UNIVERSIDADE FEDERAL DE SANTA CATARINA - UFSC

UM ESTUDO DAS METODOLOGIAS OPEN SOURCE IDENTITY MANAGEMENT E INDICAÇÃO DA MELHOR A SER IMPLANTADA NO PROJETO VIA DIGITAL

Leandro Nascimento Loi

Florianópolis – SC
2007/2

UNIVERSIDADE FEDERAL DE SANTA CATARINA - UFSC
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE SISTEMAS DE INFORMAÇÃO

UM ESTUDO DAS METODOLOGIAS OPEN SOURCE IDENTITY MANAGEMENT E INDICAÇÃO DA MELHOR A SER IMPLANTADA NO PROJETO VIA DIGITAL

Leandro Nascimento Loi

Trabalho de conclusão de curso apresentado a UFSC como parte dos requisitos para a obtenção do grau de Bacharel em Sistemas de Informação.

Florianópolis - SC
2007/2

Leandro Nascimento Loi

Um estudo das metodologias Open Source Identity Management e indicação da melhor a ser implantada no projeto Via Digital

Trabalho de conclusão de curso apresentado como parte dos requisitos para a obtenção do grau de Bacharel em Sistemas de Informação

Banca Examinadora:

_____
Prof. Dr. José Eduardo De Lucca
Universidade Federal de Santa Catarina
Orientador

_____
Rodrigo Copetti
Banca

_____
Achilles Colombo Prudêncio
Banca

AGRADECIMENTOS

Ao professor De Lucca, que me orientou durante este trabalho e aos demais integrantes da banca bem como a todos que de alguma forma contribuíram para a realização desse trabalho.

Ao meus pais, Luiz e Leda que sempre me apoiaram, em todos os momentos desde o começo da minha faculdade, assim como minha irmã Letícia que sempre me fez ver as coisas por outro ângulo.

Aos meus amigos que sempre me apoiaram em todos os momentos em que estive ausente e que não pude dar atenção.

A minha namorada Márcia que sempre me apoiou, me deu forças e compreendeu nas diversas vezes em que deixei de estar com ela para completar esse trabalho.

RESUMO

A falta de um *single sign-on* visto superficialmente pode parecer um problema relativamente simples causando apenas um inconveniente para o usuário, mas o impacto dentro das instituições é muito mais profundo. Além de sobrecarregar o sistema de ajuda ao usuário, ele aumenta os custos administrativos de cada aplicação, compromete a segurança e reduz a produtividade. A partir da proliferação de sistemas que dão suporte aos processos de um negócio, usuários e administradores de sistemas estão se deparando com um cenário extremamente complicado para integrar as suas funções. Usuários normalmente precisam se autenticar em diferentes sistemas, necessitando um número equivalente de perguntas, cada um envolvendo diferentes usuários e senhas. Administradores de sistemas enfrentam a administração das contas dos usuários de cada um dos diversos sistemas, que devem ter sua integridade de segurança garantida tendo em vista que os usuários têm acesso diferenciado dentro de cada um dos sistemas. Com o estudo efetuado é possível determinar dentre uma grande quantidade de sistemas existentes atualmente, qual o melhor para se implantado não somente no projeto Via Digital, mas em qualquer empresa e ou projeto.

Palavras-chave: Single Sign On, Gerenciador de Identidade, Privacidade, Credenciais.

ABSTRACT

The lack of a *single sign-on* in an overview at the surface is a relative simple problem causing only inconvenience for the user, but the impact inside the organizations is deeper. It's not just because overload the system of helpdesk, but it increase the risks of security and decrease the productivity. From the proliferation of system that gives support to the process of business, users and administrators of systems are coming across with an extreme complicated scene to integrate the functions. User normally needs to authenticate in different systems, needing a number of equivalents of questions, each one involving different users and passwords. Administrators of systems face administration user accounts in different systems, which must have its integrity of guaranteed security, because users have different levels of access in the system. With this study it's possible determinate between a large quantity of systems existing actually, which one is better to be used not just in Via Digital project, but in any company.

Keywords: *Single sign-on*, Identity Management, Privacy, Credential

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

- **PGP** - Pretty Good Privacy
- **SOAP** - Simple Object Access Protocol
- **SASL** - Simple Authentication and Security Layer
- **LDAP** - Lightweight Directory Access Protocol
- **LAN** - Local Area Network
- **XML** - Extensible Markup Language
- **W3C** - The World Wide Web Consortium
- **OASIS** - Organization for the Advancement of Structured Information Standards
- **URL** - Uniform Resource Locator
- **SSO** - Single Sign On
- **SAML** - Security Assertions Markup Language
- **PIN** - Personal Information Number
- **SMS** - Short Message Service
- **FIM** - Federated Identity Management
- **OSS MAP** - Open Source Identity Management Map
- **IdPs** - Intrusion Detection and Prevention Systems

SUMÁRIO

1. Introdução

Atualmente notamos um crescimento das organizações e com isso a dificuldade de gerenciar seus usuários e seus bancos de dados, pois cada vez que se cria uma aplicação surgem novos desafios de gerenciamento e manutenção das contas e restrições de acesso a certas informações. É nesse cenário em que este trabalho busca trazer uma solução para um problema do dia a dia, que é o de se autenticar diversas vezes dentro de um sistema.

Além das perdas monetárias com administração de contas ainda existe a perda de tempo gerada para os usuários que precisam se autenticar cada vez que vão utilizar um serviço diferente dentro da organização. Outros pontos devem ser observados também, que além de todos os malefícios citados a cima ainda existe a problemática das perdas de senha e do vazamento de informações privadas, tudo isso porque não existe uma segurança centralizada. Para resolver esses problemas foram desenvolvidos sistemas que buscam a unificação da autenticação, trazendo simplicidade tanto para o administrador do sistema que a partir desse momento tem apenas uma conta por usuário para gerenciar, quanto para o usuário do sistema que efetua a autenticação apenas uma vez e poderá utilizar o sistema com todas as suas permissões de acesso garantidas.

2. Tema

O tema deste trabalho de conclusão de curso é um estudo das tecnologias de *Management Identity* existentes atualmente. Em busca dessa solução serão analisadas as diversas formas de autenticação existentes no *Open Source Identity Management Map*, que é um mapa de todas das principais ferramentas disponíveis para o gerenciamento de usuários.

2.1 Delimitação do Tema

O trabalho proposto tem como objetivo o estudo do funcionamento das soluções de *Identity Management*, traçando assim comparações entre os projetos existentes no cenário atual, buscando a melhor solução de um *login* único para o projeto Via Digital.

Procura-se dentre outros objetivos, determinar qual das metodologias existentes no *Open Source Identity Management Map* mais se ajusta ao projeto Via Digital e posteriormente propor uma arquitetura viável a ser implementada, no entanto esse trabalho não tem a pretensão de colocar em prática a arquitetura proposta.

3. Objetivo Geral

Estudar mecanismos e tecnologias de gerenciamento de usuários em ambientes distribuídos na Web, baseando-se nas arquiteturas propostas pelos projetos do *Open Source Identity Management* e avaliar o grau de efetividade das mesmas.

Além de propor uma solução que possa ser utilizada no projeto Via Digital para que haja um sistema de *login* único respeitando as restrições e limitações impostas para cada usuário dentro do sistema. Os estudos poderão oferecer informações para as instituições que desejam facilitar a adesão de usuários em seus sistemas e proporcionar uma melhoria no gerenciamento dos mesmos.

3.1 Objetivos Específicos

- Estudar e compreender o ecossistema de ferramentas de gerenciamento de identidades distribuídas, em especial o conjunto definido no *Open Source Identity Management Map*.
- Especificar a necessidade de gerenciamento de identidades dos componentes do projeto Via Digital.
- Propor uma arquitetura que atenda às necessidades de gerenciamento de identidades do Via Digital de acordo com o *Open Source Identity Management.*

4. Justificativa

Atualmente uma das maiores dificuldades que se enfrenta na web é a necessidade de se ter uma conta em cada portal ou em cada sítio que você acessa. A solução para esse problema é o gerenciamento de identidade, que pode ser trabalhado com os diversos projetos dentro do *Open Source Identity Management*.

Esse é um mercado de grande potencial e que está começando a surgir agora, onde grandes empresas como Google, Yahoo e Microsoft já despertaram interesses. Tendo isso em vista, juntamente com a necessidade de uma solução para esse problema no projeto Via Digital que este trabalho irá propor uma arquitetura adequada para solucionar o problema de múltiplos *logins* nos diversos módulos e sistemas.

5. Metodologia

A metodologia utilizada para o desenvolvimento desse trabalho procurou estudar o estado da arte em que se encontram os projetos de código aberto para a realização de um *login* único. Para isso será utilizado o *Open Source Identity Management Map* em que constam todos os projetos em desenvolvimento nessa área, principalmente os que são *open source* que proporcionam aos programadores de todas as partes do mundo comecem a desenvolver seus sistemas com um sistema de *login* robusto e já pronto para ser utilizado bastando apenas adaptar as suas necessidades.

Após esse estudo preliminar será realizado um estudo de como o projeto Via Digital funciona e quais os requisitos deverão ser cumpridos.

Finalmente para a conclusão desse trabalho será feita uma proposta de arquitetura a ser utilizada para o gerenciamento dos componentes do projeto.

6. Revisão Bibliográfica

Nesse capitulo serão abordados os gerenciadores de identidade oferecidos dentro do Open Source Identity Management Map, que conduzirá a revisão bibliográfica desse trabalho.

6.1 Gerenciamento de identidade

Segundo a Wikipedia(2007), Identity Management são sistemas de informação que se referem ao gerenciamento de identidades, ou seja, o gerenciamento de uma identidade durante certo ciclo de vida. Esse ciclo é definido por três estados distintos:

1. A identidade é estabelecida: quando um nome ou número é ligado à entidade (objeto ou sujeito) que é necessário identificar, gerando assim uma nova identidade que agora possui um nome ou número ligado a ela.
2. A identidade é descrita: quando um ou mais atributos coerentes com a entidade são atribuídos à identidade sendo então descrita novamente, pois um ou mais atributos podem ter sido modificados.
3. A identidade é destruída.

A identidade de uma pessoa contém uma grande quantidade de informação pessoal a respeito de um individuo e todos os subconjuntos que

representam uma pessoa. Algumas dessas “identidades parciais” identificam uma pessoa unicamente, porém não é sempre que acontece. Sendo assim dependendo da situação e do contexto, uma pessoa pode ser representada por diferentes identidades parciais.

Um sistema gerenciador de identidades provê as ferramentas para gerenciar essas identidades parciais no mundo digital. Por exemplo, uma pessoa pode utilizar uma ou mais identidades parciais para trabalhar, para atividades de lazer, para lidar com bancos, supermercado ou serviços de internet. Algumas dessas identidades parciais contêm informações nas quais outros parceiros de comunicação geralmente conheceriam sobre essa pessoa.

No mundo digital, o gerenciamento de identidade no mundo digital relata o comportamento de uma pessoa nas atividades do dia a dia. Cada pessoa deve decidir o que informar dela mesma aos outros, após ter se contextualizado com a situação e o papel que cada um tem ao agir no relacionamento da comunicação entre as partes. Algumas vezes diferentes nomes, apelidos ou pseudônimos são escolhidos como identidade.

Ocasionalmente é necessário permanecer inteiramente anônimo, por exemplo, quando se compra algum artigo em uma loja. Em outros casos é necessário revelar informações pessoais, por exemplo, na abordagem de um policial a identidade sempre é solicitada. O anonimato não é freqüentemente aceitável, todavia somente algumas informações pessoais ou credenciais são realmente necessárias. As diferentes escolhas dependem da vontade do usuário e dos requisitos das aplicações que dão suporte aos sistemas de gerenciamento de identidades.

Entretanto, ambas as técnicas de anonimato ou autenticação são requeridas para preencher os requisitos de um gerenciador de identidade: No mundo digital onde todas as informações desprotegidas podem ser guardadas ou *linkadas*, é necessário garantir o anonimato como proteção a privacidade, especialmente na área da rede de comunicação. Por outro lado assinaturas digitais integradas em uma estrutura apropriada, provêem autenticidade para operações que necessitam um alto grau de confiança na comunicação, evitando o roubo de identidade.

6.2 Visão geral de um sistema de gerenciamento de identidade

Os identificadores de assuntos agem como pseudônimos ou como um conjunto de assuntos, onde como anônimo de um lado e com identificação única do outro, são extremos a respeito da conectividade dos assuntos, o mesmo compromete tudo se incluindo aos extremos. Por isso o pseudônimo serve como núcleo dos mecanismos de gerenciamento de usuários.

Entretanto, utilizar o pseudônimo mais de uma vez, pode-se tirar vantagem de uma reputação já estabelecida, por isso alguns tipos de pseudônimos possibilitam lidar com reclamações em caso de abuso de falsa

“ligação” do portador. Possibilitando a terceiros (investigadores ou promotores) revelar a identidade do portador do pseudônimo em caso de uma investigação policial.

Integrando as informações do pseudônimo com as da “ficha” da identidade parcial, devemos possuir algumas propriedades importantes (Andreas Pfitzmann e Marit Hansen, 2004):

- Prova de propriedade: Pseudônimos digitais podem ser notados como chaves-públicas para testar assinaturas digitais onde o portador do pseudônimo deve provar a propriedade através de uma assinatura digital que é criada usando uma chave privada. Ex.: A chave pública PGP, inserida em um e-mail é um pseudônimo digital.
- Conhecimento inicial entre o elo do pseudônimo e seu portador: Pseudônimos podem ser criados pelo usuário ou eles podem ser gerados e assinados por um aplicativo provido por uma terceira parte. No contexto de gerenciamento de identidade, o elo entre o pseudônimo e o portador não deverá ser conhecido publicamente.
- O elo entre o uso de um pseudônimo em diferentes contextos: Se o mesmo pseudônimo é utilizado em diversas situações, a informação correspondente a respeito do portador deve ser correlacionada por ser aberta em cada uso. Em geral, o anonimato é maior quanto menos freqüente e menos *context-spanning* o mesmo pseudônimo é usado. Distinguir entre operações de pseudônimos - a qual é usada unicamente para uma transação de pseudônimos de situação a qual é

usada para um contexto específico e de pseudônimos pessoais que pode ser substituído pelo nome do portador em uma identidade civil.

- Convertibilidade, por exemplo, transferência de atributos de um pseudônimo para outro: O usuário pode obter uma credencial convertida de uma organização usando um dos seus pseudônimos, no entanto pode demonstrar a posse da credencial para outra organização sem revelar seu pseudônimo inicial. Para esse propósito, uma credencial pode ser convertida para o pseudônimo utilizado no momento.
- Autorizações podem ser feitas pela credencial ou por certificados (assinatura digital), por conseguinte no caso de assinaturas digitais elas são transferíveis para outras pessoas através de assinaturas digitais ou certificados ocultos.

6.3 Autenticação

Autenticação é o ato de estabelecer ou confirmar algo como autêntico, isto é, a reivindicação de ser ou ter feito algo ser verdadeiro. Autenticar um objeto pode significar confirmar a sua procedência ou autenticar uma pessoa consiste em verificar sua identidade dependendo de um ou mais fatores que a caracterizam.

Na segurança da computação, autenticação é o processo de tentar verificar a identidade digital do remetente em uma comunicação com um pedido de “*login*”. O remetente autenticado pode ser uma pessoa usando um computador, um computador em si ou um programa de computador. Uma credencial cega em contraste não estabelece uma identidade em si, mas sim um status para o usuário ou o programa.

Na internet “autenticação” é a forma para assegurar que os usuários são quem estão dizendo ser. Isso permite que o usuário faça no sistema o que tem permissão para fazer.

O problema da autorização é que freqüentemente pensa-se que é igual à autenticação, no entanto existe uma grande diferença entre as duas onde a primeira é muito adotada como protocolo padrão de segurança, regulamentação obrigatória e até mesmo estatutos são baseados nesse conceito. Entretanto, um uso mais preciso descreve a autenticação como um processo de verificar a identidade de uma pessoa, enquanto autorização é o processo de verificar se a pessoa conhecida tem o direito de fazer certas coisas dentro do sistema. Autenticação, entretanto deve preceder a autorização. Por exemplo, quando você mostra a sua autenticação para o caixa no banco, você poderia ser autenticado pelo caixa e assim você estaria autorizado a acessar as informações da sua conta. Você não seria autorizado a ter acesso a uma conta que não fosse sua.

Dessa forma uma autorização não pode ocorrer sem uma autenticação e o termo “*Former*” é muitas vezes utilizado para significar a combinação de autenticação com autorização (Wikipedia,2007).

6.3.1 Subgrupos

A autenticação pode ser dada de diferentes modos. Existem diversos modelos inseridos no *Open Source Identity Management Map(2007)*:

- *Single Sign On* – é uma sessão ou um processo de autenticação que permite a um usuário digitar um nome e uma senha para acessar diversas aplicações. O processo autentica o usuário a ter acesso a todas as aplicações que lhe foi dado o direito e elimina a necessidade de indagá-lo novamente toda vez que troca de aplicação em uma sessão em particular.
- *Web Services* – A troca de informações de autenticação entre um *Web Service Consumer* e o *Web Service Provider* é realizado utilizando SOAP-*bound messages*. As mensagens são uma série de requisições do cliente e respostas do servidor específicas para definir o mecanismo SASL (ou modo de autenticação).
- *Strong Authentication* – É um sistema dito “Autenticação de dois fatores”, pois ele requer no mínimo dois fatores para autenticação. Sendo que um é algo que você sabe e o outro obrigatoriamente algo

que você é ou você possui. Dessa forma um sistema só é considerado com *Strong Authentication* quando possui além de uma senha uma autenticação mecânica ou biométrica.

6.4 Sistemas de Identidade

Através de um usuário e senha são caracterizados os sistemas de identificação. Eles são considerados a arquitetura 1.0 da Web, sendo difícil à identificação de um sítio para outro o que ocasiona a necessidade de se autenticar em um sítio e depois novamente em outro. Ou seja, não há qualquer tipo de relação entre uma identidade com a outra, havendo dessa forma a necessidade de criação de diversas contas. Entre as principais características desse modelo estão:

- Registro local;
- Falta de verificação;
- Diretório centralizado;
- Usuário e senha;
- Não portátil;
- Falta de transparência.

6.4.1 Subgrupos

Dentre os diversos modelos (*Open Source Identity Management Map*,2007) de sistemas de identidade para o modelo de *identity system* os mais importantes serão descritos abaixo:

- Bandit – É um conjunto de componentes de software que proporciona um serviço consistente de identidade para autenticação, autorização e auditoria. Criando uma comunidade que organiza e padroniza as tecnologias de identidade em apenas uma forma, proporcionando interoperabilidade e colaboração. Além disso, implementa os protocolos de padrão aberto e especificações de forma que serviços de identidade podem ser construídos, acessados e integrados de múltiplos sistemas de identidade.
- OSIS – É um projeto que integra diversos projetos de código aberto sobre identidades. Além de sincronizar e harmonizar a construção de uma camada de identificação com um alto grau de interoperabilidade para internet com sistemas de código aberto.
- Concórdia – Foi designado para ser uma iniciativa para puxar a frente da harmonização e a interoperabilidade de especificações das identidades e dos protocolos. Como expressado no nome, o principal objetivo desse grupo é auxiliar no desenvolvimento de cenários de

casos de uso onde múltiplas especificações de identidade e padrões ou outras iniciativas podem coexistir, reconhecendo ambientes heterogêneos.

- Login-Based – Sistemas baseados em *login* e senha que são os mais utilizados atualmente na Web. Esses sistemas permitem que uma pessoa ou uma máquina acesse serviços de terceiros. Dentro desse modelo se encontra projetos como:
    - Google Authentication;
    - Yahoo BBAuth;
    - PingLogin;

6.5 User Centric

Segundo Audun Jøsang e Simon Pope (2005), *User Centric identity* muda o foco do *domain-centric identity management* para o usuário em si, oferecendo grande flexibilidade em como e onde guardar sua identidade, controlando como essa identidade será usada e compartilhada com uma forte segurança e uma ótima privacidade. Apesar da aceitação dos valores da tecnologia *user centric*, ela não é universalmente reconhecida como um conjunto de critérios que o *user centrism* pode ser mensurado. Para alguns, o

termo significa hospedado no cliente, para outros significa dar ao usuário mais opções de como e onde na rede ele poderá guardar sua própria identidade.

6.5.1 Subgrupos

Existem diversos modelos de *user centric*. Dentro do *Open Source Identity Management Map* (2007) são esses os mais importantes:

- CardSpace – É um software cliente que possibilita ao usuário disponibilizar sua identidade digital a serviços *online* de uma forma simples segura e confiável. Ele é conhecido como um seletor de identidade: Quando um usuário precisa se autenticar em um sitio ou em um *web service*, o CardSpace abre uma interface com um conjunto de informações do cartão para que o usuário e escolhe quais informações serão disponibilizadas.
- OpenID - é um framework aberto e descentralizado, orientado a identidade digital do usuário. O OpenID parte do princípio que qualquer pessoa pode se identificar na internet da mesma forma que os sítios identificam um endereço na web. Tendo em vista que a URI é o núcleo da arquitetura Web, elas podem prover uma sólida identidade orientada aos usuários.

- Higgins – É um projeto de código aberto hospedado na fundação Eclipse. Entretanto, os componentes também podem ser usados para criar IdPs e RPs, o foco da criação de um framework com “Agentes de identidade”. Um AI é um serviço local que gerencia múltiplas identidades dos usuários. O sistema fica entre usuários remotos IdPS (ex.: bancos) mas por outro lado e vários RPs. O Higgins AI é chamado pelo browser do usuário, gerenciando as informações da identidade do usuário armazenadas remotamente no usuário.
- FOAF – É uma tecnologia simples que proporciona uma forma de compartilhar e usar a informação sobre pessoas e suas atividades para transmitir informação entre sitios e para automaticamente estender, reunir e reutilizar ela *online*.
- AIAKOS – Permite que você provenha um sistema central de *login* para uma rede de websites. Todas as atividades de *logins* e registros são feitos em um único local, permitindo que os sítios participantes recebam pacotes de autenticação criptografados pelo servidor de autenticação.

6.6 Provisioning

Segundo a Webopedia *provisioning (2004)* é o processo de prover ao usuário acesso a informações e recursos tecnológicos. O termo tipicamente é utilizado para referenciar o gerenciamento de recursos das empresas.

Podendo ser interpretado como a combinação de deveres das pessoas e dos departamentos de TI dentro de uma empresa, onde os usuários ganham acesso a repositórios de informação, autorização para utilizar sistemas, aplicativos, banco de dados baseados em uma identificação única e um hardware apropriado, tal como computadores e celulares. Esse processo implica em direito de acessos, os privilégios são monitorados e rastreados para assegurar a segurança dos recursos das empresas. Isto significa prover a consumidores ou clientes contas com o acesso apropriado e todos os direitos associados e incluindo os recursos necessários para prover o gerenciamento dessa conta.

6.6.1 Subgrupos

Existem diversos modelos de *provisioning*. Dentro do *Open Source Identity Management Map* (2007) os que mais se destacam são esses:

- SPML – *Service Provisioning Markup Language* é baseado em XML para consultas e troca de usuário, contas e recursos de provimento de requisição.
- SyncML – Provê uma linguagem para gerenciamento de dispositivos.

6.7 Gerenciamento de Usuário

O gerenciamento dos usuários de uma aplicação ou sistema proporciona que cada um possua características específicas a serem gerenciadas.

6.7.1 Subgrupos

Os principais subgrupos dentro do Open Source Identity Management Map (2007) são:

- *Meta Diretories* – É um serviço principal de dados para uma coleção de diretórios de uma rede que provê alta qualidade de interfaces com o usuário e muitas respostas interativas. No entanto, não faz monitoramento LDAP direto ou transações em tempo real.

- *Virtual Diretory* – É um serviço de virtualização de dados para coleções de diretórios de rede e banco de dados que provêem dados sincronizados, replicação e acesso dinâmico ao conteúdo.
- *Diretory Services* – É um serviço de diretório LDAP que prove um banco de dados capaz de armazenar informações heterogêneas em uma única instância.

6.8 Gerenciamento de Acesso

Segundo a definição do Open Source Identity Management Map (2007) o termo *access management* é geralmente usado para descrever grandes sistemas que utilizam os serviços de autenticação e de autorização para controlar o uso dos recursos de uma rede dentro de uma organização ou estrutura de rede.

6.8.1 Subgrupos

Abaixo os principais grupos seguindo o Open Source Identity Management Map (2007).

- *Network Access Control* – Projeto que provê uma forma de controle de acesso dentro de uma LAN, incluindo todos os tipos de dispositivos entre eles: servidores, IP-Phones, webcams entre outros.

- OpenSSO – Fornece o núcleo de serviços de identidade para simplificar a implementação de um *single sign-on* transparente como um componente de segurança em uma infra-estrutura de rede.

- *Policy Language* – É uma linguagem para descrever políticas de privacidade para os diversos recursos da Web.
- WP6 – É um estudo para definir mecanismos de processos de autorização dentro de um *framework* para um projeto de *DataGrid* e para desenvolver ferramentas para implementar esse processo.
- PAPI – É um sistema para prover controle de acesso a informações e recursos dentro da internet. A intenção é manter a autenticação na organização local em que o usuário pertença, enquanto deixar a informação totalmente controlada pelo provedor do serviço, assim como o que ele oferece.
- SWITCH AAI – É o núcleo do sistema com o objetivo de simplificar o acesso inter-organizacional para os recursos da web. Com apenas um *login*, um estudante pode acessar os sistemas de *e-learning* das diversas universidades na suíça.
- AthensAM – Auxilia na proteção dos dados pessoais dos usuários, através do uso de atributos pessoais.
- VOMS - Através de certificados dos atributos a outra parte passa a confiar nas políticas dos sistemas.
- AKENTI - É um modelo de segurança e arquitetura que tem a intenção de prover um serviço de segurança escalável em ambientes com redes distribuídas.
- PERMIS – É uma infra-estrutura que prove todas as facilidades necessárias para que os usuários gerenciem privilégios, políticas de autorização e para que as aplicações possam tomar decisões de autorização.

- Shibboleth – Provê web *Single Sign On* através ou com limites organizacionais. Ele permite que os sítios tomem decisões informadas de autorização de acesso a recursos protegidos dentro dos sistemas.

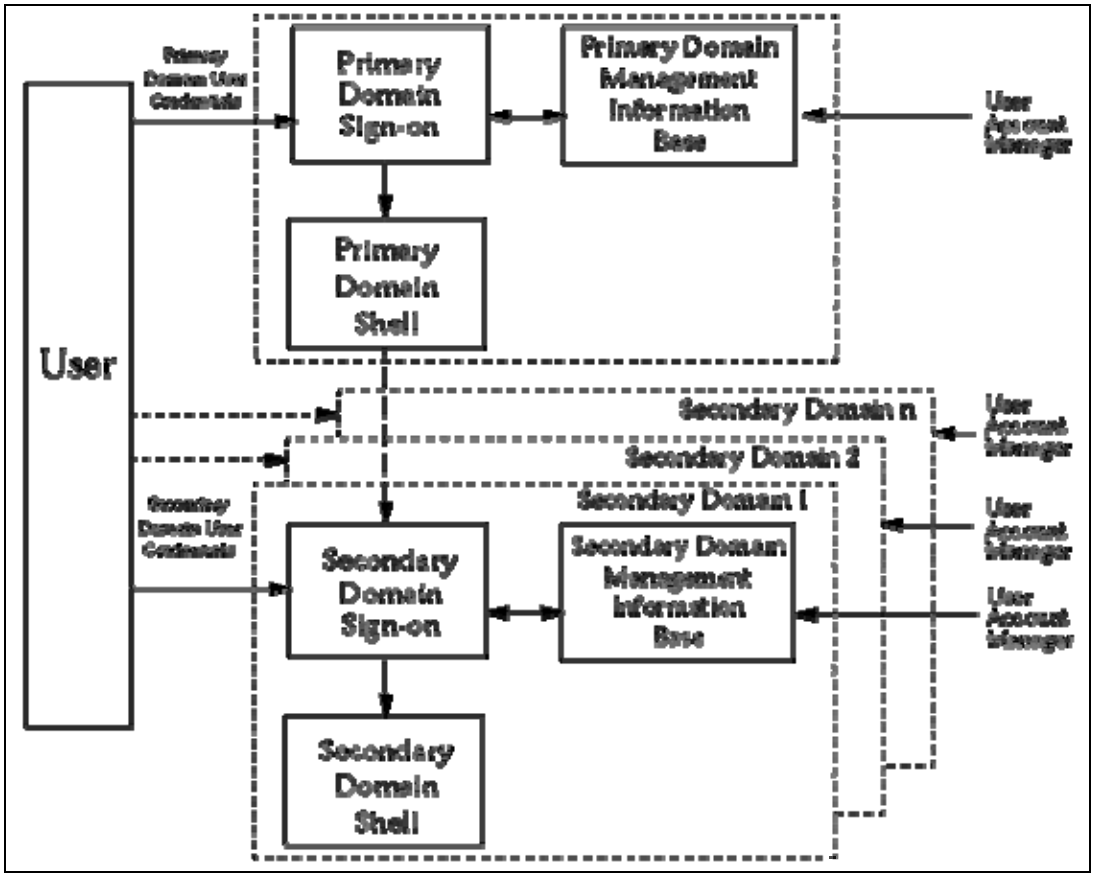
6.9 Single Sign-On

Para Vipin Samar (1999), da Oracle um único *login* tem sido um difícil problema por algum tempo. Isso não tinha sido um problema até recentemente, porque normalmente as pessoas acessavam poucas aplicações. Entretanto, hoje com a integração dos aplicativos para *desktop* com a web e com o crescente número de serviços que um usuário acessa diariamente isso tem se tornado um problema cada vez maior. Não é mais aceitável ser preciso digitar seu usuário e senha várias vezes por dia. Tirando o fato de ser extremamente inseguro é muito inconveniente e caro para o administrador.

A falta de um *single sign-on* visto superficialmente pode parecer um problema relativamente simples causando apenas um inconveniente para o usuário, mas o impacto dentro das instituições é muito mais profundo. Além de sobrecarregar o sistema de ajuda ao usuário, ele aumenta os custos administrativos de cada aplicação, compromete a segurança e reduz a produtividade.

A partir da proliferação de sistemas que dão suporte aos processos de um negócio, usuários e administradores de sistemas estão se deparando com

um cenário extremamente complicado para integrar as suas funções. Usuários normalmente precisam se autenticar em diferentes sistemas, necessitando um número equivalente de perguntas, cada um envolvendo diferentes usuários e senhas. Administradores de sistemas enfrentam a administração das contas dos usuários de cada um dos diversos sistemas, que devem ter sua integridade de segurança garantida tendo em vista que os usuários têm acesso diferenciado dentro de cada um dos sistemas. Essa realidade é mostrada na imagem abaixo:



O problema é histórico, pois os sistemas foram desenvolvidos com componentes que agem em domínios seguros de forma independente. Esses

componentes compreendem plataformas individuais com operações associadas a apenas aquele sistema e aplicativo.

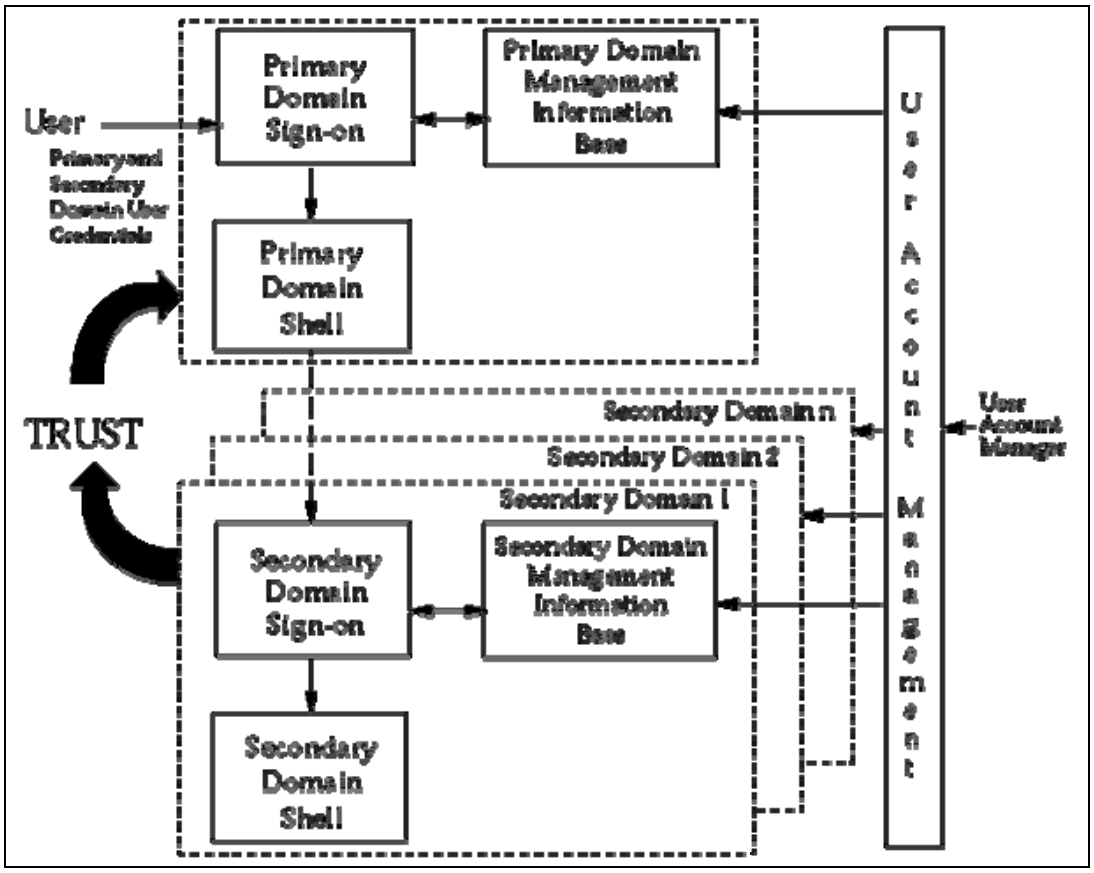
Estes componentes agem em domínios independentes no sentido que um usuário final precisa se identificar e autenticar independentemente de qual domínio ele deseja interagir. Esse cenário é ilustrado acima. O usuário final interage inicialmente com um domínio principal para estabelecer uma sessão com esse domínio primário. Isto é denominado *Primary Domain Sign-on* como mostrado no diagrama e requer que o usuário final informe sua credencial aplicável ao domínio principal, por exemplo, um usuário e uma senha. A sessão do domínio principal é tipicamente representada por um *login* no sistema operacional da máquina de trabalho do funcionário, que depois de entrar com as informações requeridas (usuário e senha, por exemplo) ele estará apto a utilizar serviços e aplicativos do próprio domínio assim como de outros domínios. Para acessar um outro serviço em um segundo domínio será necessário novamente uma autenticação, necessitando entrar novamente com um segundo usuário e senha.

Da perspectiva das empresas um serviço coordenado e integrado pode trazer diversos benefícios, entre eles:

- Redução no tempo para os usuários efetuarem *login* em domínios individuais, incluindo a redução da possibilidade de haver erro na operação de *login*.
- Aumentar a segurança através da redução de gerenciamento de múltiplas informações para autenticação.
- Redução no tempo de criação, remoção e modificação nos direitos das contas dos usuários pelo administrador.

- Aumento da segurança através da facilidade de manter a integridade dos acessos e dos serviços pelo administrador.

Esse serviço foi denominado *Single Sign-On* depois que o usuário final percebeu o impacto desse serviço. Entretanto, ambos os aspectos para o usuário final e administrador do sistema do serviço são igualmente importantes. Como pode ser observado no desenho abaixo. Quando se utiliza *single sign-on* o sistema precisa coletar do usuário todas as informações quando é feito o primeiro *login*, incluindo todas as identificações e credenciais para os outros domínios que ele tem permissão de utilizar. As informações fornecidas pelo usuário no primeiro domínio são utilizadas pelo serviço de *single sign-on* para autenticar em cada um dos outros domínios que o usuário vai interagir.



A informação fornecida no primeiro domínio pelo usuário pode ser utilizada para dar suporte nos outros domínios de diferentes maneiras:

- Diretamente: a informação fornecida pelo usuário é repassada para o segundo domínio como parte do segundo *login* (automático).
- Indiretamente: a informação fornecida pelo usuário é utilizada para recuperar outras informações a respeito do usuário e das credenciais que ele possui na base de gerenciamento do *single sign-on*. A informação recuperada é então utilizada como base para o *login* no segundo domínio.
- Imediatamente: para estabelecer uma sessão com o segundo domínio como parte do estabelecimento da sessão inicial. Isto implica que as

aplicações dos clientes são automaticamente requeridas e a comunicação estabelecida no momento da primeira operação de *single sign-on.*

- Temporariamente armazenadas e usadas no momento da requisição para o serviço do segundo domínio é feito pelo usuário final.

Da perspectiva do administrador do sistema o modelo de *single sign-on* provê o gerenciamento de uma única conta por usuário que conterá todos os domínios que serão gerenciados de uma forma coordenada e sincronizada.

O segundo domínio precisa confiar no domínio primário para:

- Corretamente identificar a identidade e as credenciais dos usuários.
- Proteger a autorização de credenciais usadas para verificar a identidade do usuário no domínio secundário evitando assim o uso não autorizado de serviços.
- As credencias de autenticação precisam estar protegidas quando transferidas entre um domínio para outro, garantindo que não há interceptação ou ataques mascarados.

Para o comércio eletrônico o *single sign-on* foi designado para centralizar as informações financeiras dos consumidores em apenas um servidor, não somente para a conveniência do cliente, mas também para aumentar a segurança limitando o número de vezes que o consumidor precisa entrar com o número de cartão de crédito e outras informações pessoais.

6.10 Web Service

Segundo a definição da Wikipédia (2007) um *web service* é uma solução utilizada na integração de sistemas e na comunicação entre aplicações diferentes. Com esta tecnologia é possível que novas aplicações possam interagir com aquelas que já existem e que sistemas desenvolvidos em plataformas diferentes sejam compatíveis. Os *Web services* são componentes que permitem às aplicações enviar e receber dados em formato XML. Permitindo que cada aplicação possa ter o seu próprio formato de dados, pois esse formato sempre é traduzido para uma linguagem universal, o formato XML. O padrão XML é de responsabilidade das instituições W3C e OASIS, que desenvolvem e regulamentam o padrão.

Para muitas empresas, os *web services* representam trazer agilidade para os processos e eficiência na comunicação entre as cadeias de produção e/ou logística. Toda e qualquer comunicação passa a ser dinâmica e segura, pois não há intervenção humana.

6.10.1 Integrador de Web services – WebEntrace

WebEntrance é um provedor de agregação de serviços proporcionando um único *login* para os usuários. Ele consiste em um subsistema de registro de usuário, um motor e um subsistema de

administração. O sistema de registro de usuário coordena as informações do registro dos usuários com os serviços correspondentes. O motor aceita a requisição dos *browsers* e passa para os serviços correspondentes, obtém-se a resposta dos serviços e retorna-as aos *browsers*. Já o sistema de Administração gerencia os usuários e os serviços dos Web Services.

6.10.2 Central Authentication Service (CAS)

O protocolo CAS foi desenvolvido pela universidade de Yale e sua implementação foi feita sob forma de código aberto. De acordo com o protocolo CAS, quando um usuário tenta acessa uma aplicação utilizando uma URL, o usuário é redirecionado para o CAS *login* através de uma conexão HTTPS, passando o nome do serviço requerido como parâmetro. O usuário digita seus detalhes de autenticação e o CAS tenta autenticar o usuário. Se a autenticação falha, a aplicação alvo nunca fica sabendo sobre a requisição e o usuário fica preso ao servidor CAS. Se a autenticação for bem sucedida, então o CAS redireciona o usuário de volta a aplicação alvo adicionando um parâmetro a URL. CAS tenta então criar um *cookie* que autoriza o redirecionamento daquele momento em diante. A aplicação então valida que o *cookie* representa um usuário válido chamando o CAS URL através da conexão HTTPS e passam à informação no *cookie* e os nomes dos serviços como parâmetros. CAS verifica que o *cookie* oferecido é válido e

é associado com o serviço requerido. Se a validação é bem sucedida, CAS retorna o usuário para a aplicação.

6.10.3 Web Service com Single Sign-On

O principal problema de se usar as arquiteturas de segurança mais comuns com Web services é que a infra-estrutura é muito distribuída e essas arquiteturas normalmente requerem que as características chaves e os algoritmos sejam implementados em todas as partes do sistema. Infelizmente todos os sistemas de segurança apresentam o mesmo nível de segurança em todo sistema, sendo assim as áreas mais criticas apresentam a mesma fragilidade que as outras áreas. Isto obviamente gera inflexibilidade porque é necessário evitar certas tecnologias em prol do compromisso com a segurança do sistema como um todo. Evitar tecnologias é geralmente impraticável e vai contra a principal razão dos Web services existirem que é ser a ponte entre qualquer tecnologia. Uma solução para isso é utilizar a arquitetura *single sign-on*.

A idéia básica da arquitetura de segurança *single sign-on* é mudar a complexidade da segurança da arquitetura para o serviço de SSO e assim liberar outras partes do sistema de certas obrigações de segurança. Na arquitetura SSO todos os algoritmos de segurança estão focados em um único servidor de SSO que age como um único ponto de autenticação de um

domínio definido. Entretanto, existe outro beneficio para utilização do SSO com autenticação/registro: um usuário terá que se logar apenas uma vez, mesmo que ele interaja com vários elementos diferentes de segurança em determinados domínios. O servidor de SSO que pode ser um Web service, age como uma película ao redor da infra-estrutura de segurança existente que exporta diversas características tais como autenticação e autorização.

Em um cenário simples, a parte autenticada primeiramente chama o servidor de SSO e pede o *token* de autenticação que o identifica em um determinado domínio. Para obter o *token*, primeiramente é necessário prover as credenciais corretas para autenticação. Há varias formas para essas credenciais, podendo ser, por exemplo, simplesmente usuário/senha ou um certificado, porem podem ser implementados novos métodos. O servidor de SSO processa a validação das credenciais do usuário utilizando a subjacente infra-estrutura de segurança e apenas então garante a passagem para o chamado da aplicação usando a autenticação de outra aplicação. Nesse cenário extremamente simples o *token* não impõe nenhuma informação especifica, é apenas a identificação única de um usuário em um escopo bem definido em certo período de tempo. Depois de chamada a aplicação o *token* é validado se repassando para o servidor SSO que então processa a validação.

As vantagens mais evidentes mesmo em um cenário tão simples são a encapsulação da infra-estrutura no servidor de SSO, facilitando a implementação, distribuição e manutenção, pois não se precisa mais implementar todos os mecanismos de segurança e características

individualmente. Alem da interface SOAP no servidor SSO que proporciona ser uma arquitetura universal.

Enquanto no cenário mais básico as chamadas ao servidor de SSO eram necessárias a cada vez que um usuário precisava de uma verificação da sua identidade. Na visão mais avançada permite que o *token* obtenha alguns valores da segurança da informação que permite a validação sem a necessidade de chamar o servidor de SSO cada vez. O *token* contém a autenticação ou a informação de autorização. Essa informação é marcada pelo servidor de SSO, para então prover o *token* à resposta com o que é confiável no servidor, sem a necessidade de verificações futuras.

Há um novo padrão para troca de informação segura em XML chamado *Security Assertions Markup Language* (SAML). Ele está atualmente sendo complementado pelo OASIS. Basicamente, a segurança da informação descrita pelo SAML é expressa na forma de indicações da afirmação de assuntos de segurança.

6.11 Strong Authentication

Existem diversas formas atualmente de se fazer uma autenticação segura, cada uma com seus prós e contras, no entanto cada uma é para algo específico e deve ser analisado caso a caso para se descobrir o que é útil para cada organização. Para uma autenticação ser considerada *strong*

authentication é necessário ter ao menos duas formas de autenticar um usuário. Para essa autenticação existem algumas categorias:

- Algo que você é: Senhas ou números de identificação pessoal (PIN) são as formas mais comuns de autenticação. Usado por você, mesmo sendo longa ou complexa, não pode ser considerada uma forma segura.
    - Autenticação baseada em autenticação – Esse cenário testa a habilidade do usuário de responder corretamente questões pessoais, tais como a cor favorita, local de nascimento ou mesmo questões financeiras a respeito da pessoa. Essa forma geralmente é utilizada como uma forma secundária de autenticar uma pessoa.
- Algo que você possui: Senha de uma única vez com uma informação essencial (*token*) – Esses dispositivos geram uma senha única a cada 60 segundos ou mais. Os usuários devem digitar a senha que foi gerada pelo servidor sincronizado de senhas para autenticação. São muito seguros, no entanto possui alguns inconvenientes, incluindo a necessidade do usuário carregar o dispositivo e de o usuário digitar a senha rápido o suficiente.
    - Certificados digitais - São identificações eletrônicas usadas como chaves públicas em sistemas (PKI). Tipicamente implementadas com *smart cards* ou chaves USB, elas podem também ser implementadas diretamente através de software hospedado no computador do usuário.

  - Cartões grade – Cada usuário possui uma matriz de números e letras personalizados. Durante a autenticação o usuário é desafiado a digitar o número ou a letra de uma célula em particular da matriz. Se as respostas coincidirem o usuário é autenticado. Os cartões grade foram desenvolvidos para serem flexíveis e de baixo custo, pois eles podem ser impressos em cartões de crédito ou facilmente distribuídos por e-mail ou fax.
  - Autenticação externa - Essa tecnologia usa dispositivos que você já possui, como celular, *pager* ou computador pessoal ao invés de dispositivos dedicados a receber uma senha. Por exemplo, uma senha é gerada e enviada por SMS ao telefone celular.

- Algo que você é:
  - Biometria – Esta categoria inclui leitores de digitais, leitura facial e ocular. Esses sistemas são geralmente extremamente seguros, pois as características biométricas são únicas em cada indivíduo. Biometria não exige que o usuário se lembre de algo e são considerados difíceis (mas não impossíveis) de serem burlados. Em algumas situações, usuários que são válidos poderão ser identificados como não válidos, quando, por exemplo, sua aparência é modificada por qualquer motivo.
- Algo conhecido sobre você:
  - Autenticação baseada em risco – Essa tecnologia emergente confia em diversos fatores passivos, incluindo geografia e uso de padrão para avaliar o grau de risco associado ao *login*.

Quando detectado que é de alto risco é o usuário é desafiado a um segundo formulário para autenticar-se. Por exemplo, um usuário geralmente efetua *login* em sua conta bancária do seu computador pessoal em casa entre às 19 e 22 horas. Se uma tentativa de *login* ocorrer às 3 da manhã de um computador de outro país, esse segundo formulário surgiria com um grau maior de dificuldade para autenticação.

- Dispositivos de identificação: Utiliza o computador do usuário como um segundo atributo para identificação do usuário. Quando o usuário se autentica pela primeira vez uma “foto” com informações públicas do dispositivo é tirada tal como o *browser* da internet e a versão. Se o usuário tentar se logar de um dispositivo com diferenças significativas da armazenadas no perfil dele, então um segundo nível de autenticação é requerido.

Na maioria desses métodos de autenticação serão utilizados com a tradicional senha ou PIN, provendo assim uma autenticação de dois fatores. Essa afirmação é especialmente verdade com esquemas baseados em dispositivos (um cartão, por exemplo). Pois cartões podem ser roubados ou perdidos e é importante ter um sistema de autenticação.

Muitas vezes não é necessário implementar a todos os usuários essas formas de autenticação, geralmente o que é feito é uma mistura de multi-fatores de segurança. Por exemplo, uma organização pode requerer o uso de um cartão e senha para acessar aplicações com alto

grau de risco, e apenas um usuário e uma senha para aplicações de baixo risco, diminuindo assim o custo de implementação de segurança da informação.

6.12 By Mechanism

É quando uma autorização é feita através de mecanismos, tais como a distribuição de chaves secretas e utilização de protocolos criptografados para transitar informações da autenticação do usuário.

Sendo assim um *authentication mechanism* define regras sobre a segurança da informação, tais como se a credencial é ou não repassada para outros processos e o formato de como a informação é armazenada.

No *Open Source Identity Management* (2007) existem diversas soluções dessa forma de autenticação, dentre elas estão:

- Kerberos
- PKI (X.509 Cert)
- *Shared Diretory*
- *Proxy Systems*
- NTLM
- Radius
- *Security Framework*

6.13 WebSSO

A web é feita de portais que agem como portas para os diversos níveis dos sítios. Alguns portais têm apenas um único foco, enquanto outros possuem diversos para todo tipo de pessoas. Um portal é freqüentemente uma interface para diversas aplicações corporativas que convergem em uma interface baseada na web, criando um ponto único de presença da organização. No começo da Web, *secure sockets layer* (SSL) eram suficientes para transportar senhas encriptografadas via *browser* pois a segurança da web era baseada na proteção das URLs, e não das aplicações. No entanto quando as aplicações e os bancos de dados começaram a ser atacados ficou claro que havia uma limitação no SSL.

Segundo Ivan Kovanov (2006), *single sign-on* é geralmente um processo que permite o usuário acessar múltiplas aplicações requerendo as credenciais apenas uma vez. O usuário primeiramente autoriza uma entidade confiável de autenticação que então garante o acesso a todos os aplicativos usando essa entidade confiável. Os sistemas de SSO normalmente preservam o estado por certo período de tempo, então o usuário pode acessar repetidamente essas aplicações sem a necessidade de se autenticar cada vez. Então a principal vantagem dos sistemas SSO é a conveniência para o usuário, alem da maior segurança, proporcionando apenas um local de autenticação para as suas aplicações. As aplicações apenas recebem

informações sobre se deixam o usuário entrar ou não. Além disso, o usuário se autentica apenas uma vez, então há uma transferência mínima de informações críticas através da rede, sem mencionar que os sistemas de SSO normalmente forçam o usuário a utilizar canais seguros de informação.

Web *single on* provê a estrutura de SSO para aplicações web. Em redes de comunicação, há freqüentemente diversas aplicações *web* e serviços designados a adicionar membros a comunidade e então pedir autenticação. Nesses casos é mais conveniente e seguro a utilização de uma infra-estrutura SSO centralizada, submetida a uma autoridade centralizada de autenticação. O exemplo mais comum do uso dessas redes são as redes das universidades, onde a maioria dos *web* SSO foi desenvolvida (Yale, Michigan, Standford,etc).

6.13.1 Visão geral

Com o *web sso*, sempre há uma autoridade central, que trabalha a autenticação do usuário. Ela pode suportar diversos mecanismos de autenticação como Kerberos, LDAP, banco de dados relacional, etc. O servidor central de autenticação pode prover também uma interface necessária para entrada das credenciais. As aplicações da infra-estrutura SSO são protegidas por uma camada chamada filtro ou clientes. Esses filtros normalmente implementados como módulos para o servidor web que está rodando a aplicação, checando se o usuário é autenticado antes de deixá-lo

acessar uma aplicação protegida. Para fazer essa verificação eles se comunicam com o servidor de autenticação diretamente ou através de redirecionamento.

Há dois cenários comuns para uma sessão de SSO (Ivan Kovanov,2006):

- *Login* primeiro – O usuário primeiro faz o login na infra-estrutura SSO e depois escolhe o serviço para acessar.
- Aplicações primeiro – O usuário primeiro tenta acessar o serviço, mas porque ainda não foi autenticado, o serviço redireciona-o ao serviço de login e depois de um login bem sucedido ele redireciona novamente ao serviço.

6.13.2 Sistemas Web SSO

Entre os diversos projetos existentes no *Open Source Identity Map* alguns são os mais importantes. Dentre eles estão o CAS (*Central Authentication Service*), WebAuth e CoSign.

6.13.2.1 CAS

O CAS é um sistema de autenticação SSO desenvolvido em java e originalmente desenvolvido pela universidade de Yale.

Sua arquitetura é similar ao modelo de Kerberos. Após uma autenticação bem sucedida no servidor do CAS, o cliente recebe um *Ticket Granting Cookie* (TGC) que permite ao cliente se identificar sem a necessidade de digitar suas credenciais novamente. Para ser garantido o acesso a aplicação o cliente precisa um *Service Ticket* (ST) do servidor CAS baseado em um TGC que foi projetado para a aplicação requerida especificamente naquele momento. O cliente então apresenta o ST (que é utilizado apenas uma vez) que é verificado e destruído em seguida. O ST não foi projetado para ser a chave da sessão, por isso a aplicação deve incorporar sua própria forma de mecanismo de registro.

Desde a versão 2.0 do CAS é possível proteger a aplicação de que usuários falsos tenham acesso a outros CAS protegidos. Uma aplicação com um ST válido obtém uma *proxy granting ticket* (PGT) que é associado com a aplicação e com o usuário que acessa a aplicação. Com o PGT a aplicação obtém um *proxy ticket* (PT) do servidor CAS e apresenta a segunda aplicação, que verifica e providencia o recurso requerido.

CAS usa o protocolo http para requerer cada um dos componentes que serão acessíveis através das URIs (Uniform Resource Identifiers). Estas

URIs tais como *login*, *logout*, validação, etc, podem aceitar um certo número de parâmetros e então retornar a informação em um formato específico.

Os requisitos para rodar o CAS são poucos, pois o servidor foi desenvolvido completamente em JAVA e rodar em qualquer *servlet container*. As autenticações do usuário são tratadas pelos tratadores de autenticação. O CAS oferece também um bom suporte ao LDAP e a certificados de autenticação. Além disso, é possível implementar autenticações customizadas escritas em JAVA

Em relação a segurança o CAS, utiliza o SSL para comunicação com o servidor em suas transações com os clientes. Os *cookies* não contém informações arbitrárias, eles possuem apenas a chave da sessão que mapeia as informações do usuário. As TGC são transferidas somente entre os clientes e os serviços alvo, dessa forma não é recomendado rodar o servidor CAS e os serviços em uma mesma máquina.

O CAS, apesar de ser um ótimo sistema, apresenta algumas limitações entre elas estão à necessidade de rodar sob um servidor Apache Tomcat, entretanto provavelmente funcione com outros *servlet containers* também.

6.13.2.2 WebAuth

WebAuth é um sistema de SSO desenvolvido pela universidade de Standfrod, e ele foi desenvolvido para ser um mecanismo de autenticação do Apache, utilizando uma estrutura já existente do Kerberos.

Sua arquitetura consiste de um servidor WebKDC (*Web Key Distribution Center*) que provê um serviço de *login*, de autenticação e diversos WAS (WebAuth-enabled Aplications Servers). O WebKDC, o WAS e os clientes utilizam *tokens* para se comunicarem uns com os outros. O WAS nada mais é que um *web server* configurado para utilizar o WebAuth para autenticar usuários. Quando um usuário não autorizado tenta acessar um serviço protegido pelo WAP um *token* é gerado para ser requerido e o usuário é então redirecionado para o WebKDC. O objetivo principal do WebKDC é lidar com as requisições de autenticação dos usuários. Todo pedido de autenticação de contém um *token* provindo do WAS, que é então utilizado pelo WebKDC para tentar fazer a autenticação. Se esta for a primeira interação desse usuário o WebKDC envia o formulário de *login* para ser preenchido pelo usuário, que depois de preenchido submete de volta. O WebKDC então verifica o usuário e gera dois *tokens*. O primeiro (*proxy token*) é usado para a autenticação futura do usuário e é guardada em forma de um *cookie*. A segunda (id *token*) é enviada ao WAS requerente através do redirecionamento do cliente. O WAS então verifica o *token* e substitui pelo *token* do aplicativo, que é guardado como *cookie* e utilizado pelo serviço para

identificar o usuário no futuro. O WebKDC também pode receber requisição em XML diretamente pelo WAS utilizando o método POST via o protocolo HTTPS. Essas requisições são utilizadas para troca de chaves ou recuperação de credenciais de um usuário. Opcionalmente o WAP pode prover autorização baseado nos privilégios de grupos LDAP. Além de recuperar atributos LDAP e colocá-los como variáveis de ambiente.

Os requisitos para rodar o WebAuth a princípio eram as plataformas Solaris e Linux, pois o servidor Apache 2 e todas as suas bibliotecas fazem parte da maioria das distribuições Linux e os módulos requeridos também fazem parte dos pacotes padrão do Linux.

Quanto a segurança o WebKDC e o WAS comunicam-se através da troca de *tokens* o que é na maioria dos casos passada através do cliente. Os *tokens* são criptografados com um par de chaves simétricas, então ninguém mais além do originador pode lê-las. Alguns dos *tokens* criptografados aparecem no URLs, mas eles são válidos somente por um curto período de tempo. A chave simétrica usada para comunicação entre o WebKDC e o WAS é chamada *sessions tokens* e possuem um limite de validade. Os *sessions tokens* são distribuídos de forma segura usando a infra-estrutura das chaves privadas. E todas as interações entre o usuário e o WebAuth são protegidos com SSL.

Uma das limitações do WebAuth é justamente o fato da falta de uma implementação para o IIS, pois os componentes foram desenvolvidos exclusivamente para o Apache 2.

6.13.2.3 Cosign

Cosign é um projeto de código aberto *web* SSO da universidade de Michigan. E ele atualmente está na versão 2.0.2a.

Sua arquitetura consiste em um sistema de SSO com três componentes – *daemon*, CGI, Filtros. Os *daemons* proporcionam as funcionalidades básicas e é são núcleo do sistema. Ele é responsável por manter o estado de todas as sessões dentro do Cosign, alem de manter o rastreamento de todos os usuários que se conectam e se desconectam ou que expiram sua sessão no sistema. O *daemon* também mantém informações sobre o serviço que o usuário acessou. O CGI age como um serviço central de login alem de prover ao usuário uma interface. Ele também é responsável por registrar cada serviço que o usuário acessa. O CGI incorpora alguns mecanismos padrões como o Kerberos, V/GSSAPI, Apache, BasicAuth e certificados X.509 além de suportar autenticações personalizadas com uma nova API que habilita o CGI a utilizar programas externos. O filtro fica na aplicação do servidor. Ele é responsável por determinar quais áreas do sítio são protegidas pelo Cosign. Se um usuário tenta acessar um serviço protegido, o filtro assume que o usuário é autenticado e passa as credenciais para o outro serviço. Caso contrário o usuário é redirecionado ao CGI para fazer a autenticação. O filtro pode requerer um ou mais mecanismos de autenticação e o usuário deve satisfazê-los todos para acessar um serviço protegido. Quando um usuário

tenta acessar um serviço protegido, o filtro checa se o *cookie* de serviço está presente. Se o *cookie* não for encontrado um novo será criado e o usuário é então redirecionado ao CGI. O CGI verifica a presença do *login* do *cookie*. Se presente, o CGI registra o *login* existente com o *cookie* do novo serviço e redireciona o usuário de volta ao serviço. Os CGI e os filtros se comunicam com os *daemon* diretamente. O *daemon* recebe do CGI informações a respeito dos usuários conectados e serviços acessados por eles. Os filtros requisitam ao *daemon* para checar pelos *cookies* de serviço.

O *daemon* e o CGI podem rodar em maquinas diferentes, entretanto não é extremamente necessário. Diferentemente do serviço que devem ser rodados em máquinas diferentes. Já o filtro foi desenvolvido para o Apache como um módulo, mas existem filtros tanto para o Windows quanto para MacOS X, além de ser possível utilizar um filtro java utilizando o Tomcat. O *daemon* suporta redundância o que torna possível rodá-lo em duas ou mais máquinas e implementar um balanceamento de carga. O CGI suporta modelos para a interface de usuário.

Quanto à segurança o *daemon* se comunica tanto com o CGI quanto com o filtro através de conexão segura TLS. Cada componente precisa seu certificado próprio, assinado e confiado por uma entidade autorizadora. Certificados auto-assinados não são suportados. Cosign implementa um *cache* local para todos os *login* e *cookies* de serviço, tanto de aplicação quanto do *daemon*. Isso é considerado de grande risco para a segurança, especialmente quando o nome do *cookie* e o valor são utilizados para dar o nome do arquivo. Existe também um limite máximo de número de arquivos

que podem ser guardados em um diretório, o que pode causar problemas adicionais.

A limitação da infra-estrutura da PKI utilizada pelos componentes do Cosign para se comunicarem é de difícil manutenção, pois requer o controle dos certificados.

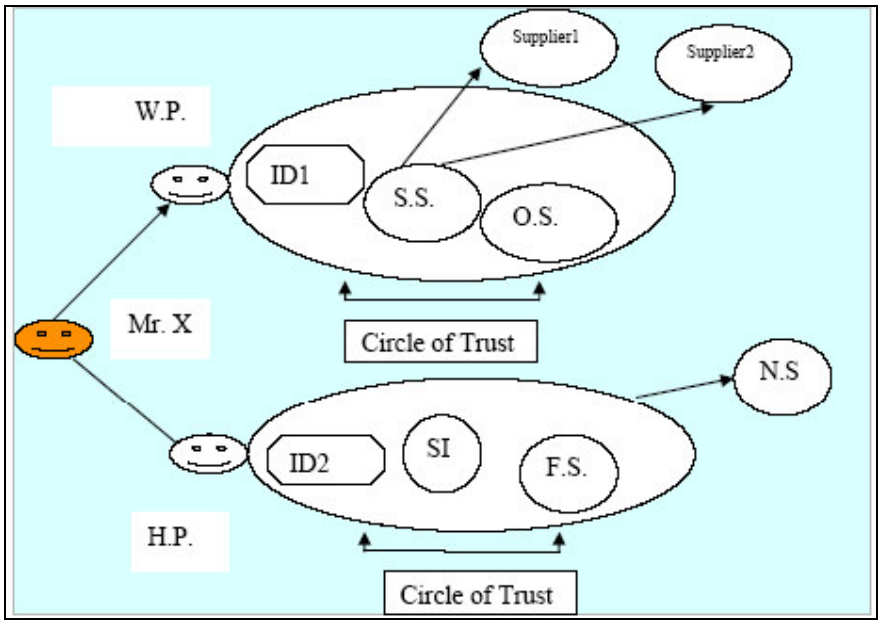
6.14. Federated SSO

O gerenciamento de identidade federada permite compartilhar informações dos usuários de diferentes provedores de identidade (organizações), retirando a necessidade dos provedores de serviço manter os usuários e senhas. Além disso, informações de autenticação dos provedores de identidade podem suprir os provedores de serviço com qualquer outra informação sobre o usuário para tomar decisão de autorização.

Certamente, solucionar o problema de um *login* único através de domínios seguros é o que levou a criação do SAML, o primeiro padrão de identidade federado. A junção de identidades, entretanto, recria um padrão que ocorre em diversas redes computacionais – a necessidade de mudar a forma de pensar o gerenciamento, segurança e o controle hierárquico da rede. Recentes depoimentos de identidades federadas realçam essa implicação da criação de identidades federadas.

6.14.1 Visão Geral

Os padrões para federações são estabelecidos pelo OASIS e pelo projeto *Liberty Alliance*, que definem mecanismos para as empresas compartilharem informações sobre as identidades entre os domínios. Como resultado de federações, as empresas podem criar identidades baseadas nas aplicações que aumentam a troca de informações sobre as credenciais. Um gerenciamento de identidade federado torna possível uma identidade autenticada ser reconhecida e propiciar serviços personalizados através de domínios múltiplos. Evitando falhas inesperadas nos servidores centrais que armazenam informações pessoais, enquanto permitem usuários associarem informações da identidade em diferentes contas. Os usuários controlam o gerenciamento de associação de contas e personalizam os serviços. Uma identidade federada requer dois componentes chaves: confiança e padrões. O modelo de confiança de identidades federadas é baseado no circulo de confiança.



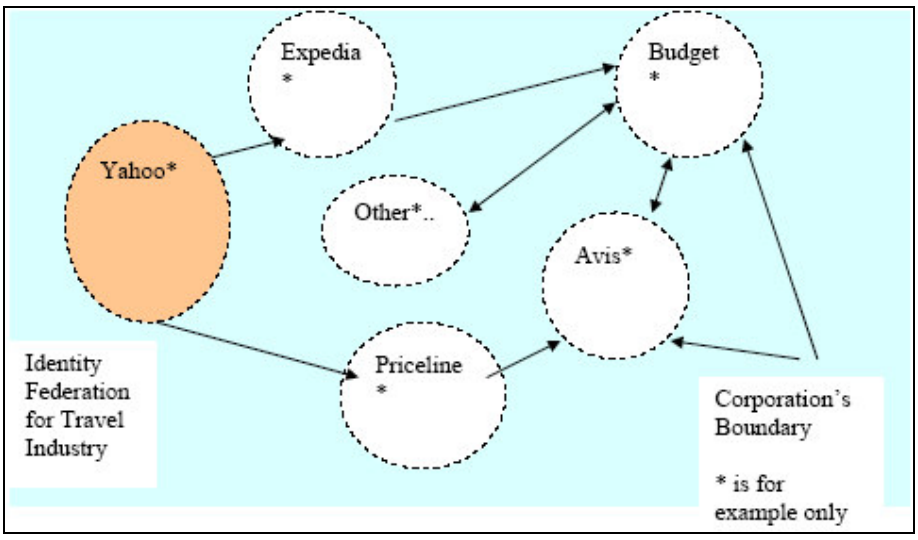
No exemplo acima a pessoa Mr. X possui uma identidade (para manter simples) e 2 perfis: Perfil de trabalho (Work Profile (WP)) e perfil de casa (Home Profile (HP)). No perfil de trabalho ele é conhecido pelo servidor de identidade ID1 através do perfil de trabalho e baseado no circulo de confiança ele possui acesso ao serviço de suprimentos (SS), o que possibilita a ele interagir com o servidor fornecedor 1 e fornecedor 2 . Ele também pode acessar os serviços do escritório (Office Services(OS)) o que permite ela utilizar email/calendário e outros serviços baseados em seu circulo de confiança. Com seu perfil de casa ele é identificado pelo servidor de identidade (ID2) e possui acesso aos serviços da família (Family Services) que pode prover outros serviços de nomes (Name Servers(NS)). Ele também tem acesso a serviços integrados (Integrated Service(SI)) que pode habilita-lo aos seus serviços de bancos, cartões de crédito, etc. Isto demonstra as identidades parciais que uma pessoa pode possuir. Circulo de confiança é um

conceito que identifica que o usuário é bem conhecido em uma comunidade de certos serviços.

6.14.2 Exemplo de uma Federação

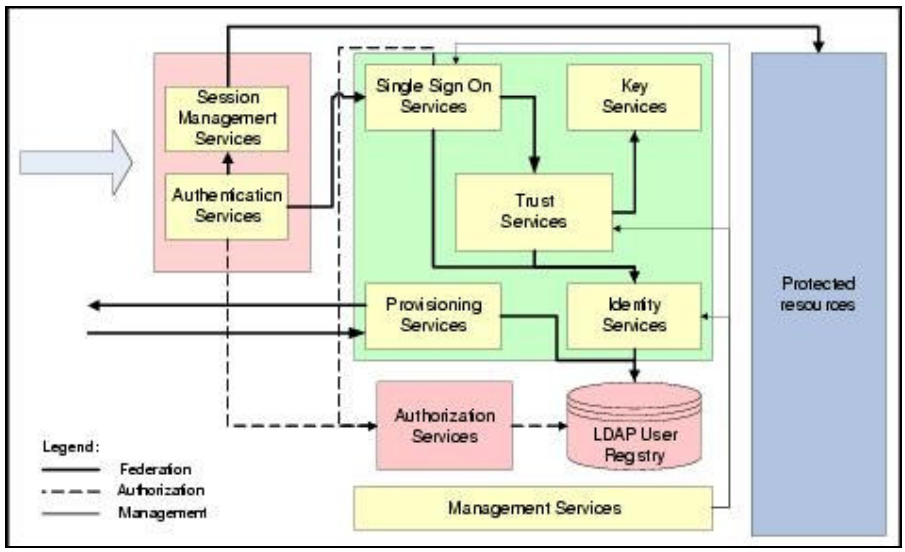
No exemplo abaixo um consumidor pode se mover entre diferentes sítios (empresas) utilizando um *login* único compartilhado. Essas empresas precisam fazer uma autenticação adequada, confiando, provendo e gerenciando as identidades.

Gerenciamento de identidades federadas (*Federated Identity Management* (FIM)), ou o gerenciamento de identidades entre corporações surgiu recentemente em resposta a necessidade dos consumidores navegarem de forma mais simples entre os sítios. FIM permite o gerenciamento de identidades entre empresas e entre organizações.



FIM entre múltiplas empresas tem se tornado normal. Além do mais, o *firewall* das grandes empresas tem se tornado pelo menos um pouco mais transparentes. Isso aumenta a necessidade para um movimento de identidades mais transparentes entre perímetros bem definidos. A IBM integrou FIM para prover mecanismos de padronização para simplificar o gerenciamento de identidades e a transformação da identidade através dos limites entre as empresas.

A IBM prove soluções de serviços de federação. Esses serviços são providos em forma de módulos para empresa. As funcionalidades do FIM são representadas logicamente pelo serviço mostrado na imagem abaixo.



6.15 Iniciativas da Indústria

Existem diversas iniciativas para simplificar os desafios do gerenciamento de identidades. As três maiores são:

6.15.1 OASIS e SAML

O SAML (Security Assertions Mark-up Language) é uma especificação baseada em XML desenvolvida pela Organization for the Advancement of Structured Information Standards (OASIS). SAML provê uma linguagem comum para os três tipos de afirmações:

a. Afirmações de autenticação: declarações a respeito da identidade do usuário.

b. Afirmações dos atributos que contentem informações a respeito da identidade de um usuário

c. Afirmações da decisão da autorização, que especificam o que é permitido ao usuário fazer em um determinado sítio.

As afirmações são emitidas por autoridades SAML (*Server Based Aplications*). Quando um indivíduo ou uma máquina requisita acessa a um recurso protegido, uma autoridade SAML emite um símbolo digital assinado que o indivíduo ou a máquina pode utilizar para pedidos adicionais para começar a re-autenticação em todo domínio que confia no SAML.

A Liberty Alliance é uma iniciativa da indústria para desenvolvimento de um gerenciador de identidade federado. Ela basicamente cobre três especificações:

1. Liberty Identity Federation Framework: Ele proporciona características para permissões baseadas em compartilhamento de atributos, descoberta de serviços de identidade, interações do serviço de segurança de perfis e modelos de serviços de identidade.
2. Liberty Identity Federation Framework: Ele proporciona características para SSO como linkagem das contas, anonimato, afiliações e opções de troca de meta dados.
3. Liberty Identity Services Interfaces Specifications: Permite serviços interoperáveis para serem construídos sobre o framework . Esses serviços podem ser simples como uma agenda, calendário ou muito sofisticados como um geo-localizador. A interoperabilidade é oferecida através do uso de contextos dependentes agregados a *schemas*. Essas especificações podem ser usadas independentes ou em combinação. A IBM recentemente uniu-se a Liberty Alliance e existe a possibilidade de sinergia entre SAML e Liberty no desenvolvimento e aceitação dos padrões.

6.16 WorkStation SSO

Essa forma de *login* é muito utilizada em empresas onde o funcionário não possui uma máquina, podendo assim efetuar a autenticação através da rede sempre utilizando o mesmo usuário e senha.

Dessa forma representa a substituição da autenticação por domínio em ambiente Windows. Permitindo que o Windows se autentique com qualquer número de autenticações existentes.

Atualmente existem dois projetos principais que estão em desenvolvimento para auxiliar nessa tarefa, são eles:

- pGina – ele permite que clientes do sistema operacional Microsoft Windows, tais como NT/2000/XP provejam apenas um método de autenticação. Esse método chama pela máquina disponível que está rodando o sistema operacional Microsoft Windows Server. Enquanto esse método trabalha bem em diversas situações, simplesmente não funciona em outras. Nesse cenário surgiu o GINA (Graphical Identification aNd Authentication) que efetua a substituição do componente DLL do Winlogon. O GINA implementa a política de autenticação do modelo interativo de *logon* e efetua todas as identificações e autenticações nas interações dos usuários. O GINA também permite a utilização de *plugins* onde ele pode ser criado para a utilização de qualquer método de autenticação.

- PingID SAML Windows *logon* – Esse projeto permite adaptar o Windows Logon baseado em SAML, sendo possível autenticar através do um servidor federado PingID.

7. Projeto Via Digital

O projeto Via Digital é um projeto que visa à informatização pública. Sendo um projeto inovador com o intuito de estimular uma nova dinâmica em torno da oferta de soluções de software livre principalmente para as prefeituras, gerando desenvolvimento tecnológico, oportunidades de negócio, emprego, renda além de capacitação e informação. Através do portal do Via Digital, reúnem-se informações, softwares, conhecimento e aproximam-se as pessoas, empresas, universidades e prefeituras, para trabalhar na construção de soluções para o setor público e de oportunidades para os empreendedores.

7.1 A oportunidade

Em pesquisa realizada pela Sociedade Softex (2004), a pedido do ITI, sobre Aplicação de SL em Prefeituras, todos os municípios pesquisados julgaram fundamental a existência de uma biblioteca pública, de livre acesso às prefeituras onde estivessem disponíveis informações importantes sobre diversos aspectos da informatização de prefeituras, ferramentas de software disponíveis, avaliação de ferramentas por parte de especialistas, melhores práticas, catálogos de fornecedores e de profissionais capacitados.

A partir dessa constatação foi que surgiu a oportunidade de criação do projeto VIA DIGITAL – a via inteligente da informatização pública. Orientado

pelo bem social com base na informatização pública e apoiada em duas grandes tendências - a componentização e o Software Livre - o projeto, primeiramente denominado FLOPREF (Free/ Livre/ Open Software para prefeituras), foi apresentado num edital da FINEP, onde obteve apoio financeiro para a sua realização.

Vencedor do Prêmio Conip em 2005 de Excelência em Informática aplicada aos Serviços Públicos, o projeto foi idealizado pelo GeNESS (Agente Softex de Florianópolis) da UFSC (Universidade Federal de Santa Catarina) e conta ainda com outros quatro participantes: CGSOFT (Agente Softex de Campina Grande) da UFCG (Universidade Federal de Campina Grande), Observatório Digital da Sociedade Softex, CenPRA (Centro de Pesquisas Renato Archer/MCT) e a empresa OpenS Tecnologia. Seu órgão financiador é o FINEP/ FNDCT (Financiadora de Estudos e Projetos/Fundo Nacional de Desenvolvimento Científico e Tecnológico).

7.2 Descrição do projeto Via Digital

O projeto Via Digital originou - se de uma pesquisa chamada - Caminho inteligente para a informatização pública: Software Livre nas Prefeituras Brasileiras: novas alternativas para a informatização da administração pública, editado pelo Softex e pelo Instituto Nacional de Tecnologia da Informação (ITI).

A pesquisa mapeia como, onde e quanto se usa software livre em prefeituras de todo o país. Analisa como as tecnologias de informação estão

instaladas nos municípios de pequeno, médio e grande porte, considerando atributos genéricos - capacidade financeira e técnica, grau de informatização, articulação - e atributos específicos - motivação, ambiente e legislação. As experiências mostram extremos como uso intensivo de software livre ou proprietário, além de expor estratégias para implementação de software livre, com ou sem indução do Estado.

A proposta do projeto via digital é o fomento ao surgimento de um ecossistema em torno de soluções de software livre destinados à gestão municipal. Este ecossistema envolve prefeituras, desenvolvedores e empresas de software e serviços, universidades e outras instituições de apoio que se inter-relacionam através de modelos de interação e de negócio próprios do domínio e da dinâmica do software livre.

O cerne do projeto é uma central de informações composta por uma biblioteca de softwares livres integráveis focados em gestão municipal e por um centro de referência, geração e difusão de informação sobre os temas pertinentes (software livre, governo eletrônico, etc.) A biblioteca será o ponto de referência para abastecer prefeituras e empresas com software livre preparando pra as necessidades comuns de prefeituras e de componentes genéticos para montagem de sistemas mais específicos, de acordo com características próprias dos municípios.

O centro de referência atende ao público em geral, mas especialmente pessoal de prefeituras, da academia, desenvolvedores, integradores e fornecedores de soluções/serviços para o poder público. As informações disponíveis incluem questões ligadas ao licenciamento, direitos autorais, modelos de negócios para empresas, modelos e ferramentas de

desenvolvimento de software livre e gestão de projetos, qualidade de software, capacitação, etc.

O modelo completo criado é potencialmente reutilizável, no todo ou em partes, em outros domínios de aplicação. Neste modelo se prevê a auto-organização dos atores em duas comunidades (com áreas em intersecção): a comunidade de desenvolvimento e a comunidade de negócios.

7.3 Comunidades Envolvidas no projeto

Existem diversas comunidades envolvidas no projeto Via Digital, entre elas estão duas bem distintas. A comunidade de desenvolvimento de software e a comunidade de negócio. Dentro dessas duas comunidades existem outras divisões segundo Delucca (2005).

7.3.1 Comunidade técnica

Esta comunidade, formada por empresas, voluntários, técnicos de prefeituras, estudantes e professores, responderia pela “alimentação” da biblioteca com código e informações técnicas sobre os softwares disponíveis. Em troca, recebem outros softwares (desenvolvidos por terceiros), interagem em uma comunidade de prática focada no tema de software para prefeituras e têm acesso a informações e serviços específicos, como avaliações de qualidade de processo e de produto, ferramentas de apoio ao

desenvolvimento colaborativo, requisitos de software e *seed code*. Em ocasiões, pode haver estímulo com premiações, contratação específica de serviços etc. Outra motivação subjetiva para participação é a ampliação das oportunidades profissionais e a aquisição e compartilhamento de conhecimento e experiências.

7.3.2 Comunidade de negócios

Formada por empresas de serviços de software, prefeituras (na ocasião de cliente) e instituições de fomento ao empreendedorismo. Nesta comunidade, as prefeituras buscam empresas ou instituições capazes de implantar e manter soluções de informatização (simples ou complexas). O efeito benéfico é a abertura de mercado de pequenas empresas que passarão a prestar serviços em sua região, a partir de um conjunto de softwares livremente disponíveis para qualquer empreendedor e em constante evolução, pela comunidade de desenvolvedores. Também abre novas oportunidades para novos empreendedores, com baixíssima barreira de entrada: sua própria competência tecnológica em compreender os softwares livres disponíveis e convertê-los em soluções para as prefeituras.

7.3.3 Interação entre os atores

Este modelo ao mesmo tempo em que oferece oportunidade de informatização para pequenas empresas – prefeituras o que traz diversos benefícios para a população – fomenta o empreendedorismo e o desenvolvimento regional com a fixação de profissionais em pequenas cidades, e estimula a prestação de serviços (implantação, treinamento, suporte, customização) de forma direta na região das prefeituras-cliente, o que dinamizará a economia regional. Além disso, procura fortalecer e qualificar uma comunidade nacional de desenvolvedores. Segundo documentação (Ecossistemas, 2005) do projeto via digital as questões transversais às duas comunidades identificadas são:

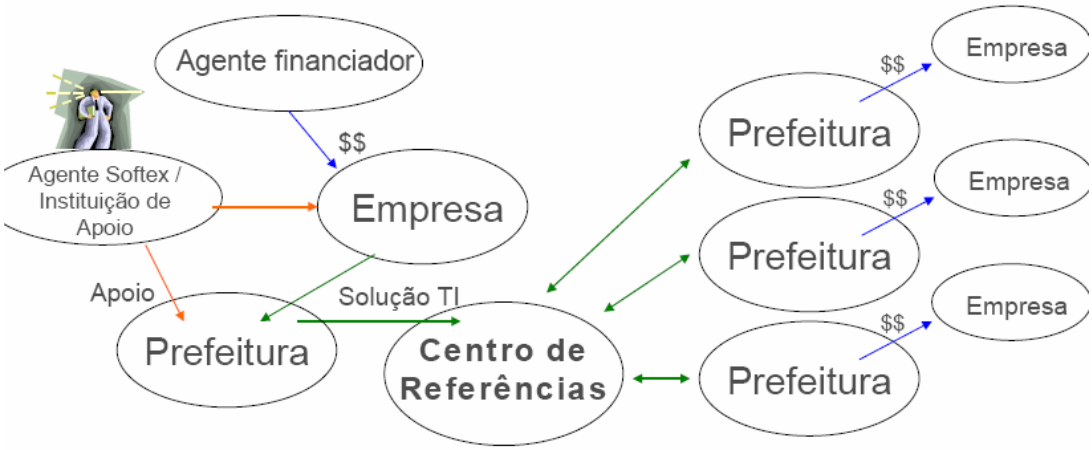
- Difusão de informação (sensibilização, divulgação, esclarecimento, conscientização)
- Suporte ao conhecimento mútuo (explicar “regras de negócio” das prefeituras para desenvolvedores, definir requisitos funcionais de software, mobilizar para busca de soluções de financiamento em parceria)
- Qualidade de software (especificações, gestão/avaliação de qualidade em processos e produtos)
- Apoio ao empreendedorismo local (incubação, incentivo ao associativismo, modelos de negócio) gerando empregos, impostos e desenvolvimento local.

- Fomento à realização de negócios (aproximação do cliente com fornecedor), inclusive aqueles “menos atraentes” aos olhos das empresas e desenvolvedores (prefeituras com poucos recursos, softwares pouco “desafiadores”).
- Suporte jurídico (questões de licenciamento, direitos autorais e contratuais).

7.4 Funcionamento do projeto Via Digital

O funcionamento do projeto Via Digital tem como uma estrutura básica ecossistemas que formam um novo arranjo formado por prefeituras, empresas e instituições de apoio que interagem e se completam através do portal do projeto, proporcionando redução de custos e desenvolvimento de soluções de TI para administração e gestão pública. Esse conceito está focado no desenvolvimento regional das comunidades de software proporcionando um compartilhamento e agregação de conhecimento através de um centro de referencia centralizado.

Esse ecossistema proporcionará uma promoção da economia e desenvolvimento local, assim como uma maior difusão da tecnologia e uma maior sinergia entre os envolvidos no projeto.



Como pode ser visto acima as interações em um ecossistema são muitas, e todas elas apresentam um peso para o equilíbrio entre os atores. Pois sem os agentes financiadores, as empresas não terão dinheiro suficiente para investir, e sem as empresas as prefeituras ficam sem soluções de TI. No entanto o centro que proporciona tudo isso é o centro de referencia que faz o canal entre as soluções de TI e as empresas e prefeituras.

No Via Digital (2005) atualmente já existem diversas prefeituras e empresas envolvidas no projeto, entre elas estão seis Estados: RS, MG, SP, PB, SC e PR.

Do ponto de vista das cidades brasileiras onde pouquíssimas tem algum tipo de gerencia de TI ou sistemas que não sejam proprietários elas poderão compartilhar o código entre si, principalmente, pois muitas necessitam dos mesmos tipos de programas muitas vezes com as mesmas funcionalidades, com pouquíssimas características distintas. Isso possibilita também um intercâmbio de dados com os mesmos sistemas federais. Alem do mais por que gastar N vezes recursos para construir o mesmo software.

Já para as empresas existem muitas vantagens também, tais como a possibilidade de evolução do código através da colaboração entre as diversas empresas espalhadas pelo país, reduzindo gastos e proporcionando uma melhor qualidade do código sem investimentos.

Sendo assim o Via Digital proporciona informação para as prefeituras, capacitação e fórum para as universidades alem de negócios para as empresas gerando oportunidade e emprego em diversas cidades.

7.5 O Repositório

O repositório proposto através do portal do via digital, visa ser um ponto de encontro de todos os softwares livres desenvolvidos e disponibilizados, se tornando assim uma biblioteca de aplicativos livres, onde será possível encontrar softwares completos, componentes de software que foram produzidos tanto em universidades, quanto por colaboradores de qualquer parte do mundo, ou até mesmo pelas empresas que fazem parte do projeto oferecendo assim alta qualidade no código dessa biblioteca.

Este repositório será ponto de referência para abastecer prefeituras com software para suas necessidades mais comuns e componentes genéricos para montagem de sistemas mais específicos, de acordo com características e requisitos específicos dos municípios.

O Via Digital também contempla uma central de referência para busca e difusão de informações sobre a dinâmica do software livre, tanto para as

prefeituras quanto para desenvolvedores, integradores e fornecedores de soluções/serviços para o poder público (organizados ou não em empresas e cooperativas) e para fomentar a colaboração entre estes atores. Aí inclui-se informações sobre licenciamentos, direitos autorais, modelos de negócio e como montar negócios, modelos e ferramentas de desenvolvimento de software e de gestão de projetos, qualidade de software, capacitação etc.

Outro ponto importante que o portal oferece é a auto-avaliação dos aplicativos depositados no repositório, onde as pessoas que utilizam os componentes e os softwares poderão opinar a respeito dos aplicativos e sugerir melhorias, gerando uma interação construtiva entre desenvolvedores e utilizadores de tecnologia. Surgindo assim uma forma de avaliação eletrônica.

No entanto talvez o maior benefício de um repositório como o desenvolvido pelo projeto via digital é justamente o fato de proporcionar um desenvolvimento colaborativo, onde diversas pessoas de diversos lugares podem cooperar em prol de um software, proporcionando ferramentas de comunidade e a hospedagem de projetos, assim como o controle de versão, *bugs*, fóruns e listas de discussões para tratar do gerenciamento do projeto.

É no repositório que os projetos serão cadastrados, gerenciados e validados. Sendo assim nele é possível fazer diversas atividades com o projeto criando-se um ambiente onde é possível trocar idéias através de fóruns e listas de discussão, alem de proporcionar a divisão de tarefas que devem ser desempenhadas assim como gerar documentação do projeto e interagir com outras pessoas através do sistema de enquete e noticias. Para

um melhor gerenciamento dos *bugs* do projeto é possível ainda gerenciar os arquivos e rastrear *bugs*.

7.6 Requisitos para um sistema único de autenticação

Diante do cenário apresentado e dos problemas que o portal do Via Digital possui, um dos requisitos necessários é que exista apenas uma base de autenticação para todos os componentes. Sendo assim não exigiria que um usuário se registrasse em duas bases de dados.

Para que os componentes sejam disponibilizados é necessário que haja certo cuidado tendo em vista que nem todas as pessoas têm permissão de acesso aos componentes.

Cada componente tem uma permissão que deve ser dada pelo super usuário determinando o que o usuário comum pode ou não usar o componente ou mesmo se ele é gerente do componente. Sendo assim a arquitetura deve prever que a partir do momento em que o usuário efetua o *login* no sistema, todo o componente que ele tem permissão de utilizar, assim como aqueles que ele tem o direito de gerenciar devem ser mostrados a ele.

8. Proposta de arquitetura para *login* único no Via Digital

A proposta para o Via Digital deve cobrir todos os requisitos citados a cima, tendo por objetivo final que o usuário ao efetuar o *login* tenha acesso aos componentes que ele tem permissão de gerenciar. Podendo assim gerenciar e utilizar os componentes que ele tem permissão. Sendo assim o objetivo é que o gerenciamento do quais componentes a pessoa tem permissão de gerencia e de utilização sejam feitas de forma rápida e fácil pelo administrador do sistema. Tornando o acesso às informações mais transparentes e mais fáceis de serem obtidas e trabalhadas, agilizando tanto o processo de registro quanto o de manutenção das contas dos usuários.

Atualmente não existem repositórios de componentes, no entanto eles já estão em desenvolvimento, muitos deles com até 70% dos projetos prontos. Porém o Via Digital ainda está em fase embrionária com perspectivas de uma grande adesão de prefeituras e participantes.

8.1 Arquitetura

O alvo das arquiteturas SSO é prover um mecanismo para os usuários se autenticarem no sistema apenas uma vez. Entretanto o sistema deve preservar a identidade e a sessão do usuário através de servidores heterogêneos como se todos os servidores fossem integrados totalmente. Amplamente, essa tarefa complexa envolve dar suporte a quatro classes de

aplicações que refletem a evolução dos modelos de arquitetura das aplicações: terminais burros, cliente servidor, web *browser*, e finalmente pequenos *browsers* encontrados em dispositivos como celulares.

Diferentes formas foram adotadas com o intuito de desenvolver soluções para SSO, mas nenhuma se tornou padrão até agora e por isso surgiram diversas arquiteturas de diversos grupos de estudo.

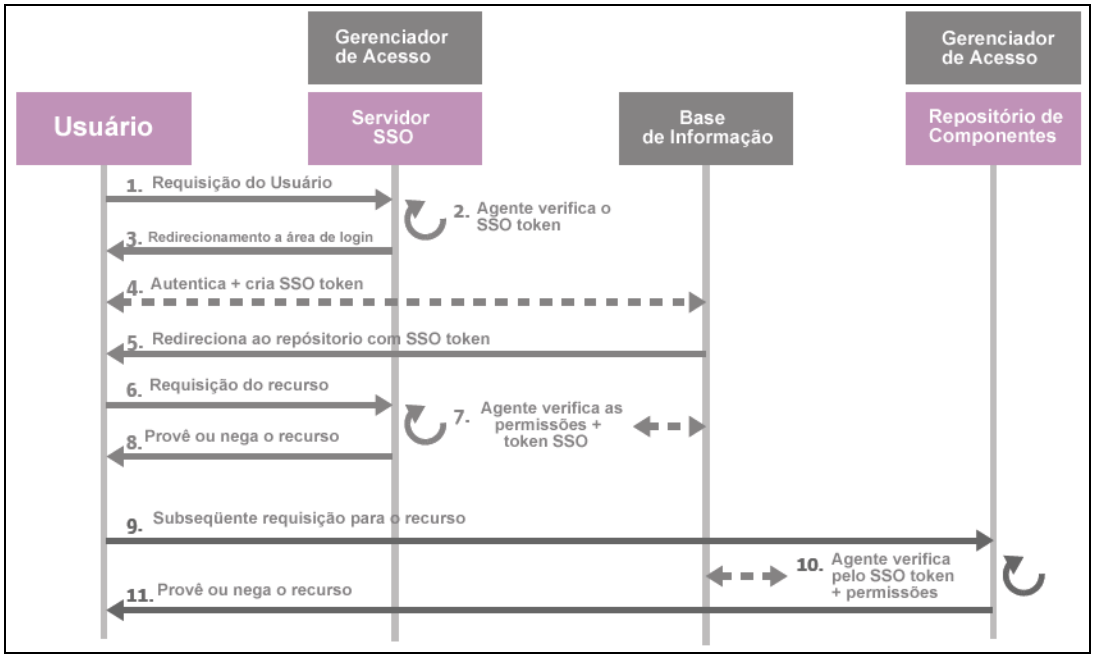
A arquitetura proposta para o Via Digital terá a adoção de um servidor de autenticação, um super usuário responsável pela manutenção das permissões dadas a cada usuário e componentes dando-se através de uma base da informação, além de um gerenciador das políticas de privacidade.

O funcionamento se dará através da autenticação do usuário no agente de gerenciamento de usuários, que poderá autorizar ou negar o acesso ao usuário. Se autorizado é feita uma verificação na tabela de informações sobre as permissões que esse usuário possui, podendo ter acesso de utilização dos componentes ou de gerencia ou ainda gerencia de alguns componentes e outros não.

O gerente de acesso de políticas faz a última validação antes de o usuário acessar os componentes validando o *token* e as políticas de acesso dos componentes. Dessa forma o usuário tem as permissões atribuídas.

8.1.1 Fluxograma

A figura a baixo foi desenvolvida para ilustrar o fluxograma da arquitetura proposta para os componentes do projeto Via Digital.



Fonte:Elaborado pelo autor.

1. O usuário faz uma requisição de acesso à página que contém os componentes.
2. O agente faz a verificação do *token* no servidor de SSO.
3. Depois de feita a verificação o servidor de SSO faz o redirecionamento para a área de *login*.
4. Nesse momento é feita a autenticação e a criação do *token* baseado nas informações da base de informação.
5. O usuário é redirecionado ao repositório com o SSO *token* criado no passo 4, contendo suas permissões.

6. O agente do usuário faz a requisição do recurso ao servidor SSO
7. O agente verifica as permissões concedidas a ele na base de informação gerada pelo administrador do sistema, alem da verificação do token SSO.
8. O servidor SSO devolve a permissão de acesso ou a negação para o usuário
9. O usuário solicita a utilização do recurso (um componente do repositório).
10. O agente verifica o *token* do SSO alem das permissões da base de informações
11. O agente provê ou nega o acesso ao componente, caso de a permissão essa pode ser de gerente ou de usuário dependendo do que estiver na base de informação.

8.1.2 Cenário

Esse fluxograma mostra a utilização e a seqüência de ações que ocorrem quando um usuário fizer o *login* para gerenciar ou utilizar um componente. Nele são apresentadas as peças chaves: usuário, servidor SSO, base de informação, repositório de componentes.

8.1.3 Usuário

É um típico usuário dos componentes, que pode ter acesso a um ou diversos componentes, podendo inclusive ser gerente de um componente.

8.1.4 Servidor SSO

Um servidor que faz o gerenciamento dos usuários e o controle de acesso à informação dos usuários. A proposta é utilizar o projeto OpenSSO da Sun Microsystems, que possui o código aberto e está em pleno desenvolvimento por toda a comunidade de software livre, tendo como grande apoiadora a própria Sun.

8.1.5 Base de Informação

Essa base é alimentada pelo super usuário tipicamente caracterizado pelo administrador do sistema que atribui à permissão de acesso de cada usuário que tem acesso a base. Dessa forma essa base de informação é um banco de dados com informações de acesso dos componentes, sendo que deverão existir basicamente três grupos de acesso a esses componentes: usuário, gerente, sem permissão.

- O usuário terá acesso ao(s) componente(s), no entanto não poderá fazer qualquer tipo de alteração neles.

- Gerente terá plenos poderes de alteração e utilização dos componentes, podendo inclusive atribuir permissões.
- Sem permissão quando o usuário não tem permissão de acesso ao componente, não podendo assim modificá-lo ou utilizá-lo.

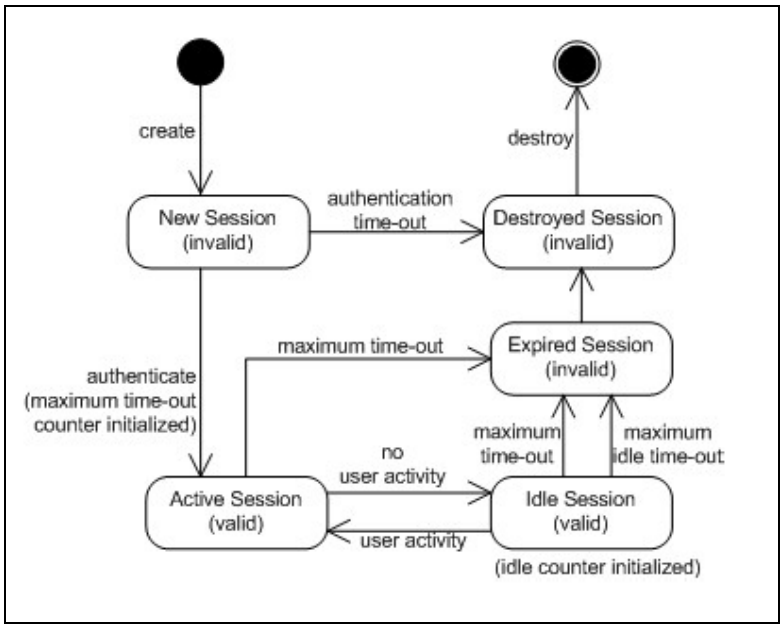
8.1.6 Arquitetura do OpenSSO

O sistema OpenSSO pode ser implementado de diferentes formas. O resultado de qualquer implementação deverá ser a habilidade de executar os requisitos que todos os sistemas possuem através da arquitetura proposta.

8.2 Proposta de implementação

O cenário central do SSO é a presença de uma sessão transitória do usuário que é criada no momento da autenticação do usuário e é destruída depois que certas condições são cumpridas. Essas condições podem incluir regras tais como um tempo máximo pra a sessão do usuário, um tempo máximo da sessão inoperante ou uma sessão não utilizada, provendo ações como deslogar ou destruí-la pelo usuário ou administrador, além de outras condições necessárias no ambiente da aplicação.

Esse cenário pode ser visto abaixo.



Fonte: System Architecture Open Web Single Sign-On, 2005.

Como uma sessão de um usuário pode circular através de vários estados dependendo do ambiente, e alem disso o número de estados distintos em que uma sessão atravessa (os estados chaves) podem ser válidos ou inválidos. Se forem válidos, a sessão representa uma sessão do usuário autenticada, ela pode representar uma sessão anônima, ou uma sessão que tenha expirado.

A sessão do usuário pode ser associada com cada pedido vindo do usuário de modo que a aplicação ou o agente SSO possam afirmar a identidade do usuário antes que o pedido possa prosseguir ao recurso desejado. Alem das diversas outras formas que existem as três a seguir representam as mais promissoras:

1. O mapeamento de um atributo esperado tal como o endereço remoto do usuário para ser lidado dentro da sessão do usuário.

2. Utilização de um parâmetro requerido tal como um token URL ou um formulário embutido escondido na sessão do usuário.
3. Utilizando uma requisição http (um cabeçalho) na sessão do usuário.

.

Cada um desses três mecanismos tem seus pontos fortes e pontos fracos e suas funções são comparáveis. O primeiro mecanismo promete simplicidade, mas pode faltar habilidade para assegurar unicidade. O segundo e o terceiro tornam difícil o gerenciamento do mapeamento da infra-estrutura para prover flexibilidade e unicidade, mas o processo de fazer o mapeamento torna a estrutura vulnerável a erros intencionais ou acidentais.

Esta proposta é baseada na terceira proposta em que existe um cabeçalho http, que é utilizado para carregar um *cookie* que será manuseado.

Um exemplo prático da imagem acima seria a manipulação de um componente em um sistema de almoxarifado. Para fazer esse processo ele seguiria os seguintes passos:

1) Cria-se uma nova sessão logando no sistema.
2) É feita a autenticação desse usuário (nesse momento já é inicializado a variável que verifica o *time-out*)
3) A sessão foi aberta, e nesse momento o usuário pode fazer todas as atividades permitidas dentro do sistema.
4) Caso o usuário deixe de utilizar o sistema automaticamente classifica essa sessão como inativa
5) Quando a sessão está classificada como inativa ela pode voltar a ser ativa ou expirar e ser finalizada.

Essa seria o cenário de utilização diária dos funcionários do setor, ilustrando como o processo parece ser muito simples, pois exige apenas o conhecimento de uma senha e de um usuário para efetuar o *login* inicial.

8.3 Requisitos

O OpenSSO foi desenvolvido totalmente em Java e pode ser rodado em qualquer servlet container como por exemplo o Tomcat, podendo ser executado portanto em qualquer plataforma tanto Linux quanto Windows. Além disso ele é suportando tanto por SAML, ID-FF, ID-WSF e PHP.

No âmbito do projeto via digital, será necessário que se utilize uma maquina (servidor) separada dos outros aplicativos, tendo em vista que o servidor de *login* é o alvo primário dos atacantes de uma rede. Esse servidor irá fornecer os dados dos usuários aos outros aplicativos do sistema, proporcionando informações sobre os usuários de forma centralizada facilitando o gerenciamento dos usuários do projeto.

8.4 Segurança

As informações associadas à autenticação e sessões do usuário devem ser manuseadas pelo OpenSSO em um gerenciador seguro e confidencial, permitindo que apenas os administradores com privilégios

possam acessa-lo impondo limites onde é necessário. Essa preocupação é o principio básico para o compartilhamento de serviços de identidades através de múltiplas aplicações para estabelecer confiança em um gerenciador seguro.

Outro fator importante é o gerenciamento da base de informações, que deve ser acessada apenas pelo administrador do sistema, que deverá atribuir à classificação do usuário a um grupo de componentes ou a componentes individuais, dessa forma apresentando uma facilidade no gerenciamento dos componentes.

8.5 Limitações

O sistema OpenSSO poderá ser aumentado quantos níveis forem necessário em prol de incluir não somente o gerenciador de componentes mas também outras aplicações web que poderão ser incluídas aos poucos, sem necessidade de se pensar na segurança de *login* pois já está implementada. Sendo assim essa arquitetura visa uma generalização para outras aplicações e não somente para o repositório de componentes.

Outro fator importante é que não existe qualquer limitação quanto ao número de usuários que poderão utilizar o sistema, por se tornar fácil o gerenciamento dos usuários e suas permissões.

No entanto um fator limitante do openSSO é o fato da necessidade de uma grande estrutura para rodar o sistema, além de uma máquina específica

para esse servidor o que significa mais uma máquina rodando, com todas as implicações disso.

9. Conclusões e Trabalhos futuros

Juntamente com os Gerentes de Identidade, muitas arquiteturas estão sendo desenvolvidas para que os recursos computacionais interajam com as pessoas e possam, de forma fácil e prática utiliza-los. É evidente que essa integração dos sistemas e as tecnologias alinhadas aos processos de negócio contribuem para o sucesso das organizações, e é o que está ocasionando investimentos tanto de grandes empresas quanto das universidades.

O uso de um dos diversos modelos de gerenciamento de identidades apresentados nesse estudo visou encontrar a melhor forma de se disponibilizar esse serviço aos usuários do repositório de componentes do projeto Via Digital para que não precisem se autenticar diversas vezes ao baixar um componente ou mesmo gerencia-lo. Para isso foi utilizado uma arquitetura promovida pela SUN chamada OpenSSO que alem de ser muito flexível proporciona uma grande robustez.

Outros trabalhos futuros podem ser realizados como complemento deste, tais como, a implantação do sistema de um projeto piloto para que através desse seja feita uma primeira avaliação juntamente com os administradores, visualizando e ajustando o sistema para uma melhor utilização.

10. Referências Bibliográficas

(1) Akenti. Disponível em:

http://dsd.lbl.gov/security/Akenti/, acesso em 07/06/2007.

(2) Authentication and Authorization Infrastructure.

Disponível em: http://www.switch.ch/aai/about/, acesso em 07/06/2007.

(3) Authorization Working Group.

Disponível em: http://grid-auth.infn.it/, acesso em 07/06/2007.

(4) Bandit Project.

Disponível em: http://www.bandit-

project.org/index.php/Project_overview,acesso em 07/06/2007.

(5) Concordia Project.

Disponível em:http://projectconcordia.org/index.php/Main_Page, acesso em 07/06/2007.

(6) Coalition for Networked Information.

Disponível em:http://www.cni.org/projects/authentication/authentication-wp.html, acesso em 07/06/2007.

(7) DIM 2006.

Disponível em:http://www2.pflab.ecl.ntt.co.jp/dim2006/DIM2006_CFP.pdf, acesso em 07/06/2007.

(8) Federated Identity Management.

Disponível em:http://www.securitydocs.com/pdf/2782.PDF, acesso em 07/06/2007.

(9) Funambol Open Source.

Disponível em:http://www.funambol.com/opensource/, acesso em 07/06/2007.

(10) ID OSS Map.

Disponível em:http://docs.safehaus.org/display/HAUS/Id+OSS+Map, acesso em 07/06/2007.

(11) Introdution to single sign on.

Disponível em:http://www.opengroup.org/security/sso/sso_intro.htm, acesso em 07/06/2007.

(12) KOVANOV I. 2006.

Disponível em: <http://www.cesnet.cz/doc/techzpravy/2006/web-sso/web-sso.pdf>. acesso em: 07/06/2007

(13) OpenID at Sun.

Disponível em: https://openid.sun.com/opensso/docs/FAQ.jsp#what, acesso em 26/06/2007.

(14) Open SPML.

Disponível em:http://www.openspml.org/, acesso em 07/06/2007.

(15) Open Mobile Alliance.

Disponível em:http://www.openmobilealliance.org/about_OMA/index.html, acesso em 07/06/2007.

(16) Open LDAP.

Disponível em:http://www.openldap.org/pub/kapurva/proxycaching.pdf, acesso em 07/06/2007.

(17) Open SSO.

Disponível em:https://opensso.dev.java.net/, acesso em 07/06/2007.

(18) Open SSO Project.

Disponível em:

https://opensso.dev.java.net/files/documents/3676/19701/architecture.pdf, acesso em 07/06/2007.

https://opensso.dev.java.net/public/use/docs/pdf/index.html, acesso em 07/06/2007.

https://opensso.dev.java.net/public/agents.html, acesso em 07/06/2007.

https://opensso.dev.java.net/public/use/docs/opensso/pdf/guidetoauth.pdf, acesso em 07/06/2007.

(19) OpenSSO Soluções para Gerenciamento..

Disponível em:http://blog.urubatan.com.br/2006/09/07/opensso-soluo-para-gerenciamento-de-identidades-open-source/, acesso em 07/06/2007.

(20) PERMIS.

Disponível em: http://sec.cs.kent.ac.uk/permis/documents/concept.shtml, acesso em 07/06/2007.

(21) PFITZMANN A. e HANSEN M., 2004. Disponível em: <http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.18.pdf>. acesso em: 07/06/2007

(22) Shibboleth Project .

Disponível em:http://shibboleth.internet2.edu/, acesso em 07/06/2007.

(23) Single sign-on using cookies for Web applications. Disponível em: http://ieeexplore.ieee.org/iel5/6520/17409/00805192.pdf?tp=&isnumber=&arnumber=805192, acesso em 07/06/2007.

(24) The Friend of a Friend (FOAF) project.

Disponível em:http://www.foaf-project.org/, acesso em 07/06/2007.

(25) The PAPI AA Framework.

Disponível em:http://papi.rediris.es/, acesso em 07/06/2007.

(26) VIPIN, S. publicado em IEEE, 1999.

Disponível em:

http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel5/6520/17409/00805192.pdf

acesso em: 07/06/2007

(27) Wikipedia.

Disponível em:http://en.wikipedia.org/wiki/Authentication ,acesso em

07/06/2007.

http://en.wikipedia.org/wiki/Strong_authentication acesso em 07/06/2007.

(28) What is.

Disponível em:

http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci211621,00.html,

acesso em 07/06/2007.

(29) What is Aiakos.

Disponível em:http://www.aiakos.net/, acesso em 07/06/2007.

(30) Webopedia.

Disponível em: http://www.webopedia.com/TERM/P/provisioning.html, acesso

em 07/06/2007.

(31) What is authentication,

Disponível em:
http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci211621,00.html, acesso em 07/06/2007

(32) What is provisioning,
Disponível em:http://www.webopedia.com/TERM/P/provisioning.html, acesso em 07/06/2007

(33) What is authentication,
Disponível em:http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci211621,00.html, acesso em 07/06/2007

(34) Wikipedia - ,
Disponível em: http://en.wikipedia.org/wiki/Authentication, acesso em 07/06/2007