Vol.2019-CE-152 No.7
Vol.2019-CLE-29 No.7
2019/11/16

オーバーレイ思考による認証基盤の再構築

梶田 将司[1,2]

概要：クラウドコンピューティングの進展により，急速に進歩を続けるプラットフォーマが台頭し，低コストで高品質の外部システム・サービスが利用できるようになっている．しかも，今後は，「デジタルトランスフォーメーション」とも呼ばれる，多様な教育・研究・事務業務の中身に踏み込んだ ICT 支援も求められている．しかしながら，厳しい予算削減・人員削減が続く中，これまでのやり方では必要な財政的・人的双方の学内リソースは先細りしていく一方で，多様な外部環境・内部ニーズには応えきれない．特に，少ない予算で対応する以上，運用管理の内製化が必須であるが，テクノロジの進展の早さや人的リソースの再教育が容易ではないことを踏まえた戦略・戦術の検討が必要である．そこで，本稿では，オープンソース LMS Sakai の認証システムとして利用している CAS (Central Authentication Service) のバージョンアップを題材に，オーバレイ的な考え方による認証基盤の運用管理について議論する．

キーワード：オープンソースソフトウェア，認証基盤，アイデンティティマネジメント，多要素認証，デジタルトランスフォーメーション

Reconstructing Identity and Access Management Infrastrcuture Using An Overlay Way of Thinking

Shoji Kajita[1,2]

Abstract: The Cloud Computing has been rapidly evolving what we call platformers such as Google, Apple, Facebook and Amazon, and it enables the central IT at higher educational institution to use the latest Information and Communication Technologies with a modest cost. However, severe budget cuts and headcount cuts force us to provide more services with less finantial and human resourdes to react more diverse needs. To tackle these circumstances, we need to adopt new strategy and tactics to move to in-house productions and operations by using rapidly growing diverse and inexpensive outside services. This paper introduces an overay way of thinking based on our experiences on version-up of Sakai user authentication system using CAS (Central Authentication Service).

Keywords: Open Source Software, Authentication Infrastructure, Identity Management, Multi Factor Authentication, Academic Digital Transformation

1. はじめに

京都大学では，研究者が学術研究に伴う計算処理を行うための全国共同利用施設として大型計算機センターを 1969 年 4 月に設置して以来，ICT の組織的な利活用に関する先端的研究及びその京都大学における整備・運用を 50 年にわたり行ってきた．特に，その前半は，計算機の利用方法やプログラミング教育に特化した活動を，計算センター（1966 年 4 月），情報処理教育センター（1978 年 4 月），総合情報メディアセンター（1997 年 4 月），学術情報メディアセンター（2002 年 4 月）への改組を伴いながら行ってきた．

一方，インターネットやパーソナルコンピュータ，スマートフォン等の個人情報端末の普及により，ICT の進展

[1] 京都大学情報環境機構 IT 企画室
IT Planning Office, Institute for Information Management and Communication, Kyoto University
[2] 京都大学学術情報メディアセンター
Academic Center for Computing and Media Studies, Kyoto University

Vol.2019-CE-152 No.7
Vol.2019-CLE-29 No.7
2019/11/16

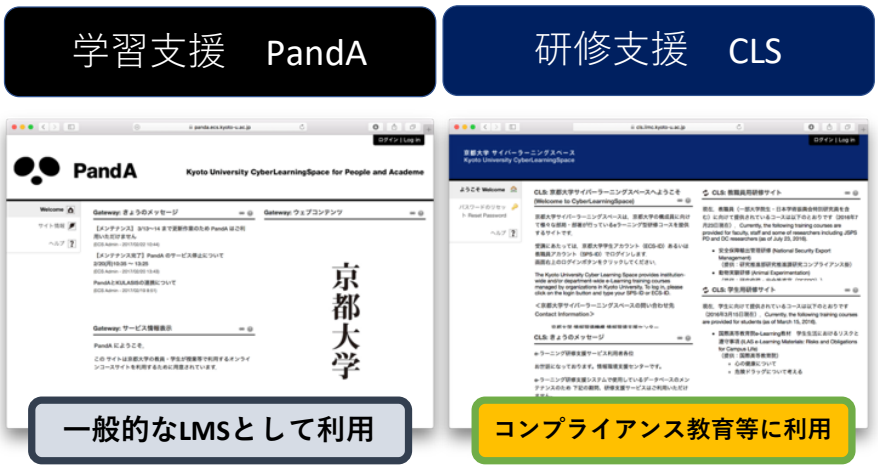

図 1　PandA と CLS.

は加速度を増し，日常業務を含むあらゆる場面で情報環境を利用することが当たり前となっていった．この流れの中で，京都大学では，2005 年 4 月，他の国立大学に先駆けて情報環境機構を本部組織に設置，事務業務の電算化を行ってきた情報部（本部組織，現在の企画・情報部）と学術情報メディアセンターの連携を強化するとともに，情報環境機構 IT 企画室の設置（2011 年 4 月）を通じて，情報環境機構の実体化を 2014 年 4 月に行った．これらの組織改廃を経て，情報環境機構として情報基盤から教育・研究・事務業務の支援まで業務を集約するとともに，教員・事務職員・技術職員が一体となって調査，企画，運営を行う，いわゆる「教職協働」体制を整備した．

また，この間，クラウドコンピューティングの進展により，ICT 環境は「所有から利用へ」と大きくシフトしており，サーバは Amazon AWS や Microsoft Azure 等の IaaS (Infrastructure as a Service) 型クラウドへ，端末は利用者がノート PC やタブレット，スマートホン等を持参する BYOD (Bring Your Own Device) へと大きく変わりつつある．また，豊富な資金の下，急速に進歩を続けるプラットフォーマと呼ばれる GAFA (Google, Apple, Facebook, Amazon) 等が台頭し，低コストで高品質の外部システム・サービスも利用することを前提にする必要がある．しかも，今後は，Digital Transformation とも呼ばれる，多様な教育・研究・事務業務の中身に踏み込んだ ICT 支援も求められている（以下「大学 DX」という）．

しかしながら，厳しい予算削減・人員削減が続く中，これまでのやり方では必要な学内リソースは先細りしていく一方で，「教職協働」体制であっても，多様な外部環境・内部ニーズには応えきれない状況になってきている．特に，少ない予算で対応する以上，開発から運用管理まで内製化による対応が今後中心になって行かざるを得ない．

そこで，本稿では，Sakai の認証システムとして利用している CAS (Central Authentication Service) のバージョンアップを題材に，運用管理の面からオーバレイ的な考え方による認証基盤の運用管理について議論する．

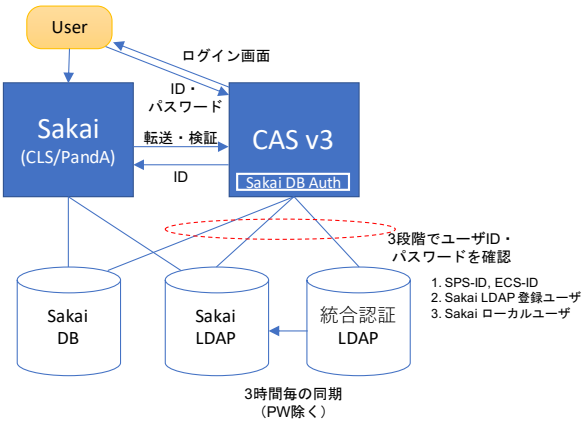

図 2　Sakai - CAS 連携.

2. CAS による Sakai のユーザ認証

京都大学では，LMS (Learning Management System) として，授業科目を対象にした学習支援サービス PandA および情報セキュリティ研修等のコンプライアンス系 e ラーニング研修支援サービス Cyber Learning Space (CLS) の 2 系統を 2012 年度から運用しており，いずれも，オープンソース LMS として世界で広く使われている Sakai を利用している (図 1 参照)．いずれも，同じバーションの Sakai を利用しているが，その構築・運用は別インスタンス・別スキンで運用管理している [1], [2].

Sakai は，データベースに登録されているユーザ情報を用いるローカルユーザ認証だけでなく，LDAP 認証やシングルサインオンが可能な Jasig Foundation[*1] の CAS (Central Authentication) や Internet2 の Shibboleth にも対応している．

もともと，CAS は Yale 大学で開発されたもので，Jasig を通じて発展，北米やフランス等の大学では，Shibboleth

[*1] 現在は Apereo Foundation.

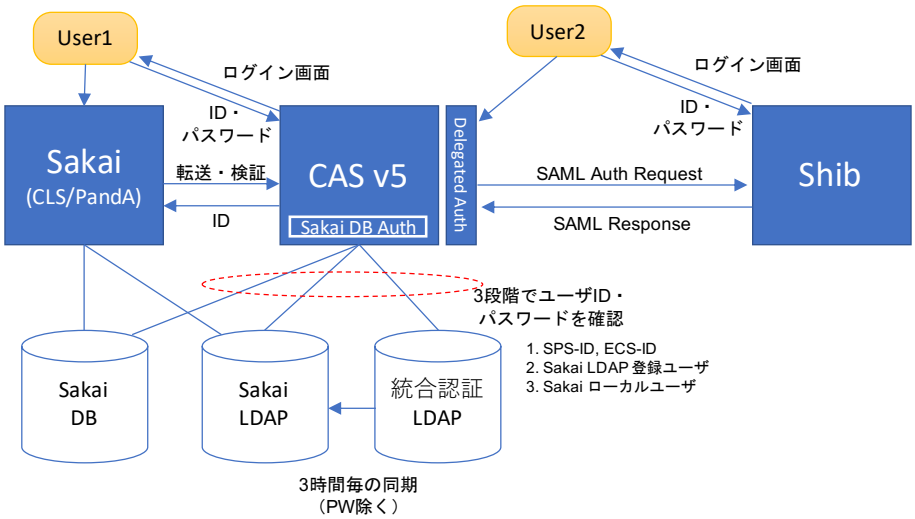

図 3 CAS v5 による SAML2 認証対応．通常の CAS 認証はこれまで通り利用でき，SAML2 認証により統合認証によるシングルサインオンを利用する場合は，CAS の Pac4JDelegatedAuthenticationHander が対応する URL にアクセスする．

認証と同じようにシングルサインオンを提供するユーザ認証システムとして利用されており，CAS は学内のウェブアプリケーション間のシングルサインオンを，Shibboleth は大学間のシングルサインオンに利用されることが多い [3]．また，CAS は「ユーザ認証はプログラムが責任をもつべきもの」との考え方が強く，一方で，Shibboleth は「ユーザ認証はコンテナサーバが責任を持つべきもの」との考え方が強く，それぞれの特長にあった使い方が求められる [3]．

我々は，すでに 7 年前のことになるが，Sakai をベースとした LMS 基盤を構築する際，CAS の拡張性に着目し，`QuerySakaiDbAuthenticationHander` を独自に開発し，(1) 全学で運用されている統合認証 LDAP，(2) 統合認証 LDAP から 3 時間おきにデータを同期させ，ユーザ情報を高速に参照するための Sakai LDAP，(3) Sakai ローカルユーザ認証，の 3 つの認証を多段で利用できるようにすることで，ユーザが特にユーザ認証方式を意識することなく，ログイン画面にユーザ ID をパスワードを入力するだけでいずれかの認証方式で利用できるようになっている．

3. CAS の Shibboleth 認証対応

教員・学生は，Shibboleth 認証に対応した京都大学の教務情報システムである KULASIS から PandA にアクセスするようになっていたため，当初から，Sakai の Shibboleth 認証対応が求められていた．Sakai は Shibboleth 認証対応も可能であるものの，「ユーザ認証はコンテナサーバが責任を持つべきもの」という Shibboleth の考え方から，Sakai が稼働するサーバ全体を Shibboleth 化するため，CAS との併用は困難であった．

その後，2016 年 3 月にリリースされた CAS Version 4.2 から導入された `Pac4J Authentication` を用いた認証代理機能により，SAML2 認証が可能になった [4]．`Pac4J` が提供する機能により，SAML2 IdP の利用以外にも，Facebook, Twitter, Google, LinkedIn, Yahoo 等が対応している OAuth2 プロバイダや OpenID プロバイダ，OpenID Connect IdP，ADFS が利用可能である．これにより，SAML2 認証にも対応している本学の Shibboleth 統合認証システムとのシングルサインオンの可能性が高まった．また，2018 年度の段階で，約 1,500 科目まで PandA の利用が伸張していることから，早急な Shibboleth 認証対応が求められていた．

4. CAS WAR オーバーレイテンプレート

一方で，高機能化[*2]する CAS のバージョンアップを容易にするため，2009 年頃から Maven2 の WAR Overlay 機能を用いて標準的な CAS WAR ファイルに対して，カスタマイズ部分のみ書き換える CAS WAR オーバレイの試行がはじまった．現在では，`cas-overlay-template` プロジェクトに対して，以下の設定を行うだけで，CAS の最新バージョンに容易にアップグレードできるようになっている:

- 追加で必要となる `<dependency/>` の `pom.xml` での指定
- `cas.properties` の設定 (図 4 参照)
- スキンの変更に必要なファイル，多言語化に必要なリソースバンドル等のカスタマイズ

[*2] 最新の CAS Version 6 では，Duo や Google Authenticator を含む多要素認証対応等，様々な認証方式への対応が進んでおり，アイデンティティサービスプロバイダとしての基本機能を Shibboleth 以上に包含している．

Vol.2019-CE-152 No.7
Vol.2019-CLE-29 No.7
2019/11/16

```
cas.server.name: https://xxxx.iimc.kyoto-u.ac.jp
cas.server.prefix: https://xxxx.iimc.kyoto-u.ac.jp/cas

logging.config: file:/etc/cas/config/log4j2.xml

cas.serviceRegistry.json.location=file:/etc/cas/services

# 統合認証 LDAP
cas.authn.ldap[0].order:              1
cas.authn.ldap[0].name:               TOGO LDAP
cas.authn.ldap[0].type:               AUTHENTICATED
cas.authn.ldap[0].ldapUrl:            ldaps://****.iimc.kyoto-u.ac.jp
cas.authn.ldap[0].validatePeriod:     270
cas.authn.ldap[0].searchFilter:       uid={user}
cas.authn.ldap[0].baseDn:             dc=kyoto-u,dc=ac,dc=jp
cas.authn.ldap[0].bindDn:             cn=****,o=****,dc=kyoto-u,dc=ac,dc=jp
cas.authn.ldap[0].bindCredential:     *****

# Sakai LDAP
cas.authn.ldap[1].order:              2
cas.authn.ldap[1].name:               SAKAI LDAP
cas.authn.ldap[1].type:               AUTHENTICATED
cas.authn.ldap[1].ldapUrl:            ldap://localhost:1389
cas.authn.ldap[1].useSsl:             false
cas.authn.ldap[1].validatePeriod:     270
cas.authn.ldap[1].searchFilter:       uid={user}
cas.authn.ldap[1].baseDn:             dc=cls,dc=kyoto-u,dc=ac,dc=jp
cas.authn.ldap[1].bindDn:             uid=***,ou=*****,dc=***,dc=kyoto-u,dc=ac,dc=jp
cas.authn.ldap[1].bindCredential:     ******

# Sakai Database Authenticator
cas.authn.jdbc.sakai[2].order=3
cas.authn.jdbc.sakai[2].sql=select a.pw from sakai_user a, sakai_user_id_map b
where a.user_id=b.user_id and b.eid=?
cas.authn.jdbc.sakai[2].url=jdbc:oracle:thin:@odb01sv:1521:orcl1
cas.authn.jdbc.sakai[2].dialect=org.hibernate.dialect.Oracle10gDialect
cas.authn.jdbc.sakai[2].user=*******
cas.authn.jdbc.sakai[2].password=********
cas.authn.jdbc.sakai[2].driverClass=oracle.jdbc.driver.OracleDriver
cas.authn.jdbc.sakai[2].fieldPassword=pw

# Delegated Authentication to SAML2 Identity Providers
cas.authn.pac4j.saml[0].keystorePassword=pac4j-demo-passwd
cas.authn.pac4j.saml[0].privateKeyPassword=pac4j-demo-passwd
cas.authn.pac4j.saml[0].keystorePath=/etc/cas/config/samlKeystore.jks
cas.authn.pac4j.saml[0].serviceProviderEntityId=https://cls2.iimc.kyoto-u.ac.jp/cas/
cas.authn.pac4j.saml[0].serviceProviderMetadataPath=/etc/cas/config/sp-
metadata.xml
cas.authn.pac4j.saml[0].identityProviderMetadataPath=/etc/cas/config/idp-
metadata.xml
cas.authn.pac4j.saml[0].clientName=SAML2Client
cas.samlCore.securityManager=org.apache.xerces.util.SecurityManager
```

図 4　cas.properties の主要な内容.

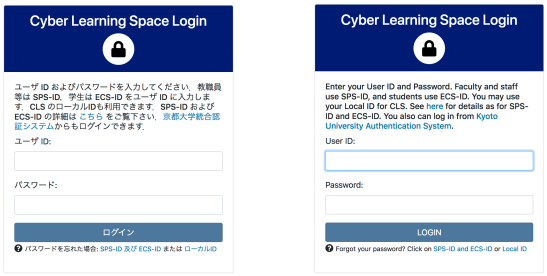

図 5　新しい CAS による認証画面（日本語版と英語版）.

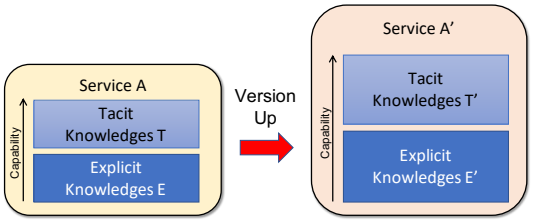

図 6　バージョンアップに伴う形式知と暗黙知.

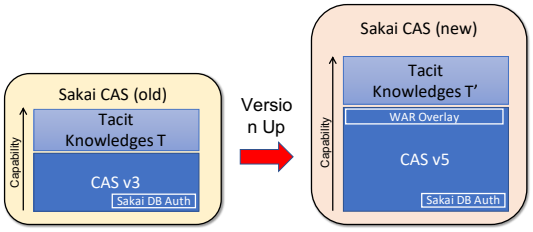

図 7　オーバレイ機能によるバージョンアップの容易化.

5. CAS Version 5 へのバージョンアップ

最新の CAS は Version 6 であるものの，2018 年 12 月にリリースされたばかりであることから，一つ前のメジャーバージョンである Version 5 を前提に次の手順によりバージョンアップを行った [5]:

(1) Apache HTTPD, Apache Tomcat サーバの構築
(2) `cas-overlay-template` を用いた CAS サーバの構築
(3) SAML2 認証の設定とテスト
(4) `QuerySakaiDatabaseAuthenticationHandler` の移植
(5) スキンや多言語設定等，現行システムへの適合 (図 5 参照)

多忙な中でのバージョンアップ作業であったものの，著者一人によるワークで総計 50〜80 時間程度[*3]で作業は実施できた[*4].

6. 議論

上述のように，長年の懸案事項であった CAS の Shibboleth 認証化を比較的短期間かつスムーズに行うことができたのは，明らかに `cas-overlay-template` を含む，CAS そのものに依拠しているところが大きい．通常，このようなバージョンアップは，オープンソースソフトウェアやそれに関するドキュメントとして「形式知 (Explicit Knowledge)」化されているものを理解し，必要なカスタマイズを行う人的リソースが不可欠である．その知識は，その人の「暗黙知 (Tacit Knowledge)」となり[*5]，これらの総和としてサービスが提供できる機能や能力 (Capability) が定まる

[*3] あくまでも感覚的な時間で計測したわけではない.
[*4] ただし，新認証システムへの切替直前に，2 度目のアクセスでは SAML 認証後に CLS へリダイレクトされない不具合が発見され，1 週間程，運用開始に遅延が発生した.
[*5] 引き継ぎ文書化等を通じて「形式知 (Explicit Knowledge)」化することはできよう.

(図 6 参照).

しかしながら，バージョンアップを行う度に必要となる形式知を理解することは，たとえ，同じ担当者がバージョンアップを行うとしてもその負荷はバージョンアップ間隔が大きくなるともに高くなる．一方で，`cas-overlay-template` のように必要な設定をそのまま引き継げるオーバレイ機能が整備されることにより，その負荷は確実に少なくすることができよう (図 7 参照).

また，CAS そのものの実行環境としての Servlet コンテナの Tomcat, ウェブサーバとしての Apache HTTPD を含む，仮想マシン全体を Docker コンテナとして既存の仮想環境に「オーバレイ」したり，Kubernetes のような負荷分散機能やフォールトトレランス機能を負荷したコンテナオーケストレーションエンジンを使用して「オーバレイ」することにより，さらにバージョンアップに伴う負荷を軽減できると考えられる．

7. まとめ

本稿では，Sakai の認証システムとして利用している CAS のバージョンアップを題材に，運用管理の面からオーバレイ的な考え方による認証基盤の今後について議論した．議論を通じて明らかになったように，オーバレイ的な考え方を推し進めるためには，「オーバレイ」の多段化が鍵となりそうであるが，その「オーバレイ」の対象自体がコスト高では全く無意味である．共同利用的な発想の下，既存の大学間連携の枠組みを大きく変える新しい協働組織体の必要性もあるのではないだろうか．

参考文献

[1] 外村 孝一郎，津志本 陽，梶田 将司， “京都大学における Sakai CLE による学習支援環境の現状と課題”， 情報処理学会第 21 回 CLE 研究発表会, 京都大学吉田キャンパス，京都, 2017 年 3 月 21〜22 日
[2] 梶田将司，外村孝一郎, “大学における e ラーニング研修支援サービス”, 情報処理学会研究報告教育学習支援情報システム（CLE）, Vol. 2018-CLE-25, No. 13, pp.1-8, 2018 年 6 月 16 日
[3] 梶田将司, 平野靖, 内藤久資, 間瀬健二, “CAS・Shibboleth による学内シングルサインオンと ID フェデレーションの連携統合”, 電子情報通信学会 2010 年総合大会, 東北大学, 2010 年 3 月 16 日〜19 日
[4] `https://apereo.github.io/cas/4.2.x/index.html` (2019 年 9 月 17 日アクセス)
[5] David A. Curry, “Deploying Apereo CAS”, `https://dacurry-tns.github.io/deploying-apereo-cas` (2019 年 9 月 17 日アクセス)