第 41 回学友会総会にて使われた講堂システムの概要

作者：Hiroki @ 2J

1　事の発端

2009 年 5 月末から 6 月初め。第 40 回学友会総会を終えて、学友会執行委員会に所属する者が頭を抱えたこと。それは 10 月に行われる第 41 回学友会総会の会場である。毎回使っている B 棟は工事のため使えず (後に工事時期がずれて使えたことが判明)、規模を縮小して C 棟か、規模を大きくして講堂で行うかで揺れ動いた。それぞれにデメリットのある中、講堂の大きなデメリットである、発言をメモしておく黒板が使えないという問題に対し、こちらが使えるアイテムは巨大なスクリーンとプロジェクターであった。

この時、私には CGI とネットワークの知識が多少あったため、「ネットワーク作って WEB サーバー立てて、複数台のパソコンが書き込んだログを、1 台のプロジェクターに接続したパソコンで出力すればいいよね。」と考え、冗談半分で先輩に話してみた。

それをきっかけに、夏休み前にその案が可決・成立し、第 41 回学友会総会は講堂で行われることとなった。私はシステムの全て (ハードからソフトまで) を担当することになった。

その中でネットワークの基礎知識やパソコンの個体差等、非常に有益な情報が得られただけでなく、個人で作るには非常に面白いネットワークを作ることに成功したため、記事として残すことにした。

2　簡単な仕組み

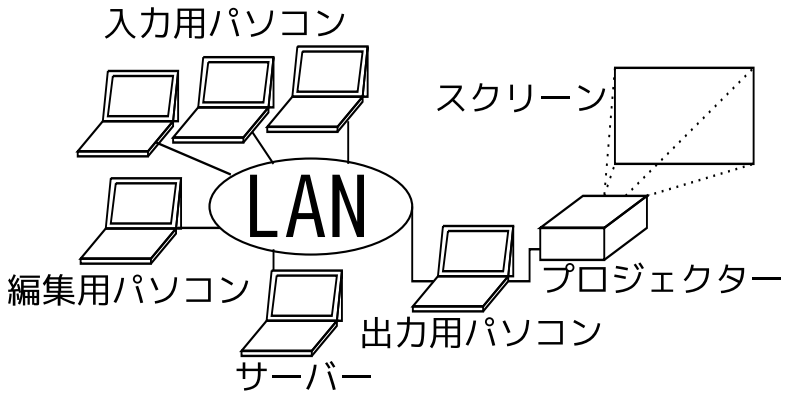
このシステムは簡単に一言で書き表せば、「掲示板の入力と出力を分けただけ」のものである。ただし掲示板の出力部は数秒間隔で更新する必要がある。

つまり準備物はインターネットと掲示板。それに入力用のパソコンと出力するためにプロジェクターとつなぐパソコンが必要である。

インターネットには、最小ネットワークである閉鎖的な LAN を使い、パソコンではブラウザを用いた。入力用ブラウザは何でもよいが、出力に関しては Microsoft 社の Internet Exprorer(以下 IE) の 7 や互換性を保った 8 を開発環境として用いた。理由は CSS の属性として、縦書きを唯一サポートしていたからである。HTML レンダリングエンジンは IE の 7 と 8 でいくつか致命的な違いがあったので、7 で使われている Trident を採用することとした。

最後に掲示板などを実行して表示するパソコンとして、サーバーを用意した。このサーバーはインターネットのホームページを持ったり、掲示板のプログラムを実行する事ができる。

ここまでで、ネットワークは次の図のような状態となる。



3　実際のネットワーク

実際にはLANには有線と無線が用意され、受付と議長団もネットワークでつながった。掲示板が作られ、それによる場内票数のやり取りも行われた。これは他の部隊との連絡にも使われ、トラブルがあった時には連携して対処することが可能となる。

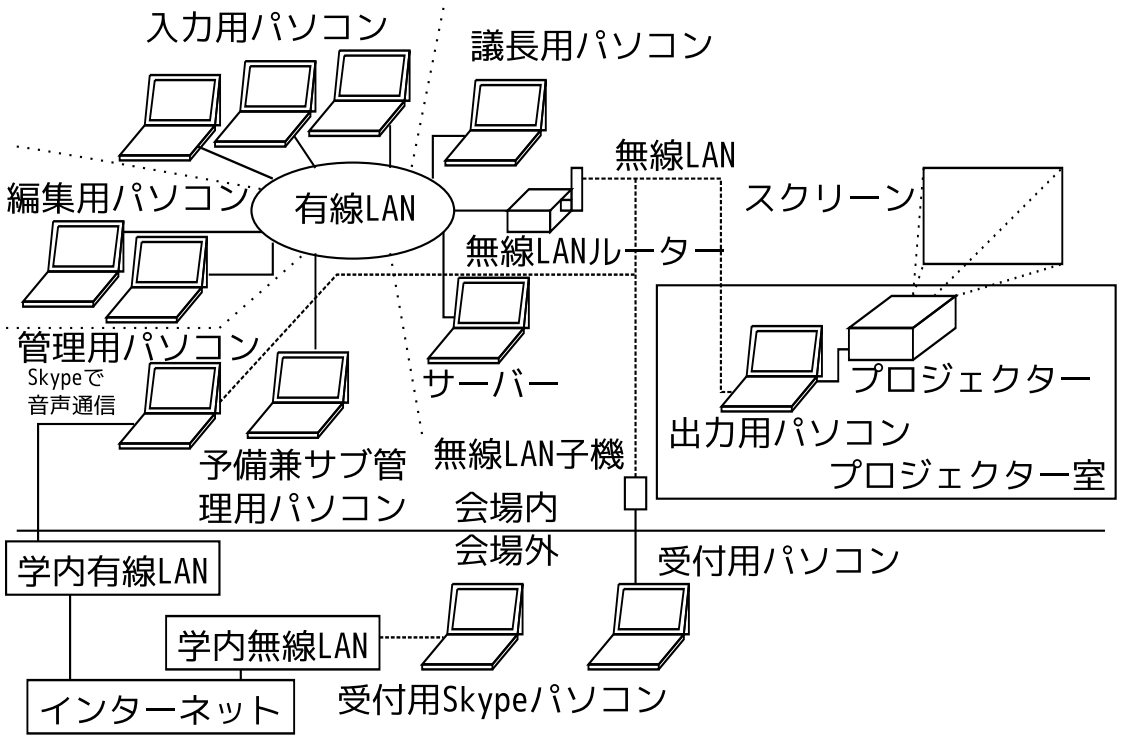
ネットワークはMicrosoft Windowsを使えば、物理的に繋いで設定を変更するだけで構築することは可能である。しかし、設定の変更はハードルを上げかねない上に、設定場所が異なる、OSが異なると今回のシステムの利点である、ブラウザによるそれなりに異なった環境でも簡単に扱えるというメリットを消してしまうので、1台のLinuxサーバーにネットワーク設定などを行わせることにした。

入力パソコンは3台のままだが、編集者用のパソコンでは、一度書き込みを見て、編集して、書き込みボタンを押すという作業が積み重なり、1台では処理しきれないので2台となった。

実際にはステージ上のプロジェクターにつなぐケーブルが使えないなどのトラブルが発生し、有線でプロジェクター出力用のパソコンとつなぐことができなかったために、出力用のパソコンをプロジェクター室に設置して無線LANでつないだ。

さらに総会前日に受付に会場内の音声を流すためのアンプが、電波が届きにくいことで使えないことが判明し、急きょSkypeで音声チャットを使うことになった。それに伴い、管理パソコンの一台は有線で外部に接続し、無線で内部と接続することとなった。

それらを総合し、以下の図のような状態で総会が行われた。



私が全てに目を通すための管理パソコンが、無線LANでないと内部に接続できないという状況になったが、1mも離れていないので十分トラブルに対処できると判断し、今回はSkypeでの音声チャットも担当させた。

予備のパソコンは入力用パソコンが落ちた時のために常に起動していたが、何もしないのももったいないので、管理パソコンのサブディスプレイ的な位置づけとなって配置された。当日にはパソコンが1台落ちて本当に交換することとなったため、予備を用意した価値は十分にあった。

4　CGI

今回の要である多人数入力・多人数編集によるログ出力 CGI だが、これを HTML の力だけで作ることには大きな壁があった。

まず HTML での自動更新だが、ブラウザの機能なのか、放置しすぎると更新しなくなることが判明した。

さらに、更新が数秒間隔で行われると、全ページを読み込み、再描写するだけで画面がちらつき、文字が見難くなってしまった。

これらの問題を解決すべく、HTML ファイルを更新せずにコンテンツを更新させる、Ajax と DOM と呼ばれる技術を使うことにした。この 2 つの技術は JavaScript で扱うことができる。

基本的な考え方は、まず HTML を読み込み、Ajax の非同期通信で CGI の実行結果を受け取る。その実行結果を、DOM を用いて HTML のある場所に代入し、コンテンツが更新される。そのため CGI では、Content-type が text/html となるのに対し、出力は HTML の断片情報のみとなる。(html タグなどは必要ない。)

これを数秒間隔で行えば、途切れることなく、画面がちらつくことなく更新することができる。

しかし、これにも問題が発生した。まず、例えば大きな画像を一面に出すと、再描写により画面がちらついてしまうということ。

さらに、IE のキャッシュ機能が強力で、同じアドレスの出力結果を保存してしまうので、一度も画面が更新されない事態が発生した。

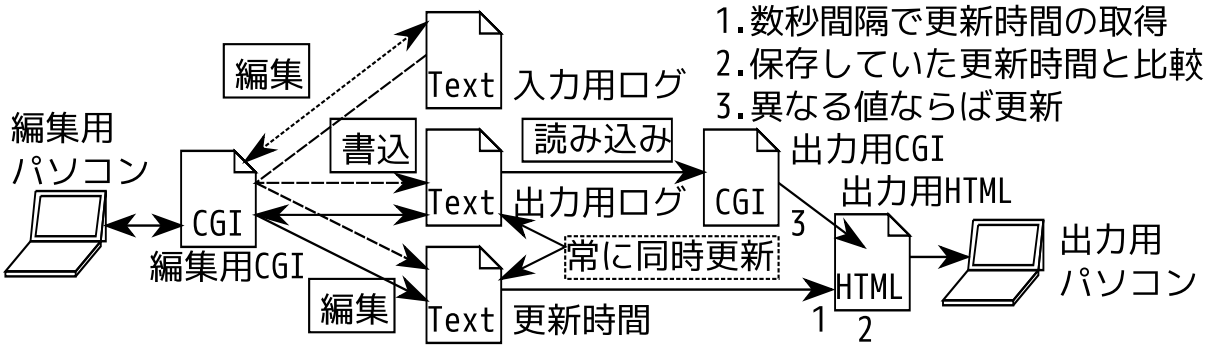
そこで、先人の知恵として、アドレスに?をつけると、それ以降が GET データとして送信されるが、それは IE のキャッシュに新規に追加されていくため、?以降のデータを毎回変更することで、確実にページを更新できるというものがある。

これを実現するために、更新回数を記録するカウンターを用意するという手もあるが、もし HTML を更新すればカウンターが初期化され、キャッシュにたまっているファイルを読み込んでしまう。これを回避すべく、現在の h 時 m 分 s 秒を取得し、h*10000+m*100+s という数値を?以降に結合することとした。

これで更新は確実となったが、画面のちらつきが抑えられない。そこで、今度は最少更新をさせる。

Ajax には同期通信・非同期通信というものがあり、同期通信は普通の関数を実行するのと同様、上から順番に実行され、処理が終わるまで次の処理は実行されない。しかし非同期通信は、通信中は処理を飛ばして次の処理を実行し、通信ステータスが変わるごとに指定した関数を実行する。例えばよく使われるのが、通信終了時に HTML の内容を更新する処理を実行させる、というものだ。

これらを使い、次のようにして最少画面更新を実現した。まず、編集者がログを更新させると、あらかじめ用意してある別ファイルに、現在時間 (今回は Perl の time() 関数で取得した値そのもの) を書き込む。次に、出力用の HTML の方では数秒間隔でそのファイルに同期通信でアクセスし、内容を今保持している値と比べる。もし値が異なるのであれば非同期通信を開始し、画面を更新するとともに、先ほどのデータを変数に保存し、次の同期通信で得られた結果と比較する。



最後に自動更新時の初期値であるが、0 を指定すると、無理矢理 F5 で更新したときに一度古い情報が出てきてしまう。そこで、初期値に乱数を入れた。これで更新時に一致することはほぼないだろう。

これにより最小更新が実装され、画面のちらつきも最小限に抑えることが可能になった。

5　ネットワーク構築について

ここまででシステムの全体ができたので、ネットワークの構築の詳しい話となる。

今回の要である LAN だが、基本的にハブを用いて物理的にパソコンをつなげただけでは作ることができない。まず IP アドレスを変更し、同じネットワーク内にいることが不可欠である。

ネットワークに関して、同じネットワークとは、ルーターのようなものを通さずに構築できる最小のネットワークのことで、具体的に IP アドレスを書けば `192.168.1.*` のようになる。

*以外の 3 つの数字が同じであれば、ルーターを通すことなくお互いに接続できるが、そうでない場合にはネットワークが異なるので、ルーターのようなもので異なるネットワーク同士をつなぐ必要がある。

さらに、IP アドレスをただ設定するだけでなく、通信相手の MAC アドレスも取得できなければ通信はできない。

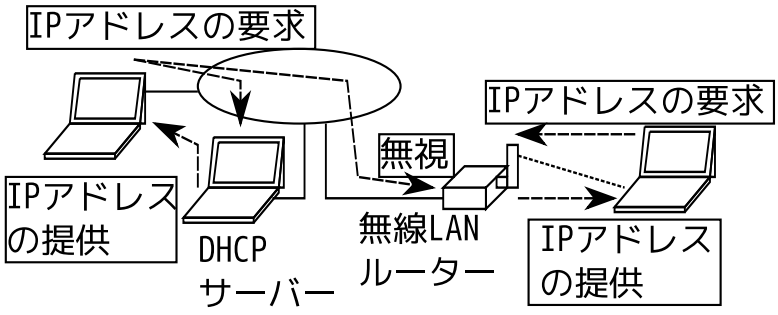
そこら辺の設定をこちらで全て設定するのは大変なので、サーバー機に自動的にネットワークを設定させることにした。

普段家庭でネットワークを利用する際、LAN ケーブルをただ挿せば接続できるが、あれは DHCP というサーバーがルーターに用意され、そのサーバーが IP アドレスを与えたり、その他接続に必要な設定を行ってくれるからである。その際に、IP アドレスを受け取るクライアント側の設定は、ネットワークに接続したときに DHCP を使うようにするだけである。この設定になっていれば、ネットワークに接続した時点で IP アドレスを要求し、DHCP サーバーがそれを受けて IP アドレスを供給することでネットワークとしてお互いがつながることができる。

よって、同じネットワーク内に複数の DHCP サーバーがあると、IP アドレスの割り当てなどで混乱が生じ、ネットワークが正常に機能しないこともある。

しかし、実は今回のネットワークには 2 つの DHCP サーバーが稼働している。有線 LAN の DHCP サーバーはサーバー機に処理させているが、無線 LAN に関しては無線 LAN ルーターに処理をさせている。

ここがポイントで、無線 LAN ルーターはほぼ確実に DHCP サーバーのあるネットワークで利用されるため、自分自身は無線で接続してきたパソコンに対して IP アドレスを割り当てるものの、有線に関しては関与しないというモードが存在する。今回は AP モード搭載の無線 LAN ルーターだったので、この機能を使った。注意事項は、この無線 LAN ルーターがルーターモードで稼働するとサーバーにつながらなくなってしまうということ。さらに IP アドレスの供給範囲も有線と無線でずらす必要がある。



これにより、有線での IP アドレスの要求にはサーバー機が、無線での IP アドレスの要求には無線 LAN ルーターが対処するということが可能となった。

最後にサーバー機に関してだが、CGI を実行する WEB サーバーと、IP アドレスを供給する DHCP サーバー以外に、DNS サーバーを稼働させている。このネットワークのままでは、例えば 192.168.1.10 がサーバーのアドレスなら、サーバーにアクセスするために、この IP アドレスを打ちこまなければならない。これでは使いづらいので、インターネットで使われている DNS と呼ばれるサーバーを使って IP アドレスに名前をつけることにした。

この DNS サーバーは自分の知っている IP アドレスと名前に対して 1 対 1 で設定を行う。もし要求があれば、最も上に存在する DNS サーバーに対し、例えば www.example.co.jp というアドレスなら、まず jp のアドレスを知っている DNS サーバーに問い合わせる。その後 jp を知っている DNS サーバーに co を知っている DNS サーバーのアドレスを聞き出し、そこに対して example のアドレスを知っている DNS サーバーのアドレスを聞き出す。ここで恐らく example の管轄下にあるサーバーにたどりつくのだが、そこで www のアドレスを知っている DNS サーバーのアドレスを聞き出す。この場合は設定ファイルに、www が付いているものは example.co.jp の名前が変わっただけです、という設定が書き込まれているので、そのサーバーの内部で解決することとなる。この最後の部分で、ようやく先ほど設定した 1 対 1 のデータが読み込まれ、IP アドレスが要求元に送られる。このように名前から IP アドレスを聞き出す「解決」という作業は、普通たらいまわしにされるのだが、今回は DNS サーバーが 1 台しかないので、要求があればすぐに解決して要求元に IP アドレスを返している。

ここまででようやくネットワークの基本部分を構築することができた。

外部ネットワークに関してだが、学内のネットワークは無線 LAN 講習会を受けた者だけが接続できるような仕組みになっている。無線 LAN の設定をしたのち、ブラウザから学内 LAN での認証ページにアクセスし、ID とパスワードを入力して、プロキシを通すことで外部に出ることができる。

Skype に関しては、一度ブラウザによってアクセスできる認証ページでログインしたのち、接続の設定でプロキシをブラウザのものと同じにすることにより、接続することができた。

最後に、サーバー機に用いた OS であるが、CentOS を使った。理由はサーバー用に作られ、安定しているということと、私自身初めて作ったサーバーの OS が CentOS であったからである。

サーバー機には富士通の Loox U/C30 という、超小型のノートパソコンを用いた。CPU は Atom の 1.33GHz 程度であったが、処理負担が大きくサーバーがパンクする、ということはなかった。

CentOS でのサーバー構築についてだが、コンソールだけでは日本語の入出力ができないので、実機で管理・編集するには GUI を入れる必要がある。今回はあまり負担をかけたくなかったので、X Window をコンソールから起動して使った。

WEB や DNS サーバーなどはファイアウォールで初めから禁止されていることが多いので、それらを切る必要もある。特に DNS サーバーは設定項目に名前がない場合があるので、53 番ポートを解放するような記述が必要となる。

6 障害

今回のシステム構築で一番大きな障害は、ハード面の問題であった。

例えば、初めは出力・管理パソコンを私のパソコンで行おうと思ったが、講堂ステージから伸びた VGA ケーブルからは出力することができなかった。さらに、プロジェクターに直挿ししても、適切なサイズである 1400*1050 の設定が存在せず、他の人のパソコンではかろうじてその設定にすることができた。

原因はパソコンの個体差で、VGA に関しては 0.7V という低い電圧で信号を送っているため、ステージからプロジェクターまでの長い距離の間に減衰し、届かなくなるようだ。他にも 5m の USB

延長ケーブルが使えなかったりと、私のパソコンは少し供給電圧が低いようだ。画面設定に関してもハード・ソフト両面の問題があるだろう。

そのような個体差があったために、何度もネットワークの構成を変えることとなった。そんな事態を想定し、例えば実行プログラムをブラウザに限定したのは他の OS、他のパソコンで利用できるという点で優秀であったし、いくつか別経由のバックアップを用意する必要もある。今回の場合は遊びで用意した無線 LAN がその別経由となったおかげで、プロジェクターへの出力が可能となった。

7　最後に

今回の開発でハード・ソフトの両方を同時に構築・設定した経験は非常に大きかったです。特に自分の得意分野から解決策を考えた後、それを実際に使うところまでのサポートを多くの人にしてもらいました。ちょっと聞いただけでは馬鹿らしいことではありますが、それを人に話し、議論し、実際に作ってみるということの大切さを学びました。今回このシステムにかかわった全ての人に感謝しています。本当にありがとうございました。

68DVD もしくは X680x0 同好会の公式サイト＞その他の作品＞デジタル版 会誌 の 2009 年度 (vol.15) に当時使用したスクリプト一式を ZIP 形式で圧縮して置いています。ぐちゃぐちゃなソースではありますが、今回使われた Perl-CGI や JavaScript、HTML 等のソースが入っています。