워크플로우 기반의 XML문서의 변경저장 및 재생성모델

Workflow based storing and reconstructing model for XML documents

배혜림*, 허원창**, 강석호**

* 동의대학교 인터넷비즈니스학과

** 서울대학교 산업공학과

초록

Recent business environments require a company to communicate frequently with other companies. This makes it essential to use XML (eXtensible Markup Language) documents. Especially in e-Business environments, change management of documents is important to record and trace the history of the documents. It is also important to reconstruct a certain version of documents that are exchanged among companies. In this paper, we propose a new method of storing a document by detecting changes automatically and reconstructing the document version when a user requests. A prototype system is implemented on top of a workflow system. Our approach provides efficiency of storage space and convenient management of documents.

1. Introduction

최근에는 e-비즈니스와 같이 여러 기업이 상호작용하는 복잡한 프로세스가 증가하고 있으며, 문서의 교환과 이에 대한 작업을 포함하는 경우가 많다. 이러한 프로세스에서는 문서의 변경에 따른 완성과정을 관리하는 것이 필요하지만, 프로세스의 복잡성과 모델링 방법의 부재 등으로 제대로 지원이 이루어지지 않고 있다.

특히, 많은 정보시스템들이 웹기반으로 전이하면서 웹 환경에서 이러한 문서작업을 프로세스와 연계하여 지원할 필요성이 대두되었다. 본 연구에서는 프로세스가 진행되는 과정에서 문서의 변경을 찾아내고 효율적으로 저장하는 방법론을 제공한다. 또한 이를 바탕으로 이전 상태의 문서에 대한 재구성 요구가 있는 경우 시스템이 효과적으로 문서를 재구성하는 방법론을 제공한다.

본 논문에서는 프로세스를 기반으로 문서를 관리하는 방법론에 대한 기존 연구를 바탕으로 문서의 저장과 재구성부분에 대하여 모델을 제시하고 프로토타입시스템을 구현하였다. 그리고, 이러한 저장방법론의 우수성을 저장공간의 효율화 측면에서 기존 방법론과 비교한다.

2. Backgrounds

2.1 비즈니스 문서와 XML

비즈니스 환경에서는 많은 데이터와 정보가 문서의 형태로 처리가 되며, 이러한 문서는 정보처리의 효율을 높이기 위하여 정형화된 형태로 구성된다. 이를 양식문서(form document)라고 하며, 1980년대부터 이의 처리에 대한 연구가 꾸준히 이루어져 왔다[4][6].

이러한 양식문서의 완성과정은 비즈니스 프로세스와 밀접한 관련을 가지게 되며, 프로세스의 진행과정에 따라 변경을 거듭하게 된다. 따라서, 문서가 중심이 된 프로세스에서는 문서의 완성과정을 프로세스의 수행과정으로 볼 수 있다.

본 논문에서는 XML(eXtensible Markup Language)을 이용하여 비즈니스 문서를 구현한다. XML기술은 SGML(Standard Generalized Markup Language)의 복잡성을 극복하면서 HTML (HyperText Markup Language)의 문제점을 해결한 차세대 문서표준이다. 특히, XML은 문서의 내용과 표현을 분리함으로써, 양식문서로 사용하기에 적합하다. 즉, 비즈니스 문서의 내용부분의 변경에만 집중하고 표현 부분은 XSL(eXtensible Styling Language)과 같은 언어를 사용하여 자유롭게 표현할 수 있음을 전제로 기술한다.

2.2 워크플로우에서의 문서관리

워크플로우 관리 시스템은 비즈니스 프로세스를 컴퓨터가 실행 가능한 형태로 정의하고 이의 제어와 관리를 도와주는 시스템으로[5][7], 최근 십여년동안 활발한 연구들이 진행되어 왔다. 워크플로우 시스템이 프로세스를 자동으로 실행하는 데는 프로세스에서 사용되는 자원의 관리가 필수적으로 수반되며 가장 대표적인 것이 문서이다. 대부분의 워크플로우 관리 시스템은 문서관리를 어느정도는 지원하고 있으나 이를 프로세스의 의미와 연계하는 체계적인 모델을 제시하지 않는다.

워크플로우 프로세스에서 사용되는 문서를 효과적으로 관리하기 위해서는 다음 그림 1과 같은 접근법이 필요하다.

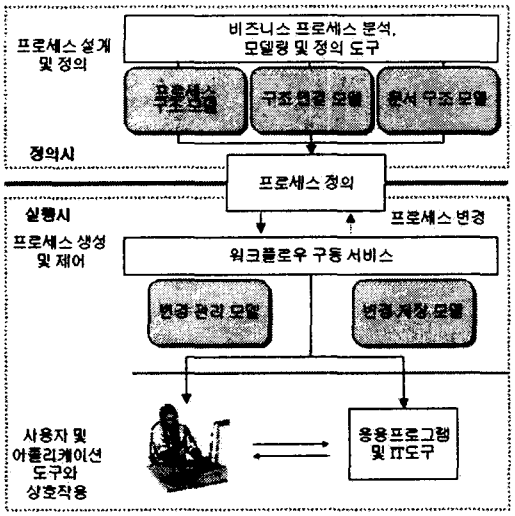

그림 1. 워크플로우에서의 양식문서 관리

위의 그림의 세부 모델 중에서 본 논문은 프로세스상에서 문서의 변경을 저장하고 추출 재구성하는 것과 관련된 변경 저장 모델만을 다루며 나머지 모델은 [1], [2], [3]을 참고하기 바란다.

3. 프로세스 및 문서 관련 정의들

프로세스 기반으로 문서를 관리하기 위하여 프로세스와 문서를 구조적으로 표현하는 것이 필요하며, 먼저 프로세스와 작업단위에 대한 정의를 다음에 제시하였다. 이에 대한 더 자세한 사항은 [1], [2]을 참고하기 바란다.

정의 1 (프로세스 구조)

프로세스는 다음과 같은 유방향 비순환 그래프인 G=(A, L) 로 구성된다.

- $A = \{a_i \mid i = 1, \ldots, N\}$ 는 단위업무 집합이다. (a_i 는 i-번째 단위업무, N 은 G에 속한 단위업무의 전체 개수)
- $L \subseteq \{(a_i, a_j) \mid a_i, a_j \in A \text{ and } i \neq j\}$ 는 링크의 집합이다. ((a_i, a_j) 는 a_i 가 a_j에 대하여 바로 선행하는 링크)

다음으로 문서의 구조는 다음과 같이 정의된다.

정의 2 (문서구조)

문서 d 는 workunit들의 집합으로 정의되며, 각 workunit은 다음과 같은 데이터 필드로 구성된다.

- $d = \{w_m \mid m = 1, \ldots, M\}$, ($w_m$ 은 m-번째 작업단위, M 은 d에 포함된 작업단위의 개수)
- $w_m = \{f_l \mid l = 1, \ldots, L_m\}$, ($f_l$ 은 l-번째 필드, L_m 은 w_m에 포함된 필드의 수)

위의 정의에서 $w_m \cap w_n = \varnothing$ $(m \neq n)$이고, 하나의 데이터필드 f_l 는 이름, 값의 쌍인 (field name, value)로 정의된다.

본 연구에서 문서나 프로세스에서 사용되는 객체는 흔히 사용되는 .을 이용하여 표현한다. 예를 들면 프로세스 p의 j번째 단위업무 a_j는 $p.a_j$ 그리고 문서 d_i의 작업영역 w_j의 필드 f_k는 $d_i.w_j.f_k$와 같이 표기한다.

4. 변경의 저장과 문서의 재구성

본 연구에서는 변경의 저장을 효과적으로 하기 위하여 문서의 변경분을 쉽게 감지하여 변경부분만 저장하고 후에 필요한 문서를 구성함으로써 저장공간을 효과적으로 줄일 수 있다.

4.1 변경의 기록

프로세스가 수행되는 동안 사용자에 의하여 문서의 변경이 발생하면 이를 저장하여야 한다. 문서에 대한 수정은 필드 단위로 저장할 수 있으며, 하나의 필드에 행해지는 변경은 다음과 같은 기본적인 네 가지 유형을 바탕으로 발생한다.

- 추가변경($\square_{ADD}$), ADD (f_l) (*value*): 필드 f_l에 *value*값을 채워졌음을 의미
- 삭제변경($\square_{DEL}$), DEL (f_l) (*value*): 필드 f_l의 *value*값이 지워졌음을 의미
- 수정변경($\square_{MOD}$), MOD (f_l) (*value1*) (*value2*): 필드 f_2의 값을 *value1*에서 *value2*로 변경
- 빈변경($\square_{\varnothing}$): 아무런 변경도 일어나지

않았음을 가리키는 단위변경

정의 3 (문서변경)

문서변경은 하나의 작업단위에 대한 변경의 기록으로, 해당 작업단위를 구성하는 필드들의 변경으로 구성한 집합이다. m-번째 작업단위에 대한 버전 변경 δ_m은 다음과 같이 구성된다.

$$\delta_m = \{\square_l \mid l = 1, 2, \ldots, L\}$$

(L은 변경의 대상이 되는 작업단위에 속한 필드의 수)

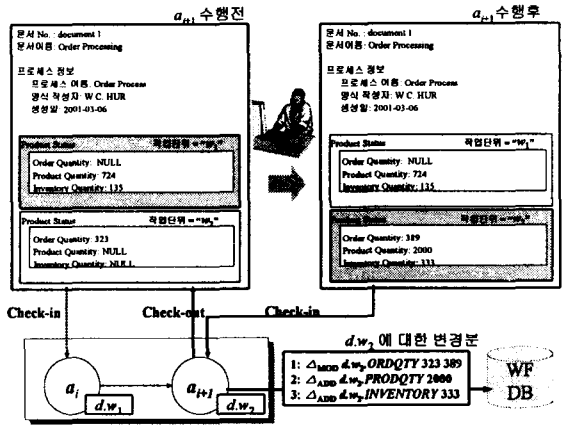

그림 2. 단위업무 수행과정

문서의 변경은 그림 2에서 보는 바와 같이 문서를 체크인할 때 작업단위에 대하여 시스템에 의해 자동으로 추출되어 변경 파일에 저장된다. 예를 들면, ADD $d_1.w_1.f_1$ "Yes"의 오퍼레이션은 필드 $d_1.w_1.f_1$에 "Yes"라는 값이 새로 채워졌음을 의미한다.

정의 4 (문서 버전과 버전 그래프)

만일 o 를 하나의 객체라고 하면, $v_p(o)$는 객체 o 의 p-번째 버전을 나타낸다. 하나의 객체에 대한 버전 그래프 (*Version Graph, VG*)는 비순환 유방향 그래프 다음과 같이 $VG = (V, E)$로 정의된다.

- $V = \{ v_p(o) \mid p = 1, 2, \ldots, P \}$
- $E = \{ (v_p(o), v_q(o)) \mid v_p(o) \in V, v_q(o) \in V, p \neq q,$ and $v_q(o) = \delta(v_p(o)) \}$

여기서 δ 는 변경 함수이고 $v_q(o) = \delta(v_p(o))$는 o의 q-번째 버전은 o의 p-번째 버전에 대한 수정으로 발생한 것임을 나타낸다.

4.2 단위 변경간의 연산

4.1절에서 정의한 문서 변경에 대하여 이들간의 연산을 정의한다. 먼저 역변경을 정의한다.

역변경: 변경 □에 대한 역변경은 $\square^{-1}$로 표현하며 각 변경에 대한 역변경은 다음과 같이 정의된다.

- 추가변경의 역변경: $\square_{ADD}^{-1} = \square_{DEL}$
- 삭제변경의 역변경: $\square_{DEL}^{-1} = \square_{ADD}$
- 수정변경의 역변경: $\square_{MOD}^{-1} = \square_{MOD}'$

변경결합: 각 필드는 프로세스가 수행되는 동안 반복적인 변경을 거치게 되며 하나의 필드에 대해서 이루어지는 순차적인 변경의 결과를 반영하기 위하여 변경 간에 결합 연산을 정의한다. 이는 추후 문서의 재구성을 빠르게 수행할 수 있도록 하기 위함이다. 같은 필드를 대상으로 하는 두 단위변경□, □'사이의 결합 연산은 □ ∘□' 과 같이 표현하며 이는 대상 필드에 대하여 □ 변경이 있은 후에 □' 변경이 추가로 이루어졌음을 의미한다. 각 타입별로 가능한 단위변경간 결합 연산은 다음과 같이 정의된다.

- **Case 1:** $\square_{ADD} \circ \square_{ADD}' = \square_{ADD}'$
- **Case 2:** $\square_{ADD} \circ \square_{DEL} = \square_{\emptyset}$
- **Case 3:** $\square_{ADD} \circ \square_{MOD} = \square_{ADD}'$
- **Case 4:** $\square_{DEL} \circ \square_{ADD} = \square_{MOD}$
- **Case 5:** $\square_{MOD} \circ \square_{MOD}' = \square_{MOD}''$
- **Case 6:** $\square_{MOD} \circ \square_{DEL} = \square_{DEL}'$
- **Case 7:** $\square_{MOD} \circ \square_{ADD} = \square_{ADD}$

위에서 제시한 각 연산들을 적용하는데 있어서 다음과 같은 법칙들이 적용된다.

- $\square_l \circ \square_{\emptyset} = \square_{\emptyset} \circ \square_l = \square_l$
- $(\square_l \circ \square_l')^{-1} = \square_l'^{-1} \circ \square_l^{-1}$
- $\square_l \circ \square_l^{-1} = \square_l^{-1} \circ \square_l = \square_{\emptyset}$
- $\square_l \circ \square_l' \neq \square_l' \circ \square_l$
- $(\square_l \circ \square_l') \circ \square_l'' = \square_l \circ (\square_l' \circ \square_l'')$

4.3 문서의 추출

문서의 저장 형태는 최초의 문서 정의와 이에 대한 변경분들로 이루어져있으므로 단위업무에서 문서에 대한 체크아웃을 요청하거나 문서의 모니터링 등의 이유로 문서를 클라이언트로 보내야 하는 경우는 문서정의로부터 변경분에 저장된 연산들을 순차적으로 적용하여 문서를 구성하게 된다.

프로세스가 진행이 되는 동안 [3]에서 설명한 대로 문서와 작업영역의 버전이 자동으로 생성이 되며 이들은 변경분만을 저장하고 있다. 그림 3에서 만일 사용자가 a_8이 이루어진 직후의 문서에 대한 요구를 한 경우 시스템은 적절한 문서를 구성하여 반환해주어야 한다.

정의 6 (변경의 결합)

서로 다른 다수의 변경은 버전 재생성의 효율을 위하여 결합된 하나의 변경으로 표현할 수 있으며 이는 다음과 같이 표현된다.

$$\delta = \delta_i \circ \delta_j = \{ \square_{il} \circ \square_{jl} \mid l = 1, \ldots L \}$$

($\delta_i = \{\square_{i1}, \square_{i2}, \ldots, \square_{il}, \ldots, \square_{iL}\}$, $\delta_j = \{\square_{j1}, \square_{j2}, \ldots, \square_{jl}, \ldots, \square_{jL}\}$, L: 해당 작업단위에 소속된 필드의 수)

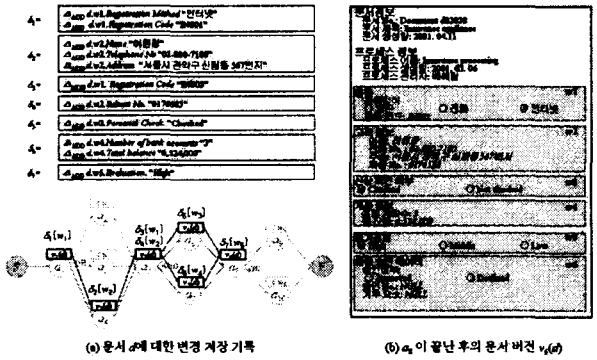

그림 3. 문서 재구성 과정

변경의 반복적인 적용은 프로세스의 특정 시점에서의 문서의 상태를 결정하게 되며, 프로세스가 수행되는 도중, 혹은 프로세스 완료 후에 특정 시점의 문서를 구성할 필요가 있다. 예를 들면, a_8가 작업을 끝내 시점의 문서 d는 v_6가 되며, 변경간의 결합을 통하여 다음과 같이 구하여 진다.

$$\begin{aligned} v_6(d) &= (\delta_1 \circ \delta_2 \circ \delta_3 \circ \delta_4 \circ \delta_5 \circ \delta_7) v_0(d) \\ &= (\delta_1 \circ \delta_2 \circ \delta_3 \circ \delta_4 \circ \delta_6 \circ \delta_7) v_0(\mathrm{d}) \\ &= \delta(v_0(d)) \end{aligned}$$

7. Experimental Results

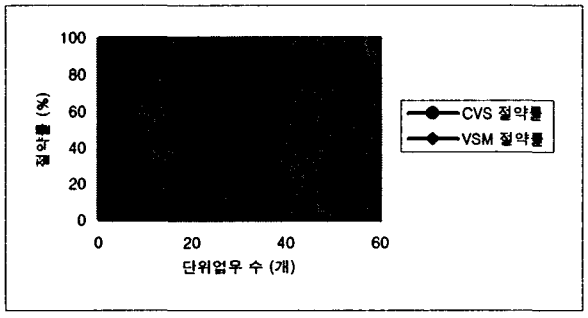

그림 4. 기존 방법론과의 비교

본 연구에서 사용한 방법론을 이용하여 문서를 저장할 때, 저장 공간의 절약을 간단한 실험을 통해 기존 방법론과 비교하였다. 문서의 변경을 변경분으로 저장하는 방법론 중 가장 일반적인 CVS(Concurrent Versions System)와 비교할 때, 더 좋은 결과를 나타내었다.

8. Conclusions

본 논문에서는 비즈니스 프로세스에서 XML양식 문서가 사용되는 경우 프로세스의 진행과정을 기반으로 문서의 변경을 자동으로 저장하고 문서를 재구성하는 모델을 제시하였다. 본 논문에서 제시하는 방법론은 사용자가 편리하게 문서작업을 하도록 함과 동시에 문서의 저장공간을 효율적으로 사용할 수 있다.

참고문헌

[1] 배혜림, "e-Business문서관리: 워크플로우 프로세스를 위한 비즈니스 문서 변경관리", 서울대학교 공학박사 학위논문, 2002.

[2] H. Bae and Y. Kim, "A document-process association model for workflow management", *Computers in Industry*, vol. 47, no. 2, pp 139-154, 2002.

[3] H. Bae, W. Hur, W. S. Yoo, B. Kawk, Y. Kim, and Y. Park, "Document Configuration Control Processes Captured in a Workflow", *Computers in Industry*, to appear soon.

[4] S. B. Yao, A. R. Hevner, Z. Shi, and D. Luo, "FORMANAGER: An Office Forms Management System", ACM Transactions on Office Information Systems, vol. 2, no. 3, pp. 235-262, 1984.

[5] D. Georgakopoulos, M. Hornick and A. Sheth, "An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure," *Distributed and Parallel Databases*, vol. 3, no. 2, pp. 119-154, 1995.

[6] D. Tsichritzis, "Form Management", *Communications of the ACM*, vol 25, no 7, pp 453-478, 1982.

[7] D. Hollingsworth, "Workflow management coalition specification: the workflow reference model", WfMC specification, 1994.