Title	大学における組織間認証連携を視野に入れた統合認証基盤構築に関する研究
Author(s)	松平, 拓也
Citation	
Issue Date	2011-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/2297/27052
Right	

大学における組織間認証連携を視野に入れた
統合認証基盤構築に関する研究

松平　拓也

平成 23 年 3 月

博士論文

大学における組織間認証連携を視野に入れた統合認証基盤構築に関する研究

金沢大学大学院自然科学研究科
電子情報科学専攻
情報システム講座
学　籍　番　号　　0723112107
氏　　名　　松平　拓也
主任指導教員名　　笠原　禎也

目次

図目次

表目次

第1章　序論

1.1 背景

インターネット技術の急速な発展にともない，サービス提供者の手続きの効率化および利用者の利便性向上を目的とし，多くの高等教育機関において，教育・研究・業務など様々な分野において情報システム化が進められてきた．しかし，情報システムの大多数は，各部署・部局など，さらには研究室ごとで独自に構築・運用されてきたため，これらの情報システムは，それぞれ独自の認証機構を備えるとともに，ユーザに対して個別の ID とパスワードの発行を行ってきた．その結果，大学のいたる所で情報システムが乱立し，たとえ個々の情報システムとしては機能しても，それらの公開方法や利用範囲に一貫性がなく，システム間の連携も考慮されていないのが実情であった．また，このような環境では，ユーザは，目的の情報システムの URL を自身で管理したり，情報システムごとの ID とパスワードの対を覚えておく必要があったりと，作業効率の低下のみならず，ID とパスワードの対をメモに残すなど，セキュリティの観点でも問題があった．さらに，情報システムの運用においても，情報システムごとに独自の認証機構を持つことは，人員の異動にともなう内部データ変更作業や認証機構の維持管理など，コストの増大を招いてきた．そのため，当初の目的とは裏腹に，情報システム提供者の負担の増加および利用者の利便性の低下を招く結果となってしまっていた．金沢大学（以下，本学と記載）も例外ではなく，これまで様々な場所で情報システムが乱立し，サービス利用者は目的の情報システムを探し出すのに苦慮し，かつそれぞれが個別の ID とパスワードを発行していたため，ポストイットなどにメモし，PC に張りつけている光景が散見されていた．さらには，認証機構に職員番号と生年月日を求めるシステムも少なくなく，セキュリティの問題も指摘されて

いた．また，乱立する情報システムに対して，管理者の数が少なく，管理者にかかる負担が増加の一途を辿っていた．

そのため，本学の情報基盤整備における次のフェーズとして，“情報化の推進”としての ICT インフラの整備にとどまらず, その上にたって行われる教育・研究・業務に必要な情報を効率良く利活用できる“より上位レベルの情報基盤整備”が重要になってきている．

上位レベルの情報基盤整備へ向けて，各所に分散する情報システムの一元化・融合化が重要であり，必要な要素技術として，シングルサインオンおよび属性共有と呼ばれる技術が注目されている．高橋[1]はシングルサインオンを「ユーザが一度認証されると，その後，複数のサービスを認証手続きなしで利用可能にする技術」，属性共有を「あるユーザに関する情報（属性情報）を複数のサービスでプライバシーを保護しつつ安全に共有する技術」と定義している．シングルサインオンおよび属性共有を利用した大学内統合認証基盤の構築においては，名古屋大学[2][3]，大阪大学[4]，東京工業大学[5]，徳島大学[6]，佐賀大学[7]などいくつかの大学で先行的に行われている例が報告されている．しかし，いずれの例においてもシンプルなシングルサインオンと最低限の属性共有の実装に留まっている．さらに，その多くは，シングルサインオン部分においては非常に高価なベンダのアプライアンスを導入していたり，属性共有部分においては作り込みの部分が大きかったりと，モデルケースとして利用することが困難な事例が多く，そのため他大学への波及効果はそれほど大きくないのが実情である．

また，一般企業などでは浸透しつつあるシングルサインオンおよび属性共有が大学で普及しにくい理由として, 情報システムの管理形態の多様さが挙げられる．大学で構築されている情報システムは，教育・研究・業務など，その目的が多岐にわたることから，当然，それぞれ利用できるユーザの範囲が異なり，また，システムによって運用ポリシも異なる. さらに, 大学には多種多様な役割（ロール）を持つユーザが存在する．たとえば，大学から給与が支給されるティーチングアシスタント（TA）やリサーチアシスタント（RA）として業務を担う学生や，大学職員でありながら，その大学の社会人学生でもある場合などが挙げられる．このようなユーザには，それぞれの属性に応じた個々の情報システムへの認証・認可権限の設定が必要となり，多くの場合，複数の ID とパスワードを配布し，それぞれの情報システムで認証および認可をする結果に至る．そのため，大学特有の複雑な所属形態にすべて適合するシステムを構築するのは容易ではない．

一方で，現在，国立情報学研究所（以下，NII と記載）が中心となり，複数の大学間での情報システムの共有・相互乗り入れの実現を目指して，UPKI イニシアティブ[8]において学術認証フェデレーション（学認：GakuNin）[9]構築プロジェクトが進められている．本プロジェクトは，個々の大学が保有する教育研究用計算機，電子コンテンツ，ネットワークなどの学術情報資源を，大学間で安全・安心に有効活用可能とするもので, GakuNin は大学間認証連携基盤の呼称である．欧米では大学間認証連携は盛んに行われている[10]ことからも今後，GakuNin の必要性はますます高まると予想される．同プロジェクトの試みにより，大学間連携におけるシングルサインオンの環境は整備されつつあるが，欧米に比べると日本の大学においてはまだ普及にはいたっていない．特に米国では EDUCAUSE[11]と呼ばれる IT の活用によって高等教育を進歩させることを使命とする，NPO（非営利団体）が形成され，米国最大級の高等教育機関のひとつとなっている．全米 1800 以上の大学・高等教育機関，約 200 社の IT 関連企業が加盟しており，米国外の高等教育関係組織も加盟している．そのため，多くの組織が協調して活動を行うため，技術の共有や普及が早いということがある．日本でも 2010 年 12 月に日本版 EDUCAUSE として「一般社団法人大学 ICT 推進協議会[12]」が設立されたため，今後は大学間における技術の共有や普及が期待される．本研究では，これまで GakuNin 構築プロジェクトに積極的に参加し，大学間連携に関する知見を蓄積してきた[13]．その中で，GakuNin は，ある程度必要な属性が決まっている大学という大きな組織単位における認証連携においては有効であるが，研究室や研究プロジェクト間といった小さな組織においては共有する属性値がそれぞれ異なるため，そのまま適用することは困難であると考えている．しかし，本学の重要な課題である研究室内に蓄積された学術情報の外部公開にむけて，研究室単位の小規模な組織間連携も視野に入れる必要がある．

上述のように，先行事例や大学間連携などの現況から，大学における統合認証基盤におけるモデルケースとなるような有効なものは存在せず, 本学においても，情報システムの一元化において，解決すべき問題が山積されているといえる．

1.2 問題提起

1.1 節では, 高等教育機関における情報システムを取り巻く環境について述べた．

次に，本学における情報システムの一元化に向けて解決すべき課題とそれらの達成のための問題点について述べる．

- シングルサインオン環境の構築

  これまで情報システムごとに個別に実装されていた認証機構の一元化を行い，シングルサインオン環境を実装する必要がある．そのことで，システム担当者における認証機構に係る運用コスト，利用者における ID/パスワード管理におけるセキュリティリスクの軽減が見込まれる．

  問題点：シングルサインオン構築に係るコストが高くあってはならない．そして，既存の情報システムの認証機構を修正する際，それが大幅なものであってはならない．また，シングルサインオンを実現するために，ID 体系の集約化を図らなければならない．さらに，GakuNin の認証連携および小規模組織間連携における互換性を考慮して構築を行う必要がある．

- 情報システム間における属性共有

  ユーザの属性情報を複数の情報システム間で安全に共有し，教育・研究・業務など，多岐にわたる情報システムにおいて，異なるユーザ利用範囲や運用ポリシに対して柔軟に対応できる機構を構築する必要がある．その際に，ユーザが複数の属性（職分等）を持つ場合でも 1 つの ID とパスワードのみで認証・認可の制御ができる機構を実現する．そのことで，認証および認可に利用するユーザ属性情報の発生源入力・発生源管理が可能になり，情報の正確さと迅速な更新の実現が可能になる．

  問題点：これまで個々のシステムが個別に保持していたユーザの属性情報の格納場所を統一化し，かつ複数の情報システム間で安全に属性を共有できる環境を構築する必要がある．そのためには，情報システムにおいて発生した情報を即時に反映し，ユーザ情報の鮮度が保たれなければならない．さらに，各情報システムによって必要なユーザ属性情報が異なるため，それぞれに必要な情報のみを受け渡すようにし，セキュリティレベルを保つ仕組みを考案する必要がある．

- GakuNin による組織間連携の配慮

  1.1 節で述べたように，個々の大学が保有する様々な学術情報資源を，大学間で安全・安心に有効活用できるように，大学間連携を見据えたうえで，シングルサインオンおよび属性共有環境の構築を行う必要がある．そのことで，柔軟に GakuNin に参加することが可能になる．

  問題点：大学内のみで運用を行っている情報システムであれば，人事データや教務データなどから正確なユーザ情報を利用することができる．しかし，他大学のユーザの本人確認を行うことは困難である．そのため，本学外の構成員の身元を判別し，なりすましの回避を行うことで，組織間においてセキュアにサービスを提供できる仕組みを確立しなければならない．さらに，他組織の構成員の属性情報を安全に扱え，属性に応じたサービスを提供するために，大学内の情報システムを学外の利用者にも利用可能とするよう，設計について配慮を行う必要がある．

- 大学間連携における小規模組織間連携への適用

  大学間連携において，大学という大規模組織間の連携だけではなく，研究室や研究プロジェクト単位などの小規模組織においても対応可能な組織間連携の仕組みを構築し，研究室内に蓄積された学術情報の外部公開を促進を実現する．

  問題点：全学規模の情報システムにおいては，所属や職分などを用いて利用の制限を行うのが一般的であり，どの大学においてもある程度必要な属性が共通しており，GakuNin への応用は可能であると考える．但し，大学間で情報システムを共有する場合は，個人情報保護の観点からも，大学名や職分などの個人を特定できない程度の属性情報の共有が主となると考えられる．一方で，研究室における情報システムでは，取り扱うデータが実験観測データなどである．それらは二度と再現できない貴重なものであるが故，他の研究室のユーザに情報を公開する際には，大学ごとで一括管理していない，研究プロジェクト名や研究チーム名などの属性情報の共有も必要になると考えられる．さらに，研究室や研究プロジェクト間といった小さな組織においては共有する属性値がそれぞれ異なり，GakuNin をそのまま適用することは困難

であると考える．そのため，大学間連携を視野に入れるとともに，研究室単位の小規模な組織間連携にも応用可能な認証連携基盤を構築する必要がある．

1.3 研究目的

本研究では，GakuNin における大学間組織連携および研究室などの小規模組織間連携を視野に入れた，本学における情報システムを一元的に取り扱うことを可能にする，大学特有の多様なシステム管理形態に柔軟に対応可能な統合認証基盤の構築を行うことを目的とする．

さらに，統合認証基盤構築の指針として，本学だけの作り込みのシステムにせず,「金沢大学モデル」として，他大学においても転用可能なモデルケースとなりうるレベルでの構築を目指す．そして，これから情報システムの一元化・融合化を検討している大学関係者の一助となるように設計に配慮を行う．

まず，1.1 節で述べた他大学の先行事例を調査し，利用されているシングルサインオンおよび属性共有技術のうちの 1 つを本学内で実際に稼働している情報システムに適用する．そして，問題点を洗い出すとともに考察を行い，本学の統合認証基盤に利用する技術を選定する．次に，選定した技術を GakuNin での認証連携実験において実際に適用し，1.2 節の問題解決に有効であることを検証する．その検証の後，本学内の統合認証基盤を構築し，今後の大学における統合認証基盤の一つの雛形として有効であることを検証する．さらに，研究室単位の多様な公開ポリシの対応が必要な情報システムにおいても適用し，小規模組織間連携として有効であることを検証する．

最後に，大学間認証連携基盤，本学統合認証基盤，小規模組織間認証連携基盤それぞれの連携についても考察を行う．

1.4 関連研究

本節では，大学におけるシングルサインオンの先行研究として，名古屋大学の統合認証基盤で利用されている Central Authentication Service[14]（以下，CAS と記

載），大阪大学，東京工業大学の統合認証基盤で利用されている Public Key Infrastructure[15]（以下，PKI と記載），NII の大学間認証連携で推進されている Shibboleth[16]について説明を行う．

1.4.1 Central Authentication Service

CAS は，Java Architecture Special Interest Group（JA-SIG）[17]のオフィシャルプロジェクトとしてオープンソースとして継続的に開発が続けられている，Web ベースのアプリケーションにおいてシングルサインオン環境を実現できるソフトウェアである．

CAS Version3 においては，Dependency Injection コンテナ[18]である Spring Framework[19]によって実装されている．CAS の特徴を以下に述べる．

- 認証においては HTTP リダイレクション，Cookie，URL パラメータなどの標準的な技術しか用いないため，簡単で軽く，プラットフォームの違いに左右されにくい．
- CAS サーバを利用して認証を行う Web アプリケーションを CAS クライアントと呼ぶ．
- CAS クライアントは Apache[20]，PHP[21]などの Web アプリケーションで標準的に使用される多様な種類のライブラリが用意されているため，様々なシステムに容易に導入することができる．
- CAS サーバ自体は認証に必要な情報（ID, パスワード）を持たないため，LDAP サーバや SQL データベースなどの外部データを利用する必要がある．
- ユーザ ID，パスワードはユーザから CAS サーバだけに送信され，CAS クライアントには送信されない．

CAS の認証メカニズムについて図 1-1 を用いて説明する．説明文中のカッコ内

の数字は，図に記載された矢印の番号と対応する．

ここでは

CAS サーバの URL は

https://cas.server/

CAS クライアントの URL は

https://cas.client/

であるものと仮定する．

最初にユーザは，https://cas.client/ にアクセスする（①）．最初は認証を行っていないため，CAS クライアントは CAS サーバのログインサーブレットである https://cas.server/login に通信をリダイレクトし，その際に，service パラメータとして自身の URL である https://cas.client/ を挿入し, CAS サーバに再転送先の URL を伝える．すなわち, ユーザの Web ブラウザは URL : https://cas.server/login?service =https://cas.client/ を受け取り，認証画面を提示する（②）．ユーザは，認証のための ID とパスワードを入力する（③）．CAS サーバは外部認証サーバ（LDAP サーバなど）を参照し，ユーザが入力した情報が正しいか認証を行う（④，⑤）．認証

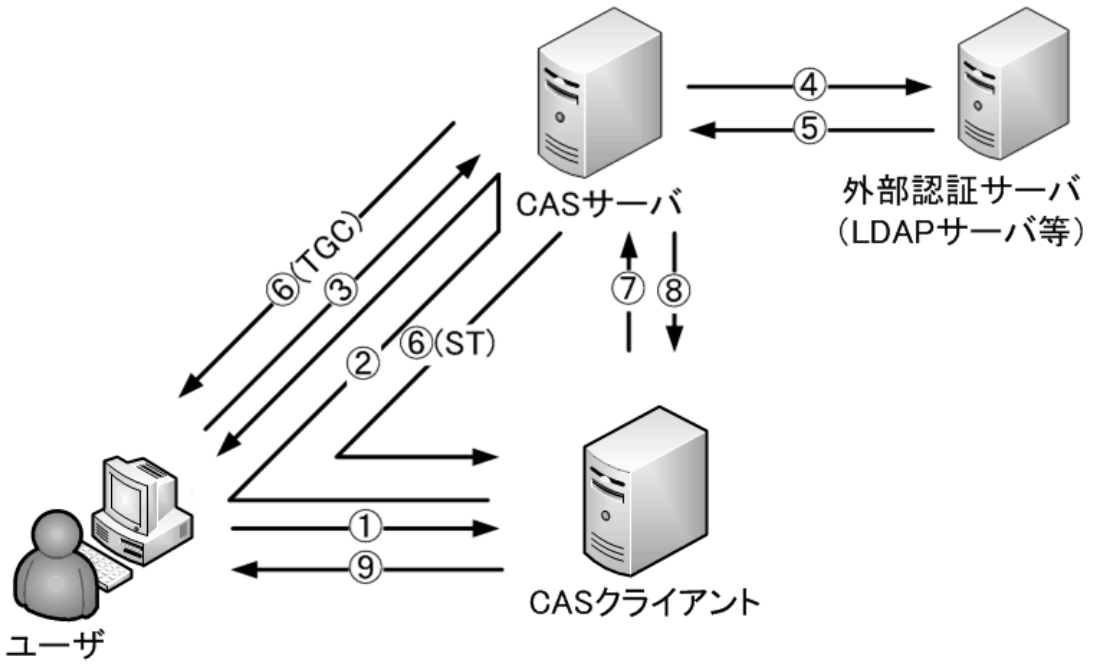

図 1-1　CAS 動作概念図

に成功すると，CAS サーバはユーザの Web ブラウザに対して，Ticket Granting coolie（TGC）と呼ばれる，ブラウザが認証済みかを判断するクッキーを配布し，service パラメータで指定された URL に対してリダイレクションを行う．リダイレクションの際に，URL には ticket パラメータとして Service Ticket（ST）と呼ばれる，CAS クライアントにアクセスする際のワンタイムチケットが含まれる．つまり，https://cas.client/?ticket=ST-xxxxxx の形になる（⑥）．ticket パラメータを受理した CAS クライアントは，ST を CAS サーバに送付する（⑦）．CAS サーバは ST の正当性検査のための Validation サーブレットで ST に問題がないことを確認し，認証を行ったユーザの情報を CAS クライアントに送付する（⑧）．そして，CAS クライアントはユーザの Web ブラウザに対してサービスを提供する（⑨）．2 回目以降は，Web ブラウザが有効な TGC を持つため，ID，パスワードの認証手続きが省略され，それ以外の処理が行われる．

さらに，CAS の動作を詳細に説明するため，以下にユーザのブラウザが送受信する HTTP ヘッダを示す．

まず，CAS クライアントの URL である，https://cas.client/service/ にアクセスすると，service パラメータに CAS クライアントの URL を引数とし，CAS サーバへのリダイレクトが発生する．

```
GET /service/ HTTP/1.1
Host: cas.client

HTTP/1.1 302 Found
Location: https://cas.server/?service=https://cas.client/service/
```

CAS サーバの login サーブレットへのリダイレクトが発生する．

```
GET /?service=https://cas.client/service/ HTTP/1.1
Host: cas.server

HTTP/1.0 302 Moved Temporarily
Location: https://cas.server/login?service=https://cas.client/service/
```

ログイン画面が提示される．

```
GET /cas/login?service=https://cas.client/service/ HTTP/1.1
Host: cas.server

HTTP/1.0 200 OK
```

ID，パスワードを送信する．TGC が Cookie としてセットされ，ST が CAS クライアントの URL の引数に ticket パラメータとして格納されリダイレクトされる．

```
POST /login?service=https://cas.client/service/ HTTP/1.1
Host: cas.server

username=sample&password=samplepw

HTTP/1.0 302 Moved Temporarily
Set-Cookie: CASTGC=TGT-117-KjEERQjVxPtBmZbcIWOG-cas
Location: https://cas.client/service/?ticket=ST-100-S36Dv9cYuv7hL-cas
```

ST が検証され，問題がなければサービスが提示される．

```
GET /service/?ticket=ST-10220-S36DvxFYTdUG9cYuv7hL-cas HTTP/1.1
Host: cas.client

HTTP/1.1 200 OK
```

次に，別の CAS クライアントである https://cas.client2/service2/ にアクセスすると，CAS サーバにリダイレクトが発生する．

```
GET /service2/ HTTP/1.1
Host: cas.client2

HTTP/1.1 302 Found
Location: https://cas.server/login?service=https://cas.client2/service2/
```

TGC を送信し，問題がなければログインは省略され，新たな ticket が発行されて，CAS クライアントへのリダイレクトが行われる．

```
GET /login?service=https://cas.client2/service2/ HTTP/1.1
Host: cas.server
Cookie: CASTGC=TGT-117-KjEERQjVxPtBmZbcIWOG-cas

HTTP/1.0 302 Moved Temporarily
Location: https://cas.client2/service2/?ticket=ST-102-Zt2FhwdHDQyT-cas
```

ST の正当性が検証されると，サービスが提示される．

```
GET /service2/?ticket=ST-10221-Zt2FhwHIdHD7oIVweQyT-cas HTTP/1.1
Host: cas.client2

HTTP/1.1 200 OK
```

1.4.2 Public Key Infrastructure

PKI は,「公開鍵基盤」と訳され,「公開鍵」暗号方式による「基盤（インフラ）」を表す．公開鍵暗号方式では，暗号化と復号化の鍵が異なるのが特徴である．図 1-2 に公開鍵暗号方式の概要を示す.

例として，A さんが B さんに文書を送付することを考える．A さんは，文書を暗号化するため，B さんの公開鍵によって文書を暗号化する．公開鍵は一般的に

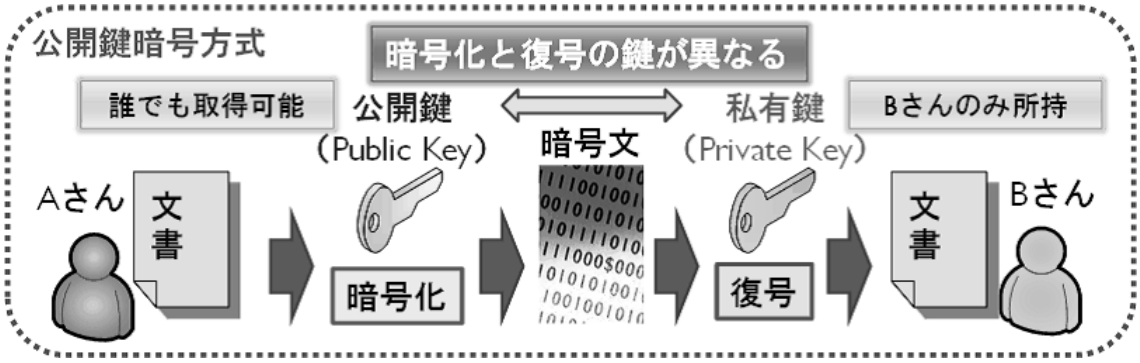

図 1-2　公開鍵暗号方式概要

は誰でも取得可能である．但し，公開鍵暗号方式では「公開鍵で暗号化した情報は，その私有鍵でないと復号できない」という性質を持つ．そのため，暗号化した文書はBさんが持つ私有鍵でのみ復号可能なため，Aさんは誰にも閲覧されることなくBさんに文書を送付できることが可能になる．

つまりPKIは，特定のアプリケーションやプロトコルを指すものではなく，公開鍵暗号方式という技術を利用したセキュリティの基盤を指す．

PKIを用いた認証方法として，クライアント認証がある．その際に，サーバにはサーバ証明書，ユーザにはクライアント証明書を配布する必要がある．但し，一般的には，証明書に付随する公開鍵が本人のものであるかを保証するために，信頼できる第三者機関（Trusted Third Party）に公開鍵の所有者を保証してもらう必要がある．第三者機関は，公開鍵とその所有者を証明する情報を含んだ証明書を発行する．その証明書を発行する第三者機関を認証局（CA；Certification Authority）と呼ぶ．

図1-3にクライアント認証の概念図を示す．CAはWebサーバに対してはサーバ証明書を，クライアントに対してはクライアント証明書を発行する．クライアントがWebサーバにアクセスすると（①），Webサーバは Helloメッセージとサーバ証明書をクライアントに送信する（②）．クライアントはサーバ証明書の検証を行う（③）．具体的にはCAが管理している証明書の発行・失効情報などが登録されているリポジトリを参照し，サーバ証明書が正当なものであるかを確認する．サーバ証明書の正当性が確認できた後，乱数を生成し，サーバ証明書に付随する公開鍵で暗号化を行う（④）．そして，②で送信されたHelloメッセージをクライアントの私有鍵で署名を行う（⑤）．そして，暗号化した乱数，クライアント証明書，署名をWebサーバに送信する（⑥）．Webサーバはクライアント証明書の正当性をサーバ証明書と同様の方法で検証し，クライアントに付随する公開鍵で署名の検証を行う（⑦）．検証後，サーバ私有鍵で乱数を復号し（⑧），乱数からクライアントとの共有鍵を生成する（⑨）．最終的に，クライアントとWebサーバはこの共有鍵で暗号化通信を行う．

セキュリティの強度は極めて高いが，CAの運用コストが非常に高いという問題がある．また，クライアント数に比例して証明書の登録や失効手続きが煩雑になる．

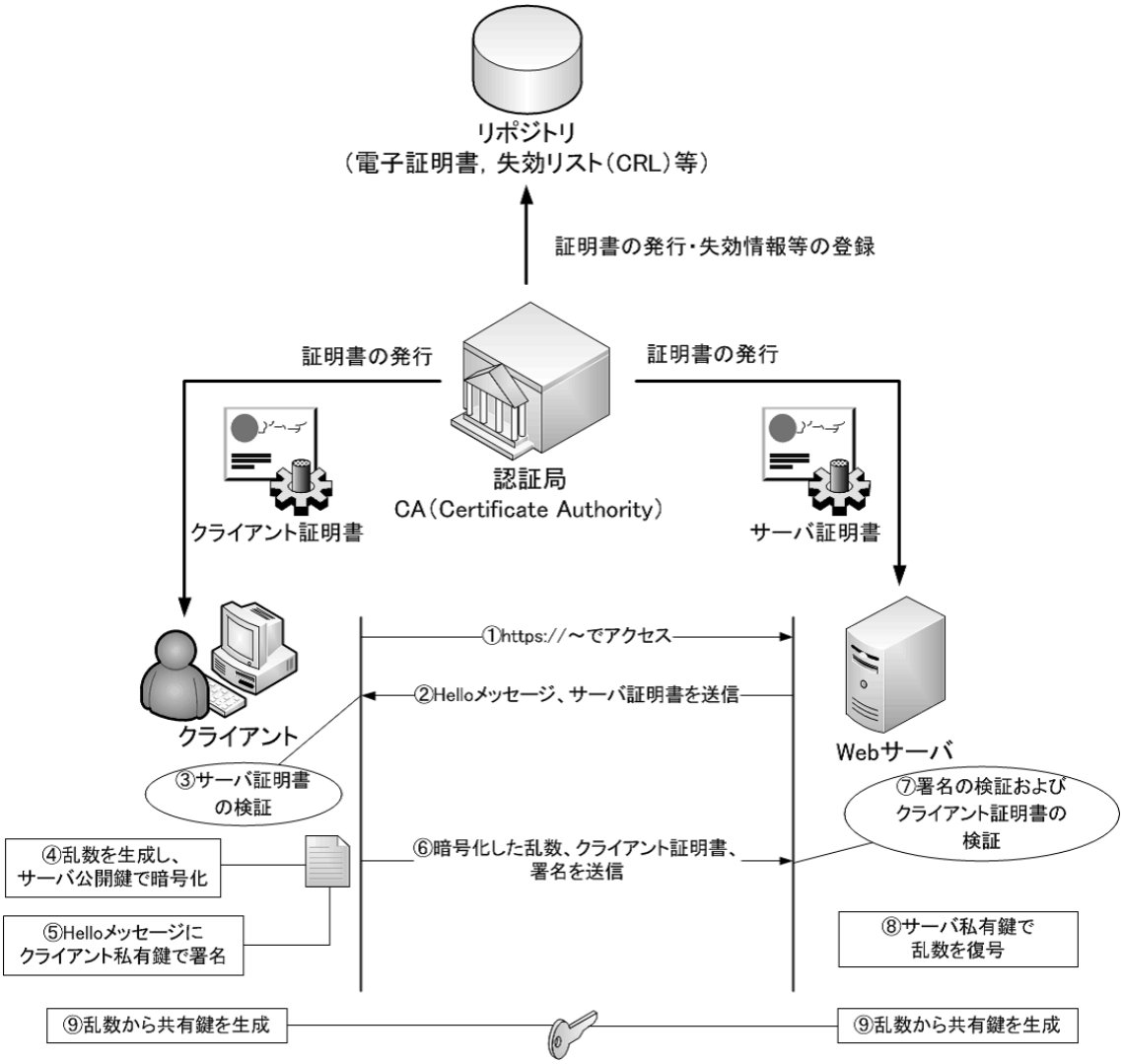

図 1-3 クライアント認証概念図

1.4.3 Shibboleth

Shibboleth は，Internet2[22]の Middleware Architecture Committee for Education プロジェクト[23]の 1 つで，SAML2.0[24]をベースとした，異なる情報システム間でのシングルサインオンおよび属性共有を実現するオープンソースソフトウェアである．なお，SAML とは，XML を基盤にした，異なる Web サービス間での認証情報，属性情報，認可情報を交換するための標準の仕様である．また，フェデレーションと呼ばれる，お互いに信頼されたサーバ間で組織を構成し，フェデレー

ション内でのみ各種情報の交換を可能とすることができる．フェデレーションはメタデータとして各サーバ間で共有される．Shibboleth では，Identity Provider（以下，IdP と記載），Service Provider（以下，SP と記載），Discovery Service（以下，DS と記載）の 3 つで構成される．以下にそれぞれの役割を述べる．

- IdP

  IdP は主にユーザを認証する役割と，ユーザの属性情報を SP に送信する役割を持つ．ユーザが SP にアクセスすると，SP は IdP にリダイレクトを行う．IdP は ID/パスワード認証やクライアント証明書認証などの方法で認証を行う．そして，SP が要求する属性を SP に対して送信する．

- SP

  SP は主に，ユーザの認証を IdP に要求する役割とユーザの属性を IdP から受信し，アプリケーションに渡す役割を持つ．ユーザが SP にアクセスすると，IdP にリダイレクトを行い，IdP から認証結果を受け取る．認証が成功したのち，IdP に対して必要とする属性を要求し，受け取ったものをアプリケーションに渡す．

- DS

  複数の IdP が存在する場合に，ユーザが適当な IdP を決定するための情報を提供する役割を持つ．ユーザが SP に対してアクセスした時，フェデレーション内における IdP の一覧を提示する．ユーザは提示された一覧から適当な IdP を選択し，認証を行う．

図 1-4 に示す Shibboleth 動作概念図に基づいて，Shibboleth の動作について説明する．説明文中の括弧内の数字は，図に記載された矢印の番号と対応する．組織 A 所属のユーザ A が組織 B の情報システム（SP）を利用したい場合を例に示す．まず，ユーザ A は組織 B の SP にアクセスを試みる（①）．このとき，SP はユーザ A の認証をしていないため， DS にリダイレクトを行い，ユーザ A に適切な IdP を選択させるために，DS は選択可能な IdP のリストをユーザ A に提示する．（②，③）．ユーザ A は組織 A の IdP で，ID/パスワード認証やクライアント証明書認証などの方法でユーザ認証を行う（④）． IdP は SP に認証結果を返し，成功

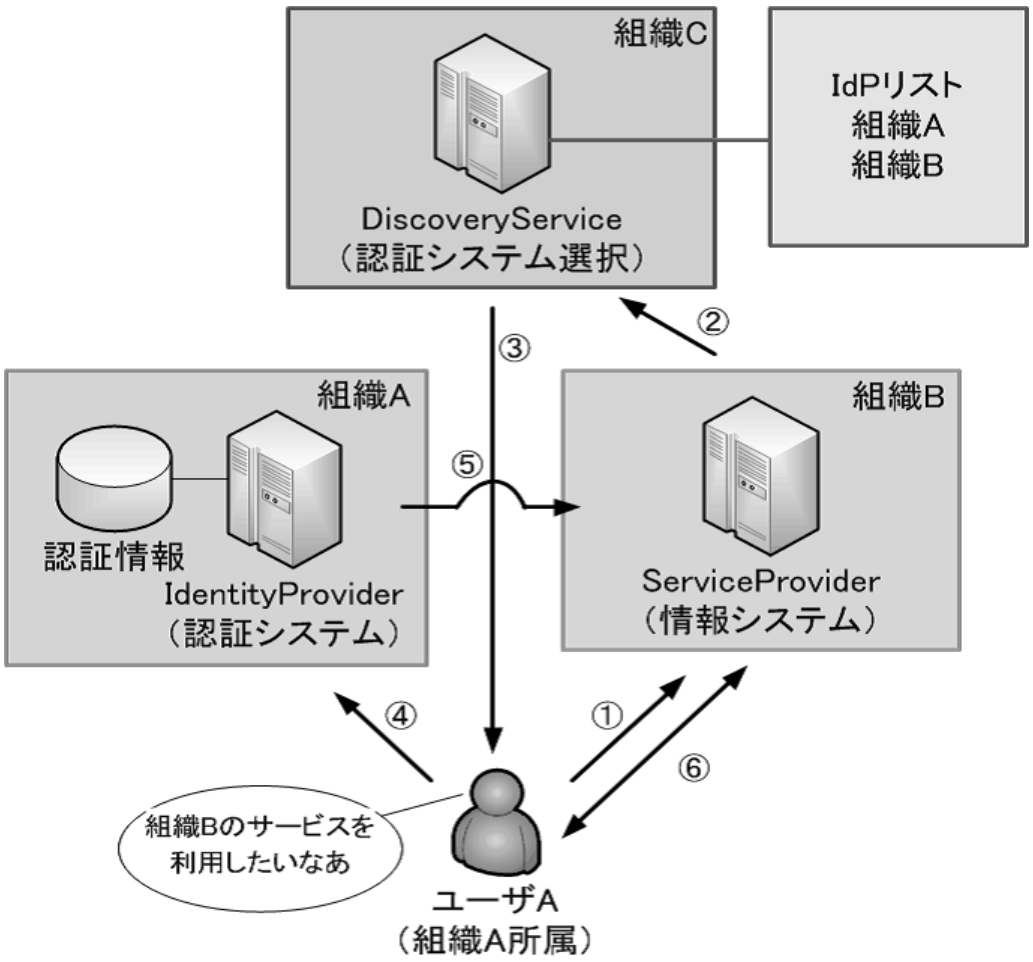

図 1-4 Shibboleth 動作概念図

の結果を受け取った場合に，SP は必要な属性を IdP に要求し，その返却値を SP のアプリケーションに渡す（⑤）．SP はその情報を基に，ユーザ A の属性に応じたサービスを提供する（⑥）．

Shibboleth の動作についてさらに詳細に説明するため，以下にユーザのブラウザが送受信する HTTP ヘッダを示す．

以下の例では，SP サーバである sp.server1 の/secure/ および，sp.server2 の /secure2/ のコンテンツに順番にアクセスしたときの HTTP ヘッダである．

まず, SP として動作している URL である https://sp.server1/secure/ にアクセスを行うと，DS へとリダイレクトが行われる．その際に，アクセスした SP の URL は_shibstate_xxxxxxxx（x は任意の英数字）の変数名で Cookie に保持される．

```
GET /secure/ HTTP/1.1
Host: sp.server1

HTTP/1.0 302 Found
Set-Cookie: _shibstate_f219155c=https://sp.server1/secure/;
Location: https://ds.server/WAYF?entityID=https://sp.server1/shibboleth-sp&return=https://sp.server1/Shibboleth.sso/DS?SAMLDS=1&target=cookie:f219155c
```

DS の画面を取得する．

```
GET /WAYF?entityID=https://sp.server1/shibboleth-sp&return=https://sp.server1/Shibboleth.sso/DS?SAMLDS=1&target=cookie:f219155c HTTP/1.1
Host: ds.server

HTTP/1.0 200 OK
```

DS が提示する IdP リストから自組織の IdP サーバを選択する．その際に，選択した IdP の情報は origin パラメータに付加される．DS から SP へのリダイレクトが発生し，IdP 情報は変数名が_saml_idp の Cookie に保持される．_saml_idp の値は, IdP の情報 (https://idp.server/idp/shibboleth) が Base64 でエンコードされたものが入る．_saml_idp は SAML で「Common Domain Cookie」として定義されており，複数の IdP が存在する状況で，IdP を特定するものである．

```
GET /ds/WAYF?entityID=https://sp.server1/shibboleth-sp&return=https://sp.server1/Shibboleth.sso/DS?SAMLDS=1&target=cookie:f219155c&returnIDxParam=entityID&cache=perm&action=selection&origin=https://idp.server/idp/shibboleth
Host: ds.server

HTTP/1.0 302 Moved Temporarily
Set-Cookie: _saml_idp=aHR0cHM6Ly9hdXRoLXNzby5kYi5rYW5hemF3YS11LmFjL
```

```
mpwL2lkcC9zaGliYm9sZXRo+;
Location: https://sp.server1/Shibboleth.sso/DS?SAMLDS=1&target=cookie:f219155c&e
ntityID=https://idp.server/idp/shibboleth
```

SP はクライアントのブラウザを経由させ，IdP に対して，認証要求プロトコルである「AuthnRequest」を送付する．AuthnRequest は SAMLRequest パラメータに base64 でエンコードされた形で IdP に送付される．

```
GET /Shibboleth.sso/DS?SAMLDS=1&target=cookie:f219155c&entityID=https://idp.se
rver/idp/shibboleth HTTP/1.1
Host: sp.server1
Cookie: _shibstate_f219155c=https://sp.server1/secure/

HTTP/1.0 302 Found
Location: https://idp.server/idp/profile/SAML2/Redirect/SSO?SAMLRequest=hZJNT4
MwHMahmeITt966YTcdfjFf60rm705cVXo3M4JbESKCcNRf+f1Xok8=&RelayState=c
ookie:f219155
```

以下に AuthnRequest をデコードしたものを以下に示す.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol" Assertion
….. ID="_0e657123a122b74685325c9d91caf792" IssueInstant="2010-12-08T01:08:5
2Z" Version="2.0">
<saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
https://sp.server1/
</saml:Issuer>
…….
</samlp:AuthnRequest>
```

それぞれ

ID：メッセージにつける一意な識別子

IssueInstant：メッセージ発行日時

Version：SAML プロトコルのバージョン，2.0.

Issuer：要求メッセージの作成エンティティ

が記載される．

IdP に AuthenRequest が送付されると，IdP はクライアント（Web ブラウザ）に認証を促すため，認証画面へリダイレクトを行う．

```
GET　/idp/profile/SAML2/Redirect/SSO?SAMLRequest=hZJNT4MwHMahmeITt966Y
TcdfjFf60rm705cVXo3M4JbESKCcNRf+f1Xok8=&RelayState=cookie:f219155 HTTP/
1.1
Host: idp.server

HTTP/1.0 302 Moved Temporarily
Set-Cookie: _idp_authn_lc_key=354b3d03-3d0c-49aa-abeb-544a7b17cdd8;
Location: https://idp.server /idp/Authn/UserPassword
```

Web ブラウザに IdP の認証画面が提示される．

```
GET /idp/Authn/UserPassword HTTP/1.1
Cookie: _idp_authn_lc_key=354b3d03-3d0c-49aa-abeb-544a7b17cdd8;
Host: idp.server

HTTP/1.0 200 OK
```

ID，パスワードを入力し，認証に成功すると_idp_session の Cookie がセットされる．_idp_session にはクライアントに与えられる，クライアントの IP アドレスとランダムな文字列が組み合わせたものを Base64 でエンコードした値が格納される．_idp_session が有効な限り，ID，パスワードの入力は省略される．

```
POST /idp/Authn/UserPassword HTTP/1.1
Cookie: _idp_authn_lc_key=354b3d03-3d0c-49aa-abeb-544a7b17cdd8;
Host: idp.server

j_username=sample&j_password=samplepw

HTTP/1.0 200 OK
Set-Cookie: _idp_session=MTMzLjI4LjI1LjIzNQ==|ZTdkZGM1ZGY5MmQ4NjllODhl
```

```
YjdiOWUyMzdmZjUzMzc4NzQwYWRhYTU1Njc0Y2YyZTUyYTdmY2Y5YWEwMzg4Mg==|Y2t9kTkQs+kDCRM9pupSvSEGZf0=;
Set-Cookie: _idp_authn_lc_key=354b3d03-3d0c-49aa-abeb-544a7b17cdd8; Version=1; Max-Age=0; Expires=Thu, 01-Jan-1970 00:00:10 GMT;
```

IdP はクライアントのブラウザを経由させ，SP に対して，「アサーション」を送付する．送付方法は HTTP-POST で送信する．アサーションはユーザの認証結果，属性情報，認可情報が含まれており， SAMLResponse パラメータに base64 でエンコードされた形で IdP に送付される．

```
POST /Shibboleth.sso/SAML2/POST HTTP/1.1
Host: sp.server1
Cookie: _shibstate_f219155c=https://sp.server1/secure/

SAMLResponse=PD94bWwgdmVyc2lvbj0iM…….S4wIiBlbmNvZGluZ

HTTP/1.0 302 Found
Set-Cookie: _shibstate f219155c=; path=/; expires=Mon, 01 Jan 2001 00:00:00 GMT
Set-Cookie: _shibsession_64656661756c7468747470733a2f2f7370312e64622e6b616e617a6177612d752e61632e6a702f73686962626f6c6574682d7370=_01733dcf957f526a76a85086739d88d6;
Location: https://sp.server1/secure/
```

アサーションである SAMLResponse をデコードしたものを以下に示す．

```
<?xml version="1.0" encoding="UTF-8"?>
<saml2p:Response xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol" ID="_17c3f4f55b43a17c6dea3481ea2d7c14" InResponseTo="_0e657123a122b74685325c9d91caf792" IssueInstant="2010-12-08T01:14:40.792Z" Version="2.0">
…….
<saml2p:Status>
<saml2p:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</saml2p:Status>
```

```
…….
<saml2:EncryptedAssertion  xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
<xenc:EncryptedData….. >
…….
<ds:X509Certificate>
MIIE1zCCA7+gAw…Q/Z9IVtw==
</ds:X509Certificate>
…….
<xenc:CipherValue>dqLMnp9JRgG6BY54QpQhzu5NA==</xenc:CipherValue>
…….
<xenc:CipherValue>sNgTTrhk2kOX6Cn…awgDfnzCg==</xenc:CipherValue>
</xenc:CipherData>
</xenc:EncryptedData>
</saml2:EncryptedAssertion>
</saml2p:Response>
```

InResponseTo で AuthnRequest と対応づけられている．<saml2p:Status>タグ内で Success と表示されており，認証が成功したことを示している．saml2:EncryptedAssertion となっているのは，アサーションを XML 暗号していることを示す．<ds:X509Certificate>タグ内には SP の公開鍵が記載してあり，1 つ目の<xenc:CipherValue>タグ内には IdP と SP 間で使用する共通鍵を SP の公開鍵で暗号化したものが記載されている．2 つ目の<xenc:CipherValue>タグ内には，ユーザの属性情報が記載されている．

上記では，アサーション部分は，SP の公開鍵によって暗号化された，SP と IdP 間の共通鍵を用いて暗号化している．暗号化しなかった場合のアサーションを以下に示す．

```
<saml2:Assertion  ……
……
<saml2:AttributeStatement>
<saml2:Attribute  FriendlyName="mail"  ……. >
```

```
<saml2:AttributeValue …….>sample@sample.ac.jp</saml2:AttributeValue>
</saml2:Attribute>
…….
</saml2:AttributeStatement>
</saml2:Assertion>
```

<saml2:AttributeStatement>タグ内に，ユーザの属性情報が記載される．<saml2:Attribute FriendlyName="mail"タグは mail という属性名で送信することを示し，<saml2:AttributeValue タグ内で属性値である sample@sample.ac.jp が記載されている．

JavaScript の onload 属性により，https://sp.server1/secure/ へのアクセスが行われる．セットされた_shibsession_xxxxxxxx（x は任意の英数字）の Cookie の値の正当性が検証されたのち，SP はクライアントに応じたサービスを提供する．

```
GET /secure/ HTTP/1.1
Host: sp.server1
Cookie: _shibsession_64656661756c7468747470733a2f2f7370312e64622e6b616e617a6177612d752e61632e6a702f73686962626f6c6574682d7370=_01733dcf957f526a76a85086739d88d

HTTP/1.1 200 OK
```

次に，別組織が運用する SP である，https://sp.server2/secure2/ にアクセスすると，再度 DS へのリダイレクトが発生する．

```
GET /secure2/ HTTP/1.1
Host: sp.server2

HTTP/1.1 302 Found
Set-Cookie: _shibstate_9c3d5504=https://sp.server2/secure2/;
Location: https://ds.server/WAYF?entityID=https://sp.server2/shibboleth-sp&return=https://sp.server2/Shibboleth.sso/DS?SAMLDS=1&target=cookie:9c3d5504
```

DS は_saml_idp の値から，IdP 情報を取得し，SP へのリダイレクトを行う．

```
GET  WAYF?entityID=https://sp.server2/shibboleth-sp&return=https://sp.server2/Shibboleth.sso/DS?SAMLDS=1&target=cookie:9c3d5504  HTTP/1.1
Host:  ds.server
Cookie:  _saml_idp=aHR0cHM6Ly9hdXRoLXNzby5kYi5rYW5hemF3YS11LmFjLmpwL2lkcC9zaGliYm9sZXRo+

HTTP/1.0  302  Moved  Temporarily
Location:  https://sp.server2/Shibboleth.sso/DS?SAMLDS=1&target=cookie:9c3d5504&entityID=https://idp.server/idp/shibboleth
```

SP（sp.server2）は IdP に対して AuthnRequest を送付する．

```
GET  /Shibboleth.sso/DS?SAMLDS=1&target=cookie:9c3d5504&entityID=https://idp.server/idp/shibboleth  HTTP/1.1
Cookie:  _shibstate_9c3d5504=https://sp.server2/secure2/
Host:  sp.server2

HTTP/1.0  302  Found
Location:  https://idp.server/idp/profile/SAML2/Redirect/SSO?SAMLRequest=fZJBU4MwEIX71fm...FAWAgJIzQaRv7=&RelayState=cookie:9c3d5504
```

_idp_session が既にセットされているので，正当性を検証後，問題がなければ ID とパスワードの入力は省略される．

```
GET  /idp/profile/SAML2/Redirect/SSO?SAMLRequest=fZJBU4MwEIX71fm...FAWAgJIzQaRv7=&RelayState=cookie:9c3d5504  HTTP/1.1
Host:  idp.server
Cookie:  _idp_session=  MTMzLjI4LjI1LjIzNQ==|ZTdkZGM1ZGY5MmQ4NjllODhlYjdiOWUyMzdmZjUzMzc4NzQwYWRhYTU1Njc0Y2YyZTUyYTdmY2Y5YWEwMzg4Mg==|Y2t9kTkQs+kDCRM9pupSvSEGZf0=;

HTTP/1.0  200  OK
```

```
Set-Cookie: _idp_authn_lc_key=8afed1bb-0e30-42db-8d78-9eaf0c2998a8;
```

IdP から SP に対してアサーションが送付され，sp.server2 とのセッションである_shibsession_xxxxxxxx が Cookie にセットされ, sp.server2 へリダイレクトが発生する.

```
POST /Shibboleth.sso/SAML2/POST HTTP/1.1
Host: sp.server2
Cookie: _shibstate_:9c3d5504=https://sp.server2/secure2/

RelayState=cookie:9c3d5504&SAMLResponse=PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0…..iVVRGLTg

HTTP/1.0 302 Found
Set-Cookie: _shibstate_9c3d5504=; path=/; expires=Mon, 01 Jan 2001 00:00:00 GMT
Set-Cookie: _shibsession_64656661756c7468747470733a2f2f656475726f616d736869622e6e69692e61632e6a702f73686962626f6c6574682d7370=_e0699a2eff38164788a84190f67df98f;
Location: https://sp.server2/secure2/
```

発行された_shibsession_xxxxxxxx の正当性を検証し，問題がなければサービスが開始される.

```
GET /secure2/ HTTP/1.1
Host: sp.server2
Cookie: _shibsession_64656661756c7468747470733a2f2f656475726f616d736869622e6e69692e61632e6a702f73686962626f6c6574682d7370=_e0699a2eff38164788a84190f67df98f

HTTP/1.0 200 OK
```

このように，クライアントと SP・IdP 間（end-to-end）では Cookie を用いているが，IdP と SP 間では SAML2.0 により，Cookie を用いることなくステートフルなアクセスを実現している.

1.5 本論文の構成

本章では，大学における情報システムの現状と，解決すべき課題およびその問題点について述べるとともに，他大学において先行的に用いられているシングルサインオンおよび属性共有の技術について考察した．

2~5 章では，GakuNin における大学間連携および研究室などの小規模組織間連携を視野に入れた，本学における情報システムを一元的に取り扱うことを可能にする，大学特有の多様なシステム管理形態に柔軟に対応可能な統合認証基盤の構築方法を考案し，大学の統合認証基盤おける 1 つの雛形になることを目指す．

以下に各章の概要を述べる．

2 章では，1.4 節で述べた大学における統合認証基盤の先行研究の中で，本学の統合認証基盤技術として，費用面および技術面において比較的導入が容易であると判断した CAS を実際に本学内の情報システムに適用した結果について述べる．そして，CAS における問題点について議論するとともに，Shibboleth と比較を行い，本学の統合認証基盤において Shibboleth を選定した理由について述べる．さらに，本研究におけるシステム開発の範囲についても議論する．

3 章では，本学における統合認証基盤構築の内，大学間認証連携基盤について取り扱う．ここでは，NII が推進する GakuNin を利用して，大学という組織を越えて情報システムを利用する際のユーザ属性情報の取り扱いについて議論する．さらに，本学は GakuNin フェデレーションを利用し，大学間における安全なデータ共有を目的としたデータ共有システムを提供しており，その構築したシステムについても説明する．

4 章では，本学における認証連携基盤の構築について述べる．3 章で蓄積した Shibboleth の知見を用いて，シングルサインオンによるユーザ認証と同時に，教員・職員・学生など各自の職分も認識でき，その職分に応じて利用許可されている情報システムが再度の認証なしに利用可能になる統合認証システムの構築について議論を行う．そして構築の際に発生した Shibboleth の問題点とその解決策について議論するとともに，システムの実装方法および構築したシステムの評価に

ついても述べる．

5 章では，本学における統合認証基盤構築の内，研究室などの小規模組織間認証連携基盤について取り扱う．研究室などの小規模組織において，組織を超えてデータを共有する際に，Shibboleth を利用してどのようにしてデータ共有相手を特定するかについて議論する．そして，多様な公開ポリシに容易に対応するために開発したデータ相互参照システムである「ARCADE」について説明する．

6 章では，3~5 章で説明した開発システムの連携について述べるとともに，最終的な開発システムの連携構成について議論を行う．

7 章では，1 章から 6 章で述べた研究成果をまとめ，本研究の総括とする．

第2章　資産管理に基づく適切なソフトウェア配布システムの構築

2.1 はじめに

本章では，1 章で説明した他大学で先行的に統合認証基盤を構築している事例の内，CAS を用いて実際に金沢大学内のサービスに適用し，検証を行った結果について述べる．CAS を選定した理由は，1.5 節で述べたとおり，PKI と Shibboleth を含めたの 3 つの技術のうち 1 番情報が多かったのと，安価に構築できると判断したためである．

本学総合メディア基盤センター[25]（以下，センターと記載）では，ウイルス対策ソフトウェアやファイル暗号化ソフトウェアなどのセキュリティ対策ソフトを中心に，学内教職員ユーザ（金沢大学が雇用している教職員，以下，ユーザと呼ぶ)に対して配布するサービスを行っている. ソフトウェアを配布する際には，配布したソフトウェアがどの PC にインストールされているかを把握する必要があり，これまではその方法として，ユーザ認証と PC の IP アドレスで管理していた．

一方，本学では 2008 年 7 月 1 日より，ソフトウェア資産管理（以下，資産管理と呼ぶ）の調査を全学的に開始し，大学所有の PC やソフトウェアライセンスの管理を行っている．この資産管理により，PC，ソフトウェアといった資産を誰が管理しているか把握できるよう，資産全てに対して管理者を特定可能な固有の ID を割り当てた．

このことで，ユーザに対して，認証と併せて資産管理上の PC 固有の ID を入力させ，誰がどの PC にインストールしたかまで把握できるソフトウェア配布システムが構築できるようになった．そして，本システムのユーザ認証部分に CAS

を用いて，シングルサインオン環境を実現した．

まず，資産管理の際に必要になる様々な ID について説明した後，ソフトウェア配布システムの概要について説明し，実装について述べる．また，システムにアクセスする際のユーザ認証についても述べる．

2.2 ソフトウェア配布システム

本節では，ソフトウェア配布システムを使用する際に必要となる各種 ID，システムを構成するサーバについて説明する．そして，ユーザがシステムからソフトウェアをダウンロードするまでの流れについて述べる．

2.2.1 システムにおける ID の役割

ソフトウェア配布システムにおいては，ネットワーク ID，資産管理用管理者 ID（以下，管理者 ID と呼ぶ），機器 ID，ソフトウェア ID を使用する．以下にそれぞれの ID の役割について説明する．

(1) ネットワーク ID

ネットワーク ID は，ユーザがセンターで提供しているサービスを利用する際に使用する識別子である．学内のネットワークを利用する際の認証や，学外から学内のネットワークを利用する際の認証（VPN）に使用する．職員番号など，ユーザ固有の ID を認証に利用すると，万一情報が漏れた場合に ID を変更できない。そこで自身で登録・変更可能なネットワーク ID を使用することでセキュリティが向上する．

(2) 管理者 ID

管理者 ID は，資産管理の際に用いるユーザ固有の識別子である．管理者 ID は教職員のみ取得可能としている．管理者 ID は職員番号を基に，独自に作成した計算式を用いて一意の値になるように生成している．この変換は，運用上の必要性から可逆なものとしているが，推定を困難とするに十分な複雑さを持たせている．職員番号やネットワーク ID ではなく管理者 ID を別途設けるのは，後述する

機器 ID やソフトウェア ID は PC，ソフトウェアパッケージなどにラベルとして貼り付ける必要があり，第三者に見られた場合，パスワード総当たり攻撃などに悪用される危険性があるからである．

資産の中には部局などで管理しているものも存在するため，組織単位でも管理者 ID を取得できるようにしている．センターから配布しているソフトウェアも組織単位の管理者 ID で管理している．

(3) 機器 ID

機器 ID は，それぞれの管理者が管理している大学所有の PC 全てに対して割り当てる識別子である．機器 ID の決定は管理者 ID を使用し，「H_管理者 ID_001」のように割り当てる．

(4) ソフトウェア ID

ソフトウェア ID は機器 ID 同様，それぞれの管理者が管理している大学所有のソフトウェア全てに対して割り当てる識別子である．ソフトウェア ID の決定においても管理者 ID を使用し，「S_管理者 ID_001」のように割り当てる．例として，センターで管理しているソフトウェアは「S_センターの管理者 ID_001」というようになる．

2.2.2 システム概要

ソフトウェア配布システムの概念図を図 2-1 に示す．ソフトウェア配布システムは，アカウントサービスシステム，アカウント管理システム，LDAP サーバ，管理者 ID 発行システム，CAS サーバ，ソフトウェア配布サーバから構成される．なお，図中の矢印 A はネットワーク ID 登録，矢印 B は管理者 ID 発行，矢印 C はソフトウェアダウンロードの流れをそれぞれ示している．

以下で各サーバの概要について説明する．

2.2.2.1 アカウントサービスシステム

アカウントサービスシステムは，ユーザがネットワーク ID を登録するためのシステムである．Web ブラウザから本システムにアクセスして，必要な情報を入力することでネットワーク ID を取得することができる．管理者 ID を発行する際

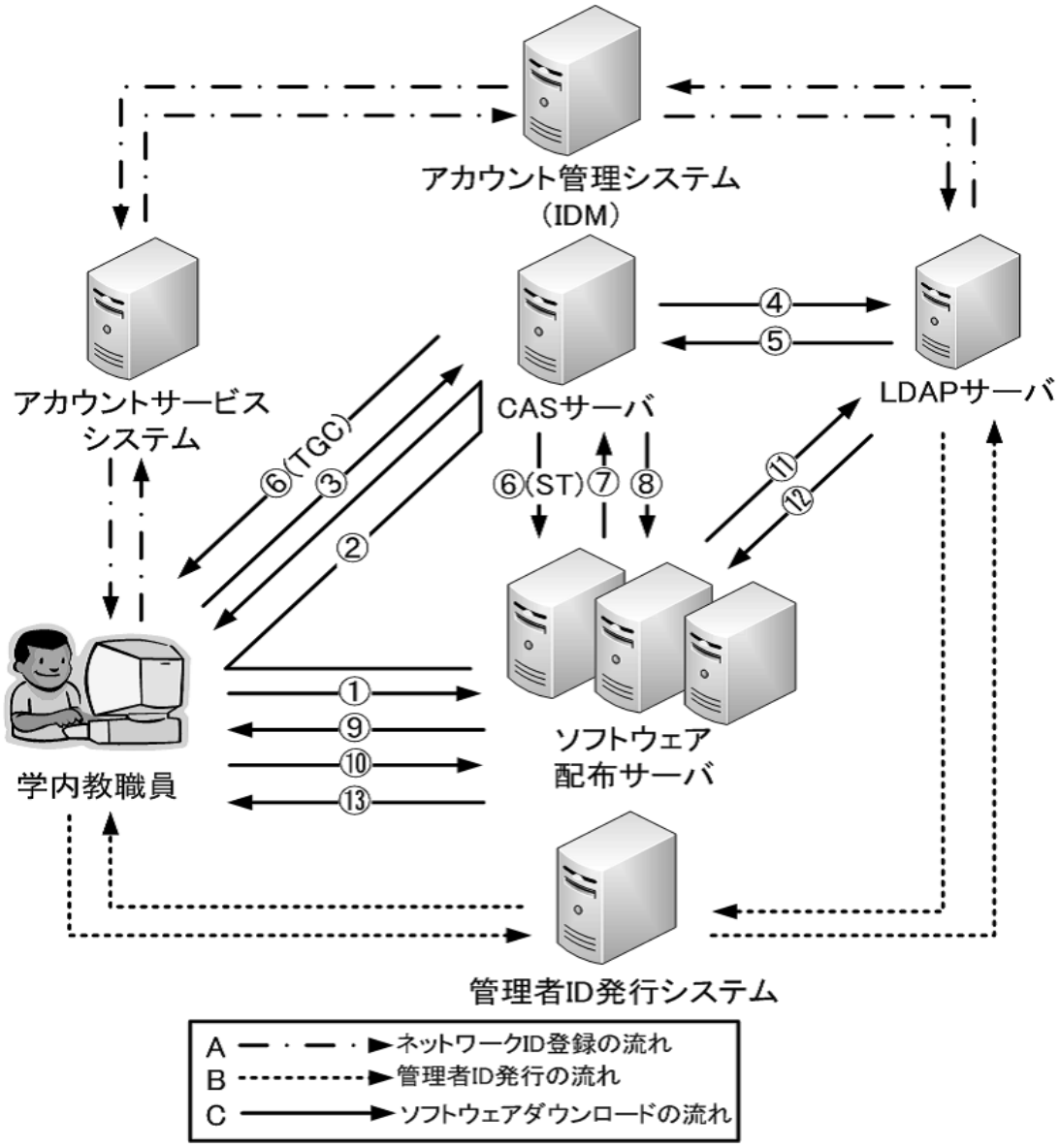

図 2-1 ソフトウェア配布システム概念図

には，ネットワーク ID とパスワードで認証し本人確認を行うため，管理者 ID 発行に先立ち，本システムにアクセスする必要がある．

2.2.2.2 アカウント管理システム

アカウント管理システムはアカウントサービスシステムで入力された情報を LDAP サーバなどのディレクトリサーバに反映させるシステムである．入力され

た情報は複数のディレクトリサーバに登録される必要があるため，本システムを用いて情報を同期させている．

2.2.2.3 LDAP サーバ

LDAP サーバは，アカウント管理システムから送られてきた情報を格納するディレクトリサーバである．LDAP サーバは，ネットワーク ID の他，職員番号や雇用形態などの情報も保持している．

2.2.2.4 管理者 ID 発行システム

管理者 ID 発行システムは, ユーザの資産管理における管理者 ID の発行を行う．ネットワーク ID とパスワードでユーザ認証を行い，認証に成功した場合に管理者 ID を発行する．なお，認証を行うための情報は LDAP サーバから取得している．

2.2.2.5 CAS サーバ

ソフトウェア配布サーバでは，適切なユーザであることを確認するためにネットワーク ID とパスワードを用いたユーザ認証を行う．しかし，Web アプリケーションごとに認証を実装していては，システム管理者にかかる負担が大きく，また，ユーザシステムごとに認証を行うことには煩雑さを感じるものと思われる．そこで，ソフトウェア配布サーバにおける認証に CAS を用いて，シングルサインオン環境を導入した．

2.2.2.6 ソフトウェア配布サーバ

ソフトウェア配布サーバは，ソフトウェアをユーザに対して配布するサーバであり，ユーザインターフェースもここが担当する．

ソフトウェア配布サーバの Web アプリケーションはすべて CAS クライアントを実装しているため，最初のアクセスではネットワーク ID とパスワードによる認証が必要であるが，セッションを切らない限りはそれ以上の認証手続きを行う

必要はない．認証が成功した場合，ダウンロード画面を表示し，ログインしたネットワーク ID に対応する機器 ID を入力させる．CAS クライアントは CAS サーバでの認証時に入力されたネットワーク ID の情報を参照し，ネットワーク ID をキーとして LDAP サーバに職員番号をバインドする．そして，それを基に該当ユーザの管理者 ID を生成し，機器 ID が正しいものであるかを判断し，正しい場合のみダウンロードを許可する．

2.2.3 動作手順

ユーザが，必要とするソフトウェアをダウンロードするまでの動作手順をフローチャートにしたものを図 2-2 に示す．

ソフトウェア取得手続きに先立ち，ユーザはアカウントサービスシステムにアクセスし，ネットワーク ID を登録する．登録した情報は，アカウント管理システムから LDAP サーバに反映される．次に，管理者 ID 発行システムにアクセスし, 登録したネットワーク ID とパスワードで認証を行い, 管理者 ID を取得する．ここまでの動作は 1 度行えば今後行う必要はない．

ネットワーク ID と管理者 ID を取得したユーザはソフトウェア配布サーバにアクセスする．既に CAS サーバで認証済みの場合は，ここでの認証手続きは不要である．CAS サーバで認証を行っていない場合は CAS サーバの認証画面にリダイレクトされる．そこで，ネットワーク ID とパスワードを入力し，認証手続きを行う．認証に成功すると，該当ソフトウェアのダウンロード画面を表示する．ユーザはインストールする PC の機器 ID を入力し，ダウンロードボタンをクリックする．ソフトウェア配布サーバはアクセスしているユーザのネットワーク ID と入力した機器 ID が整合しているかを確認する．確認は，LDAP サーバから抽出した職員番号から管理者 ID を生成し，機器 ID の管理者 ID 部分が食い違っていないかを判定することで行う．

利用記録は機器 ID をキーとして行う．アクセス元 IP アドレスで管理を行った場合，プライベートネットワーク配下の PC には対応できない問題があったが，機器 ID を用いることで，PC を一意に特定して記録することができる．

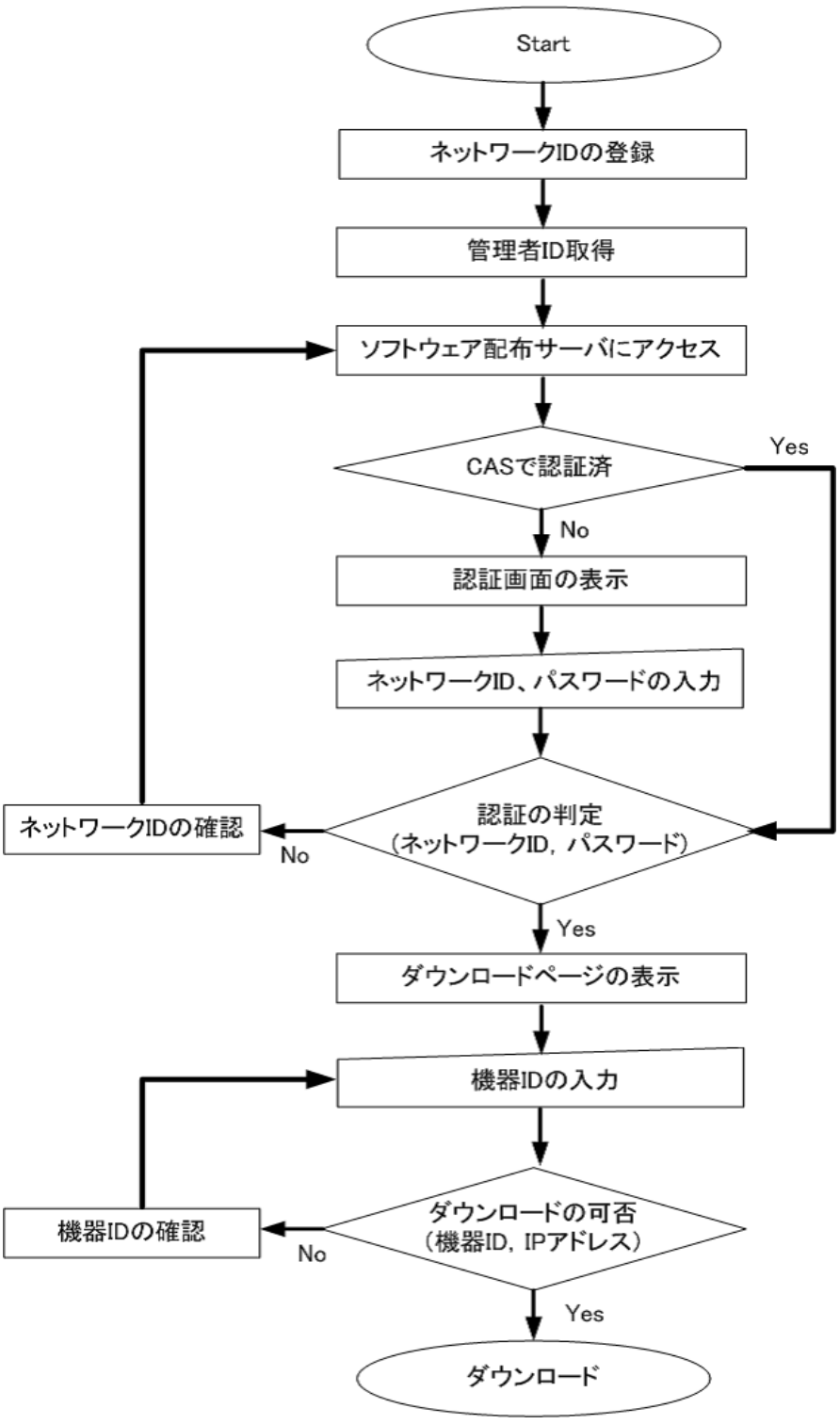
Start
ネットワークIDの登録
管理者ID取得
ソフトウェア配布サーバにアクセス
CASで認証済
Yes
No
認証画面の表示
ネットワークID、パスワードの入力
認証の判定
（ネットワークID，パスワード）
ネットワークIDの確認
No
Yes
ダウンロードページの表示
機器IDの入力
ダウンロードの可否
（機器ID，IPアドレス）
機器IDの確認
No
Yes
ダウンロード

図 2-2 動作手順

2.3 実装

2 章で説明した内容を基に，ソフトウェア配布システムの実装を行った．システム概念図（図 2-1），動作手順（図 2-2）の流れに合わせて説明する．

2.3.1 各サーバのスペック

実装を行った各サーバのスペックを示す.

[アカウントサービスシステムサーバ]
機種：PRIMEPOWER450
OS：Solaris10
CPU：SPARC64 V 1.98GHz×2
メモリ：6GB
ソフトウェア：Interstage Application Server Standard　-J Edition V8.0 [26]
[アカウント管理システムサーバ]
機種：PRIMEPOWER450
OS：Solaris10
CPU：SPARC64 V 1.98GHz×2
メモリ：6GB
ソフトウェア：Interstage Application Server Standard　-J Edition V8.0
[LDAP サーバ]
機種：Sun Fire T2000
OS：Solaris10
CPU：UltraSPARC T1 1.0GHz
メモリ：8GB
[管理者 ID 発行システムサーバ]
機種：PRIMERGY　RX200
OS：Red Hat Linux Enterprise4
CPU：Zeon 1.60GHz
メモリ：2GB
[CAS サーバ]

機種：JCS Type 1U-XEF

OS：CentOS5.0

CPU：Zeon 2.66GHz

メモリ：2GB

2.3.2　システム実装

次に実装部分をシステムの流れに沿って説明する．アカウント管理システムにおいては，アカウントサービスシステムで入力された情報を複数のディレクトリサーバで同期させる．アカウント管理システムとして，ディレクトリ統合管理システムである Sun Java System Identity Manager 6.0[27]（以下，IDM と記載）を使用している. アカウントサービスシステムで入力した情報を IDM のアカウントサービスシステム連携処理モジュールに送り，IDM エンジンから各ディレクトリサーバに登録する．なお，ソフトウェア配布システムでは LDAP サーバの情報を参照するため，LDAP サーバのシステム構成を説明する．LDAP のソフトウェアとして，Sun Java Directory Server 5[28]を使用している．LDAP サーバは重要な情報を保持しているため，図 2-3 に示すように 4 台で構成しており，LDAP マスタサーバを 2 台構成とし，データ書き込み機能を冗長化（マルチマスタ）している．

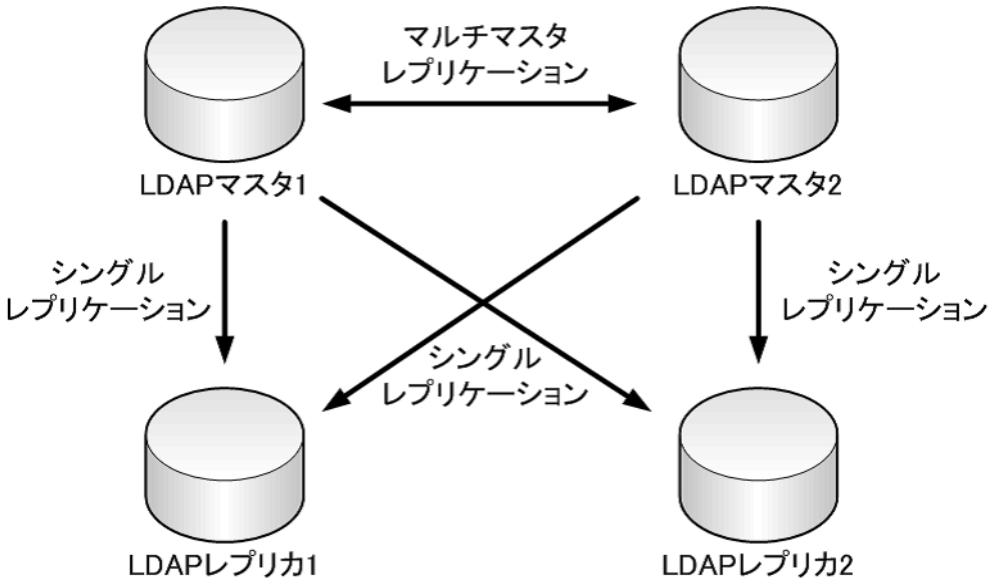

図 2-3 LDAP 構成図

管理者 ID 発行システムは Apache で Web サーバを構築し, PHP を使用して Web アプリケーションを作成している．管理者 ID 発行システムにアクセスすると図 2-4 の画面を表示し, ユーザにネットワーク ID とパスワードを入力させる. LDAP サーバから情報をバインドし, 正しい場合は図 2-5 に示す管理者 ID 取得画面を表示する.

ソフトウェア配布サーバにおける Web アプリケーションは, PHP, Perl, Apache, Asp のライブラリを利用して CAS クライアントを実装している．CAS サーバのソフトウェアはバージョン 3.2.1 を使用している．そして，CAS サーバが利用不可になった場合を想定し, 同一の設定の CAS サーバをコールドスタンバイさせている.

ユーザがソフトウェア配布サーバからダウンロードするまでの動作を図 2-1 の矢印 C に基づき説明する．説明文中の括弧内の数字は，図 2-1 に記載された矢印の番号と対応する.

ここでは
ソフトウェア配布サーバの URL は
https://app.oooo/dl.php
CAS サーバの URL は
https://cas.oooo/
であるものと仮定する.

最初にユーザは https://app.oooo/dl.php にアクセスする（①）．最初は認証を行っていないため, CAS クライアントは https://cas.oooo/login に通信をリダイレクトし, その際に service パラメータとして自身の URL である https://app.oooo/ dl.php を挿入し，CAS サーバに再転送先の URL を伝える．すなわち，ユーザの Web ブラウザは URL https://cas.oooo/login?service =https://app.oooo/dl.php を受け取り，図 2-6 に示す認証画面を表示する（②）．認証画面において，ユーザは自分のネットワーク ID とパスワードを入力すると（③），CAS サーバは外部認証サーバ（LDAP サーバ）で，ユーザから入力された情報が正しいか認証を行う（④，⑤）．認証に成功すると，CAS サーバはユーザの Web ブラウザに対して，Ticket Granting Cookie（TGC）と呼ばれる，ブラウザが認証済みかを判断するクッキーを配布し，service パラメータで指定した URL に対するリダイレクションを行う．URL には tick

et パラメータとして Service T icket（ST）と呼ばれる CAS クライアントにアクセスする際のワンタイムチケットが含まれる．つまり，https://app.oooo/?ticket=ST-xxxxxx の形になる（⑥）．ticket パラメータを受理したソフトウェア配布サーバは ST を CAS サーバに送付する（⑦）．CAS サーバは Validation サーブレットで ST に問題が無いことを確認し，認証を行ったユーザの情報をソフトウェア配布サーバに送る（⑧）．ソフトウェア配布サーバはユーザの Web ブラウザに図 2-7 に示すようなダウンロード画面を表示させる（⑨）．

ダウンロード画面においてユーザが連絡先 E-mail アドレスと，ダウンロードするソフトウェアをインストールする PC の機器 ID を入力すると（⑩），ソフトウェア配布サーバは，参照可能なネットワーク ID を基に，LDAP サーバから職員番号をバインドする．そして該当ユーザの管理者 ID を生成し，機器 ID として使用している管理者 ID が正しいものであるかを検証し（⑪，⑫）ダウンロードを許可する（⑬）．以上の仕組みにより，ユーザ及び機器 ID の正当性が検証でき，配布したソフトウェアを誰がどの PC にインストールしたかを把握することができる．

図 2-4 管理者 ID 発行システムログイン画面

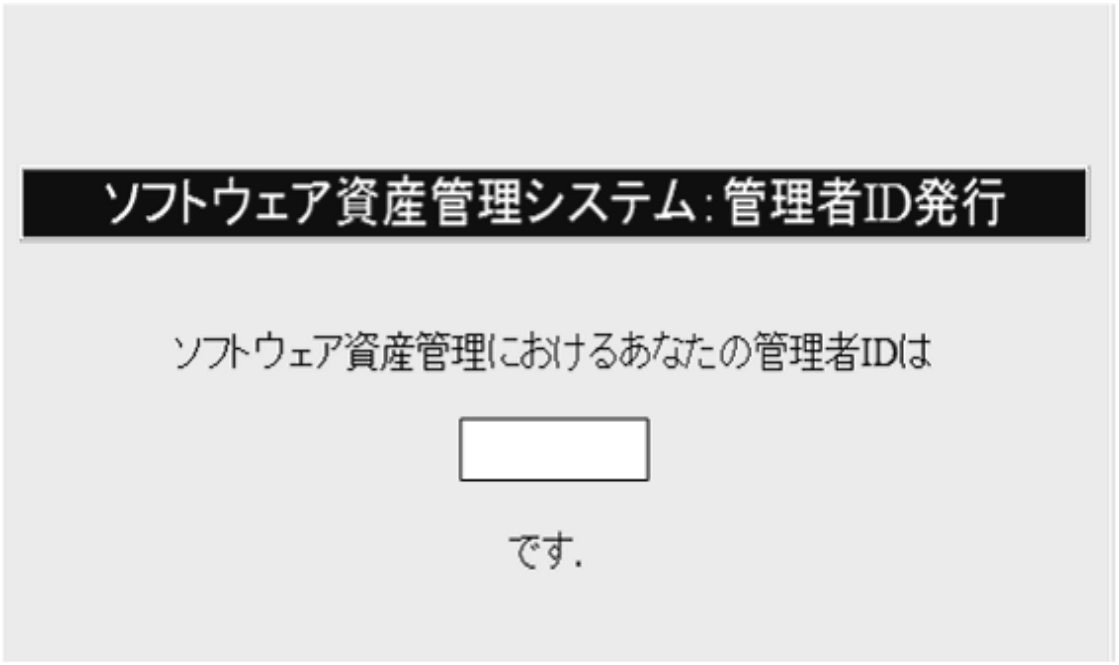

図 2-5 管理者 ID 発行システム ID 発行画面

図 2-6 CAS 認証画面

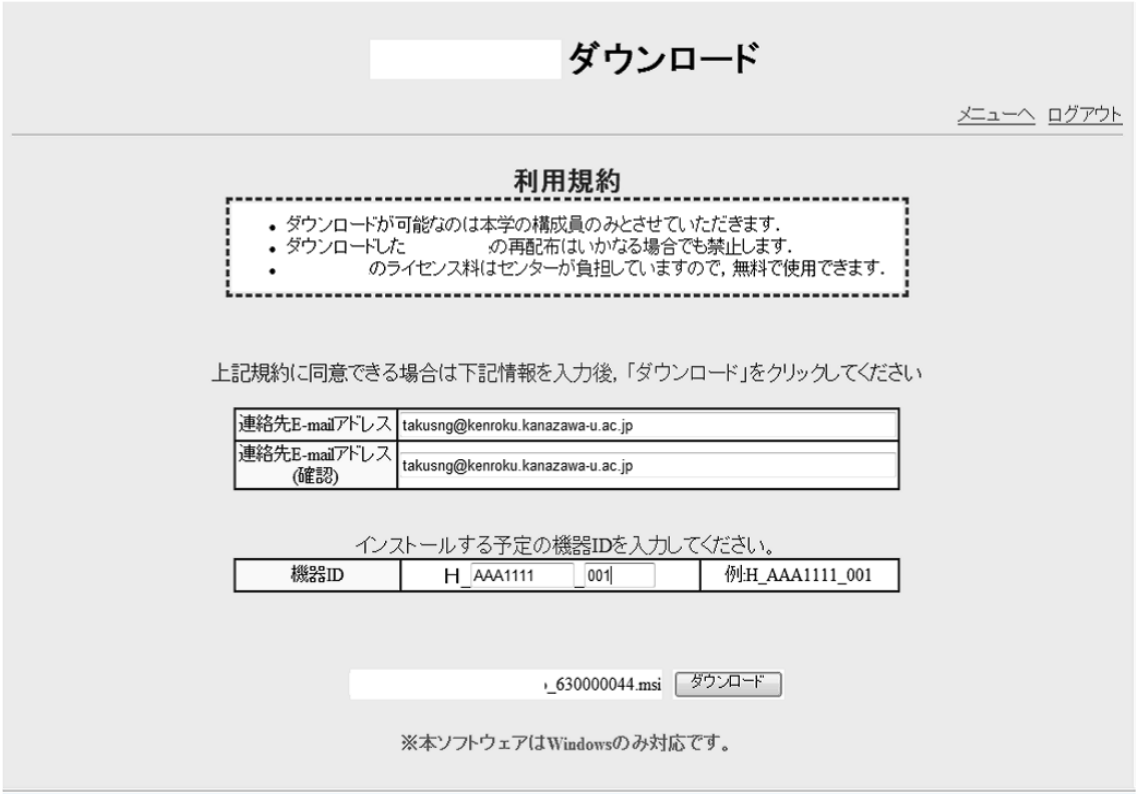

図 2-7 ソフトウェアダウンロード画面

2.4 考察

2.4.1 CAS の問題点

CAS を実際の本学内情報システムに適用し，運用を行ったが，その際に以下の問題点が明らかになった.

(1) CAS サーバへのアクセス制限に対応していない

CAS サーバを利用できる CAS クライアントに制限がなく，サーバに CAS クライアントのモジュールを適用し, 認証を受ける CAS サーバをクライアント側で一方的に指定するだけで, どのサーバでも CAS クライアントとして動作が可能である.

(2) クライアントに応じて必要な属性を渡すことができない

CAS サーバは，CAS クライアントに応じて適当な属性を返すことができず，CAS クライアントが属性を要求すると，CAS サーバはどの CAS クライアントにも全ての属性を送信する．

(3) Form 入力のメソッドにおいては GET にしか対応していない

CAS の認証で, ユーザの識別に使われる ST パラメータは常に URL の引数として受け渡しが行われる．そのため，(1)で述べたように，CAS サーバは無制限にアクセスを受け付けるため，悪意あるユーザが ST をランダムに発行し続け，他のユーザの ST と一致した場合はそのユーザ情報が搾取される危険性がある．

(4) クロスドメインシングルサインオンに未対応

CAS はクロスドメインシングルサインオンに未対応である．クロスドメインシングルサインオンとは，異なるドメイン間においてシングルサインオンを行うことを指す．具体的には，金沢大学と NII のサーバ間に対してシングルサインオンができるかどうかということである．

2.4.2 Shibboleth 利用の利点

2.4.1 節で CAS の問題点について述べた．これらの問題のため，CAS の仕組みを利用するのは困難と判断し, NII が大学間連携で推進している Shibboleth の利用を検討した．CAS の問題点における Shibboleth の動作は以下の通りである．

(1) 認証サーバへのアクセス制限が可能

Shibboleth では，フェデレーションを組み，フェデレーション内のサーバ間でのみアクセスが可能である．フェデレーションとは，あるポリシ（規定）のもとで相互に信頼し認証情報を交換することに合意した組織（サービス）の集合を示す．つまり，お互いに信頼（トラストサークル）を形成し，その閉じた範囲で通信を行う．フェデレーションはメタデータにより形成される．

(2) 認証クライアントに応じた属性の受け渡しが可能

Shibboleth では，認証クライアント（SP）がユーザ属性情報を要求した場合，認証サーバ（IdP）側で，認証クライアントに応じて情報を送信することが可能である．そのため，認証クライアントに不必要な情報を送信することを避け，セキュリティを向上することができる．

(3) ユーザの識別は Cookie で行う

ユーザが認証済みであるかどうかの判断は Cookie で行うため，URL パラメータにセッション情報が出ることはない．

(4) クロスドメインシングルサインオンに対応

Shibboleth はクロスドメインシングルサインオンに対応している．Shibboleth では，認証，認可に SAML という規格を採用しており，異なるドメイン間においても認証，認可を行うことが可能である．そのため，組織間連携の実現においても柔軟に対応することが可能である．

Shibboleth は CAS における問題をクリアしており，異なるドメインにまたがるシングルサインオン環境の実現のために，次章以降では Shibboleth を採用したシステムについて検討することとした．

2.5 まとめ

本章では，CAS によるシングルサインオン環境を本学に実際に適用し，その動作を検証した．そして CAS の問題点を考察するとともに，Shibboleth がこれらの問題点を解決可能であることを確認した. 次章以降, 統合認証基盤構築に際して, システムの開発を 3 つの範囲に分けて行う. 図 2-8 にシステム開発範囲図を示す. 図 2-8 では，A 大学と B 大学を例として挙げている．A 大学のユーザ A は，大学内統合認証基盤において，A 大学内の情報システムへのアクセスを行う．また，大学間連携として，A 大学の所属で B 大学の情報システムへアクセスを行う．さらに，研究室間における連携として，A 大学の A 研究室の所属で，B 大学の B 研

究室で管理している情報システムへのアクセスを行う.

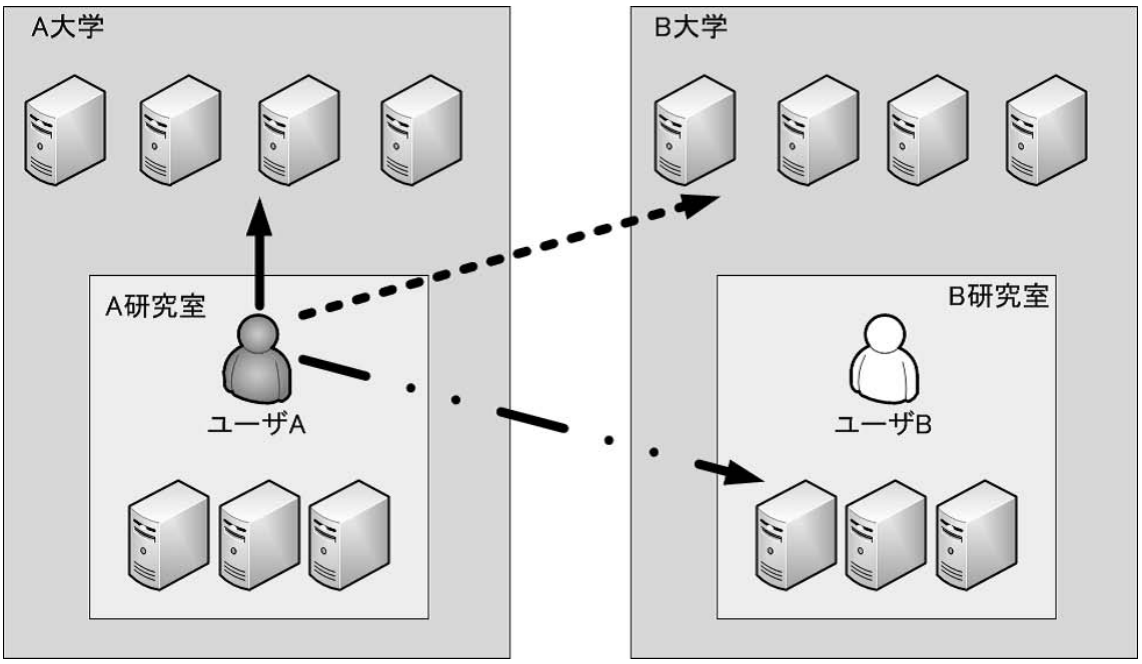

図 2-8　システム開発範囲

第3章　GakuNin 認証連携基盤に基づく大学間における安全なデータ共有システム構築

3.1 はじめに

本章では，Shibboleth を利用して本学統合認証基盤を構築する上での大学間連携部分について述べる．大学間連携について最初に議論する理由として，NII が平成 20 年度に, 認証連携基盤実現のために技術的及び制度的な検証を行うことを目的とした,「シングルサインオン実証実験（以降，実証実験と記載)」を実施したことがある．本研究ではそれに参加することで，Shibboleth の知見を得るとともに，GakuNin を利用して，大学間における安全なデータ共有システムの開発を行った．

NII と全国共同利用情報基盤センターが主導となり，平成 18 年度より，全国大学共同電子認証基盤構築事業を 3 年計画で行われてきた．本事業では，大学が共有する教育研究用計算機，電子コンテンツ，ネットワークなどのリソースを大学間において安全・安心に有効活用することを目的とし，本事業における認証基盤は GakuNin と呼称される．

本学では，全学におけるポータルシステムや統合認証基盤を立ち上げるにあたり，単に大学内に閉じた環境ではなく，将来的な大学間連携にも活用できることを視野に入れるために，本実証実験に積極的に参加した．その結果，GakuNin の性質を利用した非常に有効なシステムを構築することができた．さらに，構築したシステムを実運用していくにあたり，考察すべきことについてもいくつか検証を行った．

本章では，まず GakuNin の概要について述べ，実際に構築したシステムについて述べた後，考察を行う．

3.2 GakuNin 概要

GakuNin とは，学術情報システムを利用する大学，学術情報システムを提供する機関・出版社などから構成されたフェデレーションを指す．各機関はフェデレーションが定めた規程（ポリシ）を信頼しあうことで，相互に認証連携を実現することが可能となる．認証連携が実現できれば，学内でのシングルサインオンの実現が可能になるとともに，他大学や商用のサービスにおいても，ID・パスワードの再入力を行わずに利用できる環境を実現できる．具体例として，他大学の無線 LAN を，いつも大学で使用している ID とパスワードで利用でき，かつ自大学が契約している電子ジャーナルへシームレスにアクセスすることも可能となる．

図 3-1 に GakuNin 概念図を示す．実証実験において，IdP 及び SP の構築は各参加大学が担当し，DS の構築は NII が担当した．そして，それらを用いて大学間で連携が取れるか実験を行った．

本学は，実証実験において積極的に参加し，IdP の構築および SP の提供を行った．構築したシステムの詳細については次章以降に述べる．そして，GakuNin は平成 21 年度から試行運用を実施し，システム運用基準を定めたのち，平成 22 年度より本格運用を開始した．現在，システム運用基準をクリアした IdP および SP は運用フェデレーションへ，未達成のものはテストフェデレーションへと分けて運用を行っている．本学で構築したシステムは，他大学に先駆けて運用フェデレーションに適用されるとともに，GakuNin のタスクフォースメンバーとして NII と GakuNin の普及へ向けた活動を行っている．

3.3 システム実装

3.3.1 構築したシステムの概要

GakuNin 認証連携基盤を利用する最大のメリットは，他大学からの利用者について身元が保証されることである．本研究では，実証実験において，この性質を利用し，これまで当センターにおいて管理運用の実績がある 2 つの情報システムに対して，GakuNin による大学間連携を導入する方法について検討した．具体的には，「GakuNin を用いたファイル送信サービス[29]」と「DSpace[30]によるデジタルコンテンツ公開サービス[31]」の 2 つの SP の構築を行った．

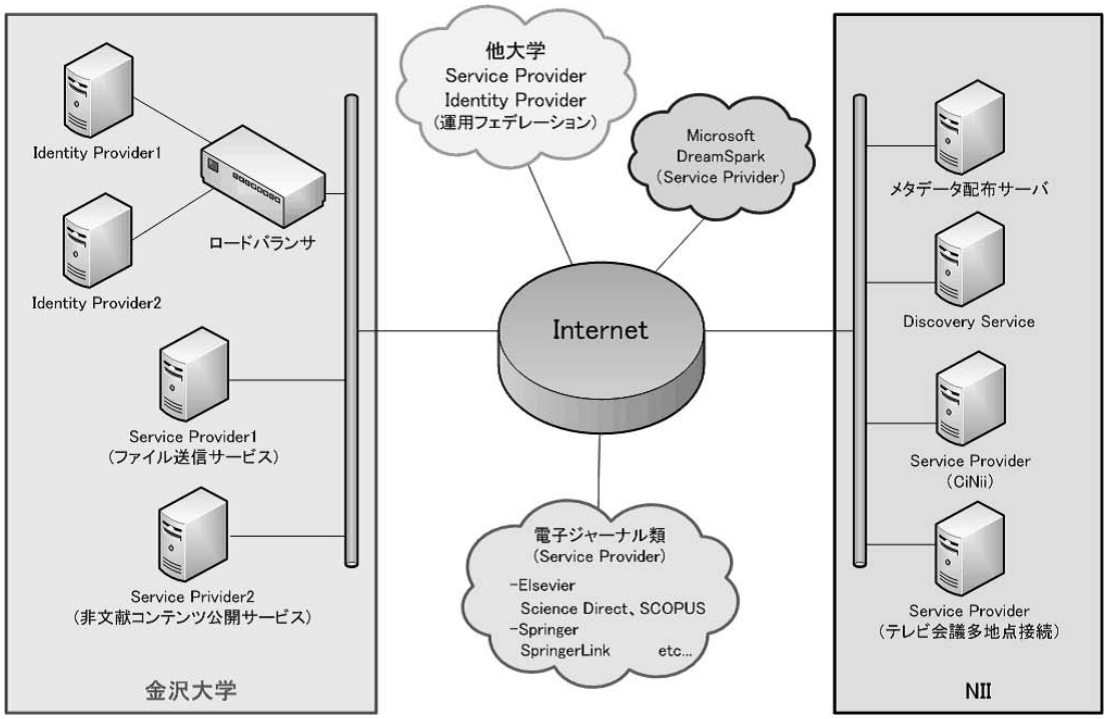

図 3-1 GakuNin 概念図

まず，「GakuNin を用いたファイル送信サービス」は，メールでは添付できない大容量のファイルを相手に送信したい場合に，ファイルの転送を行う SP である．業務や研究など様々な場面において，メールでは添付できない大容量のファイルをやり取りしたいケースは非常に多く，このようなサービスのニーズは非常に高い．事実，センターでは 2005 年から本学構成員を対象にファイル送信サービスを提供しており，毎月 700 件程度の利用がある．しかしながら，このようなサービスを行ううえで問題となるのは利用者の管理である．本学構成員だけに提供する場合は問題にはならないが，他大学の構成員に対してデータを送信する場合や逆に受信する場合，あるいは他大学の構成員同士で利用する場合は，悪意ある第三者に不正利用されるのを防ぐため，他大学の構成員の本人性を確認する必要がある．しかし，現状ではこの問題を解決する根本的な手立てが存在しなかった．そこで，GakuNin を利用して認証に成功したユーザだけ利用できるように設計することで，本人性を確認したうえでサービスを提供することができるようになると考え，本 SP の構築を行った．

一方，「DSpace によるデジタルコンテンツ公開サービス」は大学で生産された

様々な実験データなどを共有したり，公開したりするサービスである．DSpace は，オープンソースのリポジトリ構築ソフトウェアである．本 SP では，学術論文，紀要，研究報告書などの書誌系の情報ではなく，自然科学系実験データなどの画像や動画などの実験観測データをリポジトリ化し公開することを目的としている．実験観測データなどは貴重なデータであるため，原則として誰にでも公開する書誌系の情報とは異なり，特定の組織やグループ限定で見せたいケースが多い．そこで，本実証実験においては GakuNin で認証が成功したユーザだけにデータを公開するように SP の構築を行った．

3.3.2 各サーバのスペック

実証実験において，本研究では IdP を 2 つ，SP を 2 つ構築した．構築した IdP 及び SP のスペックは以下のとおりである．

[IdP 用サーバ 1]
Dell PowerEdge T300
CPU：Intel Core2Duo E6305
メモリ：4GB
HDD：500GB(RAID1)
OS：CentOS5.5(64-Bit)

[IdP 用サーバ 2]
Epson Endeavor NP11-V
CPU：Intel Atom230(1.6GHz)
メモリ：1GB
HDD：160GB
OS：CentOS5.4(64-Bit)

[SP 用サーバ 1（GakuNin を用いたファイル送信サービス）]
Dell PowerEdge T300
CPU：Intel Core2Duo E6305
メモリ：4GB

HDD：500GB(RAID1)
OS：CentOS5.4(64-Bit)
アプリケーション：apache2.2.3，php5.1.6，PostgreSQL8.1.11

[SP 用サーバ 2（DSpace によるデジタルコンテンツ公開サービス）]
CPU：Intel Core2Duo E8400
メモリ：2GB
HDD：250GB
OS：CentOS11.1(64-Bit)
アプリケーション：DSpace1.4.2

3.3.3 GakuNin を用いたファイル送信サービス

まず,「GakuNin を用いたファイル送信サービス」について説明する. 本 SP は, ユーザ間で安全に大容量のデータを共有することを目的としたサービスである. GakuNin を利用したファイル送信サービスの動作概念図を図 3-2 に示す．説明文中の括弧内の数字は，図に記載された矢印の番号と対応する.
送信者は A 大学所属，受信者は B 大学所属とする．最初に送信者は，金沢大学の SP であるファイル送信サービスにアクセスする (①). ファイル送信サービスは, 送信者に認証を促すため，NII の Discovery Service にリダイレクトし，送信者に IdP を選択させる（②）．送信者は A 大学の IdP を選択し，認証を行う（③，④，⑤）．A 大学の IdP は送信者の情報をファイル送信サービスに送信する（⑥）．ファイル送信サービスはユーザの確認を行い，ファイル送信サービスの画面を表示する（⑦）．送信者は自分の情報の入力及び，受信者の情報を入力する（⑧）．受信者は図 3-3 に示すメールを受け取り（⑨），メールに記載されている URL にアクセスする（⑩）．受信者も送信者と同様に DS にリダイレクトされ（⑪），受信者は B 大学の IdP を選択し，認証を行う（⑫，⑬，⑭）．B 大学の IdP は受信者の情報をファイル送信サービスに送信する（⑮）．認証後，受信者は図 3-4 の画面が表示され，ファイルをダウンロードすることができる（⑯）.

このように，GakuNin 認証基盤を用いることにより，ユーザ間において安全に大容量のデータを共有するサービスを構築することができる.

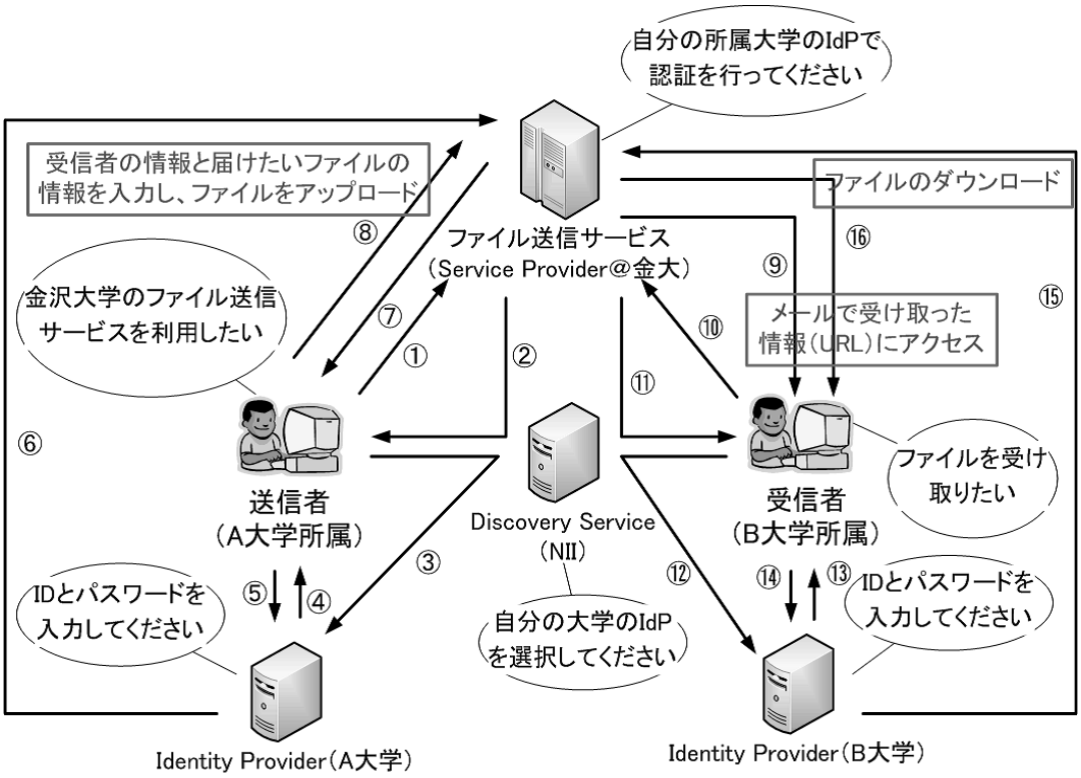

図 3-2 GakuNin を利用したファイル送信サービス概念図

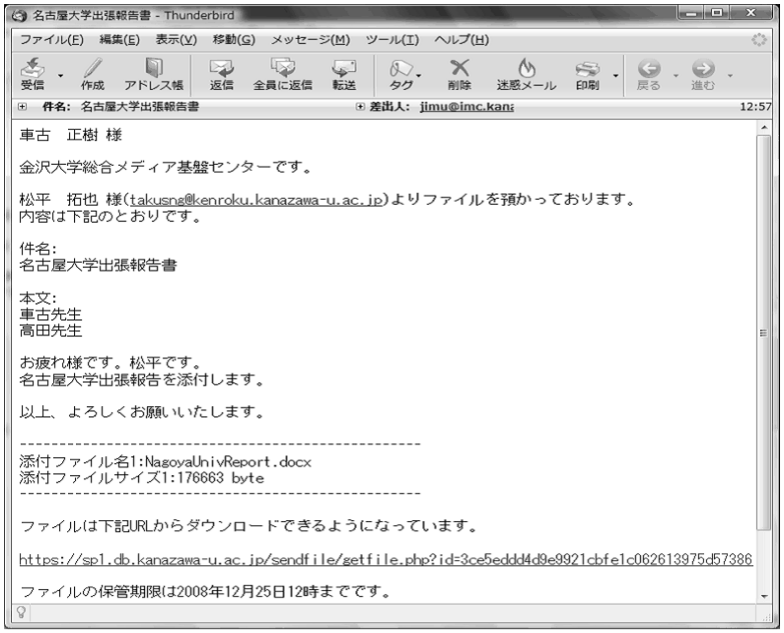

図 3-3 受信者通知メール

図 3-4 ファイル取得画面

3.3.4 Dspace によるデジタルコンテンツ公開サービス

次に「DSpace によるデジタルコンテンツ公開サービス」について説明する．本 SP は, 大学で生産された実験観測データなどを特定の組織やグループに限定して公開を行うことを目的としたサービスである．今回の実証実験では，科学観測衛星「あけぼの」による地球周辺の電波観測データのスペクトル画像を PNG 化したものを，GakuNin で認証が成功したユーザだけにデータを公開するように構築を行った.

図 3-5 に DSpace によるデジタルコンテンツ公開サービスの画面を示す．GakuNin を用いたファイル送信サービス同様，本 SP にアクセスに来ると，自組織内の IdP での認証を求められる．そして認証に成功したのち，データの閲覧を行うことができる．サムネイルの一覧を表示でき，汎用的なフォーマットに変更したデータを表示する.

このように GakuNin 認証連携基盤を利用することにより，実験観測データなどを，特定の組織やグループ限定で見せるサービスを構築することができる.

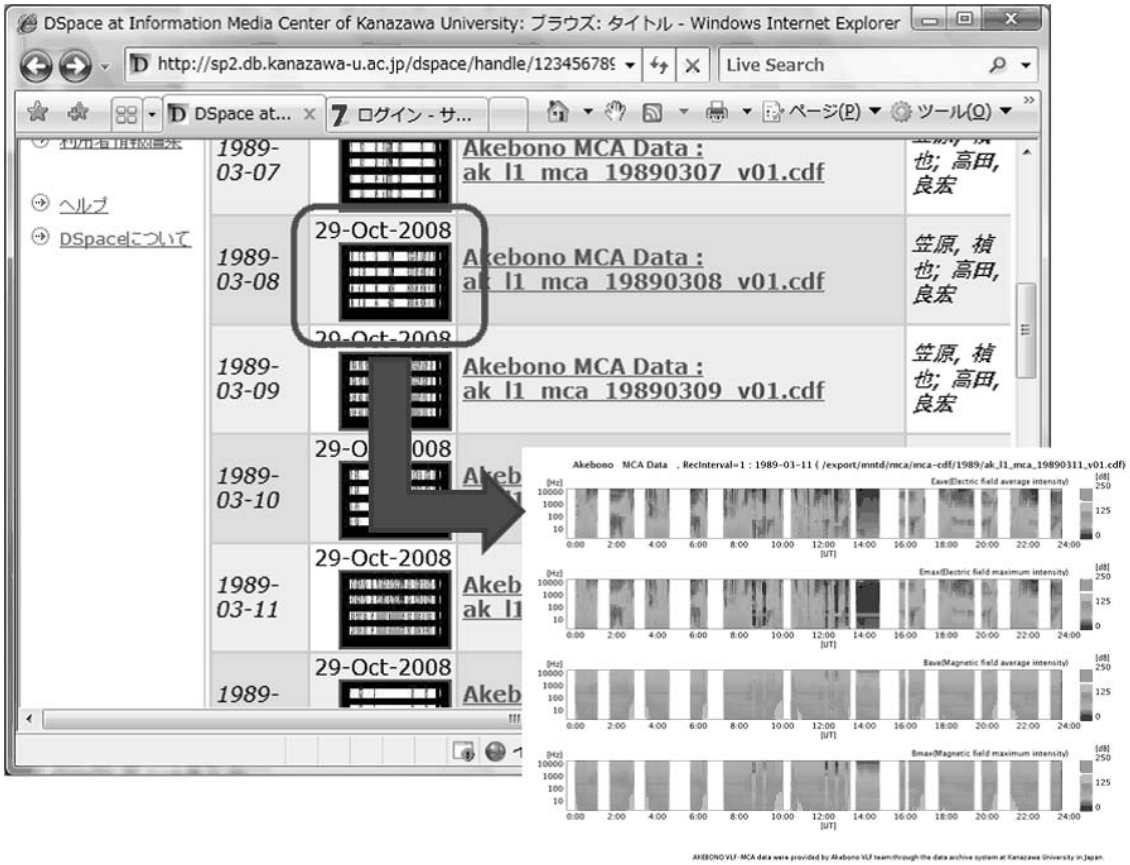

図 3-5 非文献コンテンツ公開サービス

3.4 考察

本節では，今回の実証実験で構築したシステム及び今後の取り組み方について考察する．

3.4.1 SP における認可設定

以下において，今回の実証実験で構築したシステムについて考察する．今回構築したシステムにおいてはGakuNinでの認証が成功したユーザすべてに対してサービスが利用できる設定にしている．GakuNin という信頼されたフェデレーションで認証されたユーザということが特定できている限り，サービスを悪用される可能性は低い．しかし，今後はセキュリティを考慮し，構築したそれぞれの SP において，特定のユーザや組織などに限定して利用を許可したいといったケース

が存在することが十分考えられる．

まず，ファイル送信サービスにおいては，学生に閲覧されることが好ましくない大学運営や業務に関わる情報ファイルの交換の用途で利用するときは，大学構成員の中でも教職員だけにアクセスを制限したいケースが存在する．その解決策として，Shibboleth では Apache のアクセス制御ファイルである.htaccess で認可を行うという方法がある．図 3-6 にその設定例を示す．この例では，この.htaccess が配置されたディレクトリについては教員と職員だけがアクセス可能という意味を持つ．そのため，たとえ学生が GakuNin フェデレーションに属している組織に在籍していたとしても，サービスへのアクセスを制限することができる．このように設定することで，職種区分に応じた特定のユーザだけにサービスを提供することができるようになる．

さらに，DSpace によるデジタルコンテンツ公開サービスでは，職種区分だけではなく，組織情報や，所属情報などさらに詳細な公開範囲を指定する必要がある．その場合，Apache による認可設定では不十分と考えられる．そのため，このようなケースにおける解決策として，Shibboleth の XML ファイルを用いて認可を行う方法がある．図 3-7 に設定例を示す．この例では，secure ディレクトリでは，A 大学の工学部または理学部に所属する教職員に対してアクセスを許可するが，それに所属する学生のアクセスに関しては許可しないという設定になる．

このように，XML を用いる方法では，AND，OR，NOT の論理式を用いて，より柔軟に認可設定を行うことができるため，DSpace によるデジタルコンテンツ公開サービスでの複雑な認可を実現できると考えられる．

3.4.2　全学的 GakuNin 対応への取り組み

つづいて，今後の取り組み方について考察する．GakuNin によって大学間連携を実現するためには，各大学において GakuNin を利用できる環境構築が必須である．本学内においても，全構成員が GakuNin を利用できる環境を整備する必要がある．つまり，本学の全構成員に GakuNin で利用できる ID を提供する必要がある．

本学においては，全学構成員を対象に交付される ID として，「ネットワーク ID」と呼ばれるものが既に存在する．本研究においては，これを用いた認証・認可を全学用 LDAP サーバで構成することを検討した．ネットワーク ID は任意で取得

する ID であるが，本学構成員がセンター提供のサービスを利用する際に使用する ID で，現在，センターが提供するサービスとして，学内ネットワーク利用認証や VPN 認証などに用いている．そのため，ネットワーク ID はほぼ全構成員が既に取得済みであり，GakuNin を用いた大学間連携においても，このネットワーク ID を用いるのが最も有効と考えられる．但し，現在運用されているネットワーク ID を GakuNin に使用する際には下記の 2 つの問題点を解決する必要がある．

1 つ目は，既存の LDAP サーバを使用する場合，その属性情報に GakuNin で必要とされるものが，必ずしもすべて存在するとは限らない点である．GakuNin において，SP に送信する ID は，eduPerson スキーマ[32]の eduPersonPrincipalName 属性を用いる．しかし，eduPerson スキーマは標準的なものではないため，本学の LDAP にも設定されていない．

2 つ目は ID の外部への漏えいの問題である．GakuNin で使用する ID は組織外の SP に送信されるため，アクセス先の SP が不正アクセスされた場合にネットワーク ID の漏えいにつながる危険性がある．3.4.1 節では SP 側である程度ユーザの情報を受け取ったうえで認可を行うことを検討したが，個人情報である ID については情報を保護する方法を検討する必要がある．

これらの問題の解決策として，IdP において eduPerson へのマッピングを，ID そのものでなく対応付けられた異なる文字列にする方法がある（図 3-8）．Shibboleth では，設定ファイル内で各属性値の定義を行う．その際に，ID の文字列を eduPersonPrincipalName にマッピングして送信することができる．そのことで，既存の LDAP に新たにスキーマを追加する必要がなくなる．そして，設定ファイル内で ECMAScript を使用して，値の変換を行うことができる．図 3-8 ではネットワーク ID である uid を md5 に変換したものを 16 進数にし，@kanazawa-u.ac.jp を付加している．そうすることで，変換後の値からネットワーク ID を割り出すのは難しく，この値が万一外部に漏えいしても問題ないと考えることができる．なお，IdP 側でこれらの設定を行っても，SP 側では eduPersonPrincipalName の値が送られてくるだけであるため，特に変更は不要である．

```
AuthType shibboleth
ShibRequireSession On
requireAffiliation faculty staff
```

図 3-6 .htaccess による認可設定例

```
<Host name="sp2.db.kanazawa-u.ac.jp" authType="shibboleth" requireSession="true">
    <Path name="secure">
        <AccessControl>
            <AND>
                <Rule require="o">Auniversity</Rule>
                <OR>
                    <Rule require="ou">Engineering</Rule>
                    <Rule require="ou">Science</Rule>
                </OR>
                <NOT>
                    <Rule require="Affiliation">student</Rule>
                </NOT>
            </AND>
        </AccessControl>
    </Path>
</Host>
```

図 3-7 XML による認可設定例

```
<resolver:AttributeDefinition id="principalName" xsi:type="Script"
    xmlns="urn:mace:shibboleth:2.0:resolver:ad"sourceAttributeID="uid">

    <Script>
      <![CDATA[
        importPackage(・・・・);
        uniqueValue = uid.getValues().get(0) + "xxxxx";
        localpart = DigestUtils.md5Hex(uniqueValue);
        principalName = new BasicAttribute("principalName");
        principalName.getValues().add(localpart + "@kanazawa-u.ac.jp");
      ]]>
    </Script>
</resolver:AttributeDefinition>
```

図 3-8 IdP 設定ファイル（一部）

3.5 まとめ

本学では，実証実験において独自の IdP 及び SP の構築を行った．GakuNin を使用することにより，他大学からの利用者について身元が保証され，大学間においてサービスを安全に提供することができるようなった．そして，その性質を利用して，これまでは学内のみで提供していた「ファイル送信サービス」，「デジタルコンテンツ公開サービス」を，大学間においても安全にサービスを提供できるシステムを構築することができた．さらに，今回構築した SP における認可の方法及び GakuNin で利用する ID の秘匿性について考察を行い，個人情報を保護しながらも，適切に認可を行う方法を示すことができ，セキュリティ面においても検証することができた．

第4章　大学における Shibboleth を利用した統合認証基盤の構築

4.1 はじめに

3 章で GakuNin における大学間連携において Shibboleth が有効であることを検証した．そこで，本章ではその知見を用いて，本学統合認証基盤の構築について議論を行う．

1 章で述べたとおり，多くの大学では，教育・研究・業務など様々な分野において情報システム化を進めてきたが, 構築した情報システムの大多数が, 各部署・部局などで独自に構築・運用されてきた．そのため，これらの情報システムは，それぞれ独自の認証機構を備え，ユーザに対して個別の ID とパスワードが発行され，大学のいたる所で情報システムが乱立し，たとえ個々の情報システムとしては機能しても，それらの公開方法や利用範囲に一貫性がなく，システム間の連携も考慮されていないのが実情であった．そのため，ユーザは，目的の情報システムの URL を自身で管理したり，情報システムごとの ID とパスワードの対を覚えておく必要があったりと，作業効率の低下のみならず，ID とパスワードの対をメモに残すなど，セキュリティの観点でも問題があった．また，情報システムの運用においても，情報システムごとに独自の認証機構を持つことは，人員の異動にともなう内部データ変更作業や認証機構の維持管理など，コストの増大を招いてきた．

このような背景のもと，本研究では，全学的な視野で情報化を戦略的に整備・推進することを目的として，全学の情報システムを一元的に利用する窓口であるポータルサイトおよびユーザ認証を一元的に行う統合認証システムを構築した．全学におけるイニシアティブをとるために，全学的な情報基盤整備をミッション

とする金沢大学情報戦略本部[33]の傘下で, 平成 21 年度より“全学ポータル WG”（平成 22 年度より“統合認証・ポータル整備 WG”に改称）という活動形態をとることでこの取り組みを実施した.

本研究においては，各所に分散する情報システムの一元化・融合化に必要な要素技術として，シングルサインオンおよび属性共有を掲げている．しかし，大学で構築されている情報システムは，教育・研究・業務など，その目的が多岐にわたることから，当然，それぞれ利用できるユーザの範囲が異なり，また，システムによって運用ポリシも異なる．そのため，大学特有の複雑な所属形態にすべて適合するシステムを構築するのは容易ではない．また，これまでは，ユーザを人ではなく現在有効な職分などの属性で判断していた．これに対し，本研究では，ユーザが複数の属性を持つ場合でも 1 つの ID とパスワードのみで認証・認可の制御ができる機構の実現を目指した.

一方，3 章で述べたように，現在，NII が中心となり，複数の大学間での情報システムの共有，相互乗り入れの実現を目指して，GakuNin プロジェクトが進められている．同プロジェクトの試みにより，大学間連携におけるシングルサインオンの環境は整備されつつある．また，佐賀大学では本プロジェクトとは別に，Opengate[34]と呼ばれる佐賀大学内の認証システムにおいて，Shibboleth を用いてシングルサインオン環境を実現している．しかし，大学間認証連携および佐賀大学においてはシングルサインオンに関する議論が中心となっており，ユーザの属性に応じた細かな制御についてはこれからである.

本研究では，GakuNin プロジェクトに積極的に参加し，蓄積した Shibboleth に関するノウハウを用いるとともに，先行事例や大学間連携などの現況を踏まえ，Shibboleth を用いてシングルサインオンによるユーザの認証を行うと同時に，教員・職員・学生など各自の職分も識別でき，その職分に応じて利用許可されている情報システムが，再度の認証動作なしに利用可能になる統合認証システムの構築を行った.

本章では，4.2 節で本学の情報システム一元化の指針について説明し，4.3 節で Shibboleth の問題点について議論する. 続いて, 4.4 節でその解決策について述べ, 4.5 節で実装について述べる．4.6 節で構築したシステムの評価を行い，最後に成果と今後の計画について考察する.

4.2 情報システム一元化の指針

本節では本学における情報システム一元化に向けた統合認証基盤構築の指針について述べる．特に「金沢大学 ID」，「ロール」，「アカンサスポータル」，「シングルサインオン対象システム」について以降順番に説明を行う．

4.2.1 金沢大学 ID

4.1 節で述べたとおり，本学では従来，情報サービスの整備は部局別・目的別に独立して行われてきたため，各情報システムが独自に ID を発行し，その認証方式も様々であった．そのため，認証システムの統一，すなわちシングルサインオンを実現するためには ID の集約を行う必要がある．

まず，学内の様々な各種システムについて，その利便性と管理効率向上を図るために，“金沢大学 ID”を導入した．金沢大学 ID は，常勤教職員・非常勤教職員，学生・研究生などを問わず，本学に関わる全構成員に対して 1 人に 1 つずつ付与する ID である．金沢大学 ID の採番方法については，金沢大学 ID の検討段階で既に全学規模で運用を行っていた名古屋大学の名古屋大学 ID[35]の方式を採用した．金沢大学 ID の基本的な付け方として，ランダムに与えたアルファベット 3 桁と数字 5 桁の 8 桁とし，容易に推測できないようにしている．また，転学類にともなう学籍番号変更や，卒業後に本学に就職した場合などの属性の変更によらず，同一 ID を使用できる．そして，生涯 ID として，卒業・退職後も ID を抹消されることなく，同一 ID で同窓会向けサービスなど，本学 OB としての情報サービスを受け続けることができる．

教職員番号や学籍番号のように，1 度付与されたら変更ができないものを ID とせず，金沢大学 ID に置き換えることにより，1 ユーザ 1ID を実現できるように設計を行った．金沢大学 ID は 2 年前から，後述の教育用ポータルシステム（アカンサスポータル）で限定的に利用されていたが，全学的な情報システムの一元化および統合認証に用いる“生涯 ID”として同 ID の本格導入を決定した．なお，金沢大学 ID は 2010 年 5 月 7 日現在，44158 件発行を行っている．

4.2.2　ロール

大学には多種多様の情報システムが存在し，それらを使用できるユーザは，システムにより様々である．たとえば，給与明細オンラインシステムは大学から給与が支給される教職員や TA・RA が対象であり，履修登録システムは講義を受ける学生が対象となる．そのため，各情報システムは，ユーザの職分などの属性情報から，ユーザが当該システムを利用できるかどうかを判断する必要がある．そのため，本研究では，ユーザの属性を“ロール”という名称で定義し，それぞれの情報システムで必要とされる区分分けを行った．表 4-1 にロール種別一覧を示す．学生は，在学中や既卒など 4 パターン，教員は，常勤や研究員など 8 パターン，職員は，常勤や派遣職員など 10 パターン，その他，一般公開講座受講生や医師など 13 パターンを設定し，予備も含め全部で 55 パターンに区分した．ユーザが複数のロールを持つ例として，本学の学部を卒業し，博士前期課程を修了した後に本学の職員になった場合は，学生（学部既卒），学生（修士既卒），職員（常勤）という 3 つのロールが与えられる．また，本学の職員に採用後に，本学の博士後期課程に社会人入学した場合は，職員（常勤）と学生（博士在学）の 2 つのロールを持つ．

このように，ロールを詳細に定義することで，情報システムがユーザの利用可否をコントロールできるように設計を行った．但し，複数ロールが持つユーザであっても，金沢大学 ID は 1 つとなるように留意した．

表 4-1　ロール種別一覧

ロール区分	ロール名	説明
学生	入学前	入学が決定した学生
	在学	在学中の学生
	既卒	卒業した学生
	退学	除籍または退学した学生
教員	常勤	常勤として勤務している教員
	非常勤	非常勤として講師をしている教員
	退職・転出	退職、別の学校に移動した教員

	教諭	附属学校の教員
	研究員	博士研究員
	TA	アシスタント
	RA	アシスタント
	学外教授	名誉教授、客員教授など
職員	係担当学務系以外	学務系以外の係担当
	係担当学務系	学務系の係担当
	常勤学務係	学務系の係に所属する個人としての職員
	常勤学務係以外	学務系以外の係に所属する個人としての職員
	非常勤学務係	学務系の係に所属する非常勤として勤務している職員
	非常勤学務係以外	学務系以外の係に所属する非常勤として勤務している職員
	教務補佐員	教務補佐員
	非職員学務系・秘書含	学務系の秘書や派遣など
	非職員学務系以外・秘書含	学務系以外の秘書や派遣など
	退職	退職した職員
その他	役員	役員(学内理事など)
	管理者	管理者
	一般公開講座受講生・聴講生	公開講座の受講生、聴講生
	金沢大学サポーター	本学に所属しない一般の人
	管理者(システム別用)	システム別管理者
	一般公開講座受講生・聴講生　修了	公開講座の受講生、聴講生の修了
	家族など	学生の保護者など
	研究協力員(共同研究会社社員含)	研究協力員(共同研究会社社員も含む)
	医師	附属病院の医師
	看護師	附属病院の看護師
	病院職員(技術系)	附属病院の技術系職員
	病院職員(事務系)	附属病院の事務系職員
	開発業者(共同研究除く)	共同研究以外の開発業者

4.2.3　アカンサスポータル

アカンサスポータルは平成 18 年度入学生からの携帯パソコンの必携化に合わせて導入された全学共通教育用の学習管理システム（LMS; Learning Management System）から出発し，毎年改良を重ね，時間割表示，成績照会，図書館サービス，メッセージ機能，コミュニケーションサイト（SNS; Social Network Service），スケジューラなど学生生活にはなくてはならない本学の教育用ポータルサイトである[36].

本研究においても，大学全体の情報システムの一元的な窓口として，ポータルサイトが必要となるが，複数のポータルサイトを学内に立ち上げるのは本来の趣旨に反するうえ，時間的・コスト的にも無駄なため，既存のアカンサスポータルに改良を加え，同ポータルシステムの掌握範囲を教職員の教育・研究・業務・社会貢献など様々な分野へ拡張することとした．統一認証のキーはもちろん前述の金沢大学 ID である．ただし，統合認証を用いた情報システムの一元化を実現するため，これまでアカンサスポータルと教育系の情報システム間で独自仕様の連携を行っていた部分を分離し，アカンサスポータル自身も一つの情報システムとして扱えるようにした．すなわち，アカンサスポータルを全ての情報システムの入り口とする一方，ユーザ認証は今回開発した統合認証システムで金沢大学 ID による認証を行うことで，ロールに応じて利用可能な情報システムをポータルから選択できるように設計した.

4.2.4　シングルサインオン対象システム

本節では, 本研究でシングルサインオンの対象にしたシステムについて述べる. まず，4.2.3 節で説明したように，従来のアカンサスポータルと独自仕様で連携を行っていた以下の教育系システムをシングルサインオン対象システムとした.

- WebClass（ウェブクラス社の LMS）
- 教務システム
- Web シラバス
- SNS（OpenPNE[37]を用いたコミュニティーサイト）
- NALIS（NTT データ九州社の大学図書館の各種業務を支援するシステム）
- 電子掲示板

次に，教育系以外の情報システムとして，以下を新規に対象とした．

- 給与明細オンラインシステム
- 源泉徴収関係届出システム
- サイボウズ（サイボウズ株式会社のグループウェア）
- マイクロソフト配布サーバ（Windows 系 OS および Office を教職員に配布）
- 電子職員録
- 施設管理システム
- ファイル共有アプリケーション（大容量ファイルを JavaApplet で共有）
- ファイル送信サービス（大容量ファイルをメールで共有）

Shibboleth の詳細および導入に当たって克服すべき問題点は次節で述べる．

4.3 Shibboleth 利用における問題

本研究では，Shibboleth を用いてシングルサインオンおよび属性共有を実現した．Shibboleth を採用した理由は，Shibboleth は Apache や IIS などのミドルウェアと連携して動作するため，情報システムのアプリケーション部分の修正が最小限で済むことがある．また，属性についてはサーバ環境変数で取得可能なため，情報システム間での属性共有が行いやすいという利点もある．また，GakuNin プロジェクトに代表される将来的な大学間連携を見据えていることも選定した理由として大きい．

Shibboleth の概念で本学のシステム構成を考えた場合，本学統合認証サーバとして IdP を新規に構築し，4.2.4 節で示した各種システムを SP として動作させることで，シングルサインオンおよび属性共有の実現が可能となる．しかし，4.2 節で述べた指針を実現するには，以下に述べる三つの問題をクリアする必要がある．

(ア) 複数ロールへの対応

Shibboleth では，IdP から SP に対して必要な属性を渡す．但し，複数ロールを持つユーザにも対応できるようにするには，属性の受け渡しにおいて，複数ロー

ルに対応した属性判断ができる機能が必要である．

(イ) シングルログアウト

Shibboleth はシングルログアウトに対応していないため，共用端末などでブラウザを閉じ忘れた場合は，不正に利用される危険性がある．そのため，シングルログアウトを行う手段を考える必要がある．

(ウ) IdP の冗長化

IdP は全ての SP の認証に用いられるため，多数のユーザが日常的に使用する環境に耐える状態にするには，1 台が故障した場合でもサービスが提供できるように複数台用意し，冗長化を行う必要がある．但し，そのためには，複数の IdP 間でセッション情報を共有する必要がある．

次節で，Shibboleth におけるこれらの問題について，本研究において解決した方法について述べる．

4.4 設計

本節では，4.3 節で述べた Shibboleth 利用における問題に対する解決方法について説明する．

4.4.1 複数ロールへの対応

Shibboleth では，認証のための情報および属性情報の管理を行う方法がいくつか存在するが，本研究では，その中で LDAP を選択して構築を行った．LDAP には OpenLDAP[38]を選定した．その理由は，OpenLDAP はオープンソースソフトウェアであるため安価に構築できることと，Web 検索において，Shibboleth との連携に関する実績が一番多かったことが挙げられる．但し，4.2.2 節で述べたとおり，複数ロールを多くのユーザが持つことを前提に，LDAP の設計を行う必要がある．

LDAP のスキーマを作成するにあたり，本研究では Shibboleth の属性値の返却

方法に着目した．Shibboleth では，IdP が属性値を SP に対して送信し，SP のアプリケーションがサーバ環境変数として受け取る際には，セミコロン（ ; ）で区切って 1 行の文字列として受け取る．そこで，全てのロールで同じ LDAP スキーマを使用することで，ユーザはロール数分の LDAP レコードを持つことから，セミコロンでつながったそれぞれの属性値が，どのロールの属性値かをその順番から判断することとした．

前述の通り，ロールの数はユーザによって大きく異なるので（1～10 個程度を想定），LDAP スキーマは，各ロールで共通なデータを持つベース部分を 1 つ，ロールごとに 1 つずつ持つロール部分と分離させ，ロール部の個数を可変とし，複数ロールに対応しつつ，データの無駄な重複を防ぐ設計とした．

例として，本学工学部情報工学科情報システムコースを卒業し，本学工学研究科電子情報システム専攻通信ネットワーク系を修了し，本学に就職し，企画部情報戦略課情報推進係に所属した場合に，どのように属性値が送られるかを図 4-1 に示す．このように，ベース部分は氏名や生年月日などロールで共通なものが格納されている．そして，ロール部分については，このように，セミコロンを縦に見ることで，セミコロンの各属性値がどのロールのものかをアプリケーションが容易に判断できるように環境変数として送信されてきているのが分かる．但し，Shibboleth では，属性値が同一の値であった場合は値をマージしてしまうという特性を持っている．そこで，本研究では IdP でこの動作を行っているソース部分を改変し，同一値をとる場合でも値をマージしないように修正した．さらに，Shibboleth では属性値が空白の場合は前詰めを行う部分を，値が空白のときはアットマーク（@）を属性値にセットするように改変した．このような改変を加えることで，複数ロールを持つユーザが存在する環境下でも Shibboleth で対応を行うことができるように構築を行った．

4.4.2 シングルログアウト

Shibboleth を利用することで，ユーザは IdP で 1 度認証を行えば，他の情報システムにアクセスした場合でも再度認証を行う必要がなくなる．一方で，ログアウトを行った際には全ての情報システムでログアウトを行う“シングルログアウト”の機能が必要である．しかし，SAML2.0 ではシングルログアウトの仕様が策定されているが，Shibboleth にはまだ反映されていない．そのため，本研究では，全ての情報システムの窓口であるアカンサスポータルにログアウト機能を配置し，

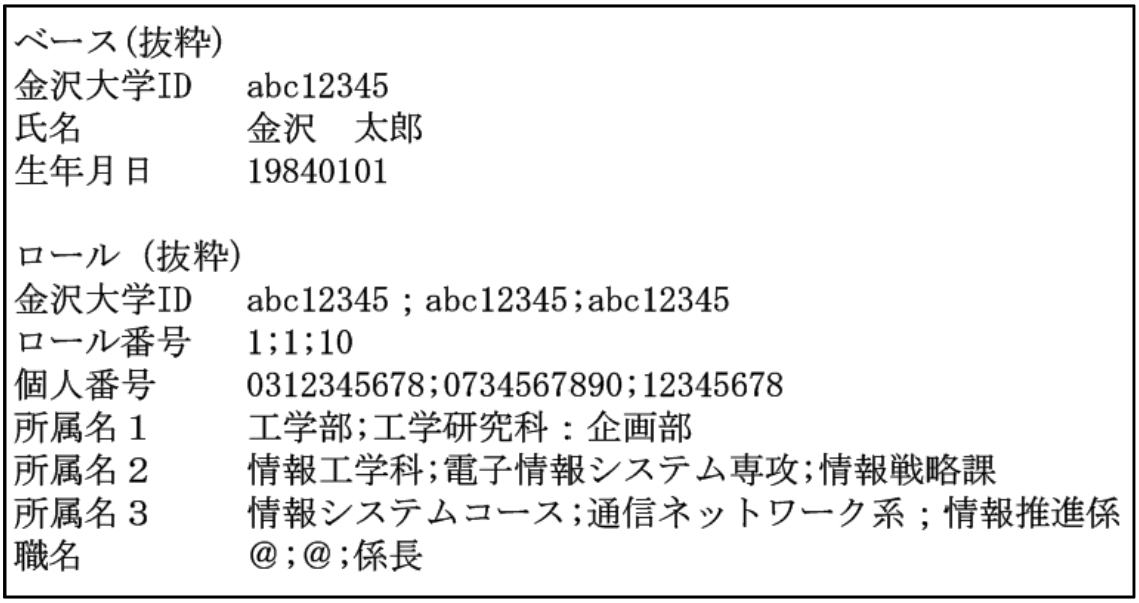

図 4-1 複数ロールにおける属性値例

アカンサスポータルのログアウトを実行することで他の全ての SP からログアウトを行うシングルログアウト機能を構築した.

まず IdP に対してログアウト処理を行い，そのあと順番に全ての SP に対してログアウト処理を行うことでシングルログアウトを実現している．図 4-2 にシングルログアウトの概念図を示すとともに，以下に IdP，SP それぞれのログアウト方法を記載する.

(ア) IdP のログアウト方法

IdP においては，バージョン 2.1.5 ではログアウト機能は実装されていない．そこで，ユーザ側での IdP セッション管理方法を利用して，セッション情報を破棄するスクリプトを実行することで IdP のログアウトを実現した．具体的には，ユーザのブラウザ上では IdP とのセッション情報は Cookie で管理されているため，該当する Cookie と同じ変数名の Cookie を新規に生成し，その値を空にセットしてユーザのブラウザへ送信する．そうすることで，IdP とユーザ間のセッションはリセットするため，IdP からのログアウトが完了する.

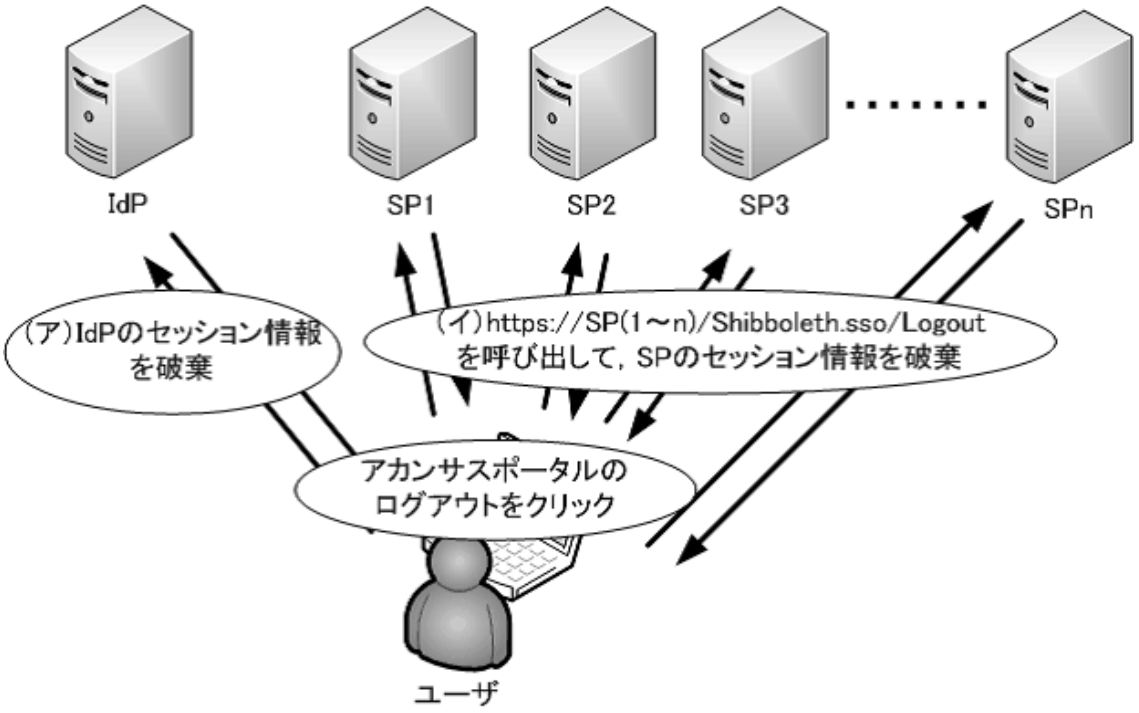

図 4-2 シングルログアウト概念図

(イ) SP のログアウト方法

IdP でログアウトが完了しても，既にログインされた SP については，セッションがタイムアウトになるまで IdP に再度問い合わせを行わないため，ログインが可能な状態になっている．そのため，それぞれの SP においてもログアウト処理を行う必要がある．本研究では，それぞれの SP 単体でのログアウトは実装され，

https://SP サーバ名/Shibboleth.sso/Logout

にアクセスすることでユーザのブラウザとのセッションが破棄されることに着目し，上記 URL を全ての SP から呼び出すことで，全 SP からのログアウトを実現した．

（ア），（イ）の動作を一連の流れで動作するようにスクリプトに実装し，シングルログアウトを実現した．

4.4.3 IdP のクラスタ化

IdP は全ての SP の認証に用いられるため，認証システムの要として位置づけられる．その為、サーバは 1 台で運用するのではなく，複数台用意し冗長化を図る必要がある．しかし，IdP は SP とのセッション情報を管理しているため，単純に負荷分散装置で管理することはできない.

その為，Internet2 では IdP のクラスタ化を行う方法として，Terracotta[39]の使用を推奨している. Terracotta は複数の JavaVM 上で同じ Java オブジェクトを共用できるオープンソースのミドルウェアである. Terracotta を用いることで, 複数の IdP 間でセッションの共有を行うことが可能になる．本学では，クライアントの IP アドレス単位でロードバランシングを行い，Terracotta でセッション情報の共有を行うように設計を行った．また，Terracotta のパフォーマンスチューニングとして，Internet2 が推奨する JavaVM メモリのセッティングを行った[40].

負荷分散装置において, クライアント IP 単位で負荷分散を行う設定にしておくことで，IdP の冗長化を実現した.

4.5 実装

本節では，本学における統合認証環境のシステム構成について説明するとともに，認証・認可を行うためのユーザ情報の管理について述べる．さらに，ユーザの統合認証の流れについても説明する.

4.5.1 全体システム構成

本学に構築したシステムの概要を図 4-3 に示す. 図中にあるポートフォリオ DB は，アカンサスポータルが参照するユーザデータの集まりである．アカンサスポータルサーバ（SP），IdP サーバ，LDAP サーバはそれぞれ 2 台ずつ用意し，冗長化を行うとともに，負荷分散を行っている．それぞれのサーバのスペックなどの基本情報を表 4-2 に示す．また，負荷分散装置には，Fujitsu 社 IPCOM BX1200LB を用い，アカンサスポータル，IdP，LDAP へのアクセスは必ず，負荷分散装置を経由するように設計し, アクセス単位はクライアント IP アドレスによるノード単位で負荷分散を行うように構築した.

4.2.4 節で述べたシングルサインオン対象システムは，アカンサスポータルと同様に SP として動作するように実装を行った．各 SP は，IdP から送信される複数ロール情報をセミコロンで区切り，配列に格納させることで，ユーザに対して，ロールに対応したサービスを提供できるようにした. そして図 4-3 に示すとおり，全ての SP へはアカンサスポータルのリンクから辿るように実装を行った．ユーザはアカンサスポータルにログインする必要があるため，シングルログアウト機構をアカンサスポータル内に配置することで，シングルログアウトの問題を解決した．また，IdP においては，Terracotta を用いて 2 台の IdP 間でセッション共有を行い，負荷分散装置経由でアクセスすることで IdP クラスタ化を実現した.

表 4-2 サーバ構成

用途	サーバ諸元	OS	ミドルウェア
IdP	Fujitsu PRIMERGY BX620 CPU: Xeon X5570 (2.93GHz/8MB)×2 Memory: 8GB HDD: 300GB(2.5inch SAS 10000rpm (RAID1))	Red Hat Enterprise Linux 5.4(x64)	ShibbolethIdP2.1.5 Apache2.2.3 Jdk1.6.0update17 Tomcat6.0.20 Ant1.7.0 Terracotta3.1.1
SP (ポータル)	Fujitsu PRIMERGY BX620 CPU: Xeon X5570 (2.93GHz/8MB)×2 Memory: 8GB HDD: 300GB(2.5inch SAS 10000rpm (RAID1))	Red Hat Enterprise Linux 5.4(x64)	ShibbolethSP2.3.1 Apache2.2.14
LDAP	Fujitsu PRIMERGY BX620 CPU: Xeon X5570 (2.93GHz/8MB)×2 Memory: 8GB HDD: 147GB(2.5inch SAS 10000rpm (RAID1))	Red Hat Enterprise Linux 5.4(x64)	OpenLDAP2.4.19

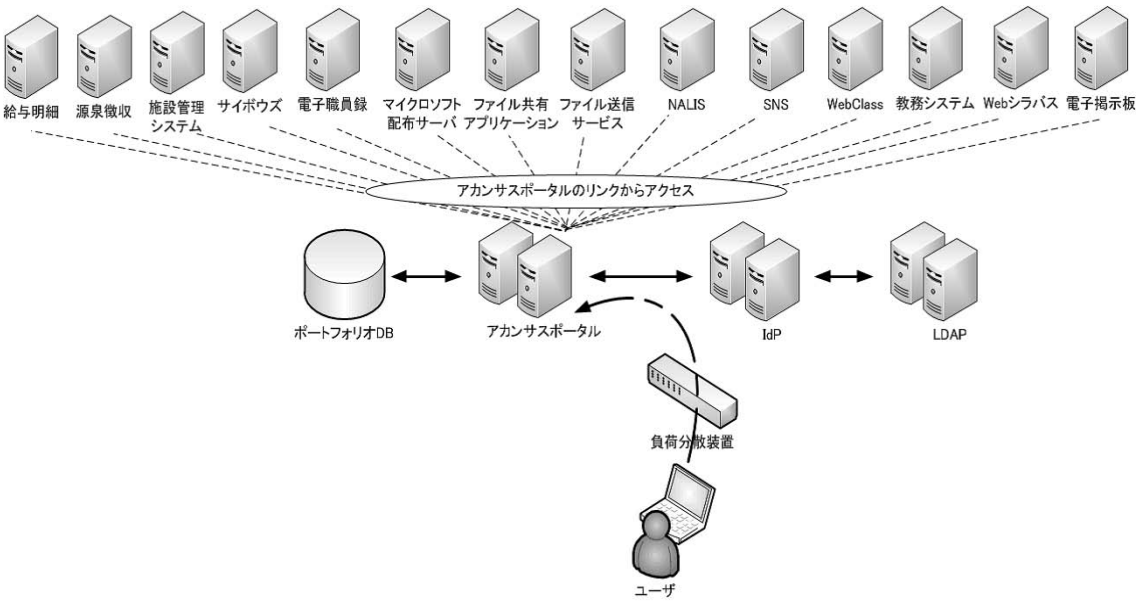

図 4-3 金沢大学における統合認証システムおよびポータルシステム概要

4.5.2　ユーザ情報の管理

本節では，ユーザにおける認証情報および属性情報の LDAP への反映方法について述べる．教職員および学生に関する情報は，人事システムおよび教務システムから取得を行う．人事システムおよび教務システムから取得した情報は，アカンサスポータルを経由して SOAP（Simple Object Access Protocol）通信により LDAP に反映される．アカンサスポータルでは，新規に取得した情報と既に LDAP に登録されている情報の比較を行う．LDAP に存在しないユーザは「新規」，両方に情報が存在し情報に差異がみられるユーザは「変更」，新規データ内に情報が存在しないユーザは「退職」となる．なお，「退職」になっても，ロールが変更になるだけで，ユーザ情報の削除は行わない．また，派遣職員や共同研究会社社員など人事システムおよび教務システムのどちらにもユーザ情報が存在しない場合は，アカンサスポータルに配置したユーザ管理画面で登録および変更を行う．

4.5.3　統合認証の流れ

ユーザは，アカンサスポータルにアクセスすると，IdP にリダイレクトされ，図 4-4 に示す統合認証画面において認証を行う．認証に成功したのち，アカンサスポータルには，IdP からユーザの基本情報およびロール情報が送信され，その情報を基に自分が与えられたロールでログインを行う．ユーザは，自分が現在所属しているロールを選択することが可能で，アカンサスポータルは，ユーザが選択したロールに応じて利用可能な情報システムを提示する．そしてユーザは，アカンサスポータルを経由して利用したい情報システムにアクセスするという手順をとる．ユーザはアカンサスポータルから他の情報システムにアクセスを行っているように感じるが，アカンサスポータルもその他の情報システムと同様に，ShibbolethSP として構築を行っているため，実際は Shibboleth によるシングルサインオンを行っていることになる．すなわち，ユーザが直接各情報システムにアクセスをした場合も，認証前であれば統合認証画面が表示され，1 度でも認証された後であれば直接その SP のサービスを受けられることになる．SP では，IdP から送信されてきた属性情報を基に，ユーザの利用制限を行う．ユーザが複数のロールを持っていた場合は，アプリケーションサイドでセミコロンを分割し，属性値を確認するように構築を行っている．4.4.1 節で説明したように LDAP スキーマを設計したことにより，アプリケーションサイドでの作業は最小限に留まる工

図 4-4 認証画面

夫がなされている.

このように，ユーザは情報システムへの窓口がアカンサスポータルで統一化されるとともに，金沢大学 ID で 1 度認証を行えば，傘下のサービスが一元的に利用できる環境が実現した.

4.6 評価

本節では, 4.5 節で説明した本学における統合認証環境の運用実績について述べる.

4.5 節で述べた統合認証環境は平成 22 年 3 月 22 日から本格稼働を開始した. 図 4-5 に平成 22 年 4 月 1 日から 4 月 30 日の 1 ヶ月間の IdP での認証数の推移を示す. 4 月は学生の履修登録や，全入学生対象の情報処理基礎の講義があり，アカンサスポータルを経由しての LMS の利用が多いため, 1 年で最もアクセスが集中する月である．7 日の入学宣誓式の日に各学類によるオリエンテーションがあり，そ

の際にアカンサスポータルの説明があったため, 認証数が 5100 回とアクセス数が前後の日よりも多い．また，12 日の前期授業開始から認証数が増加して推移しているのがわかる．そして， IdP での認証は 21 日の 148,320 回が 4 月の最大値である．アカンサスポータルを経由して，WebClass を多くの講義で利用していたことに起因している．このように，圧倒的に認証数の多い 21 日においても，認証処理ができないなどの問題は発生しなかった．次に，4 月 21 日の IdP の認証数とそれにともなう CPU 負荷率およびメモリ使用率を図 4-6 に示す．図 4-6 から，15 時頃から大量のアクセスが発生しているのがわかる．これは，情報処理基礎で，181 名の学生が同時に講義を受講しているのと同時に，複数の学類でもアカンサスポータル経由で WebClass を用いた講義を行っていたためと考えられる. 認証数の増加にともない，メモリの使用率が大きく増加しているのが見て取れる．一方で，CPU については 2～3%程度しか使用していない．すなわち，IdP においては認証数とメモリについては相関があり，認証数の増加にともない，メモリの使用率は増加する．しかし，CPU においては認証数が増加しても負荷にはほとんど影響を及ぼさない．そして，これだけの認証数があってもメモリは 20%以上の余力があり，CPU にもほとんど負荷がかかっていないことから，IdP の動作には問題ないことが確認することができた.

21 日のアカンサスポータルサーバにおいて，IdP サーバと同様に認証が成功した数とその時の CPU 負荷率およびメモリ使用率をグラフにしたものを図 4-7 に示す．このように，アカンサスポータルサーバにおいてはメモリ，CPU 共に認証数が増減してもほとんど変化していないことがわかる．そのため，SP においてもリソースに負荷を与えることなく運用できていることを確認できた.

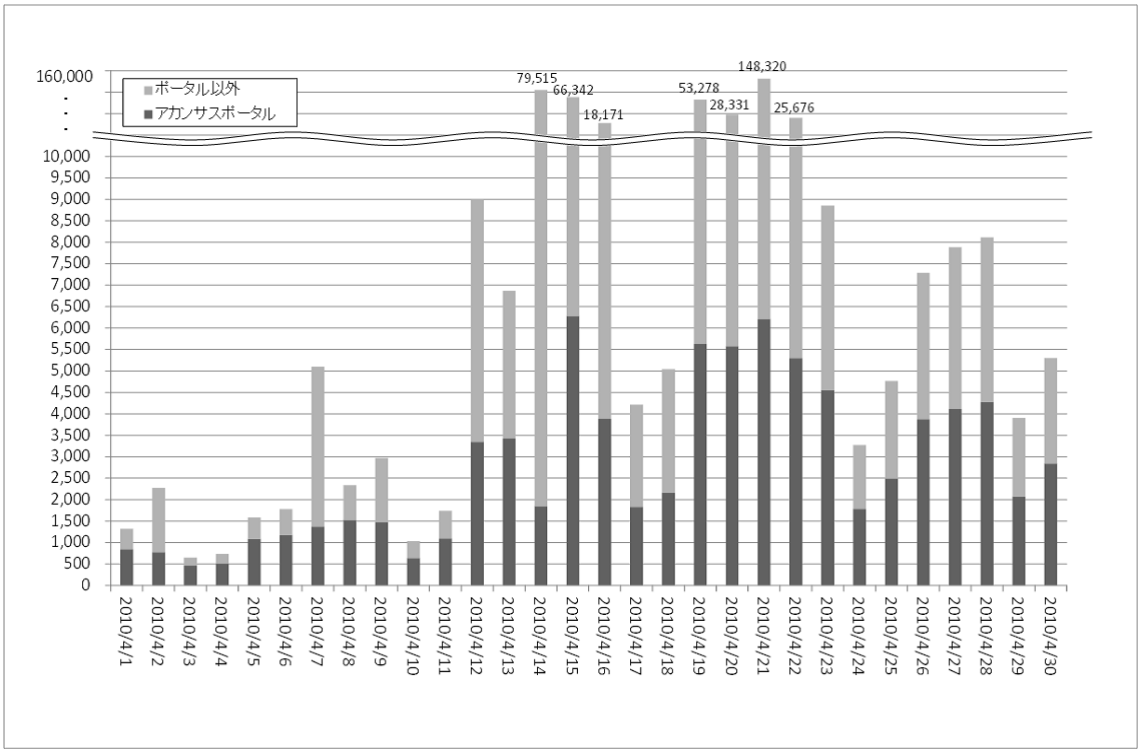

図 4-5 IdP における認証数の推移(2010/4/1～2010/4/30)

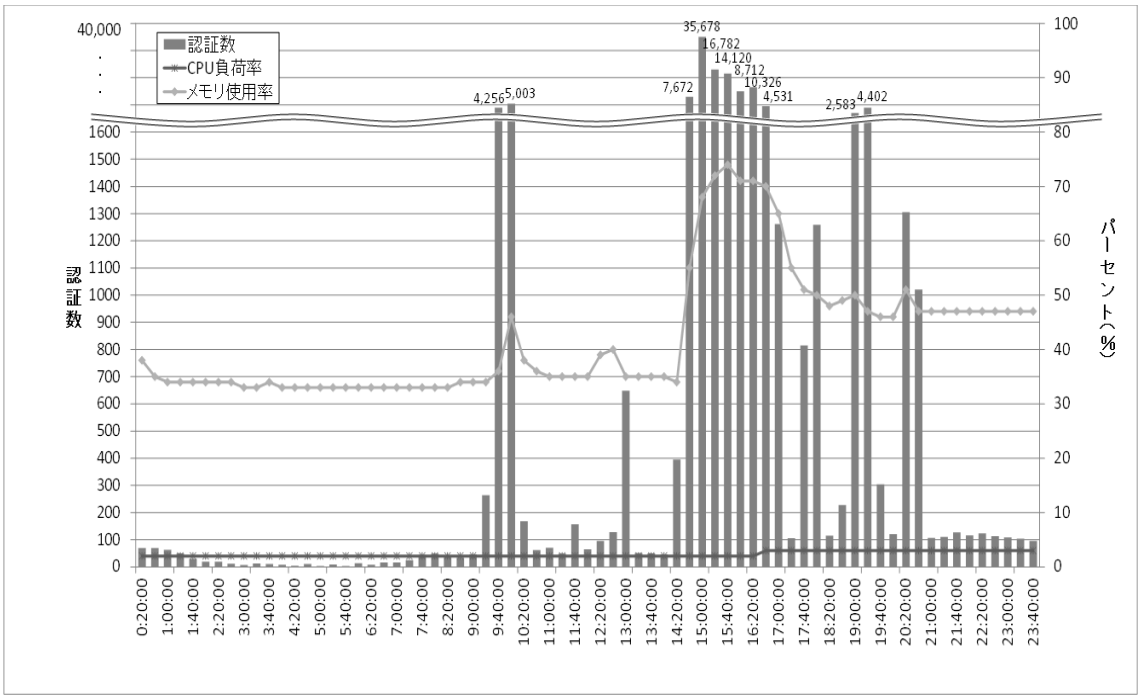

図 4-6 IdP サーバにおける認証数と CPU 負荷率およびメモリ使用率(2010/4/21)

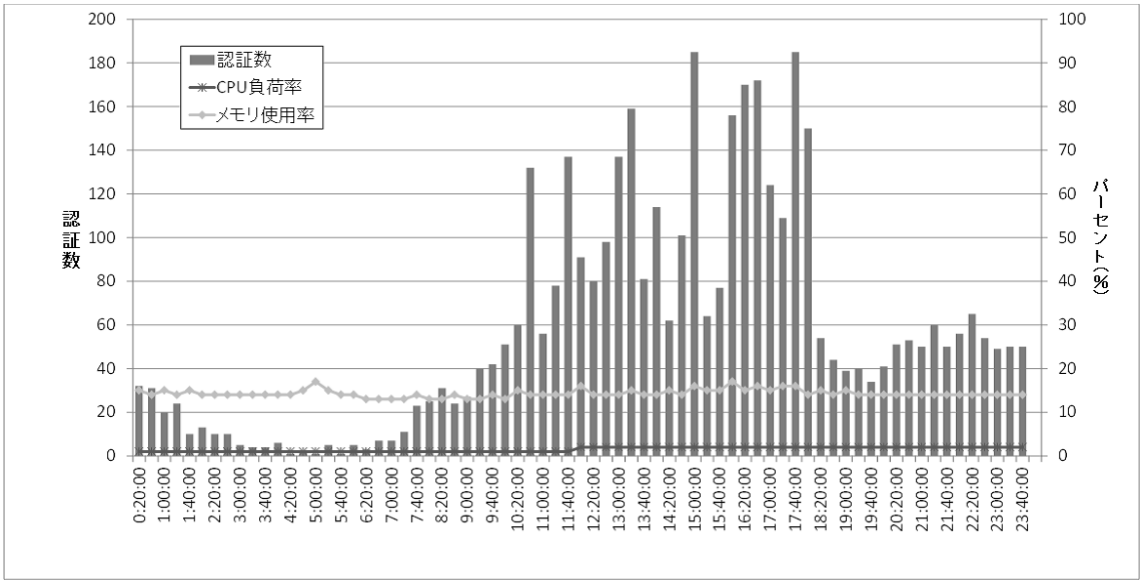

図 4-7 アカンサスポータルにおける IdP 認証数と CPU 負荷率およびメモリ使用率(2010/4/21)

4.7 まとめ

4.7.1 成果

本章では，本研究で行った本学における統合認証システムの設計，実装，システム評価について述べた. 本システムにより実現した項目は以下のとおりである.

- Shibboleth を導入することにより，これまでは独立していた各種情報システムにおける認証機構の一元化を実現した.

- 金沢大学 ID という生涯 ID を導入するとともに，LDAP スキーマの設計に配慮し，ShibbolethIdP の属性情報の送信形式を利用することにより，複数ロールに対応できる仕組みを実現した．これにより，各情報システムはロールに応じたサービスを提供できるようになった.

- Shibboleth では未実装のシングルログアウトを実装したことで, セキュリティを向上することができた.

- Terracotta を用いることで IdP のクラスタ化を実現し，信頼性の高い認証機構を構築した．

- 構築したシステムの評価を行うことで，実運用にも十分耐えうるシステムであることを実証した．

- 大学特有の複雑な所属形態にも対応し，将来的に大学間連携にも活用できる統合認証基盤環境を構築できた．

本学のように Shibboleth を研究的な側面で利用するのではなく，教育と業務に拡大し，日常的に利用されている情報システムに適用している点が，これまで報告されているものと大きく異なる．そのため，本学の取り組みはこれから Shibboleth を導入する様々な組織においても十分転用可能なレベルに達していると言える．また，これから情報システムの一元化を検討している大学関係者の一助となることを期待している．

4.7.2 今後の計画

今後，本学統合認証システムの発展において以下の計画を考えている．

- GakuNin との連携

NII が中心となって大学間連携を推進しており，平成 21 年度からは電子ジャーナルや一部の NII のサービスが GakuNin を利用し，シングルサインオン環境で利用できるようになってきている．本学も 3 章で述べたとおり，積極的に活動に参加を行っている．しかし，現在は GakuNin で本学が認証を行う IdP は，今回構築した統合認証システムのものとは別のサーバを構築し，運用を行っている．理由としては，金沢大学 ID は半永久的に有効であり，GakuNin で提供している大学在籍者に提供するサービスと十分整合が取れていないことが挙げられる．そのため現在は，DS を構築して学内 SP 用 IdP と学外 SP 用 IdP をユーザに選択させるとともに，GakuNin 用の LDAP サーバを構築し，学外 SP 用 IdP に参照させる必要がある．この問題を解決するために，本稿で扱ったロールによる認可制御を GakuNin でも活かせる仕掛けの実現が必要と言える．

- 携帯電話への対応

アカンサスポータルや WebClass などは携帯電話からのログインを許可している．しかし，Shibboleth では認証に Cookie を用いているが，一部の携帯電話では Cookie を使用することができない．その為，今後 Shibboleth のソースを一部改修し，Cookie を用いない方法で Shibboleth 認証を行えるようにすることを検討している．今後は携帯電話を使ったアクセスは増えると考えられるため，対応することは非常に重要である．

- SP の冗長化について

今回，WebClass のアクセスが多い原因として，3 台構成で DNS ロードバランスを行っていることがある．そのため，SP をクラスタ化することで，IdP における認証数を大幅に削減できることが予想される．SP のクラスタ化に関しては，Shibboleth の SP 機能を用いることで実現可能であり，テスト環境においては正常に動作し，IdP での認証数が大幅に減っていることを確認している．しかし，現在は ShibbolethSP デーモンの冗長化は行われておらず，ShibbolethSP デーモンを起動しているサーバに障害が発生した場合は，結局サービスを継続することができなくなる．そのため，できるだけ早急に ShibbolethSP デーモンの冗長化を実現することが望まれる．ShibbolethSP デーモンの冗長化に成功したのち，本番環境に適用することを考えている．

- 名寄せの対応

職員となった後に社会人学生になった場合や，学部を卒業してから職員になった場合などは，最初はそれぞれ異なる金沢大学 ID がロールごとに割り当てられる．その後に，アカンサスポータルの名寄せ管理画面から，金沢大学 ID をどちらかに寄せる処理を行う必要がある．今後は，名寄せ処理の完全自動化について検討を行う予定である．

- ユーザ情報の自動反映

4.5.2 節で述べたとおり，統合認証基盤で利用するユーザ情報は人事システムおよび教務システムから取得している．現在は教務システムとはアカンサスポータルを通じて連携を行い，変更があった場合に自動で即時反映が可能である．しかし，人事システムはデータの更新は手動で月 1 回となっている．そのため，今後

はシステムの運用部署である総務部人事課とアカンサスポータルとの自動連携に向けた協議を進めていく予定である．

- IdP および SP サーバの仮想化

IdP および SP については，CPU の負荷がほとんどなく，メモリの使用率もまだ余裕があることがわかった．その為，今後 IdP や SP を追加していく際には，サーバを仮想化し，リソースの節約を行うことが望まれる．

これらの機能を実現することにより，さらに大規模で，柔軟性に富んだ統合認証システムへと発展していくことができると考えている．

第5章　多様な公開ポリシに対応した分散データ相互参照システムの構築

5.1 はじめに

3 章，4 章で説明したとおり，大学内および大学間連携において Shibboleth を利用した統合認証基盤を実現できた．但し，研究室などで情報システムの利用を制限する場合は，大学で公式に管理されているユーザ属性情報だけでは不足する場合が多い．そこで，本章では研究室単位での分散データの相互参照における多様な公開ポリシに対応するときの Shibboleth 適用について議論を行う．

大学の研究室や研究機関には，実験観測データや学術論文など，様々な学術情報が蓄積されている．さらに日々の研究により，新しい学術情報が生産され続けている．特に価値の高い学術情報は，情報を保有する研究機関のみならず，国内外に点在する研究グループ，さらには分野を超えた研究者からも学際的な研究利用のために，広く相互参照の要望が高まっている．

学術論文などの文献に関しては，国立情報学研究所が中心となり，機関リポジトリとして，電子的な形態で集中的に蓄積・管理を行う手法を推進している[41]．本学においても金沢大学学術情報リポジトリ（KURA）として，学内における学術論文を公開している[42][43]．

このように，機関リポジトリ上に蓄積される学術論文は，完成された研究成果として，一般公開を原則とする機関リポジトリのポリシに沿って広く公開される．しかしながら，実験観測データなどは，学術論文などの形で成果公表される前の基礎的な学術資料として利活用される場合も少なくない．すなわち，実験観測データにおいては，一般公開を原則とする運用ポリシは馴染みにくいものといえる．

実験観測データを扱う研究分野においては，研究プロジェクトを形成し，複数

の研究室や研究チームが，ときには大学や研究機関をまたいで共同研究を進めることが少なくない．そのため，実験・観測で得られたデータを単に広く公開するだけでなく，研究者個人間でのみ共有したい場合や，プロジェクト内に限定して共有したい場合など，様々なユースケースが存在する．実験観測データを共有している例として，天文学の分野では，国立天文台が海外の研究機関と協力して推進する Virtual Observatory（VO）[44], 地球惑星科学分野においては, STARS[45][46], IUGONET プロジェクト[47]などが挙げられる．STARS においては，データを中央で集中管理することを前提として設計を行っている．その為，情報を閲覧するためのユーザ管理や，データのアップロード作業およびアクセス制限は全て中央のシステム管理者に委ねることになり，データ公開までの作業が煩雑である．このような中央管理的なデータベースシステムは，大きな研究プロジェクトを推進する目的には有効でも，小規模グループが相互にデータを共有・参照する形態には馴染まず，結果的にデータの多くが研究者個人の PC や研究室内サーバに埋没し，データの流通性が十分に確保できないという問題がある．一方，VO や IUGONET プロジェクトにおいては，実験観測データについては，保有する組織に分散した運用を想定している．しかし，VO は天文学固有の画一的なフォーマットを用いているため, 分散管理技術の他分野への応用が難しい. また, IUGONET プロジェクトは，主幹大学が分散保有するデータに関するメタデータを統合的に扱うことを目的としており, 実データの相互参照は将来の検討課題となっている．これらの問題を克服する有効な手段が確立していないのは，組織を超えてデータを共有する際に，どのようにしてデータ共有相手を特定するかが大変困難であることに起因する．

このような背景から，本研究では，組織を超えて利用可能な，多様な公開ポリシに容易に対応したデータ相互参照システムとして，ARchive system for Cross-reference Across Distributed Environment （ARCADE）を開発した．本研究で提案する ARCADE は，異なる組織間で認証情報，属性情報，認可情報を安全・安心に交換することを第一目標に掲げている．さらに ARCADE は，ユーザ情報を所属する組織ごとに独立して管理可能であることに特徴がある．また，実験観測データの実体は，データを保有する組織内に配置したままで，公開ポリシを設定できるため，中央に集中させる必要がない点も大きな長所である．更に，ARCADE を Java アプリケーションで構築することで，ドラッグアンドドロップなどを駆使し，IT スキルに乏しいユーザであっても視覚的に操作可能なインター

フェースを確立した.

本章では，5.2 節で ARCADE の設計概念およびシステム諸元について述べる．5.3 節で ARCADE の実装および動作詳細を紹介する．5.4 節で実証運用について述べ，5.5 節でまとめる.

5.2 ARCADE 設計概念

5.2.1 達成すべき課題

本節では, ARCADE の設計に当たり, 解決すべき重要課題として, ユーザ管理, アクセス管理，ユーザビリティについて説明する.

5.2.1.1 ユーザ管理

本節では，実験観測データを扱う研究分野のプロジェクト構成をモデル化し，それに基づくユーザ管理の課題について議論する．図 5-1 は，本研究でターゲットにする ARCADE ユーザで構成される研究プロジェクトをモデル化したものである.

この図では，プロジェクト A は 3 つのチームからなり，チーム A1 は A 大学 A 研究室の A さんと B さん, A 大学の B 研究室の C さんと D さんから構成される．また，A さんと B さんはチーム A3 にも所属し，そこには C 大学 D 研究室の G さんと H さんも属している．このように，1 人のユーザが複数の研究プロジェクトや研究チームに属する場合を想定すると，プロジェクトの結成や解散の度にユーザ情報を変更する必要が生じる．そのため，ユーザ属性として一時的な情報を設定するよりも，そのユーザを特定可能な必要最低限の情報を設定したほうがユーザ属性の管理が容易になると考えられる．また，ユーザ情報を常に最新のものに維持するためには，研究室単位，つまり，組織としての最小単位でユーザ情報を管理することが望ましい.

そのため，ユーザ管理においては，組織単位でユーザが管理でき，かつ組織間で共通のユーザ情報を持つ機構が要求される.

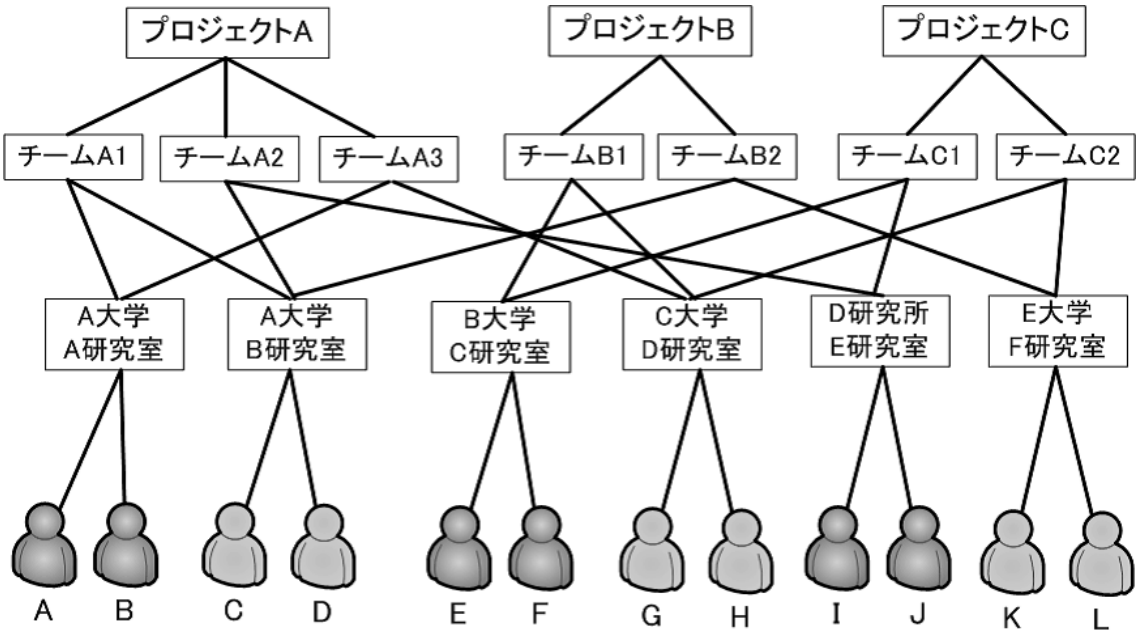

図 5-1 研究プロジェクトのモデル図

5.2.1.2 アクセス管理

次に，データを保有する各組織が決定するデータのアクセス管理の課題について述べる. 実験観測データの公開を促進するためには, データを保有する組織が, データを自組織内に配置したまま，他組織のユーザに公開できる仕組みを構築することが最も簡便かつ望ましい方法と言える．さらに，データ公開者が，公開するデータに対して，簡単にきめ細やかに公開ポリシを管理できる必要がある．

そのため，実験観測データのアクセス管理については，各組織がデータを自組織内で管理でき，かつ簡便に公開制御可能な機構が要求される．

5.2.1.3 ユーザビリティ

本節では, ARCADE の構築におけるユーザビリティに関する課題について述べる．システムに要求される技術としては，5.2.1.1 節および 5.2.1.2 節で説明した課題を克服する必要があると述べたが，解決した結果，非常に管理が複雑なものであってはならない．そのため，ARCADE においては，ユーザビリティについても重要な検討課題として掲げた．

5.2.2 開発方針

前節で挙げた課題を解決し，ユーザの IT スキルに依存しない，多様な公開ポリシに対応した分散データ相互参照システムを構築するために本研究で提案するシステム開発方針について説明する．ユーザ管理については，本研究では LDAP を用いて構築を行うこととした．その際，ソフトウェアは OpenLDAP を用いた．また，アクセス管理は，Shibboleth を用い，ユーザ管理で使用している LDAP と連携させることで，アクセス制御を行うこととした．いずれも，オープンソースソフトウェアを用いることにより，規模の小さな研究室でもコスト的に問題なく導入できるように配慮した．さらに，ユーザビリティについては，ARCADE が，ユーザがデータ公開やデータのダウンロード及びアップロードを視覚的にかつ簡便に操作できるよう，Java アプリケーションで構築を行った．

5.3 ARCADE の構築

本節では，ARCADE の概要述べるとともに，実際の画面を含め，動作について説明する．ARCADE の動作概念図を図 5-2 に示す．以降の説明においては，KanazawaUniversity，SampleUniversity，TestUniversity の 3 つの組織間でデータの共有を行う場合を想定して話を進める．前節で述べたように，ARCADE におけるシステム基盤には，Shibboleth を採用している．

5.3.1 ユーザ管理

ユーザ管理においては，5.2.1.1 節で説明したように，組織単位でユーザが管理でき, かつ組織間で共通のユーザ情報を持つように LDAP の構築を行った. 表 5-1 に設計した LDAP スキーマを示す. ユーザが持つべき情報として, ローマ字名字, ローマ字名前，ログイン ID，パスワード，組織名，所属名，身分，メールアドレス，管理者フラグ，備考を属性として用意した．管理者フラグは LDAP のユーザ情報を変更できるかどうかで，1 の場合は変更可能とした．また，属性型は LDAP で標準的に利用されているものを使用するように配慮した．

Shibboleth と組み合わせることで，情報システムごとに認証機構を構築する必要がなくなり，自組織のユーザ管理だけを行えばよくなる．

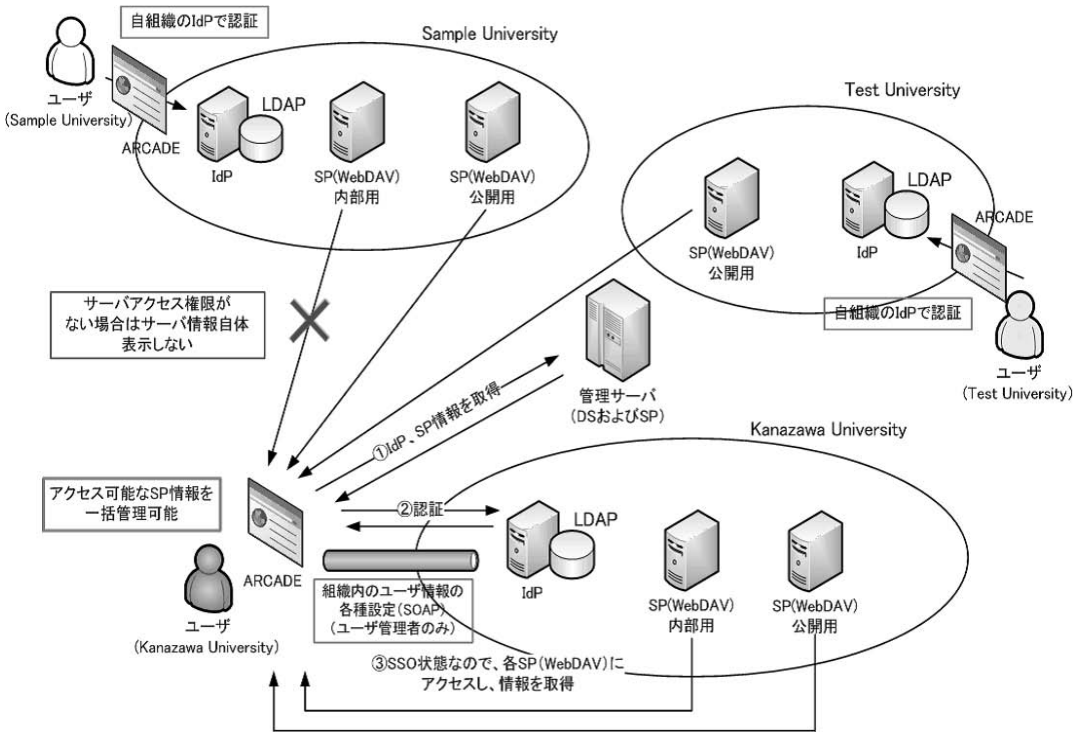

図 5-2 ARCADE 動作概念図

5.3.2 アクセス管理

アクセス管理においては，5.2.1.2 節で述べたように，各組織がデータを自組織内で管理でき，かつ簡便に公開制御が行えるように，Shibboleth 環境を適用した．Shibboleth は，Web サービスにおいて，LDAP の情報を利用して，簡単にアクセス制限を行うことができる．特に，Shibboleth は Apache と親和性が高く，Apache のアクセス制限方法の 1 つである，.httaccess を利用することで，簡単にディレクトリ単位でアクセス制限を行うことが可能である．図 5-3 にその設定例を示す．この例では，表 5-1 のログイン ID である uid をキーとして，uid が「ishikawa」と「kanazawa」のユーザに対してカレントディレクトリの参照権限を与えるという設定になる．

このように，Apache のアクセス制限を利用することで，簡単にユーザの属性に応じたアクセス制限を行うことが可能となる．

また，各研究機関のデータは，SP 内に格納する．そこで，データ管理者がデー

表 5-1 LDAP 属性

属性種類	LDAP	属性値例
ローマ字名字	sn	Ishikawa
ローマ字名前	givenName	Taro
ログイン ID	uid	ishikawa
パスワード	userPassword	ishikawasan
組織名	o	KanazawaUniversity
所属名	ou	IshikawaLab
身分	employeeType	Professor
メールアドレス	mail	ishikawa@kanazawa-u.ac.jp
管理者フラグ	title	1
備考	description	administrator

```
AuthType shibboleth
ShibRequireSession On
require uid ishikawa
require uid kanazawa
```

図 5-3 アクセス制限例

タをアップロードしたり，データ利用者がデータをダウンロードしたりする仕組みとして，本研究では WebDAV（Web-based Distributed Authoring and Versioning）[48]を利用することを考えた．WebDAV は Web アクセスの技術をベースにした異なる OS のマシン間でファイルアクセスを実現するためのプロトコルである．つまり，クライアントから Web サーバ上のファイルやディレクトリを管理することができる．WebDAV は，https のプロトコルを使用できるため，Shibboleth と WebDAV を組み合わせても，443 ポートのみのアクセスで済むというメリットがある．

WebDAV サーバ機能については，Apache においては，標準で対応しており，アクセス制限の簡便さも含め，本研究では Apache によって SP の構築を行った．

5.3.3 ユーザビリティ

ユーザビリティを確保するためには, ARCADE 利用者には, IdP, SP, DS, LDAP など, 環境が複数のサーバによって構成されていることを意識させないようにし, 全ての操作が ARCADE 上で行えるように実装を行う必要がある．そこで, ARCADE は Java アプリケーションとして開発した．その際，Standard Widget Toolkit[49]（以降，SWT と記載）を用いて実装した．SWT は Eclipse[50]開発のために設計された GUI を作成するためのツールキットである．但し，SWT 自体は Eclipse に依存しておらず, 単独の GUI ツールキットライブラリとして利用できる. SWT の大きな特徴として，ボタンやテキストボックスなどのウィジェットを OS ネイティブのものを採用している点がある．その為，Java 独自のウィジェットを使用するよりも動作が軽快でかつ，見た目が OS ライクなため利用者のユーザビリティが向上するメリットがある.

また，ARCADE は Java Web Start[51]を採用している．Java Web Start ソフトウェアを使用することにより，Web から Java アプリケーションをダウンロードして実行することが可能になる．その利点として以下のことが挙げられる.

- アプリケーションを 1 回のクリックで起動
- 常に最新バージョンのアプリケーションを実行
- 複雑なインストールやアップグレード作業が不要

これらの工夫を駆使することで，ARCADE では，GUI ベースで，IT スキルに乏しいユーザであっても簡単に操作を行うことが可能となった．次節以降で ARCADE の実際の動作について説明する.

5.3.4 ARCADE の動作

本節では，ARCADE を使用するユーザの動作について説明する．本節における説明は，図 5-2 の環境を用いることとする.

5.3.4.1 認証動作

本節では，ARCADE における認証動作について説明する．ARCADE のアプ

リケーションを配置しているサーバを管理サーバと定義する．管理サーバは，DS と SP の役割を持つとともに，各研究組織に分散している SP 情報の管理を行う．Java Web Start から起動すると，ARCADE は管理サーバにアクセスを行う．管理サーバは各組織が保有する SP 情報を管理しているため，Shibboleth 認証を必要とする．そのため，DS へとリダイレクトを行う．ユーザには図 5-4 に示すように認証先を選択する画面を提示し，自組織の IdP を選択する．次に，図 5-5 に示す認証画面を提示し，ユーザは，組織から配布された ID とパスワードで認証を行う．認証に成功したのち，ARCADE は各組織の SP 情報を管理サーバから取得することができる．

5.3.4.2　アクセス制御動作

本節では，5.3.4.1 節で認証が成功したユーザにおけるアクセス制限について説明する．図 5-6 に ARCADE のデータ管理画面を示す．画面はディレクトリ表示（①），ファイル情報表示（②），権限設定（③），ユーザ情報閲覧（④），ユーザ管理（⑤）によって構成される．数字は図に記載された数字と対応する．

5.3.4.2.1 ディレクトリ表示

本節は，ディレクトリ表示（①）の機能について説明する．ARCADE においては，SP 情報は XML で管理を行う．各サーバの URL に対してラベルを用意し，ユーザにはラベル部分だけを提示する．また，組織内では表示させるが，組織外には非公開とする SP においては，SP のルートディレクトリに対して，組織の属性を持つユーザのみアクセスを許可する設定にしておくことで対応する．複数の SP 情報を一度に表示することで，ユーザは簡単に SP を横断的に参照することができる．また，ユーザには，参照権限があるディレクトリツリーのみを表示する．また，ユーザは，ツリー上で右クリックを行うことで，ディレクトリの作成及び削除を行うことができる．

図 5-4 DS 画面（IdP 選択）

図 5-5 認証画面

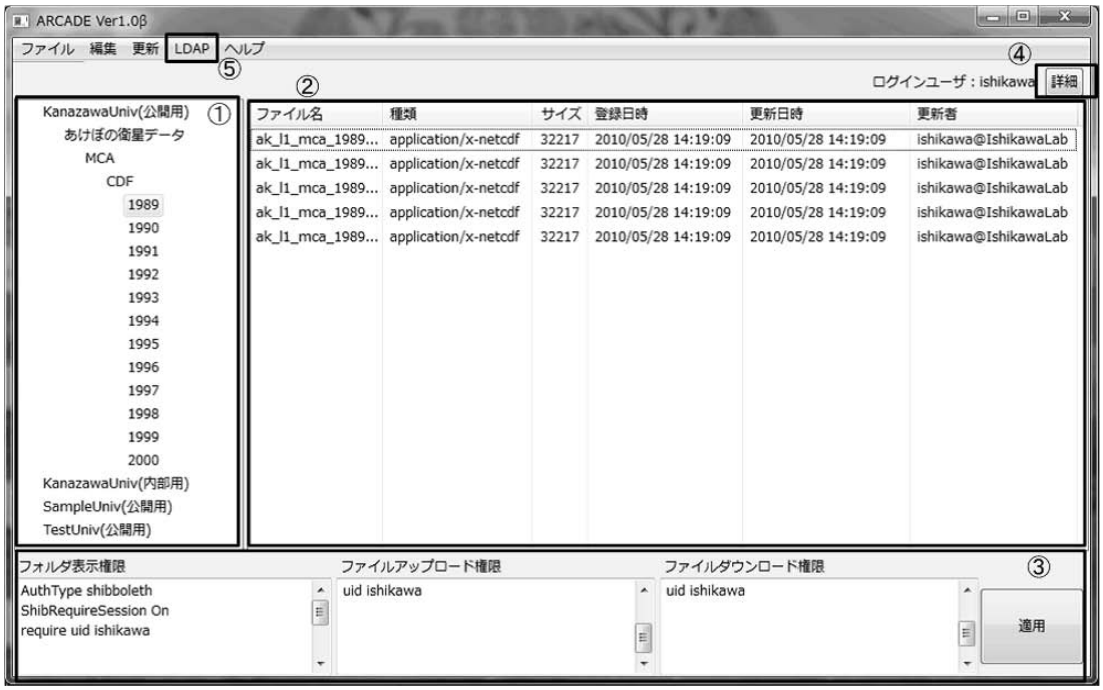

図 5-6 ARCADE 画面

5.3.4.2.2　ファイル情報表示

本節は，ファイル表示（②）の機能について説明する．5.3.4.2.1 節において，ユーザがディレクトリを選択すると，ディレクトリ内のファイル情報の一覧を表示する．ファイル情報として，「ファイル名」，「種類」，「サイズ」，「登録日時」，「更新日時」，「更新者」を持つ．なお，ユーザは，後述するファイルアップロード権限またはファイルダウンロード権限を持つ場合は，ドラッグアンドドロップでファイルを操作できるように配慮した．

5.3.4.2.3　権限設定

本節は，権限設定（③）の機能について説明する．権限については，「フォルダ参照権限」，「ファイルアップロード権限」，「ファイルダウンロード権限」の 3 種類を設定することができる．また，権限はフォルダ単位で行うことができる．

「フォルダ参照権限」は，ユーザに対して，ディレクトリを表示させるかどうか設定を行う．

「ファイルアップロード権限」および「ファイルアップロード権限」は，フォルダ上にあるファイルをアップロードしたり，ダウンロードしたりできるように制限を行う．権限の指定方法はフォルダ表示権限と同様，LDAP の属性値を用いて行う．但し，アップロードおよびダウンロードの機能の制限は，Apache で行うことが困難なため，フォルダ参照権限と同様の設定方法で行えるように，ARCADE 独自に実装している．なお，ユーザの属性値については，最初に管理サーバにアクセスし，自組織の IdP での認証が済んだ後に，IdP から SP としての管理サーバにユーザの属性値が送られる．その属性値を ARCADE が受信することで，この情報を基に，ファイルのアップロードおよびダウンロードを制限する．また，権限の設定を変更可能なユーザは，これらの 3 つの権限が許可されているユーザのみとし，権限設定の誤操作を防いでいる．

5.3.4.2.4　ユーザ情報閲覧

本節は，ユーザ情報閲覧（④）の機能について説明する．5.3.4.2.3 節でのユーザのアクセス制限を設定するためには, 自分の属性値を把握しておく必要がある．そこで,「詳細」ボタンをクリックすることで，表 5-1 と同様の画面を表示し，ユーザが自分の属性を確認できるように設計を行った．ユーザは，自分の属性を確

認し，データ公開者に対して自分の属性を伝え，その属性値を権限に追加してもらうことで，データへのアクセスが可能になる．なお，description に関しては，ユーザが自由に変更できるようにし，ユーザ間で独自の属性値でやり取りができるように配慮した．

5.3.4.2.5　ユーザ管理

本節は，ユーザ管理（⑤）の機能について説明する．LDAP の属性値の管理者フラグである title に 1 がセットされたユーザは，自組織のユーザ情報の追加，変更，削除を行うことができる．自組織の LDAP サーバを判断するために，ユーザが認証を行った IdP サーバの URL を ARCADE で保持している. そして, その URL の IdP に対して，SOAP（Simple Object Access Protocol）[52]を用いて，LDAP に対して，ユーザの追加・変更・削除のコマンドを呼び出す．SOAP を用いることで，LDAP へのアクセスを 443 ポートで呼び出すことが可能なため，ARCADE は全ての操作を 443 ポートで行うことができる．このように，ユーザの管理においても全て GUI 上で作業を可能とし，ユーザの操作性を保っている．

5.4 実証運用

本節では，5.2 節，5.3 節で説明した ARCADE において，実証運用を行った結果に述べる．

5.4.1　共有例

共有例として，図 5-2 に示す環境を実際に構築し，実証運用を行った．実証運用概要は以下のとおりである．

- Kanazawa University のユーザ ishikawa が Sample Univetsity の sample01 に対して，あけぼの衛星の実験観測データの一部を公開
- Test University のユーザ test01 に対してはデータの存在も含め，一切公開を行わない
- あけぼの衛星のデータの 1989 年, 1990 年, 2000 年のデータのみを sample01 に対してダウンロードのみを許可する

5.4.2　実証運用結果

KanazawaUniversity のユーザ ishikawa は, ARCADE にログインし, 1989, 1990, 2000 年のフォルダに対してアクセス制限を行う. ishikawa は sample01 の属性値の 1 つのメールアドレス「sample01@sample-u.ac.jp」をフォルダアクセス権限，ファイルダウンロード権限に追加する.

上記の状態で ishikawa，sample01，test01 がそれぞれ ARCADE にログインした際のフォルダ階層の状態を図 5-7 に示す．ishikawa は 1989~2000 年全てのフォルダを参照可能で，権限の設定を行うことができる状態になっている．sample01 は 1989，1990，2000 年のフォルダを参照可能で，ファイルのダウンロードのみ可能な状態になっている. test01 は, フォルダの参照が不可能になっている. sample01, test01 ともに，フォルダの権限設定部分については設定を変更することができない.

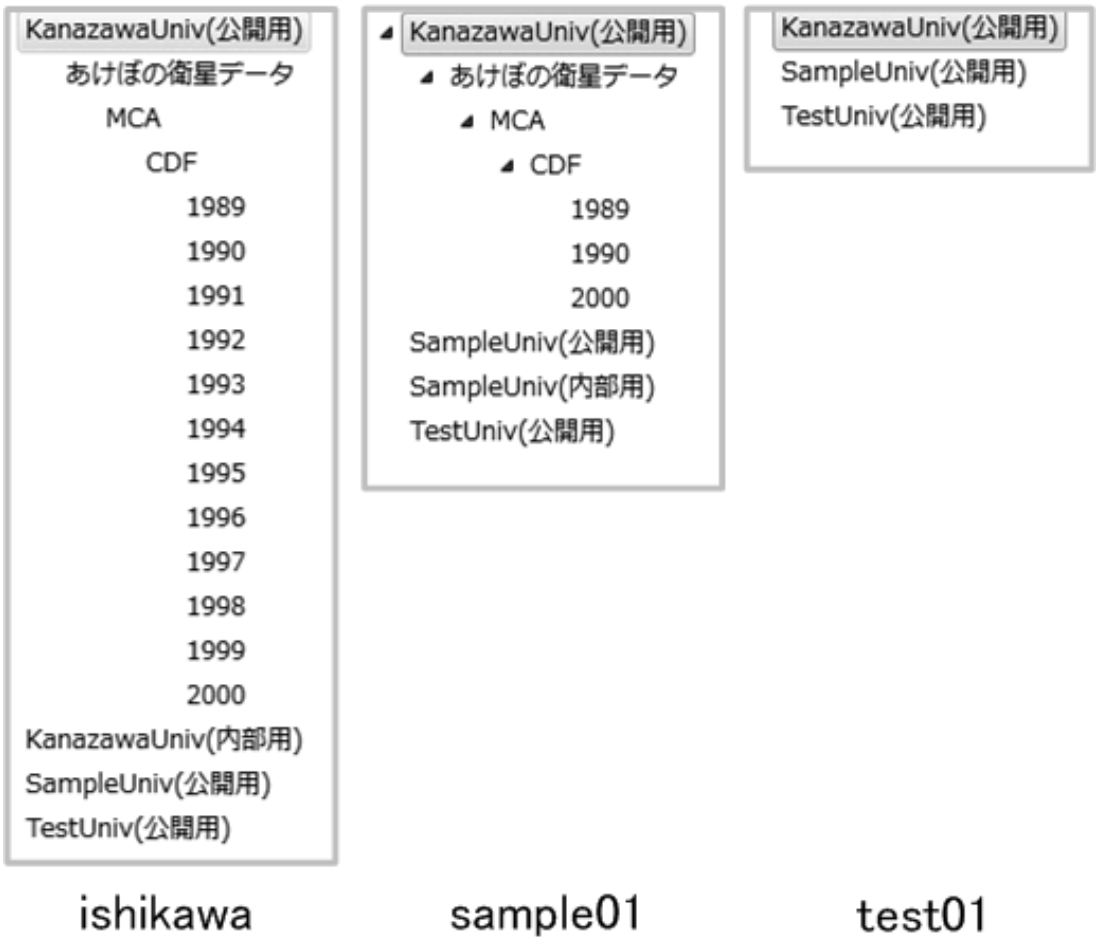

図 5-7 ユーザのアクセス制限に応じたディレクトリ参照状態

5.5 まとめ

本研究において，ARCADE を開発したことにより，組織を超えてデータを共有する際のデータ共有相手を特定でき，公開者ごとに様々に異なるデータの公開ポリシをシンプルに設定できるようになった．また，開発環境に必要なソフトウェアは全てオープンソースを用いたことで，小規模な研究室においてもコスト的に問題なく導入できる．さらに，ARCADE は GUI で簡単に操作可能であり，理系，文系問わず，様々な研究組織で容易に利用可能であると考えている．そのため，複数の組織にまたがる研究者間のデータ公開の促進と，異領域間でのデータ相互参照が進み，研究の発展が期待できる．

今後は，地球惑星科学分野での利用が予定されており，実運用を進めていきながら，さらなるユーザビリティの向上に努めていきたいと考えている．また，Gakunin 上で利用することにより，実験観測データに留まらず，教材データや業務データなど，様々な分野における安全・安心なデータ共有へと発展していくことを期待する．

第6章　開発システムの連携

6.1 開発システムのそれぞれの位置付け

3 章から 5 章において，2.5 節で述べた開発システムにおける認証連携の範囲に基づき，統合認証基盤を構築し，安全なデータ共有ができることを実証した．本研究で開発したシステムのうち，金沢大学内での開発システム概念図を図 6-1 に示す．このように，大学間認証連携においては，GakuNin を利用して，ID としてはネットワーク ID を用いる．また本学内統合認証基盤においては，金沢大学 ID を利用する．そして，研究室間認証連携においては研究室が発行する ID を利用する．

6.2 各組織での IdP および SP の集約

6.2.1　集約方法

6.1 節で示した通り，現在，本学においてはネットワーク ID，金沢大学 ID および研究室用 ID の 3 つを使い分ける必要がある．それは各範囲においてそれぞれ IdP を構築していることに起因する．また，SP においても GakuNin 用ファイル送信サービスと学内用ファイル送信サービスがあり，同一の機能を提供する場合であっても，別途サーバを構築している．それは，GakuNin と本学内ではそれぞれを利用できるユーザの範囲が異なるためである．利用範囲が異なる IdP および SP を同一フェデレーションとして運用するのはセキュリティレベルの低下を招くため好ましくない．しかし，サーバの台数が増えることは管理者の負担の増大および利用者の利便性の低下を招く恐れがある．

そこで，ユーザの利用範囲を考慮しながらも IdP および SP の集約化を行う必要があると考える．

6.2.2 集約例

6.2.2.1 IdP の集約

6.1 節で述べたとおり，現在組織としては，大学間，学内，研究室間と 3 つの分

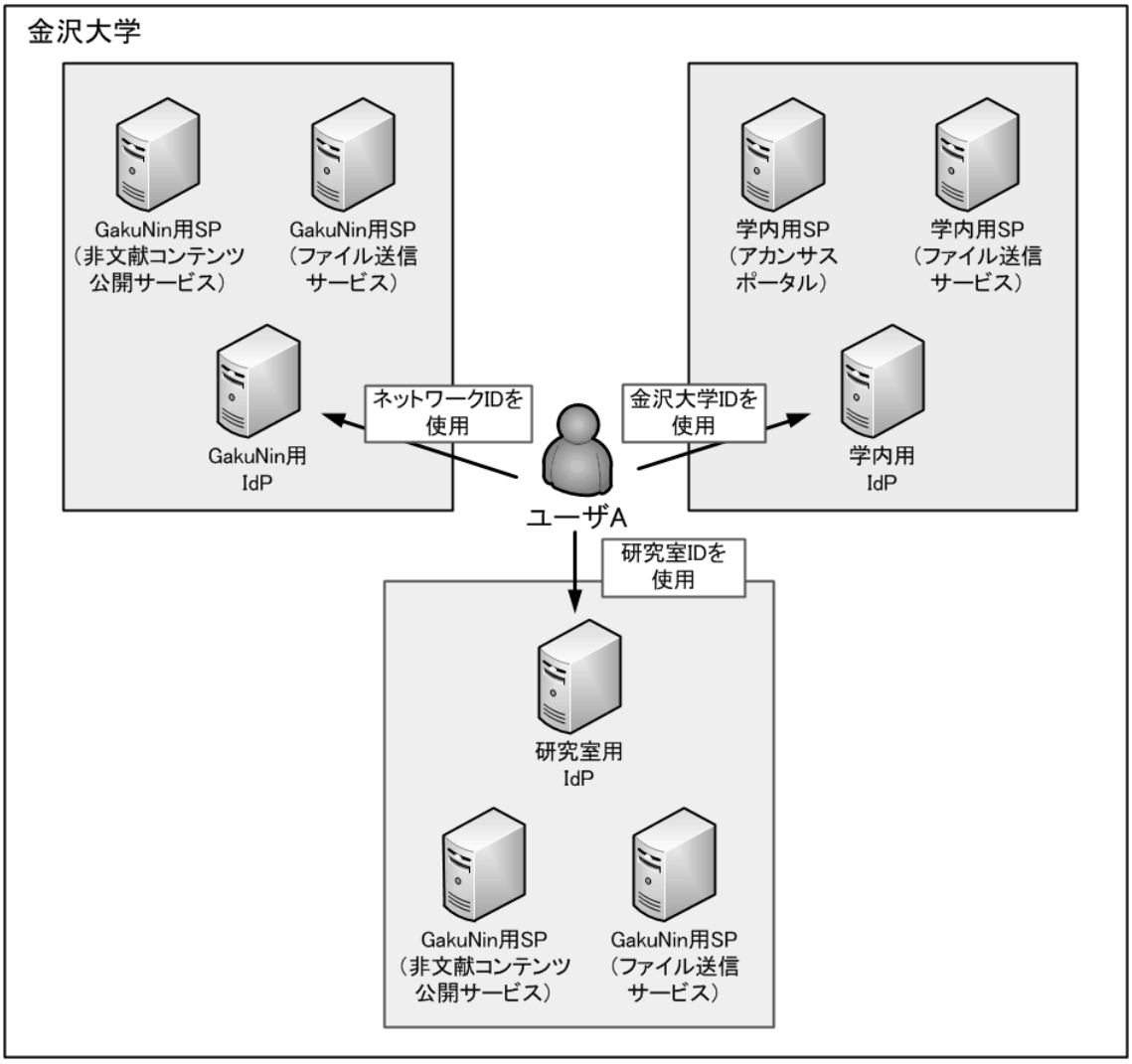

図 6-1 開発システム概念図

類をして，それぞれに対して Shibboleth 環境を独立して構築している．

その中で，大学間と研究室間は「金沢大学外」の SP を利用することを視野に入れて構築を行っている．そこで，GakuNin 用 IdP と研究室用 IdP を統合し，GakuNin 用 IdP を研究室間認証連携でも利用するという方法が考えられる．GakuNin の IdP は GakuNin で認定を受なければフェデレーションに接続できないため，研究室で独自で構築するよりも認証の信頼性が高い．また，研究室で IdP を運用する必要がなくなるため，研究室のサーバ管理者の運用コストを抑えることも可能になる. 但し, 研究室間でやり取りするための属性情報を GakuNin 用 IdP ですべて賄うことはできないため，研究室側で別途 GakuNin 用 IdP と紐づけて属性情報を持つ必要性が出てくる．

一方で，GakuNin 用 IdP と金沢大学内 IdP は利用ユーザの範囲が異なるため，統合することは困難であり，分けておくことが望ましいと考えられる．

6.2.2.2 SP の集約

GakuNin の考え方から，将来的には大学内情報システムを学外のユーザに提供を行う環境を想定する必要がある．図 6-1 に示すとおり，現在，ファイル送信サービスは GakuNin 用と金沢大学内用の SP をそれぞれ別に構築している. しかし，別に構築することはサーバ自身のコストと管理者の運用コストがかかる．

そこで，同一のサービスを提供する SP を統合し，GakuNin 利用時でも学内のファイル送信サービスを利用するという方法が考えられる．そのことで，SP の運用コストを抑えることができる．但し，SP 側で GakuNin からのアクセスか本学内からのアクセスかを IdP から送付される属性から判断し，属性に応じたサービスを提供するように SP 側での実装が必要になると考えられる．

6.2.2.3 DS の配置

6.2.2.1 に示す IdP の集約により，本学のユーザは学内用 IdP と GakuNin 用 IdP の 2 つの認証を使い分ける必要がある．本学統合認証基盤内に DS を適切に配置することにより，本学ユーザの利便性が向上すると考えられる．

6.3 最終的な開発システム連携構成

6.2 章で述べた IdP および SP を集約した将来的な開発システム連携構成図を図 6-2 に示す.

このように，IdP は「学内用」と GakuNin 用および研究室用を集約した「学外用」の 2 つにする．学内用 SP のアカンサスポータルにアクセスした場合は，学内用の IdP で認証を行う. GakuNin 用 SP の非文献コンテンツ公開サービスにアクセスした場合は GakuNin 用 IdP で認証を行う．そして，ファイル送信サービスの

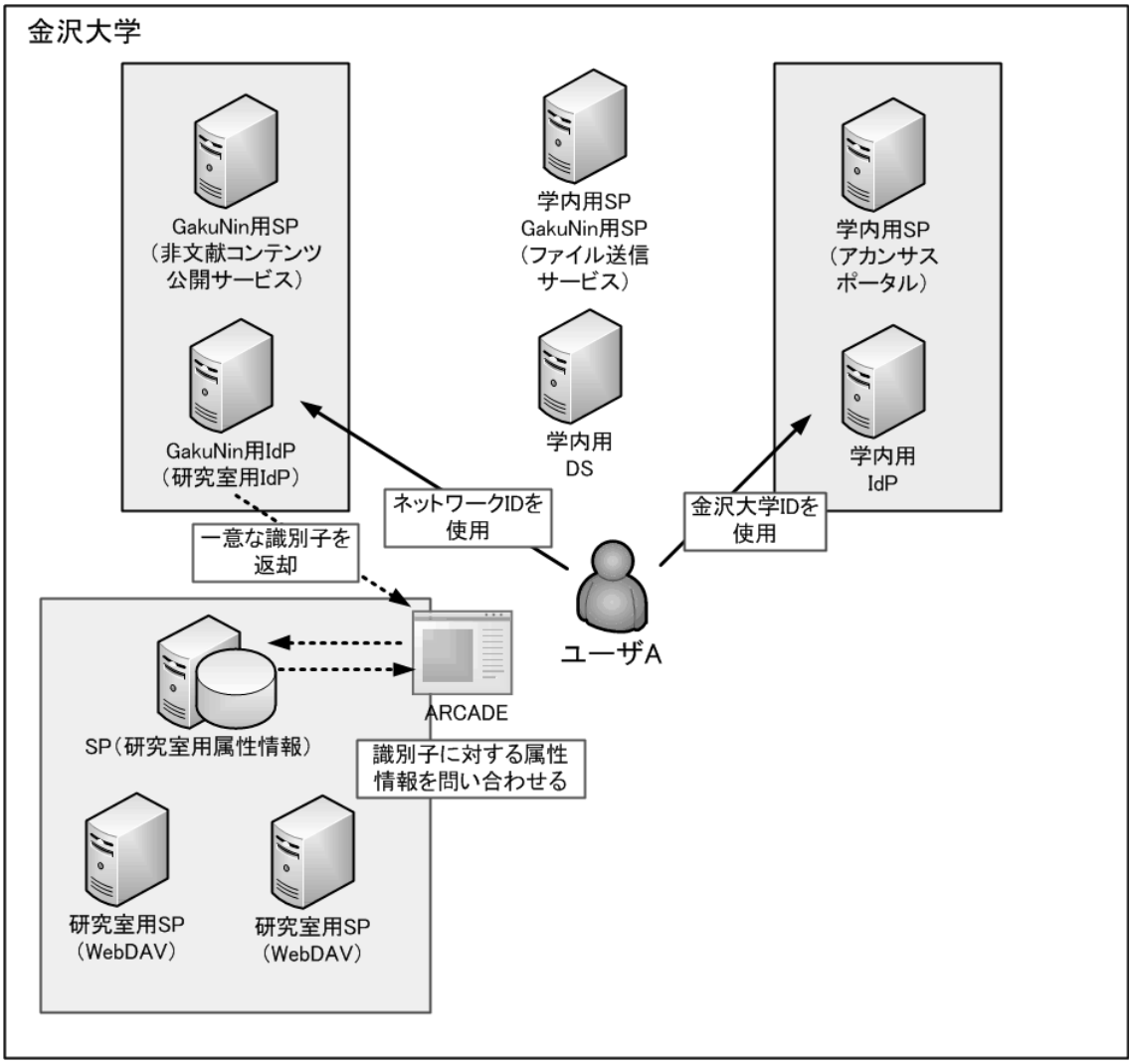

図 6-2 将来的なシステム連携概念図

ように，学内と学外両方にサービスを提供する SP は 1 つに集約する．一般的に学内からのアクセスはアカンサスポータルを経由するため, 認証が不要と考える．一方で学外からのアクセスは SP に直接アクセスするため, 学内用 DS を新規に構築し，学内ユーザなら学内用 IdP，学外ユーザの場合は GakuNin の DS を選択するように設計を行う．また，研究室用 IdP を GakuNin 用 IdP に統合したことで不足する属性情報は，各研究室に研究室用属性情報をもつデータベースを SP として用意し，アプリケーション側（ARCADE）で吸収する．具体的には，GakuNin 用 IdP で認証を行うと, ARCADE に対してユーザ固有の一意な識別子が送付される．ARCADE には研究室単位で管理している属性情報を保持している SP を登録しておく．この SP は研究室用 IdP として利用していた LDAP を参照できるシステムとする．LDAP 情報の内，uid 属性の値を，研究室 ID から，IdP から送付される識別子に変更を行う．そして，ARCADE は登録された研究室用属性情報 SP に対して与えられた識別子に対する属性情報を問い合わせる．そして一致した属性情報を取得し，ARCADE 内に一時的に記憶させておく．そうすることで，認証部分を GakuNin の IdP で行い，属性共有を ARCADE に吸収することが可能になると考える．

以上のように，IdP および SP を集約し，DS を新規に配置することにより，さらに効率的に組織間連携が行えると考えられる．

第7章　結論

本研究では，GakuNin における大学間組織連携および研究室などの小規模組織間連携を視野に入れた，本学における情報システムを一元的に取り扱うことを可能にする，大学特有の多様なシステム管理形態に柔軟に対応可能な統合認証基盤の構築を行った．そして，本学の統合認証基盤を構築するにあたり，本学だけの作り込みのシステムとはせず,「金沢大学モデル」として，他大学においても転用可能なモデルケースとなりうるレベルでの構築に配慮した. さらに, 構築した「金沢大学モデル」を，大学 ICT 推進協議会に対してモデルケースを示すことで，本学の統合認証基盤が大学の統合認証基盤の雛形になることを目指した.

1 章では，大学における情報システム環境の現状について述べるとともに，本学の情報システムの一元化を達成するための課題とそれを達成するための問題点の洗い出しを行った．そして，他大学で先行的に導入されている事例を調査し，そこで利用されているシングルサインオンおよび属性共有の技術である CAS, PKI および Shibboleth についてそれぞれの特徴について議論した．その結果，PKI によるクライアント認証方式は費用面および技術面において本学の統合認証基盤には適さないと判断し，CAS または Shibboleth での開発を決めた．そして，名古屋大学での導入実績，費用面，技術面において一番導入が容易であると判断した CAS を実際に本学の情報システムに適用し，運用を行ったうえで CAS か Shibboleth の選定を行うようにした.

2 章では, 1 章で述べた大学における統合認証基盤の先行研究の中で本学の統合認証基盤技術として，費用面および技術面において一番導入が容易であると判断した CAS を実際に金沢大学内の情報システムに適用し，CAS の動作について議論を行った．そして，CAS においては，CAS サーバへのアクセス制限がない，CAS クライアントに応じた適切な属性情報の送信方法がない，セッション情報を

URL パラメータに記載しなければならない，クロスドメインシングルサインオンに未対応であるという問題点を明らかにした．そして Shibboleth においてはこれらの問題をクリアしていることを確認し，本学における統合認証基盤技術として，Shibboleth を選定した．さらに，本研究における開発システム認証連携基盤の範囲について，大学間認証連携基盤，大学内統合認証基盤および小規模組織間認証連携基盤と 3 つに分類し，3〜5 章でそれぞれの認証連携について議論した．

3 章では，本学の統合認証基盤を構築するうえで視野に入れている大学間認証連携基盤について取り扱った．NII が推進する大学間フェデレーションである GakuNin を利用して，大学という組織を越えて情報システムを利用する際のユーザ属性情報の取り扱いについて議論した．さらに，GakuNin フェデレーション内に SP として，大学間における安全なデータ共有を目的とした「GakuNin を用いたファイル送信サービス」および「DSpace によるデジタルコンテンツ公開サービス」を構築し，大学間において，本学で他大学の利用者情報を管理する必要なしに，他大学の利用者の身元を保証したうえで，安全に本学の情報システムを提供することが可能な仕組みを構築した．さらに，今回構築した SP における認可の方法及び GakuNin で利用する ID の秘匿性について考察を行い，個人情報を保護しながらも，適切に認可を行う方法を示すことができ，セキュリティ面においても検証を行った．

4 章では，3 章で得られた Shibboleth の知見を用いて本学内統合認証基盤の設計，実装，システム評価について議論した．Shibboleth を導入することにより，これまでは独立していた各種情報システムにおける認証機構の一元化を実現した．また，金沢大学 ID という生涯 ID を導入するとともに，LDAP スキーマの設計に配慮し，ShibbolethIdP の属性情報の送信形式を利用することで，教員・職員・学生など各自の職分も認識でき，その職分に応じて利用許可されている情報システムが再度の認証なしに利用可能とした．そして Shibboleth では未実装のシングルログアウトを独自に開発し，セキュリティレベルを向上することができた．さらに，構築した統合認証基盤の評価を行い，実運用にも十分耐えうるシステムであることを実証した．また，大学特有の複雑な所属形態にも対応し，将来的に大学間連携にも活用可能な統合認証基盤を構築した．本学のように Shibboleth を研究的な側面で利用するのではなく，教育と業務に拡大し，日常的に利用されている情報

システムに適用しているのは日本の大学では例がなく，さらに，IdP や SP の冗長化を行い，可用性を高めている点では，海外の大学でもほとんど見られず，国内外問わず本学の統合認証基盤は研究成果としても波及効果は非常に大きいと考えている．

5 章では，本学における統合認証基盤構築の内，研究室などの小規模組織間認証連携基盤について取り扱った．Shibboleth を利用し，研究室間でデータを共有する際に，どのようにしてデータ共有相手を特定するかについて議論し，多様な公開ポリシに容易に対応可能なデータ相互参照システムである ARCADE を開発し，動作を検証した．ARCADE を開発したことにより，データ共有相手を特定でき，公開者ごとに様々に異なるデータの公開ポリシをシンプルに設定できるようになった．また，開発環境に必要なソフトウェアは全てオープンソースを用いたことで，小規模な研究室においてもコスト的に問題なく導入できるようにした．さらに，ARCADE は GUI で簡単に操作可能であり，理系，文系問わず，様々な研究組織で容易に利用可能であると考えており，複数の組織にまたがる研究者間のデータ公開の促進と，異領域間でのデータ相互参照が進み，研究の発展が期待できる．

6 章では，3~5 章で説明した開発システムの連携について議論した．それぞれの認証基盤におけるユーザの利用範囲を考慮し，大学間認証連携と小規模組織間認証連携における IdP の集約を行い，GakuNin 上で大学間および小規模組織間両方の認証連携基盤として動作するように検討を行った. そのことで, 研究室で IdP を構築する手間が省略できるとともに，GakuNin フェデレーションの認定を受けるため，認証における信頼度が向上できるようになると考えられる．そして，本学統合認証基盤内に DS を適切に配置し，GakuNin フェデレーションと本学統合認証基盤のどちらを用いるのか利用者に委ねる方式を採用することで，ユーザビリティの向上につながると考えている．さらに，本学の情報システムを，アクセスを受ける IdP に応じたサービスを提供できるような仕組みについても検討を行った．以上のことから，大学間，大学内，研究室間において，さらに柔軟な連携を行うことができると考えている．

以上の研究結果から分かるように，本研究で提案した本学における組織間連携

を視野に入れた統合認証基盤は共通プラットフォームとして,本学だけに限らず,他大学でも十分転用可能なモデルケースとして十分に提供することができると考えている．また，本研究における成果は，国内の大学だけではなく，海外の大学においても，今後情報システムの一元化を検討している大学関係者の一助となることを確信している．

本研究は今後,「金沢大学モデル」として，大学情報システムにおける，大規模で，柔軟性に富んだ統合認証システム普及への発展に寄与するものであると考えている．

謝辞

本研究を進めるにあたり，懇切なる御指導，御鞭撻を賜りました主任指導教員である金沢大学大学院自然科学研究科笠原禎也教授に深く感謝いたします. また，大変きめ細かな御指導，御助言を賜りました金沢大学大学院自然科学研究科村本健一郎教授，八木谷聡教授，佐藤賢二准教授，平野晃宏講師に深く感謝いたします.

本研究を進めるうえで，有意義な御助言を頂きました国立情報学研究所の中村素典教授，山地一禎准教授に深く感謝いたします.

また，本研究に関わる環境構築を御支援して頂き，また，貴重な御助言と有益な討論をして頂きました金沢大学総合メディア基盤センターの同僚でもあり，通信情報工学研究室（笠原研）の先輩でもある金沢大学総合メディア基盤センター高田良宏助教に深く感謝いたします.

そして，本研究を進めるにあたり，研究開発に関して有意義な御助言を頂きました金沢大学総合メディア基盤センター東昭孝特任助教，二木恵氏，松原志野氏に深く感謝いたします.

本研究は，筆者が金沢大学総合メディア基盤センターにおける研究開発業務と並行して行ったものであり，在職中の修学機会を与えて頂きました歴代の金沢大学総合メディア基盤センター長の方々に深く感謝いたします．また，業務と並行して博士後期課程で研究を行うことに御理解を頂き，さらに，大変暖かい御支援と御協力，励ましを頂きました金沢大学総合メディア基盤センターの教職員の皆様に深く感謝いたします.

また，多くの有益な討論を行い，御助言を頂きました金沢大学大学院自然科学研究科後藤由貴助教，通信情報工学研究室（笠原研）の修了生，卒業生，および，在学生の皆様に深く感謝いたします.

本研究は，以上の方々をはじめとする多くの方々の御理解と御支援のもと達成できたものであり，本研究に関わってくださった全ての方々に謹んで御礼申し上

げます.

最後に，勤務と学業の両立を全面的に支援してくれた妻 悠子と娘 佳歩に最大級の感謝をして本論文を結びます.

参考文献

[1] 高橋 健司: 「アイデンティティ管理の現状と今後」, 電子情報通信学会誌, Vol.92, No 4, pp.287-294, (2009)

[2] 内藤 久資, 梶田 将司, 小尻 智子, 平野 靖, 間瀬 健二: 「大学における統一認証基盤としての CAS とその拡張」,情報処理学会論文誌, Vol. 47, No. 4, pp.1127-1135 (2006).

[3] 梶田 将司, 内藤 久資, 小尻 智子, 平野 靖, 間瀬 健二: 「CAS によるセキュアな全学認証基盤の構築」, 情報処理学会研究報告, Vol.2005, No.39, pp.35-40 (2005).

[4] 秋山 豊和, 寺西 裕一, 岡村 真吾, 坂根 栄作, 長谷川 剛 ほか 4 名: 「大阪大学における全学 IT 認証基盤の構築」, 情報処理学会論文誌, Vol. 49, No. 3, pp.1249-1264 (2008).

[5] 飯田 勝吉, 新里 卓史, 伊藤 利哉, 渡辺 治: 「キャンパス共通認証認可システムの構築と運用」, 電子情報通信学会論文誌, J92-B(10), pp.1554-1565, (2009)

[6] 金西 計英, 松浦 健次, 大家 隆弘: 「徳島大学におけるポータルシステムの構成とその運用について」, 大学情報システム環境研究, Vol.12, pp.34-42 (2009)

[7] 大谷 誠, 江藤 博文, 渡辺 健次, 只木 進一, 渡辺 義明: 「シングルサインオンに対応したネットワーク利用者認証システムの開発」, 情報処理学会論文誌, Vol. 51, No. 3, pp.1031-1039 (2010).

[8] UPKI イニシアティブ: https://upki-portal.nii.ac.jp/ (accessed 2010.12)

[9] GakuNin: https://www.gakunin.jp/docs/fed (accessed 2010.12)

[10] Shibboleth Federations: https://spaces.internet2.edu/display/SHIB/ShibbolethFederations (accessed 2010.12)

[11] EDUCAUSE: http://www.educause.edu/ (accessed 2010.12)

[12] 一般社団法人大学ICT推進協議会: http://www.ipe.media.kyoto-u.ac.jp/axies-prep/?%C0%DF%CE%A9%C1%ED%B2%F1 (accessed 2010.12)

[13] 松平 拓也, 笠原 禎也, 高田 良宏, 井町 智彦:「UPKI認証連携基盤に基づく安全なデータ共有システム構築の試み」, 学術情報処理研究, No13, pp.84-90(2009).

[14] Central Authentication Service: http://www.jasig.org/cas (accessed 2010.12)

[15] Public Key Infrastructure: http://www.ipa.go.jp/security/pki/index.html (accessed 2010.12)

[16] Shibboleth: http://shibboleth.internet2.edu/ (accessed 2010.12)

[17] Java Architecture Special Interest Group: http://www.jasig.org/ (accessed 2010.12)

[18] A beginners guide to Dependency Injection: http://www.theserverside.com/news/1321158/A-beginners-guide-to-Dependency-Injection (accessed 2010.12)

[19] Spring Framework: http://www.springsource.org/ (accessed 2010.12)

[20] Apache Http Server Project: http://httpd.apache.org/ (accessed 2010.12)

[21] PHP: http://www.php.gr.jp/ (accessed 2010.12)

[22] Internet2: http://www.internet2.edu/ (accessed 2010.12)

[23] MACE: http://middleware.internet2.edu/MACE/ (accessed 2010.12)

[24] SAML2.0: http://www.oasis-open.org/specs/index.php (accessed 2010.12)

[25] 金沢大学総合メディア基盤センター: http://www.imc.kanazawa-u.ac.jp/ (accessed 2010.12)

[26] Fujitsu: “Interstage Application Server”, http://interstage.fujitsu.com/jp/apserver/

(accessed 2010.12)

[27] Sun Microsystems: “Sun Java System Identity Manager”, http://blogs.sun.com/edu/entry/sun_java_system_access_manager

[28] Sun Microsystems : “Sun Java Directory Server” http://blogs.sun.com/edu/entry/sun_java_system_directory_server

[29] 松平 拓也, 車古 正樹, 井町 智彦: 「spam メール及びウイルスメール対策システムの構築と運用」, 学術情報処理研究, No9, pp.45-54 (2005).

[30] DSpace: http://www.dspace.org/ (accessed 2010.12)

[31] 高田 良宏, 笠原 禎也, 西澤 滋人, 森 雅秀, 内島 秀樹: 「非文献コンテンツのための可視性と保守性に優れた学術情報リポジトリの構築」,情報知識学会誌, Vol.19, No.3, pp.251-263 (2009)

[32] eduPerson & eduOrg Object Classes: http://www.educause.edu/eduperson/ (accessed 2010.12)

[33] 金沢大学情報戦略本部: http://www.imc.kanazawa-u.ac.jp/info/publication/kouhou2008.pdf (accessed 2010.12)

[34] 只木 進一, 江藤 博文, 渡辺 健次, 渡辺 義明: 「利用者移動端末に対応した大規模ネットワークの Opengate による構築と運用」, 情報処理学会論文誌, Vol. 46, No. 4, pp.922 -929 (2005)

[35] 太田芳博, 梶田将司, 田島嘉則, 田島尚徳, 平野靖, 内藤久資, 間瀬健二: 「大学における生涯 ID のための名寄せ手法」, 情報処理学会論文誌, Vol. 51, No. 3, pp.965 -973 (2010).

[36] 松本豊司, 鈴木恒雄, 佐藤正英, 堀井祐介, 井町智彦: 「e-Learning の全学展開を考慮した情報処理基礎教育システムの構築」, 教育システム情報学会誌, Vol. 25, No 1, pp.87-99(2008).

[37] OpenPNE: http://www.openpne.jp/ (accessed 2010.12)

[38] OpenLDAP: http://www.openldap.org/ (accessed 2010.12)

[39] Terracotta: http://www.terracotta.org/ (accessed 2010.12)

[40] Shibboleth 2 Documentation: https://spaces.internet2.edu/display/SHIB2/IdPClusterIssues (accessed 2010.12)

[41] 国立情報学研究所(NII): ”学術機関リポジトリ構築連携支援事業”, http://www.nii.ac.jp/irp/ (accessed 2010.12)

[42] 金沢大学学術情報リポジトリ(KURA): http://dspace.lib.kanazawa-u.ac.jp/dspace/ (accessed 2010.12)

[43] 橋 洋平: 「金沢大学学術情報リポジトリ KURA の構築と課題」, 大学図書館研究, Vol79, pp.18-26 (2007)

[44] JVO: http://jvo.nao.ac.jp/ (accessed 2010.12)

[45] 村田健史, 岡田雅樹, 阿部文雄, 荒木徹, 松本紘: 「太陽地球系物理観測の分散メタデータベースの設計と評価」, 情報処理学会, Vol.43, No.SIG 12(TOD 16), pp.115-130 (2002).

[46] 木村映善, 村田健史: 「RSS/RDF を利用した太陽地球系物理観測データのメタデータ配信の検討」, 情報処理学会, Vol.47, No.4, pp.1051-1062 (2006)

[47] IUGONET プロジェクト: http://www.iugonet.org/ (accessed 2010.12)

[48] WebDAV: http://www.webdav.org/ (accessed 2010.12)

[49] SWT: http://www.eclipse.org/swt/ (accessed 2010.12)

[50] Eclipse: http://www.eclipse.org/ (accessed 2010.12)

[51] Java Web Start: http://java.sun.com/javase/technologies/desktop/javawebstart/index.jsp (accessed 2010.12)

[52] SOAP: http://www.w3.org/TR/#tr_SOAP (accessed 2010.12)

研究業績

参考論文

(1) 松平 拓也, 車古 正樹, 笠原 禎也, 高田 良宏, 井町 智彦, “資産管理に基づく適切なソフトウェア配布システムの構築”, 学術情報処理研究, No.12, pp.13-19, September 2008.

(2) 松平 拓也, 笠原 禎也, 高田 良宏, 井町 智彦, “UPKI 認証連携基盤に基づく安全なデータ共有システム構築の試み”, 学術情報処理研究, No.13, pp.84-90, September 2009.

(3) Takuya Matsuhira, Yoshiya Kasahara, Yoshihiro Takata, "Development of a file-sharing system for educational collaboration among higher-education institutions", INTERNATIONAL JOURNAL OF EDUCATION AND INFORMATION TECHNOLOGIES, Issue 2, Volume 5, pp.149-156, January 2011.

(4) 松平 拓也, 笠原 禎也, 高田 良宏, 東 昭孝, 二木 恵, 森 祥寛, “大学における Shibboleth を利用した統合認証基盤の構築”, 情報処理学会論文誌, Vol.52 No.2, pp.703-713, February 2011.

副論文

(1) 井町 智彦, 松本 豊司, 車古 正樹, 松平 拓也, 鈴木 恒雄, “金沢大学認証無線ネットワークシステムの構築と評価”, 学術情報処理研究, No9, pp.5-13, September 2005.

(2) 松平 拓也, 車古 正樹, 井町 智彦, “spam メール及びウイルスメール対策システムの構築と運用”, 学術情報処理研究, No9, pp.45-54, September 2005.

(3) 車古 正樹, 松平 拓也, 井町 智彦, 中野 三智子, “spam フィルタに関する統計”, 学術情報処理研究, No9, pp.55-62, September 2005.
(4) 松平 拓也, 車古 正樹, 井町 智彦, “SpamAssassin による spam メール認識率に関する統計”, 大学情報システム環境研究, vol.10, pp.40-49, November 2006.
(5) 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, September 2007.
(6) 松平 拓也, 車古 正樹, 井町 智彦, 高田 良宏, “spam メール対策の多段化における効果, 学術情報処理研究”, No.11, pp.60-67, September 2007.

国際会議

(1) Yoshihiro Takata, Yoshiya Kasahara, Nobuhiro Mouri and Takuya Matsuhira, “Development of Science Database System Applicable to Various Access Restrictions”, International Symposium: Fifty Years after IGY - Modern Information Technologies and Earth and Solar Sciences -, 45, Tsukuba, Japan, 10 November, 2008.
(2) Takuya Matsuhira, Yoshiya Kasahara, Yoshihiro Takata, "ARchive System for Cross-Reference Across Distributed Environment (ARCADE) Applicable to Sharing of Educational Materials among Inter-University Consortium", 9th WSEAS International Conference on Education and Educational Technology (EDU '10), pp.167-170, Iwate, Japan, 6 October, 2010.

学会・研究会報告

(1) 松平 拓也, 井町 智彦, 笠原 禎也, 高田 良宏, “金沢大学における shibboleth 構築と SP 実装例”, シングルサインオン実証実験中間報告会, 口頭発表, 国立情報学研究所, 10 November, 2008.
(2) 松平 拓也, 井町 智彦, 笠原 禎也, 高田 良宏, “金沢大学における大学間

連携への取り組み～UPKI 仕様に基づく Shibboleth を用いた IdP 及び SP 構築～”, UPKI シンポジウム 2009, ポスターセッション, 国立情報学研究所, 23 February, 2009.

(3) 松平 拓也, 井町 智彦, 笠原 禎也, 高田 良宏, “UPKI における金沢大学の取り組み ―学内認証基盤から大学間連携まで―”, 第 8 回金沢大学データベースフォーラム, 口頭発表, 金沢大学, 19 March, 2009.

(4) 松平 拓也, 笠原 禎也, 高田 良宏, 井町 智彦, “金沢大学における IdP, SP 構築の現状と今後の展望”, 平成 20 年度 シングルサインオン実証実験報告書：国立情報学研究所, pp.69-80, 2009.

(5) 松平 拓也, 笠原 禎也, 高田 良宏, “安全・安心な大学間情報共有を実現するサービスプロバイダの構築と今後の展開”, UPKI シンポジウム 2010, ポスターセッション, 国立情報学研究所, 12 March, 2010.

(6) 松平 拓也, 笠原 禎也, 高田 良宏, “Shibboleth をベースとした異領域間でのデータ相互参照システムの開発”, 日本地球惑星科学連合 2010 年大会, 予稿 CD-ROM, 幕張メッセ国際会議場, 24 May, 2010.

(7) 松平 拓也, 笠原 禎也, 高田 良宏, 東 昭孝, 二木 恵, 森 祥寛, “Shibboleth によるポータルシステムを中心とした教育用情報システムの連携”, 平成 22 年度情報教育研究集会, 講演論文集, pp141-144, 京都テルサ, 11 December, 2010.

招待講演

(1) “UPKI に基づく金沢大学での Shibboleth を用いた IdP 及び SP 構築について”,第 9 回東海地区 CSI 事業報告会プログラム, 名古屋大学, 24 December, 2008.

(2) “学内の認証基盤整備と学内システムの Shibboleth 化”, 第 7 回国立大学法人情報系センター協議会分科会シンポジウム　世界をリードする最先端学術情報基盤 -学術認証フェデレーションの未来を探る-, 東京海洋大学, 15 July, 2010.

(3) “金沢大学における全学ポータルシステム「アカンサスポータル」の開発”, 大学 e ラーニング協議会　第 2 回公開フォーラムプログラム「先導的な ICT

活用の取り組みと大学連携」（e-Learning World2.0），東京ビッグサイト, 29 July, 2010.

(4) “大学間認証連携と金沢大学統合認証基盤構築における Shibboleth 利用”, 東海北陸地区技術職員研修（情報処理コース）「統合認証基盤と全学ポータルシステム」，金沢大学, 1 September, 2010.

(5) “Shibboleth による金沢大学統合認証基盤の構築と今後の展開”，第 4 回統合認証シンポジウム，佐賀大学, 22 December, 2010.

科学研究費補助金

(1) 松平　拓也, “大学間教育連携を実現する分散型教育情報共有システム”，科学研究費補助金 若手研究（B），課題番号:22700809, 交付額:2,800 千円 (2010-2012).