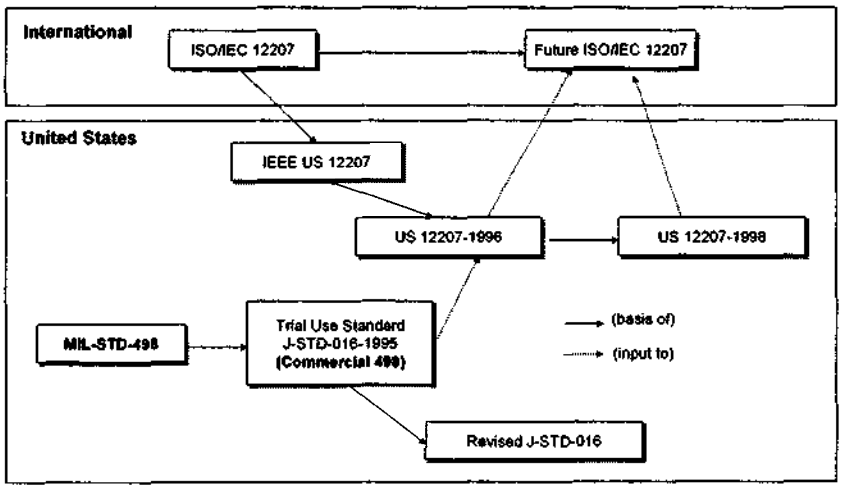
2) 표준에서 명시한 프로세스 요구사항과 이들 프로세스가 생산하는 데이터의 의미를 명확히 하기 위한 일련의 프로세스 목적 및 데이터 목적

3) 조직 수준의 준수를 강조하고, 이러한 준수 수단의 문서화를 요구하는 문구

12207.1은 12207.0의 데이터 목적을 확장하기 위한 권고사항을 제공하는 지침 문서이다. 특정한 문서 혹은 전자 데이터가 특정한 프로세스에 어떻게 연관되는지에 대한 지침을 원하는 사용자를 위하여, 이 부분은 다양한 문서의 내용에 대한 권고사항을 제공한다. 12207.2는 미국이라는 상황에서 IEEE/EIA 12207 프로세스를 구현하는데 필요한 권고사항을 제공하는 지침 문서이다.

2.5 소프트웨어 수명주기 프로세스 발전 동향

지금까지 소프트웨어 개발 표준인 MIL-STD-498과 이에 관련된 다른 소프트웨어 수명주기 프로세스의 종류와 차이점에 관하여 살펴보았다. 미국방성은 MIL-STD-498을 국제 표준과 미국가 표준으로 발전시키겠다는 계획 하에 제정하였으며, 2년간의 사용 목적으로 MIL-STD-498을 공표하였다. 현재 이러한 계획의 일환으로 ISO/IEC 12207이 1995년 8월에 국제 표준으로 승인되었고, IEEE/EIA 12207이 1998년 5월에 미국가 표준으로 채택되면서 공식적으로 MIL-STD-498의 사용을 취소하였다. 또한 기존에 MIL-STD-498을 적용해 왔거나 계속적으로 MIL-STD-498을 적용하기를 원하는 조직과 프로젝트를 위하여 J-STD-016을 사용하도록 하고 있다. J-STD-016은 시험적인 사용기간이 종료되는 대로 완전한 상태의 표준으로 전환될 것이며, 2001년까지는 존속할 것으로 예상된다. MIL-STD-498을 중심으로 한 소프트웨어 수명주기 개발 프로세스의 발전 동향은 다음 <그림 II-1>과 같다.



<그림 II-1> 소프트웨어 수명주기 발전 동향

III. 국방분야 소프트웨어 수명주기 프로세스

3.1 개요

현재 국내의 국방분야에서는 정보시스템 구축의 효과적이고 효율적인 관리를 위하여 국방분야의 정보시스템 개발절차를 개발하여 적용하고 있다. 현재 적용되고 있는 개발절차는 MIL-STD-498을 중심으로 한 소프트웨어 개발 위주의 프로세스 이외에 사업의 시작단계부터 종료단계까지의 포괄적인 수명주기 프로세스를 포함하고 있다. 이 수명주기는 정보시스템의 소요를 제기하고 결정하는 기획단계, 사업 추진을 확정하기 위한 계획단계, 예산반영을 위한 예산단계, 정보시스템 개발을 위한 집행단계, 정보시스템 개발 후의 유지보수를 위한 운용유지 단계 등으로 구분되어 있다. 그러나 MIL-STD-498을 기반으로 만들어진 이 개발절차는 기존의 개발방식 및 고정 관념을 탈피하지 못하여 MIL-STD-498이 강조하는 개발자(developer)에 대한 융통성(flexibility)을 제공하기에는 아직까지 미흡한 실정이다. 본 장에서는 현재 국방분야에 적용되고 있는 소프트웨어 수명주기 프로세스 중에서 개발중심의 프로세스(집행단계)에 대한 취약점을 분석하고, IV 장에서 현 국방분야 소프트웨어 개발 프로세스의 개선방향을 제시하였다.

3.2 1990년대 소프트웨어 동향

급변하는 정보기술의 발전에 따라 관리적인 측면과 기술적인 측면에서 1990년도에 들어 다음과 같은 경향들이 나타나기 시작하였으며, 소프트웨어 수명주기 프로세스에도 이러한 개념들이 수용되었다.

3.2.1 Integrated Product Team Approach

이 개념은 과거의 발주자와 수주자 사이의 관계를 개선해야 한다는 의미를 지닌다. 즉, 과거의 발주자와 수주자의 대립 관계로 프로젝트를 진행하였던 관념을 탈피하여 공동체 의식을 갖고 프로젝트를 진행하자는 개념이다. 이러한 개념은 과거의 공식적 검토회의를 지양하고 비공식 검토회의를 양측이 공동의식 속에서 활성화하여 문서화의 부담을 경감하자는 의도이다. MIL-STD-498의 경우 Joint management review와 Joint technical review가 이러한 개념을 포함하고 있다.

3.2.2 The Three Rule of S/W Development

이 것은 프로젝트를 관리하는데 있어서 프로세스 관리의 중요성을 강조하는 의미이다. 과거의 프로젝트 진행방식이 한 프로세스가 끝난 후 다음 프로세스를 순차적으로 진행하는 방식을 탈피하고 적적한 시기에 적절한 프로세스들을 선택하여 순차적, 반복적 혹은 중복적으로 사용할 수 있다는 의미이다.

3.2.3 Plan Your Work, Work Plan

프로젝트 계획을 수립하고 나면 그 계획을 준수하면서 프로젝트를 진행하라는 의미이다. 그러므로 실질적인 계획을 수립하기 위해서는 프로젝트 계획의 의미가 매우 중요해질 수밖에 없다. 이러한 의미에서 MIL-STD-498에서는 소프트웨어 개발계획서(SDP: S/W Development Plan)를 프로젝트의 관리 및 통제 수단으로 상당히 강조하고 있다.

3.2.4 Supportability

근래의 정보시스템 개발에 있어서 유지보수에 대한 비중이 날로 증가하고 있다. 또한 아웃소싱의 형태로 정보시스템을 획득하는 비중이 증가하면서 개발이 끝난 후에 개발조직에서 계속적으로 유지보수를 한다는 것은 불가능해지고 있다. 이러한 의미에서 계획단계부터 유지보수에 대한 준비를 고려해야 한다는 것이 운영가능성(supportability)의 개념이다.

3.2.5 Object-Oriented Methodologies

90년대에 접어들면서 객체지향 개발방법론이 실용화되기 시작하였고 현재의 많은 프로젝트들이 객체지향 개발방법론을 적용하여 진행되고 있다. 객체지향 개발방법론에서는 기존의 단계적 개발방법보다는 반복적 개발방법을 선호하고 있다. 이러한 소프트웨어의 개발방법론의 변화도 소프트웨어 수명주기 프로세스에 필수적으로 수용됨으로써 프로세스의 관리를 반복적 혹은 진화적으로 할 수 있는 골격을 제공하고 있다.

3.3. 국방분야 소프트웨어 수명주기 프로세스

국방분야에서는 정보시스템을 관리하기 위한 소프트웨어 수명주기 프로세스를 개발하여 계속적으로 보완하면서 국방분야의 모든 정보시스템 개발에 적용해오고 있다. 이 프로세스는 MIL-STD-498의 개념을 기본 골격으로 하고 있으나 주로 기존의 폭포수 모형(waterfall model)을 따르고 있다. 다음 <표 III-1>은 현재의 국방분야 소프트웨어 수명주기 중 집행단계의 절차를 보여 주고 있다.

현재의 프로세스는 공식검토를 위주로 진행되기 때문에 문서화 작업이 많을 뿐 아니라 IPT의 개념을 살릴 수 없다. 또한 단계적 개발절차만을 따르기 때문에 반복개발 프로세스를 구현할 수 없는 단점을 갖고 있다. 특히 이 문제는 객체지향방법론을 적용할 때 매우 비효율적이다. 또한 운영가능성(supportability)의 중요한 활동이 누락되어 있으며, 각 활동들과 활동을 수행함으로써 생산되는 정보들 사이에 약간의 불일치를 보이고 있다.

가장 중요한 취약점은 대규모 정보시스템이 각각의 하부시스템으로 분리되어 개발될 때 이를 위한 프로세스 관리를 지원할 수 있는 프로세스 관리 구조가 존재하고 있지 못하다.

단 계	활 동	문 서	검토 및 확인
체계 설계	체계 요구사항 분석	S/W개발계획서 (SDP) 운영개념기술서(OCD) 체계명세서(SSS) 연동요구명세서(IRS)	체계요구사항 검토 (SRR) -필요시-
	체계 설계	체계설계기술서(SSDD) 연동설계기술서(IDD)	체계설계검토 (SDR)
S/W 규격 확정	S/W 요구사항 분석		
	S/W 요구규격 확정	S/W요구명세서(SRS) IRS S/W시험계획서(STP)	S/W명세검토 (SSR)
S/W 설계	S/W 개략 설계	S/W설계기술서(SDD) IDD DB설계기술서(DBDD)	개략설계검토 (PDR) -필요시-
	S/W상세설계	SDD, IDD, DBDD	상세설계검토 (CDR)
S/W 구현	단위 S/W 구현 및 시험	S/W개발화일(SDF) S/W시험기술서(STD) S/W시험결과보고서(STR)	
단위 S/W 구현	단위 S/W 통합 및 시험	STD, STR	CSCI시험 준비사항 검토 (TRR)
체계통합 및 시험	CSCI 성능시험	S/W설치계획서(SIP) STD, STR	
	CSCI/HWCI 통합 성능시험	STD, STR S/W이전계획서(STrP)	
	체계통합 성능 검증시험	STD, STR	기능적구성확인 (FCA) 물리적구성확인 (PCA)

<표 III-1> 국방 소프트웨어 개발절차

IV. 국방 소프트웨어 수명주기 프로세스 발전방향

4.1 개선된 국방 소프트웨어 수명주기 프로세스 모델

현재의 프로세스를 개선하기 위해서 우선적으로 현재의 프로세스에서 누락되어 있는 주요 활동들을 보완하였다. 보완된 활동에는 소프트웨어개발계획 및 설치계획, 이전계획, 시험계획을 수립하기 위한 프로젝트 계획 및 관리 활동을 추가하였다. 또한 소프트웨어 인수 및 설치를 위한 활동으로 소프트웨어 사용준비, 소프트웨어 인계준비 활동을 추가하였으며, 전반적인 프로세스의 용어와 불필요한 활동들을 제거하였다. 또한 개발 관리 지원을 위한 프로세스들을 추가하였다. 본고에서 제안하는 프로세스 및 관련사항 들은 다음<표 IV-1>과 같다.본고에서는

CALS/EC 개발전략의 일환으로 소프트웨어 프로세스 수명주기 중 국제표준인 ISO/IEC 12207과 현재 미 국가 표준으로 채택된 IEEE/EIA 12207에 관하여 고찰해보고, ISO/IEC 12207의 모체라 할 수 있는 미국방성의 소프트웨어 개발 프로세스인 MIL-STD-498과 비교 분석 하였다. 또한 현재 국내 국방분야에서 적용하고 있는 소프트웨어 수명주기 프로세스에 조명해보았다.

MIL-STD-498을 근간으로한 국방분야의 소프트웨어 수명주기 프로세스는 그 내용이 아직도 반복개발의 개념이나 프로젝트 특성에 따른 프로세스의 관리 등 새로운 개발 철학들을 수용하고 있지 못한 실정이다. 본 연구에서는 현 국방분야의 소프트웨어 수명주기 프로세스의 개선안으로 프로세스를 재정립한 후, 점진적 개발접근방법, 진화적 개발접근 방법의 수용과 프로세스의 특성에 따라 프로세스들을 선택.적용할 수 있는 MIL-STD-498을 기반으로 한 테일러링 방법을 제시하였다.

구 분	활 동	산 출 물	검토 및 확인
개발단계 프로세스	프로젝트 계획 및 관리	SDP, STP, SIP, STrP	비공식적인 합동기술검토 및 관리검토로 수행
	소프트웨어 개발환경 설정	SDL	
	체계 요구사항 분석	OCD, SSS, IRS	
	체계 설계	SSDD, DBDD, IDD	
	소프트웨어 요구사항 분석	SRS, IRS	
	소프트웨어 설계	SDD, DBDD, IDD	
	소프트웨어 구현 및 단위시험	SDF	
	단위통합 및 시험	SDF	
	CSCI 자격시험	STP, STD, STR	
	CSCI/HWCI 통합 및 시험	SDF	
	체계 자격시험	STP, STD, STR	
	소프트웨어 사용준비	SPS, SVD, SIOM, SUM, COM	
	소프트웨어 인계준비	SPS, SVD, CPM, FSM	
관리 프로세스	소프트웨어 구성관리 소프트웨어 산출물 평가 소프트웨어 품질보증 교정활동 합동 기술검토 및 관리검토	구성관리 및 품질보증은 별도의 계획으로 분리 가능	

<표 IV-1> 개선된 국방 소프트웨어 수명주기 프로세스

개선된 프로세스에서 소프트웨어 개발환경 설정의 활동은 개발자의 자체적인 정보보유 형태인 소프트웨어 개발 라이브러리(SDL: Software Development Library)형태로 산출되는 개발정보를 유지하고, 개발자 내부 시험 활동인 소프트웨어 구현 및 단위시험, 단위통합 및 시험, CSCI/HWCI 통합 및 시험은 소프트웨어 개발화일(SDF: Software Development) 형태로 개발동안 산출되는 정보를 유지한다. 또한 구성관리 및 품질보증 활동은 관리 목적상 필요시 별도의 계획으로 분리할 수 있으며, 개발 전과정에 걸친 검토는 중간 이정표 상에서의 공식검토가 아닌 합동 기술검토 및 관리검토의 형태로 비공식적으로 수시로 진행한다. 단, 비공식 검토에서 결정되지 않는 사항에 대해서만 공식검토의 형태로 진행할 수 있다.

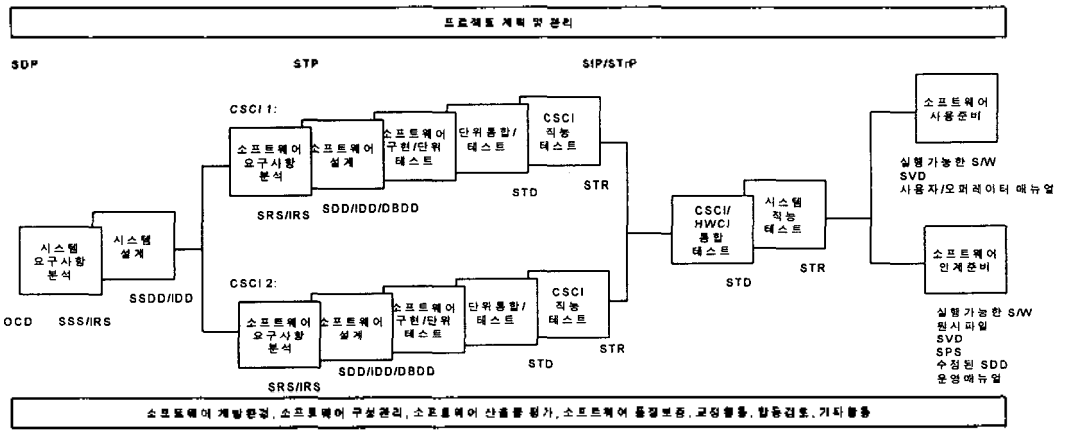
4.2 소프트웨어 수명주기 프로세스 테일러링 절차

소프트웨어 수명주기의 프로세스들을 관리하는 가장 중요한 방안은 프로젝트의 성격 및 환경에 따라 어떤 프로세스들을 선택하여 적용할 것인가에 있다. 그러므로 적용할 프로세스와 그에 관련된 산출정보들을 테일러링할 수 있는 방법이 포함되어야 한다. 개선된 프로세스에서는 국방 소프트웨어 수명주기 프로세스의 집행단계가 시작되는 시점 즉, 개발조직을 결정하는 시점에서 프로세스에 대한 테일러링을 할 수 있는 방안을 제시하였다. 테일러링 절차는 다음과 같은 단계로 구성된다.

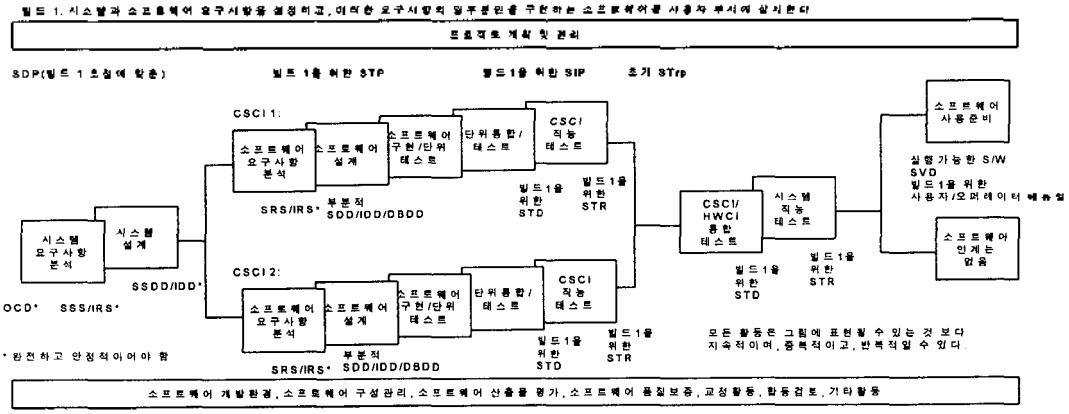
4.2.1 정보시스템 획득전략 결정

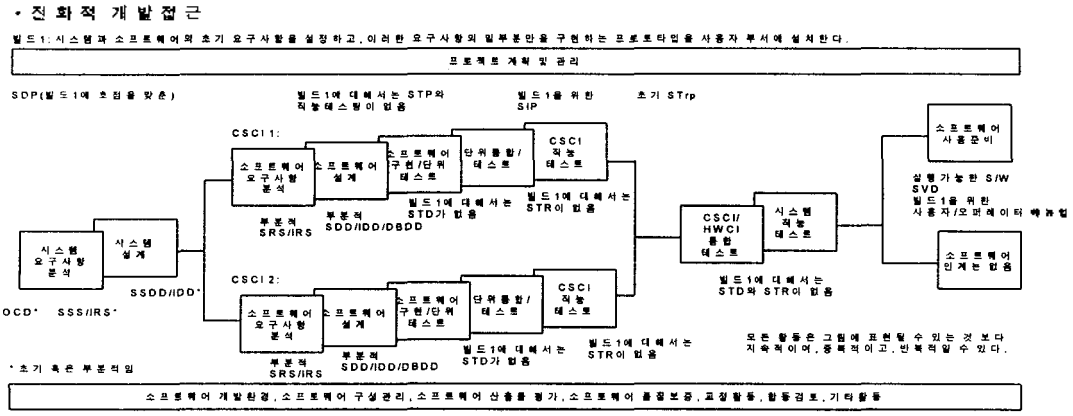
우선 정보시스템을 획득하기 위한 전략을 결정해야 한다. 획득전략은 크게 3가지로 구분할 수 있다. 3가지 획득방법은 단계적 개발접근, 점진적(incremental) 개발, 진화적(evolutionary)개발 접근 등이 있으며 구체적인 개발방법은 <그림 IV-1>과 같다.

• 단계적 개발접근



• 점진적 개발접근





<그림 IV-1> 정보시스템 획득전략

4.2.2 빌드계획 수립

정보체계의 획득전략에 고려하여 빌드의 개수를 정의한다. 단계적 개발접근은 빌드가 1개 존재하며, 점진적 개발접근 및 진화적 개발접근은 빌드가 2개 이상 존재할 수 있다. 빌드를 정의할 때는 각 빌드의 목적을 정의해야 하며, 빌드는 프로젝트가 진행됨에 따라 점차적으로 명확해질 수 있다.

4.2.3 빌드별 프로세스(활동) 테일러링

이 단계는 각 빌드의 목적 및 특성에 따라 개발활동들의 적용여부 및 적용수준을 적절히 판단하여 테일러링하는 것이다. 테일러링을 할 때 중요한 사항은 문제가 되고 있는 활동이 현재 빌드의 목적을 지원하는지, 앞으로의 빌드목적을 지원하는지, 혹은 소프트웨어 운영개념의 목적을 지원하는지 등에 대하여 결정하는 것이다. 각 빌드에 대한 개발활동의 테일러링도 프로젝트가 진행됨에 따라 점차적으로 명확해질 수 있다.

4.2.4 빌드별 데이터 항목 요구사항 테일러링

이 단계는 개발활동의 지침으로써 데이터 항목 요구사항을 테일러링하는 것이다. 특정 소프트웨어를 개발하기 위해서 데이터 항목 요구사항에서 요구하는 정보를 모두 생산할 필요는 없다. 개발활동의 테일러링은 관련되는 각각의 데이터 항목 요구사항을 검토하여 프로젝트, 빌드, 소프트웨어 유형에 따라 불필요한 부분을 테일러링으로 제거한다. 각 데이터 항목 요구사항에 대한 테일러링은 프로젝트가 진행됨에 따라 명확해질 수 있다.

4.2.5 산출물 정의 및 소프트웨어개발계획서 작성

개발활동과 데이터 항목 요구사항을 테일러링한 후, 공식적으로 인도 받을 산출물들을 정의한다. 다음으로 인도 받을 납품물을 기존 문서의 형태, 파일형태, CASE 도구 데이터 형태등 여러 가지 유형의 납품 받을 형식을 지정한다. 또한 산출물을 인도 받을 일정을 판단한 후 제안서를 제출할 때, 초기 테일러링한 결과를 반영한 소프트웨어 개발계획서를 첨부

하도록 함으로써 소프트웨어 수명주기 프로세스를 관리할 수 있는 기반을 마련한다.

V. 결 론

본 고에서는 CALS/EC 개발을 위한 소프트웨어 수명주기의 발전동향을 살펴보고 현재 국내 국방분야 및 미 국방분야에서 많이 적용해오던 MIL-STD-498과의 차이점을 비교.분석하였다. 또한 현 국방분야의 소프트웨어 수명주기 프로세스를 조명해보고 취약점을 분석한 후, MIL-STD-498을 근간으로 한 개선방향을 제시하였다. 개선안으로써 누락된 개발활동을 추가하고 단계적 개발접근 방법외에 점진적, 진화적 개발 접근방법을 추가하였다. 또한 프로젝트 특성에 따른 프로세스 관리를 위해 집행단계의 초기에 프로세스를 테일러링할 수 있는 방안을 제시하였다.

아직까지 국내 정보산업계의 현업에서 정보시스템의 규모가 대형화됨에도 불구하고 소프트웨어 수명주기 프로세스의 효과적이고 효율적인 적용이 미비한 실정이다. 조속히 국가차원의 소프트웨어 수명주기 프로세스의 정립이 필요하며 이에 대한 기반조성이 시급할 것으로 사료된다.