What's in a name?

Becoming a master of domains

Beyond the friendly face of the World-Wide Web seethes a variety of low-level activity, rather like Richard Feynman's description of a vacuum. At any given time, everything appears placid and calm, but when you look closely, thousands of tiny transactions bind high-level language to machine-level events.

For instance, when you type in an http colon slash slash double-u double-u double-u, the name that follows gets turned into something that your machine can use to connect to another system over the Internet. How that process works has been a puzzle to most people, but forms one of the fundamental underpinnings of the Internet.

The structure that allows you to type a name—called in this context a "domain name"—and have it mapped to another machine is DNS (Domain

Naming System), one of the many examples of how the Internet distributes rather than centralizes information. In fact, DNS was one of the programs that helped build the Net, and it played a significiant role in tilting the Net toward decentralization when there was still a choice to be made.

Sitting on top of DNS administratively is InterNIC (Internet Network Information Center), an entity empowered and partly funded by the National Science Foundation (NSF). Its Registration Services division provides a top-level structure for the assignment of numbers and names for the United States portion of the global network, and helps coordinate international issues. InterNIC ensures that all numbers and names assigned on the Net are unique and properly integrated into the overall system with regular updates.

DNS makes the Net go; InterNIC does the registration that gets you onto the Net in the first place. Let's look at both pieces. The former we'll cover in this column; the latter in the September 15 issue of *adobe.mag*.

Making a name for yourself

DNS, at its heart, is an elegant and simple system. It's a mechanism that lets users avoid dealing with the nitty-gritty of how their applications—like Netscape Navigator—communicate with remote systems over the Net.

Domain names are only the human-readable part of DNS; they get matched up with Internet Protocol (IP) numbers, which are used to connect one machine's query with another's response. To cast it in everyday terms, think of the domain name as the person's name, street address, city, and state—the address on an envelope that represents how we envision the destination. Think of the IP number, in contrast, as the nine-digit ZIP code. To the average person, it doesn't mean much, but the Post Office uses it to get the mail within throwing distance of where it has to go. (The IP number, of course, gets Internet information to exactly where it needs to be.)

Domain names and qualifications

Whenever you use the Internet, even to send E-mail, you're encountering

domain names. When you type a URL (Uniform Resource Locator), like "http://www.adobemag.com," into a browser location dialog box, you're using what's called a "fully qualified domain name." That is, no additional information besides "www.adobemag.com" is necessary to fully identify the machine you're trying to reach.

Fully qualified domain names consist of elements separated by periods, and are read *right to left*, from the most general category down to the specific machine. In the United States, the rightmost part is the top-level domain (TLD), which defines the category of domain—".com" means commercial and ".org" means noncommercial organization, to take two common examples. Countries outside the United States use a suffix, such as ".fr" for France. The next piece to the left (in this case, "adobemag") is the specific name registered with the InterNIC.

Anything to the left of the registered domain name that corresponds to an actual destination machine, like a Web server, can also be called a host name. In this example, "www" is the host name for the Web server at

adobe.com, and Adobe's system administrators have mapped adobemag.com to www.adobe.com.

Some companies and organizations define "subdomains," which are additional hierarchies for different departments or sublevels. Sun Microsystems, for instance, has the "eng" subdomain under sun.com. A Web server there might have the name "www.eng.sun.com," in which the host name is "www," the subdomain is "eng" (for engineering), the registered domain is "sun," and the top-level hierarchy is com (commercial). This tells you a fair amount about what you're looking at.

Take a number

The flip side to domain names, as noted above, are IP numbers or addresses. IP numbers are what client programs, like browsers and server programs—such as Web servers (the software that runs on the remote machine to handle Web requests)—use to communicate with each other.

IP addresses consist of a set of four numbers separated internally by

periods: 192.0.0.1 for example. Each of the four numbers can range from 0 to 255. Addresses are assigned in ranges, where each range comprises a "network." For instance, my company has 205.199.66.0 assigned to it; the first three numbers uniquely define the network. The zero in the last position means that we can assign any value between 1 and 254 (0 and 255 being reserved) to machines on our network.

The first number in the "dotted quad," as it's often called, defines what classes the address belongs to, and the different classes indicate how many addresses have been assigned to the organization for which the network is defined. A Class A address has just the first number of the four defined; for instance, Stanford's network is called 36.0.0.0. The three zeroes at the end mean that Stanford can assign approximately 16 million machine addresses that start with 36.

Turning names into numbers

The fully qualified domain name is turned into, or "resolved," into an IP num-

ber through a cool process that is both distributed—that is, no central authority is required for it to work—and recursive—the same process is conducted in turn on each element of the domain name until the name is resolved.

You might imagine that a big machine somewhere (or several) holds a full list of all domain names out to the nth subdomain and the IP numbers corresponding to them. That's the way it was until about 1987, when the ARPANET (the nascent Internet) started to grow too rapidly to support that. In actuality, a few main machines at a very high level hold just enough information to point resolution down to the next level (the next one to the left in any "dot address"). Let's say you've got Netscape Navigator open and want to go to www.adobe.com. After you punch the URL in and hit return, the browser triggers the "resolver" software in your operating system that starts the querying process.

DNS servers, or nameservers, at the company or organization level are seeded with the names and addresses of the top-level DNS servers that have the detailed information about the various hierarchies, like ".com" and ".net."

The nameservers run by individual organizations and companies are functionally indistinguishable from the top-level nameservers; the difference is that the individual nameservers only store information about domains for which they are "responsible." The top-level nameservers, of which there are fewer than a dozen located around the U.S. and in a few other countries, contain information about where to find nameservers for every registered domain.

The browser calls the resolver in your operating system (OS) which queries one of these top-level machines, chosen at random, about adobe.com. The top-level name server says, "all information about hosts and subdomains for adobe.com is found at the machine called ADOBE-DNS.ADOBE.COM"—i.e., Adobe's name server; the top-level name server also passes along the actual IP number for the Adobe name server.

The OS's resolver, armed with that information, turns to ADOBE-DNS.ADOBE.COM and asks about "www.adobe.com"; the Adobe name server says, "Ah ha! I know that machine, and its IP number is 192.150.2.100." The

resolver can then return the information, and the browser's request to go to the Adobe home page is initiated.

A long road for a short return?

It may seem like a long row to hoe in order to get a short distance, but the DNS-to-IP number scheme allows several benefits, key among them that each organization is able to update its naming and numbering without constant interaction with a central authority. Once network numbers (the IP number ranges) are assigned and domain names are registered, all control of domain name and IP-number assignment is local.

Another benefit is that, since there's no centralized information, you don't get a bottleneck at the top as the Net grows. Since the load at the top is only on getting information to point to the next level, the load is distributed throughout the entire Internet to hundreds of thousands of name servers. Also, information is stored, or cached, for awhile at the many local name servers, so that, after you go to www.adobe.com, your local name server

retains that information for as long as Adobe has specified—usually, about a week—before going out and doing the query again.

Next issue, you'll learn how to navigate the murky waters of actually registering a domain name—and how the hierarchies now in place may change radically in the coming months.