Memo: Things about visitors

Daniel Hahn

November 22, 2001

Abstract

These is a memo to structure some ideas, and it's a work in progress.
Comments are always welcome: dhahn@gmx.de

What it is about

The whole topic is quite broad, but loosely the following stuff needs to be done:

- find out *things* about the visitors to our site.
- filter out the *things* needed.
- present the the *things* to the user in a friendly way.

Sounds vague? Don't know what a *thing* is? Well, at the moment it could be almost anything: Information about the user himself[1], the user's homepage or other places on the web the user has connections with. The *things* we are looking for will depend on what the system should achieve. Some possibilities include:

- Just making the host aware of the visitors, or the number of visitors.
- Making the host aware of patterns in the visitor's behaviour.
- Actually pointing the host to a single visitor, getting as close as possible.
- Use information learned from the visitor to find interesting places on the web.
- Try provide an entry into a community that both the host and the visitor are part of.

All theses things are probably interesting to explore, but the scope of a Studienarbeit will probably only allow to implement one of them.

Find things

There are several sources of information that could be tapped to learn more about the visitor. For now we assume from all these sources we can learn something about *visits*, where a *visit* is the action of actually looking at a web page. (A line in a log file would correspond to a *visit*).

[1] No pun intended to the ladies, but in this case I'm writing the text as if the visitor is a male geek. She could also be a female geek, though.

Logfiles

Web server logfiles contain a record for each request that has been served. In the best case, there is information about the **hostname** or **ip address** the request came from, a **timestamp**, a **status** indication (if the transaction was successful or not), information about the **user agent** that made the request and also about the **referring page**, which sent the user to this server.

If a users logs into a site using built-in security mechanisms, the respective **username** could also be made available.

In addition, by looking at the whole logfile we could also answer questions like: *How often does a visitor come back?* or *What type of visitor is this?*

Transfer of additional information(through protocol extensions)

Although there is a lot of data available in the logfile, it is not meant to lead to interesting places. An obvious solution would be to extend the http protocol, to allow the visitor (or rather, his user agent) to provide additional information. Interesting information could include: The visitor's **most frequently visited site**, his **bookmark collection** or simply his **own homepage**.

The major setback (apart from privacy considerations) of this approach is that it depends on modifications both in the server and the user agent.

User provided information

Users often provide useful information themselves: By writing **guestbook** entries, by **signing up** for services or by completing **surveys**.

Secondary sources - the web and more

Once we learned some information about a visit, we can always try to look up more interesting facts. Of course a **WWW search** is the most obvious example, but we could also use things like **local databases**, **traceroute information** or **WHOIS lookups** to name but a few.

Another source of secondary information may be the **host himself** who can tell the system which results he liked best (or worst).

Select the things to play with

Once we have gathered information about a lot of visits we will find that most of it is probably not very interesting in a given context. For example, although a *visit* by a search robot is an interesting thing for some webmasters, the same *visit* is quite uninteresting if you are looking for visitors with the same interests than the host.[2]

Look at the things you got

The first step could be a classification of each *visit*. Taking into account the information learned both from this visit and previous visits we could start asking questions like: **What kind of visitor is this? What is he looking at?** or **Did he come from a university?**

[2] Unless the host is interested in indexing web pages all day, of course...

Throw away the unwanted things

Once we have learned a lot of *things* about our visitors, we are likely to throw out the things we don't need. For example, we could now eliminate visitors which are **searchbots**, come from a certain **service provider** and so on.

Rank the things remaining

Once we have a set of *things* that is potentially interesting, we want to rank them (the most interesting first...) Indications for the quality of a thing could include: The **type of the visitor**, the **number of visits** by that visitor, **similiarity** to our own page, and many more...

Dish out the good things

Once we have selected the most interesting *things* for our purpose (whatever that may be), we are ready to present them to the interested public.

Web page

The simplest approach to present the results. This could be a page that the host explicitly calls to learn about the visitors. This could also be a small frame/overlay in an existing page that feeds the results back to visitors and hosts alike.

Ambient display/Installation

A gentler method of making the host aware of the visitors could be an ambient display. This could be as simple as a light that grows brighter when more visitors come, or very sophisticated as an "artwork" that actually conveys a lot of information.

Community display

This could be a display that stays in the background (like an ambient display) but uses a lower level of abstraction and allows the user to actually surf into the pages if he chooses to do so.

Personal display

This would probably consist of something as a web pad, working much like the community display does but intended to be used rather like a newspaper.