卒業論文 1999年度(平成11年度)

WWWドキュメント更新APIの構築

指導教員
慶應義塾大学環境情報学部
徳田　英幸
村井　純
楠本　博之
中村　修

慶應義塾大学　環境情報学部
中村 勇樹

卒業論文要旨 1999年度(平成11年度)

WWWドキュメント更新APIの構築

本研究は，WWWブラウザ上のHTMLドキュメント/XMLドキュメント(以下WWWドキュメント)をより柔軟に更新するフレームワークの提案を第1の目的とする．また，同フレームワークを特定アプリケーションに依存しないで汎用的に使用するためのAPIを定義することを第2の目的とする．

WWWブラウザには表示したWWWドキュメントを更新する機能がある．この機能はWWWブラウザの独自拡張であり，標準化はなされていない．例えば，「特定の領域にポインタを重ねてからマウスをクリックする」といったユーザからの入力に応じてあらかじめWWWドキュメントに埋め込んでおいたプログラムを実行する．このプログラムは「文字の色を変える」などWWWドキュメントを更新する指示をWWWブラウザに送り，最終的にWWWドキュメントを更新する．

従来のWWWブラウザが持つWWWドキュメントを更新する機能には2つの制約がある．第1点は，WWWドキュメントとWWWドキュメントを操作するプログラムを分離できないものとして扱っているため，対象のWWWドキュメントを発行したあとからWWWドキュメントを操作するプログラムを実行させることができない．第2点は，WWWドキュメントを操作するプログラムが保有するデータを一時的なデータとして扱っているため，他のWWWドキュメントを閲覧する際や次にWWWブラウザを起動した際にこのデータを活用できない．

本研究では上記の制約を解消し，より一般的にWWWドキュメントを更新するためにプログラムに提供するAPI(以下WWWドキュメント更新API)を提案する．WWWドキュメント更新APIはWWWドキュメントとWWWドキュメントを操作するプログラムの間に関連付けを作成し，この関連付けを永続化する仕様を定義する．WWWドキュメント更新APIを実装したライブラリを利用することで，アプリケーションはWWWドキュメントを柔軟に更新することができる．さらに本研究では，WWWドキュメント更新APIを実装したライブラリと，ライブラリを構築しWWWドキュメント更新APIを試用するためのサンプルアプリケーションを設計・実装する．

本論文では，まず従来のWWWブラウザが対象のWWWドキュメントを更新する手順の問題点を指摘し，より柔軟に対象のWWWドキュメントを更新するAPIとしてWWWドキュメント更新APIを提案する．次に，同APIを利用するための手順を分析し，同APIを実装する．また，同APIを試用するためのアプリケーションを設計，実装する．

中村 勇樹

慶應義塾大学 環境情報学部

Abstract of Bachelor's Thesis
API for Updating WWW Document

In this research, the main object is the proposal of a framework for flexible renewal of HTML and XML documents on the WWW browser. The second object is to structure API so that this framework can be used widely.

The WWW browser has a function unique to this system to renew the document and it is not standardized. For example, in response to an input such as when the mouse is pointed to a certain area and clicked, a program that is originally planted into the WWW document is executed. The program the sends out a command to change the color of the text to the WWW browser, and in turn, the WWW document is renewed.

Existing WWW browsers have two limitations to the function of renewing WWW documents. First of all, the WWW document and the program that controls the document can not be distinguished. After the target WWW document is published, the program can not be controlled. Second, the data held by the WWW document program is only temporary, so it is impossible for other WWW documents to refer to this data or use it when starting up.

This research proposes a solution to these limitations, a more general API for both WWW document renewal program and WWW document renewal. The WWW document renewal API creates a relation between the WWW document and WWW document renewal program, and with this, a more premenant specificaion is defined. Here, a library of WWW document renewal API is implemented, a sample application for structuring the library and testing the WWW document renewal API is designed and implemented. Using the implemented WWW document renewal API, the application can alter the WWW document flexibly.

In this paper, first, problems concerning the procedure taken by existing WWW browser whe renewing WWW documents are given, a more flexible WWW document renewal API is proposed. Also, with this API, the procedures for using this API is analized and implemented, a application that tests this API is designed and implemented.

Yuuki Nakamura

Faculty of Environmental Information

Keio University

目次

第1章　序論 8
- 1.1 本研究の背景 . . . 8
- 1.2 本研究の目的 . . . 9
  - 1.2.1 事例 . . . 10
- 1.3 基本的な用語の定義 . . . 11
- 1.4 本論文の構成 . . . 11

第2章　WWWドキュメント更新APIの設計 12
- 2.1 概要 . . . 12
- 2.2 事例アプリケーション . . . 12
  - 2.2.1 事例アプリケーションの基本要件 . . . 12
  - 2.2.2 事例アプリケーションの機能分析 . . . 12
- 2.3 基本要件 . . . 13
- 2.4 設計方針 . . . 13
  - 2.4.1 コンテナ独立性 . . . 14
  - 2.4.2 オブジェクト永続性 . . . 14
- 2.5 WWWドキュメント更新APIの概要 . . . 14
- 2.6 全体構成 . . . 14
- 2.7 動作手順 . . . 16
  - 2.7.1 ドキュメント，ドキュメントオブザーバを永続化する手順 . 16
  - 2.7.2 ドキュメントにアクセスする手順 . . . 17

第3章　WWWドキュメント更新APIの実装 19
- 3.1 概要 . . . 19
- 3.2 実装環境 . . . 19
- 3.3 実装方針 . . . 19
  - 3.3.1 拡張可能シーケンス . . . 19
- 3.4 クラス定義概要 . . . 20
  - 3.4.1 EnvelopePresentationクラス . . . 20
  - 3.4.2 EnvelopeManagerクラス . . . 20
  - 3.4.3 EnvelopeStoreクラス . . . 20

3.4.4 Document クラス . . . 20
3.4.5 Envelope クラス . . . 20
3.5 主な相互作用 . . . 21
3.5.1 ドキュメントまたはドキュメントオブザーバを永続化する手順 21
3.5.2 ドキュメントにアクセスする手順 . . . 21

第4章 WWW ドキュメント更新システムの設計 24
4.1 概要 . . . 24
4.2 基本要件 . . . 24
4.3 設計方針 . . . 24
4.4 WWW ドキュメント更新システムの概要 . . . 25
4.5 全体構成 . . . 25
4.5.1 アプリケーション . . . 25

第5章 WWW ドキュメント更新システムの実装 26
5.1 概要 . . . 26
5.2 実装環境 . . . 26
5.3 Presentation の実装 . . . 26
5.3.1 Presentation の実装詳細 . . . 26
5.3.2 実装に使用したライブラリ . . . 26
5.4 Manager の実装 . . . 27
5.4.1 Manager の実装詳細 . . . 27
5.4.2 実装に使用したライブラリ . . . 27
5.5 Store の実装 . . . 28
5.5.1 Store の実装詳細 . . . 28
5.5.2 実装に使用したライブラリ . . . 28
5.6 WWW ドキュメント更新システムの実装 . . . 28

第6章 おわりに 29
6.1 まとめ . . . 29
6.2 今後の課題 . . . 29
6.2.1 手順パターンを増やす . . . 29
6.2.2 同ライブラリを活用するアプリケーションを構築する . . . 30

図 目 次

1.1 WWW ブラウザ上で動作するプログラム 9

2.1 ユースケース 15
2.2 全体構成 16
2.3 ドキュメントまたはドキュメントオブザーバを永続化する手順 . . . 17
2.4 ドキュメントにアクセスする手順 18

3.1 ドキュメントまたはドキュメントオブザーバを永続化する手順 . . . 22
3.2 ドキュメントにアクセスする手順 23

5.1 EnvelopePresentation クラスと EnvelopeRequestHandlerFactory クラス 27
5.2 EnvelopeManager クラスと EnvelopeStore クラス 28

表 目 次

3.1 実装環境 19

5.1 実装環境 26
5.2 Presentationの実装に使用したライブラリ 27
5.3 Managerの実装に使用したライブラリ 27
5.4 Storeの実装に使用したライブラリ 28
5.5 実装状況 28

第1章　序論

1.1　本研究の背景

ユーザがWWWブラウザを使用してページを閲覧する手順は以下のように進む．図1.1にWWWブラウザ，ドキュメントとドキュメントオブザーバの関係を示す

1. ユーザはWWWブラウザに閲覧したいページのURLを入力する．
2. WWWブラウザはユーザが指定したURLに対応するファイルをダウンロードする．
3. ダウンロードしたファイルがドキュメントである場合，WWWブラウザはドキュメントの構造と内容をウィンドウに表示可能な形式にレンダリングし，ウィンドウに表示する．
4. ドキュメント内にプログラムを用いるという指示文が記述されている場合，例えば「マウスポインタが特定のアンカー領域の外側から内側に侵入した(JavaScriptで言えば，onMouseOverイベント)」などのイベントが発生した際，WWWブラウザは該当するプログラムを実行する．この結果，ドキュメントの構造と内容が変更された場合は，変更が生じた箇所を再描画する．
5. WWWブラウザはドキュメントへの変更情報を新たに起動した時まで保持しない．

この手順には以下に示す制約がある．

1. XML [1]はドキュメント構造とドキュメント体裁を分離する仕様は定めているが，ドキュメント構造・ドキュメント体裁とドキュメントオブザーバを分離し，柔軟に運用する仕様は定めていない．よって，対象のドキュメントが指定したプログラム以外のプログラムは対象のドキュメントを操作できない．
2. ドキュメントオブザーバが保有するデータは対象のドキュメントをWWWブラウザで閲覧している間だけ有効である．他のドキュメントを閲覧する際や次にWWWブラウザを起動した際に，このデータを活用できない．

例えば，ページを閲覧する手順中にドキュメントを更新する処理を実現したいとする．検索エンジンSaでキーワードを入力してページを捜しているという状況がある．ユーザが検索エンジンSaからハイパーリンクを辿ってあるページPaを

---

[1] XML(Extensible Markup Language : 拡張可能なマーク付け言語)とは，Internet上でのSGMLの利用を容易にすることを目的として設計されたマークアップ言語である[1]．

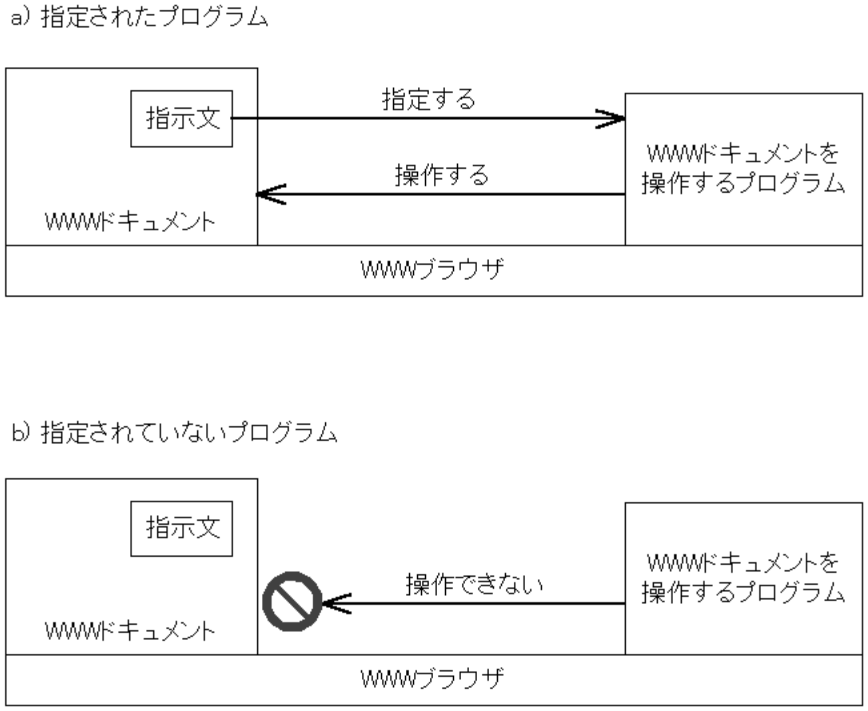

図 1.1: WWW ブラウザ上で動作するプログラム

訪れた際，そのページは別のURLに引っ越しをしていた．その後，ユーザは検索エンジンSaの代わりに別の検索エンジンSbでキーワードを入力し，検索エンジンSbからハイパーリンクを辿ってあるページを訪れたところ，そのページは先のページPaと同じサイトにあるページであり，ページPaと同様に引っ越しをしていた．このとき，たとえSaのセッションとSbのセッションが分離していたとしても，検索エンジンSbでのハイパーリンクが引っ越し先に接続するよう自動的に修正するという要件を実現したい.

従来のWWWブラウザでドキュメントを閲覧する場合，対象のドキュメントが指定したプログラム以外のプログラムが，対象のドキュメントを操作できないので，上記のような機能を実現できない．

1.2 本研究の目的

本研究は，検索エンジンのハイパーリンクに限らず一般的にページを修正する機構の構築を目指す．

1.2.1　事例

バージョン管理　アプリケーションがコンテキストAにおいてバージョン1のドキュメントを閲覧した．次に，アプリケーションが別のコンテキストBにおいてバージョン2のドキュメントを閲覧した．このとき，以下のうちいずれかの処理を行う．

- 古いバージョンを削除し，新しいバージョンを残す．
- 古いバージョンから新しいバージョンへのハイパーリンクを設置する．
- 新しいバージョンから古いバージョンへのハイパーリンクを設置する．
- 古いバージョンの直後に新しいバージョンを連結して，一体にする．
- 古いバージョンの直前に新しいバージョンを連結して，一体にする．
- 古いバージョンを表示する再に新しいバージョンのサムネイルも表示する．
- 新しいバージョンを表示する再に古いバージョンのサムネイルも表示する．
- 両バージョンをマージして，一体にする．

自己消滅　何らかのイベントを受け取ったときにドキュメントを自己消滅させる．以下のような消滅させる理由，条件がある．

- 催し物の終了期間後は催し物を告知するドキュメントは不要となる．このような種類のドキュメント指定時間経過すると消滅させる，ないし指定時刻を経過すると消滅させる．ここでのイベントは「時間」または「時刻」である．
- 定員制限と出欠．催し物の定員を出席希望者が越えたとき，参加受付を停止する場合，ドキュメントを消滅させる．ここでのイベントは「参加受付停止を知らせるドキュメント」である．

依存関係　「ドキュメントAを更新したら常にドキュメントBも更新しなければならない，さもないとドキュメントAとドキュメントBの整合性が崩れる」のような条件付けを課す．

- ある科目の教育カリキュラム文書が更新されたとき，古いバージョンのカリキュラムに準拠した教科書は「カリキュラム準拠」資格を失う．
- あるソフトウェアSのリリース記事A1が公開された．別の記事B1ではソフトウェアSの最新リリースを参照する意図があり，A1へのハイパーリンクを付加して公開された．その後，ソフトウェアSの新しいリリースが公開されてリリース記事A2が公開された際に，記事B1にA2へのハイパーリンクを付加する．

1.3　基本的な用語の定義

議論の混乱を避けるため，本論文において用いられる用語の定義を以下に示す．

ドキュメント　ドキュメントとは，HTML ドキュメントおよび XML ドキュメントである．

1.4　本論文の構成

2 章では，WWW ドキュメント更新 API の基本要件を踏まえてから，同 API の主要な手順を分析する．

3 章では，WWW ドキュメント更新 API の Java 言語への実装について述べ，同 API が駆動する際の主要な手順を定める．

前章までの API を試用するために，本研究ではサンプルアプリケーション (WWW ドキュメント更新システム) を設計・実装した．4 章では，WWW ドキュメント更新システムの基本要件について述べる．

5 章では，WWW ドキュメント更新システムの実装について述べる．

6 章では，本研究のまとめと今後の課題について述べる．

第2章　WWWドキュメント更新APIの設計

2.1　概要

本章では，WWWドキュメント更新APIの設計について詳細に述べる．まず，WWWドキュメント更新APIの基本要件，設計方針，全体構成，動作手順について述べる．次に，主要パッケージであるPresentation，Manager，Storeの詳細な設計を行う．

2.2　事例アプリケーション

本章では以下のアプリケーションを想定してWWWドキュメント更新APIを設計する．このアプリケーションはWWWドキュメント更新APIを利用する．

2.2.1　事例アプリケーションの基本要件

タイトル　(仮称)サイトの引っ越し情報をハイパーリンクへ自動的に反映させるユーティリティ

基本要件

- 本ユーティリティの第1の目的はHTMLドキュメントとXMLドキュメントにおけるハイパーリンクのリンク先URLが不正であることを自動的に検出し，リンク先URLを自動的に修正することである．
- URLが不正であるか否かの情報は通常のページ閲覧の最中に行い，この情報を特定のセッションやWWWブラウザのプロセスに依存することなく永続化する．

2.2.2　事例アプリケーションの機能分析

本ユーティリティは以下に示す機能を持つ．

機能1 ページからハイパーリンクを通じて辿ったページの状態を判定する．これはhttpのレスポンスコードが400番台であるかどうかで知ることができる．

機能2 ページの状態とURLの記録を永続化する．

機能3 ユーザがページを閲覧する際に，ページ内にあるハイパーリンクのリンク先URLと永続化しておいた記録にあるURLを照合する．URLが一致してページ状態が正しいページでない場合，ページからこのハイパーリンクを除外する．URLが一致してページが別のURLへ引っ越していた場合，ハイパーリンクのリンク先URLを引っ越し後のURLへ修正する．

2.3 基本要件

WWWドキュメント更新APIとは，ドキュメントにドキュメントオブザーバを動的に関連付けて，必要なときに実行するアプリケーションのためのAPIである．WWWドキュメント更新APIが提供すべき主要な機能を以下に示す．

- ドキュメントとドキュメントオブザーバを動的に関連付ける機能
  - 対象のドキュメントと対象のドキュメントオブザーバを，対象のドキュメントを永続化するタイミングと対象のドキュメントを閲覧するタイミングの間に関連付けする．
  - 他のコンテキストや新たなWWWブラウザを起動する際も関連付けを再利用できるように，関連付けを永続化する．
  - 関連付けを永続化することにより，上記事例アプリケーションでの機能2と機能3を実現できる．
- ドキュメントを処理する際に，関連付けしたドキュメントオブザーバを実行する機能
  - ドキュメントを閲覧するなどの処理を行う際に，対象のドキュメントに関連付けをしたドキュメントオブザーバを全て列挙して実行する．
  - 閲覧中にドキュメントオブザーバを実行するにより，上記事例アプリケーションでの機能3を実現できる．

2.4 設計方針

上記の基本要件を満たし，特定のアプリケーションに依存しないAPIを実現するために，本APIの設計方針としてコンテナ独立性，オブジェクト永続性を挙げる．

2.4.1 コンテナ独立性

従来のWWWブラウザでは，ドキュメントオブザーバはコンテナ(そのプログラムを埋め込んだドキュメント)を操作できるが，他のドキュメントを操作できない．WWWドキュメント更新APIでは，ドキュメントオブザーバとドキュメントを分離して扱う．例えば，ドキュメントの最初のバージョンをhttpでダウンロードした後，最初のバージョンに対する差分修正を行うドキュメントを電子メールで受け取った場合，この2つのドキュメントをマージすることができる．

2.4.2 オブジェクト永続性

従来のWWWブラウザ上で実行するプログラムは特定のセッション内でのみ有効である．すなわち，WWWブラウザはセッション終了時にプログラムが保有するオブジェクトを全て破棄して，別のセッションではオブジェクトを持ち越さずに最初からやり直す．WWWドキュメント更新APIでは，ドキュメントオブザーバを関連付けするコンテキストと，実行するコンテキストが同一でないケースも想定するため，オブジェクトを永続化する．

例えば，ユーザがWWWブラウザを使ってサーチエンジンAにキーワードを入力して検索結果ページA1を得た．検索結果ページA1からハイパーリンクを辿ると，キーワードに合致するページへのリンクだけでなく，リンク先のURLが無効になっているリンクもあった．後日，ユーザがWWWブラウザを使ってサーチエンジンBにキーワードを入力して検索結果ページB1を得た．このとき，検索結果ページA1からハイパーリンクを辿る際に得た関連付けを元に，検索結果ページB1からリンク先のURLが無効になっているリンクを除外することができる．

2.5 WWWドキュメント更新APIの概要

WWWドキュメント更新APIの基本要件を図2.1に示す．文法はUMLユースケースダイアグラムである．アプリケーション(例えば，WWWブラウザ)はユーザからの「指定するドキュメントをウィンドウに描画させる」などの指示に応答するために，WWWドキュメント更新APIを通じてドキュメントを取得する．ドキュメントオブザーバは任意のアプリケーションがドキュメントを取得するイベント発生時に駆動する．

2.6 全体構成

従来のWWWブラウザでは一塊となっていた処理をPresentation，Manager，Storeという3つのオブジェクトに分割する．これによりPresentationとStoreを

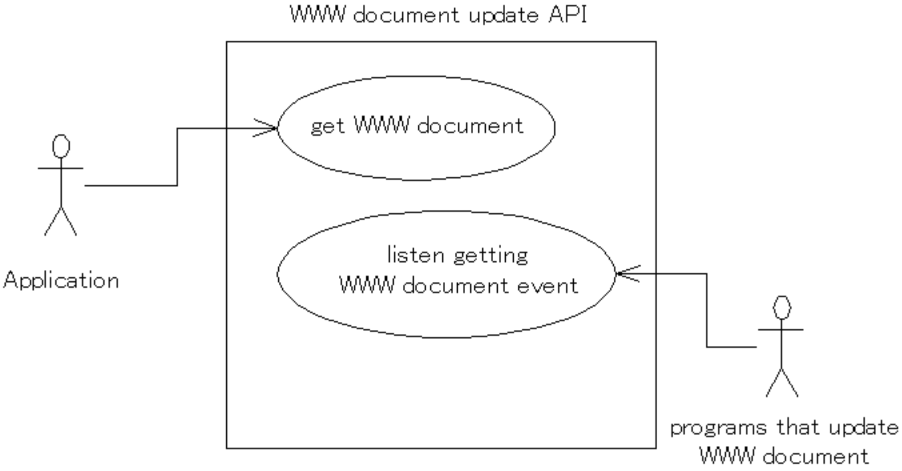

図 2.1: ユースケース

変更したり，複数のPresentationや複数のStoreを用いるようにプログラムを構成しやすくなる．それぞれ以下に示す役割を持つ．図2.2に全体構成を示す．文法はUML コラボレーションダイアグラムである． 全体を大まかに図解するため，主な相互作用を併記し，番号付けは外した．メッセージの詳細は後述する

- Presentation
  - 外部とのインタフェースを司る．例えば，ユーザの操作をManagerに伝達する，ドキュメントをユーザが指定したウィンドウに描画する，などの機能を担当する．
- Manager
  - ドキュメントとドキュメントオブザーバを処理する．例えば，Storeにドキュメントを追加する際にドキュメントオブザーバとの関連付けを作成する，ユーザがドキュメントを閲覧しようとする際に，実行するドキュメントオブザーバがどれであるかを判定する，などの機能を担当する．
- Store
  - ドキュメントとドキュメントオブザーバを永続化する処理を担当する．

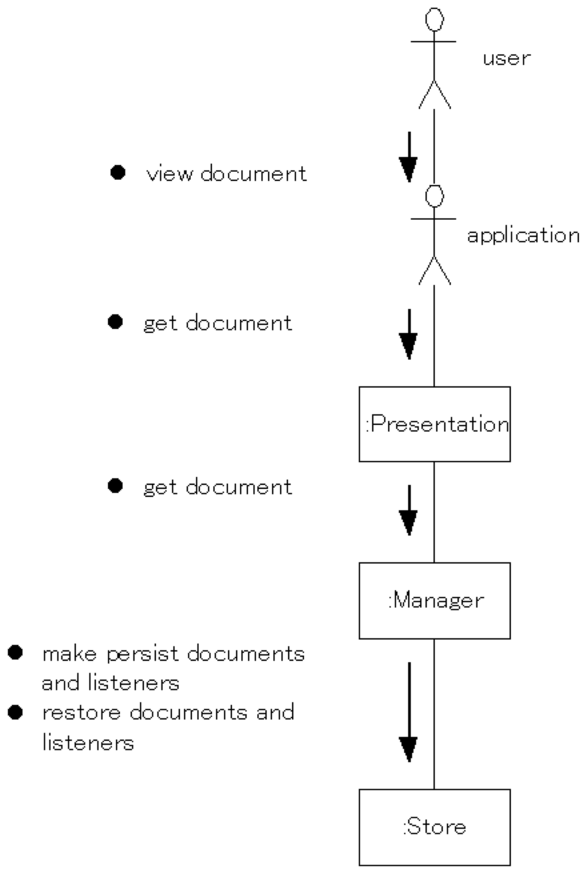

図 2.2: 全体構成

2.7 動作手順

2.7.1 ドキュメント，ドキュメントオブザーバを永続化する手順

図2.3 にアプリケーションがPresentaion，Manager，Storeを通じてドキュメントまたはドキュメントオブザーバを永続化する手順を示す．文法はUML シーケンスダイアグラムである．説明のためメッセージ名は擬似コードで記述した．

ユーザがアプリケーションにドキュメントを永続化するよう指示すると，この手順が開始する．Presentationはアプリケーションと Managerの対話を仲介する．Managerは Storeを通じてリスナを復元する．リスナとは，ドキュメントに関係するイベントの通知を受け取るオブジェクトのことである．ドキュメントオブザーバはリスナとしてのロールがある．Managerはイベントの通知を希望するリスナに対して，それぞれ関連付けを作成する．関連付けを作成した後，ManagerはStoreを通じてドキュメントと関連付けを永続化する．

本API を実装したライブラリを利用してアプリケーションを構築するプログラマは，Evnelopeクラスのサブクラスを作成し，askListenメソッドをオーバーライドする．AddEnvelopeRequestHandlerはaskListenメソッドを呼び出し，返り値が

TRUE であればオブジェクトを永続化する．

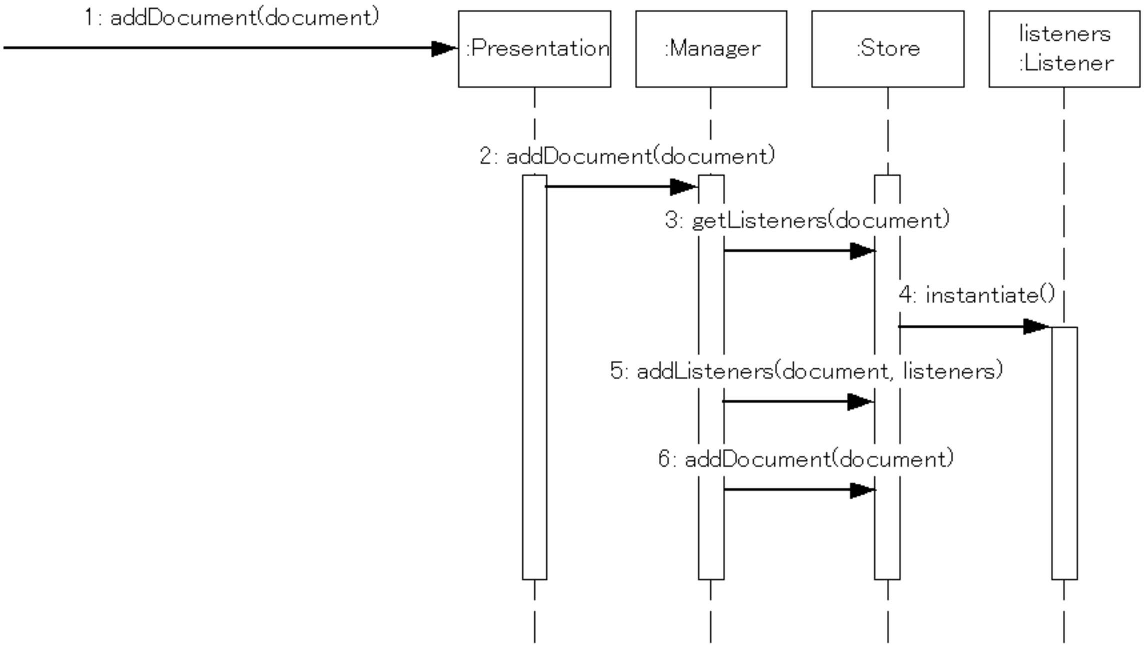

図 2.3: ドキュメントまたはドキュメントオブザーバを永続化する手順

2.7.2 ドキュメントにアクセスする手順

図 2.4 にアプリケーションが Presentaion，Manager，Store を通じてドキュメントまたはドキュメントオブザーバにアクセスする手順を示す．文法は UML シーケンスダイアグラムである．説明のためメッセージ名は擬似コードで記述した．

ユーザがアプリケーションへドキュメントにアクセスするよう指示すると，この手順が開始する．Presentation はアプリケーションと Manager の対話を仲介する．Manager は Store を通じてドキュメントを復元する．次に，Manager は Store を通じてリスナを復元する．リスナとは，「ドキュメントにアクセスする」イベントの通知を受け取るオブジェクトのことである．ドキュメントオブザーバはリスナとしてのロールがある．Manager は各リスナにイベントを通知する．リスナがドキュメントオブザーバオブジェクトである場合，ドキュメントオブザーバオブジェクトはドキュメントを操作する．通知が完了した後，Manager は操作後のドキュメントを呼び出し元に返す．

本 API を実装したライブラリを利用してアプリケーションを構築するプログラマは，Listener クラスのサブクラスを作成し，update(Document) メソッドをオーバーライドする．update メソッド内でドキュメントを操作する必要があれば，Document オブジェクトを用いたコードを記述する．

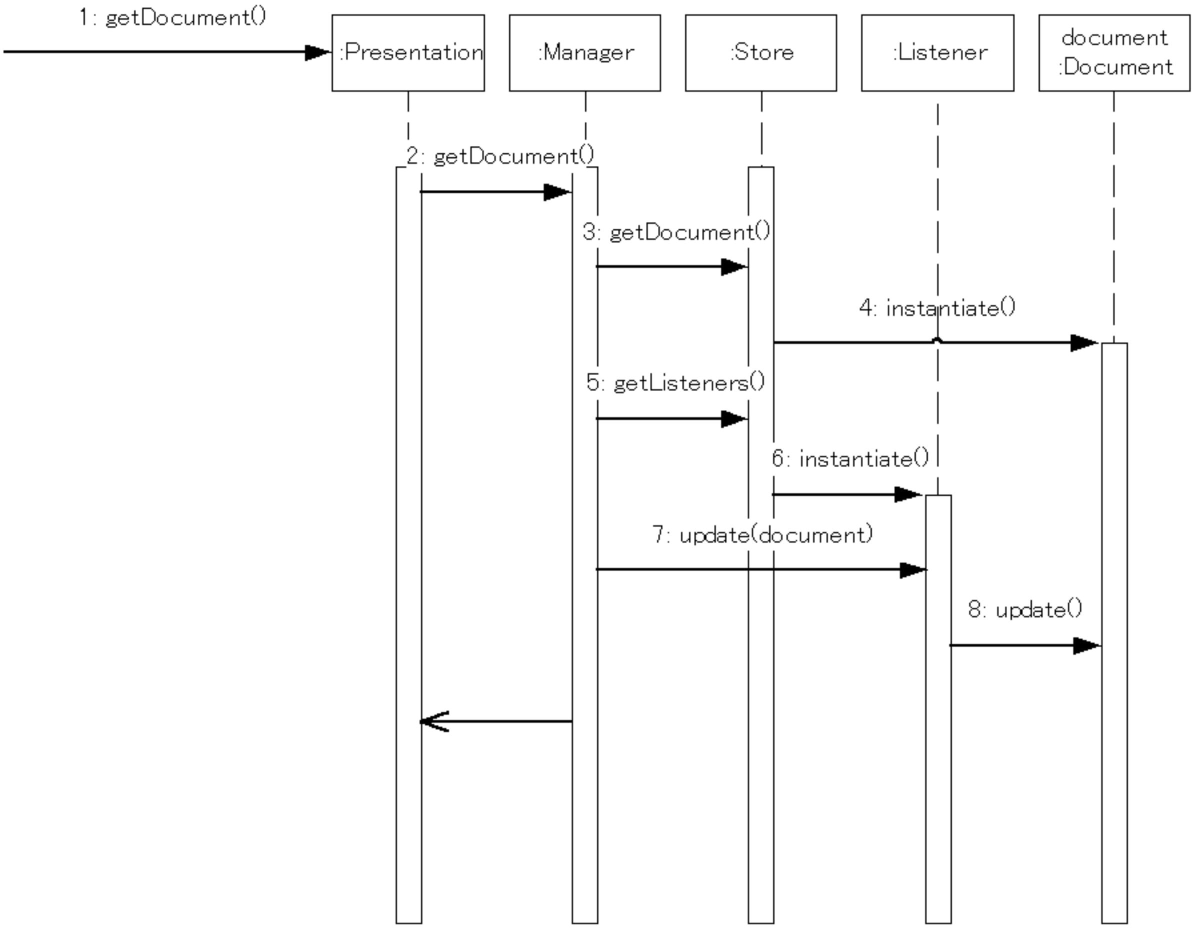

図 2.4: ドキュメントにアクセスする手順

第3章　WWWドキュメント更新APIの実装

3.1　概要

本章では，前章に基づいてWWWドキュメント更新APIのJava言語への実装について詳細に述べる．3.2節，3.3節では実装環境，実装方針について述べる．3.4節では3.5節のための予備説明を行う．

3.2　実装環境

WWWドキュメント更新APIの実装はJava言語を用いて行った．

表3.1にWWWドキュメント更新APIの実装環境を示す．

表3.1: 実装環境

項目	説明
ハードウェア	TOSHIBA Portege3020
オペレーティングシステム	MS-Windows98
実装言語	Java言語 (Sun JavaTM Development Kit Version 1.2)

3.3　実装方針

前章で述べた設計を元にWWWドキュメント更新APIを実装する．WWWドキュメント更新APIの実装方針として，拡張可能シーケンスを挙げる．

3.3.1　拡張可能シーケンス

前章で設計した「ドキュメントにアクセスする手順」や「ドキュメントまたはドキュメントオブザーバを永続化する手順」を実装するために，手順毎に専用のメソッドを定める場合，後にシステムに手順を追加する際にインタフェースを修正

しなければならない．本研究ではAbstract Factoryパターンを活用し，動的に手順を処理するようAPIを実装した．具体的な実装については，3.5にて解説する．

3.4 クラス定義概要

3.5節に登場するクラスについての概要を説明する．クラスの関係と振る舞いについては3.5節で述べる．

3.4.1 EnvelopePresentation クラス

EnvelopePresentationクラスは前章で説明したPresentationの，本実装における実装物である．

3.4.2 EnvelopeManager クラス

EnvelopeManagerクラスは前章で説明したManagerの，本実装における実装物である．

3.4.3 EnvelopeStore クラス

EnvelopeStoreクラスは前章で説明したStoreの，本実装における実装物である．

3.4.4 Document クラス

Documentオブジェクトは1つのドキュメントを表す．

3.4.5 Envelope クラス

Envelopeオブジェクトは1つのドキュメントを表す．Documentとの違いは，以下に示す2種類の両方を表す点である．

- 生のドキュメント
- ドキュメントオブザーバを格納したコンテナとしてのドキュメント

3.5 主な相互作用

3.5.1 ドキュメントまたはドキュメントオブザーバを永続化する手順

図3.1にドキュメントまたはドキュメントオブザーバを永続化する手順を示す．文法はUML シーケンスダイアグラムである． ユーザがアプリケーションにドキュメントを永続化するよう指示すると，この手順が開始する．EnvelopePresentationはEnvelopeRequest クラスのサブクラスであるAddEnvelopeRequest クラスのインスタンスを作成し，EnvelopeManager に渡す．EnvelopeManager はhandleEnvelopeRequest メソッド内でEnvelopeRequest オブジェクトの種類に応じて適切なEnvelopeRequestHandler クラスを選択し，EnvelopeRequest オブジェクトの処理をEnvelopeRequestHandler オブジェクトに委譲する．この選択と委譲によって3.3.1で述べた拡張可能シーケンスを実現している．

実際に本APIを利用してアプリケーションを構築する際は，EnvelopeRequestHandlerオブジェクトがドキュメントとドキュメントオブザーバを関連付けすることができる．本API では処理内容を規制することはない．ゆえに，API 利用者は必要に応じてEnvelopeRequestHandler クラスのサブクラスとEnvelopeRequest クラスのサブクラスを追加し，コンパイル時または実行時にシーケンスを修正することができる．

3.5.2 ドキュメントにアクセスする手順

図3.2にドキュメントにアクセスする手順を示す．文法はUML シーケンスダイアグラムである． ユーザがアプリケーションへドキュメントにアクセスするよう指示すると，この手順が開始する．EnvelopeRequestHandler オブジェクトに委譲する点は3.5.1と同様である．EnvelopeStore から Envelope オブジェクトとEnvelopeListener クラスのサブクラスである GetEnvelopeRequestListener のオブジェクトを復元し，各リスナにドキュメントを取得するイベントを通知する．

本APIではupdate メソッド内で処理内容を規制することはない．ゆえに，API利用者は必要に応じて EnvelopeRequestHandler クラスのサブクラスと EnvelopeRequestListener クラスのサブクラスを追加し，コンパイル時または実行時にシーケンスを修正することができる．

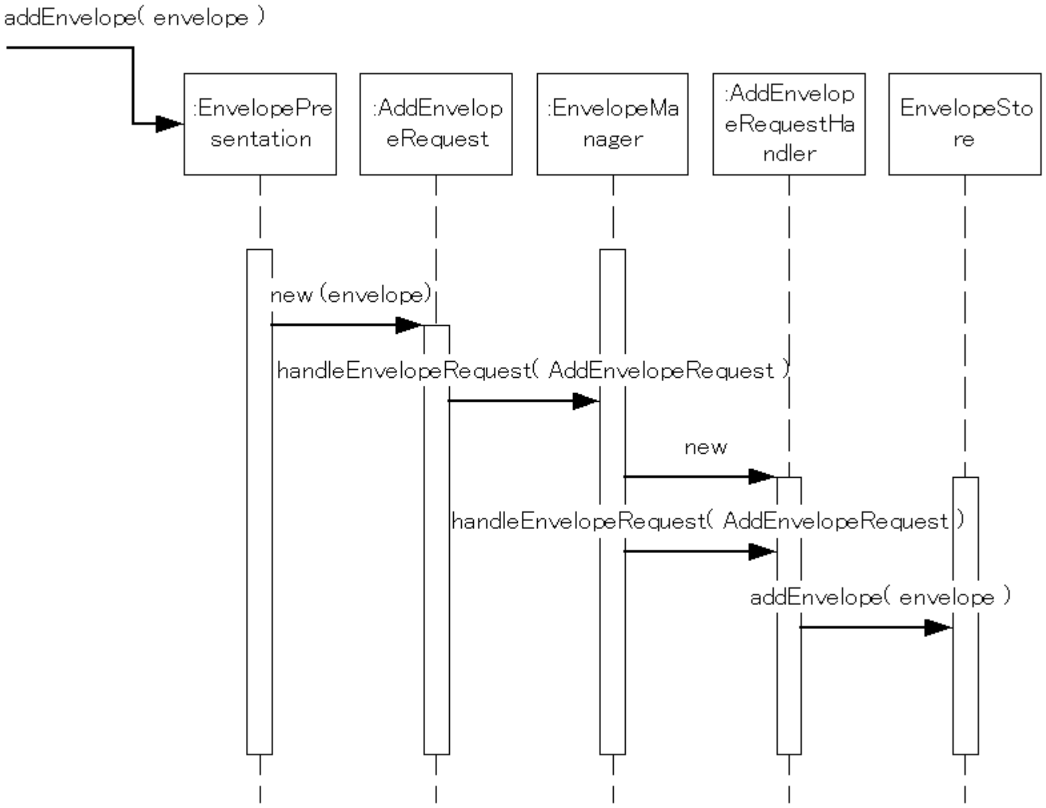

図 3.1: ドキュメントまたはドキュメントオブザーバを永続化する手順

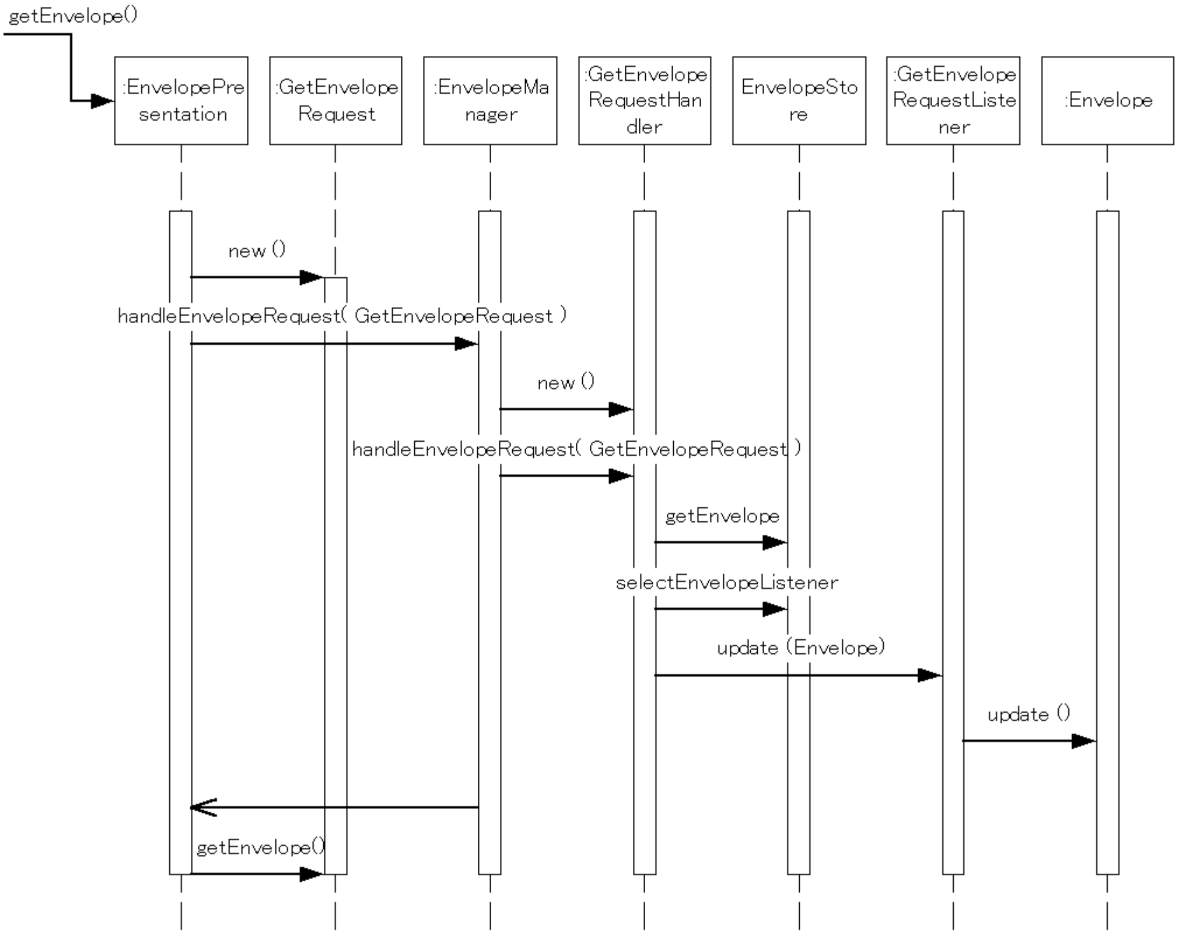

図 3.2: ドキュメントにアクセスする手順

第4章　WWWドキュメント更新システムの設計

4.1　概要

本研究では，WWW ドキュメント更新 API を構築・試用するためのサンプルアプリケーション (以降，WWW ドキュメント更新システム) を設計・実装する．本章では，WWW ドキュメント更新システムの設計について述べる．

4.2　基本要件

WWW ドキュメント更新システムは WWW ドキュメント更新 API を構築・試用するためのサンプルアプリケーションである．

WWW ドキュメント更新システムの目標を以下に示す．

- WWW ドキュメント更新 API を特定のアプリケーションに依存することなく実装できるかどうかを検査する．
- WWW ドキュメント更新 API を拡張性を保つよう実装できるかどうかを検査する．

4.3　設計方針

WWW ドキュメント更新 API では「ドキュメント，ドキュメントオブザーバを永続化する手順」の際に関連付けを行うことができるが，ここで 2 通りの判定方式がある．1 つはドキュメントオブザーバが関連付けを行うかどうかを判断する方式である．もう 1 つはここでは判断しないで関連付けを行う方式である．前者はリスナの数を少なく押さえることができるが，永続化しないドキュメントと関連付けを行うことができない．後者は全てのドキュメントオブザーバがイベントを受信することになるが，永続化しないドキュメントに関するイベントを受信することができる．

上記の 2 方式を比較することは WWW ドキュメント更新システムの目的である．

4.4 WWWドキュメント更新システムの概要

本システムは以下に示すシナリオを想定する．

ユーザはWWWブラウザを使い，WWWのサーチエンジンAにキーワードを入力して検索結果ページA1を得る．検索結果ページA1からハイパーリンクを辿り，キーワードに合致するページやURLに正しいページが存在しないことを告知するページなどを訪れた．後日，ユーザはWWWブラウザを使い別のサーチエンジンBに前回と同じキーワードを入力して検索結果ページB1を得る．このとき，検索結果ページA1からハイパーリンクを通じて辿った際に得た情報を元に，あるハイパーリンクのリンク先のURLが無効になっている場合，検索結果ページB1から当該ハイパーリンクを除外する．ユーザは簡単な操作だけで，正しいページが存在しないページへ2回訪れることを防止できる．

このシナリオを実現するために必要な機能を以下に示す．

機能1 検索結果ページA1からハイパーリンクを通じて辿ったページの状態を判定する．これはhttpのレスポンスコードが400番台であるかどうかで知ることができる．

機能2 ページの状態とURLの記録を永続化する．

機能3 ユーザが検索結果ページB1を閲覧する際に，検索結果ページB1内にあるハイパーリンクのリンク先URLと永続化しておいた記録にあるURLを照合し，URLが一致してページ状態が正しいページでない場合，検索結果ページB1からこのハイパーリンクを除外する．

機能1はhttpサーバと通信するhttpクライアントプログラムを作ることで実現する．機能2は機能1で得られたページの状態をURLとともにWWWドキュメント更新APIを通じて永続化するプログラムを作ることで実現する．機能3は検索結果ページB1の構造を解析するプログラムと，WWWドキュメント更新APIを通じて記録を復元するプログラム，そして記録を照合して検索結果ページB1を更新するプログラムを組み合わせることで実現する．

4.5 全体構成

4.5.1 アプリケーション

Presentationに接続するアプリケーションは最低限の機能だけを備える簡単なプログラムとする．このツールのユーザは標準入力またはテキストファイル入力により指示する．

第5章　WWWドキュメント更新システムの実装

5.1　概要

本章では，WWW ドキュメント更新 API を用いたサンプルアプリケーションの実装を解説する．

5.2　実装環境

WWW ドキュメント更新システムの実装は Java 言語を用いて行った．

表 5.1 に WWW ドキュメント更新システムの実装環境を示す．

表 5.1: 実装環境

項目	説明
ハードウェア	TOSHIBA Portege3020
オペレーティングシステム	MS-Windows98
実装言語	Java 言語 (Sun JavaTM Development Kit Version 1.2)

5.3　Presentation の実装

5.3.1　Presentation の実装詳細

WWW ドキュメント更新システムでは，Presentation を試用のため最低限の機能を持つ単純なコマンドラインアプリケーションとして実装した．CommandlinePresentation クラスは CommandlineShell クラスを特化し，EnvelopePresentation インタフェースを継承する．

5.3.2　実装に使用したライブラリ

表 5.2 に Presentation の実装に使用したライブラリを示す．

表 5.2: Presentation の実装に使用したライブラリ

機　　能	ライブラリ
Java 標準	Sun Java2 Standard Edition (JavaTM Development Kit Version 1.2)

5.4 Manager の実装

5.4.1 Manager の実装詳細

WWW ドキュメント更新システムでは，EnvelopePresentation クラスが Presentation パッケージの中核を成す．EnvelopePresentation クラスと関連するクラスとの関係を図 5.1 に示す．文法は UML クラスダイアグラムである．

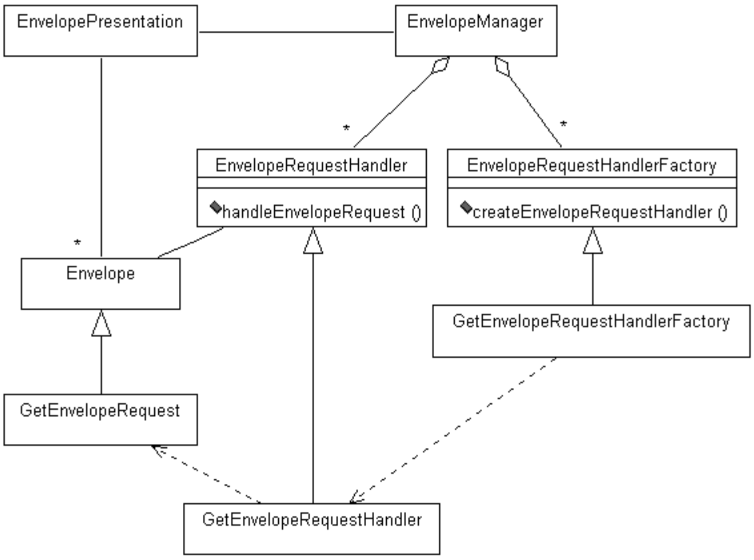

図 5.1: EnvelopePresentation クラスと EnvelopeRequestHandlerFactory クラス

5.4.2 実装に使用したライブラリ

表 5.3 に Manager の実装に使用したライブラリを示す．

表 5.3: Manager の実装に使用したライブラリ

機　　能	ライブラリ
Java 標準	Sun Java2 Standard Edition (JavaTM Development Kit Version 1.2)
XML プロセッサ	IBM XML Parser for Java version 2.0.9

5.5 Storeの実装

5.5.1 Storeの実装詳細

WWWドキュメント更新システムでは，EnvelopeManagerとEnvelopeStoreの関係を図5.2に示す．文法はUML クラスダイアグラムである．

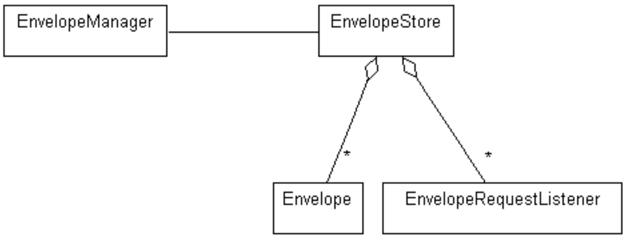

図 5.2: EnvelopeManagerクラスとEnvelopeStoreクラス

5.5.2 実装に使用したライブラリ

表5.4にStoreの実装に使用したライブラリを示す．

表 5.4: Storeの実装に使用したライブラリ

機　　能	ライブラリ
Java標準	Sun Java2 Standard Edition (JavaTM Development Kit Version 1.2)
DBMS	PSE for Java Release 2.0

5.6 WWWドキュメント更新システムの実装

表5.5に実装状況を示す．

表 5.5: 実装状況

機　　能	実装状況
WWWドキュメント更新API Presentation	△
WWWドキュメント更新API Manager	△
WWWドキュメント更新API Store	○
httpクライアント	×
サーチエンジン検索結果ページを更新する処理	×

第6章　おわりに

6.1　まとめ

本論文では，従来のWWWブラウザがドキュメントを更新する機能を分析し，その制約を考察した．従来のWWWブラウザにおける問題点は，ドキュメントとドキュメントオブザーバを分離できないため柔軟にこのプログラムを実行することができない点，またこのプログラムが保有するデータが一時的であるため他のコンテキストでこのデータを活用できない点である．この制約により例えば，参照した画像に新しいバージョンが発行されているときに当該バージョンの画像を新しいバージョンの画像で置き換える，といった処理ができない．

本研究ではこれらの制約に対してより柔軟にドキュメントを更新するための基本要件を分析し，WWWドキュメント更新APIを提案した．

本研究では，WWWドキュメント更新APIを実装し，同APIを利用する手順を定義した．同APIを実装したライブラリを利用することで，アプリケーションはドキュメントを発行するタイミング以外でもドキュメントとドキュメントオブザーバを関連付けておき，これを永続化することができる．

6.2　今後の課題

WWWドキュメント更新APIの今後の課題として以下の2点が挙げられる．

6.2.1　手順パターンを増やす

本論文で述べたWWWドキュメント更新APIを実装したライブラリは拡張機構と必要最低限の枠組だけを備えており，実際にアプリケーションから同APIを利用するためには利用する手順のためのクラスを追加しなければならない．よく使われる手順をライブラリに追加して，再利用できるようにすれば，ライブラリ利用者への負担を減らすことができる．

6.2.2 同ライブラリを活用するアプリケーションを構築する

上記のライブラリを利用するアプリケーションを構築する．例えば，送信したドキュメントに訂正を加える際に全体を再送信するのではなく，変更箇所だけを収めたドキュメントと「古いバージョンと変更箇所の差分をマージする」プログラムを送信することで，ユーザの手動操作に頼らずにドキュメントを自動的に訂正する，といった要件を実現する．

参考文献

[1] W3C, “Extensible Markup Language (XML) 1.0” http://www.w3.org/TR/REC-xml

[2] W3C, “Document Object Model” http://www.w3.org/DOM/

[3] IBM，“ XML Parser for Java” http://www.alphaworks.ibm.com/tech/xml4j, http://www.trl.ibm.co.jp/projects/xml/index.htm

謝辞

本研究を進めるにあたり，絶えず懇切丁寧な御指導を賜わりました，慶應義塾大学環境情報学部教授徳田英幸博士に深く感謝いたします．

慶應義塾大学徳田・村井・楠本・中村研究会の諸先輩方には，折りにふれ貴重な示唆や御助言，御指導を頂きました．特に慶應義塾大学政策・メディア研究科修士の中澤仁氏，藤村浩士氏には，本論文執筆にあたって絶えざる励ましと御指導を賜りました．また，Keio Media Space Family-Architecture(KMSF) 研究グループのメンバーには，本研究に関する様々な議論をして頂きました．ここに，深い感謝の意を表します．

2000年1月31日
中村 勇樹