

NAIST-IS-MT9451011

修士論文

広域ネットワーク型ハイパーテキストシステムにおける アノテーションサービスモデルの提案

植 達二

1996年2月16日

奈良先端科学技術大学院大学
情報科学研究科 情報システム学専攻

本論文は奈良先端科学技術大学院大学情報科学研究科において
修士(工学) 授与の要件として提出された修士論文である。

提出者： 植 達二

指導教官： 横矢 直和 教授
千原 國宏 教授
竹村 治雄 助教授

広域ネットワーク型ハイパーテキストシステムにおける アノテーションサービスモデルの提案*

植 達二

内容梗概

広域ネットワーク上に提供された文書に対してコメントや注釈を書き込む処理を行いたいというのは、紙面上に出力された文書に書き込むのと同様に当然の要求である。アノテーションは、ハイパーテキストシステムの文書に対し元の文書を直接書き換えることなく書き込みを実現するための記述方式である。広域ネットワーク型ハイパーテキストシステムにおいて、アノテーションを複数の利用者で共有できれば、効率の良い協調作業を実現できる。本論文では、アノテーションの情報空間を定義することにより、アノテーションを広域ネットワークを通じて、複数の利用者が複数のハイパーテキストシステムで共有できるアノテーションサービスモデルを提案する。本モデルではアノテーションにキーワードをつけることにより、異なるシステムおよび異なる文書中の同一情報に対するアノテーションを参照することもできる。これにより、従来単なるコメントとしてしか扱われていなかったアノテーションの再利用を可能にする。

キーワード

アノテーション, 広域ネットワーク, ハイパーテキストシステム, 協調作業

*奈良先端科学技術大学院大学 情報科学研究科 情報システム学専攻 修士論文, NAIST-IS-MT9451011, 1996年2月16日.

An Annotation Service Model in Hypertext Systems on Wide Area Network*

Tatsuji Ue

Abstract

It is a natural requirement that we want to write notes to a document on wide area network as we write notes on a book, a magazine and so on. An annotation is a description which enables a person to write notes on a document in hypertext systems without editing an original document directly. The sharing annotations between many users via wide area network enables the distributed editors to collaborate effectively. This paper describes an annotation service model in which many users can share annotations among different systems via wide area network. Adding a key word to each annotation, this model also enables users to refer an annotation to the same information in different systems and documents. This model gives users the possibility of reutilizing annotations which have been recognized only as mere comments.

Keywords:

annotation, wide area network, hypertext system, collaboration

*Master's Thesis, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-MT9451011, February 16, 1996.

目次

| | |
|---|----|
| 1. はじめに | 1 |
| 2. アノテーションの共有 | 3 |
| 2.1 アノテーションの共有の意義 | 3 |
| 2.2 従来のアノテーション共有システム | 3 |
| 2.2.1 NCSA HyperNews | 4 |
| 2.2.2 Cornell Annotation System(CoNote) | 4 |
| 2.2.3 ComMentor | 6 |
| 2.3 新しいアノテーションサービスの必要性 | 9 |
| 3. アノテーションサービスモデルの提案 | 10 |
| 3.1 スケーラビリティの考慮 | 10 |
| 3.2 設計理念 | 12 |
| 3.3 システム設計 | 13 |
| 3.3.1 パブリックアノテーションサーバ | 14 |
| 3.3.2 グループアノテーションサーバ | 15 |
| 3.3.3 ユーザアノテーションサーバ | 18 |
| 3.3.4 アノテーションブラウザ | 19 |
| 3.4 アノテーションのコミュニケーションモデル | 19 |
| 4. アノテーションのデータモデル | 21 |
| 4.1 提案モデルにおけるアノテーションデータ | 21 |
| 4.1.1 オリジナル空間情報 | 23 |
| 4.1.2 オリジナルユニット情報 | 24 |
| 4.1.3 AnnotateObject | 25 |
| 4.1.4 アノテーションツリー情報 | 26 |
| 4.2 データモデルの考察 | 30 |
| 4.2.1 アノテーションに対するアノテーション | 30 |
| 4.2.2 ユニットの適合度 | 31 |

| | |
|---|-----------|
| 5. システムの機能詳細 | 33 |
| 5.1 アノテーションの登録 | 33 |
| 5.2 アノテーションの検索 | 34 |
| 5.2.1 特定のハイパーテキスト文書に対するアノテーションの検索 | 35 |
| 5.2.2 同一情報に対するアノテーションの検索 | 35 |
| 5.2.3 アノテーションの属性による検索 | 38 |
| 5.3 アノテーションの参照 | 39 |
| 5.4 アノテーションの消去 | 39 |
| 5.4.1 ハイパーテキスト文書の消去 | 40 |
| 5.4.2 アノテーションツリーの再構成 | 41 |
| 5.5 共有範囲の変更 | 42 |
| 5.6 アノテーションキャッシュ | 42 |
| 5.7 アノテーションの既読管理 | 44 |
| 6. プロトタイプ的设计と実装 | 46 |
| 7. むすび | 49 |
| 謝辞 | 50 |
| 参考文献 | 51 |
| 付録 | I |
| A. プロトタイプシステムのプロトコル仕様 (アノテーションサーバ-アノテーションブラウザ) | I |
| A.1 ステップ | I |
| A.2 認証および権限取得 | I |
| A.2.1 private annotation を可能とするために | II |
| A.2.2 group annotation を可能とするために | II |
| A.2.3 現在どの範囲で annotation を検索しているのか | III |
| A.2.4 active な group/private を inactive にするために | IV |

| | | |
|-------|--------------------------------|------|
| A.3 | 登録 | IV |
| A.3.1 | 登録 1,2 | VI |
| A.3.2 | 登録 3,4 | IX |
| A.4 | 消去 | XII |
| A.4.1 | アノテーションオブジェクトの削除 | XII |
| A.4.2 | アノテーションオブジェクト TREE の削除 (将来の拡張) | XII |
| A.4.3 | アノテーションユニットの削除 | XIII |
| A.4.4 | アノテーションカプセルの削除 (将来の拡張) | XIII |
| A.5 | 取得 | XIII |
| A.6 | 終了 | XV |

目 次

| | | |
|----|----------------------------|----|
| 1 | NCSA HyperNews の画面 | 5 |
| 2 | CoNote の画面 | 7 |
| 3 | ComMentor の画面 | 8 |
| 4 | アノテーションの情報空間 | 13 |
| 5 | システム概念図 | 14 |
| 6 | グループアノテーションサーバへのランダムアクセス | 17 |
| 7 | gIBIS のコミュニケーションモデル | 20 |
| 8 | アノテーションのコミュニケーションモデル | 21 |
| 9 | アノテーションのデータモデル | 22 |
| 10 | WWW へのアノテーションの例 | 22 |
| 11 | 複数ハイパーテキストシステムでのアノテーションの共有 | 26 |
| 12 | エントリ番号の規則 | 28 |
| 13 | オリジナル空間情報の例 | 30 |
| 14 | ユニットの例 :元の文書 | 31 |
| 16 | アノテーションの検索 | 34 |
| 15 | アノテーションの登録 | 36 |
| 17 | 同一情報に対するアノテーション | 36 |
| 18 | 生成される単語列の例 | 38 |
| 19 | アノテーションの参照 | 39 |
| 20 | 既読管理ファイルの例 | 45 |
| 21 | プロトタイプシステムの構成 | 46 |

表 目 次

| | | |
|---|---------------------------------|----|
| 1 | グループメンバ情報の構成 | 15 |
| 2 | グループパスワード情報の構成 | 16 |
| 3 | アノテーションのデータフォーマット | 22 |
| 4 | オリジナル空間情報 (Original Info) の構成要素 | 23 |

| | | |
|----|---|----|
| 5 | オリジナルユニット情報 (Unit_Info) の構成要素 | 24 |
| 6 | Unit_Type による値の例 | 25 |
| 7 | AnnotateObject の構成要素 | 27 |
| 8 | Secondary_Src の構成要素 | 27 |
| 9 | Primary_Src の構成要素 | 28 |
| 10 | Annotation_Tree の構成項目 | 29 |
| 11 | Annotation_Tree のデータ例 | 29 |
| 12 | 共有範囲の変更と実際の操作 | 42 |
| 13 | 要素データごとの番号 | 44 |

1. はじめに

現在、広域ネットワーク上では情報を共有するために様々な工夫が行われている。匿名でのファイルの取得を許したファイル配布サービスである anonymous FTP サービス、相互情報交換を目的とした NetNews や Mail、広域ネットワーク上でのデータベース検索サービスを行う WHOIS[1] や WAIS[2] サービス、画像や音声を含んだ文書の表示が可能なブラウザの登場により爆発的な人気を誇るハイパーテキスト情報提供システム World Wide Web[3] 等が有名である。これらの情報提供サービスはいずれも情報提供者が提供する情報本体については第三者からの書き込みを禁止することによって、情報の管理者を明確にし、情報の整合性を保っている。

しかし、紙面上に出力された文書や、製本化された雑誌や本に読者が付箋や下線を付け加えたり、自分なりのコメント等を書き込む行為は日常生活では頻繁に行われていることである。このように書き込まれた付加情報は、書き込みを行った本人にとって意味があり、本来の情報に付加価値を与えることになる。また、この付加情報が他人にとっても意味のあるものであれば、付加情報の書き込まれた情報を参照する人達の間で付加価値も共有することになる。

上記の情報提供サービスにおいても、提供された情報に対して計算機上でコメントや注釈を書き込む処理を行いたいというのは、近年著しく向上し続けている計算機環境下においては当然の要求であると言える。書き込みを行った個人にとどまらず、広域ネットワーク上の計算機利用者の中で付加情報を共有することができれば、新たなコミュニケーションの可能性を広げ、効率の良い協調作業を行うことが可能となる。例えば、雑誌の原稿の査読等を広域ネットワーク上で複数人で行うことにより、従来非常に煩雑であった出版までのプロセスを円滑に行うことが可能になる。WWW では既に図書館の電子化のためのプロジェクトが各地で広まっており、このような電子図書館と関係することにより、付加情報の書き込みの技術は電子出版物の管理、運用をも円滑にする。また、記述された付加情報が利用しやすい状態でインターネット上に存在すれば、より多くの利用者が知識共有の場を拡大することができる。

このような要求に対して電子ニュースや電子メールなどでは、元の情報本体を

引用し、その引用された文書に対する個人の意見や注釈を記述する形で提供された文書に対する書き込みを実現している。しかし、この場合引用された元の文書の編集が可能であるため、元の情報が欠落している可能性がある。WWW 等のハイパーテキストシステムでは、提供された文書にコメントや注釈を記述するために、情報本体を直接書き換えずに書き込みを実現する手法の1つとして、アノテーション [4] を記述する手法がある。

現状の広域ネットワーク型ハイパーテキストシステム [3][5] においては、アノテーションを容易に複数人で共有できる仕組みにはなっていない。従来、WWW では読者が投稿した文字列を自動的に文書に埋め込む仕組みを用意することにより、アノテーションの共有を可能にしているシステム [6][7][8] も存在するが、その多くはアノテーションの対象となる文書中の具体的な位置が不明確で、かつ元の文書を参照しなくてはその文書に対するアノテーションを参照することはできない。そのため、共有のために記述されたアノテーションが誰にも参照されないままに放置される場合が多数存在する。また、WWW という特定のシステムに依存しているため、アノテーションの共有範囲も必然的に狭められている。

本論文では、上記の問題の克服を目的として、アノテーションを広域ネットワークを通じて複数人かつ異なるハイパーテキストシステムで共有でき、アノテーションの再利用性の向上を目指した新たなアノテーションサービスモデルを提案する。以下 2 章ではアノテーションシステムの現状とその中でのアノテーションの共有の意味と問題点について述べる。3 章では提案モデルの概要とスケーラビリティについて説明し、4 章でアノテーションのデータモデル、5 章で機能詳細について説明する。さらに 6 章ではプロトタイプの実装について説明した後、最後に 7 章でまとめと今後の課題について述べる。

2. アノテーションの共有

2.1 アノテーションの共有の意義

アノテーションとはハイパーテキストシステムの文書に対して、第三者が元の文書を直接書き換えることなく書き込みを行うための記述である。アノテーションを記述することは、コメントや注釈によって著者が提供した文書に付加価値を与えることを意味する。つまり、アノテーションは元の文書と合わせて一種の知的生産物であると言える。

このようにして個人が用意したアノテーションを元の文書と共に公開することにより、より多くの人々が元の文書のみを参照するよりも付加価値の高い情報を得ることが可能となる。しかし、ハイパーテキストシステムにおいて誰がいつ文書を参照してもアノテーションが記述された状態であることは、例え元の文書を直接書き換えていなくても書き換えたことに等しく、もはやそれはアノテーションの枠組を越えたものとなる。重要なことは文書を参照している読者がさらに付加価値の高い情報を得たいと思った時に限り、いつでも自由に付加価値情報としてのアノテーションを参照したり、記述できたりできることである。ハイパーテキストシステムで著者によって提供された文書を複数人が共有しているように、記述されたアノテーションを他人に公開することはアノテーションの著者と公開された人々との間でアノテーションを共有することになる。

このようにハイパーテキストシステムにおいて複数人でアノテーションを共有することを可能にするシステムのことをアノテーションシステムと呼ぶ。より多くの人々の間でアノテーションを共有するためには、既に基盤の整っている広域ネットワークを介したアノテーションの共有が有効である。本章では、現在広域ネットワークを対象としたアノテーションシステムの研究例と問題点、そしてその対策について述べる。

2.2 従来のアノテーション共有システム

現在、WWW を対象としたアノテーションシステムが幾つか実装されている。その中から代表的なものを紹介する。その多くは WWW での協調作業を支援す

るためのシステムである.

2.2.1 NCSA HyperNews

NCSA HyperNews[6] は WWW projects on collaboration の一貫として開発されたアノテーションシステムで, WWW での協議といった意見の交換を支援するものである. HyperNews の画面を図 1 に示す.

HyperNews では, アノテーションの記述の対象となる文書を **basic article** と呼び, HyperNews の読者は任意の basic article に対してアノテーションを記述できる. basic article とはあらかじめ HyperNews のサーバの管理者が利用者が自由に書き込みを行うために用意したページのこと, 実際の会議に例えると会議室に相当する. basic article は議論の分野ごとに分けられていて, 利用者は任意の basic article のページを参照して, 誰でも議論に参加することができる. 当然アノテーションに対するアノテーションの記述も可能で, 図 1 からわかるように, それぞれのアノテーションは木で階層的に構成されたリストとして表示される. 階層構造の深さに制限はない. また, 各アノテーションの著者はアノテーションに対するアノテーションの関係をアイコンによってグラフィカルに表現することができる. このように HyperNews は WWW の任意のページに対してアノテーションを記述できるわけではなく, システムの管理者によって提供されたアノテーションの共有の場所にアノテーションを記述することによって, 複数人が共同で文書を構築していくものである. この構築された文書は WWW で行われた議論の議事録とも言えるが, 元の文書をすっかり書き換えてしまっている.

2.2.2 Cornell Annotation System(CoNote)

CoNote[7] も利用者が WWW 上での情報交換を行うという点で HyperNews と良く似た性質をもつ. ブラウジング時の画面を図 2 に示す. 大きく異なる点は HyperNews が主に不特定多数の利用者によるアノテーションの共有を目指しているのに対し, CoNote は小規模のグループによるアノテーションの共有を目的としている. CoNote のグループアノテーションは, 文書の著者がグループのメンバーに誰がアノテーションを参照でき, 誰がアノテーションを参照/記述できるかとい

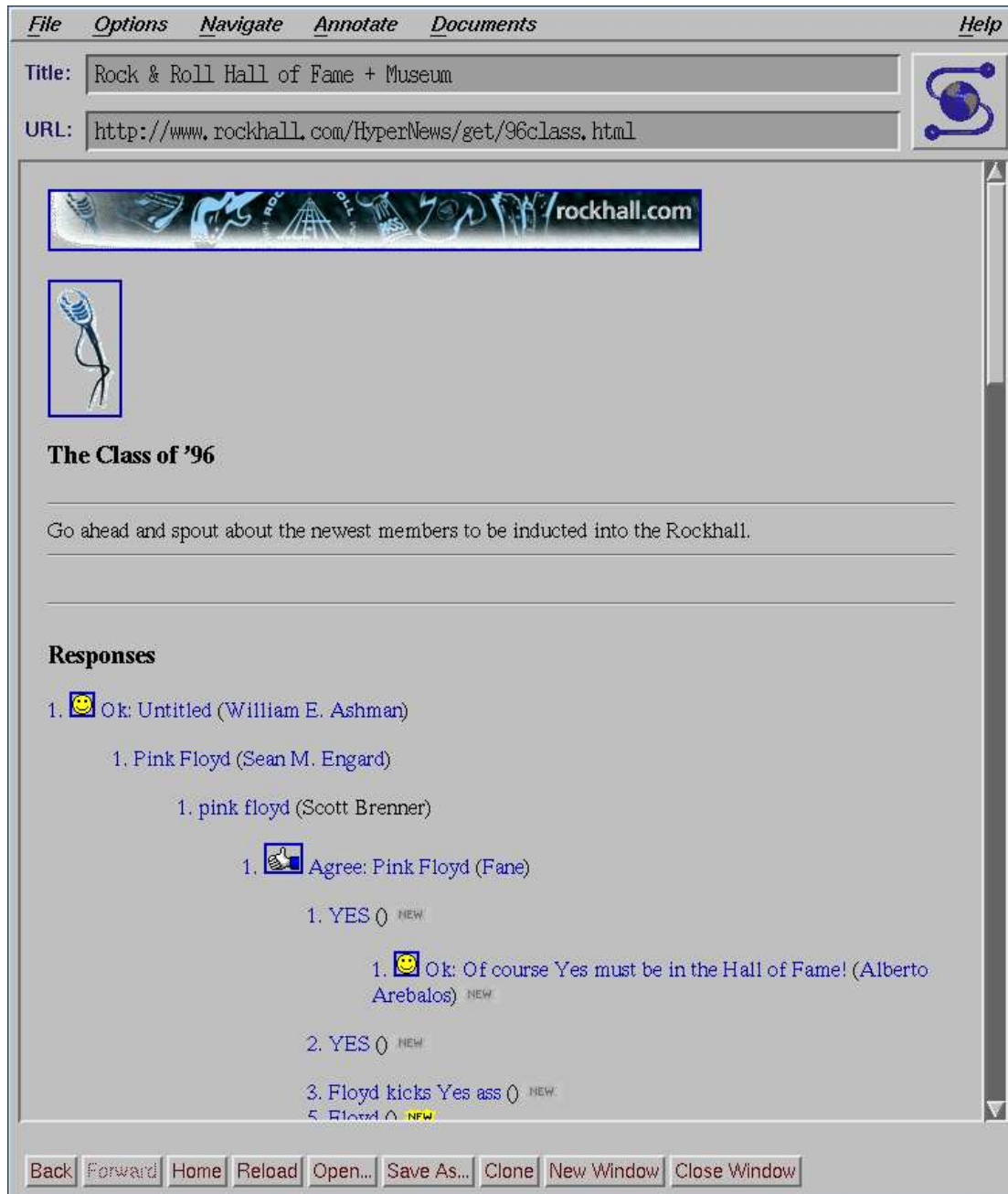


図 1: NCSA HyperNews の画面

う役割を決定する仕組みになっている。役割とは **viewer, reader, user, author** である。 **viewer** はアノテーションの対象となる文書を参照できるがアノテーションは参照できない。 **reader** はアノテーションを参照できるが記述ができない。 **user** は参照と記述が可能で、 **author** は文書の著者でさらに消去ができる。 **author** に限りアノテーションに対する操作の全権限を所有し、アノテーションを記述できる位置も **author** が文書内で定めた位置に限られている。

元々コーネル大学内で、教官と学生のコミュニケーションをとるために WWW で実装されたシステムであった。教官が用意した文書をアノテーション共有の場として、グループ内の役割が限定されているものの学生が質問し教官がそれに答えるといったことが繰り返されている。確かに大学のセミナー程度の小規模のグループにおいては、その目的と手段を明確にできるため CoNote を有効利用できるかも知れないが、大規模なグループとなると全員の役割を決定する **author** に非常に負荷のかかるシステムである。

2.2.3 ComMentor

ComMentor[8] はスタンフォード大学の電子図書館プロジェクトの一貫として設計されたアノテーションシステムである。このシステムではアノテーションを管理するためのアノテーションサーバ、アノテーションと元の文書を合成するためのドキュメント合成サーバ、アノテーションを登録、参照するためのブラウザから構成される。

ComMentor の主な特徴はドキュメント合成サーバにあり、アノテーションをポップアップウィンドウで表示 (図 3) することで、ユーザインタフェースとして優れたシステムと言える。

WWW の全てのページにアノテーションを記述することが可能であるが、一度記述されたアノテーションはアノテーションの実体そのものがアノテーションサーバに登録されるため、登録後のアノテーションの編集や消去等の操作が行えない。アノテーションの実体の蓄積場所も 1 つのアノテーションサーバに集中するため、保存領域において運営上の問題点も出て来る。また、アノテーションの記述の対象となる元の文書を参照しなくては、アノテーションを参照することはで

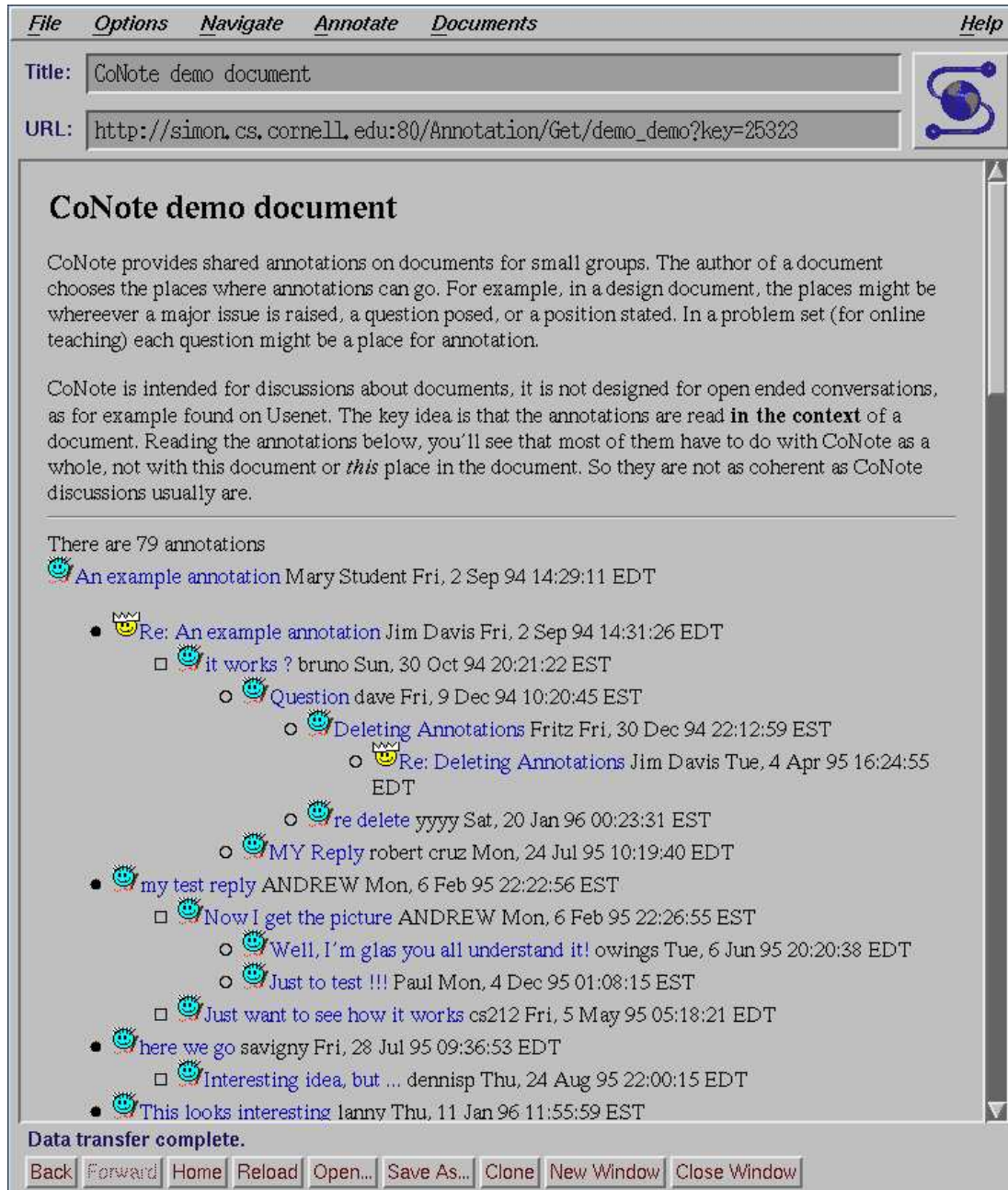


図 2: CoNote の画面

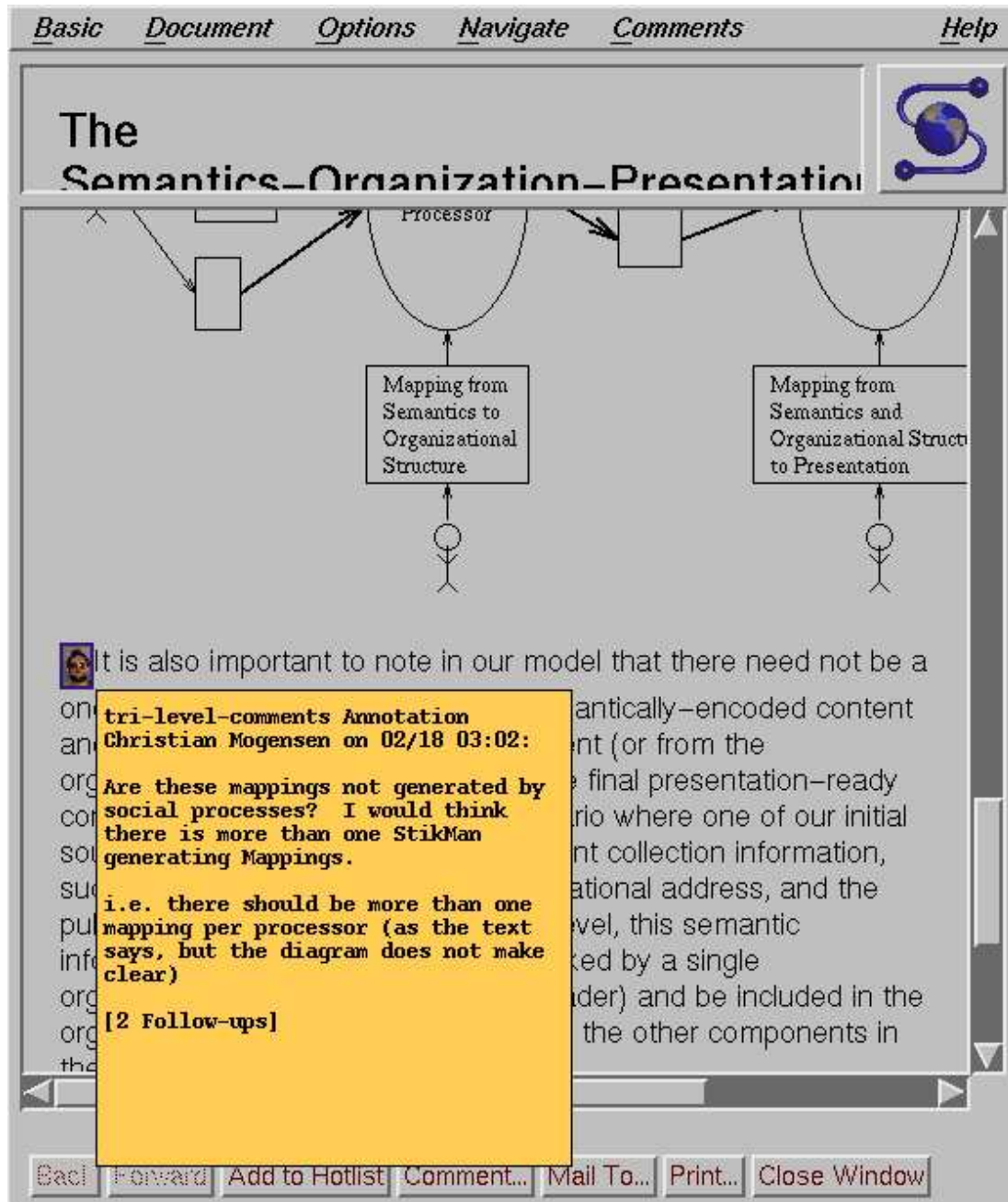


図 3: ComMentor の画面

きない。

2.3 新しいアノテーションサービスの必要性

これまでに紹介したアノテーションシステムの共通の問題点としては次のことがあげられる。

- アノテーションの対象となるページを参照しなければ、そのページに対するアノテーションを参照することができない。
- アノテーション記述後の操作があまりにも限定される。

これらの他に、アノテーションの記述の対象となるものが文字列でしかも文字列によるアノテーションしかできなかつたり、WWW というシステムに依存しているためアノテーションの共有範囲も必然的に狭められるという点も問題点として挙げられる。また、HyperNews と CoNote に限れば、アノテーションを記述することによって本来の文書を直接書き込みを行っており、情報本体の元の姿を変更してしまっている。さらに HyperNews では限界のある保存領域の問題から、個人が記述できるアノテーションの数を規制することもある。これらの問題は限られた WWW の機能の限界から生じてきている。

以上の問題を解決するために、アノテーションの情報空間をハイパーテキスト情報空間とは別に用意し、ハイパーテキストシステムに不足している上記の機能を補う必要がある。これらの機能によって、より再利用性の高いアノテーションの情報空間を構築することが可能となる。また、利用者が記述したアノテーションを利用者のローカルなディスクに保存することで、保存領域の問題も解消される。アノテーションの情報空間は決してハイパーテキスト情報空間から完全に独立しているのではなく、互いに関連し合う形態を維持する必要がある。関連し合うハイパーテキスト情報空間には複数の広域ハイパーテキストシステムであることが望まれ、それにより広い意味でアノテーションの共有を実現できる。しかし、それは現状のハイパーテキストシステムの枠組を崩さないことが要求される。

このような要求を満足する新たなアノテーションサービスモデルが必要である。

3. アノテーションサービスモデルの提案

3.1 スケーラビリティの考慮

広域ネットワークを介して複数人がアノテーションを共有するシステムでは、複数人が自由に書き込みを行うための場所を確保することは必須である。システムの利用者は、この共通の書き込み場所を参照することによってアノテーションを参照し、また互いに共有することが可能となる。このような共通の書き込み場所は利用者にアノテーションという文書を管理、提供しているため、以降アノテーションサーバと呼ぶ。

広域ネットワークを対象とするアーキテクチャを設計する際には、ネットワークトラフィック、計算機の負荷、保存領域等の問題を十分考慮し、膨大な利用者のアクセスにも対応可能なシステムを設計を行う必要がある。本節では、このような膨大な利用者に対しアノテーションサーバが誰にどのようなサービスを提供すべきかについて説明する。

任意の利用者によるアノテーションの共有 現状の広域ネットワーク型ハイパーテキストシステムでは特別な制約条件がない限り、利用者は世界中のハイパーテキストサーバにアクセスでき、世界中に分散されているハイパーテキストの文書を世界中の利用者で共有することが可能になっている。任意の利用者によるアノテーションの共有 (以下パブリックアノテーションと呼ぶ) においても同様に世界中の利用者によって共有されるのが望ましい。最も簡単な実現方法は世界中の利用者がアクセスできて自由に書き込みを行う場所 (アノテーションサーバ) を1つ用意することである。しかし、このように計算機、ネットワークが世界中の人に対して提供されることはまず考えられない。そこで、あるコミュニティ程度まで共有範囲を限定し、今後その拡張が可能である設計をせざるを得ない。故に、パブリックアノテーションを管理するアノテーションサーバは各コミュニティに1つ存在し、このサーバに登録できるのはそのコミュニティ内の利用者であるとする。ここでコミュニティは例えば日本の関西や関東といった地域ごとの組織や、企業や学術機関ごとに識別されている JPNIC[9] から割り当てられるドメイン別に存

在すると考えてよい。

グループによるアノテーションの共有 グループによるアノテーションの共有(以下グループアノテーションと呼ぶ)では, 利用者を限定することによりある程度スケーラビリティの問題を解消できる. グループの規模としては, 数十人程度の者から数百人, 数千人というものが考えられる. 小規模なグループであれば, 複数のグループで1つの計算機を共有することにほとんど困難はないと思われる. しかし, 大規模なものになると一度に数百数千の利用者によるアクセスが考えられ, ネットワークトラフィック, 計算機の負荷を必要以上に高めてしまう恐れがある. この問題はパブリックアノテーションにおいても同様に重大な問題の1つであったが, グループアノテーションにおいても十分考慮する必要がある. そこで, グループアノテーションにおいてもグループの規模が膨れ上がった時には, グループアノテーションを管理するアノテーションサーバが存在する LAN 内に限り任意にグループアノテーションのデータを別の計算機に複製(ミラー)することを考える. WAN と LAN におけるネットワークバンド幅を比較した場合, 一般に LAN の方が広いと言えるため, WAN 上でのミラーは不必要にネットワークバンド幅を消費する可能性がある. 従って, ホストの負荷を軽減させるためのミラーは LAN 内の方が有効である.

アノテーションの実体の保存領域 アノテーションの共有範囲がいかなるものであれ, アノテーションの保存領域が1つの計算機に集中することになれば, その限られた計算機ハードディスクの容量から, 利用者が自由にアノテーションを記述することが困難になる. そこで, ハイパーテキスト文書と分離して管理するアノテーションには必要最低限の情報だけを蓄積し, アノテーションの実体そのものはアノテーションを記述する著者が所有するローカルな計算機ハードディスクに蓄積する. これはアノテーション登録後の編集を可能にし, 利用者の操作の幅を広げることにもつながる.

以上のことからアノテーションを管理するアノテーションサーバは, アノテーションの共有範囲ごとに用意されのが有効であると考えられる.

3.2 設計理念

2章で指摘した従来システムの問題点は、主にWWWという特定のシステムに依存するための限界から生じたものであった。これらの根本的な問題に対して、提案モデルではアノテーション独自の情報空間を構築することによって、異なるハイパーテキストシステム間でのアノテーションの共有を始め、様々な問題に対処する。同時にアノテーションの情報空間は、複数の利用者が協調して作業を行う協調作業空間でもある。このために、アノテーションの情報空間を効率的に構築するようにアノテーションのデータ構造が工夫されていなければならない。

本研究では、前節で述べたスケーラビリティ的な問題を背景に、2章で指摘した問題点を解決するために、以下のような設計理念に基づく新しいアノテーションサービスモデルの設計を行った。

- 広域ネットワークを通じた複数人でのアノテーションの共有が可能。共有範囲としては、ワーキンググループを想定したグループアノテーションと不特定多数で共有できるパブリックアノテーションを設定。
- 異なる広域ネットワーク型ハイパーテキストシステム間でアノテーションの共有が可能。
- 利用者がアノテーション登録後の編集、消去等の操作が可能。
- アノテーションの対象となる文書中の位置が明確。
- 文字以外のアノテーションが可能。
- 元の文書を参照しなくても同一情報に対するアノテーションが参照可能。
- アノテーションから元の文書が検索可能。
- アノテーションに対するアノテーションの記述により、効率の良い協調作業を実現。

後半の5項目においてはこれらを可能にするデータモデルの定義を4章で示す。

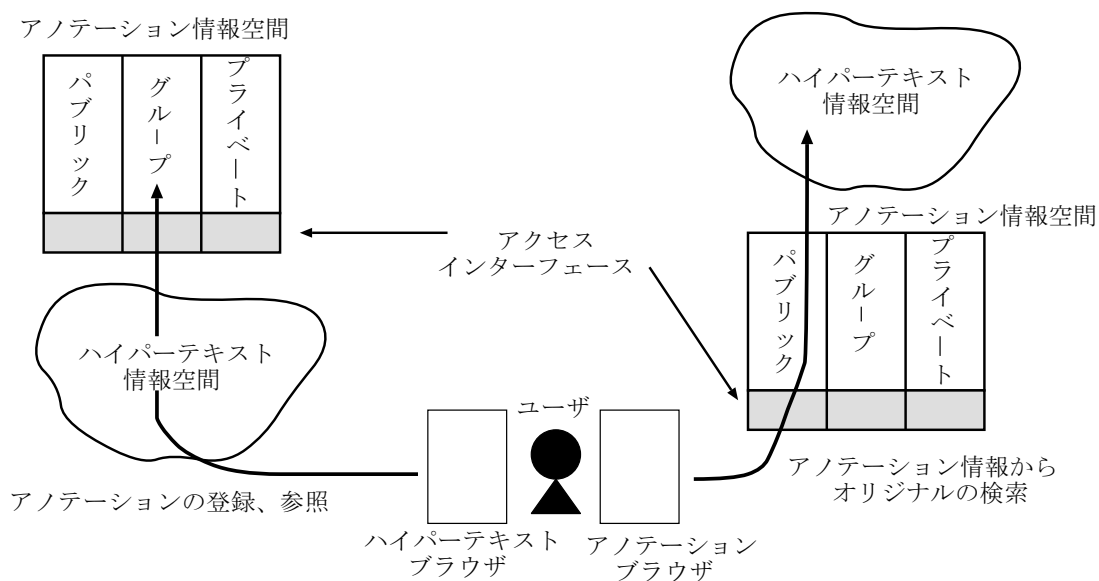


図 4: アノテーションの情報空間

以上の設計理念から、図 4に示されるハイパーテキスト情報空間と分離したアノテーション情報空間の構築を目指す。利用者は各ハイパーテキスト情報空間から固有のアクセスインターフェースを介してアノテーションの書き込み、参照を行い、逆にそのようにして構築されたアノテーションの情報空間を検索、参照することによってハイパーテキスト情報空間にアクセスすることができる。

3.3 システム設計

本システムはアノテーションを管理、提供するための各種アノテーションサーバとアノテーションを取得、参照するためのアノテーションブラウザからなるサーバ・クライアントモデルに基づいている(図 5)。各利用者は利用者個人ごとに用意されているアノテーションブラウザを通してアノテーションを参照することができる。

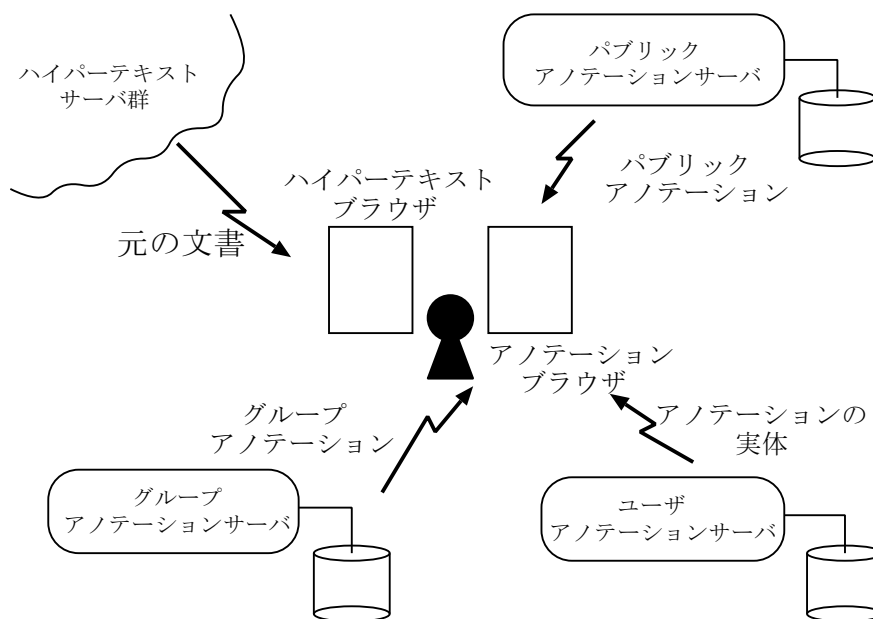


図 5: システム概念図

3.3.1 パブリックアノテーションサーバ

パブリックアノテーションを管理する。アノテーションブラウザが要求する文書に対するパブリックアノテーションを返す。パブリックアノテーションサーバによって管理されるアノテーションは不特定多数の利用者によって共有されるが、1つのパブリックアノテーションサーバは全ての利用者には書き込みの場所を与えるわけではない。利用者の属するコミュニティ内にパブリックアノテーションサーバがない場合は情報の発信(アノテーションの登録)はできないが、アノテーションの参照は可能である。

複数のコミュニティが存在するようになり、パブリックアノテーションサーバが複数分散されて存在するようになると、利用者はアノテーション検索、参照するために複数の分散されたパブリックアノテーションサーバにアクセスすることになる。その際、各利用者がどのようにしてパブリックアノテーションサーバの存在を知るかが問題となる。そこで、DNS(Domain Name System) にアノテーショ

ンサーバの新たなリソースレコードを用意し、外部へオープンにしたい Public Annotation Server が立ち上がるたびにそのレコードに登録する。各利用者はドメイン単位で DNS に問い合わせに行くことによって、各 Public Annotation Server の存在を知ることが可能となる。

3.3.2 グループアノテーションサーバ

グループアノテーションを管理し、パブリックアノテーションサーバと同様に指定されたグループアノテーションを返す。一つのグループアノテーションサーバで複数のグループのアノテーションを管理し、メンバのアクセス制御を行う。グループのメンバにはグループアノテーションにアクセスするためのパスワードが配布され、メンバの認証はそのパスワードによって行われる。

各グループアノテーションサーバで管理されるグループの情報は、グループのメンバを管理するグループメンバ情報、グループの認証パスワードを管理するグループパスワード情報がある。それぞれの構成を表 1,2 に示す。基本的にこれらの情報さえあればグループアノテーションサーバを介したグループコミュニケーションが可能である。

表 1: グループメンバ情報の構成

| 構成要素 | 項目の値 |
|-------------------|---------------|
| Group_Name | グループ名 |
| Group_Manager | グループ管理者名 |
| Group_Member_List | メンバの電子メールアドレス |
| Host_Type | ホストの属性 |
| Original_Host | オリジナルホスト名 |
| Mirror_Host | ミラーホスト名 |

グループ名 グループ名は次の規則に従うものとする。

グループ名 = グループ ID + "@ " + 管理ホスト名 + ". " + 管理ホストドメイン

表 2: グループパスワード情報の構成

| 構成項目 | 項目の値 |
|---------------|---------------------|
| Group_Name | グループ名 |
| Group_Manager | グループ管理者名 |
| Group_Passwd | グループメンバ認証の パスワード |

グループ ID = (英数字 | アルファベット)*

この形式はメーリングリストのようにグループ名の後にホスト名とドメインを記述することによって、名前空間の一意性を保つ。従って、同じホスト内で重複がない限りグループ ID の命名の仕方は任意である。管理ホスト名とドメインは DNS のエントリと同じにする。また、グループのメンバ名は電子メールアドレスを採用することにより、これもメンバ名の一意性を保つ。電子メールアドレスの正当性を確実なものとするためにはグループメンバに加わるための手続きは電子メールを介して行う必要がでてくる。

ホストの属性 ホストの属性 (Host_Type) は、そのグループアノテーションサーバのホストが元々存在していたもの (Original) かあるいはミラーサイトとして増設されたもの (Mirror) かを指定する。さらに、他にもミラーホストが存在する場合は Mirror_Host にミラーのホスト名を指定し、オリジナルのホストが存在する場合は Original_Host にオリジナルのホスト名を指定する。ミラーリングの方法は、グループの管理者以外のメンバが第 2 管理者として管轄内のグループアノテーションサーバにミラーホストとしての申請を行う。ミラーであるグループアノテーションサーバが運用された後は、オリジナル側に登録、消去されたアノテーション情報はミラー側へ、ミラー側で登録、消去されたアノテーション情報はオリジナル側へと交互に通信される。また、ミラーホストが複数存在する場合、1 つのミラーホストの変更がオリジナルホストに通知されると、オリジナルホストは残りのミラーホストへも変更内容を通知する。一方、クライアントはミラーホストの情報を

知るためには必ず一度はオリジナルのグループアノテーションサーバにアクセスしなければならない。グループアノテーションサーバへのアクセスでクライアントはミラー情報を知り、それ以降のアクセスはランダムあるいは順番にLAN内に分散されたグループアノテーションサーバにアクセスすることになる(図6)。このようにミラーホストはオリジナルホストと等価なものであるため、いずれかのホスト内のデータ構造に変更があった場合は必ず他のホストにデータの変更内容が通知される。

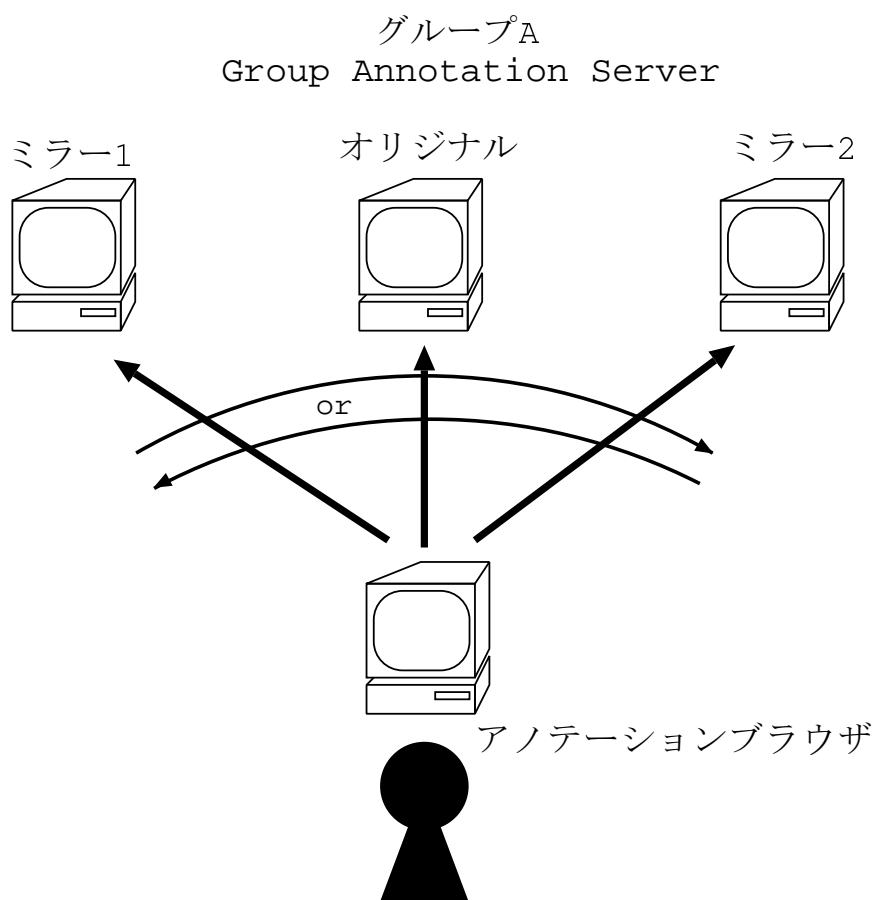


図 6: グループアノテーションサーバへのランダムアクセス

グループアノテーションの運用 1つのグループ情報はすべて、そのグループの管理者がグループ結成の申請を行ったグループアノテーションサーバで管理される。グループアノテーションのグループはメーリングリストのようにグループを代表する管理者のもとに、自由にメンバの登録、除名ができるものである。グループのメンバになるためのドメインの制約は存在せず、電子メール等を利用してグループの登録作業を行う。メンバ認証のためのパスワードの通知も同様の手段を通して行われることになるが、安全性においては特に考慮していない。なぜなら、ここで想定するグループアノテーションはグループ内で秘密情報を守るためのものではなく、グループのメンバ以外の利用者に余計な情報を見せないという方針から安全性に着眼点を置く必要はないと考えている。しかし、今後は十分に考慮する余地のある問題であることも確かで、電子メールへの署名や暗号化を行う pgp [10] のような仕組みをハイパーテキストシステムに適用することも考えられる。グループの管理者は、自分のドメイン内にグループアノテーションサーバが存在する者は誰でもなれるが、その際グループの管理者はグループアノテーションサーバの管理者にアノテーションを共有する場所を得るための申請を行う必要がある。グループのメンバはそのサーバをアノテーション共有の場として自由にアノテーションの登録、参照ができるようになる。

3.3.3 ユーザアノテーションサーバ

利用者ごとに用意されており、利用者が記述するアノテーションの文書そのもの(実体)を管理する。各利用者のアノテーションブラウザが要求するアノテーションの実体を返す。アノテーションの実体のアドレスは、アノテーションブラウザがそれぞれのアノテーションサーバから受け取るアノテーションのデータに含まれている。また、アノテーションを記述した本人しか参照することのできないプライベートアノテーションを管理する。さらに、アノテーションのキャッシュ機能を持ち、1度参照したデータを一時保管する。

ここに蓄積されるアノテーションの実体はアノテーションを記述した利用者の管理下にあるため、自由に編集が可能となる。

3.3.4 アノテーションブラウザ

ハイパーテキストの情報空間とアノテーションの情報空間とのインタフェースに当たり、また同時に利用者とアノテーションの情報空間との入出力インタフェースでもある。ハイパーテキストブラウザと連係して動作し、ハイパーテキスト文書に対するアノテーションの検索、アノテーション実体の取得、アノテーションの登録、削除等の種類のメッセージを各アノテーションサーバに送信する。

また、ハイパーテキストブラウザがハイパーテキストサーバから受け取るメッセージは全てアノテーションブラウザに送られる。一方、アノテーションブラウザからハイパーテキストブラウザへのメッセージは、アノテーションの検索結果として返ってきたハイパーテキスト文書の情報識別子のみが送られる。

3.4 アノテーションのコミュニケーションモデル

従来のアノテーションシステムがアノテーションの共有をコミュニケーションの手段としているように、提案モデルにおいてもコミュニケーションにより円滑な協調作業を実現するための手段としてアノテーションの共有を行う。gIBIS[11]では、ハイパーテキストシステムにおけるコミュニケーションのモデル化をしており、そのモデルを図7に示す。このモデルはソフトウェア等の設計問題の解決を支援するものであるが、実験によりモデルの有効性が示されている。

アノテーションによるコミュニケーションを行う場合においても、何らかの問題を提起しその問題に対する回答を得るような場合も十分考えられる。このコミュニケーションモデルを提案システムに応用することにより、効率の良い協調作業の実現が可能と考える。アノテーションの著者がアノテーションを記述する際には、必ずそのアノテーションを記述するに至った意図や目的が存在する。特にアノテーションに対するアノテーションを記述する際には、この意図や目的をある程度一般化することで、一連のアノテーションによるコミュニケーションを円滑に行うことが可能になると考える。アノテーションを記述する著者の意図や目的には次のようなものが挙げられる。

- コメント

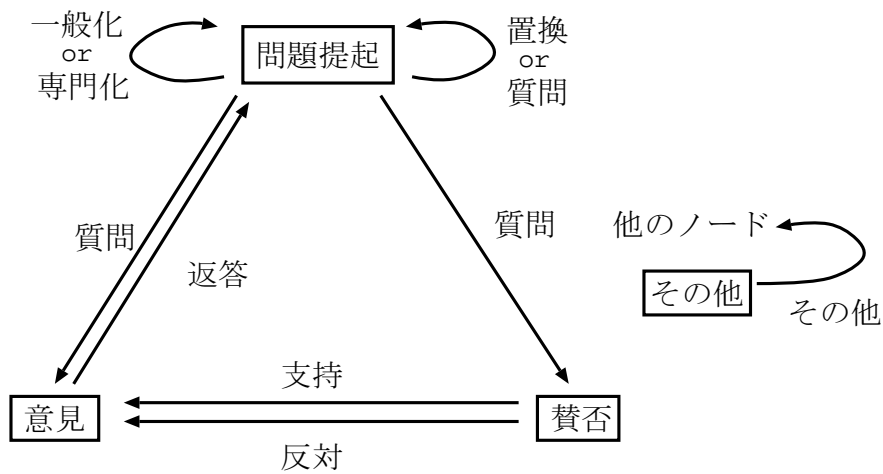


図 7: gIBIS のコミュニケーションモデル

- 参考文献
- 批評
- 質問
- 返答
- 支持
- 反対

gIBIS[11] のコミュニケーションモデルを提案モデルに適用した結果を図8に示す。長方形はコミュニケーションにおける状態(ノード)を表し、矢印はアノテーションを記述する著者の意図(リンク)を表す。混乱を避けるために「その他」のノードおよびリンクは取り除いた。図8からも分かるように、状態の遷移はリンクの種類によって一意に定まる。

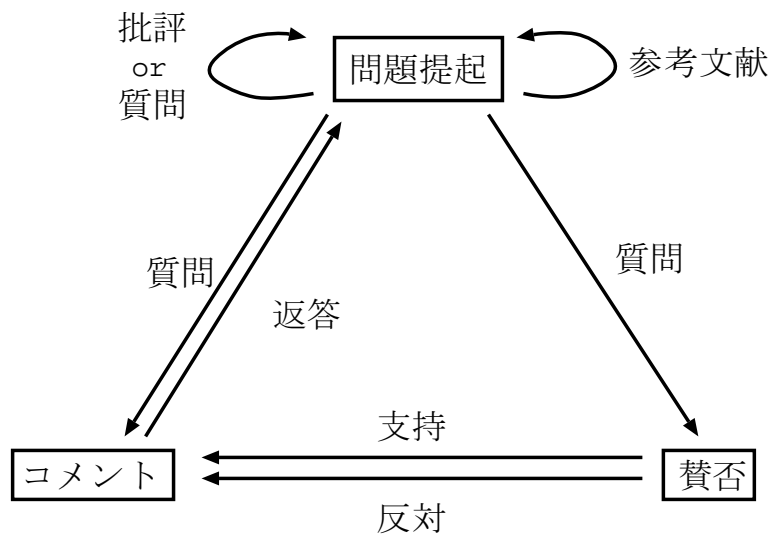


図 8: アノテーションのコミュニケーションモデル

4. アノテーションのデータモデル

異なるハイパーテキストシステム間でアノテーションを共有するためには、アノテーションの情報空間をハイパーテキストの情報空間と分離する必要がある。情報空間の分離はアノテーションのデータモデルを新たに定義することによって行う。また、データモデルは異なるハイパーテキストシステムにおいて対応が可能でなければならない。そのためには、アノテーションの記述の対象となるハイパーテキスト文書に関する詳細な情報をアノテーションデータ中に持つことが不可欠となる。

4.1 提案モデルにおけるアノテーションデータ

本節では、第 3.2 節で述べたことを可能にするために定義したアノテーションのデータモデル (図 9) について説明する。また、WWW の文書にアノテーションを記述した場合の例を図 10 で示す。また、実際に蓄積されるデータ (Anno_Data) の構成項目を表 3 に示す。

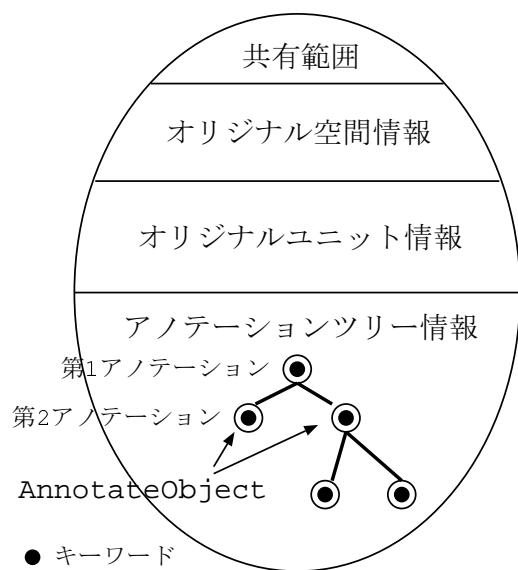


図 9: アノテーションのデータモデル

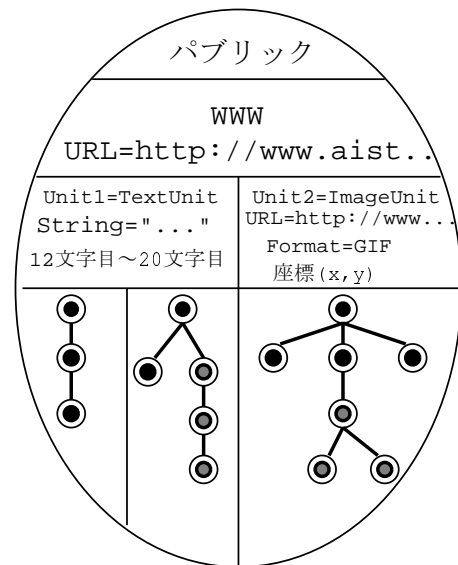


図 10: WWW へのアノテーションの例

表 3: アノテーションのデータフォーマット

| 構成項目 | 名称 | 項目の値 |
|------------------|-----------------|---|
| アノテーションデータエントリ番号 | Data_Entry | エントリ番号 |
| 共有範囲 | Share_Level | "private" "group:"Group_Name "public" |
| オリジナル空間情報 | Original_Info | 表 4 |
| オリジナルユニット情報 | Unit_Info | 表 5 (複数可) |
| アノテーションツリー情報 | Annotation_Tree | 表 10 (複数可) |

表 4: オリジナル空間情報 (Original Info) の構成要素

| 構成項目 | 項目の値 |
|----------------|---------------------------------------|
| Space_Info | 元の文書の情報空間名 (例 :WWW, Hyper-G, etc.) |
| Info_Specifier | 情報識別子 (例 :URL) |
| Unit_Count | ユニットの個数 |
| Units | ユニット群 (ユニットのエントリ番号列) |

個々のアノテーションデータはエントリ番号がつけられ, その番号の元で管理される. 共有範囲は `private`, `group`, `public` のいずれかの文字列を指定し, `group` の場合はコロン (`:`) とその後にグループ名を指定する.

以下, オリジナル空間情報, オリジナルユニット情報, `AnnotateObject`, アノテーションツリー情報と順に説明する.

4.1.1 オリジナル空間情報

ハイパーテキストシステムにおいてはシステム固有の情報識別子がある. ここではアノテーションの対象となる情報空間名とその情報識別子を格納する. 共有範囲とオリジナル空間情報が同じアノテーションは全て一括管理される. 表 4 に構成要素を示す.

`Unit_Count` ではオリジナル情報空間の情報識別子 (ある文書) に含まれるユニット (次節参照) の個数が, `Units` では各ユニットのエントリ番号がそれぞれ格納される.

表 5: オリジナルユニット情報 (Unit Info) の構成要素

| 構成項目 | 項目の値 |
|-------------------|--|
| Unit_Entry | ユニットのエントリ番号 |
| Unit_Type | ユニットの種類 •Page_Unit •Text_Unit •Image_Unit |
| Unit_Content_Type | ユニットのデータタイプ |
| Unit_Position1 | ユニットの位置の指定 1(省略可) |
| Unit_Position2 | ユニットの位置の指定 2(省略可) |
| : | : |
| Unit_Position n | ユニットの位置の指定 n (省略可) |
| Unit_Entity | ユニットの実体へのポインタ |
| Tree_Count | アノテーションツリーの個数 |
| Trees | アノテーションツリー情報へのポインタ群 (ツリーの ID 番号列) |

4.1.2 オリジナルユニット情報

アノテーションの対象となる文書中の位置を明確にするためにユニットという概念を用いる。ユニットとは、ハイパーテキストシステムで使用できる文字列や画像といった素材を表現するもので、1文字、1ピクセルから情報識別子単位までの指定が可能である。アノテーションの対象が一字でも異なれば、それは異なるユニットとみなす。同様に構成要素を表 5 に示す。

Tree_Count では各ユニットに付随するアノテーションツリー情報(次々節参照)の個数が、Trees では各アノテーションツリー情報の個数がそれぞれ格納される。Unit_Position は可変長で、複雑なデータ型の位置の指定を可能にする。

また、Unit_Type による値の例を表 6 に示す。

Image_Unit において、Unit_Entity の画像ファイルの情報識別子が Info_ID と識別子が異なる場合は、それも合わせて画像ファイル名とする。

表 6: Unit_Type による値の例

| | Page_Unit | Text_Unit | Image_Unit |
|-------------------|-----------|---|---|
| Unit_Content_Type | — | <ul style="list-style-type: none"> •Plane •RichText | <ul style="list-style-type: none"> •GIF •BMP •IRIS •PICT •jpeg •EPS •SUN |
| Unit_Entity | 情報識別子 | 対象文字列 | 画像ファイル名 |
| Unit_Position1 | — | 文字列開始位置 (オフセット) | 1 次座標 |
| Unit_Position2 | — | 文字列終了位置 (オフセット) | 2 次座標 |
| Unit_Position3 | — | — | 3 次座標 |
| : | — | — | : |
| Unit_Position n | — | — | n 次座標 |

4.1.3 AnnotateObject

アノテーションの実体に関する情報を管理する。1つの AnnotateObject が1つのアノテーションに相当する。ここではデータの所在, データのフォーマットを定義し, 従来文字主体であったアノテーションに対し, 文字以外のアノテーションの記述を可能にする。

また, 利用者はアノテーションを記述する際に, 同一情報を検索する手がかりとなる任意のキーワードを入力できるが, キーワードは AnnotateObject の一部の情報として管理される。このキーワードと文書中の文字列が同じものは同一情報とみなし, 本来別のユニットに対して記述されていたアノテーションを現在参照している文書中のユニットに対するアノテーションとして見せることにより, 異なるシステムあるいは異なる文書に対するアノテーションの共有が可能となる (図 11)。AnnotateObject は大きく分けてアノテーションのエントリ番号を示す Entry_ID と 2 次情報である Meta_Info, 1 次情報の Obj_Info の 3 要素から構成される (表

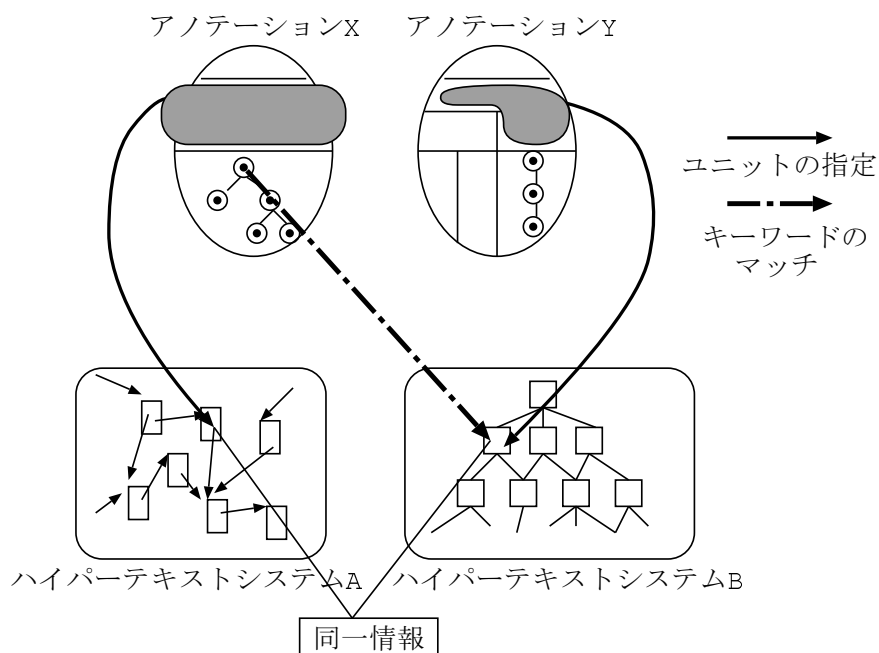


図 11: 複数ハイパーテキストシステムでのアノテーションの共有

7). Secondary_Src と Primary_Src はさらに表 8, 9の要素から構成される.

表 8でアノテーションの著者はユーザ認証を経たユーザ ID が登録される. Communication_Type ではアノテーションに対するアノテーションの記述の際のコミュニケーションの種類 (リンクタイプ) を指定する. ハイパーテキストの文書に対するアノテーションの記述では, 1つのユニットが1つの問題提起に相当する. Obj_Content_Type の取り得る値は Unit_Content_Type と同じである (表 6). また, Pri_Info を複数指定することによって同時に複数のデータタイプのアノテーションの記述を可能にする.

4.1.4 アノテーションツリー情報

アノテーションに対するアノテーションの記述を可能にするため, 個々の AnnotateObject 間の関係を木構造で管理する. 最初に行われるアノテーションを第 1 アノテーション, 以降第 2, 第 3 と続く. キーワードの入力がない場合には一つ

表 7: AnnotateObject の構成要素

| 構成項目 | 項目の値 |
|---------------|--------------------------|
| Obj_Entry | アノテーションの実体のエントリ番号 (図 12) |
| Secondary_Src | アノテーションの実体の 2 次情報 |
| Primary_Src | アノテーションの実体の 1 次情報 |

表 8: Secondary_Src の構成要素

| 構成項目 | 項目の値 |
|--------------------|---------------------|
| Keyword | キーワード |
| Author | アノテーションの著者名 |
| Date | アノテーションを行った時点の日付と時間 |
| Communication_Type | アノテーションのリンクの種類 |

前の AnnotateObject のキーワードが継承される。アノテーションツリー情報の個々の AnnotateObject はエントリ番号で管理する。エントリ番号は図 12 に示される規則に従ってつけられる。またアノテーションツリー情報自体についても 1 つの木にそのアノテーションが管理されているアノテーションサーバ内で固有の ID 番号を割り当て、木が更新されるたびに ID が 1 増える。1 つの木に対して同じ ID 番号が割り当てられることはなく、その ID 番号の空間が完全に消費されない限り一意である。ID 番号の空間が消費された場合は 0 にリセットされ、再度同じことが繰り返される。ID 番号の一意性は上位 8 ビットにエントリとしての番号、下位 8 ビットにそのバージョン番号が割り当てられる。アノテーションツリー情報の構成を表 10 に示す。また、Annotation_Tree のデータを図 12 を例にして表 11 に示す。

表 9: Primary_Src の構成要素

| 構成項目 | 項目の値 |
|------------------|--|
| Obj_Type | アノテーションのデータタイプ (・Text_Obj ・Image_Obj |
| Obj_Content_Type | 各データタイプにおけるフォーマット |
| Obj_Location | アノテーションの実体の場所 (例 :/alpha214.aist-nara.ac.jp/~tatuji-u/) |

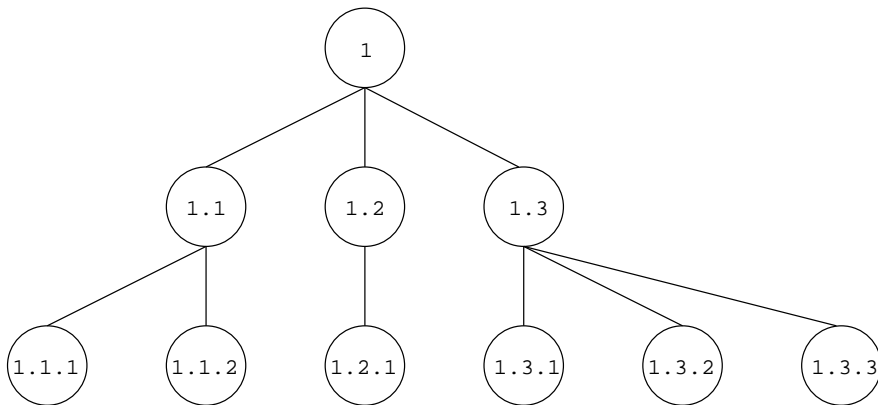


図 12: エントリ番号の規則

表 10: Annotation_Tree の構成項目

| 構成項目 | 項目の意味 |
|-----------|------------------------------------|
| Tree_ID | アノテーションツリーの ID 番号 |
| Record_ID | 木における AnnotateObject の登録番号 |
| Tree_LMC | 子の最左の登録番号 |
| Obj_Entry | AnnotateObject のエントリ番号 |
| Tree_LMS | 1 つ右の兄弟である AnnotateObject の登録番号 |

表 11: Annotation_Tree のデータ例

| Tree_ID | Record_ID | Tree_LMC | Obj_Entry | Tree_LMS |
|---------|-----------|----------|-----------|----------|
| 100001 | 1 | 2 | 1 | - |
| | 2 | 5 | 1.1 | 3 |
| | 3 | 7 | 1.2 | 4 |
| | 4 | 8 | 1.3 | - |
| | 5 | - | 1.1.1 | 6 |
| | 6 | - | 1.1.2 | - |
| | 7 | - | 1.2.1 | - |
| | 8 | - | 1.3.1 | 9 |
| | 9 | - | 1.3.2 | 10 |
| | 10 | - | 1.3.3 | - |

4.2 データモデルの考察

本節では定義したデータモデルについての考察を行う。特にアノテーションに対するアノテーションを木構造で管理する有効性を示す。

4.2.1 アノテーションに対するアノテーション

第 4.1 節のデータモデルではアノテーションに対するアノテーションを実現するために、個々の `AnnotateObject` 間を木構造で管理している。この他にアノテーションに対するアノテーションを可能にするデータの形式としては、オリジナル空間情報の情報空間にアノテーションの情報空間を与え、ユニットにアノテーションを記述する対象となる元のアノテーションを割り当てることも考えられる。図 13 でその一例を示す。

```
Information_Space = ANN
Information_ID = ANN.aist-nara.ac.jp:tatuji-u:/,
               ~tatuji-u/annotation1
```

図 13: オリジナル空間情報の例

ここで、`AnnotateObject` を木構造で管理する方法と比較検討を行う。木構造で管理する場合の利点は、以下のことが考えられる。

- 個々の `AnnotateObject` の関係を容易に把握することができる。1 つのツリーをまとめてアノテーションブラウザに引き渡すことにより、アノテーションブラウザでもそれらの関係を可視化する際に有効である。
- アノテーションを消去した際に、`AnnotateObject` 間の変化を容易に変更できる。これは木を再構成することで可能である。
- 1 回のアノテーションの取得操作であるユニットに対するアノテーションを全て取得することができる。

以上の利点に比べ、オリジナル空間情報にアノテーションの情報空間を指定する場合の利点は、ネットワークトラフィックが多少軽減する程度だと思われる。しかし、木構造で管理する方においてもそれぞれのアノテーションのデータには実データは含まれておらず、転送するデータ量も全体を通して見れば微妙に多くなる程度である。アノテーションのデータに実データを含む設計をしておれば、後者の方がネットワークトラフィックをかなり軽減するが、本システムのモデルでは、木構造で管理する方が利点が多いと考えられる。

また、このモデルは1つのユニットに対する個々のアノテーションの履歴を容易に管理できるため、電子出版物の査読システム等へ応用が可能である。

4.2.2 ユニットの適合度

ユニットにおいて、適合度の考慮は極めて重要である。例えば、文字列のユニットで”ABCD”と”AB”を同じユニットとするか否かで大きく異なる。

まず、両方を同じユニットとみなす場合について考えてみる。例えば図14の文書に対するアノテーションを記述することを考える。

WWWの日本語化あるいは国際化

現在、LineModeブラウザ、Lynx、tkWWWなどのクライアントは日本語のテキストを表示する事が可能である。また、Mosaic for Mac、Mosaic for WindowsではShift-JISで書かれたデータの場合のみ、日本語の表示が可能であることが確認されている。

(参照データ URL=<http://www.brl.ntt.jp/~takeda/docs/www-intro/>)

図 14: ユニットの例 :元の文書

今、ある利用者がこの文書のタイトル中の”WWW”という文字列に対してアノテーションを記述したとする。このユーザはWWWの背景やシステムの概要などについて自分なりのコメントを記述するかもしれないし、WWWに関するページ

へのリンクを張るかもしれない。そこで別の利用者が、同じタイトル中の”WWWの日本語化”という文字列に対してアノテーションを記述した場合、これら2つのアノテーションを記述する対象はWWWという共通の文字列を含んでいるが、内容は全く別のものかもしれない。文字列のユニットが文脈上等価なものであるかの判断は、提案モデルとは切り離して、独立したエージェントの手助けを得なければならない。このようなエージェントに関する研究は、本研究のねらいから逸脱するため本論では省略する。アノテーションの記述の対象となるユニットにおいては、異なる文字列は明示的に別のアノテーションであることを示すためにデータモデルで定義した。これは情報発信者側の制限の無い情報発信に対し、情報受信側である程度情報をフィルタリングする機能を果たし、利用者が本当に参照したいアノテーションのみを参照できるようにしている。

より多くのアノテーションを参照できるように、ユニットの適合度を緩めずにアノテーションのキーワードとのマッチングやあいまい検索を行う。情報をフィルタリングするかしないかは、利用者側で任意選択が可能であることが望ましい。

5. システムの機能詳細

本章では、提案モデルに基づき設計されたアノテーションシステムの以下の要素機能について述べる。

- 利用者の操作に直接関連する機能
アノテーションの登録、検索、参照、消去のプロセスおよび消去に伴うアノテーションツリーの再構成。
- 利用者に直接関連しない機能
アノテーションキャッシュ、アノテーションの既読管理等。

5.1 アノテーションの登録

アノテーションの登録は共有範囲ごとに異なるアノテーションサーバに登録する(図 15)。

ハイパーテキストブラウザによってアノテーションの対象となるハイパーテキスト文書が参照されている時にアノテーションの登録が可能で、以下の項目の指定が要求される。

1. 共有範囲を指定。
パブリック、グループ(グループ名を含む)、プライベートのいずれか。
2. ユニットを指定。
アノテーションデータのフォーマットに従う(表 5,6)。
3. アノテーションの実データのアドレスを指定(表 9)。
4. キーワードの登録。

アノテーションに対するアノテーションを記述する場合は、アノテーションの参照中に上記の共有範囲、ユニットの指定の代わりに対象となるアノテーション(AnnotateObject のエントリ番号)とリンクの種類(Communication_Type)を指定する。

また、アノテーションの実体は利用者のローカルなストレージに蓄積されるため、アノテーション登録後の編集も可能である。

5.2 アノテーションの検索

アノテーションの検索は大きく分けて次の2種類を用意する。

- 現在参照しているハイパーテキストの文書に対するアノテーションの検索
- 属性によるアノテーションの検索

前者はさらに次の2種類の検索方法を用意する。

- 特定のハイパーテキスト文書に対するアノテーションの検索
- 同一情報に対するアノテーションの検索

それぞれの検索は共有範囲によって異なるサーバに問い合わせに行く (図 16)。

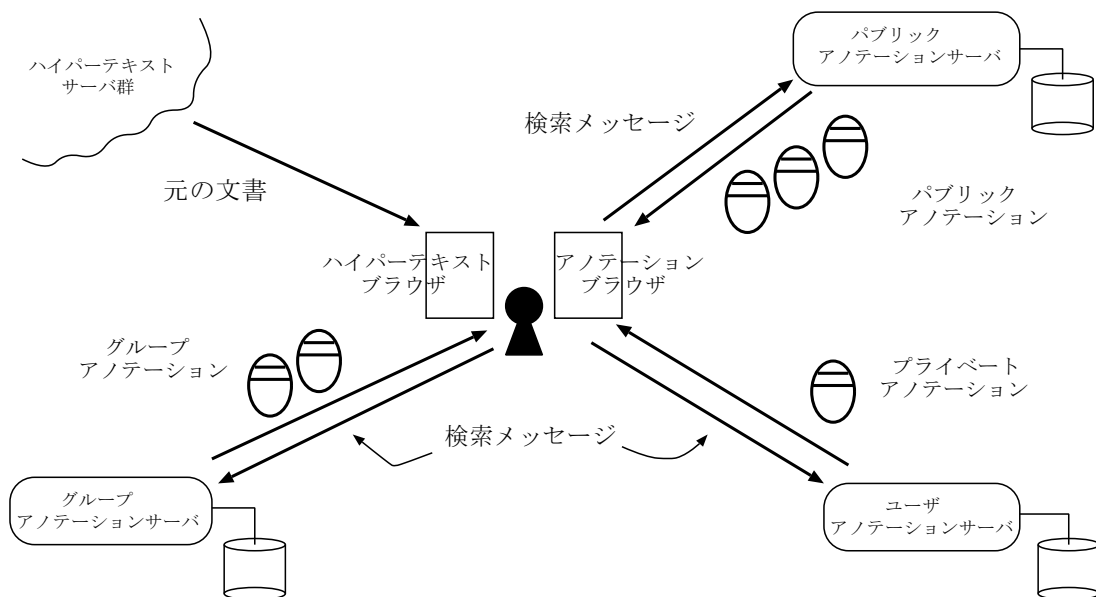


図 16: アノテーションの検索

5.2.1 特定のハイパーテキスト文書に対するアノテーションの検索

各アノテーションサーバで管理するアノテーションデータは全てのユニットに対するアノテーションを情報識別子単位でまとめて管理しているため、アノテーションを普通に検索する場合も情報識別子単位で検索する。各サーバは該当するアノテーションを返す。

ユニット単位の検索も考えられるが、ユニットはアノテーションの著者が任意に指定しているため、元のハイパーテキスト文書を参照しただけでは利用者には分からない。

5.2.2 同一情報に対するアノテーションの検索

同一情報に対するアノテーションの検索は以下の2通り用意する。

1. 個々のアノテーションに付随するキーワードと文書中の文字列のマッチングによる検索。
2. 文字列のユニットと文書中の文字列とのあいまい検索。

これらはいずれも本来別のユニットに対して記述されていたアノテーションを、現在参照している文書中のユニットに対するアノテーションとして見せるための仕組みである(図17)。現在参照中の文書中の文字列と文字列ユニットが同一情報であるかどうかの識別は、前者が利用者がアノテーションを記述する際に入力する任意のキーワードと比較し完全に一致する文字列を同一情報とみなすのに対し、後者はユニットの文字列と文書中の文字列が類似しているものを同一情報とみなす。

前者は検索のメッセージにハイパーテキストの文書中の文字列を含め、これを受け取ったアノテーションサーバが、各アノテーションに付随するキーワードと全文検索を行う。検索対象は第1アノテーションに限定する。ハイパーテキストの文書中の文字列と同じキーワードを所有する `AnnotateObject` を発見した各アノテーションサーバは、アノテーションブラウザへの応答メッセージにこの `AnnotateObject` に付随するアノテーションツリー情報を含めて返す。また、検索結果の有効時間内の応答を可能にするために、1回の検索で結果として返るアノ

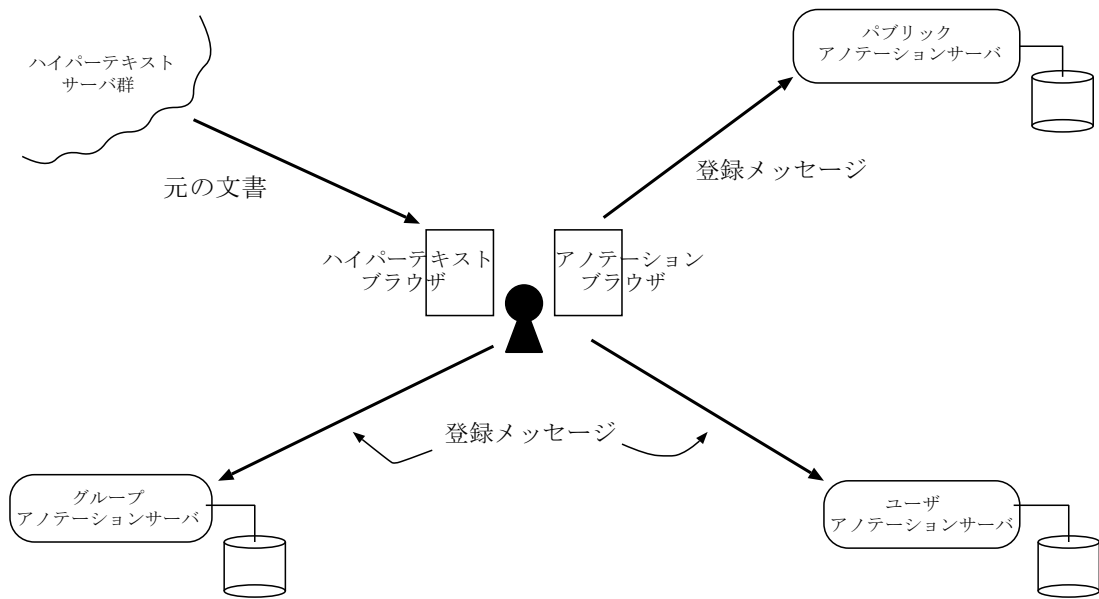


図 15: アノテーションの登録

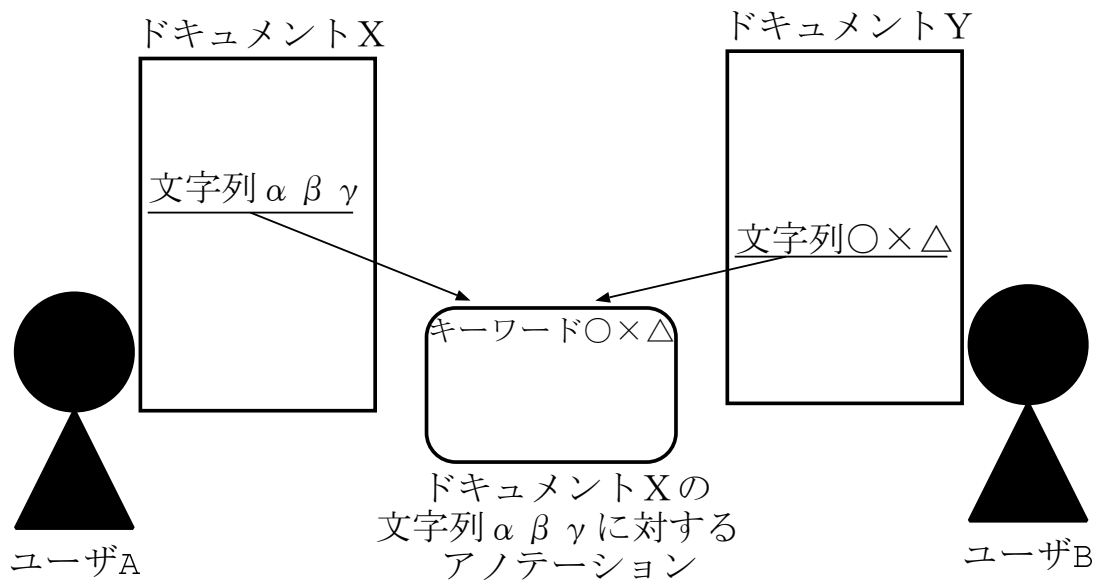


図 17: 同一情報に対するアノテーション

テーションの数を限定する。また、文書中の文字列”ABCD”に対して、キーワード”ABC”がマッチしてしまう問題は、JUMAN[12]等の形態素解析プログラムを利用して品詞分解を行うことにより避ける方法がある。

あいまい検索は各アノテーションデータ中のユニットの文字列をキーワードに、現在参照中の文書に対して全文検索を行う。検索のアルゴリズムは次のような簡単な手法を採用する。

begin あいまい検索の手法

文字列のユニットから2文字ずつ取り出し、2文字の単語列を生成;
検索開始位置を文書の始めに設定;

MAX_score = 0;

while 文書中の全文字列を検索 **do begin**

最初の単語を全文検索;

if 検索にヒット **then begin**

文書中の検索ヒット箇所からユニットの文字列数分だけ2文字ずつ取り出し、ユニットと同様に2文字の単語列を生成;

while 全単語を比較 **do begin**

score = 0;

前方から1単語ずつ比較;

if 1文字のみマッチ **then** *score* = *score* + 1;

if 2文字ともマッチ **then** *score* = *score* + 3

end;

if *score* > *MAX_score* **then** *MAX_score* = *score*

end

end

end.

図18に生成される単語列の例を示す。

文書中の文字列に対して最も得点の高い文字列が、ユニットの文字列と近い文字列とする。利用者は検索にヒットする最低得点を指定し、各アノテーションサー

広域ネットワーク型ハイパーテキストシステム

(元の文字列)



広域 ネットワーク型
ハイパーテキスト
システム (取り出された単語列)

図 18: 生成される単語列の例

は、この最低得点より高い得点を得たユニットに対するアノテーションを返す。

これらの仕組みは、従来元の文書を参照しなければその文書に対するアノテーションを参照することができなかった問題に対する対策として提案する。

5.2.3 アノテーションの属性による検索

ハイパーテキストの情報空間を直接参照しなくても、アノテーションの情報空間を直接検索することが可能である。検索の属性として

- アノテーションに付随するキーワード
- アノテーションの著者
- アノテーションが登録された日付

を指定でき、AND、OR の論理検索を行う。検索を行う利用者は任意の属性の値を入力し、アノテーションブラウザはメッセージに検索の属性値を含めて各アノテーションサーバに送る。各アノテーションサーバは該当するアノテーションを返す。また、返されたアノテーションデータのオリジナル空間情報から元のハイパーテキストの文書の情報識別子が確認でき、ハイパーテキストブラウザと関係してオリジナルにたどることが可能になる。ハイパーテキスト情報空間を検索するための新たな検索手法となる可能性がある。

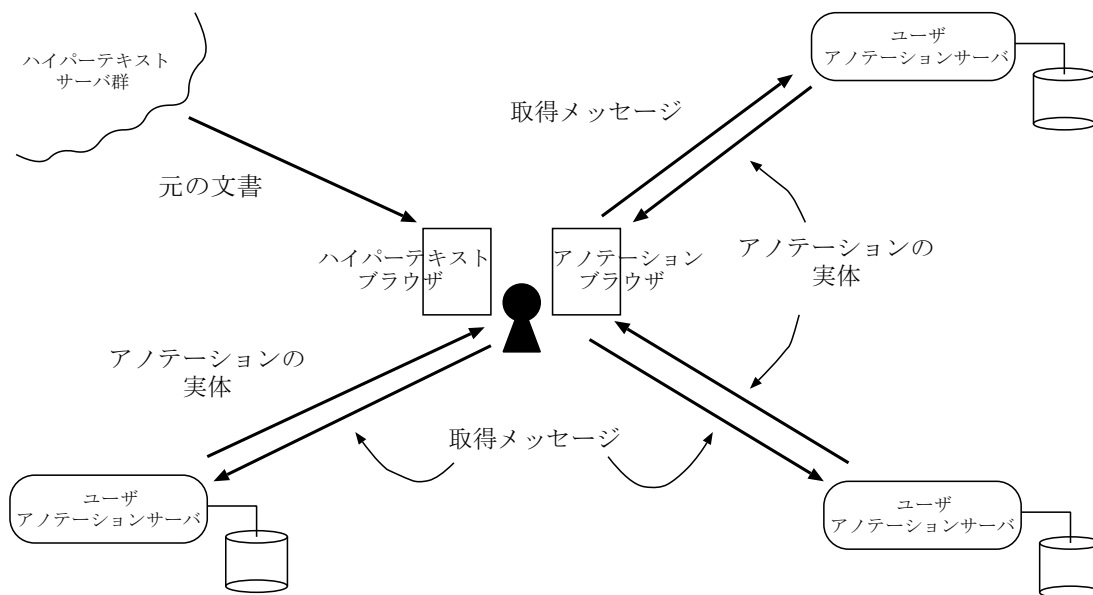


図 19: アノテーションの参照

5.3 アノテーションの参照

アノテーションの検索結果から利用者は参照したいアノテーションを選択する。アノテーションブラウザは選択されたアノテーション (AnnotateObject) の実体をアノテーションの著者のユーザアノテーションサーバに取得しに行く (図 19)。

5.4 アノテーションの消去

アノテーションの消去については以下の場合が考えられる。

- アノテーションを記述した著者が消去する場合。
アノテーションが登録されているアノテーションサーバにアノテーション消去のメッセージを送る。各アノテーションサーバはアノテーションデータのアノテーションツリーの中にある指定された AnnotateObject を消去する。消去後は、一定の法則に従いアノテーションツリーが再構成される。

- 元の文書が消去された場合.
アノテーションは元の文書があり, それに付随して初めて付加価値のあがるものであると一般的に考えられている.
- ディスクが溢れた場合.
アノテーションサーバの管理者が処理する問題である. しかし, ちょっとしたコメントなどは短期間で消去してもらうよう利用者側に要求する必要があるかもしれない.

5.4.1 ハイパーテキスト文書の消去

アノテーションは元の文書があり, それに付随して初めて付加価値のあがるものである. もし, ハイパーテキストの文書が消去されてその文書に対するアノテーションだけが残っていても全く意味のないものとなり得る. 故に元の文書が消去された時にはその文書に対するアノテーションもうまく消去していかなくてはならない. しかし, アノテーションの情報空間はハイパーテキストの情報空間とは独立しているため, アノテーションの対象となるハイパーテキストの文書が消去されたことを自動的に検知するのは困難である.

提案モデルにおけるアノテーションのリンクの方向はアノテーションからハイパーテキストの文書の向きであるため, アノテーションの情報空間からは頻繁にリンク先の文書を調べない限りハイパーテキスト文書の消去ど同期してアノテーションを消去することは不可能である. リンク方式で単方向リンクを採用している現状の WWW においてもリンク先の文書の消去は, 利用者がその文書を参照するためにリンクをたどることによって初めて認識することが可能となる.

従って, アノテーションの情報空間においてもハイパーテキストブラウザとアノテーションブラウザを並行して利用する利用者の行動によってのみハイパーテキスト文書の消去を認識することが可能となる. そこで, ハイパーテキスト文書の削除が明らかになった時点でアノテーションの削除を試みる.

アノテーションブラウザはハイパーテキストブラウザがハイパーテキストサーバから得るステータスコードを読みとり, 消去に対応するステータスコードとハイパーテキスト文書の情報識別子をアノテーションサーバへのメッセージに含め

る. アノテーションサーバは該当するオリジナル空間情報に対するアノテーションの消去に取り掛かる.

5.4.2 アノテーションツリーの再構成

本節ではアノテーションを消去した時のアノテーションツリーの再構成法について説明する.

アノテーションツリーではアノテーションのリンクタイプ(コミュニケーションタイプ)による固有のコミュニケーションの構成の仕方を定義した. アノテーションを消去した場合にも, コミュニケーションのモデルを維持する必要がある. ここでアノテーションの消去は具体的には次の3つを意味する.

1. ノード問題提起の消去
2. ノードコメントの消去
3. ノード賛否の消去

消去の方針は, 3つのノードに優先順位を付け, 消去されるノード以下のノードはこの優先順位に従い, 下位ノードを消去するか残すかを決定する. 優先順位は, 問題提起, コメント, 賛否の順とする. この順ではノード問題提起は一連のアノテーションのコミュニケーションの鍵を握るものであるため, 問題提起の著者の意図以外は消去されない.

まずは最も簡単な3の場合について説明する. ノード賛否にリンクする可能性があるのはノード問題提起のみなので, ノード賛否にノード問題提起がリンクされていれば, そのノード問題提起が本来のユニットに対する第1アノテーションとしてツリーが再構成される.

ノードコメントが消去された場合, これにリンクしているノード賛否は全く意味の無いものとなる. 故に下位ノードの賛否は消去され, それ以降のノード問題提起は本来のユニットに対する第1アノテーションとしてツリーが再構成される.

ノード問題提起が消去された場合, それにリンクしているノードコメントも消去される. それ以降のノード問題提起は以上と同様に第1アノテーションとしてツリーが再構成される.

ただし、第1 アノテーションが消去された時に限りその AnnotateObject を根とするアノテーションツリーは全て消去される。

5.5 共有範囲の変更

ハイパーテキストシステム中の文書の著者が文書を自由に変更、削除できるように、アノテーションの共有範囲も利用者側で任意に変更可能であるのが望ましい。共有範囲の狭いものから広いものへの変更はアノテーションデータへのアクセス制限を緩めることで容易に可能である。その逆も操作は容易であるが、一連のアノテーションツリーの途中のノードが変更された時は意味的に不都合が生じる。そこで、両者ともアノテーションの共有範囲の変更は、アノテーションの削除と登録を繰り返すものとし次のように割り当てる (表 12)。

表 12: 共有範囲の変更と実際の操作

| 共有範囲の変更 | 実際の操作 |
|--------------|-------------------|
| プライベート→パブリック | 新規パブリックアノテーションの登録 |
| パブリック→プライベート | パブリックアノテーションの消去 |
| プライベート→グループ | 新規グループアノテーションの登録 |
| グループ→プライベート | グループアノテーションの消去 |
| パブリック→グループ | 新規グループアノテーションの登録 |
| グループ→パブリック | — |

グループアノテーションからパブリックアノテーションへの変更はグループの方針に反することかもしれないため、この場合の共有範囲の変更機能は取ってはなした。

5.6 アノテーションキャッシュ

ユーザアノテーションサーバは一度取得したデータを一時保管するためのアノテーションキャッシュを持つ。アノテーションキャッシュでは次の各データがキャッ

シユされる。

- パブリックアノテーションサーバのアドレス
- 共有範囲ごとのアノテーション
- それぞれのアノテーションの実体

パブリックアノテーションサーバへの最初のアクセスは利用者が明示的に指定するホスト名により行うが、一度キャッシュされると特に指定する必要はない。パブリックアノテーションサーバが複数存在する場合においても、キャッシュされている複数のホスト名からある異なるコミュニティのパブリックアノテーションサーバのホスト名を知ることができる。

特定のハイパーテキスト文書に対するアノテーションを検索する場合にはアノテーションデータ中の各エントリ番号がアノテーションキャッシュから引き出される。

アノテーションの実体を取得する際はまずアノテーションキャッシュを参照し、そこになれば目的のユーザアノテーションサーバに取得しに行く。利用者が明示的にリロードのアクションを起こさない限り、アノテーションの実体は一定期間アノテーションキャッシュに蓄積される。

アノテーションブラウザは特定の文書に対するアノテーションを検索する際にも、まずアノテーションキャッシュに問い合わせに行く。そこで、同じ文書に対するアノテーションデータが存在する場合は、そのデータを検索メッセージに含め各アノテーションサーバ内で更新されたデータのみを取得する。そのために提案モデルのアノテーションデータでは各要素でエントリ番号、カウンタ、ID 番号を多用している。これらの番号からデータが更新されるようすについて説明する。

各要素データで用いられているカウンタ類をまとめると表 13 のようになる。

実際には表 13 に示されている要素データを検索メッセージに含め、各アノテーションサーバは送信されてきた番号と独自で管理する番号とを比較して、更新されたデータを送信する。次のフローにより更新されたデータを取得する。

if Unit_Count が更新

表 13: 要素データごとの番号

| | 要素データ | 番号の意味 |
|----------------|------------|--------------------------|
| Anno_Data | Data_Entry | 1 文書に対するアノテーションのエントリ |
| オリジナル空間情報 | Unit_Count | 1 文書中のユニットの個数 |
| | Units | Unit_Entry の列 |
| オリジナルユニット情報 | Tree_Count | 1 ユニットに付随するアノテーションツリーの個数 |
| | Trees | Tree_ID の列 |
| アノテーションツリー情報 | Tree_ID | 個々のアノテーションツリーに固有の ID |
| AnnotateObject | Obj_Entry | AnnotateObject のエントリ |

then 増えた Unit_Entry に付随する全てのデータを取得
if Tree_Count が更新
 then 増えた Tree_ID に相当するアノテーションツリー情報を取得
if Tree_ID が更新
 then 更新された Tree_ID に相当するアノテーションツリー情報を取得

各データの ID やカウンタを比較することにより、分散された Group Annotation Server 間におけるアノテーション情報の一貫性を保つことも可能となる。

5.7 アノテーションの既読管理

アノテーションによっては1度見れば次からは見る必要の無いもの、何回でも見たいものがあると考えられる。しかし、それはアノテーションを参照する利用者の立場によってそれぞれ異なると思われる。故に1度参照したアノテーションをもう一度参照するか否かは利用者の判断に委ねるのが望ましい。

提案モデルにおいてはアノテーションの実体はアノテーションを記述した著者

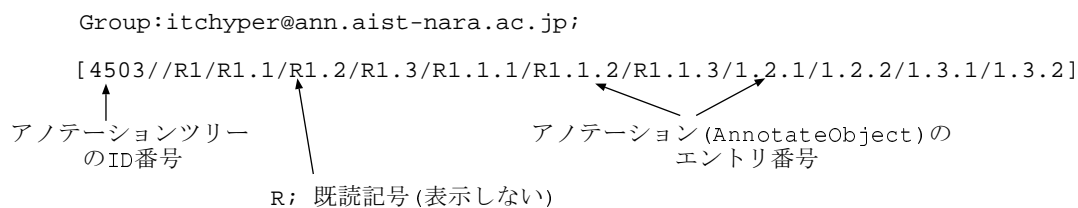


図 20: 既読管理ファイルの例

のローカルなディスクに蓄積されているため、アノテーションを参照する側では見たい時に必要に応じてアノテーションの実体を取得しに行くことになる。故にアノテーションを参照する利用者が1度見たアノテーションを見るか否かは実際にアノテーションの実データを取得するイベントをアノテーションブラウザに対して引き起こすか否かで決定されることになる。

しかし、アノテーションの検索結果はノーマルのモードであれば現在参照しているハイパーテキスト文書の情報識別子に対するアノテーションのリストが返って来ることになる。このリストは時には膨大な数になることも考えられ、そのリストを見るだけでも利用者のストレスを溜らせ、非常に使い勝手の悪いものにもなり得る。この問題を解決するために、NetNewsのようにアノテーションブラウザ側でもアノテーション参照の履歴管理を行う情報を蓄積する必要がある。利用者は既読管理ファイルを編集することによって表示の有無の指定が可能である(図20)。

6. プロトタイプの実装

アノテーションを複数人で共有し、アノテーションのコミュニケーションモデルを評価するために WWW を対象としたプロトタイプを実装した。WWW を対象とするのは、Mosaic や Netscape 等のブラウザの登場により爆発的な人気を誇っており、現状においては広域ネットワーク型ハイパーテキストシステムにおいて最も利用者が多いと考えられるからである。プロトタイプとしてはとにかく多くの利用者が馴染みにのあるシステムの枠組の中で運用することが適切である。

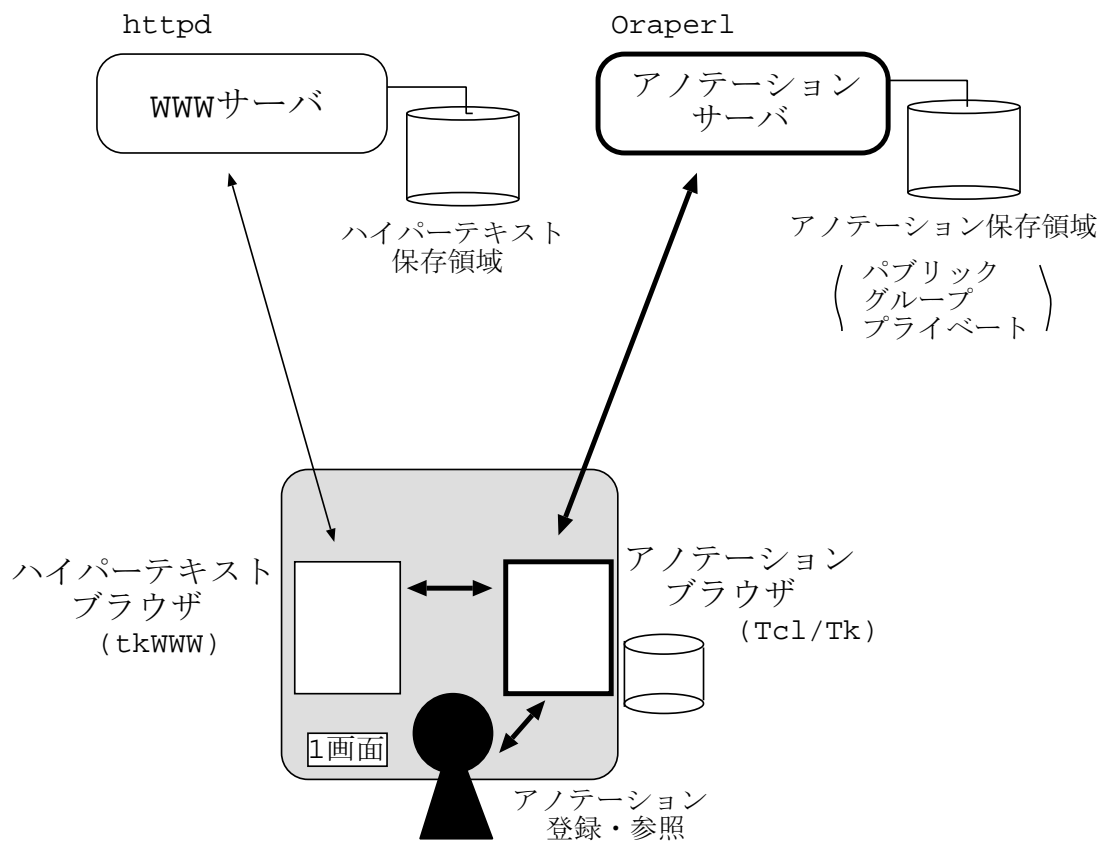


図 21: プロトタイプシステムの構成

プロトタイプの構成を図 21 のようにし、プラットフォームこのプロトタイプは本

学, 奈良先端科学技術大学院大学内において試験的に運用することから考える. このプロトタイプでは, 「同一情報に対するアノテーションの検索」を行うことは考慮せず, アノテーションの登録, 検索, 参照等基本的な機能のみを与える. 今回必要となるシステム要素は以下の通りである.

ハイパーテキストサーバ 現状の httpd をそのまま利用する. ハイパーテキストブラウザが要求する WWW の文書を提供する.

アノテーションサーバ プロトタイプにおいては, パブリックアノテーション, グループアノテーション, プライベートアノテーションおよびそれらの実体そのもの全てをまとめて管理するサーバ(アノテーションサーバ)を一つだけ用意する. アノテーションを木構造で管理し, アノテーションブラウザが要求する URL 文書に対するアノテーションを提供する. また, パスワードによるユーザの認証も行う.

ハイパーテキストブラウザ WWW の文書を参照するためのブラウザ. 参照中の文書の URL をアノテーションブラウザに通知する. また, アノテーションの記述の対象となるユニットを指定する. ユニットが指定されると文書内にタグ付けを行い, アノテーション参照時には指定されたユニットをアノテーションブラウザに通知する.

アノテーションブラウザ アノテーションを登録, 検索, 参照するための Tcl/Tk プログラム. プロトタイプシステムでは次の操作が可能である.

- アノテーションの登録
アノテーションのコミュニケーションモデルに基づく, アノテーションに対するアノテーションの記述機能も実装.
- アノテーションの検索
 - 特定の URL に対するアノテーションの検索
 - アノテーションの属性による検索

検索結果は木構造で表示.

- アノテーションの参照
- アノテーションの消去

7. むすび

本論文では、広域ネットワーク型ハイパーテキストシステム上の文書に対して記述されたアノテーションの複数人での有効活用を可能にするアノテーションサービスモデルの提案を行った。ハイパーテキスト情報空間とアノテーション情報空間を分離するためのアノテーションシステム的设计、アノテーションデータモデルの定義を示し、効率の良い協調作業を実現する可能性を示した。また、WWWを対象にプロトタイプの実装を行った。

提案モデルにより、複数人かつ複数ハイパーテキストシステム間でのアノテーションの共有が可能になり、従来のアノテーションシステムよりも共有範囲が拡大された。また、アノテーションを単独に検索したり、同一情報に対するアノテーションの検索手法を提案することにより、元の文書を参照しなくてもその文書に対するアノテーションを参照することができ、アノテーションの再利用性の向上に役立つと考えられる。アノテーションに対するアノテーションの記述では、一連のアノテーションをツリー構造で管理し、ツリーのノードおよびノード間の関係を定義することによって、利用者がアノテーションに関する操作を自在に行えるようになった。この一連のアノテーションツリーの共有が有効なコミュニケーションを可能にすると考えられる。さらに、アノテーションはアノテーションの記述の対象となる文書中の素材単位で管理しているため、アノテーションが記述される文書中の位置を明確にし、文字列以外のアノテーションの記述ができるため、ブラウザの開発が容易になると考えられる。

今後の課題であるが、実装したプロトタイプシステムの評価手法を検討し、提案モデルの評価を行うことが挙げられる。アノテーションの記述の対象となるユニットの文字列が日本語に代表されるマルチバイト文字列やアラビア語のように右から左へ書かれる言語に対しても正確な位置情報の取得について考慮する。また、現状の広域ネットワーク型ハイパーテキストシステムにおいても課題である情報の更新、消去時に情報空間の無矛盾性を維持する仕組みについて検討する必要もある。

上記の課題を考慮しつつ、かつこの評価実験の結果に基づく広域ネットワークにおける実装も今後の課題にあげられる。

謝辞

本研究の遂行中、終始理解ある御指導と激励を賜りました奈良先端科学技術大学院大学 情報科学研究科の横矢直和教授、千原國宏教授ならびに竹村治雄助教授に深く感謝致します。

また、本研究中、常に適切な御指導、御助言をいただき研究の達成へと導いてくださった、奈良先端科学技術大学院大学 情報科学研究科 馬場始三助手、岩佐英彦助手、そして奈良先端科学技術大学院大学 情報科学研究科 ソフトウェア基礎講座の皆さんに深く感謝致します。

参考文献

- [1] Ken Harrenstien and Vic White. Nicname/whois. Network Working Group – Request for Comments, Mar. 1982. RFC-812.
- [2] M. St. Pierre, J. Fullton, J. Goldman, et al. Wais over z39.50-1988. Network Working Group – Request for Comments, June 1994. RFC-1625.
- [3] Tim Berners-Lee, R. Cailliau, J-F Groff, and B. Pollermann. World-Wide Web: The information universe. *Electronic Networking: Research, Applications and Policy*, Vol. 2, No. 1, pp. 52–58, Spring 1992.
- [4] Wayne C. Gramlich. Annotation system issues, surveys many annotation issues. [http://playground,sun.com:80/~gramlich/1994/annotate/](http://playground.sun.com:80/~gramlich/1994/annotate/), 1994.
- [5] Frank Kappe. Hyper-g: A distributed hypermedia system. In *INET'93*, pp. DCC1–DCC9, Aug. 1993.
- [6] Daniel LaLigerte. Welcome to hypernews. <http://union.ncsa.uiuc.edu/HyperNews/get/hypernews.html>.
- [7] Jim Davis. Conote(annotation)homepage. <http://dri.cornell.edu/pub/davis/annotation.html>.
- [8] Martin R Cheisen, Christian Mogensen, and Terry Winograd. A platform for third-party value-added information providers: Architecture, protocols, and usage examples. Technical report, Stanford Integrated Digital Library Project, Coumputer Science Dept., Stanford University, Nov. 1994.
- [9] Welcome to the japan network information center(jpnic) webserver. <http://www.nic.ad.jp/>.
- [10] Kazuhiko Yamamoto. Pgp mime integration. Internet-Draft, Oct. 1995. draft-kazu-pgp-mime-00.txt.

- [11] Conklin J. and Begeman M. L. gibis: A hypertext tool for exploratory policy discussion. *ACM Transactions on Office Information Systems*, Vol. 6, No. 4, pp. 303–331, Oct. 1988.
- [12] 松本裕治, 黒橋禎夫, 宇津呂武仁, 妙木裕, 長尾真. 日本語形態素解析システム juman 使用説明書 version2.0. Technical Report NAIST-IS-TR94024, 奈良先端科学技術大学院大学, July 1994.
- [13] Daniel LaLiberte and Alan Braverman. A protocol for scalable group and public annotations. <http://union.ncsa.uiuc.edu/~liberte/www/scalable-annotations.html>, 1995.
- [14] Martin R Cheisen, Terry Winograd, and Andreas Paepcke. Content rating and other third-party value-added applications for the workd-wide web. In *D-Lib Journal*. CNRI, Aug. 1995.
- [15] Martin R Cheisen, Christian Mogensen, and Terry Winograd. Beyond browsing: Shared comments, soaps, trails, and on-line communities. <http://www-diglib.stanford.edu/diglib/pub/reports/brio-www95.html>, apr 1995. Paper presented at the Third International World-Wide Web Conference in Darmstadt, Germany.
- [16] Martin R Cheisen, Christian Mogensen, and Terry Winograd. Interaction design for shared commenting. In *CHI 1995*, Denvor, 1995.
- [17] 坂本佳則. Dexter モデルに基づいた広域ハイパーテキストシステム. 奈良先端科学技術大学院大学情報科学研究科 平成 6 年 度修士学位論文, Feb. 1995.
- [18] Rob McCool. The common gateway interface. <http://hoohoo.ncsa.uiuc.edu/cgi/>, Sept. 1994.
- [19] Paulette Bush Timothy Catlin and Nicole Yankelovich. Internote: Extending a hypermedia framework to support annotative collaboration. In *Hypertext*

'89 Proceedings, pp. 365–378, Nov. 1989.

- [20] N. Borenstein and N. Fread. Mime(multipurpose internet mail extensions):mechanisms for specifying and discribing the format of internet message bodies. Network Working Group – Request for Comments, June 1992. RFC-1341.
- [21] Peter J. Denning and Bernard Rous. The acm electronic publishing plan. *Communications of the ACM*, Vol. 38, No. 4, pp. 97–103, Apr. 1995.

付録

A. プロトタイプシステムのプロトコル仕様 (アノテーションサーバ-アノテーションブラウザ)

本章ではプロトタイプシステムにおけるアノテーションサーバ, アノテーションブラウザ間のプロトコル仕様について述べる. また, 以下ではアノテーションサーバを AS, アノテーションブラウザを AC, ハイパーテキストブラウザを HC と呼ぶことにする

A.1 ステップ

1. ap.aist-nara.ac.jp への 4000 ポートへの接続
2. 認証および権限取得
3. 登録 / 消去 / 検索 / 取得
4. 終了

A.2 認証および権限取得

何も権限取得されなければパブリックアノテーション情報のみを利用することができる. プライベートおよびグループアノテーションを行なう場合は, それぞれに対して認証を行ない, 権限取得する必要がある. 権限を取得するタイミングは通信中いつでも問題ない. 認証が終了した時点からこれまで隠蔽されていたアノテーション情報へアクセス可能となる. また, プライベートは一つのみ, また グループは複数の権限を取得することができる. プライベートおよびグループの名前空間は, 一意性が高くなるように, FQDN 方式のホスト名を@をはさんでつけることを強く推薦する.

A.2.1 `private annotation` を可能とするために

仕様 (要求) USER ユーザ名 パスワード
(返答) 100 Login succeeded.
(返答) 401 Login incorrect.

USER コマンドで `tatuji-u` のパスワード認証を行なう例を示す.

```
» USER tatuji-u@is.aist-nara.ac.jp パスワード
« 100 Login succeeded.
```

A.2.2 `group annotation` を可能とするために

仕様 (要求) GROUP グループ名 パスワード
(返答) 100 Login succeeded.
(返答) 401 Login incorrect.

まず利用するアノテーションサーバにおいて、どのようなグループが利用可能かを知る必要がある. そのために利用するのが `GROUP LIST` である.

```
仕様 (要求) GROUP_LIST
(返答) 100 グループ名 (1)
      100 グループ名 (3)
      100 .....
      .
```

```
» GROUP_LIST
« 100 ali-baba
```



```

<< 100 wwhs
<< 100 yokoya-lab
<< .

```

次に, 得られたグループリストから利用したいグループを選び, パスワード認証を行う.

```

>> GROUP ali-baba open_sesame
<< 100 Login succeeded.

```

A.2.3 現在どの範囲で `annotation` を検索しているのか

```

仕様 (要求) PERMISSION
(返答) 100 グループ/プライベート名 (1) ON
        100 グループ/プライベート名 (2) ON
        100 .....
        .

```

複数のグループやプライベートと認証をパスすると, 複数の権限でアノテーションデータベースへアクセスすることができる. そこで, 現在どの権限をパスしているのか知るためのコマンドが `PERMISSION` である.

```

>> PERMISSION
<< 100 ali-baba ON
<< 100 tatuji-u@is.aist-nara.ac.jp ON
<< 100 public ON
<< .

```

`active` になっているグループおよびプライベートを返す. ちなみにパブリックは常に `active` である.

A.2.4 active な group/private を inactive にするために

仕様 (要求) USER ユーザ名 パスワード OFF
(返答) 100 Logoff succeeded.

仕様 (要求) GROUP グループ名 パスワード OFF
(返答) 100 Logoff succeeded.

いったん認証に成功して active になった group/private だが, 一々検索時に検索されるアノテーションが多過ぎて困る場合は, inactive にする必要がでてくる (検索範囲を絞るとのこと) .

```
>> baba@is.aist-nara.ac.jp ue off
>> USER
<< 100 Logoff succeeded.
>> GROUP ali-baba open_sesami
<< 100 Login succeeded.
```

A.3 登録

アノテーション情報の登録を行う. アノテーション情報を登録するためには, まずアノテーションをおこなうハイパーテキスト文書のユニットを指定しなければならない. したがって, AC にてアノテーションを登録する手順は次のようになる.

1. アノテーションする (文書, ユニット) を AS へ申請
2. AS でユニットを作成し, ユニット ID を AC へ返す
3. AC でユニット ID を利用して, アノテーションオブジェクトを AS へ申請

4. AS から指定された (文書, ユニット) に対して TREE 構造で登録された アノテーションオブジェクト ID を AC へ返す. この時, TREE 構造を示す情報は再構築されないとはいえないが, その再構築を行なうのは AS 側で行なう. したがって, アノテーションオブジェクトを登録するたびに AS から AC へ返されるのは,

- ユニット ID
- アノテーションオブジェクト ID から構成される TREE 情報となる

5. AC は AS から返された TREE 情報をもとに, 指定されたユニットに関して TREE を再描画する.

AC 側でユニットが存在していないものに対して, アノテーションオブジェクトを作成しない. AS は, ユニット ID が存在しないアノテーションオブジェクトは作成せずに登録エラーを AC へ返す.

データベースとのトランザクションを考えた結果, AS からアクセスするデータベース側では次のテーブルへ情報を分けて管理する.

1. アノテーションカプセル表
2. アノテーションユニット表
3. アノテーションオブジェクト表
4. アノテーション TREE 表

1 カプセルには複数ユニットが登録される. また, 1 ユニットには複数オブジェクトが登録される. また, 1 ユニットに属するオブジェクト間には TREE (親子) 関係がある. 検索効率を考えると, オブジェクト内に登録される実際のアノテーション情報を構成する一次情報と二次情報を分けて管理してもよかったが, アノテーション情報登録のためのトランザクションを 1 トランザクションで終了させたかったため, あえて, 一つの表にまとめる. TREE 表には, 1 オブジェクトごとに 1 つの親と 1 つの子を管理させる. また, 仕様では 1 ユニットには複数の TREE が存在

し得るため, TREE 表には, 一意の TREE id を用意する. そして, TREE (親子) 関係は, TREE id ごとに一意となるように管理される.

基本的なプロトコル仕様は次の通り.

REGIST {unit|object} subcommand data\n

unit/object : unit もしくは object (annotation) として登録

subcommand : 登録時のデータ項目ごとに用意される

data : 登録されるデータ

返り値として AS から AC に対してデータを受信するごとに, 次のデータを返します.

100 ACK

一方, AS 側のデータ受信時に何か問題があれば, 次のデータを返す. なお, このエラー時の返り値は, エラー情報が細かく設定できるたびに増えていく. このエラーが返ってきた場合は, 登録作業を最初からやり直す必要がある.

402 NACK (今回の場合は何らかのエラー情報)

以降では実際に登録する手続きにしたがって, 順に説明する.

A.3.1 登録 1,2

- (1) アノテーションする (文書, ユニット) を AS へ申請
- (2) AS でユニットを作成し, ユニット ID を AC へ返す

HC におけるアノテーション作成イベントをうけて AC では AS に対してユニット ID を申請する. 今回の AC の仕様では, すでに内部で持っているユニット ID であればわざわざ AS まで問い合わせない. したがって, AS 側へ送られてくる

REGIST unit subcommand data\n

プロトコルは, ユニットの新規登録もしくは, まだ AC で管理していないユニット ID の取得となる.

AS 側の振舞いは, 送られてきた (ハイパーテキスト文書, ユニット情報) をもとに, まずアノテーションカプセルが存在しているかどうか, 調べる. カプセル表

に対して送られてきた情報の中から必要な検索条件を抽出して検索を行ない、カプセルが存在していれば、そのカプセル ID を持つユニット群に対してさらに申請されたユニットが存在しているかどうか調査する。

ユニットが存在しなければ、既存のカプセルに対して新しいユニットを作成する。一方存在していればそのユニット ID を AC へ返す。

```

>> REGIST unit scope data\n
<< 100 ACK
>> REGIST unit space data\n
<< 100 ACK
>> REGIST unit content_type data\n
<< 100 ACK
>> REGIST unit url data\n
<< 100 ACK
>> REGIST unit regist\n
<< 100 unit_id
or
<< 402 various error message caused by Oracle7 via Oraperl
```

1. 共有範囲 (scope)

```
REGIST unit scope data\n
data := {public|group_name|private_name}
(例) REGIST unit scope public
(例) REGIST unit scope ali-baba
(例) REGIST unit scope tatuji-u@is.aist-nara.ac.jp
```

2. オリジナル空間情報 (space)

```
REGIST unit space data\n
data := {www|hyper-g|netnews|mail}
(例) REGIST u_space www
今回は www のみのサポートとなる。
```

3. オリジナルユニットのコンテンツ種別

```
REGIST unit content_type data\n
data := {text/html|image/gif|image/jpeg|...}
(例) REGIST unit content_type text/html
今回はほとんど text/html と思われる.
```

4. ユニット位置 [図=(x,y) テキスト=(start bytes, end bytes)]

```
REGIST unit position data\n
data := [0-9]+ [0-9]+
(例) REGIST unit position 10 14
なお, 特別な例として, position の data が (0,0) であれば, ユニットを含むハイパーテキスト文書そのものを示すユニットであるとする.
(例) REGIST unit position 0 0
```

5. ユニットの实体位置情報 (URL)

```
REGIST unit url data\n
data := [a-zA-Z]+
(例) REGIST unit url http://www.aist-nara.ac.jp/index.html
```

6. ユニット ID 申請

```
REGIST unit regist\n
data := なし
(例) REGIST unit regist
```

これまで AC から AS へ送信したさまざまなユニット情報をもとにユニット ID を申請する. この申請を行なわない限り, ユニット ID は AC 側で取得できない. AS 側での既存のユニット ID の検索, および新規ユニット ID の作成に成功すれば, ID を AS から以下の形式で返す.

```
100 unit_id\n
```

なおこの unit_id は AS のデータベース内で一意の数字である.

A.3.2 登録 3,4

(3) AC でユニット ID を利用して, アノテーションオブジェクトを AS へ申請

(4) AS からユニット ID およびオブジェクト TREE を AC へ返答

ユニット ID を無事に取得した AC は, その ID をもとにアノテーションオブジェクトの作成に移る. もともとそのユニット ID を取得していた場合は先の (1)/(2) のステップを省略する.

```

>> REGIST object unit_id data\n
<< 100 ACK
>> REGIST object keyword data\n
<< 100 ACK
>> REGIST object author data\n
<< 100 ACK
>> REGIST object status data\n
<< 100 ACK
>> REGIST object tree data\n
<< 100 ACK
>> REGIST object parent data\n
<< 100 ACK
>> REGIST object encoding data\n
<< 100 ACK
>> REGIST object body data1\n
<< 100 ACK
>> REGIST object body data2\n
<< 100 ACK
>> REGIST object body ..... \n
<< .....
>> REGIST object body dataN\n
```

```

<< 100 ACK
>> .
<< 100 ACK
>> REGIST object regist\n
<< 100 unit_id tree_id object_tree.information
or
<< 402 various error message caused by Oracle7 via Oraperl

```

1. ユニット ID (アノテーションオブジェクトを管理するユニット)

```
REGIST object unit_id data\n
```

```
data := [0-9]+
```

(例) REGIST object unit_id 10001

ユニット ID は, 既に登録済のユニットに対する登録操作 (append) の場合に必要となる.

2. keyword (二次情報)

```
REGIST object keyword data\n
```

```
data := [A-Za-z0-9]+
```

(例) REGIST object keyword hypertext

現在の実装段階では, 複数入力は不可. また ”” による空白を含めることもできない.

3. author (二次情報)

```
REGIST object author data\n
```

```
data := [A-Za-z0-9]+
```

(例) REGIST object author tatuji-u

4. status (コミュニケーションのステータス)

```
REGIST object status data\n
```

```
data := { 問題提起 (issue)| コメント (comment)| 賛否 (object)}
```

(例) REGIST object status 問題提起

5. tree (アノテーションオブジェクトが属する TREE id)


```
REGIST object tree data\n
```

data := 0 (将来の拡張のために用意されたもの. 現在 0 のみ)

(例) REGIST object tree 0

将来, 1 ユニット内で複数 TREE 持てるようになった場合に備えて, TREE id 情報の受け渡しを行う. 現在 0 を入力する.
6. parent (親ノードにあたるアノテーションオブジェクト ID)


```
REGIST object parent data\n
```

data := objecttation_object id

(例) REGIST object parent 10021
7. encoding (アノテーション情報本体の符号化方式)


```
REGIST object encoding data\n
```

data := {uuencode|base64}

(例) REGIST object encoding uuencode
8. body (アノテーション情報本体)


```
REGIST object body data1\n
```

```
REGIST object body data2\n
```

```
REGIST object body ....\n
```

```
REGIST object body dataN\n
```

```
.\n
```

data1..dataN := アノテーション情報 (図+テキスト)

(例) REGIST object body あいうえおかきくけこ

```
REGIST object body さしすせそたちつてと
```

```
REGIST object body なにぬねのはひふへほ
```

.
9. オブジェクト ID 申請


```
REGIST object regist\n
```

data := なし

(例) REGIST object regist

これまで AC から AS へ送信したさまざまなオブジェクト情報をもとにオブジェクト ID を申請する. 同時に再構築された TREE 情報を求める. この申請を行わない限り, 新規オブジェクト ID は AC 側で取得できない. AS 側での既存の TREE ID の検索, および新規ユニット ID の作成に成功すれば, ID を AS から以下の形式で返す.

```
100 unit_id tree_id TREE\n
```

なおこの unit_id/tree_id は AS のデータベース内で一意の数字となる.

A.4 消去

アノテーション情報の削除を行う.

A.4.1 アノテーションオブジェクトの削除

```
>> DELETE object object_id\n
```

```
<< 100 unit_id tree_id TREE
```

or

```
<< 403 NACK
```

AS 側で TREE を再構成するため, tree_id および TREE 情報を返す. 現在の設計では tree_id は 0 しか返さない.

A.4.2 アノテーションオブジェクト TREE の削除 (将来の拡張)

```
>> DELETE tree tree_id\n
```

```
<< 100 ACK
```

or

```
<< 403 NACK
```

AS 側でまだサポートされていない.

A.4.3 アノテーションユニットの削除

```
>> DELETE unit unit_id\n<< 100 ACK  
or  
<< 403 NACK
```

A.4.4 アノテーションカプセルの削除 (将来の拡張)

```
>> DELETE capsul capsul_id\n<< 100 ACK  
or  
<< 403 NACK
```

A.5 取得

アノテーション情報の取得を行う。具体的には、AC 側で表示されるアノテーションオブジェクト TREE のノードをマウスでクリックした場合に、ノードに相当するアノテーション情報の実体を AS から取得しなければいけない。取得するための方法として次の方法をとる。

1. HC で見ている文書にアノテーションが存在しているか？
2. AC から URL をもとに AS へ問い合わせる
3. AS から存在するユニット ID を全て返す
4. ユニットごとにオブジェクト (TREE) 情報を AS へ問い合わせ
5. 得られた UNIT/TREE 情報をもとに AC で UNIT/TREE 描画
6. TREE 中のノードをクリックした際にオブジェクトを取得

```
GET capsule url\n
```

capsule : ハイパーテキスト文書単位の情報を知りたい

url : 文書を示す URL

返答 : unit_id のリスト

GET unit unit_id\n

unit : ユニット単位のオブジェクト TREE 情報を知りたい

unit_id : unit_id

返答 : 指定されたユニットの TREE 情報

GET object object_id\n

object : ユニット単位の

オブジェクト TREE 情報を知りたい

object_id : object_id

返答 : 指定されたオブジェクトのアノテーション情報

» GET capsule http://www.aist-nara.ac.jp/index.html

« 100 unit_id(1)

« 100 unit_id(3)

« 100 unit_id(...) « 100 unit_id(N)

« .

or

« 405 not found

» GET unit unit_id

« 100 tree_id TREE

or « 405 not found

» GET object object_id

« 100 keyword data

« 100 author data

« 100 date data

« 100 status data

```

<< 100 encoding data
<< 100 body data1
<< 100 body data2
<< 100 body .....
<< 100 body dataN
or
<< 405 not found

```

今回の実装では行わないが、アノテーション情報の実体が図などのマルチメディア情報を含むためにデータ量が巨大な場合、64 bytes ごとに分けて AS から AC へ送信される。また、バイナリ情報を含む場合は、何らかのエンコーディング方式 (uuencode/base64) で符号化されているものとする。

A.6 終了

終了するためのコマンドは QUIT.

仕様 (要求) QUIT

QUIT を入力すると、即サーバとの接続は切断される。