電子流通基盤 SPAgent の目指すもの

～ 新しいプログラミング・パラダイムの提案と実践 ～

高橋俊成[†] 梶浦正浩[‡] 後藤哲也[‡]
秋山浩一郎[†] 半田豊[‡]
takahasi@isl.rdc.toshiba.co.jp

[†]㈱東芝 研究開発センター コンピュータ・ネットワーク・ラボラトリ
[‡]㈱東芝 iバリュー・クリエーション社 WebTop 事業部

インタネットが普及しつつある，と誰もが思う世の中になった．しかし，実生活での恩恵は意外と少ない．家庭の端末でのバンキングサービスは20年前にもあった．役所に足を運ぶ回数も減らない．紙の消費量も増えた．良く出来た通信販売の WEB サイトより昔ながらのぶ厚い通版カタログが有益である．通信コストの高さを指摘する声があるが，ならば無料で無限のネットがあったら何をするのか建設的な提案を聞くことも無い．単なるマルチメディア配送システムとしての活用は誰でも思い付く．しかし，それを社会基盤と呼ぶのはあまりに寂しい．

今日もまた講演者は雨に凍えた体で，いつ到着するか情報の得られないバスをひたすら待ちつつ，IT技術が進歩したように見えて実は趣味の開拓者が主導で動いただけの20世紀を振り返って愚痴るのであった．

こんな極めて曖昧な motivation を抱きつつ，インタネットの原点に戻ってセキュリティシステムの実装を考えるべく，電子流通基盤モジュール SPAgent(※1)[1] を設計し，完成させたので報告する．

1. はじめに

近年インタネット等を通じたデジタルコンテンツ／情報流通が広まっているが，それが社会基盤システムと呼べる程には一般化しない理由のひとつは，主としてネットサーフィンを目的に始まったインタネットブラウザの世界を前提にしたサービス提供の形態にあると考える．

標準化の努力は W3C[8] 等でなされているが，現実には小数企業の節操無く続く改革を追認し軌道修正を謀るにとどまざるを得ない．中でもセキュリティや著作権保護の仕組みについては，個別業界団体等による検討仕様が乱立し，社会基盤としての統一的なコンセプトが存在するわけではない．

一方では，それら製品群の流れを無視して独自の専用クライアントや回線限定で提供されるサービスもあるが，導入の障壁が高くサービス利用法が統一できないなどの問題がある．既に世の中は“インタネット”＝“WWWブラウザによるサービス”であり，これを否定するのは難しい現状である．

我々は，ブラウザとは独立に安全性を確保しつつも，既存のブラウザと連携して動作する電子流通基盤 SPAgent を開発した．SPAgent は暗号化，認証，データ圧縮，コピープロテクション，回線切断からの復旧といったデジタルコンテンツ流通一般の課題を解決し，ブラウザに新たな機能を追加するセキュリティ・モジュールである．ユーザは好みのブラウザを使いながら無意識に SPAgent のセキュリティ機能を利用することにより，一定の安全性が保証される．また，サービスプロバイダは一般の CGI(Common Gateway Interface)プログラミングと同等の手間でサーバ／クライアントの連携した新型サービスを提供することができる．

2. WWW を使ったサービスの現状を考える

現状広まっている WWW を用いたサービスの基本は「ある信頼できるサーバ(サービスプロバイダ)」と，「不特定多数のユーザ(顧客)」との情報交換にあると仮定され，その安全性は通信の暗号化とサーバ認証によって保証される．典型的には SSL(Secure Socket Layer)が良く用いられ，サーバの証明書に署名をする CA(Certification Authority)の公開鍵を予め(また後に)知るインタネットブラウザが安全に(偽造されずに)ユーザに配布/利用されることを暗黙の前提としてサービスが展開されている．

† TAKAHASHI Toshinari, et al.: Computer & Network Systems Laboratory, R&D Center, Toshiba Corp.

‡ KAJIURA Masahiro, et al.: WebTop Services Division, iValue Creation Compy, Toshiba Corp.

※1) SPAgent™ - (Service Providing Agent) エスピイエイジェント

この方法では，WWWサーバの信頼を，それを運営している団体の名称の信頼とリンク付けし人間系の力で保証しようとする．しかし，現実にはWWWサーバの運営主体とサービス提供主体は一致するものではなく，全てのコンテンツおよびリンクの内容が保証されていると期待することは非現実的である．またサービスが多様化すると個人情報の提供や個人のPC内情報の公開／書換等を余儀なくされるが，それをユーザが認めるか否かはあくまでサービスの必要性とのバランスによって個別に決まるものであり，単純にサーバが信頼できる／できないのレベルの問題ではない．

この問題は，サービス提供に(専ら小さくて数の多い)プログラム配布が伴う場合，より顕著に現れ，多くの場合ユーザは安全性を全部捨てるかサービスを全く受けないかの選択しか与えられていない．

一方，サーバからもユーザを特定してサービス提供を行うケースが増えている．これについては多くのサイトでCookieやHTTPのBasic authenticationや個別に発行したパスワードなどを使っているが，肝心の秘密鍵の保存が不安全であったり，沢山のパスワードが覚え切れないなどの問題がある(そもそもencodeされたcookie情報の送信可否などユーザには判断ができない)．個人に発行された証明書を使う方法もあるが，単にユーザを「区別」したいだけの目的としてはあまりに大げさである．例えばSSH(Secure Shell)[10]のように秘密鍵をパスワードで暗号化して保存するようなシンプルなモデルを基にシステム構築したいものである．

これら多くの問題が生じる理由のひとつは，安全性に関してブラウザ組み込みの機能に依存しすぎていることにあると考え，ブラウザはGUIに過ぎないという基本を改めて思い出し，3.7節で述べる信頼モデルに基き，基盤モジュールを構築した．

3. 実装

SPAgentは既存のhttpd(WWWサーバ)と，既存のブラウザを使い続ける形で，独立したセキュリティ管理と新機能を提供するものである．本章では，その構成および設計ポリシーに関して説明する．

3.1. システム構成とデータの流れ

SPAgentはクライアント(現在Windows98等)に常駐する'SPAproxy'と，httpdから起動されるcgiプログラムである'SPAserv'の対が基本となって動作する．両者間でSPAgent独自の暗号化／認証／圧縮が行われ，独自のデータが既存のHTTP通信路を流れる．図1はその構成を示したものである．

サーバ側のアプリケーション(通常のシステムではCGIプログラムで記述される部分に相当する)は'SVP' (Service Program)と呼び，後述するようにパッケージとして認証の単位となっている．SVPは，ブラウザからSPAserv経由で(SPAservというcgiプログラムをSVP名を引数として)起動される．SVPはCGIプログラム準拠の独自拡張データである'SPA-HTML'をSPAservに返し，SPAservはそれを適宜変換し，WWWサーバ経由でリクエストしたブラウザ(またはSPAproxy)に返す．

SPAproxyはブラウザから見るとproxyとして存在

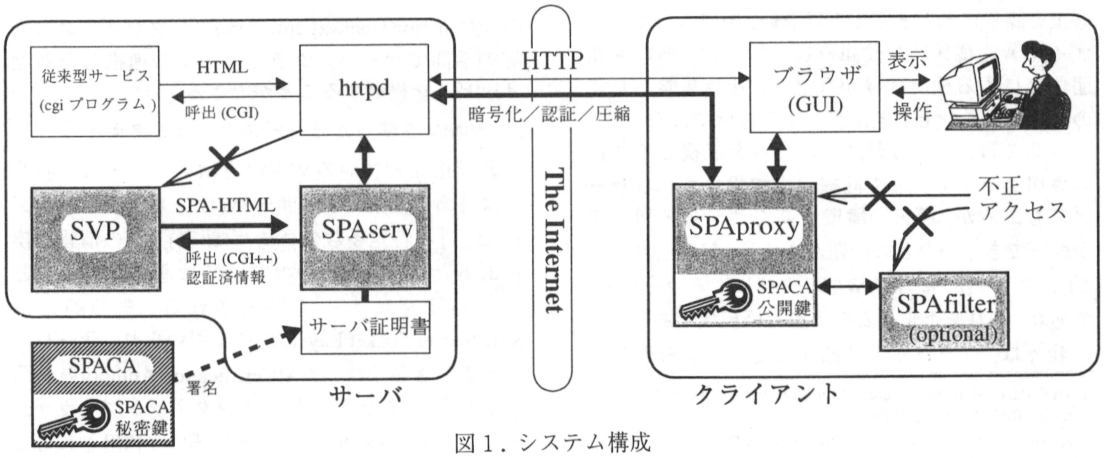

図1．システム構成

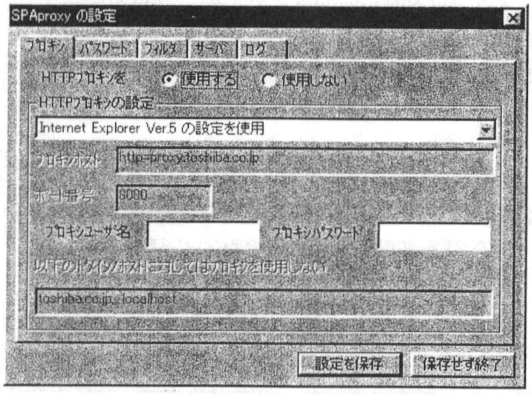

図２．SPAproxy の通信の設定画面

し，通信が行われる．つまり，SPAgent を用いる通信は localhost(127.0.0.1) の特定ポートに送信されるよう，HTML内のリンクのURLが自動記述される．

ブラウザが元々一般のproxyを使って通信している場合，本来のproxy通信との整合はSPAproxyの設定画面で行うが，多くの場合，既に使用中のブラウザの設定を継承して自動設定される(図2).

またSPAproxyは，クライアント内で別のプログラムを起動する機能を持つ．起動されるプログラムは予めインストールされている 'SPAfilter' と呼ぶモジュール(現実装ではWindowsのdll)に限定されている．SPAfilterはHTML出力データへのフィルタとして実装されていることからこの名が付いている．

以上各モジュールの処理の流れは，SVPの出力であるSPA-HTMLによってコントロールされており，SPAproxyはブラウザやSPAfilterとのセッションを管理している．つまり，特定のSPAfilterを起動させる(怪しい)web pageを作成したり，特殊な引数を与えてSPAproxyの想定外のPOSTデータを送り付けたりする細工によって，SVP作成者の意図しない動作を引き起こされる可能性を排除している．

SVPの出力するSPA-HTMLの具体例および，それがSPAservによって変換された後の形式を示したのが図3aおよび図3bである．

SPA-HTMLの文法は，既存のHTMLタグを変更せず，属性定義する形での独自拡張であるため，将来HTML文法の変更があっても柔軟に対応できる．

なお現在，サーバはFreeBSD/ Linux/Solaris，クライアントはWindows95/ 98である．

```
<HTML>
<BODY spabackground='/foo/soapston.jpg'>
<A target=_blank spahref='/foo/bar.html'>
bar のテスト</A>
</BODY></HTML>
```

図３a．SPA-HTML の記述例

```
<HTML>
<BODY background='http://127.0.0.1:31400/?SPpRK
OnDy2Wi2Y1Xu10eLA.AkAEBA.CEB.AcQa.GQtE5M9VesqnY
utBOEmH4L7mAnsxBOEKDli.iKes7exdklE.....Xtyslp'>
<A href="http://127.0.0.1:31400/?SPpR=xkASiSkLN
B4Q2VHaMP5GA.AiAEBA.CEBA.AbQA.Ggg2EGOOVIBXlI1oJ
JQg3pVkzMkgBLDgIIlk.....Kdleyde" target=_blank>
bar のテスト</A>
</BODY></HTML>
```

図３b．ブラウザの受信する HTML

3.2. サーバ認証

ユーザが各SPAservを初めて訪れた際に，SPAservは己のサーバ証明書をHTML内に埋め込んで送出する．その証明書には'SPACA' (SPAgent Certification Authority)のデジタル署名が付いており，SPAproxyが予め持っているSPACAの公開鍵でそれを検証する．つまり，本システムにおけるサーバ認証は技術的には従来のブラウザと同じであり，SPACAが「SPAservを使ったサービス」を運用するサーバの運営主体を認証するという運用上の違いがある．

3.3. クライアント認証

本システムでは，クライアント証明書を用いない方式を選択した．

あるユーザがあるSPAservに初めてアクセスした時に，SPAuserID(※2)が生成される．SPAuserIDはSPAgentのクライアント認証の単位であり，対応する認証キーと共に各サーバごと／各ユーザ毎にそれぞれサーバ／クライアント間で共有される番号(認証キーのindex)である．ユーザのPC内ではこの認証キーはユーザのPassphrase等で暗号化後に保存されており，起動したSPAproxyがSPAgent機能を初めて使う時に復号される(図4).

この認証の単位は，クライアントOSの持つユーザの概念とは独立であり，自由に設定することができる．元々はユーザ管理が曖昧に運用されるOS

※2) SPAuserID -- 文献[1]ではSPAproxyIDと呼んでいたが，SPAproxy 固有の値であるという誤解を招くため本論文ではSPAuserIDと呼ぶことにした．

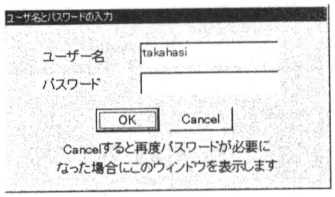

図4．PassPhrase 入力により SPAproxy を活性化

に対応するための仕様であったが，結果的にユーザ管理の単位として適当な選択であったと思う．

各SPAuserIDおよび対応する認証キーは，サーバ／クライアント間で永遠に共有し続ける．つまりSPAgentのクライアント認証は，従来のブラウザで，比較的小数(最大百サイト程度)の“高級”サーバが，それぞれ比較的小数のユーザ(最大百万人程度)に対して“特別の”サービスを提供することを簡便に実現するための仕様であり，全てのWWWサーバをリプレイスすることを狙った世界征服仕様ではない．

3.4.アクセスコントロール

WWWでのサービスが高度化すると，従来用いられてきた"信用できるサーバ"や"データの送信を許可する"，"プログラムの実行を許可する"等によるアクセスコントロールは限界になる．例えば，パソコンメーカの修理依頼ページに自分のPCの設定ファイルの内容を書き換えさせることと，初めて訪れたWWWサイトに乱数のようなCookieを送ることを許可することと，どちらが安全かと言うことはできないし，また設定もできない．

SPAgentシステムにおけるアクセスコントロールは，“このサーバ”(SPAserv)の運営する“このサービス”(SVP)が“このプログラム”(SPAfilter)の“この機能”を実行することを許可するか否かという概念に基いている．SPAfilterは原則として例えば"ローカルファイルを読み出す"といった単一機能を提供するため，ユーザがアクセス許可を出す際の判断材料が明確になっている．

例えば図5は“pcrescue”という名前のサーバが運営する“PCrescue”という名前のSVPが“registry”という名前のSPAfilterを起動することにより(Windowsの)registry情報を読もうとしたときに出る警告ダイアログである（通常は表示しない設定にする)．

これらアクセス権(ACL)の設定は，SPAproxyの設定画面(図6)によりユーザが行えるが，この煩雑な設定をユーザが一度に行うのは現実的でないので，初めてそのSPAfilterが呼ばれた際に意思を確認する機会を設けた(図7)．この例を見ると判る通り，各SPAfilterの持つ“機能”に関する設定の粒度は一般に各SPAfilterごとに若干異なる．

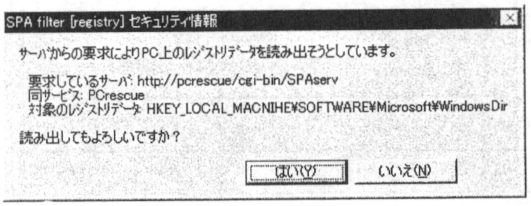

図5．'registry' SPAfilter を起動する際のダイアログ

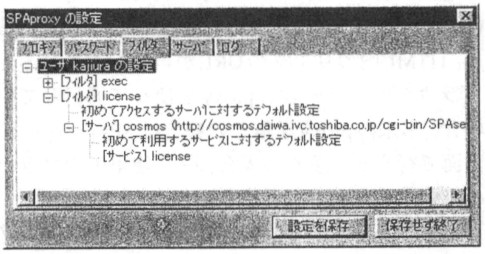

図6．SPAfilter 起動に関する ACL 設定画面

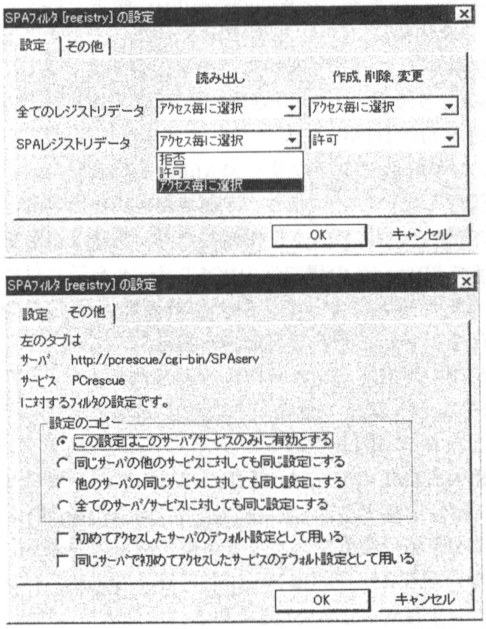

図7．SPAfilter の ACL 設定を on demand で行う

3.5. クライアント内認証

従来，認証とは主にサーバとクライアントPCの間で行うものだという考え方により設計されたシステムが主流で，ウイルスの発生を容易にしていた．近年ではライブラリ(dll)の読み込みをセキュアにするなどの実装が徐々に広がりつつある．

クライアント内のプログラムの相互動作の安全性を確保することは，サーバ／クライアントの認証と同等に重要であると考えられる．なぜなら，ネットワーク接続されたPCは，IP経由でのアタックの可能性があるだけでなく，PCローカルで実行されるプログラムが直接PC資源にアクセスすることもできるからである．クライアント内であっても本来，暗号技術に基く相互の認証下で動作しなければならない．

SPAgent システムでは，SPAproxy，ブラウザ，SPAfilter，さらにそこから起動されるプログラムの4者間の通信をセキュアなものにするため，適宜暗号鍵等で認証を行う．特にSPAproxyについては，既に述べた通り，ブラウザ等からのPOSTデータをセッション管理し，ありえないリクエストを排除している．

3.6. SPAgent使用ページと不使用ページとの整合

SPAgentは従来のWWWサーバと従来のブラウザで使えるものである．SPAgent機能を動作させる最初のステップはSPAservを実行することであるが，HTMLにはSPAservを(CGIプログラムとして)起動するためのURLが記述されており，これは一般のHTTPの仕組みと同じである．ブラウザの特性として，リンクはどのようにでも組み合わせて記述できるため，一つのページにSPAgentを使うものと使わないものが混在して構わない．

クライアント側にSPAgentがインストールされていない場合，SPAproxyへの通信(POST等)が行えないため，ユーザは初めてSPAgent機能を使う際にブラウザ画面の反応が無くなり異常に気付く．これを未然に防ぐため，SPAgent機能への入口のページにSPAproxyの動作の確認を行うJavaScriptを書く等のad hocな解決を行うこともできるが，ブラウザの仕様に依存した実装はSPAgentが求める方向性と異なるのでアプリ開発者には推奨していない．

SPAgentのクライアントはインストールしたがまだSPAservにアクセスしていない(鍵交換を終えていない)場合，(SPAproxyはその事実を知っているので)特別のプロトコル(入会セッション)で動作する．このため自然にSPAgent使用セッションに移行することができる．

3.7. 信頼モデル

サービスに対する信頼というのは本質的に人間系に依存するものであるから，インタネットのシステム構築においては，安全でかつ理解し易いコンセプトを決めることが最も重要なポイントとなる．暗号技術者はしばしばこの原則を忘れ技術的に「完備な」システム設計を目指す失敗を犯す．

ユーザに受け入れ難いコンセプトは形骸化する．例えば，無条件にcookieを受け付けるのが流行している今日の惨状がそれである(誰が適切に送信を禁止することが選択できようか)．

「安全で使い易い」システムの設計は一般に非常に困難であるが，SPAgentはあくまでひとつの選択として，以下に述べる信頼モデルを前提に設計された．

3.7.1. サーバ認証，サービス認証

正当なCAがサーバの運営会社(名)を保証しても，そのサーバで提供されるサービスの質は様々であり，サービス運営企業が責任の無い単なる通信業者の場合も珍しくない．

SPAgentシステムにおいてはサービス(SVP)に対する正当性という新しい概念を採り入れた．

a. SPACA の正当性は，SPAgent クライアントがSPACAの公開鍵を予め持つことと，ユーザがSPACA運営団体を(人間系で)信用することによって保証する．
b. サーバ(SPAserv)の正当性は，SPACAがURLや企業名等を保証し，サーバ証明書を発行することによって保証する．
c. SVPの区別はSVP名の文字列による．SPAservは，そのSVPがその名称であることを「技術的に」保証する．SVP名はある程度サービスの内容が推測できる名称にする．SVP運用者(サービス提供者)がその名称と異なるサービスを不正に行っていないことは，「人間系で」保証する．SVPはサービスの単位であり，複数のCGIプログラムで構成された大きな1アプリと捉えることができる(SVPをあまり細かく分けるとアクセス権の設定が繁雑になるので推奨しない)．

SPAgentシステムにおいては，この3段階の信頼関係の存在をユーザが信用することを前提としており，その前提をユーザが各サービス享受の度に意識しなくても不正が起こりにくいという点が大きな特徴であり，またその効果を実証する新しい試みである．

この3段階の信頼関係は，クレジット購入において，カード会社(および決済銀行)，加盟店，その店で売っている商品という3段階の信頼関係を暗黙のうちに(経験的に)消費者が知っていることによりリーズナブルな販売システムとなっているのと対比してみると面白い．

3.7.2. ユーザ認証，個人情報提供

HTTPにはBasic-Authentication[9]というusernameとpasswordによる最も基本的な認証メカニズムがある．また目的によってはCookie[9]を駆使した認証を行う場合もある．しかしいずれもブラウザが生の鍵を保存する機能を持つなど安全面で問題がある．

WWWサイトにおいてユーザ認証を行わなければならない理由はさまざまである．単にOne-To-Oneの広告を効率的に出す目的であったり，前回訪問時のデータを利用して使い勝手を向上させるなどの便宜的なものから，購入した商品の支払を確実にしたい，複数の人格を使っての偽装投稿や特典の重複享受を防ぐ，など，必要とされる認証の確実性も，保証したい情報の質も，さまざまである．

例えば住所・氏名の保証された個人証明書を持っていたとしても，それが各サービスで必要となるユーザ認証に適する場合はむしろ稀である．例えば音楽販売サイトでは，本人の住所が確実に判ることよりもむしろ正当なクレジットカード番号を入力させる方が運営も楽でありかつ実際的である．

本システムにおいては，これらさまざまな「ユーザ認証」のパタンを網羅することをそもそも目指していない．SPAuserIDというアクセス主体の発行を裏で管理し，それを認証することが目的である．各WWWサイトが必要とする認証情報(例えばその人の氏名や年齢)は，各WWWサイトの都合によってSPAuserIDとの対応を管理すれば良いとする．これは一人の人間が複数の電子メールアドレスを持てるようなものだと直観的に捉えることができる．

SPAgentを使うユーザは，己のSPAuserIDやその発行／取得について意識する必要は無く，また，訪れるサイトごとのパスワードを記憶するような繁雑な手続きを踏む必要も無い．SPAgentによるサービス提供者(開発者)もSPAuserIDの発行について意識する必要は無く，SPAuserIDによる管理では不足するようなサービス(例えばバンキングサービス)を提供する場合にのみ，その独自の認証番号とSPAuserIDとを対応付けたデータベースを持てば良い(ページごとに認証を意識した記述をする必要は無い)．

以上述べたように，SPAgentは，従来不明確に捉えられていた"電子的なアクセス主体の認証"と"実社会で個人が持つプライベート情報の確認"とを切り分けることにより，WWWサービス向きの扱い易いユーザ認証メカニズムを提供することができた．

3.7.3. "SVPを信頼する"ことの意味

本システムにおいては，SVPを単位としてそのサービスを認証し，またそのサービスに対してどこまでのアクセス権を与えるかを設定することを3.4節で説明した．しかし，SVPを単位としてユーザが承認を与えることは，常に明白な手続きというわけではない．

例えば，パソコンメーカの提供する"PCの障害を修復する"SVPに対し，ユーザがファイルの差し替えを許可したとする．その際，そのSVPの提供するWEBページの中に，OSメーカの提供する"ブラウザを自動バージョンアップする"第2のSVPへのリンクがあり，クリックすると共有ライブラリが差し替えられるかもしれない．このような場合，ユーザは第2のSVPに対してもファイルの差し替えを許可したと考えるべきか，以下の2つの考え方がある．

a. SVPを承認するとは，もともとそのSVPが悪さをする可能性があることを考慮した上で認めることであるから，承認したSVPが悪意で第2のSVPを利用させる心配をする価値は無いし，それを禁止すると，見た目上1つのサービスに対して複数の設定が発生して判りにくい．
b. SVPを承認するとは，そのSVPに悪意が無いことを信用しているに過ぎず，そのSVPが誤って悪意のSVPへリンクする場合もあるし，また第2のSVPが善意の場合であってもユーザが第1のSVPへの設定時に想定したのと違う期待しない結果になる可能性を持つ．その可能性まで含めてSVPを承認することはできない．

本システムにおいては後者(b.)の考えを採用した．これにより，"アプリケーションのパッケージの単位であるSVP" という概念が明確になり，またユーザが何に対するpermissionであるのか理解し易くなった，と考えている．

従来のブラウザでも，サイトが移動する時に警告が出るものがあるが，こういった粗い単位であるとユーザは無条件に"OK"を押す選択しか持たないのが実情である．本システムにおいては，ユーザの予め設定した基準に従って必要かつ十分な警告だけを表示させることが可能である．

3.8. プロセスの流れ

従来のブラウザでは，基本的にWWWサーバとブラウザ間を往復するプロセスだけがあり，それにhelper applicationとして予め指定したクライアントソフトを起動したり，サーバから受信したプログラムをPCローカルに実行する際の認証機能があるのみである．

本システムでは，SPAproxyがnodeとなって，さまざまなフローのプロセスを生成することができる．例えば，ブラウザを使ってPCローカルなプログラムを起動するには，SPAfilterを起動する命令の記述されたPOSTフォームをSPAproxyに対して投げればWWWサーバを通さずに直接実行することができる(そのようなPOSTフォームを生成するのはSVPの役目である)．また，SPAproxyがSPAservから受信したデータに基きSPAfilterを起動した後，ブラウザの表示とは関係なく，処理後のデータをSVPに返すこともできる．さらにはSPAproxyがブラウザを使ってユーザにメッセージを表示させつつ，SPAfilterにデータを渡して別の通信を発生させるなどの多重化も容易にSVPで記述できる．

つまり，ネットワーク・コンピュータシステムにおいて従来から使われているremote commandやshellの機能等の汎用的な仕組みを，インタネット・ブラウザ上でGUIとして疑似的に実装したものであると捉えることもできなくはない．この実装をするメリットは言うまでも無く一般のブラウザによるサービスとシームレスにリンクできる点にあり，また ブラウザが使える環境であればどこでも(Internet Protocolが通らなくても)遠隔サービスが受けられる等の付随するメリットもある．

4. SPA ファミリの実装

SPAgentシステムの核はSPAservとSPAproxyの対であり，その上に実装されるSVPやSPAfilterはアプリ開発者が任意に作成するものである．しかし，典型的な操作は数多いものではないため，基本的な機能はSPAファミリとして予め用意している．このうちSPAfilterはSPAproxyと密に連携して動作するため，事実上SPAgentの核の一部と言っても良い．

一般のアプリ開発者は，多くのサービスについては，予め用意されたSPAファミリのSPAfilterおよびSVPを組み合わせて使うだけでSVP記述することができ，SPAfilterを開発する必要は起こらない．

4.1. 基本操作フィルタ群

'registry' SPAfilter は Windows 系 OS における registryの読み書きを行う．'fileload' SPAfilterはローカルPC内にある任意のファイルの読み出しを行う．'filesave' SPAfilterはファイルの書き込みを行う．'machine' SPAfilterはローカルPCのハードウエア情報等を読み取る(ハード種別に合わせたソフトの配布や5.1節で述べるPCサポート等に利用できる)．

呼び出しは例えば "<FORM spahidden='SPpFn=1&SPpFl=**registory**+-action+w+-writetype+REG_BINARY+-writdata+00+HKEY_LOCAL_MACHINE+Enum\\ESDI\\TOSHIBA_MK4310'>" のようにHTML内に記述する．

4.2. 'license' SPAfilter

ソフトウエアの起動用暗号鍵の取得権限などのライセンス情報を配布するためのSPAfilterである．これは，配布されたライセンスを，取得権利のあるユーザ自身にも解読できない形式でライセンスファイルに格納する．また，ライセンスはSPAuserID単位ではなく，ライセンスをダウンロードしたPC単位に与えられる．ライセンスファイルを読み出して利用する解読機能は含まない．

4.3. 'exec' SPAfilter

他のクライアントプログラムを起動する特殊なSPAfilterである．この機能を多用すると従来のブラウザ並に安全性が下がる危険があるので，使用は音楽販売サイトが音楽プレーヤーを起動する等，目的が明白であるケースに限られるのが望ましい．また，プログラムを起動するサーバから配布されたプログラム以外は起動できないというオプション設定も持つ．

次項以下に述べるSPAファミリは，このexec SPAfilterから起動される専用クライアントと，それぞれの専用SVPが協調して動作する．

4.4.ネットワークインストーラSVP

SPAgent開発のきっかけは，ソフトウエアダウンロード販売システムを作成するにあたりインタネット独特の問題点を克服する必要があったことである[3]．そのシステムの特徴は，ブラウザの操作で自動インストール，課金，バージョンアップ，不正コピー防止，ソフトウエアの試用，ができることであり，インタネット回線の特性を考慮して差分送信，通信の圧縮，ダウンロード中の回線切断からの復旧(継続ダウンロード)等の機能を実装した．今回，このシステムを完全にSPAgent上に移行してセキュリティ機能部分はSPAgentを利用するようにしたと同時に，ローカルなインストーラとも連携するよう機能強化した．例えば，Windowsの「アプリケーションの登録と削除」メニューによって，ネットワークインストールした際のバージョン管理情報や(本来のインストーラの関知しない)registry登録情報等と不整合が起こらない工夫がある．

例えばCD-ROMからVersion 3のソフトをインストールしたユーザがブラウザでダウンロードページをアクセスし，Version 5のダウンロードボタンを押すと，自動的に足りないファイルを検出，ダウンロードして，必要に応じて再設定が行われる[2]．

4.5. アップロードSVP，ダウンロードSVP

アップロードSVP，ダウンロードSVPはいずれも前節のネットワークインストーラSVPの簡略機能である．ブラウザによる単純なアップロード，ダウンロードと異なる点は，それらの行為がセッション管理され，中断や再開が容易にできることにある．SPAgent上に実装されているので，勿論，データの圧縮や暗号化，認証なども行える．

5. 応用システム例

SPAgentは，従来のブラウザを利用しつつ高セキュリティを保証するための基盤モジュールであるが，同時に従来のブラウザでは実現困難であった高機能を提供するものでもある．SPAgentおよびSPAファミリを利用することにより，一般ユーザの視点でどんな「面白み」を提供することができるのか，それを実感する応用システムを試作した．

5.1.PCヘルプデスク

今日普及しているPC用のOSは(UNIXも含め)使い易さの意味で決して完成度の高いものではない．疑問やトラブルは多岐にわたり一般のマニュアルや索引では到底解決できない問題が多い．特に，あるアプリをインストールすると別のアプリが動作しないといった依存性のある障害対策は手に余る．これら事例の多くは，PCメーカーのサポートセンターに辛うじて蓄積されるものの，専任の電話オペレータでさえ最新の情報を基に常に適切な指示を出すことは困難な状況になっている．

これを解決するため，知識情報の共有技術[4]を用い，サポート情報(特にQ&A集)を予め意味処理した上でデータベース化し，日本語の自由文で検索すると該当事例が引用できるという形で検索するシステムが実用化されており，WWW上でもアクセス可能となっている．この種のデータベースは単純なキーワード検索ではマッチする事例が多過ぎて実用的でないため，高度な自然言語処理が良い効果を示す典型的なシステム事例である．

しかし，ことPCのヘルプデスクに関しては，好運にして自分の求める事例を引き当てたとしても，その内容が理解できなかったり，操作が繁雑で実行できなかったり，OSのレビジョン等の細かい前提条件が異なっていて復旧不能な操作を誤って行うなどの危険がある．我々は，SPAgentの機能がインタネットとブラウザを利用したPCヘルプデスクの実現に向くことに着眼し，プロトタイプシステムを作成し検証した[5]．

PCのヘルプデスクは「検索」，「診断」，「修復」の大きく三段階のステップで実現される．まず日本語の質問文の入力により事例検索を行う(ユーザが可能な場合には手で条件を付加しても良い)．検索の際にSPAgentの機能によりPCの状態を遠隔調査し一事例に断定することも原理的には可能であるが，一般には可能性のある複数の事例を提示し，ユーザが該当事例をブラウザで選択することになる．

障害修復の事例の場合，次に診断のステップに入る．サポートが自動化されブラウザ上での選択で修復できるようになると，ユーザは徐々にシステムに頼り過ぎる傾向が現れ，遂には提示された最高ポイントの事例を無条件にマウスでクリックするという行動に出る．従って，ユーザが誤った事例を選択していないかどうかを可能な限り予め診断する．

最後に実際に修復を行い，サポートを終了する．以上のステップは，PCのファイル(Windowsの場合はregistry)の読み出し，書き込み，ソフトのインストール，PCの再起動，といったSPAファミリの基本機能の組み合わせで実現できる．願わくばこの自動修復が万一誤っていた場合に取消する機能が欲しいと感じるが，現実の事例は修復したことにより間接的に別のファイル等が変更されるものがほとんどであり効果が出ないため実装していない．

SPAgentを使ったPCヘルプデスク・システムの場合，その事例データベース作成業務が繁雑になるという危惧があるが，元々HTMLで記述されている(SPAgentを使わない場合の)事例集に診断修復用の特殊リンク部分を書き足すことで実現できるため，コンテンツの維持管理が容易であり，日本語の意味処理のためのエンジンも従来使用のものがそのまま使える．また，障害対策が単に"特定registryを書き換える"といった単純な操作のものも多く，その種の事例に対してはユーザを誤り無く導く説明文を用意するよりも，SPAgentによる自動処理を記述した方がコンテンツ作成が容易にできる．システムのトータルでは従来と同等の手間でコンテンツ作成ができ，ユーザの利便性は著しく向上する[6]．ただし，修復そのものの記述よりも，選択した事例が本当に求めるものであるかを確実に診断する記述の方がはるかに煩雑であるため，常に「一発検索・自動修復」できるようなシステムを目指す場合はそれ相応のコンテンツ作成コストが掛かると考える．

5.2.ソフトウエア販売／レンタルシステム

筆者らはインタネットを利用したソフトウエア販売システムの開発と運用を行っている[3]が，それは4.4節にも述べた通りインタネット回線特有の性質や，商品説明の困難さを原因とする返品要求等のトラブルを防ぐ工夫を施したものである．このシステムの持つ特徴を継承しつつ，SPAgentの機能を利用したネットワークインストーラSVPを開発し，またそれを用いたソフトウエア販売／レンタルシステムSVP(アプリ)も開発した．

特に，ソフトウエアの実行(起動)ライセンスの管理については，ソフトウエアのダウンロードシステムとは分離し，4.2節に述べた'license' SPAfilterの機能を用いて実装することにより，SPAgentのユーザ管理機能をそのまま利用したソフトウエア配布の機能を実現した．

ソフトウエアは自己暗号化されて配布され，license SPAfilterによって配布されたライセンス・ファイルに記述されたライセンス条件を読み取った上で実行時にメモリ上で自己解凍するなどの処理を経て実行される．ライセンス・ファイルには，起動できる時刻や起動オプションなども記述でき，ライセンス条件が柔軟に指定できる．

なお，これらのライセンス付与は，ライセンス・チェック・ルーチンを組み込んでコンパイルするような手間をかけず，直接プログラム(.exeファイル)のバイナリを修正／暗号化して配布できるのが特筆すべき点である[7]．

本ソフト販売システムにおいては，例えば以下のような販売方法のバリエーションが用意できる．

a. 機能限定のソフトをダウンロードし，動作することを確認した後に限定解除ライセンスを購入する．
b. ソフト本体(無料)はCD-ROMからインストールし，ライセンスのみダウンロード購入する．
c. 1か月期間限定のライセンスを購入する．
d. CD-ROMからインストールした有料ソフトを，ブラウザの操作で自動バージョンアップする．すなわち，必要なファイルのみをネットワークインストーラSVPが自動配信する．
e. ダウンロード購入したソフトを(Windowsの)「アプリケーションの登録と削除」機能を使ってオフラインでアンインストールする．
f. CD-ROMからインストールしたソフトを，ブラウザでアクセスしたサポートページのボタンを押してオンラインでアンイストールする．
g. ソフトウエアのインストーラを全く作成せず，ネットワークインストーラSVPの機能のみで(Windows推奨の方法で)インストールを行う．

特に，既に市販されているアプリをネットワーク販売する場合，元々アプリの持っているインストーラをそのまま使いたいという要求が強く，我々のソフト販売システムの管理するソフトウエアバージョン管理／ネットワーク・インストーラ機能と，ローカルなインストーラ／アンインストーラとを整合させることが困難な課題であったが，ネットワーク・インストーラが，ローカル・インストーラの動作結果を管理する等の細々とした工夫により，統合することができた．

6. 評価と問題点

従来のブラウザの使い勝手を残しつつ，安全性の高いセキュリティ機能をアプリ開発者が手軽に利用できる基盤を完成させその使い勝手を実証した．

また，新しい信頼モデルに基く遠隔操作の機能を実現したことにより，PCヘルプデスクのような複雑なフローを持つサービスをCGIアプリとして記述し安全に提供することに成功した．

これら当初の目的に関しては設計通りの有益性が確認できたと考えている．ただしその運用を考えた時に，セキュリティ機能等をユーザに見せるという観点で下記いくつかの問題点が感じられた．

6.1.SPAgentの動作が見えない

ユーザがブラウザを操作している際，SPAgentがどう動作しているのか感じられない．もともと目指しているものがそういうシステムであるから，これは問題点というよりむしろ特長でもあるのだが，例えばダウンロードSVPにより確実なダウンロードを行っても，ユーザには一般のブラウザのダウンロード機能と見分けがつきにくく，SPAgentのありがたみを感じるチャンスが少い．

6.2.SSL等従来技術との整合に関して

SPAgent機能と，既存のHTTPのリンクを統合することは容易にできるが，SSLが混在した場合，暗号化しないリンクも含めて3種類のセキュリティ・スキームが理解し難い上，SSLを前提とした既存システム(Servlet等)がSPAgentを併用するメリットが出しにくい．特に，ブラウザはSPAgentの安全性に関知しないため，SSLのページからSPAgentのページに移行する際やPOSTの際に「安全ではない」旨の正反対の警告が出る場合があり，運用が難しい．

また，どんなにSPAgentが安全性を高めても，SPAgentを使わないサイトのサービス受信を制限しない限り危険性が残ることは避けられない．

7. おわりに

SPAgentは，従来に無い新しいコンセプトを多々含んだシステムであるが，今のところ性能をフルに生かしたキラー・アプリの数が少ない．実用目的に使うのみならず，研究用の題材，または「こんなことができるのか！」と驚かせる趣味のページ作りに役立てて頂けたら幸である．

謝辞

PCヘルプデスクの実証実験においてデータ提供や評価にご協力頂いた㈱東芝 デジタルメディア社およびPCダイヤルセンタの皆様，また，ソフトウエア・レンタルシステムの運用を通じて技術要求事項を提供していただいた㈱東芝 情報・社会システム社e-ネット事業部の皆様に感謝いたします．

参考文献

SPAgentの設計に関するもの

[1] 梶浦 正浩，他: 電子流通実装基盤SPAgent，情報処理学会 分散システム・インターネット運用技術(DSM)研究会他，1999/7.

SPAファミリの設計に関するもの

[2] 後藤 哲也，他: クライアントマシンの自動設定・修復を行うヘルプデスクシステム パッケージ化されたファイル群の差分転送方式，情報処理学会 全国大会，1999/9.

システム・コンセプトに関するもの

[3] 高橋 俊成: ソフトウエアの電子流通システム ソフトパーク，金融情報システム，平成9年7月号，1997/7.

SPAgent応用システムに関するもの

[4] 中山 康子，他: 知識情報共有システム(Advice/Help on Demand)の開発と実践，情報処理学会 HI研究会 インタラクション'97，pp. 103-110，1997.

[5] 半田 豊，梶浦 正浩，他: PCの障害の診断・修復機能をもつヘルプデスクシステム，情報処理学会 HI研究会他 インタラクション'99 インタラクティブセッション 論文集，1999/3.

[6] 半田 豊，他: クライアントマシンの自動設定・修復を行うヘルプデスクシステム エンドユーザサポートに応用した場合の効果の検証，情報処理学会 全国大会，1999/9.

[7] 秋山 浩一郎，他: 安全性と信頼性に配慮したソフトウェア期間貸し方式，情報処理学会 コンピュータ・セキュリティ・シンポジウム(CSS2000) 大会，2000/10.

既存のインタネット技術に関するもの

[8] http://www.w3.org/

[9] RFC1945: Hypertext Transfer Protocol -- HTTP/1.0，1996/5

[10] http://www.ssh.org/