다중측면 모델을 이용한 문서화 지원 시스템

이관호* · 김창화** · 백두권*

* 고려대학교 전산과학과

** 강릉대학교 전자계산학과

— 요 약 —

본 연구의 목적은 문서의 다중측면을 고려한 문서화 지원시스템을 구현하는 것이다. 기존의 소프트웨어 개발 문서를 지원하기 위한 시스템은 소프트웨어 개발단계 측면에서의 문서들을 작성하기 때문에 개발 완료후 소프트웨어 이용자를 위한 문서나 서비스 측면에서 필요한 문서들은 새로운 작업을 통해 작성되어야 한다. 따라서 중복되는 자료로 인한 저장 공간의 낭비와 개발 비용의 상승을 가져온다. 본연구에서는 다중측면 모델을 이용하여 소프트웨어 문서를 정의하고 정형화된 표현으로 문서화과정을 정의했다. 또한 소프트웨어 개발 단계 측면에서 만들어지는 문서 뿐만 아니라 기능측면, 이용자측면에서의 문서를 제공함으로써 다양한 문서지원이 가능한 문서화 지원시스템을 설계한다.

1. 서 론

소프트웨어 개발 문서 작성을 지원하기 위한 시스템이 소프트웨어 생산성 향상 및 개발비용의 절감을 위해 개발되어 왔다. 소프트웨어 개발시 작성되는 문서를 지원하는 시스템에 대한 최초의 연구는 1970년대 University of Michigan의 ISDOS 프로젝트 부터 시작되었다고할 수 있다. 자동화된 분석도구를 만드는 이 프로젝트에서 정보처리 시스템에 대한 구조적 문서와 분석을 위한 컴퓨터지원 기술인 PSL/PSA(Program Statement Language/Program Statement Analyzer) 의 개발로 형식적 언어를 이용하여 기능적 명세와 요구사항을 서술하였다. 이 PSL/PSA는 소프트웨어 개발 과정에 있어 특정단계의 자동화를 위해 도입된 도구라 할 수 있다 [DAN77].

1980년대 들어 비정형자료인 텍스트를 기존의 DB를 이용하여 처리하려는 노력이 있었다. 특히 데이타 표현이 간단하고 조작이 용이한 관계형 데이타베이스를 확장하여 비형식적 데이타를 처리하는 연구가 있었다[El186a],[El186b]. 이런 연구들은 개발과정에서 생성되는 문서의 관점에서 구현된 단계별 문서중심의 시스템이라 할 수있다.

또 다른 연구에서는 저장된 정보를 사용자의 자연스런 연상작용에 의해 사용

자의 필요와 관심에 따라 단위 정보인 노드와 노드 사이의 링크를 통해 검색하는 하이퍼 텍스트를 이용 문서를 다루고 있다[Jak 90].

소프트웨어 개발과정을 살펴보면 요구 분석에서 규정된 문제 중심의 비형식적 서술이 도형중심의 도구나 언어중심의 도구를 이용해 기술되는 디자인 단계 및 특정언어에 의한 구현단계를 거치면서 형식적 서술으로 변화된다는 것을 알 수 있다. 이렇게 소프트웨어 개발과정을 특정 기능 구현을 위한 변환과정이라고 규정할수 있다.

본연구에서는 문서화과정을 형식적 서술로 정의하고 다중측면 모델을 이용해 문서를 규정했으며 문서 사이의 관련성을 규정하여 요구사항 변화로 인한 문서의 유지보수를 보다 효율적으로 가능하게 한다.

DOCUS(DOCUmentation Support system)에서는 소프트웨어 개발과정중 일어나는 문서화를 지원하기위해 다음과 같은 기능을 갖는다.

- 미리 정의된 문서 구조를 이용해 문서를 수정하는 기능
- 사용자가 새로운 문서구조를 정의하는 기능
- 개발단계 측면의 문서를 지원하는 기능
- 문서수준을 고려한 이용자 측면에서의 문서를 제공하는 기능
- inter-,intra-relationship을 이용해 문서의 기능적측면의 문서를 제공하는 기능

DOCUS에서 정의되는 소프트웨어 생명 주기는 고전적 모델인 water-fall모델을 기본으로 1) 요구사항 분석 단계, 2) 기능정의 단계 3) 설계 단계, 4) 구현단계, 5) 테스팅 단계로 구분한다[Rob86].

2. 다중측면 모델에 의한 문서의 표현

2.1 소프트웨어 개발문서의 종류

소프트웨어 개발시 다양한 종류의 문서가 만들어지는데 크게 이용 측면에 따라 구분하는 방법과 문서작성 시기에 따라 구분할 수 있다 . 이용 측면에 따라 구분하는 방법은 실제로 문서를 작성하는 사람과 문서를 이용하는 사람의 관점에서 구분하는 방법으로 다음 3가지로 구분할 수 있다.

1) 생명 주기와 관련된 개발 단계 측면에서의 문서.

2) 소프트웨어 이용자들의 교육(training) 측면을 고려한 문서.

3) 문서들이 제공하는 서비스 측면을 고려한 문서로 나눌수 있다[황88a], [DAN77].

소프트웨어 문서를 문서작성 시기에 의해 구분하면 다음과 같다.

1) in-process 문서 : 소프트웨어 개발과정 중에 만들어지는 문서로 생명주기와 관련된 문서가 있다.

2) post-process문서 : 소프트웨어 개발후에 만들어지는 문서로 서비스 측면에서 개발되는 문서와 이용자 측면에서 제공되는 문서가 있다.

이와 같이 문서의 종류는 문서가 이용목적, 이용대상, 그리고 이용시기에 의해 구분되어진다. 이런 분류방식에 의해 유추할 수 있는 문서의 종류를 문서가 가지고 있는 계층적 성질을 잘 표현하고 문서의 다양한 종류를 여러측면에 의해 표현하기 용이한 EA(Entity Aspect)모델에 의해 표현할 수 있다[김89]. EA모델에 의한 문서표현에 있어 고려되어야 할 측면의 종류와 기능은 다음과 같다.

1) 역할측면 : 개체가 가지는 기능 및 역할을 표현하는 측면으로 문서가 가지는 기능 및 역할은 크게 기록참조, 교육훈련, 통신수단이다.

2) A_PART_OF 측면 : 개체를 구성하는 구성요소들에 의해 개체를 표현하는

측면으로 aggregation을 나타낸다. 문서의 구성요소는 머릿말과 몸체로 구성되며 몸체는 다시 section과 subsection으로 구성된다.

3) IS_A 측면 : 개체를 서브클래스로 나누는 관계 또는 여러 클래스로 부터 공통적 속성을 유추하여 새로운 상위 클래스로 통합화하는 관계를 나타내는 generalization개념을 표현하는 측면으로 문서는 개발문서, 이용자문서, 서비스문서 등의 서브크래스를 가진다.

4) 속성측면 : 개체를 나타내는 성질을 표현하는 측면으로 문서의 내용은 텍스트 또는 그림의 속성을 가진다.

위에서 언급한 EA 모델을 이용해 문서를 나타내보면 그림 1과 같다.

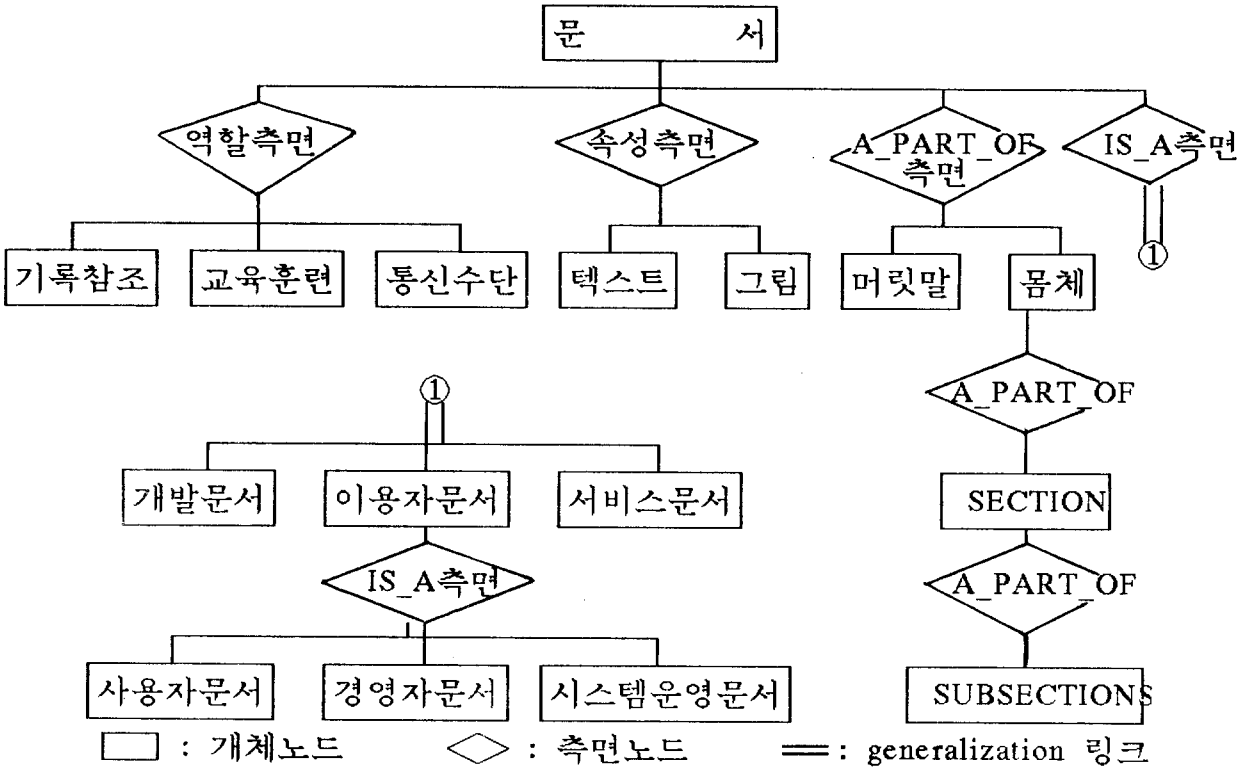


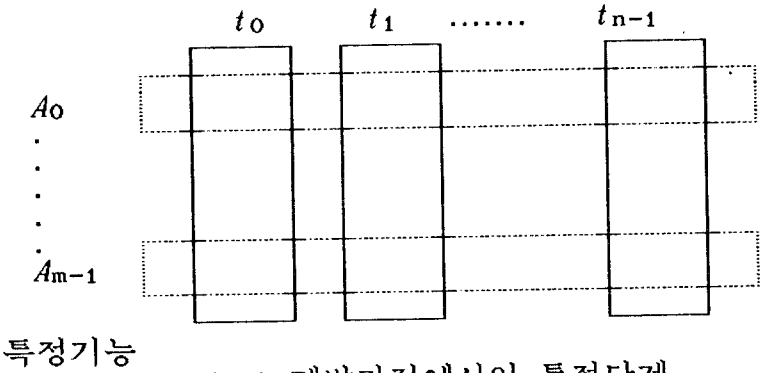
그림 1. EA 모델을 이용한 문서의 구분

2.2 문서화 과정에 대한 정형화된 표현

소프트웨어 개발을 특정 기능 구현을 위한 변환과정이라고 고려할 수 있다. 즉 요구분석 단계에서 목적달성을 위해 시스템 요소를 정의하고 입력, 출력, 처리에 이용되는 상태를 나타내는 각자의 기능을 규정하는 것으로 기능 X를 수행하기 위한 연산, 논리 알고리즘, 요구되는 입력, 기능 X에 의해 만들어지는 출력 등을 in형식적하게 정의한다. 설계 단계에서는 요구단계에서 정의된 기능 X에 대한 비형식적 서술을 기본으로 데이타 흐름도, Warnier-Orr Diagram, NS(Nassi-Shneidermann) Chart와 같은 도형중심의 도구나 PDL과 같은 언어중심의 도구를 이용해 기능들과의 인터페이스를 고려한 시스템을 정의한다. 구현단계에서는 설계 단계의 생산물을 근거로 소프트웨어 개발를 위한 소프트웨어와 하드웨어 환경에 맞는 언어로 구현한다. 이런 소프트웨어 개발과정에서의 특징은 다음과 같다.

1) 비형식적서술이 형식적 서술로 변환.
2) 인간 중심적 서술에서 기계 중심적 서술로 변화.
3) 개발단계가 증가할 수록 소프트웨어 모듈간의 독립성이 강조된다.

소프트웨어 개발 시스템과 문서와의 관계는 특정 기능 X의 형태들로 연관지어 나타낼 수 있다. 그림 2는 소프트웨어 개발 시스템을 개발단계 측면과 개발되는 대상인 기능 측면으로 모델링한 것이다. 즉 소프트웨어 문서란 소프트웨어 도구를 이용해 요구에 맞는 소프트웨어를 제공하기 위한 일련의 작업인 소프트웨어 개발과정에 대한 서술로 각 개발단계에서 일어나는 행위에 대한 서술이라 볼수 있으며 작업의 관점에서 볼때에는 각 단계별로 요구사항을 만족하는 소프트웨어를 개발하기 위해 소프트웨어의 요구사항을 만족하는 변형된 형태의 모임이라 볼수 있다.



t_i : 개발과정에서의 특정단계
A_j : 요구사항을 만족하는 특정기능
그림 2. 문서화 과정의 모델링

소프트웨어 개발단계에서 만들어지는 문서의 내용은 해당단계에서 특정 기능에 대한 서술을 나타내는 엔티티의 모임이라 할수 있다. 따라서 개발단계에서의 문서화 시스템은 문서, 기능 X의 서술인 엔티티, 소프트웨어 개발단계 사이의 엔티티 변환을 가능케하는 함수들에 의해 나타낼 수 있다[Ber76a],[Ber84b], [Hor88].

문서화 시스템의 형식적 서술은 다음과 같다.

$S=(D, A, \delta, T)$ 로 정의된다. 여기서

$D=(D_0, D_1, D_2, \cdots D_{n-1})$: 개발단계별 문서들의 집합

$A=(A_0, A_1, A_2, \cdots A_{m-1})$: 요구조건을 만족하는 특정기능에 대한 서술들의 집합

$\delta=(\delta_0, \delta_1, \delta_2, \cdots \delta_{n-1})$: 단계 사이의 변환을 가능하게 하는 변환함수
(단 δ_0=I인 단위함수로 $\delta_0(a_i)=a_i$이다.)

$T=(t_0, t_1, t_2, \cdots t_{n-1})$:개발단계를 나타내는 단계들의 집합

문서들의 집합 D는 각 개발단계의 수행후 만들어지는 문서를 나타낸다. 요구사항을 만족하는 특정기능들의 집합 A는 소프트웨어 개발과정에서 같은 목적을 가지는 기능의 서술인 엔티티들의 모임이라 할 수 있는데 엔티티는 변형함수 δ에 의해 각 과정마다 모듈 설명, 그래픽 도형, 구현코드 등의 다른 형태로 나타난다.

α_i는 2가지 구성요소를 가지는데 하나는 실제내용 서술을 나타내는 d와 문서

의 수준을 나타내는 $\alpha_i=(d,L)$로 정의 하며 여기서 d는 특정기능의 서술을 나타내는 그림이나 문자열로 구성되고 L은 각 엔티티의 문서수준 집합이다. 즉 $L=(l_0, l_1, \cdots l_{n-1})$로 각 엔티티가 가지고 있는 문서의 수준을 나타낸다. 문서의 수준은 저장되어 있는 문서내용 정보에 대한 검색시 문서의 다중측면에 의해 고려되어야할 문서의 지원을 위해 다양한 이용자 계층에게 각 계층의 수준에 맞는 정보의 검색가 가능하도록 한다.

여기서 변환함수 δ는 $\delta:\alpha\rightarrow\alpha$이다. 여기서 A_i는 i단계에서의 특정기능 A_i에 대한 서술을 나타내는 α_i의 변형된 형태로 나타낼 수 있다. 즉 엔티티 A_i는 $A_i=(\alpha_i, \delta_1(\alpha_i), \cdots \delta_{n-1}(\alpha_i))$으로 표현될 수 있다. 이때 특정단계에서의 엔티티를 나타내기 위한 변환함수인 δ의 합성함수 δ_i는 $\delta_i=\delta\circ\delta\circ\delta_1\cdots\circ\delta_i$로 정의한다. 즉 첫번째 소프트웨어 개발단계의 임의의 엔티티 α_i는 변환함수 δ에 의해 다음 단계에서 $\delta_1(\alpha_i)$로 변환되고 마지막 단계에서는 $\delta_{n-1}(\alpha_i)$로 변환된다. 그러므로 집합 A_i는 같은 목적을 갖는 엔티티들의 변환된 형태의 모임이다.

따라서 첫번째 소프트웨어 개발단계에서의 문서 D_{t1}은 $D_{t1}=(\alpha_0, \alpha_1, \cdots \alpha_{m-1})$이다. 즉 초기 엔티티들의 모임으로 t_i단계에서의 문서내용을 규정한다.

임의의 단계 t_i에서의 문서 D_{ti}는 엔티티에 대한 변환이 t_i 만큼 일어난 모든 엔티티들의 모임으로 $D_{ti}=(\delta_{ti}(\alpha_0), \delta_{ti}(\alpha_1), \cdots \delta_{ti}(\alpha_{m-1}))$로 표현될수 있다.

문서 집합 D의 모든 구성요소들의 합집합 $D_0\cup D_1\cup\cdots D_{n-1}$은 엔티티 집합 A의 모든 구성요소들의 합집합 $A_0\cup A_1\cup\cdots A_{m-1}$과 같다. 임의의 단계 t_i에서 엔티티 집합 A_j의 특정 엔티티 구성요소의 표현은 $\alpha_{ti}=\delta_{ti-1}(\alpha_j)$이다.

개발과정에 엔티티의 변형은 각 단계별로 일어나지만 엔티티의 초기단계에 가지고 있던 기능의 속성은 일정하게 유지된다.

2.3 Inter-relationship 과 Intra-relationship의 정의

소프트웨어 문서화 시스템에서 문서들의 관련성은 크게 각 단계별로 만들어지는 문서들 사이의 관련성인 inter-relationship과 문서내의 내부 구성요소들 사이의 관련성인 intra-relatioship으로 나눌수 있다. 문서내부의 관련성을 나타내는 경우 문서의 구성요소인 모듈설명, 텍스트, 프로시듀어, 그림들 사이의 내부 관련성을 가지게 되는데 문서내의 특정기능을 나타내는 엔티티 관련성으로 표현될 수 있다. intra-relationship을 통해 문서의 텍스트와 그림의 연결을 제공하거나 모듈간의 참조를 나타낸다. 단계 t_i의 문서에 대한 intra-relationship r은 $r_{ti}=(\alpha_i,\alpha_j)$으로 표현된다(단, $i\neq j$).

내부적 문서사이의 관련성을 규정하는데는 저장단위인 paragraph 사이의 관련성을 규정함으로써 내부적 문서사이의 관련성을 통해 subsection을 구성하는 텍스트들의 참조 관련성, 텍스트와 그림사이의 연결을 제공해 주는 것으로 내부적 관련성인 intra-relationship은 문서내부에서 이용되는 그림와 텍스트를 연결하여주는 그림참조링크와 문서구조의 물리적인 연결을 제공하는 구조링크가 있다. 구조링크는 문서의 머릿말로 부터 문서를 구성하는 section, subsection등의 연결을 제공하며 DOCUS의 저장단위인 텍스트, 그림까지의 문서구조에 대한 연결 제공함으로써 텍스트나 그림에 대한 검색를 가능하게한다. 그림 3은 문서의 구성 요소와 intra-relation을 계층적 구조에 의해 나타낸것이다.

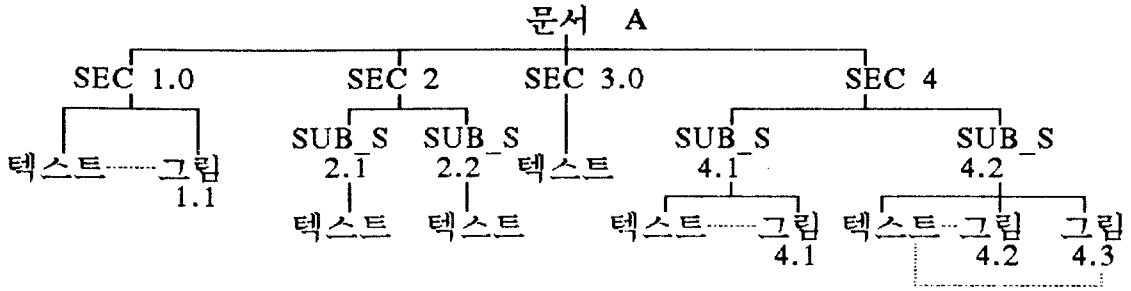


그림 3. 문서의 계층적 표현

개발단계에서 작성되는 문서들간의 관계인 inter-relationship을 통해 특정 엔티티에 대한 검색가 가능하다. 즉 유지보수 작업에 들어 갔을때 개발단계에서 작성되는 많은 문서의 수정을 돕기위해 특정 엔티티와 관련된 여러 문서를 제공해 줌으로 효과적인 유지보수 작업이 가능하다. inter-relationship R은 $R=R_{t0}, R_{t1}, t2, \cdots R_{tn-2}$로 표현될 수 있다. 문서내 임의의 엔티티 α_j에 대한 t_{i-2}와 t_{i-1}단계사이의 inter-relationship은 $R_{ti-2}=(\delta_{ti-2}(\alpha_j), \delta_{ti-1}(\alpha_j))$로 표현될 수 있다[Mar 87]. 즉 inter-relationship은 현단계와 이전단계 및 이후단계에서의 특정 엔티티에 대한 관련성을 제공하는 것으로 그림 4에서와 같이 문서들간의 관련성인 inter-relationship을 정의할 수 있다.

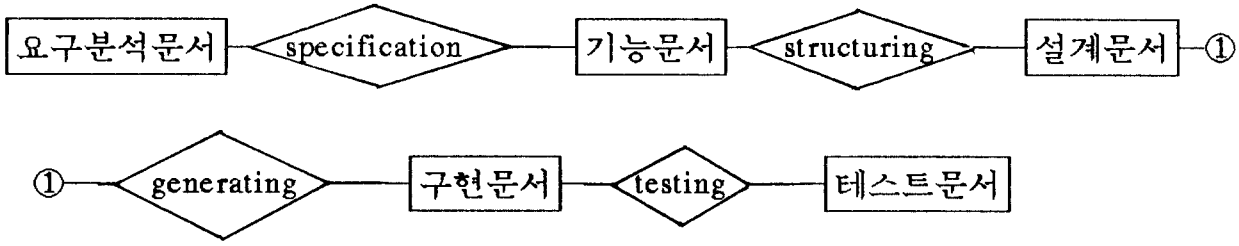


그림 4. inter-relationship

1) specification : 요구분석 명세와 기능문서사이의 관련성으로 기능문서는 요구분서문서에 있는 요구사항들을 만족하기 위한 기능들을 정의한다. 예를들어 기능문서에는 요구조건의 상호참조를 포함하는 내용인 그림 5를 가지고 있는데 이는 기능적 요구사항들이 소프트웨어 설계에 의해 만족되고 특정 요구사항을 구현 시키는 모듈이 필요하다는 것을 나타낸다[황88b] .

모듈이름 / 요구구문	모듈 A	모듈 B	.....	모듈 Z
SUBSECTION 2.1		X		
SUBSECTION 2.2	X	X		
SUBSECTION 4.1				X

그림 5. 요구조건의 상호참조표

2) structuring : 기능문서와 설계명세사이의 관련성으로 기능문서에 제시된 기능들이 소프트웨어적인 표현으로 바뀌는 과정으로 기능정의가 시스템에 대한 논리적 구조로의 변환되는 과정으로 설명할 수 있다.

3) generating : 설계문서와 구현문서사이의 관련성으로 소프트웨어에 대한 논리적 구조에 대한 설명인 설계명세는 각 모듈명세를 나타내는 section에 대한 구현코드를 generating하는 과정을 거치면서 구현문서가 작성된다.

4) testing : 구현문서와 테스팅 문서사이의 관련성으로 특정 엔티티를 수행하기 위한 구현코드에 대한 설명을 갖고 있는 구현문서의 각 section들은 요구사항에 대한 만족 여부를 조사하기위해 모듈별 테스팅과정을 거쳐 테스팅 결과에 대한 문서를 만들게 된다.

2.4 문서의 구조

다양한 문서들은 요구에 맞는 내용들을 포함하고 있지만 문서의 추상화 구조는 일정하다. 즉 PART_OF 측면에서의 문서는 문서의 추상화 구조를 나타내는 것으로 그림 5와 같이 계층구조로 나타낼수 있다.

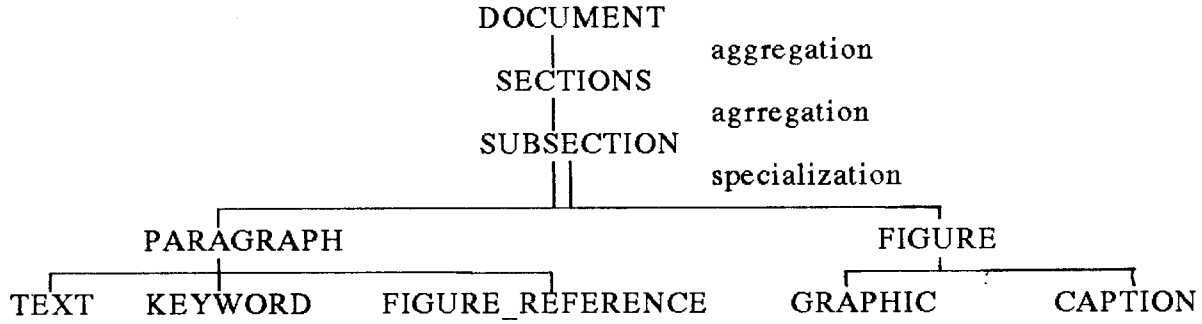


그림 5. PART_OF 측면에서의 문서

DOCUMENT는 SECTION들의 모임으로 나뉘고 SECTION은 다시 SUBSCETION 들의 모임으로 구성된다. SUBSECTION은 PARAGRAPH와 FIGURE로 specialization 할 수 있다. PARAGRAPH는 소프트웨어 개발시 이용자 요구분석에 의한 문제정의, 특정 기능을 수행하는 모듈에 대한 설명, 알고리즘, 구현 코드로 나타난다. PARAGRAPH는 문자열로 구성된 TEXT와 TEXT 검색를 가능하게 하는 KEYWORD 그리고 관련된 그림을 연결 시키는 FIGURE REFERENCE로 구성되며 FIGURE는 소프트웨어 개발과정에서 데이타 흐름도, 순서도로 나타난다. FIGURE는 그림 자체를 나타내는 GRAPHIC 그리고 FIGURE에 대한 설명을 가지고 있는 CAPTION으로 구성된다. 문서들은 추상화된 문서 구조인 DOCUMENT CLASS에 대한 하나의 인스턴스로 나타낼 수 있다.

3. 문서 베이스

시스템 이용자는 문서의 내용을 검색하기 위해서는 구조생성기를 통해 문서의 구조를 만들어내어 문서구조가 subsection의 제목과 문서내용에 대한 키워드를 이용하여 문서베이스를 접근하게 된다. 여기서 사용자는 문서내용 전체에 대한 직접적인 확인 없이도 다니 문서구조에 의해 내용 검색이 가능하다. 그림 6은 문서베이스에 대한 접근 방법을 나타낸 것이다.

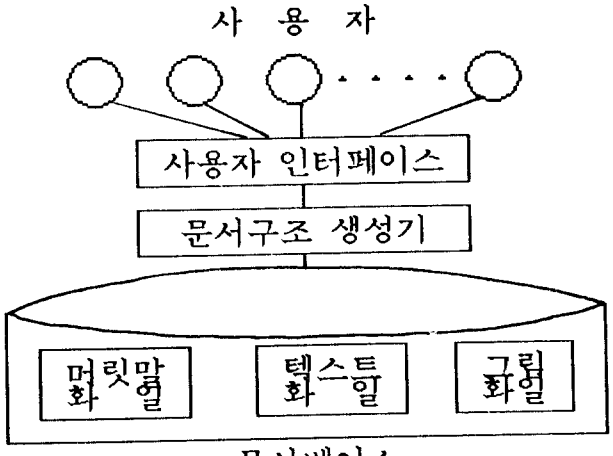


그림 6. 문서베이스의 접근 방법

문서베이스는 문서의 내용을 저장하고 있으며 문서의 내용은 크게 머릿말 화일과 텍스트 화일 그리고 그림 화일이 저장되어 있다. 머릿말 화일을 각 문서의 외적(external)정보를 가지고 있는 부분으로 머릿말 구조는 다음과 같다.

```
structure header { int    doc_number;     /* 문서의 일련번호 */
                   char   title[];        /* 개발하고자하는 시스템의 이름 */
                   char   author[];       /* 문서를 작성한 사람 */
                   char   date[];         /* 문서를 작성한 날짜 */
                   char   abstract[][];   /* 문서에 대한 요약 */
                   char   keyword[];      /* 문서내용에 대한 keyword */
                 }
```

텍스트 화일과 그림 화일은 실제 문서내용을 저장하고 있는 부분으로 subsection단위로 저장되어 있으며 텍스트 화일의 구조는 다음과 같다.

```
structure text {int  subsec_number;  /* 문서를 구성하는 subsection 번호 */
                char subsec_title[]; /* subsection의 제목 */
                char fig_ref;    /* 텍스트가 참조하는 그림에 대한 포인터 */
                char level_type []; /* 문서내용의 수준을 나타내는 문자 */
                }
```

텍스트 화일 내용중 fig_ref는 subsection내 텍스트 내용이 참조하는 그림에 대한 포인터를 가지고 있는 것으로 실제로는 그림화일을 가르키는 포인터이다. 또한 level_type은 문서내용의 수준을 나타내는 것으로 문서내용 수준은 문서의 이용자 측면에서 제공되는 문서의 종류에 의해 나눌수 있으며 내용은 다음과 같다.

1) 일반사용자 수준 : 시스템의 최종 사용자들이 원하는 정보를 나타내는 것으로 시스템 사용자 인터페이스에서의 메뉴에 대한 정보, 시스템 이용 방법 등에 대한 정보 포함

2) 경영자 수준 : 시스템개발에 관련한 경영정보로 시스템을 개발하는데 요구되는 개발을 위한 하드웨어 및 소프트웨어환경 정보, 개발계획 등에 대한 정보 포함

3) 시스템 운영자 수준 : 개발된 시스템을 운영하는데 요구되는 정보로 운영을 위한 하드웨어 환경, 오류발생시 복구를 위한 개발된 시스템의 구현코드 등의 정보 포함

그림 화일은 문서내에 존재하는 테이블, 다이어그램 등의 정보를 저장하고 있으며 텍스트의 fig_ref 연결에 의해 검색된다. 그림 화일의 구조는 다음과 같다.

```
structure text {int   figure_number;      /* 그림의 번호 */
                char figure_title[];      /* 그림의 제목 */
                char caption [][];        /* 그림에 대한 설명 */
                graphic                   /* 그림이 저장되어 있는 부분 */
               }
```

4. DOCUS(DOcument Supporting System)의 설계

DOCUS는 소프트웨어 개발시 문서 작성 및 검색, 이용자 요구변화에 대한 문서수정을 위한 문서내용 검색를 지원해주는 시스템으로 소프트웨어 문서 작성자가 DOCUS를 이용할 수 있게 메뉴방식으로 제공해주는 사용자 인터페이스, 문서작성과,문서수정을 가능하게하는 텍스트 모듈과 그래픽 모듈 그리고 문서검색를 가능하게하는 문서검색 모듈로 구성된다. DOCUS의 시스템 구조는 그림 7과 같다.

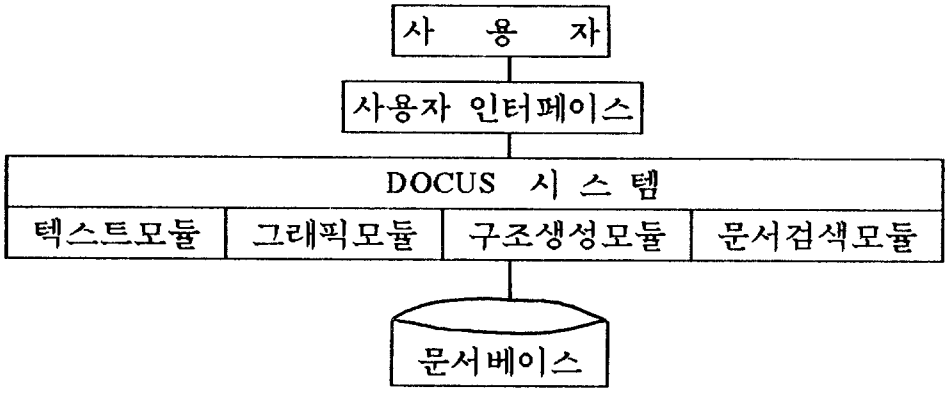


그림 12. DOCUS의 구조

(1) 사용자 인터페이스

문서를 효율적이면서 일관성있게 검색, 저장, 편집하는 것을 목적으로 하여 사용자의 문서작성 또는 검색을 위한 안내를 해주는 부분으로 2단계 메뉴로 구성된다. 메뉴는 초기상태를 나타내는 초기메뉴와 문서의 작성 및 수정을 위한 편집 메뉴, 문서에 대한 검색을 위한 검색 메뉴, 문서구조에 대한 정의를 위한 구조생성 메뉴로 구성된다. 메뉴구조는 그림 13와 같다.

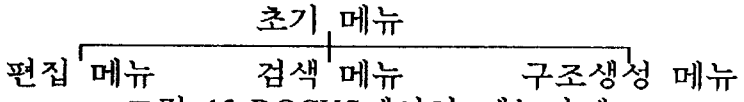


그림 13.DOCUS에서의 메뉴단계

(2) 텍스트 모듈과 그래픽 모듈

DOCUS에서 지원되는 문서의 속성이라 할 수 텍스트와 그림에 대한 작성과 수정을 지원하는 부분으로 다음과 같은 기능을 갖는다.

- 문서 작성 기능은 소프트웨어 개발시 개발자들에 의해 요구되는 문서작성을 담당하는 것으로 각 문서의 저장단위인 subsection별로 작성된다.
- 문서편집 기능은 사용자의 요구변화로 인해 유지보수시 문서의 수정을

담당하고 있다.

(3) 구조생성 모듈

구조생성 모듈은 문서 검색를 위해 문서의 구조인 목차를 작성하는 기능을 가지고 있는 모듈로 구조생성시 subsection의 번호와 제목 그리고 문서내용에 해당하는 키워드를 정의하게 된다. 또한 문서의 다중측면을 지원하기 위해 새로운 문서 구조작성시 subsection 번호, 제목, 키워드와 문서 내용의 수준을 결정한다. 문서내용의 수준을 고려하여 생성한다.

(4) 문서 검색모듈

사용자로 하여금 문서의 내용을 검색할 수 있도록 제공하는 부분으로 DOCUS 사용자 목적에 따라 기능 관점에서의 검색와 개발문서의 관점에서의 검색 그리고 이용자들의 수준을 고려한 검색으로 구분할 수있으며 기능관점에서의 검색은 문서의 inter-relationship을 통해 특정기능에 대한 각 단계에서의 서술을 검색하며 개발단계 관점에서의 검색은 각 단계의 문서의 구조를 가지고 있는 구조화일을 이용해 검색한다. 이용자들의 수준을 고려한 검색은 새로운 문서구조 생성후 문서구조의 내용에 의해 검색한다. 이러한 문서 검색으로 인해 유지 보수시에 수반되는 문서의 수정을 효율적으로 돕는다.

5. 결 론

본 연구에서는 개발단계 측면과 구현되는 요구사항에 대한 기능적 측면을 제공하고 이용자 측면의 문서들을 문서구조를 통해 자동생성하는 문서화 지원 시스템인 DOCUS를 설계하였다. EA 모델을 이용해 다중측면을 고려하여 문서 모델을 표현하였고 문서를 각 단계별로 요구사항을 만족을 위해 수행되는 여러 기능에 대한 설명의 모임이라 보고 단계별 특정 기능에 설명의 변환형식을 규정하였고 DOCUS설계를 위해 문서의 내용 수준을 고려한 문서화에 대한 형식적 서술을 제시하였다.

DOCUS의 설계로 문서작성에 있어 개발자들이 시스템 요구사항 변화로 인한 문서의 수정시 문서 전체에 대한 검색 없이도 관련있는 기능을 연결 시켜주는 문서의 inter-relationship을 이용하여 효율적인 문서의 갱신이 가능하게된다. 특히 문서의 외적 관련성인 inter-relationship과 내부적 관련성인 intra-relation을 이용하여 특정 기능을 서술하는 문서에 대한 검색과 단계별 문서에 대한 검색이 가능하다. 또한 다양한 문서를 기능, 개발단계, 이용자 측면에서의 문서 검색이 가능하다.

참고문헌

[Ber76a] Bernard P. Zeigler, Theory of Modelling and Simulation, Wiley-Interscience Pub.,1976

[Ber84b] Bernard P. Zeigler, Multifacetted Modelling and Discrete Event Simulation, Academic Press, 1984

[Dan77] Daniel Teichroew and Ernest A. Hershey III, "PSL/PSA : A Computer-Aided Technique for Structured Documentation and Analisis of Information Processing System", IEEE Trans. Software Eng, Vol, SE-3,NO.1,Jan.1977, p41-p48

[Ell86a] Ellis Horowitz and Ronald C. Williamson,"SODOS:A Software Documentation Support Environment-Its Definition",IEEE Trans. Software

Eng.,vol.SE-12,NO.8,Aug. 1986,p846-p859
[Ell86b] Ellis Horowitz and Ronald C. Williamson,"SODOS:A Software Documentation Support Environment - Its Use", IEEE Trans. Software Eng.,vol.SE-2, NO.11, Nov. 1986,p1076-p1087
[Hor88] Horst Duchene,Manfred Kaul and Volker Turau,"VODAK Kernel Data Model", Proc. the 2nd Int'l Conf. on Object-Oriented Database System, pp. 242-261,1988
[Jak90] Jakob Nielson, Hypertext and hypermedia, Academic Press,Inc.,1990
[Mar89] Marc R. D'Alleyrand, Image Storage and Retreival Systems, McGraw-Hill Book Company, 1989
[Mar87] Margaret E. Singleton, Automating Code and Documenation Management (CDM), Prentice-Hall,Inc.,1987
[Rob86] Robert N. Charette, Software Engineering Environments Concepts and Technology, McGraw-Hill,Inc. NewYork, 1986
[김89] 김창화, "EA모델에 의한 지식 표현 모델링", 고려대학교 박사학위논문,1989
[황88a] 황종선, 백두권, 소프트웨어 문서화, 교학사, 1991
[황88b] 황종선, 백두권, 소프트웨어 공학, 교학사,1989