Routes, Addresses, Handles, and Names in Networked Communications
(DRAFT 0.2)

Michael J. O'Donnell

17 August, 2002

Abstract

Networked communications inherently depend on the ability of the *sender* of a message to indicate through some token how the message should be delivered to a particular *recipient*. The tokens that refer messages to recipients are variously known as *routes*, *addresses*, *handles*, and *names*, roughly according to their relative nearness to network topology vs. human meaning. All four sorts of token refer in some way to a recipient. Substantial confusion has arisen regarding which sorts of tokens are needed in various network operations and applications.

This article provides a conceptual overview of the function of routes, addresses, handles, and names in a communications network, with a rough description of the relevance to the global Internet. I find that all four sorts of tokens are important to maximizing the utility of the network. But the value in four levels of reference cannot be understood in terms of the network technicalities of efficient implementation of individual acts of communication. Rather, the value of such multilevel reference derives from administrative considerations based on the multiplicity of parties who should enjoy authority over, and bear responsibility for, different aspects of network operations and applications.

0.1 I think this first version is already fairly readable, but the target audience is not clear. At some point, different sections of this article may make more sense organized into different articles.

0.2 Added Figure 1, a bibliographical item, and fixed the STUB describing 0.1.

1 Criteria Driving Network Token Design

Realtors are reputed to claim that the three most important qualities of property are

location, location, and location

Software and protocol designers might use a similarly insightful, but unfortunately less catchy, phrase to describe the three most important considerations affecting their designs:

> binding time, binding time, and binding time

"Binding" refers particularly to the act of attaching a value to a variable, and more generally to establishing the value of any unit of information in a system. When only one computer or person or relatively coherent organized group makes bindings for some process, then the time at which it must make each binding is crucial to its ability to interact effectively with the rest of the process. For example, the utility of a system for discovering air fares may depend critically on the time order in which starting point, destination, flying time, class, price limit, etc. are bound by the user. When several separate parties interact with the same system, I interpret the word "time" in my slogan more generally. It includes all the conditions under which a binding is made—in particular, who has authority to make the binding, as well as when that authority may be exercised.

I find that the right criteria for designing the use of tokens to refer to recipients in a computer network have to do with matching the various authorities, responsibilities, and incentives associated with different parties to network operations against the operations supported on the various sorts of reference tokens.

2 Parties to Network Operations

I derive the need for the four different sorts of tokens that may be used in a network—routes, addresses, handles, and names—from the need to accommodate a number of different sorts of parties to network operations, each one assigned different authority and responsibility, and each one feeling different incentives. I derive my list of parties intuitively from my observations of the global Internet. A different list of parties might demand a different design.

In this article, an *agent* may be any entity in which we care to invest authority. That includes human beings, offices performed by human beings, corporations, departments within corporations, groups of human beings acting somewhat co-operatively, computer programs, and lots of other possibilities. A document may be thought of as a relatively passive agent, or the curator of a document may be treated as a different agent when acting as curator than when performing other roles. Essentially, something is an *agent* whenever it is convenient to think of it as an *agent* for the sake of our discussion.

Parties in network operations play different roles, including:

Routers: Agents that read information in a message somehow indicating the desired recipient, and direct the message in order to reach that recipient. Today, the word "router" often refers to a particular sort of network hardware designed especially to serve as a router. But all "hosts" in Internet terminology are also routers. For the purpose of my conceptual overview, a router may also be a particular piece of software running on a computer host, or any other identifiable agent that participates in routing. A network system normally includes a large number of routers.

Network administration: The unique collective agent that determines the rules by which a network system operates. Network administration is normally a loose organization comprising a variety of computers, individuals, and participating organizations with different detailed incentives. But to the extent that they all co-operate in the interest of effective network operation, I regard them as constituting a single collective agent.

Members: All agents that send and receive messages across the network.

Humankind: All human beings acting as a very loose sort of co-operating group. Actually, I don't like the speciesist restriction, but "humankind" will do for the present discussion.

A particular agent may participate in the network in more than one of the roles above, or as a member of a larger agent, but I derive my observations about reference tokens from the separate actions of agents in their different roles. Notice again that there are lots of routers and lots of members, each constituting an individual agent. There is only one network administration, and only one humankind, although each one is a large collective agent.

3 Routes, Addresses, Handles, and Names

3.1 Tokens

Routes, addresses, handles, and names, are all sorts of *tokens* that refer to destinations for messages in a network. But what is a *token*? In the design of software and protocols, it is often impossible, difficult, or at least undesirable to define something by *what it is.* Rather, we define things by *what we can do* with them—the operations on them and the relations that may hold between them. The abstraction of such definitions makes understanding more difficult, but once we understand such definitions, our minds may be opened to techniques that we missed before. In this article, I will give an abstract definition of *token*, along with a concrete discussion of typical tokens.

Token: An object that we can copy, transmit, and test for equivalence against other copies of tokens.

Concretely, in a network system a token is typically represented by a sequence of binary bits. We know how to copy and transmit sequences of bits. Two copies of sequences of bits can be considered equivalent if the same bits occur in the same order. Sometimes we interpret sequences of bits, for example as strings of characters. In such cases we may liberalize the notion of equivalence, for example by considering different lengths of white space as equivalent (making "`Hi there`" equivalent to "`Hi    there`"), or considering different capitalizations to be equivalent (making "`Hi there`" equivalent to "`hI THerE`"). For those who prefer numbers to sequences of bits, we may equally well think of tokens as integer numbers, as long as we don't take the numerical operations (addition, subtraction, multiplication, etc.) and inequality relations (less than, less than or equal, divisor, etc.) as part of their quality as tokens.

The concrete understanding of tokens as sequences of bits, or as integers, is sufficient for most discussions. But occasionally it may lead us to overlook really good ideas. In some cases it may be useful to let the act of copying a token from one context to another change the sequence of bits that represents it, and always send tokens into a common context to text equivalence. For example, if we let sequences of bits represent character strings, we may want to translate between different character representations, such as ASCII and Unicode.

Anyway, the "concrete" notion of sequence of bits is more abstract than we often think. When we copy a sequence of bits from one computer system to another, we often change the electrical form in which 0s and 1s are presented, the order in which bits are collected into bytes, words, and sequences of words, and perhaps other qualities. When we copy an integer from one computer system to another, we may change the way that it is presented as a sequence of bits. Some day, we might have to translate tokens represented with binary bits into a computer system that uses trinary notation (digits, or maybe they should be "trits," 0, 1, and 2). Whenever we are able to, we are better off keeping in mind that computation inherently deals with abstract representations, and real concrete meanings only arise when we interpret the inputs and outputs of computation in the external world.

3.2 The Four Types of Tokens

If you're still confused about the precise meaning of *token*, pretend that you're not, and just think of tokens as integer numbers, used like telephone numbers so that you fortunately don't have to remember how to add and subtract them, much less do long division.

Unfortunately, the definitions of routes, addresses, handles, and names introduce another source of confusion. I will define these four types of tokens in terms of the operations that we can perform on them. But you might notice that in most cases we can translate between the four types of tokens, so in principle whatever we can do to one we can do to the others. The real differences lie in which operations must be very efficient, and in who has the authority to determine the translations. When we translate from names to handles to addresses to routes, networking jargon says that we *resolve* a name to a handle, etc.

Route: A token that we can attach to a message directing routers how to deliver that message.

Address: A token associated with a particular target location in a network. At any location in the network (not just the target location), an address should resolve to a route leading to the target location.

Handle: A token associated with an agent participating in a network. A handle should resolve to an address at which we may communicate with that agent.

Name: A token carrying some humanly understandable meaning. A name should resolve to the handle corresponding to its humanly understandable meaning.

I use the word "should" advisedly in these definitions. A network system is intended to support the resolution of names to handles to addresses to routes in a fashion that satisfies all of the "should"s. But the requirements for resolution of names is inherently subjective, so the resolution cannot be perfectly reliable. As we go down the list from names toward routes, the objective quality and reliability of resolution improves, but it never becomes perfect. The success of the global Internet depends critically on our willingness to work with protocols that *should* produce a particular result, but that fail occasionally. We have found that it is better in many cases to guard against the consequences of occasional failures than to try to prevent them.

In this article, I am trying to show the value of including all four levels of reference, and all three resolution steps, in the design of a network. But many network designs reduce the number of levels to three, two, or even one (if there's only one, it must be routes) by omitting levels and/or conflating adjacent levels. I am not aware of any network design in serious use today that includes all four levels and keeps them clearly separate. When a level is missing, the tokens at a higher level (nearer to names) just skip over that level to the next lower one (nearer to routes) that is included in the design.

By composing the different resolution steps, all network reference tokens resolve down to routes. In many cases, we can also run resolution backwards (e.g., the current *whois* service maps IP addresses back to domain names that resolve to them). The relative ease with which each type of token may be mapped to each other type makes it hard to keep track of the differences. Whenever the support for a particular type of token is missing, we tend to use another type of token to approximate it. For example, the English phrase "Editor of the *Journal of Irreproducible Results*" is a name that works pretty well outside of the network as a handle, referring continuously to the abstract agent who edits *JIR* no matter how that role passes from person to person and how the people playing the role move from one address to another.

In the definitions above, I differentiated addresses, handles, and names according to the different objects that they should refer to consistently, even while the routes change. An address should always refer to the same location, a handle to the same agent, and a name to the same humanly understandable meaning. The meaning of a name is clearly subjective, but in fact the notions of location and agent are also fuzzy. Instead of refining the definitions of location and agent, which I am pretty sure can never be made satisfactorily objective, I will distinguish addresses, handles, and names more carefully according to where we invest the authority for their resolutions. In effect, this means that a *location* is whatever the party with authority over address resolution wants it to be, an *agent* is whatever the party with authority over handle resolution wants it to be, and *meaning* is whatever the party with authority over name resolution wants it to be.

Address to route: Network administration has authority over all resolutions of addresses to routes. In practice, the network administration as a whole usually delegates some of this authority to smaller agents participating in the network administration.

Handle to address: Each member of the network may become owner of one or more handles. The owner of a handle has authority over all resolutions of that handle to an address. Network administration may have authority to assign handles to owners.

Name to handle: Humankind has collective authority over the resolution of names to handles.

The intention in the assignment of authority above is to make the authority over a resource, the responsibility for acceptable use of that resource, and the incentive to derive value from the resource, coincide as much as possible. If I have identified the types of parties well, and matched the resources to them well, then these authorities, responsibilities, and incentives will coincide well.

Did you notice the circularity of definition? An agent is whatever the owner of a handle wants it to be. The owner of a handle is a member of the network. Network members are agents that send and receive messages through the network. Strangely, this circularity is not a problem in practice. To put it whimsically, network agents invent themselves. More sensibly, the overall behavior of the network makes it valuable to use a handle in a certain way, and we think of the responses to messages to that handle as the behavior of an agent. New schemes for using the network will open our minds to new sorts of agents. Sometimes we may recognize an agent in the world outside of the network, and find that there is no effective way to associate a network handle with such an agent. This may stimulate us to improve the flexibility of our network tools to allow the association. Although it's ontologically circular, this sort of interplay generates useful progress in network operations and applications.

As one more exercise in thinking about the four types of reference and the three types of resolution, consider the ways in which each type of resolution can vary. All three vary over time, to deal with mobility on the network, changes in jargon, etc. But at a given time, they vary differently.

Address to route: Resolution varies according to starting location in the network, so that the ending location is constant.

Handle to address: Resolution does not vary except over time.

Name to handle: Resolution varies according to context, local differences in language and culture, and anything that affects the way people think.

Figure 1: System of parties, reference tokens, and referents.

Roughly, address→route resolution varies within the network, handle→address resolution doesn't vary at all, and name→handle resolution varies outside the network. OK, some of you know about tricks like multihosting and caching, particularly as Akamai uses them to improve Internet efficiency. These methods let the resolution of handles to IP addresses vary according to location in the network, and perhaps in other ways. What's really going on here is that certain agents are spread around more than one IP address. In CS jargon, these are *distributed* agents. For the discussion in this article, it is best to think of distributed agents as inhabiting distributed locations associated with distributed addresses. IP addresses are not the only sorts of addresses, and routes do not always lead in only one direction. The basic principles discussed in this article appear not to depend on whether agents are concentrated locally or distributed.

There's an interesting symmetry in the three resolution steps. The outermost two—name→handle and address →route—are each controlled by a single collective agent: network administration holding collective authority over the network, and humankind holding collective authority over civilization outside of the network. Individuals control the middle step—handle→address. I don't know exactly what to make of this symmetry, but I think there's something right about it.

Also notice that the outermost resolution steps vary, one according to network location and the other according to cultural context. The middle, individually controlled resolution does not vary, except over time. In effect, the location/context variation is all taken care of by the outermost two resolution steps, freeing the middle one of the need for such variation. From another point of view, handles are explicitly engineered to encode all required variation into their tokens.

4 The System of Parties and Reference Tokens

The considerations above suggest a system of parties and reference tokens with the structure shown in Figure 1. Figure 1 works best in color, but black and whit renditions should be comprehensible.

- The items in blue rounded rectangular boxes on the top row represent the three types of parties with authority over resolution—a single network administration, any number of individual network members, and a single humankind. These parties are abstract agents who participate in the network, but the network design doesn't define them precisely and formally.

- The items in black square boxes in the middle row represent the four types of reference tokens. These types of tokens must be given explicitly and formally in the network design.

- The items in gold ovals in the bottom row represent the types of mental concepts that the four types of reference tokens are intended to capture. The act of message delivery in the network is rather precisely defined, but moving from left to right, the concepts become less objective and more ambiguous, as suggested by the fuzzier sorts of boundaries.
- The red lines from the top to the middle row represent the authority of the three types of parties to control the three types of resolution. The multiple connections from individual members to arrows from handles to addresses indicate that each handle owner has separate authority over her handle(s). Although each sort of authority is exercised by controlling the contents of certain tables stored by network hosts and routers, the exercise of that authority is problematic when wielded collectively by all of humankind. The squiggly red lines from humankind to the name→handle resolution suggest the complexity of humankind's collective exercise of authority.
- The black right-left arrows in the middle row represent the three types of resolution. These methods of resolution are implemented through some sort of tables in various network hosts and routers.
- The gold left-right arrows in the bottom row represent the conceptual connections that allow each of these concepts to determine one to its right. Each delivery leads to a particular location. Each location contains a particular agent. Each agent is responsible for data or services with a particular human meaning.
- The green up-down arrows between the middle and bottom rows represent the intended associations of deliveries with routes, locations with addresses, agents with handles, and meanings with names. The association of routes with deliveries is determined precisely and formally by the network operations. From left to right, the associations become less objective and more ambiguous, and suggested by the fuzzier sorts of arrows.

Notice that there is generally more than one connection between each type of reference token in the middle row and the corresponding conceptual type in the bottom row. There is a direct green arrow, and possibly one or more connection following black arrows to the left, then a green arrow down, then gold arrows to the right. The design and implementation of a system of network reference tokens is intended to make all of those paths connect the same individual items. In particular, when we resolve a particular handle into an address and then into a route, then use that route to deliver a message to a location, the agent receiving the message at that location is intended to be the one associated conceptually with the given handle. As network architects and engineers, we can only control the mechanisms for the black arrows. A design and implementation are successful if they make the black resolution arrows work in such a way that it is possible (with high reliability, but not absolute perfection) to think up sensible conceptual interpretations of the green and gold arrows that make these different connections equivalent.

There are a lot of details involved in making such a system work efficiently. For example, although each party should keep a table of the resolutions directly under its authority, and that table should be the final resort to resolve tokens correctly, all sorts of routers and other agents should keep local tables, called *caches*, of the resolutions that they are using regularly, to save the traffic and the delay associated with sending to the authoritative source for each resolution. Furthermore, local caches don't necessarily correspond directly to address→route, handle→address, and name→handle resolution. If a particular agent is concerned with the correspondence between names and addresses, it should cache a table of the direct resolution of names to addresses, derived by composing the name→handle and handle→address resolutions. With this sort of transitive caching, the cost of multilevel resolution is not much more than one-level resolution.

4.1 Reference Tokens and Resolutions in the Current Internet

Routes. In the current Internet, routes do not need to be written down in one place. address→route resolution interleaves with the execution of a route, so that the route is implicit in the path by which a message is forwarded. The relationship between IP addresses and routes is a bit more complicated than this article suggests, since several different routes to the same address may be used at the same time, and messages may be broken up en route and reassembled at the end.

Addresses. The IP routing protocol is described in a form that uses IP numbers as addresses. IP numbers are just 32-bit numbers (in the range 0 to 4,294,967,295). But IP numbers are not the only sorts of addresses used in the Internet. The UDP protocol uses the combination of an IP number and a port number as an extended address. IP numbers only allow a message to be addressed to an entire computer, called a *host* in Internet jargon (OK, some of you know that they actually address a particular network connection on a host, but that doesn't make much difference). UDP addresses allow a message to be addressed to a particular *application* running on a particular host, such as a particular sort of server, or a mail recipient. Other protocols have other notions of address—the HTTP protocol supporting the World Wide Web uses URLs as addresses. A URL essentially addresses a particular file on a particular host.

Although I'm not sure that they are recognized explicitly as addresses, networking efficiency sometimes requires *distributed addresses* for servers requiring the resources of several hosts. For example multihosted Web servers share the messages to a particular address among several different hosts. As far as I know, distributed addresses on the current Internet are simulated implicitly through some tricks with routing tables, but conceptually they are perfectly good addresses. Mobility and intermittent connection call for time-dependent addresses. For the future, we should open our minds to the possibility that any sort of instructions for contacting a particular agent may be thought of as an address.

5 A Pseudohistory of Network Reference

One way to understand the value of the four layers of reference to the effective use of a network is to consider a slightly fantastic, but realistic, history of network development as it might have happened. Real history happened sort of this way, but not exactly so. Put another way, "it could'a happened."

In the beginning, there were routes. A network cannot deliver messages without routes. The UUCP system directed all messages by routes of the form `host1!host2!host3`... `!hostn`, describing the entire sequence of "hosts" (acting as routers in my terminology) on the route. Each host/router kept its own table of hosts/routers with which it communicated directly. System administration required little more than agreeing on the general format for describing routes—all operational routing details could be handled independently by hosts/routers.

Routes allowed great support for distributed routing, but they were not portable. If Sally at the host `gargoyle` discovered the route to a great online candy store run by Grampa, and wished to share it with her friend Paul at `foghorn`, she could not merely send the route token to Paul—someone had to translate the route. The only general and reliable way for Sally and Paul to translate the route was to append the route between `foghorn` and `gargoyle`, which they must have known in order to communicate. This led to nasty long routes with inefficient forwarding. Even if Sally were selfish, and kept the candy supply to herself, she had to translate the route when she moved from `gargoyle` to `juniper`. Worse, selfish Sally could rest immobile at `gargoyle` and still find that a change in network topology invalidated her treasured route to Grampa's candy.

Routes begat addresses. In small local networks, with all participants connected rigidly and directly, routes are pretty much indistinguishable from addresses. The design of IP for ARPANet and the Internet made useful sense of globally meaningful addresses in a dynamically changing network with many different sorts of hops between particular communicating hosts. Addresses in IP were just 32-bit numbers (numbers from 0 to 4,294,967,295), but they were usually written in the form `n1.n2.n3.n4`, where `n1`, `n2`, `n3`, `n4` are 8-bit numbers (numbers from 0 to 255) written in base 10. For example, `216.227.0.100` is a typical presentation of a numerical IP address, corresponding to the 32-bit binary number 11011000111000100000000001100110, and to the number 3,638,755,428 in base 10.

Network administration had authority to assign these numbers, but it could delegate the authority to assign numbers within a subrange. Each host/router kept track of its own address, the addresses of hosts/routers with which it communicated directly, and the direction in which to forward a message with each possible address. Since 2^{32} =4,294,967,296 was somewhat too large to allow each host/router to store a table with a separate entry for each possible address, routing tables held entries forwarding all addresses in some numerical range in the same direction, and providing a default direction for addresses not in the table.

If it worked, IP routing would clearly provide global addresses, whose meanings would not change according to the location from which they are used. This allowed Sally to keep track of Grampa's candy store, and share it with Paul at will. It takes some thought to make sure that the routing could work. In fact, with a very feasible amount of communication to update routing tables, Internet network administration was very successful at supporting correct IP routing. Essentially, network administration had to make sure that every host/router had tables that were sufficient to send each message one step closer to the recipient address, or one step closer to a router that knew where to send it.

With IP addresses to pass around, Sally, Paul, and Grampa were all quite happy, until the candy store's address changed. The address changed once because Grampa moved his server from space rented on a shared computer to his own computer, once because he moved the store to another state with lower business taxes, and several other times because network topology changed. Even though IP numerical addresses were not tied rigidly to routes, they had to be assigned so that the routing tables could keep information on a limited number of numerical ranges, forwarding all addresses within such a range in the same direction. Both the candy store's own mobility, and the requirement to maintain efficient routing through a change in network topology, prevented Grampa's IP address from sticking reliably to the candy store.

IP routing never required anyone to resolve an address into an explicitly written token presenting the route. Routes were implicit in the joint distributed actions of all of the hosts/routers. In effect, the resolution of an address to a route was interleaved with the actual performance of the route. But routes were still there. Those who really wanted to write them down could generally get them from the `traceroute` program.

Addresses begat handlenames. The designers of the Internet realized very early that effective use of the network required the ability to refer permanently and reliably to an agent whose address might change. So, they invented *domain names* of the form `bottomdomain.subdomain....subdomain1.topdomain` to serve as permanent handles. Network administration had authority to assign domain names, but it could delegate that authority hierarchically even more flexibly than the authority over IP addresses. Network administration maintained tables translating domains to addresses. But only the translation of top-level domains (`edu`, `com`, etc.), which appear at the right-hand end of complete domain names, needed to be available globally. Each top-level domain name could resolve to the address of a server keeping tables for that domain only, and so on down the hierarchy. Furthermore, each individual host could maintain its own local cache of recently or frequently used domain name translations, avoiding repeated appeal to the authoritative name servers. This worked *really* well in practice.

With domain names, Grampa could acquire the domain name `candy.com`, and subdivide the business into `chocolate.candy.com`, `halvah.candy.com`, etc. at will. Sally could keep track of `candy.com`, and perhaps her favorite subdomains, use these domain names from any host on the network, communicate them to Paul at will. Furthermore, whenever his own mobility, or a change in network topology, caused the address of the candy store to change, Grampa could merely update the entry for `candy.com` in the appropriate authoritative table, and let it spread around to all of the local caches. This also worked *really* well.

In the story so far, domain names have served as handles, providing permanent reference to an agent through changes of address. But Grampa, and all of his actual and potential customers, got a big bonus as well. The domain name `candy.com` served as a humanly-meaningful name. Sally and Paul found `candy.com` fairly easy to remember, type in their emails to one another, spell out over the telephone, and even to guess at before they knew of the candy store or whenever they lost their record of its name. Even had it been permanent, a numerical IP address would not have been so convenient.

The fall: names slew handles. Unfortunately, the very knowledge of the humanly-meaningful semantics associated with `candy.com`, giving it value as a name, became incompatible with its function as a handle. A number of larger and more powerful candy companies, as well as the multinational corporation *C and Y*, all claimed rights to `candy.com`, and it was taken away from Grampa. The bonus value of `candy.com` as a name led to administrative action violating its use as a handle.

It is tempting to blame those nasty big companies for stomping on Grampa, but in fact, humanly meaningful names are inherently subject to forces beyond the authority of an individual handle owner. Human meaning, by definition, is determined by humankind. Due to the distributed fashion in which humankind wields its collective authority, an individual may determine the human meaning of a name among a circle of friends who accept his influence. But it is fundamentally infeasible to keep human meaning in line with the arbitrary exercise of authority that we would like to invest in the owner of a handle. No matter how cleverly we assign names to start with, some change in society will ruin the scheme.

We have arrived about at the present time, around the turn of the millennium 1999–2000 or 2000–2001 or whatever. We can invest a lot of effort into improving the fairness with which conflict over domain names is resolved, and supply more and more domain names to trade off mnemonic quality against cost. But Grampa's ownership of `candy.com` is inherently a lucky and unsustainable windfall, which we cannot provide to everybody who wants it. Whatever contest we set up, only the winner of that contest may have it.

Handles lie down by names. At this point, our story ceases to be mangled history and becomes futuristic science fiction. Unless and until we reach the real millennium, when everyone will live together in peace, names are very likely to attract conflict because of their insidious impact on behavior. Long before highways were super, much less informational, Cyrus Avery recognized the importance of naming when he promoted the development of Route 66 through his hometown of Tulsa Oklahoma. Even after the Road Designation Committee had agreed to construct pavement connecting Chicago, Tulsa, and Los Angeles, Avery fought to have that route given a single name, and the highly mnemonic name of Route 60. He settled for 66, which was at least more mnemonic than 64 or 68. This naming coup established the notion that people would travel from Chicago to Los Angeles through Tulsa, enticed songwriters and television producers to advertise that notion for free, and brought lots of tourist money to Oklahoma that would otherwise have landed elsewhere [1]. The conflict over Internet domain names is the natural successor to the conflict over road names, and it will not be resolved as long as domain names have an impact on commercially valuable behavior.

If we cannot avoid conflict over network names, perhaps we can at least provide conflict-free permanent handles. By locating a system of handles without human meaning at a level of abstraction between names and addresses, we can provide Sally, Paul, and other lovers of grandfatherly candy with a permanent token by which they may reach Grampa as long as he cares to respond to it. We can't help Grampa and his customers hold on to the wonderful mnemonic value of `candy.com`, but they can't keep that anyway, and it's better to keep the handle than to fight a losing battle for the name and keep nothing. `candy.com` is inherently one of a very small number of short memorable names that naturally suggest online acquisition of sweets to all English-speaking Internet users, and we can't give everyone with an interest in candy full authority over it—we should expect it to go to the strongest contender.

Without `candy.com`, how will Grampa attract attention to his business? The same way he always did before his unsustainable domain-name windfall. Although Grampa's candy handle is opaque and unmemorable, friends and satisfied customers will pass copies of it around, using Web browsers and other software that will cater to users' needs to keep track of unmemorable handles. In a pinch, they will read it off to one another as, say, a 12-digit decimal number (somewhere between a 10-digit telephone number and a 16-digit credit card number). The handle will appear behind the scenes in pointers, such as the links in Web pages and their technical successors. People will keep personal directories that resolve "My favorite candy store" to Grampa's candy handle. Grampa will advertise in venues that match his natural clientele and advertising budget, and those venues will associate his handle *locally* with humanly meaningful words, pictures, and other tokens, since he can't afford to acquire and defend a global association. Aggressive indexing services, such as the current *Yahoo* and *Google*, will organize Grampa's candy handle into their own presentations of the informational structure of the Web and its technical successors. And, as long as global domain names last, Grampa can fight for `candy.com`, or make a strategic retreat to `grampascandy.com`, or fall back further to `grampascandyonmainstreetintinytownusaearthsolarsystem.com`, or .... But I think that, in the long run, he will get more satisfaction from the alternatives.

References

[1] Quinta Scott and Susan Croce Kelly. *Route 66: the highway and its people.* University of Oklahoma Press, Norman OK and London, 1988.