A Unified Model for End Point Resolution and Domain Conversion for Multi-Hop, Multi-Layer Communication

Yu-Shun Wang and Joseph D. Touch
USC / Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292
1 310 8221511
{yushunwa,touch}@isi.edu

John A. Silvester
University of Southern California
University Park Campus
Los Angeles, CA 90089
1 213 740 2311
silvester@usc.edu

ABSTRACT
This paper presents an architectural model called the Multi-Domain Communication Model (MDCM) to describe the relationship between protocol layers, network hops and regions of multi-hop, multi-layer communication systems. MDCM treats communication processes as a series of recursive domain conversions and propagation within each domain. The concept of domain is a generalization of protocol layers and transit hops. MDCM includes end point resolution to map source and destination from one domain to the other, such as name/address resolutions, forwarding lookups, and content searching. MDCM integrates these aspects of communication processes and abstracts the core functionality into a simple, recursive model. It can describe a wide range of communication systems, and provide a new way of thinking regarding communication processes and system architecture designs.

Categories and Subject Descriptors
C.2.1 [**Network Architecture and Design**]: Network communications

Keywords
Communication Model, Protocol Stack, Network Architecture.

1. INTRODUCTION
The Multi-Domain Communication Model (MDCM) proposed in this paper unifies protocol layers and communication hops into a single logical construct called domains. It describes communication as a process of recursively traversing through a series of domains, and propagating within each domain from the source to the destination. MDCM assumes that an instance of communication starts when a **source** tries to send a **message** to a **destination** at a certain level (**domain**). The source and destination relative to each domain may change through the course of communication. MDCM describes how communication determines and converts from one domain to another, and how the domain-specific source and destination are resolved through the resolution function in each domain.

The focus of this paper is to define MDCM, and demonstrate its applications in system design. MDCM abstracts the core functionality from all aspects of communication processes into a simple model, provides a new way of thinking and examining systems and architectures. It also highlights an important aspect of communication, the resolution functions that glue different domains together. These include mechanisms for name/address resolutions, forwarding lookups, and content searching. Part of the motivation for MDCM is to investigate these resolution mechanisms.

1.1 From BGP/ARP/DNS to Google
BGP, ARP, and DNS are some of the most commonly used protocols in the Internet. ARP *resolves* IP addresses into corresponding Media Access Control (MAC) addresses in a broadcast LAN. BGP computes the Autonomous System (AS) paths which are used in routing lookups to *resolve* packet destinations into exit gateways in a transit AS. DNS *resolves* hostnames into IP addresses in the Internet. The common thread of BGP, ARP, and DNS is that they *resolve* a given entity from one domain into a corresponding entity in another domain. Going further, Google [9] (or any other web searching engines) *resolves* queries into a set of uniform resource locators (URL) in the Web domain.

This observation shows that the resolution mechanism plays an important role throughout communication at all stages. It also leads to the characterization of communication between two entities as successive resolutions of end point identities and domain conversions, plus propagation within each domain. The result is MDCM:

"**Communication between two entities in a multi-hop, multi-layer environment will go through a series of domains recursively, resolving the effective source and destination for each domain, and propagating within each domain until the message is delivered to the destination.**"

Note that "domains" in this context has a broader definition than just protocol layers; it encompasses administrative regions such as AS's, and name spaces such as DNS. MDCM also incorporates the recursive aspect of traversing protocol stacks into domain conversions. Although derived from observations of network communication, MDCM can be applied to a wide range of communications such as the inter-person communication example shown in Figure 1.

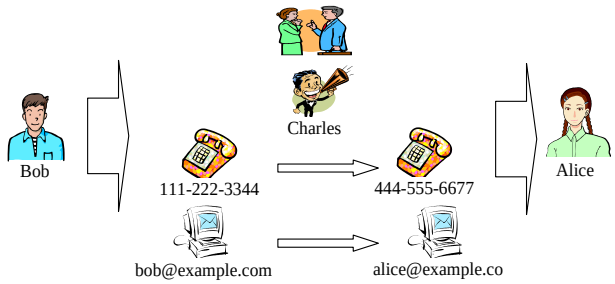

Figure 1. Inter-Person Communication Example

In Figure 1, Bob has a message for Alice. Each option in the picture, (direct conversation, message passing, phone, email) corresponds to a different domain in MDCM. It also shows the domain-specific source and destination respectively. Note that Figure 1 simplifies one subtle detail: direct conversation, message passing through Charles, and phone all belong to the conversation domain. In MDCM, the communication enters the *conversation domain* with Bob as the source and Alice as the destination. Within the conversation domain, if Bob could speak to Alice directly, the message is delivered. Otherwise, Bob needs to choose between the *message passing domain* and the *phone domain*. Bob will also need to *resolve* the source and destination for the domain he chooses, Bob and Charles for the first hop of the message passing domain or the phone numbers for the phone domain.

In the inter-communication example, MDCM covers different styles of communication systems, from single-hop, single-layer channels (direct conversation), to multi-hop, multi-layer network communications (email, phone). It decomposes communication into domains, and illustrates the resolution of domain-specific sources and destinations, although the details of message format transformation and operations inside the more complex domains such email and phone are not included.

1.2 Rationale for MDCM

MDCM presents a complete picture of multi-hop, multi-layer communication processes. It decomposes communication into domains, and captures both protocol layer traversing and multi-hop forwarding behaviors through searching and selecting domains, and the resolution of source and destination entities relative to each domain. This unique combination of MDCM complements other existing models. The concept of domains and the process-centric approach also provide a new way of thinking for system designs from the network architectural perspective down to node-level implementation details.

1.2.1 Other Communication Models

"Communication Models" are often categorized by two mostly disjoint principles: Shannon's model[1] [16] of communication system and protocol stacks such as the ISO/OSI 7-layered protocol stack [10] and the TCP/IP protocol stack [5]. Shannon's model, as shown in Figure 2, is used to analyze signal transmission, effects of noise in the channel, and how "information" is derived from such a signal. It assumes the source and destination are known and fixed. This simple model corresponds to a Physical domain in MDCM, concerning *how* a message makes forward progress. MDCM, on the other hand, emphasizes more on *where* to deliver a message through the successive resolutions of source and destination entities.

[1] Shannon is famous for many models in the field of communication and information theories. The model mentioned in this document is concerning the simple system presented in [16] used for studying channel signal encoding.

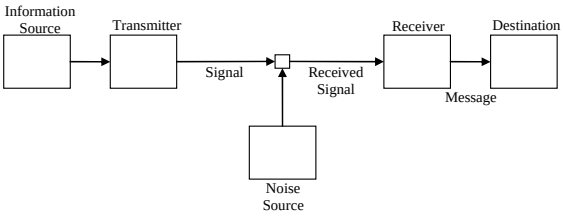

Figure 2. Shannon's general communication system (adapted from [16])

Protocol stacks define the relationship between different protocol layers according to their functionality. Although they form the foundation for multi-layer communication systems, as shown in Figure 3, protocol stacks treat communication as point-to-point processes. The multi-hop aspect of communication is introduced by the concept of routing and forwarding which concatenates hops of protocol stacks together to form a multi-hop system, as shown in Figure 4 depicted by the two curly arrows. Routing and forwarding mechanisms together define the topological relationship between nodes *within* a single protocol layer, the IP layer in this example, to form a multi-hop domain. They operate within a single layer and do not affect the relationship with other protocol layers. This explains why routing and forwarding are often treated orthogonally outside of protocol stacks.

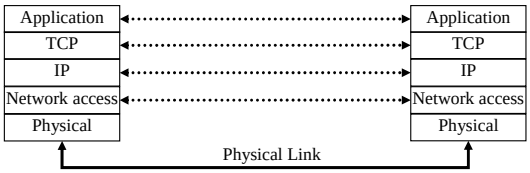

Figure 3. Multi-layer (TCP/IP) end-to-end communication system

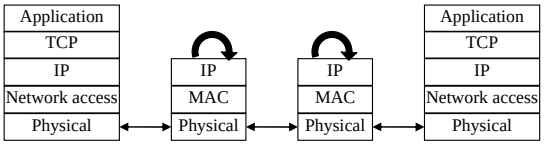

Figure 4. Multi-hop, multi-layer communication system

Another feature of protocol stacks is that they define the syntactic transformation of information. Each protocol layer embeds various control information onto the message by encapsulating protocol headers, trailers, or options. This rules out any operations that do not result in encapsulation or de-capsulation of protocol headers/trailers, further limits the scope of protocol stacks in describing communication processes, such as IP forwarding in the Internet.

MDCM complements protocol stacks by integrating routing and forwarding lookups explicitly into the model. Because MDCM defines domains as encompassing protocol layers and communication regions and hops, the act of selecting the next domain covers both traversing a protocol layer vertically and propagating toward a next hop horizontally. Resolutions of source and destination identities in the next domain correspond to the mapping of identities across protocol layers and also the resolution of the next hop entities.

To summarize, MDCM allows non-deterministic specification of the destination, and built into the model the successive refinement resolution of the source and destination at each domain based on the given criteria. The explicit end point resolution in MDCM is essential to describe multi-hop communication systems where the intermediate source and destination change with respect to each domain it traverses. Searching and selecting next domains in MDCM also integrates several important mechanisms loosely categorized under name and route lookups, and content searching functionality that are not included in protocol stack specifications of communication systems. These key differences of MDCM represent the often missing processes and bridges the semantic gaps left by existing communication models.

1.2.2 New Ways of Thinking

From the comparison above, the key differences between MDCM and other existing communication models are the broader definition of domain, the inclusion of routing/forwarding, and the emphasis on identity resolution in MDCM.

The integration of routing and forwarding into MDCM is a direct result of broadening the definition of domains to including hops and regions. Although routing protocols and forwarding mechanisms are extensively studied and developed for

production use in the current Internet for quite some time, it is still important to integrate the design perspective of routing and forwarding into any future models. The current Internet has routing and forwarding only at the IP layer. Anything above or below IP is either end-to-end or point-to-point in comparison. But the trend is changing. Routing at layers above and below IP is being explored for advanced services or alternative infrastructures. Peer-to-peer services, content-based routing, inter-planetary networks (IPN) [3], and proxy service redirections are among the forefront to adopt routing at layers above IP. At the other end of the spectrum, MPLS (Multi-Protocol Label Switching)/ATM, and other multi-hop layer 2 topologies (wireless, cellular, and satellite networks) introduce routing at infrastructures under IP. These examples all introduce new multi-hop, routable domains into the existing communication systems, resulting in stacking of multi-hop routing domains.

MDCM provides a new way of thinking for designing such systems. Domains in MDCM supersede the concept of protocol layers. It not only defines the relationship between a new domain and layers above and below, just like the protocol stack, but also provides a pragmatic way of defining both intra-domain and inter-domain information exchange. The intra-domain information exchange determines the forwarding behavior and the topology among entities at the same layer, while the inter-domain information exchange is closely related to the resolution of identities across domains. The later is of particular importance when stacking two routable domains together, as the example shown in Figure 5. The information exchange within the upper domain between *S1* and *D1* will require some IBGP-like protocol to establish mappings of identities onto the lower routable domain, *S1→S2* and *D1→D2*, as illustrated in Figure 5. Section 3 will discuss the application of MDCM to analyze information exchange requirements for routable, multi-hop domains in more details.

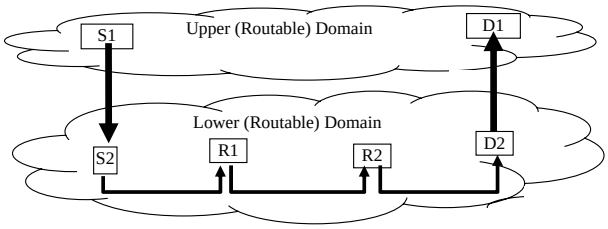

Figure 5. Stacking of two routable domains

MDCM can also be used to guide the design of per-node communication processes with regard to new systems or domains. MDCM focuses on describing the processes of a communication instance traversing through a series of domains as a combination of various mechanisms happening at each node; namely the mechanisms to search and select the next domains, and to resolve the source and destination in the chosen domain. It breaks down architectural aspects of domains into node-level operations. This process-centric approach complements existing protocol design methodology, and provides a universal framework for implementing multi-hop, multi-layer communication functionality down to the node level.

From the overall system perspective, MDCM distills the processes of communication and various protocol layers, and abstract the core functionality relevant to communication into a simple, recursive model. Although this minimalist approach ignores certain properties and details of specific systems, it preserves only the necessary mechanisms to make forward progress. As a result, it is very easy to compare and analyze seemingly different systems. The intent is to reuse similar methodology and components in a modular fashion, reduce duplicate functionality, and improve overall efficiency in system design.

1.3 Organization

The paper is organized as follows: Section 2 describes MDCM in details, including the conceptual framework, components, and operations. Section 3 explores the applications of the model. Section 4 discusses the prior and related work. Although the model itself encompasses the whole range of communication systems, the applications and most of the discussion concentrate on the aspect of computer networks, especially the Internet.

2. MDCM – MULTI-DOMAIN COMMUNICATION MODEL

This section defines the components of MDCM, and its operational details. A symbolic, function form description of MDCM is presented at the end of this section. The Model assumes that any communication starts in a domain with a source trying to send a message to a destination. Before proceeding, the model is re-stated below:

Communication between two entities in a multi-hop, multi-layer environment will go through a series of domains recursively, resolving the effective source and destination, and propagating within each domain until the message is successively reproduced or delivered to the original destination.

2.1 MDCM – Domains

This section describes the concept of **domains** in the Multi-Domain Communication Model (MDCM), covering its definition, components and various types of domains.

2.1.1 Definition of Domains

"Domain" is the central building block of the model. The other components and functions facilitate selection of domains and resolving the entrance and exit points for each domain. The major part of the communication is spent in traversing domains, including physical propagation and logical, syntactic transformation. Domain could take many forms, such as a segment of physical media, a hop in a network, a layer in a protocol stack, or a name space. The following definition of domain captures the core characteristics of any domain, and treats other features as optional properties of each individual domain.

Definition 1. **A domain in the Multi-Domain Communication Model is defined as a single name space encompassing the source and destination connected by a physical or logical channel in the context of the current instance of communication.**

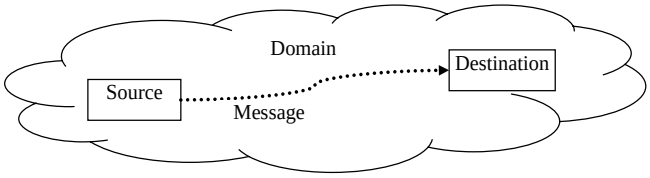

Figure 6. Diagram of a domain

Figure 6 shows a conceptual diagram of a domain. There are two types of domains: **Protocol Layer Domains** and **Transit Domains**.

Protocol Layer Domains, as the name indicates, apply functionality to a message defined by the corresponding protocol layer. They perform protocol-specific control operations and encapsulate the resulting control information on a message in the form of headers, trailers, or options. **Physical Domains** are a special case of Protocol Layer Domains that correspond to physical layer protocols. A message entering a Physical Domain will be converted into appropriate formats and transmitted across the physical channels. Because a message can only make forward progress in Physical Domains, Physical Domains always terminate when a message reaches its destination, and the communication leaves the Physical Domain and returns to the previous domain, usually at a higher layer. Same as in protocol stacks, Physical Domains are always the bottom domains in the picture. Figure 7 shows two Protocol Layer Domains (IP and Ethernet) and the corresponding messages.

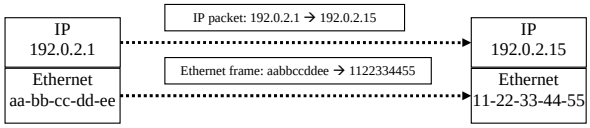

Figure 7. IP and Ethernet Domains with end point addresses

Transit Domains correspond to intermediate hops and regions in a multi-hop communication setting. Converting from a multi-hop domain into a specific hop, or Transit Domain, changes the nature and scope of communication from end-to-end at the higher domain into a single hop. Figure 8 shows the relationship between end-to-end domains and Transit Domains.

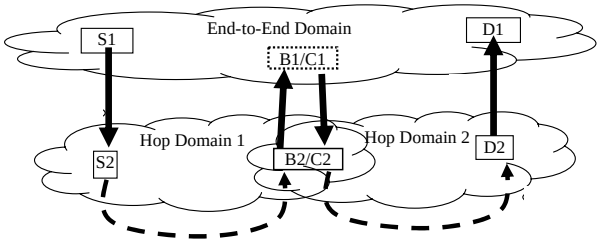

Figure 8. End-to-end Domain and Transit Domains

The fundamental difference between Protocol Layer Domains and Transit Domains is that the former make vertical, syntactic transformation across domains while the latter are responsible for horizontal progress within the same domain. Another

difference between the two is that Transit Domains often do not require protocol header/trailer encapsulation and de-capsulation on messages. If the protocols are the same for the End-to-End Domain and the Hop Domains in Figure 8, which means *S1=S2, B1=B2, C1=C2,* and *D1=D2* as is the case for IP forwarding, the Hop Domains often forgo the protocol encapsulations to reduce processing and header overhead. Although this creates an inconsistency between the source and destination on the message header (*S1, D1*) and the actual source and destination entity in the Hop Domains (*S1, B2,* and *C2, D2*). The logic channels in these non-encapsulating Transit Domains are called **implicit channels** in this paper.

The example in Figure 8 shows that a domain could be either single-hop or multi-hop. If a domain is multi-hop, the name space of the domain must include the names of all nodes within the domain. Next section looks at the components of a domain in more details.

2.1.2 Components of Domain

From the definition above, domains consist of a source, a destination, a channel, a message, and the name space encompassing all entities in the domain.

(1) Name Space: Names, addresses, and other forms of identifiers are all "**names**" used to identify communication entities within a domain. The collection of all the names in a domain forms a name space. The set of names are constructed according to certain rules. A name space could be structured (ordered, hierarchical) or random. It must contain names that could unambiguously identify both the source and the destination, except for certain domains where the identities of the source and destination are implied by the exclusiveness of the channel in between.

(2) Source and Destination: These are the domain-specific source and destination. Externally, they mark the entrance and exit points of a domain. Internally, they are the source and destination of the message inside a domain. During the communication, the domain-specific source and destination will change with respect to domains the message traverses. Their scope is limited to their associated domain. The recursive nature of traversing domains ensures that the completion of communication in the current domain will return it back to the previous domain, which has the source and destination in that domain.

(3) Channel: A channel between the source and destination denotes that the destination is the semantic recipient of the message in the current communication context. In Physical Domains, channels correspond to physical paths. For other types of Domains, channels are logical connections between the sources and destinations.

(4) Message: A message represents a "unit" of information being sent from the source to the destination in a domain. Depending on the style of different domains, a message could be a signal, cell, packet, frame, segment, or even a session, a connection. A message also represents the notion of the controlling thread of communication. It defines the "first-person" view of MDCM.

2.2 MDCM – Operations

Section 2.1 describes the basic concept of "domains" and their core components relevant to the purpose of communication. This section focuses the operations for an instance of communication based on MDCM. An instance of communication is specified by its starting domain $\boldsymbol{\Delta}$, source $\boldsymbol{S}$, destination $\boldsymbol{D}$, and the message (payload) $\boldsymbol{I}$. Written in a function form, the communication could be expressed as a function $\boldsymbol{C}$ (for "communication"):

$$C(\Delta, S, D, I)$$

In the following discussion, assume the return values of this and other functions defined are ignored. In practice, some reliable communication domains will utilize the notion of return values to verify the success and failure of communication. Also assume that all instances of communication are unidirectional. One could combine two unidirectional communications of reverse directions to form a bidirectional instance of communication.

Overall, the goal of communication in a domain is to deliver the information from the source to the destination. Once the message reaches the destination in a domain, the communication in the current domain is complete, and will exit the current domain to return to the previous domain. The previous domain will examine the progress of communication and continue to propagate toward the destination. If the communication can not make any forward progress inside a domain, it will search for and convert to the next domain to continue the progress. The whole process is complete if the destination is the final destination in the original communication domain. The detailed operations are described below.

2.2.1 Transforming Message Format

The first task upon entering a domain is to transform the message into proper format for transmission. The process depends on the specific communication protocols and media of the domains in question. The goal is to encode protocol control information onto messages by encapsulating or modifying headers. This process may also transform messages into media- and protocol-specific transmission formats. The detailed operations include fragmentation, framing, compression, modulation, and translation, etc. Note that this process is not required for non-encapsulating Transit Domains because they do not change the messages. The following function ***F*** (for "formatting") represents this transformation of the message from ***I*** to ***I'***:

$$I' = F(\Delta, S, D, I)$$

2.2.2 Making Progress within a Domain

The goal in a domain is to advance the message from the source to its destination in the current domain. MDCM captures this process with the following steps: verify the current location; try to move forward; if not, then explore other domains. These steps are repeated until either the destination in the current domain is reached, or the communication fails.

*2.2.2.1 **[Step 1]** Is it the destination?*

The first step is to check whether the current location is the destination ***D***. Once the message reaches the destination, communication in the current domain is complete, and will exit the current domain to return to the previous domain. It is equivalent to the successful completion of the domain communication function ***C***. If the current domain is the top-level one, the entire communication is complete. Or the function returns to the previous domain.

Otherwise, proceed to the next step.

*2.2.2.2 **[Step 2]** Move forward*

If not yet at the destination, it then examines whether the current domain is a **Physical Domain** connecting the source and destination. If it is, the message is transmitted immediately through the physical channel to the destination. The physical propagation attempt can be formulated as a function, ***P*** (for "propagation"):

$$P(\Delta, S, D, I')$$

After the physical propagation, the current location becomes the destination and the procedure returns to Step 1. Otherwise, proceed to the next step.

*2.2.2.3 **[Step 3]** What's next (domain)?*

This is the most important step in the MDCM operations. The previous steps are still within the functionality of protocol layers. This is where MDCM introduces different concepts regarding domains in the communication processes.

At this point, the communication could not make forward progress in the current domain because it is not a Physical Domain. It will have to find another way (domain) to move the message *closer* to the destination. After choosing the next domain and resolving the new, domain-specific source and destination, the communication enters the next domain. The following takes a closer look at these operations.

[Step 3.1] Determine the next domain.

First the communication selects a domain from a set of available domains. MDCM introduces the concept of the set of available domains. It is not new, the forwarding tables in the IP domain are examples of such a set (sets of available next hop domains). The selection criteria are specific to each domain, but the most common rule is to find the domain that is *closest* to the destination. The following function ***S*** (for "searching") collects the available options for the current domain and returns a set of domains:

$$Set\{\Delta'\} = S(\Delta, S, D, I')$$

The actual selection will be applied to this set. The simplest form is to enumerate through the set, as will be shown later in the complete function representation of MDCM. Although in practice the choice of the next domain is often fixed or hard-coded within applications or protocol stacks, the set, or the concept of the set, exists for every domain. MDCM emphasizes the concept of searching and selecting the next domain not just for forwarding choices, but extending to protocol layer stacking. This matches the trend toward a more flexible, diversified communication infrastructure.

[Step 3.2] Resolve new source and destination.

After the next domain Δ' is selected, it is usually necessary to map the source ***S*** and destination ***D*** to their corresponding entities, ***S'*** and ***D'***, in the new domain. This is required when the name space of the new domain is different from the current one. Written as a function ***R*** (for "resolve"), the entity resolution function takes the originating domain, the target domain, and the identities of original source and destination as the input arguments, and returns the new source and destination identities in the target domain, as shown below:

$$(S', D') = R(\Delta, \Delta', S, D, I')$$

MDCM emphasizes the importance of resolution functions. Although they are often not in the direct processing path of most protocol stacks, resolution functions define the relationship between entities from the domains they connect together. This relationship determines the information exchange between the two domains necessary to support the connection. It is one of the most important aspects when designing and introducing any new domains to existing systems.

Although finding the next domain and resolving the new entities are described as separate functionality, there are domains that combine these two into a single operation in which the new domain and its corresponding source and destination are returned all together as a result of searching the next domain.

[Step 3.3] Enter the next domain.

At this point, the communication is ready to enter the next domain, ***Δ'***, with the new source, ***S'***, and destination, ***D'***, transmitting the message ***I'***. This corresponds to the recursive function call to the original communication function ***C*** with the new arguments:

$$C(\Delta', S', D', I')$$

The communication instance in the new domain Δ' follows the same procedures starting from Step 1 as described above. Operation in the current domain, Δ, is suspended waiting for the next domain. When the communication in the next domain ***Δ'*** completes, that is, the message reaches the destination ***D'***, it returns to the current domain but at the new location of ***D'***. Care must be taken before immediately returning to Step 1 to check the current location because ***D'*** is an identity from the name space of domain ***Δ'*** which could be different from the name space of the current domain Δ. Because a reverse resolution function between the name spaces of Δ' and Δ might not exist, the comparison is often done by comparing ***D*** with the all names of the current location in the same name space. With the updated current location, the communication returns to Step 1.

2.3 MDCM – Functional Representation

This section presents a pseudo-code representation of MDCM in a function form based on the description in the previous section. It is not in a formal notation, but an attempt to demonstrate the simple form of the model in a clear, recursive format. The functions and subroutines, ***C***, ***F***, ***P***, ***S***, ***R***, are described in the previous section. Note the slight variation in the order of events when checking for current location and checking whether the current domain is a Physical Domain. Although the description above is conceptually correct and clear, the function form shows the inefficiency of direct translations from the steps into pseudo code. A simple optimization results in the following function. This is yet another example of why symbolic representation is useful in formulating models.

```
C(Δ, S, D, I){
  I′ = F(Δ, S, D, I);
  if(P(Δ, S, D, I′)){
    return success;
  }else{
    while(Current_Location ≠ D){
      Finish_a_hop = 0;
      Set{Next_Domains} = S(Δ, S, D, I);
      for Δ′ in (Set{Next_Domains}){
        if((S′,D′) = R(Δ, Δ′, S, D, I)){
          if(C(Δ′, S′, D′, I′)){
            Finish_a_hop = 1;
            break;
          }
        }
      }
      unless(Finish_a_hop){
        # Could not find another domain. Communication has failed.
        return failure;
      }
    }
  }
  return success;
}
```

3. APPLICATIONS OF MDCM

This section discusses several examples of applying MDCM to system design and analysis. Different aspects of MDCM apply to different parts of system architecture. The following examples range from examining a complete protocol stack, designing a single protocol layer, and analyzing information exchanges and resolution functions across domains. The analysis using MDCM is not very different from other types of models. First, the model is used to describe or 'fit' the systems in question. Then the result is analyzed and examined to find problems of existing systems or provide recommendations for new systems. The goal of this practice is to demonstrate how MDCM could be used in system design. As a result, it may raise more questions than providing answers or insights. The solutions to each of the problems listed below are subjects of their own papers, and are thus out of scope for this paper.

3.1 Protocol Stack Analysis – TCP/IP

The first example is to apply MDCM to the TCP/IP protocol stack. Note that this exercise does not intend to reveal new insights into the Internet protocol stack because it has been extensively studied; but rather as an example of how MDCM could help in analyzing a communication system.

Figure 9 shows the beginning of a web transaction up to the first hop gateway. The communication process is decomposed into domains, and the figure also shows the changes in domain-specific source and destination entities with the corresponding resolution functions. It demonstrates how MDCM operations can be applied to a communication system, in this case, the Internet. What is not shown in the figure is that at the first hop gateway, the communication will exit the Ethernet domain and the IP Hop domain, and return to the IP domain. Similar picture of another IP Hop domain and below will repeat until the final destination of the IP domain, 192.0.2.233, is reached. For the following discussion, it is enough to consider only the first hop.

A simple comparison of all domains in the figure reveals that every domain except Transport and IP Hop domains encapsulates the source and destination identities in the headers of the message. No header is encapsulated in IP Hop domain, while the Transport domain header (TCP) does not contain complete identities of the communication entities.

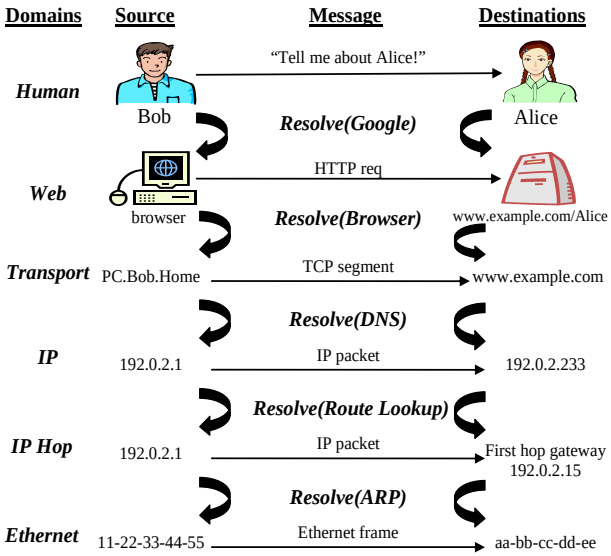

Figure 9. Web communication example with MDCM

IP Hop domain is an example of non-encapsulating Transit Domains. It changes the communication scope from end-to-end at the IP domain (between Bob's PC, 192.0.2.1, and the web server, 192.0.2.233) to the first IP hop (between Bob's PC and the first hop gateway, 192.0.2.15). IP Hop domain uses the same protocol (IP) as its parent domain, the IP domain. Encapsulating the packet with an additional IP header will not change the communication behavior, but will result in additional processing overhead. But the same reason does not apply to the Transport domain.

The omission of end point identifiers such as DNS names in the Transport domain has a more profound effect. This creates a gap in identifiers between the Web domain and the IP domain in this example, effectively binds the communication instance at Transport domain to the IP addresses. As a result, IP addresses become both end point identifiers (names) for Transport domain connections, and locators (addresses) at the IP domain. It is not an issue if each host has only one static IP address. As multi-homing and mobility become more prevalent, this assumption no longer holds, and connections can not survive the change of end point IP addresses (re-homing) under this architecture. There have been many proposals for supporting multi-homing or mobility at the IP layers. Next example will use MDCM to examine one generalized solution to this problem.

3.2 Design Methodology – End Point Identity Domain for Mobility/Multi-Homing

Several mobility and multi-homing solutions add a new layer for end point identifiers (EID) to separate IP addresses into the role of locators only. This section will not explain the details of these solutions, but will examine the concept of the EID layer with MDCM, and show how MDCM can help construct a new domain. The process is divided into two parts: the domain itself and the relationship of the new domain with the others. However, issues from both parts are closely related to each other.

(0) Domain type: The first step is to determine the type of domain. For the case of the EID layer, it is clearly a Protocol Layer Domain. The components of a domain include a name space, source and destination entities, a channel, and messages. Different domains will emphasize on different components. The following steps go though the list of components.

(1) Name space: The focus of the EID domain is on the name space. One needs to consider both the format and scope of the name space. For header encoding, a fixed-length, structured format is more suitable. DNS names and X.500 name spaces both suffer from less rigid formats, but are nonetheless more human understandable. On the contrary, IPv4 and IPv6 are good examples of structured name spaces with fixed formats designed for efficient header encoding, though they are impractical for human manipulation. Scopes of a name space include both physical scope and temporal scope. The former determines the physical and administrative boundary of name spaces, whether they are globally unique or locally unique. The latter concerns whether a name space is persistent or opportunistic.

(2) Source and **destination:** These are about the assignment of names to entities. It depends on the properties of the name space. Permanent name space can be assigned centrally or delegated to certain local administrations. Opportunistic identifiers are often negotiated between the communication parties in real time.

(3) Channel: Properties of the **channel** are defined by the control functions the protocol layer supports.

(4) Message: Messages define the unit of communication, and are related to issues of formatting and fragmentation.

The next part determines the relationship of this new EID domain with other domains. The following questions all have to be answered:

(5) Where does this domain belong in the communication process? What are the previous and following domains for the new domain? For the EID domain, it will have to come before the IP domain. It could be place between Transport and IP domains, or above the Transport domain. Note that neither approach will change the fact that Transport domains (especially TCP and UDP) in the Internet still do not have a complete end point identifier. The location of a new Protocol Layer Domain also dictates the scope of modifications for its immediate neighboring domains.

(6) How does the previous domain see/choose it? For most end-to-end Protocol Layer Domains, the choice is either to encode the domain ordering in the protocol stack implementation, or within the application configurations.

(7) How does the previous domain resolve entities to the new domain? This is partly determined by the type of the selected name space. Assume the previous name space is DNS, a persistent EID name space could probably be resolved by DNS queries with new data types. On the other hand, an opportunistic EID name space could probably be resolved by on-demand negotiation exchange. But the latter approach assumes the sender has the knowledge to establish initial contacts with the receiver at this stage.

(8) What (next) domains are available at this new domain? This is related to how and what domains are made available to the EID domain. If only a single domain is available, one can encode the selection into the current domain. Multiple options require mechanisms in the current domain to detect and select available domains. In this example, the obvious answer is the IP domain, though it would be useful to have both IPv4 and IPv6 available from the EID domain. This could enable more flexible communication structure.

(9) How does the new domain resolve entities to the next domain(s)? Finally entering the next domain, the new EID names need to map to the addresses in the succeeding domain. The resolution function depends on the type of name spaces. One possibility is to resolve both the EID and IP address from DNS names in one step. This approach also implicitly defines the mapping of the resulting EID to IP address. Or use the IP address to negotiate an opportunistic EID. The advantage of this proposal is it utilizes existing infrastructure and services. But it assumes the target hosts are reachable through the addresses stored in the DNS database, which might not be true for all mobility and multi-homing models.

This simple exercise goes through the components and operations of domains, and shows how MDCM can be used as a guideline in defining new domains. It can also provide a systematic way of verifying and comparing existing domains. Note that the example describes a non-routable protocol layer domain. The focus and issues will be different for other types (multi-hop routable or region/hop) domains. Next section considers the resolution function between two multi-hop, routable domains.

3.3 Resolution Function – BARP

This example describes a problem presented by a new Virtual Internet architecture [20] and discusses the corresponding solution through the new resolution function called BARP and the inter-domain information exchange.

Virtual Routers and BARP are the real motivations behind the DNS/BGP/ARP observation presented earlier. DynaBone [19] extends the generic virtual network architecture of the X-Bone [18] to construct parallel, multi-layer virtual networks for fault-tolerance against denial of service (DoS) attacks on the infrastructure. It encapsulates the lower level virtual networks as routers at the upper level as shown in Figure 10.

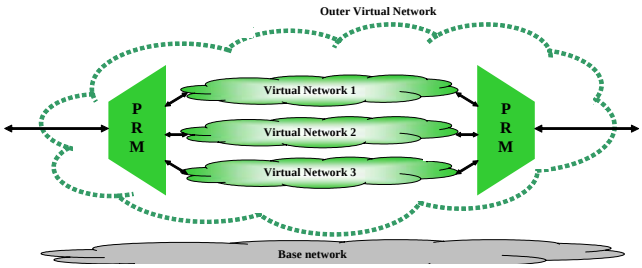

Figure 10. DynaBone Block Diagram

BARP is developed to resolve the entrance and exit points inside a virtual network router (Inner Domain) for given source and destination at the outer virtual network domain (Outer Domain). Figure 11 shows the diagram of a virtual network router and the corresponding packet headers for BARP operations. The name "BARP" came from the combination of ***B***GP+***ARP***. Both BGP and ARP "resolve" exit points of given destination addresses in a domain (AS and Ethernet LAN respectively). ARP encapsulates the IP packets with the newly resolved addresses, BGP does not. On the other hand, BGP establishes the forwarding information (internal BGP or IBGP) within a multi-hop AS, ARP only works for a single, broadcast LAN. The combination of BGP and ARP provides a solution to the virtual network router problem.

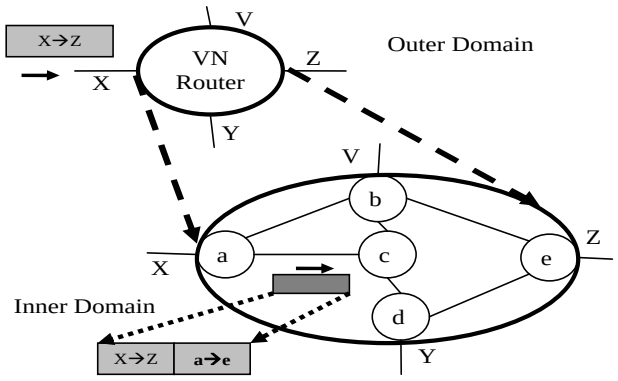

Figure 11. Virtual Network Router & BARP packet diagram

For the BARP resolution function to work, the *inter-domain* information exchange between the Outer Domain and the Inner Domain is the same as IBGP if the Inner domain is treated as an AS. This is because IBGP passes *external* information within an AS (*domain*) to establish the corresponding *exit* points for external entities. Following this analogy, EBGP in this picture becomes the *intra-domain* information exchange within the Outer Domain. The result of this analysis applies to the information exchange between any two routable, multi-hop domains stacked together. This has become increasingly common when implementing multi-hop, routable infrastructure above and below IP as discussed in Section 1.2.2.

Rbridges [14] is very similar to the virtual network router architecture in DynaBone. But instead of virtual IP networks as routers, Rbridges enclose bridged multi-hop Layer 2 networks. Rbridges could also omit the encapsulation of the entrance and exit point (the additional header in Figure 11) when transmitting only IP packets.

4. RELATED WORK

Section 1 compares MDCM with both Shannon's Communication Model and protocol stacks. This section reviews several network architectures with similar concepts as MDCM. It also includes some new system designs that can be categorized as new domains in MDCM. The last part summarizes common resolution functions currently used in the Internet.

4.1 Network Architecture

Most new network architectures are designed to solve specific problems of existing networks, such as the Internet. The majority proposed changes to different parts of existing networks, such as adding new domains (protocols, layers or namespaces) or new resolution mechanisms. Few proposed completely new communication systems. Some works included here are based on similar concepts as those of MDCM, such as the region-based architecture. Others attempt to define new models with completely different approaches or propose new domains.

4.1.1 Region-Based Internetworking Architecture

Several projects proposed the concept of "region" in the Internet as an explicit architectural component to represent a partition of the network with "consistent control, state, or knowledge." A collection of interconnected regions represents a connected set of heterogeneous networks. Special gateway (or "waypoints") entities reside on the region boundaries to facilitate mapping of identities, exchange of routing information, and translation of message formats across the boundaries. **Catenet** [4] is one of the earliest models based on this notion[2]. **Metanet** [23] is a recent white paper on this type of architecture; the **Regions** project [17] is a derivative of the Metanet.

[2] The term "Internet" originated in part from the notion of a single network formed by **inter**-connecting heterogeneous (sub-) **net**works, which correspond to "regions". Catenet was one model describing certain aspects and criteria of the early Internet model in the late 1970.

The concept of "region" is similar to that of "domains" in MDCM, as is the aspect of communication traversing multiple regions to reach the destination. The key difference is that regions are used to capture the partitions of homogeneity in the larger interconnected heterogeneous networks. They have the notion of membership associated with each region, and the operations are to make horizontal propagation across regions only. MDCM, on the other hand, is a generic model for multi-hop, multi-layer communication, the definition of domains is much broader than regions in that any switching of contexts is considered a domain with some domains being ephemeral. MDCM recognizes and defines generic operations of searching and selecting domains, resolving identities across domains that exist at any point of the communication rather than only when crossing boundaries of different network paradigms.

Plutarch [7] is another architectural framework based on the "region" concept, called "contexts" in Plutarch. Plutarch's "contexts" are similar to "domains" in MDCM, but the operations are different. Plutarch relies on explicitly retaining a "chain of contexts" to achieve end-to-end communication. It establishes states as chains of contexts, similar to those of virtual circuits in Asynchronous Transfer Mode (ATM). This forms a virtual layer to interconnect heterogeneous networks. In MDCM, the searching and selecting of the next domain happens within each domain. MDCM is orthogonal to whether the selection relies on information established in the form of stateful circuits or stateless forwarding entries. Finally, MDCM is proposed as a model to describe generic multi-hop, multi-layer communication with a generic set of operations rather than a virtual layer connecting different networks as stated by Plutarch. **4+4** [21] is similar to Plutarch, but is limited in the network layer (IP) to bridge multiple NAT regions. **Realm-Specific IP** (RSIP) [1] is also based on the concept of regions, but uses tunneling across non-native regions rather than translating packets at the waypoints or gateways on the borders.

4.1.2 Role Base Architecture

Role-Based Architecture (RBA) [2] is a departure from the layered model of protocol stacks. It defines different "roles" for various features arranged as an un-ordered heap in the header instead of layered encapsulation. This allows more flexibility in inserting features or performing operations (both are roles in RBA) onto a packet at any point in the network without worrying about modifying protocol layers or layering violation.

By defining hop-by-hop forwarding as one of the "roles", RBA could include the forwarding/routing aspect of multi-hop communication into its architecture. But RBA focuses more on the decomposition of protocol layer functionality into modularized roles while MDCM concentrates solely on searching, selecting domains and resolving the entities across domains. MDCM is orthogonal to whether the processing uses either traditional layered encapsulated headers or the RBA-style heap headers.

4.1.3 Architectures with New Domains

These remaining new architectures generally fall into the new domain category according to MDCM; that is, they proposed new domains in the forms of protocol layers, new name spaces, or both for different purposes. Adding a new domain could serve to bridge or extend existing name space to cover heterogeneous regions, add new routing paradigms, provide abstractions or indirections, or introduce new services. The following list is by no means complete, but rather a sample of various new domains.

TRIAD [6] adds a shim layer (content layer) and uses URLs as the name space to support content routing, caching, and transformation. **IP Next Layer** (IPNL) [8] adds a new protocol layer using the DNS name space for both routing and end point identification. **Network Pointers** [22] add a new layer to provide a framework and mechanism that hides the details of the underlying multi-hop network and presents it as a single-hop LAN.

Host Identity Protocol (HIP) [11] defines a new layer reside logically between DNS names and IP addresses to support multi-homing and mobility. The host identifier namespace is used to establish a shim layer between Transport and Network layers. This separation of names and addresses enables connections to survive re-homing (or renumbering). The earlier **8+8** [12] and its successor **GSE** (Global, Site, and End-system address elements) [13] proposals represent another approach to solve the same problem regarding name spaces. Instead of introducing new name spaces, they split existing name spaces (IPv6 addresses) into fields with different semantics.

4.2 Resolution Functions

Resolution functions in MDCM resolve the source and destination identities in the current domain to their corresponding entities in the next domain. Almost all resolution functions are specific to the domains they bridge together. As a result, most research regarding resolution mechanisms and protocols was either part of the "new" architectures involving new domains or namespaces, or linking existing namespaces that were not previously connected. This is not counting

performance optimizations or feature enhancements of existing resolution mechanisms. The following briefly describes some existing resolution functions (mechanisms, protocols).

DNS, ARP, and BGP are among the most common resolution functions currently used in the Internet. Internet web search and database query portals are another form of resolution functions. Indexing is another type of resolution functions commonly utilized in peer-to-peer systems with different query mechanisms such as complete or random flooding, algorithmic computation (e.g., various DHT approaches [15]), or central database search. Content Delivery/Distribution Networks (CDNs) utilize customized resolution functions to resolve target "contents" to their corresponding locations containing the contents. Overlay Networks add an addition layer on top of the corresponding layer they "overlay," and most require customized resolution functions to map a destination at the overlay layer to its underlying layer. HIP represent a new trend of on-demand, real time resolution functions of opportunistic tags or identities through mutual handshakes or negotiations.

5. CONCLUSION AND FUTURE WORK

This paper presents MDCM, a model for multi-hop, multi-layer communication processes. It defines the concept of domains as the main logical construct for communication, and describes the components and operations with regard to domains. MDCM integrates routing/forwarding behavior and traversing protocol stacks under a single framework, emphasizes the importance of resolution functions in the process.

MDCM is a very flexible model capable of describing a wide range of communication systems. It also bridges different stages and aspects to form a complete description of any communications. This paper shows several examples of applying MDCM to different scopes of system design and analysis.

The future and on-going work on MDCM aims to fully define the supporting functions of MDCM, propose a symbolic representation of the model, perform semantic analysis, and build a node-level prototype of the framework.

6. ACKNOWLEDGMENTS

The authors wish to acknowledge Lars Eggert, Amy Hughes, Venkata Pingali, Joshua Train, Joseph Bannister, Ahmed Helmy, and Christos Papadopoulos for their feedback.

7. REFERENCES

[1] Borella, M., Lo, J., Grabelsky, D., and Montenegro, G., "Realm Specific IP: Framework", RFC 3102, October 2001.

[2] Braden, R., Faber, T., and Handley, M., "From Protocol Stack to Protocol Heap -- Role-Based Architecture", HotNets-I, October 2002; also in *Computer Communication Review*, Vol. 33, No. 1, January 2003.

[3] Burleigh, S., et al., "Delay-Tolerant Networking: An Approach to Interplanetary Internet", IEEE Communications Magazine, June 2003.

[4] Cerf, V., "The Catenet Model for Internetworking", Internet Experiment Notes IEN 48, July 1978.

[5] Cerf, V. and Kahn, R., "A Protocol for Packet Network Intercommunication", *IEEE Transactions on Communications*, Vol. 22, No. 5, May 1974, pp. 647-648.

[6] Cheriton, D. and Gritter, M., "TRIAD: A New Next-Generation Internet Architecture", http://www.dsg.stanford.edu/triad, July 2000.

[7] Crowcroft, J., Hand, S., Mortier, R., Roscoe, T., and Warfield, A., "Plutarch: An Argument for Network Pluralism", *Proceedings of the Workshop on Future Directions in Network Architecture (FNDA) at ACM SIGCOMM, Karlsruhe, Germany*, Aug. 2003. Intel Research, IRB-TR-03-023, June 2003.

[8] Francis, P., and Gummadi, R., "IPNL: A NAT-Extended Internet Architecture", *Proceedings of ACM SIGCOMM 2001*, San Diego, CA, August 2001.

[9] Google, http://www.google.com.

[10] ISO, "Basic Reference Model for Open Systems Interconnection", ISO 7498, 1984.

[11] Moskowitz, R. and Nikander, P., "Host Identity Protocol Architecture", (work in progress), September 2003.

[12] O'Dell, M., "8+8 - An Alternate Addressing Architecture for IPv6", (work in progress), October 1996.

[13] O'Dell, M., "GSE - An Alternate Addressing Architecture for IPv6", (work in progress), February 1997.

[14] Perlman, R., “Rbridges: Transparent Routing”, *Proceedings of Infocom 2004*, April 2004.

[15] Ratnasamy, S., Shenker, S., and Stoica, I., "Routing Algorithms for DHTs: Some Open Question", *First International Workshop on Peer-to-Peer Systems (IPTPS)*, March 2002, Cambridge, MA.

[16] Shannon, C. and Weaver, W., “The Mathematical Theory of Communication”, University of Illinois Press, 1963.

[17] Sollins, K., "Designing for Scale and Differentiation"**,** Proceedings of the Workshop on Future Directions in Network Architecture (FNDA) at ACM SIGCOMM, Karlsruhe, Germany, August 2003.

[18] Touch, J., “The X-Bone”, Workshop on Research Directions for the Next Generation Internet, May 1997.

[19] Touch, J., Finn, G., Wang, Y., and Eggert, L., “DynaBone: Dynamic Defense Using Multi-layer Internet Overlays”, *Proc. 3rd DARPA Information Survivability Conference and Exposition (DISCEX-III)*, Vol. 2, April 2003, pp. 271-276.

[20] Touch, J., Wang, Y., Eggert, L., and Finn, G., “Virtual Internet Architecture”, ISI Technical Report, ISI-TR-2003-570, March 2003.

[21] Turanyi, Z., Valko, A., and Campbell, A., “4+4: An Architecture for Evolving the Internet Address Space Back Toward Transparency”, *Computer Communication Review*, Vol. 33, No. 5, October 2003, pp. 43-54.

[22] Tschudin, C. and Gold, R., “Network Pointers”, HotNets-1, October 2002; also in *Computer Communication Review*, Vol. 33, No. 1, January 2003.

[23] Wroclawski, J., “The Metanet”, Workshop on Research Directions for the Next Generation Internet, May 1997.