平成 20 年度

卒業論文

研究記録管理・公開・検証システム arχves の数式およびグラフへの対応

指導教員　小林　聡　准教授

島根大学　総合理工学部
数理・情報システム学科　情報工学コース
S043052　島貫稚華

目次

1 はじめに

昨今、研究成果の捏造が社会問題となったことは記憶に新しい。技術的にこれらの問題の発生を食い止める事は困難であるが、改竄が困難な手法によって研究記録が保存されているならば、少なくとも研究記録の検証に関しては大いに助けになるであろう。この際、研究記録の真正性と非改竄性の保証をいかに行うかが課題となる。

このような課題に対して、原田らは電子カルテを念頭に、モデルの提案を行っている[1]。また、陳らはXML文書の時刻認証を伴った管理を試みている[2]。同様なサービスとして、SNS/blog機能に電子文書へのタイムスタンプ付加機能を加えた、SNS/blogサービスも存在する[3]。しかし、これは日本電子公証機構のサービスを利用しているため、比較的高額なサービスとなっている。このような、タイムスタンプを活用したサービスはDVCS（Data Validation &Certification Server Protocol）やTAP（Trusted Archival Protocol）などに分類される。DVCSは、電子文書のアーカイブは行わないことを原則としており、対してTAPはアーカイブも行うことを原則としている。また宇根らはDVCSやTAPで用いられる各種タイムスタンプによる可用性・安全性に関して報告している[4]。

また、今日、e-mailやWorld Wide Webに代表されるインターネット環境は、多くの人にとって重要なコミュニケーション・メディア/コミュニケーション・ツールとなっている。そのようなネットワーク環境を利用し、関連した研究を行う者同士が、論文の公刊前に、研究の進め方や研究・実験の成果について互いに意見交換が可能な環境が提供されれば、研究を進める上で大きな利点があると考えられる。

そして記録、コミュニケーションと並んで、コンピュータが果たすことができる重要な役割として、ストレージ機能がある。研究資料や実験試料、あるいはプログラムなどを手軽に公開・参照できる機能が実現できれば、研究者間での研究資料や実験試料などの流通も盛んになり、研究自体が活性化するであろう。

ただし、記録機能においても、ストレージ機能においても、研究の遂行途中においては一般には秘匿したい、あるいは秘匿する必要のある情報も存在する。そのため、情報を公開する範囲を柔軟に制御できなければならない。

現在、World Wide Webを用いたSNSなどのサービスにおいて、記事の公開範囲はある程度の制御が可能ではあるが、[3]のサービスも含め、概ねその設定の自由度は低い。

Masuiらや江渡らは、QuickML[5]やqwikWeb[6]により、情報にアクセスできる者を柔軟に変更可能とするシステムの構築と運用を試みた。また、SNSを基盤としたシステムとして、永田らはEnzinを[7]、高井らはACSを開発し[8]、運用実験を行った。高田らは，公開Web-DBとWeb-DB管理システムを分離可能であり、かつきめ細かなアクセス制御が可能なWeb-DB管理システムを構築している[9]。また、一般のSNS/blogにも、記事ごとに公開範囲の設定を可能とするサービスが登場しており[10]、公開範囲の柔軟な制御の需要が伺える。

本研究では上述のような研究記録などの管理を主目的として、公開範囲を柔軟に変更可能かつ、記録の真正性および非改竄性を可能な限り保証する、研究記録の管理・公開・検証を行うシステムであるar χ ves*1[11、12]の改良を試みた。

*1 ar χ ves:Archive system for Research logs by Kato=Shimanuki/Kobayashi Satoshi, and VErification System
→ARKSVES.KSをxに、さらにxを類字形の χ に置き換え

2 システムの特徴

2.1 概要

本システムは，研究記録などの管理を主目的とし，公開範囲を柔軟に変更可能かつ、記録の真正性および非改竄性を可能な限り保証するシステムの構築を目的としている。

そこで、記事およびデータの真正性と非改竄性の保証をある程度可能とするために、記事およびデータの投稿時に、記事データに対して二重署名の処理を行う実装を行い、また平易な操作で検証を行えるようにした。また、電子情報の簡便な記録・保存と扱いやすさのために、基本的な操作は近年広く普及しているSNS/Blogをベースとした実装とした[13]。そして柔軟な公開範囲制御の実装のため、一般のSNS/BlogのようにBlog コンテンツ全体の公開範囲を指定するのではなく、投稿される記事ごとに公開範囲を指定し、公開範囲の設定段階も自由度の高い指定を可能とした。

本システムの特徴を要約すると、以下の3点となる。

● 真正性と非改竄性の一定の保証を可能とする、電子データに対する署名および検証機能

● SNS/Blogをベースとした実装

● 記事ごとに公開範囲の指定を可能

以降の各節で機能の詳細について説明していく。

2.2 利用ツール

本システムで使用したツール等を表1に示す。

Ruby on Railsは本システムの基幹となるSNS/Blog機能の構築を中心に活用し、データベースにはMySQLを利用した。電子署名の付与および検証については、全てGnuPGを活用している。

表1　利用ツール

開発言語:	Ruby ver. 1.8.6
フレームワーク:	Ruby on Rails ver. 1.2.6
公開鍵暗号ソフト:	GnuPG ver. 1.4.9
データベース:	MySQL ver. 5.0.27
ウェブサーバ:	Apache ver. 2.0.6
SSL ライブラリ:	OpenSSL ver. 0.9.8

2.3　記事の真正性と非改竄性の保証－データへの二重署名

2.3.1　検印モデル

企業における、記録の真正性および非改竄性を保証する方法として、部下が書いた記録(日報など)に、上司が検印を捺すことが広く行われている。このような、記述者とは異なる者が検印ある

いはそれに類する行為/操作を行うモデルを、本論文では「検印モデル」と呼ぶ。検印モデルの例としては、原田らは公開鍵暗号を用い、記述者が自分自身の秘密鍵で電子署名をすると共に、文書管理システムに持たせた秘密鍵を使った電子署名により検印をするモデルを提案している[1]。本論文においても、この検印モデルを採用した。ただし、原田らの提案においては、文書管理システムが検印を行うが、本論文で述べるシステムにおいては、記事やデータに検印を行うサーバ(以下「検印サーバ」)と、文書を保管するサーバ(以下「文書サーバ」)」は異なることを想定しいる。文書サーバは、ユーザが通常、記事を記述する際に用いるサーバとした。

これはタイムスタンプ技術[13]と類似しているが、文書作成者の真正性を、ユーザ自身の電子署名も自動的に付加することにより明示する点が異なる。

検印については、大学程度の規模の組織を越えて、相互に検印を行うモデルを想定している。このようにシステムを分散することで、導入および管理コストの低減を期待している。

このようなシステムの利用サイトが相互に検印を行う手法によって、真正性や非改竄性の保証の強度は、企業が提供するサービスに比べて幾分低くなると思われる。しかし、企業が提供するタイムスタンプ・サービスは高価であったり、利便性に欠ける部分がある。そこで、本論文で示すシステムと、企業が提供するサービスとは使い分け/棲み分けができると考えられる。

また、ラボノートの運用および証拠能力から、本システムも十分な証拠能力を持っていると考える。

2.3.2 検印モデルによる実装

記事を記入後、投稿ボタンをクリックすることで以下のような手順で二重署名が行われる。

① 投稿された記事に対して、文書サーバがタイムスタンプを押すとともに、ユーザの秘密鍵で署名を行う。

② ①で作成されたデータを検印サーバへ転送する。

③ 文書サーバから送られたデータに対し、検印サーバがタイムスタンプを記録し、更にデータ全体に対し署名を行う。そして署名部(図1、2参照)のみを検印サーバに保存する。

④ 検印サーバは文書サーバに2重に署名済みのデータを送り返す。

⑤ 文書サーバにデータを保存する。

以上の一連の流れをを図式化したものが図1である。図2は、実際に二重署名が行われた文書の構成を示している。

文書サーバと検印サーバ間のデータの通信には、データ保護のためSSLを用いている。

また文書サーバと検印サーバのそれぞれで行われる署名の直前に、データに対して時刻情報を書き込み、タイムスタンプのシンプル・プロトコルと同様な作業を行っている。

一般的に普及しているタイムスタンプではデータ漏洩への対策も考慮し、ハッシュ値を用いてデータの非改竄性の保証を行っているが、本研究では閲覧者がスクリプトによらない検証を行えるようにする為、Clear*2署名で2重署名作業を行っている。

*2 Clear署名：Gnu Pgで復号しなくても、文書の中身を確認できる形で残して署名を付与する方式。署名を行う対象そのものは暗号化させず、署名を行う。

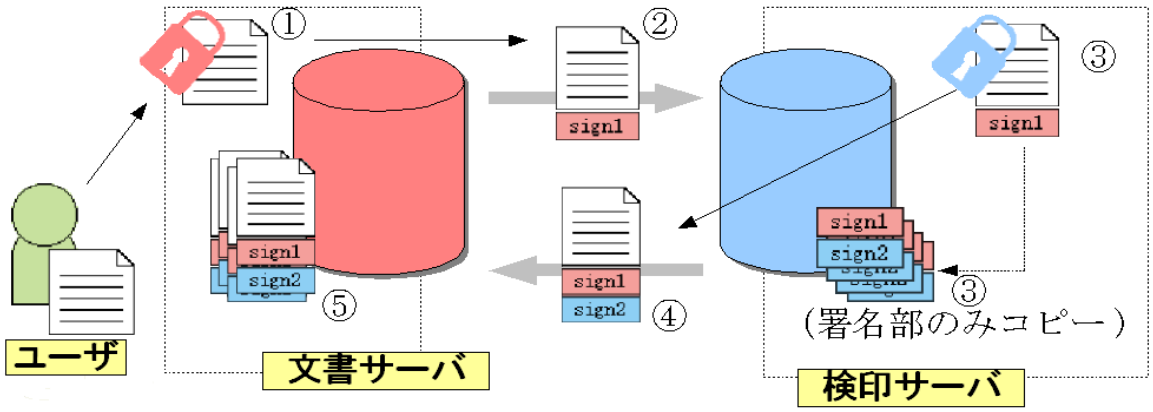

図1　二重署名の処理の流れ

図2　二重署名された記事データの構成

2.4 記事および添付ファイルに対する検証機能

前節で二重のデジタル署名によって、投稿されたデータの真正性および非改竄性の保証が行われているが、それが確かであるかどうかは署名の検証作業によってのみ確認が可能である。

また、電子商取引推進協議会、認証・公証WG [14]のガイドラインでは、デジタル署名の有効性を長期的に維持する為の用件として、次の4つを挙げている。

1. 署名検証時に，署名再検証に必要な情報を明確にしておくこと

2. 署名検証時の時刻を明確にしておくこと

3. 署名再検証時に必要な情報を改竄検出可能な状態にすること.

4. 署名再検証時に必要な情報を保存すること

（1）は、検証を行う際に確かにその署名が有効であることを示す情報（あるいは失効情報）を明示しておくことである。そして、署名の検証が確かに保証されていることを確認した日時を明確にし、(2)その時点まで確かにデータの保証されていることを記録し、(3）再検証を正しく行えるように改竄の検出を可能にして、(4)署名の再検証に必要な情報を保存して、第三者でも再検証を行えるようにしておくということである。これらの用件を満たすことがデジタル署名の有効性の長期的な維持には必要であるとされている。

このように、本当にその署名が確かであることを確認したり、あるいは長期的な有効性の維持には「署名の検証」が必要となってくる。本システムでは、その検証を簡単に行える機能を実装している。

各記事およびデータには、「簡易検証」機能および「通常検証」機能が用意され、閲覧者はそれらのリンクをクリックすることで簡便な検証を行なえる。なお、「簡易検証」は文書サーバのみで、ユーザの公開鍵と検印サーバの公開鍵を用いて行う検証である。通常検証は、検印サーバに記録されている署名データとの照合も含めて検証を行う。検証を行なった画面例を図3に示す。また、記事本文に対してのみ改竄が行われた場面の画面例を図4に示す。このとき、検印サーバにも保存されている署名部に対しての改竄は行われていない。

検証結果

投稿者:管理者

検証記事タイトル:公開鍵暗号を用いた研究記録管理

投稿日時:2009-02-18 16:51:01

ユーザ主鍵フィンガープリント:
F4BB BDCE 6667 7ACB 60AE AC0B 15FB B2C1 8AAE EFB8

検証結果:サーバー署名

gpg: 02/18/09 16:51:03にDSA鍵ID 8696ACBDで施された署名
gpg: "server (sarver's key)"からの正しい署名

検証結果:ユーザ署名

gpg: 02/18/09 16:51:01にDSA鍵ID 8AAEEFB8で施された署名
gpg: "MikuKatou "からの正しい署名
gpg: 警告: この鍵は信用できる署名で証明されていません!
gpg: この署名が所有者のものかどうかの検証手段がありません。
主鍵の指紋: F4BB BDCE 6667 7ACB 60AE AC0B 15FB B2C1 8AAE EFB8

検印サーバ側の署名部との照合結果

検印サーバ側の署名部と一致しています。
・ユーザ署名とサーバ署名が共に一致しています。

図3　検証結果(成功時)

検証結果

投稿者:管理者

検証記事タイトル:公開鍵暗号を用いた研究記録管理

投稿日時:2009-02-18 16:51:01

ユーザ主鍵フィンガープリント:
F4BB BDCE 6667 7ACB 60AE AC0B 15FB B2C1 8AAE EFB8

検証結果:サーバー署名

gpg: 02/18/09 16:51:03にDSA鍵ID 8696ACBDで施された署名
gpg: "server (sarver's key)"からの 不正な 署名

検証結果:ユーザ署名

gpg: 02/18/09 16:51:01にDSA鍵ID 8AAEEFB8で施された署名
gpg: "MikuKatou "からの 不正な 署名

検印サーバ側の署名部との照合結果

検印サーバ側の署名部と一致しています。
・ユーザ署名とサーバ署名が共に一致しています。

図4　検証結果(記事本文改竄時)

しかし、本システムはコンピュータ・プログラムとして実現されている以上、スクリプトを書き換えることにより、あたかも検証を行なったように見せかけることも可能である。そのため、閲覧者が独自でも検証を行なえるよう、ユーザおよび検印サーバの公開鍵は画面上から取得可能としている(図5)。検証手順については、ブログリストのサイドバーから、手引きを参照できるようにしている。

図5　公開鍵のダウンロード画面

2.5 SNS/Blogベース

2.5.1 概要

本システムは，研究室～学科程度の規模を単位としての記録の保存を想定しており、過度に分散せず、過度に集約せず、適度に分散しつつネットワークを構成するシステムを想定している。それに加えて電子情報の簡便な記録・保存と扱いやすさのために、基本的な操作は近年広く普及しているSNS/Blogをベースに実装している。

本システムにアクセスすると図6のようなトップページコンテンツが表示される。ここでユーザとして認証を受け、ログインすると、図7のような画面になる。

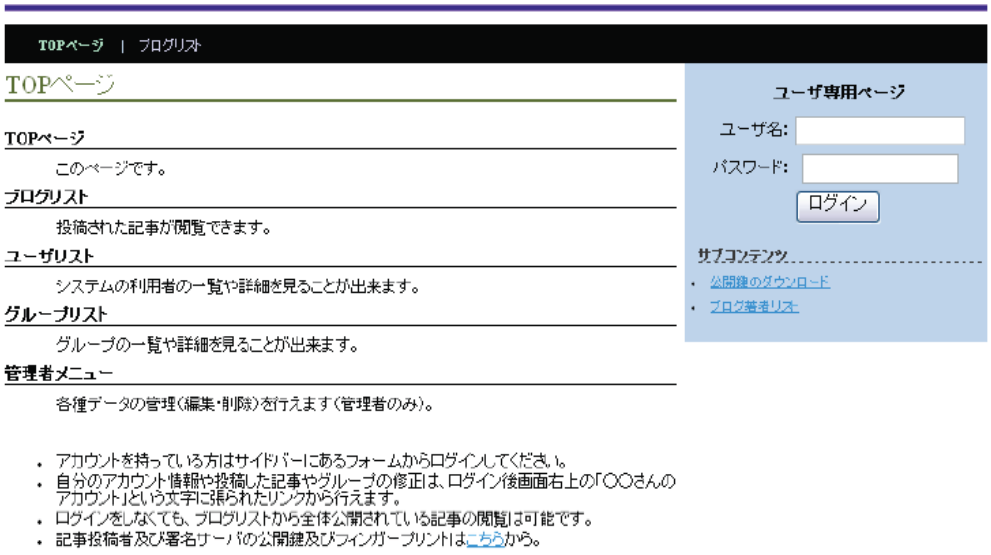

図6 トップページ画面

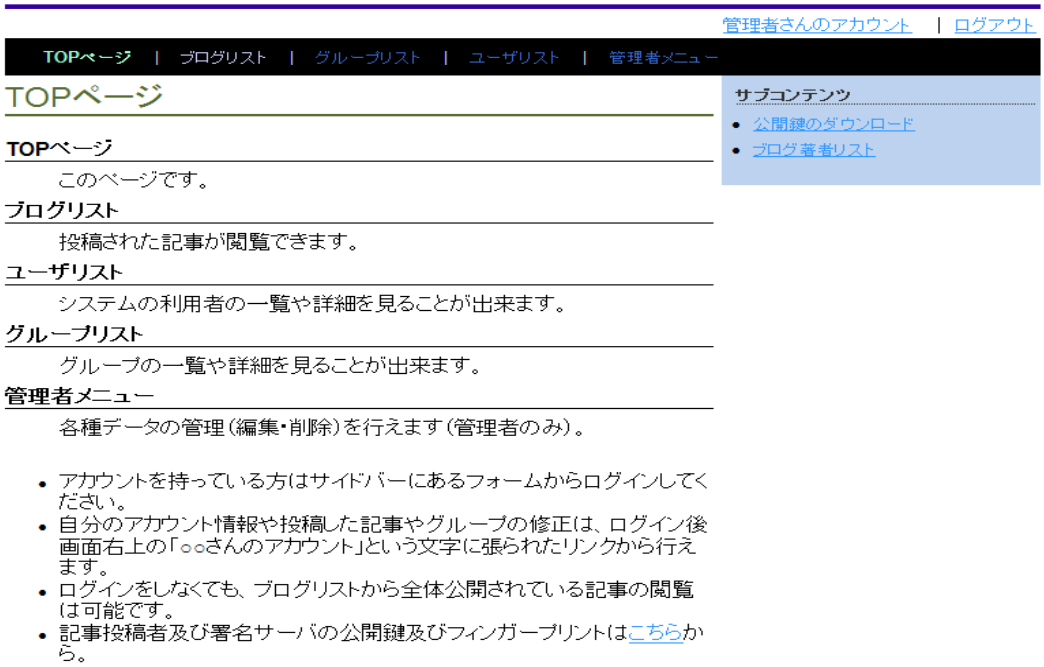

図7 トップページ画面(ログイン後)

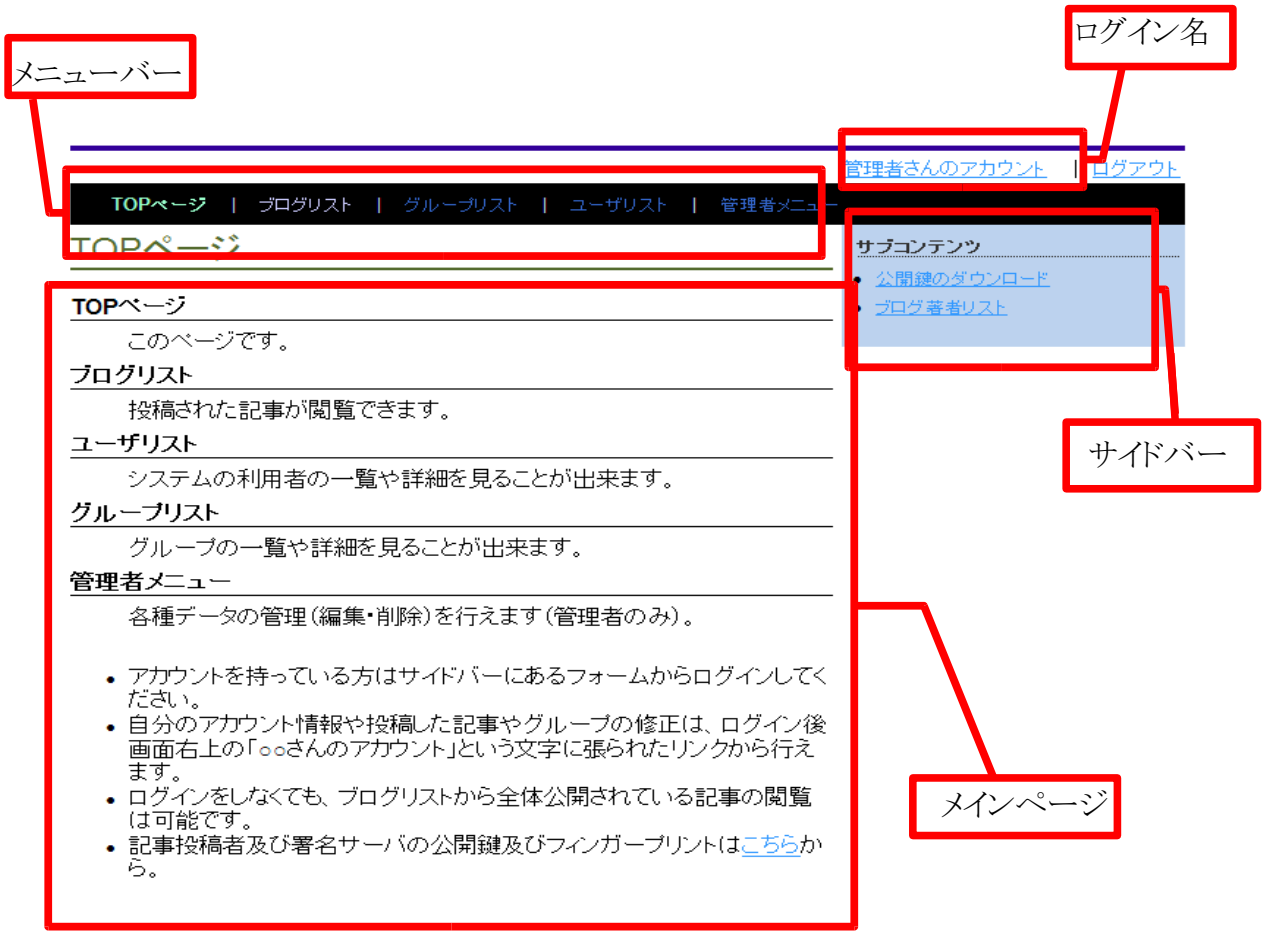

図8　トップページ画面(詳細)

図8に示すようにページは大きく上からメニューバー、メインページ(左)、サイドバー(右)で構成されている。

メニューバーには各コンテンツへのリンクが張られており、メインページには各コンテンツの内容が、サイドバーにはコンテンツ内のサブコンテンツ・関連ページへのリンクやコンテンツ内の検索機能が表示される(図9)。

図9　グループリストにおける検索機能(枠線内)

またログインすることでメニューバーの右上部にアカウント名が表示され、アカウントメニューへのリンクとログアウト用のリンクが表示される。メニューバーおよびサイドバーは、ログインした利用者の権限ごとに表示されるコンテンツが異なる（詳細は「2.5.2利用者の分類」を参照）。

以下でコミュニティを形成する利用者およびグループや、利用者による記事の投稿をメインとした交流とその管理を行う為の機能について示す。

2.5.2 利用者の分類

本システムの利用者は、本システム上に何らかのアカウントを持っている。

このシステムの利用者は、システムを利用できる権限によって「ゲスト」、「ユーザ」、「管理者」に分類される。

利用者ごとの大まかなアクセス権限を示したものが表2である。

「ゲスト」は、システム内のblog記事の投稿やグループの作成は行えないが、グループに所属し、閲覧許可のある記事の閲覧は可能な利用者である。

「ユーザ」は、システム内に自身のblogを作成し、記事を投稿することができる利用者で、グループの新規作成や、自身のアカウント情報を修正することができる。また、自身が投稿した記事の各々について公開範囲の変更や、カテゴリの変更が行える。

「管理者」は、システムの管理者であり、管理者メニューにアクセスできる唯一の利用者である。

本システムはTOPページからID とパスワードを利用したユーザ認証を受けてログインすることで、利用者の権限に応じてコンテンツが表示される。また、システムにログインしていなくても、公開範囲に制限を加えられていない記事に関しては閲覧が可能である。

表2にある「アカウントメニュー」はログインした利用者自身の情報の編集、削除が行えるコンテンツであり、また「管理者メニュー」はシステム全体の各データの編集、削除が行えるコンテンツである。

表2　アクセス権限

利用者 / コンテンツ	管理者	ユーザ	ゲスト	一般
TOP ページ	○	○	○	○
ブログリスト	○	○	○	○
グループリスト	○	○	○	
ユーザリスト	○	○	○	
アカウントメニュー	○	○	△	
管理者メニュー	○	○		

△：アカウント情報の閲覧のみ可能

2.5.3 グループについて

グループは、一人以上の利用者からなる利用者の集合である。その種類は、階層別に「ルートグループ」、「トップレベルグループ」、「子グループ」に分けられる（図10）。

ルートグループは、アカウントを持つ利用者全てが所属するグループである。そのため、利用者は必ず一つ以上のグループに所属していることになる。

トップレベルグループは、ルートグループの直下に作られるグループであり、利用者が自分で作成できる一番上位のグループである。

子グループは、トップレベルグループよりも下層に作られるグループを指す。必ず親としてトップレベルグループか、子グループが一つ必要となる。

以上より、グループには階層構造を持たせることが可能である。グループの種類と命名規則についての詳細は表3に示す。

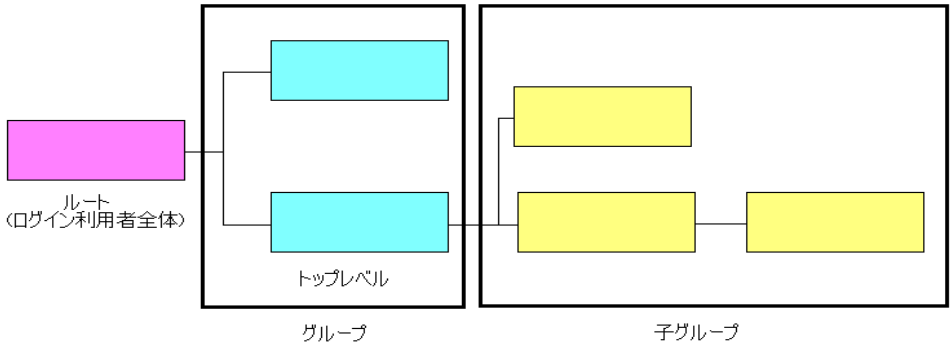

図10　グループの階層構造

表3　グループの種類と命名規則

グループの種類	階層	名前規則	新規作成
ルートグループ	0	: :	不可
トップレベルグループ	1	: :グループ名	可
子グループ	2～n	: :トップレベルグループ名 : 子グループ名 : ・・・・・・・ : グループ名	可

2.5.4 利用者の管理（ユーザリスト）

システム利用者の情報の閲覧は、ログインしている利用者であれば，コンテンツの「ユーザリスト」から誰でも見ることができる。TOPページの名前に張られたリンクから、各自の詳細な情報を閲覧できる。

ユーザの新規登録および修正や削除は管理者メニューの「ユーザ管理」から管理者のみが行えるようになっている。利用者の、「管理者」、「ユーザ」、「ゲスト」という権限の指定は、管理者のみが設定できる。

ただし、利用者自身の情報（アカウント情報）は、ユーザであれば一部の情報（ハンドル名、パスワード、鍵情報、所属グループ、備考）の変更が可能となっている。アカウントメニューの「ユーザー情報の変更」から行える。

2.5.5 グループの管理（グループリスト）

グループリストの閲覧はログインしている利用者であれば、コンテンツの「ログインリスト」から誰でも閲覧できる。

グループの新規作成は、ユーザと管理者であれば誰でも作成可能となっている。新規作成できるグループは、トップレベルグループと子グループの二つがあり、どちらを作成するか選択できる。子グループは親となるグループのIDを指定することで作成することができる。また、子グループを親とするグループの作成をすることも可能である。

また、グループの管理は作成したユーザと管理者が行うことができる。この二者を「グループ管理者」と呼ぶ。グループ管理者は、グループの編集、削除、参加者の変更を行える。そして管理者は、上記に加えてグループ作成者の変更が可能である。

各グループの詳細な情報は、TOPページのグループ名のリンクから閲覧できる。

2.5.6 記事の投稿・管理（ブログリスト）

記事の閲覧は、コンテンツの「ブログリスト」からできる。コンテンツへのアクセスは誰でも出来るが、実際の閲覧については記事ごとに指定された閲覧範囲によって制限されている。つまり、公開範囲によってはシステムへのログインをしていなくても記事を閲覧することが可能となっている（公開範囲の詳細は次節「2.6記事毎の公開範囲の指定」を参照）。閲覧者が公開範囲に含まれていない記事は、TOPページでは記事内容の代わりに「閲覧権限がありません」と表示されるが、タイトルのみの閲覧や投稿日時や記事の検証は行うことができる。

記事の新規作成は、ログインしているユーザと管理者であれば「ブログリスト」コンテンツのサイドバーより行える。投稿画面は図11のようなものとなる。

記事投稿の際の二重署名の処理についての詳細は「2.3.2検印モデルによる実装」で述べた通りである。その二重署名も含めた全文を表示したページが図12である。

また、各記事にはファイルを添付することができる。このファイルについても記事と同様に二重署名が行われる。添付ファイルのダウンロードは、二重に署名されたままのファイルと、システムが自動的に復号処理を行ったファイルの両方を、記事の詳細ページよりダウンロードが可能となっている。

なお、現在、投稿後の記事の修正・再編集は認めていない。記事の削除については、管理者のみが行える。また、添付データの削除は投稿者も自身の記事のアカウントメニューの「ブログ記事の閲覧範囲の変更／添付ファイルの削除 」から行えるが、削除理由を明記した上で削除可能としている。また添付データが存在していた記録が残るようにしている。これは、研究記録の非改竄性を考慮してのことである。このように編集や削除に制限をかけているが、記事などの管理にはDBMSを利用しているので、DBMSを直接操作することにより、記事の修正や削除は可能である。

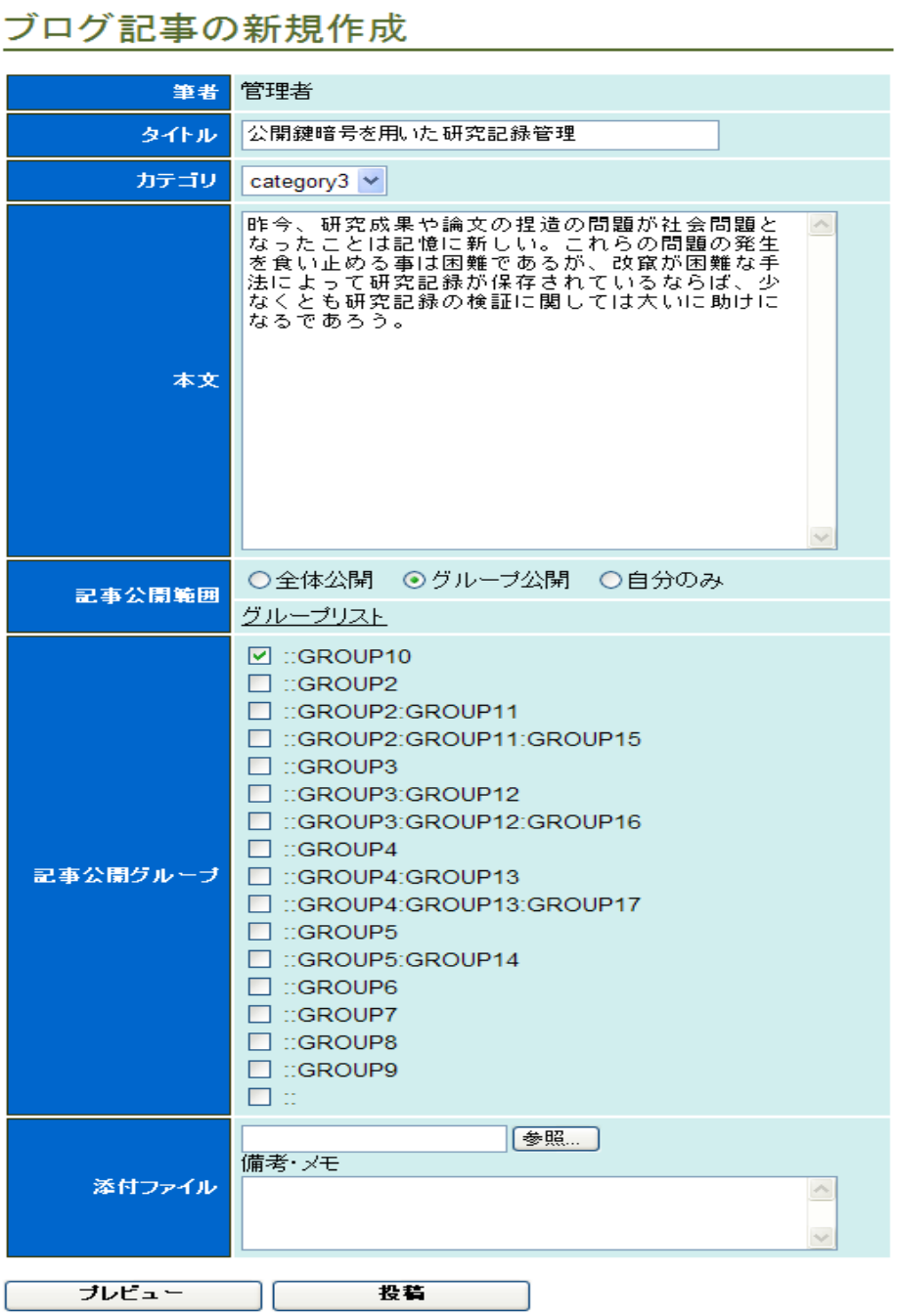
ブログ記事の新規作成
筆者
管理者
タイトル
公開鍵暗号を用いた研究記録管理
カテゴリ
category3
本文
昨今、研究成果や論文の捏造の問題が社会問題となったことは記憶に新しい。これらの問題の発生を食い止める事は困難であるが、改竄が困難な手法によって研究記録が保存されているならば、少なくとも研究記録の検証に関しては大いに助けになるであろう。
記事公開範囲
全体公開
グループ公開
自分のみ
グループリスト
記事公開グループ
::GROUP10
::GROUP2
::GROUP2:GROUP11
::GROUP2:GROUP11:GROUP15
::GROUP3
::GROUP3:GROUP12
::GROUP3:GROUP12:GROUP16
::GROUP4
::GROUP4:GROUP13
::GROUP4:GROUP13:GROUP17
::GROUP5
::GROUP5:GROUP14
::GROUP6
::GROUP7
::GROUP8
::GROUP9
::
添付ファイル
参照...
備考・メモ
プレビュー
投稿

図11　記事投稿画面

公開鍵暗号を用いた研究記録管理

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

- -----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

昨今、研究成果や論文の捏造の問題が社会問題となったことは記憶に新しい。これらの問題の発生を食い止める事は困難であるが、改竄が困難な手法によって研究記録が保存されているならば、少なくとも研究記録の検証に関しては大いに助けになるであろう。

ClientTimeStamp: 2009-02-19 19:34:49
- -----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.9 (MingW32)

iEYEARECAAYFAkmdNckACgkQFfuywYqu77ijjQCgyaXqo1HlGy+wvXfHgkEEjgWV

hjcAoJk9VkmVgiJLXVmyJ0qMFWuhJXcI
=7/NL
- -----END PGP SIGNATURE-----

Server TimeStamp. 2009-02-19 19:34:52
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.9 (MingW32)

iEYEARECAAYFAkmdNcwACgkQNkcjG4aWrL140QCfYwbsC/KaG8rPihlZSPSb2zHO

1cMAn28F6E1BR7toH5HGggNt8PMxpqTA
=bnHm
-----END PGP SIGNATURE-----

公開範囲

::GROUP10

2009-02-19 19:34:49 | 管理者 | 簡易検証 | 通常検証 | コメント(0) | category3

コメントを書く

図12　図11の記事投稿における二重署名も含めた記事全文

2.5.7 アカウントメニュー

「アカウントメニュー」は「ユーザ」および「ゲスト」がアクセスできるコンテンツで、ログインしている利用者自身の情報や投稿データの編集、削除が行えるコンテンツである。ただし、「ゲスト」は自身のアカウントデータが閲覧できるだけで、編集や削除作業は一切出来ない。

メニュー項目については表4を参照。

表4 メニュー項目

アカウントメニュー	管理者メニュー
アカウント情報の閲覧 アカウント情報の変更 管理グループの変更 ブログタイトルの変更 カテゴリの作成・変更 ブログ記事の閲覧範囲の変更 添付ファイルの削除	ユーザ管理 グループ管理 ブログ記事管理 コメント管理 カテゴリ、ブログ名管理

2.5.8 管理者メニュー

「管理者メニュー」は「管理者」のみがアクセスできるコンテンツで、システム全体の各データの編集、削除が行えるコンテンツである。
ただし、ブログ記事の本文データの編集は、管理者であっても出来ない。

2.6 記事毎の公開範囲の指定

前述したように本システムでは記事単位で公開範囲を指定することが出来る。
公開範囲は大きく「全体公開」,「指定グループ」,「自身のみ」の3種類に分けて指定することができる。
「全体公開」は、システムの認証(ログイン)を受けなくても閲覧可能な公開範囲である。
「自身のみ」は, 記事を書いたユーザ自身のみが閲覧可能な公開範囲である。
「指定グループ」による公開範囲は、グループを単位として公開範囲を指定する。この時複数のグループを指定でき、指定されたグループを上位に持つ子グループも自動的にその公開範囲に含められるような、グループの階層構造を生かした公開範囲制御を行っている。
なお、記事の投稿後にも公開範囲の変更は可能である。閲覧範囲は、記事を書いたユーザのみが変更可能である。

3 改良点

3.1研究記録に関して

現在、Webブラウザで数式を表示する手段としてMathML(Mathematical Markup Language)というマークアップ言語がある。しかしこれを人間が記述するのは手間である。そこで今回、人間にとっても記述しやすい言語であるJipsenによるASCIIMathML[15]を用いて記事中への数式の投稿を可能とした。
例としてブラウザに数式 $a+b-3$ を表示させるとき、MathMLは図13のような記述になるが、ASCIIMathMLであれば『amath a+b-3 endmath』と記述すればよい。ASCIIMathMLの方が人間にとって記述しやすいだけでなく、直感的に分かりやすいことが分かる。

```
1 <math xmlns="http://www.w3.org/1998/Math/MathML">↓
2   <mi>a</mi>↓
3   <mo>+</mo>↓
4   <mi>b</mi>↓
5   <mo>-</mo>↓
6   <mn>3</mn>↓
7 </math>[EOF]
```

図13　MathMLによる記述例

図14は数式を含む記事の投稿例、図15は図14の記事の閲覧画面である。例えば投稿したい数式が $a^b+c=d$ であるならば、『amath a^b+c=d endmath』というように数式の前にamath、数式の終わりにendmathと入力することで数式が投稿できる。このとき、数式、amath、endmathはいずれも半角英数で入力する(数式の書き方はASCIIMathML[15]を参照)。

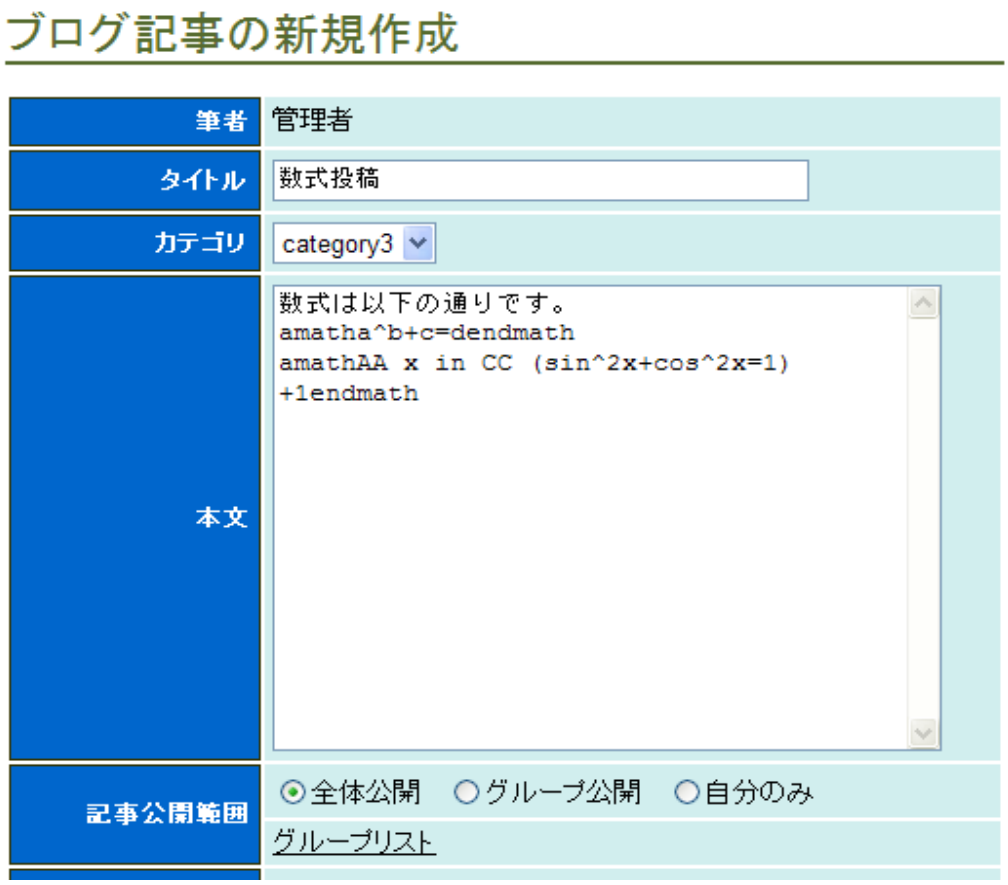

図14　数式を含む記事の投稿

図15　図14を投稿した際の記事本文

3.2ユーザインタフェースに関して

今まで記事投稿の際、グループの指定をテキストエリアで入力していたのは煩雑な作業であった。今回、グループの指定をDBからグループを引き出し、チェックボックスで選択可能にすることを可能にした(図11参照)。

3.3セキュリティに関する改良

今回、文書サーバと検印サーバ間でのIDとパスワードでの認証の実現を行った。これは他のサーバからのなりすましを回避するものである。またこの変更によって、これまで、検印サーバと文書サーバは一対一の対応をとっていたが、一つの検印サーバで複数の文書サーバにも対応可能となった。図15は設定ファイルがあるとしたときの設定ファイル(但しプログラム中には存在する)であり、それぞれの文書サーバが持つID、パスワード、署名部が保存されるディレクトリを定義するものである。

図16は検印サーバに保存される文書サーバの情報を含むファイルの一覧である。idは記事を送ってきた文書サーバのid、kiji_idは記事のID、pwは文書サーバのパスワード、dirは記事の署名部を保存しているディレクトリ名を示し、どの記事がどの文書サーバから送られてきたかを判別可能となっている。図16ではid:XXXXを持つ文書サーバから送られてきた記事がsite_aというディレクトリに署名部が保存されていることを示す。パスワードは文書サーバから検印サーバに送る際に、ランダムに生成された文字列を用いて暗号化して送信している為、一定の値ではない。

図15　設定ファイルの例

```
id:XXXX    pw:YYYY    dir:site_a
id:AAAA    pw:BBBB    dir:site_b
id:CCCC    pw:DDDD    dir:site_c
   ・          ・          ・
   ・          ・          ・
   ・          ・          ・
   ・          ・          ・
```

図16　検印サーバに保存される文書サーバの情報を示したファイルの例

```
id:XXXX    kiji_id:112    pw:68RzkLUTLBg5M     dir:site_a
id:XXXX    kiji_id:113    pw:86jMgBgYuacss     dir:site_a
id:XXXX    kiji_id:114    pw:46NpIUXiQcRUw     dir:site_a
id:XXXX    kiji_id:115    pw:771nch4GhykGs     dir:site_a
id:XXXX    kiji_id:116    pw:79yaVcj0tzCzA     dir:site_a
id:XXXX    kiji_id:117    pw:39/aAQPhiDWQM     dir:site_a
id:XXXX    kiji_id:118    pw:993io5QjHViQQ     dir:site_a
id:XXXX    kiji_id:119    pw:41aFXm0pfiY3.     dir:site_a
id:XXXX    kiji_id:120    pw:26hSuiV8m4rYY     dir:site_a
   ・          ・              ・                  ・
   ・          ・              ・                  ・
   ・          ・              ・                  ・
```

4 今後の課題

4.1 セキュリティに関して

本システムの大きな課題として、まずセキュリティの問題がある。

真正性の保証については、ユーザの秘密鍵をICカードやUSBメモリなどの外部記憶メディアに保管し、記事を投稿するときにのみにそれを参照することで解決できよう。また、本システムにおいて、成りすましが行われる可能性もある。しかし、ユーザの秘密鍵をICカードやUSBメモリなどの外部記憶メディアに保管することにより、成りすましでの記事を投稿することはできなくなる。

非改竄性の証明の強度を高めるためには、リンキングやヒステリシス署名、履歴交差などの技術の導入の検討も必要であろう[4,16]。関連して、現在GnuPGに依存したシステムである為、多様な署名方式に対応可能とすることによっても利便性と共に非改竄性の向上が期待できる[17]。

また電子署名の長期利用に関しては、電子署名の危殆化などの問題もあり、評価・検討が必要である[18,19]。同様に一定期間ごとの検証記録を保存することによって、どの時点まで真正性が保持されていたかを証拠として残していくことも今後必要となってくるだろう[14]。

現状では、検印サーバでの署名のために、記事やデータそのものを通信している。そのため情報漏えいの危険性がある。現在は、SSLでの通信を行うことで対処しているが、情報漏えいの対策として十分とは言いがたい。記事やデータのハッシュ値のみの通信に限ることも含めて今後検討を要する。

また、公開鍵の正当性の確保のため、信用の輪（Web of Trust）も含めたPKIの利用の検討も必要であろう。

4.2 ユーザインタフェースに関して

柔軟な公開範囲の設定については、概ね既存のもの[7,8]に近い自由度となっている。しかし、ログインしている利用者に対しての公開範囲の指定は現状十分に出来るが、ログインしていない一般に対しての公開範囲は全体公開しか存在しない。記事の公開をユーザだけではなく、ある特定の人物にも行いたい場合において、今後その人がその他の記事の閲覧・コメントをする機会がほぼないときのように、その人に対して特別にグループを用意するまでもないケースも出てくるだろう。したがって今後、システム利用者と一般を区別せず、任意の記事に対して特定のパスワードを入力することで閲覧を可能とする公開範囲というものも考えられる。

また本システムではグループの階層化を取り入れたが、既存のグループの上位に新たに階層を作成することはできない。木構造の表現を可能とするacts_as_tree という特殊なカラムの命名規約（MagicFieldNames）がRuby on Railsに存在するので、それを利用したグループの作成も考えられる。また、異なる上位グループを持つグループ同士であっても、そのグループの構成員の間には何らかの関連がある場合も現実にはある。そのような縦方向以外での関連を持たせる実装も含め、今後さらに自由度の高いグループ管理を可能としていくことも課題となる。

交流の支援という観点からは、本システム、または本システムと同種のシステムが複数稼動した場合、ゲストが各ユーザのBlogを閲覧するたびに、個々にログインを要求されるのは煩雑である。そこで、本システムあるいは同種のシステム間において、認証のDelegate機能も必要であろう。またRSSの配信やトラックバックなど一般のBlogに存在する機能についても、閲覧範囲が記事単位で可変であるため難しいが、今後必要に応じて検討を要する。同じく一般のBlogシステム・サービスによくある、他システムからのデータのインポート、また逆に他システムへのエクスポート機能の実装も考えられる。

研究支援という観点からは、記事中での画像や表・グラフ、化学式への対応も不可欠である。このような問題については、JipsenのASCIIsvg[20]が利用/応用可能であろう。また漢字における異体字や国字に対しての対応も検討していきたい。

また現システムでは、投稿された記事が投稿後、改竄されていないことを証明することは可能であるが、投稿された記事自体が捏造されたものかどうかまでは確認することができない。したがって、記事を投稿し公開する前に、内容を確認する者への対応及びそのインターフェースを用意する必要があると考えられる。

その他、現在は記事投稿の際にグループを全て表示し指定することなどに対しての、Ajax技術などを用いたユーザインタフェースへの改善も今後の課題である。

4.3 その他

記録の保存ということを考えると、現在のデータベースをベースとした記事管理はサーバに負担をかけてしまうと考えられる。そのため、DBMSは必要だが、それに全て埋め込むのではない、記事やユーザ情報の保存・管理方法が課題として上げられる。

また、現在基盤となるシステムをRuby on Rails1.2.6で作成しているが、より柔軟な機能作成や今後蓄積されるデータの柔軟な活用を行っていくことを考慮すると、HTMLだけではなくXMLやJSONなどの複数のフォーマットでの出力も可能なRuby on Rails2.0への移行が望ましいと考えられる。

そして、ユーザインタフェースとも関連するが、現在グループはシステムの利用者全体で共有し、ログインしている利用者であれば誰でも閲覧が可能で、ユーザであれば任意のグループに所属できるようになっている。しかし、現実にはグループの存在やそこに所属するメンバーを秘匿しておきたい場合も考えられる。それに関連して、ユーザの情報も現時点ではログインしている利用者であれば、誰でも閲覧が可能となっている。昨今個人情報の保護が問題とされる機会も多く、一般のSNSでも特定の知人のみにしか自分のユーザ情報を開示しない機能を持つものは多い。それらの問題も含め、今後情報の開示および管理の権限を誰にどの程度まで認めるのか検討を要する。

５ 終わりに

本研究では、研究データの捏造に対する対策として記事の検証を行え、また研究内容の進捗状況や内容によって公開できる範囲を柔軟に指定できる、研究記録管理・公開・検証システム arχvesについて主にユーザインタフェースに関しての改良を試みた。

今後は、「4. 今後の課題」でも取り上げた本システムの課題への対応や改良、そして蓄積される記事の知的処理に関しても研究を進めたい。

参考文献

[1]原田　篤史, 西垣　正勝, 曽我　正和, 田窪　昭夫,「ライトワンス文書管理システム」, 情報処理学会論文誌, Vol.44, No.8, pp.2093-2105, 2003.

[2]陳　明強, 吉川　正俊,「時刻認証付きXML文書のデータベースによる管理について」, 電子情報通信学会第16回データ工学ワークシップ(DEWS2005), 5A-i10, 2005.

[3](株)イースト,"SynetLaboNote", http://www.labonote.jp

[4]宇根　正志, 松本　勉,「可用性および安全性の観点からみた各タイムスタンプ方式間の関係」, 情報処理学会論文誌, Vol.43, No.8, pp.2644-2658, 2002.

[5]Toshiyuki Masui,　Satou Takabayashi, "Instant Group Communication with QuickML", Proc.ACM Conference on Supporting Group Work(Sroup '03), pp268-273, 2003.

[6]江渡　浩一郎, 高林　哲, 増井　俊之,「quickWeb:メーリングリストとWikiを統合したコミュニケーションツール」, 情報処理学会研究報告, 2004-HI-111, pp.5-11, 2004.

[7]永田　周一, 安村　通晃,「Enzin:情報の公開範囲を手軽に変更できるコミュニケーションツール」, 情報処理学会論文誌, Vol.48, No.3, pp.1134-1143, 2007.

[8]高井　一輝, 河口　信夫,「ACS:多様な人間関係を表現可能なソーシャルネットワーキングシステム」, 情報処理学会論文誌Vol.48, No.7, pp.2328-2339, 2007.

[9]高田　良宏, 笠原　禎也, 毛利　信浩, 松平　拓也,「多様なアクセス制限に対応した自然科学データベースシステムの開発」, 学術情報処理研究, No.11, pp.50-59, 2007.

[10](株)エイミー, "Media Wagon", http://mw.aimy.jp

[11]加藤　未来,「GnuPG を用いた研究記録管理・公開・検証システム構築」, 島根大学卒業論文, 2008.

[12]加藤　未来, 小林　聡,「ar χ ves:公開鍵暗号を用いた研究記録管理・公開・検証システム構築の試み」, 学術情報処理研究, No.12, pp.43-51, 2008.

[13]黒田　努, 佐藤　和人　共著, 株式会社オイアクス　監修,「基礎Ruby on Rails」, インプレスジャパン, 2007.

[14]電子商取引推進協議会, 認証・公証WG,「電子署名文書長期保存に関するガイドライン」, 2002.

[15]Peter Jipsen, "ASCIIMathML", http://www1.chapman.edu/~jipsen/asciimath.html

[16]洲崎　誠一, 松本　勉,「電子署名アリバイ実現機構－ヒステリシス署名と履歴交差」, 情報処理学会論文誌Vol43, No.8, pp.2381-2393, 2002.

[17]山田　竜也, 宮地　充子, 双紙　正和,「オープンネットワークにおける安全な暗号方式の更新に関する考察」, 情報処理学会論文誌Vol.4, No.8, pp.2102-2109, 2000.

[18]宮崎　邦彦, 吉浦　裕, 岩村　充, 松本　勉, 佐々木　良一,「第三者機関への依存度に基づく長期利用向け電子署名技術評価手法の提案」, 情報処理学会論文誌Vol.44, No.8, pp.1955-1969, 2003.

[19]小森　旭, 花岡　悟一郎, 松浦　幹太, 須藤　修,「署名鍵漏洩問題における電子証拠生成技術について」, 電子情報通信学会「暗号とセキュリティシンポジウム」予稿集, pp.983-988, 2003.

[20]Peter Jipsen, “ASCIIsvg”, http://www1.chapman.edu/~jipsen/asciisvg.html

付録A　DBの初期化と管理者の初期設定

・DBの初期化
>rake db:initialize
このコマンドを使うことでデータベースの作成からマイグレーション、データの投入まで一度にこなすことができる(参考文献[13]を参照)。

・管理者の初期設定

name	管理者
login_name(ユーザー名)	admin
pass(パスワード)	pass

付録B　ASCIIMathMLのブラウザへの対応

Firefoxの場合
2009年3月時点での最新版Firefox 3がダウンロードされていれば、特に何もしなくてよい。

IEの場合
Internet Explorer 6 +
MathPlayer(http://www.dessci.com/en/products/mathplayer/download.htm)をダウンロード。

付録C　テーブルリストの補足

参考文献[11]のテーブルリストに書かれていないテーブルとしてgroupsテーブルとusersテーブルの結合テーブルであるgroups_usersがある。

groups_users	型
group_id	integer
use_id	integer

付録D ユーザ登録について

ユーザの新規登録の際に図16のような画面が表示される。まず、Gnu PGで鍵の種類や鍵長、鍵の有効期間、ユーザーIDなどを登録し、鍵ペアを生成する。そして取得したユーザーID(図16では鍵ID)とフィンガープリント、秘密鍵のパスフレーズを図16の画面に登録する。

・Gnu PGでの登録例

uid(ユーザー ID)	chika_s
pub(公開鍵)	1024D/B4C070D3
指紋(フィンガープリント)	F2C9 51F2 8F04 1C0B 1C38 19A7 6487 4442 B4C0 70D3

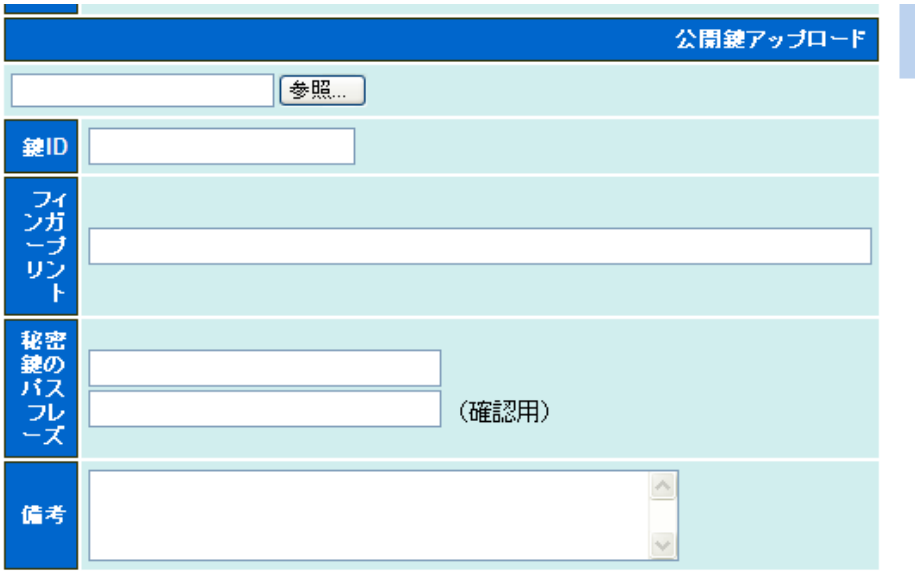

図16　ユーザ新規登録画面

参考文献
結城 浩,「新版 暗号技術入門－秘密の国のアリス」, ソフトバンククリエイティブ, 2008.

付録E　グループ指定に関する対応

グループ指定に関するフォルダ及びファイル名は以下のようになる(なお、加藤バージョン、島貫バージョンともにフォルダ名、ファイル名に変更はなし)。
フォルダ名：app¥controllers
ファイル名:blog_kijis_controller.rb