STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LX**, Number 2, 2015

DOMAIN MOBILE AMBIENTS FOR NETWORK ROUTING SCHEME

GABRIEL CIOBANU AND DAN COJOCAR

Abstract. Ambient calculus is a calculus for mobile computing introduced to describe the movement of processes or devices. Until now the domain capability of an ambient was defined like a static resource. In this paper we add the composition and choice operations for domain attribute in order to express different addressing and routing schemes. The new formalism can be used to easily describe routing schemes like anycast and multicast.

1. Introduction

Ambient calculus was introduced by Cardelli and Gordon in 1999 as a need to express the movement of processes or devices, even between different administrative domains [1]. The ambient calculus differs from other formalisms such as π-calculus [2] in such that the computational model is based on *movement* instead of *communication*. An ambient represents a unit of movement. Ambient mobility is controlled by the capabilities like: in, out, and open. Mobile ambient capabilities are similar to prefixes in CCS [3] and π-calculus. Several variants of the ambient calculus have been proposed by adding and/or removing features of the original calculus [4, 5, 6].

In [1] the definition of mobile ambients is related to network communication. Aman and Ciobanu proposed a new formalism named Timed Mobile Ambients (*tMA*) that adds timers to capabilities and ambients in order to express timeouts in network communication [7]. In [8], Ciobanu extended the formalism for mobility with timers by adding *capacity*, *weight* and *domain* as ambient attributes.

In this paper we extend the use of composition and choice operations, that were defined in [1] for process operations, to be used on *domain* attribute

Received by the editors: August 29, 2015.

2010 *Mathematics Subject Classification.* 14D15,68M14.

1998 *CR Categories and Descriptors.* D.2.4 [**SOFTWARE ENGINEERING**]: Software/Program Verification – *Formal methods*; C.2.4 [**COMPUTER-COMMUNICATION NETWORKS**]: Distributed Systems – *Distributed applications*.

Key words and phrases. formal methods, mobile ambients, DNS lookup.

also. Using these two operations we are able to express routing and addressing scheme like: unicast, anycast and multicast.

The rest of this paper is structured as follows. Section 2 introduces the mobile ambients and Coordination of mobile agents formalism. In Section 3 we present the formalism changes and the reductions rules that address the domain attribute, and how to apply the changes using a DNS anycast sample. Conclusions and further works are presented in Section 4.

2. Mobile Ambients

Cardelli and Gordon introduced the mobile ambients with the main purpose to model the mobility of agents that are involved in distributed computing [1]. An ambient was defined as a boundary place where computation will happen (laptop, web-page, network routing equipments, etc.). Moving operations, like *go*, *in* and *out* were defined in order to express the movement between ambients. An ambient has a name, a collection of local capabilities which control the ambient, and a collection of subambients.

2.1. **Formal Syntax.**

The following table describes the syntax of coordinated mobile ambients.

Table 1. Coordinated Mobile Ambient Syntax

n,m,p		ambient names	P,Q	::=	processes
C	::=	capabilities		0	inactivity
	in n	can enter n		C.(P,Q)	movement
	out n	can exit n		$(n^{\Delta t}_{(l,h,d)}[P], Q)$	ambient
	open n	can open n		P|Q	composition
	go y	migration to y		P+Q	choice
				$M^{\Delta t}.(P, Q)$	movement
				(vn)P	restriction
				*P	replication

A migration *go y* changes the domain d to a domain y. A capability $open^{\Delta t}n.(P, Q)$ evolves to P whenever in Δt the process becomes sibling to an ambient n ; otherwise evolves to Q. An output action $! < m >^{\Delta t} .(P, Q)$ evolves to P whenever in Δt the process becomes sibling to a process which is intending to capture the name m; otherwise evolves to Q. More details about time-stepping function and semantics of coordinated mobile ambients can be found in [8].

3. Domain Mobile Ambients Behavior

In this section we present our proposal to extend the model presented in [8]. By overloading the domain attribute, in order to enhance the local knowledge of an ambient, we can encapsulate the domain choice of an action. We will be able to express or formalize all sorts of actions and mechanism, like: the choices that a student has when leaving the university location. We can imagine that the student has the choice to visit the following locations: the campus, a library or a local restaurant. Also we could easily formalize TCP/IP unicast, anycast, multicast and broadcast routing schemes [9].

By overloading the domain attribute we can rewrite existing migration rules where the domain is the choice. This way we will be able to write more concise rules.

In the student case, the choice is limited to a single option, the student is not able to visit multiple locations at the same moment. But we can imagine other situations, where a choice can result in multiple parallel action. When the same subject has the choice and the ability to perform the same action on different locations at the same time. Such an example could be a beam of light hitting the surface of a lake, a part of the beam will be refracted and the rest will be reflected.

With different LAN topologies, interconnected by network equipments and all sort of link types, the Internet is a giant graph [10]. In this way all routing problems can be reduced to graph search algorithms. Each vertex (router) in these graphs are maintaining tree structures with informations about other vertexes [11], in order to quickly adapt to network failures.

To be able to apply different classic graph search algorithms, like Dijkstra's algorithm [12] and others, to anycast and multicast schemes, we need a way to easily represent different types of endpoints.

Since the differences between the routing schemes are primary on the endpoint levels, we propose to add composition and choice operations to the domain attribute to be able to formalize all the defined routing schemes.

3.1. **Extending Domain Attribute of an Mobile Ambient.**

To represent an endpoint group, we use the notation d_m^n, where:

- m - represents the total number of nodes in the domain.
- n - represents the number of nodes that are used.

Using the above notations for each routing scheme we can define the following endpoint groups:

- *unicast group* $= d_1^1 \equiv d_1$ - the client request will be routed to the single node that represents the unicast group.

- *anycast group* = $d_m^1 \equiv d^1$ - the client request will be routed to only one node that is part of the anycast group. Here the anycast group has m nodes, but in this case we will contact only a single node.
- *multicast group* = $d_m^n \equiv d^n$ - the request will be routed to n destinations from the total of m nodes of a network. n represents the total number of nodes in this multicast group, n being less then m.
- *broadcast group* = $d_m^m \equiv d_m$ - the request is routed to all the available nodes, n from the multicast group is equal to m.

For each routing scheme the domains are defined using composition or choice operations:

- *unicast group* - will use the following domain:

$$d_1 = d_i$$

  where i is between 1 and m, meaning that the domain d contains only a single subdomain.
- *anycast group* - will use the following domain:

$$d^1 = d_1 + d_2 + ... + d_m$$

  meaning that the domain d^1 contains the d_m subdomains and a packet send to this domain will go to only one of the defined subdomains.
- *multicast group* - will use the composition operation instead of choice and it will have the following form:

$$d^n = d_1|d_2|...|d_n.$$

  A packet send to a d^n domain will go to all n defined subdomains. Here d_1, d_2, ..., d_n are part of d_1, d_2, ...,d_m network.
- *broadcast* - similar to multicast will have the following domain:

$$d_m = d_1|d_2|...|d_m.$$

Considering the above domain definitions we can define an ambient with an unicast domain in the following way:

$$(n^{\Delta t}_{(l,h,d_1)}\,[P]\,, Q) \equiv (n^{\Delta t}_{(l,h,d_i)}\,[P]\,, Q)$$

where i is one of the subdomains.

Also an ambient with anycast domain has the following form:

$$(n^{\Delta t}_{(l,h,d^1)}\,[P]\,, Q) \equiv (n^{\Delta t}_{(l,h,d_1+d_2+...+d_m)}\,[P]\,, Q)$$

and it will use choice operation on domain field.

Likewise an ambient with multicast or broadcast domain will be represented as follows:

$$(n^{\Delta t}_{(l,h,d^n)}\,[P]\,, Q) \equiv (n^{\Delta t}_{(l,h,d_1|d_2|...|d_n)}\,[P]\,, Q)$$

and it will use composition on domain field.

The reduction rules for migration *go y* operations on ambients with unicast, anycast, multicast or broadcast groups, are presented next:

- Unicast domain:

$$\frac{t^{'}=0}{(n^{\Delta t}_{(l,h,d)}\left[go^{\Delta t'}d_1.P\right],Q)\rightarrow(n^{\Delta t}_{(l,h,d_1)}\left[P\right],Q)}\equiv$$

$$\frac{t^{'}=0}{(n^{\Delta t}_{(l,h,d)}\left[go^{\Delta t'}d_1.P\right],Q)\rightarrow(n^{\Delta t}_{(l,h,d_1)}\left[P\right],Q)}$$

- Anycast domain:

$$\frac{t^{'}=0}{(n^{\Delta t}_{(l,h,d)}\left[go^{\Delta t'}d^1.P\right],Q)\rightarrow(n^{\Delta t}_{(l,h,d^1)}\left[P\right],Q)}\equiv$$

$$\frac{t^{'}=0}{\substack{(n^{\Delta t}_{(l,h,d)}\left[go^{\Delta t'}d^1.P\right],Q)\rightarrow\\(n^{\Delta t}_{(l,h,d_1)}[P],Q)+(n^{\Delta t}_{(l,h,d_2)}[P],Q)+...+(n^{\Delta t}_{(l,h,d_m)}[P],Q)}}$$

- Multicast/Broadcast domain:

$$\frac{t^{'}=0}{(n^{\Delta t}_{(l,h,d)}\left[go^{\Delta t'}d^n.P\right],Q)\rightarrow(n^{\Delta t}_{(l,h,d^n)}\left[P\right],Q)}\equiv$$

$$\frac{t^{'}=0}{\substack{(n^{\Delta t}_{(l,h,d)}\left[go^{\Delta t'}d^n.P\right],Q)\rightarrow\\(n^{\Delta t}_{(l,h,d_1)}[*P],Q)|(n^{\Delta t}_{(l,h,d_2)}[*P],Q)|...|(n^{\Delta t}_{(l,h,d_n)}[*P],Q)}}$$

3.2. **Anycast DNS Lookup.** Sarat and Terzis in [13] have measured that up to 55% of the DNS queries are answered by the topologically closest server when using anycast addresses. Using the above proposed domain attribute changes we can express the workflow of such DNS queries.

A scenario where a client that performs DNS queries, using two public DNS servers that are scattered between three locations, is presented in Figure 1. Because we are using anycast addresses the queries will be handled by the closest server [13]. For example when using 8.8.8.8 address we are only two hops away from server A, compared to three hops to the next nearest server. If we use 8.8.4.4 address the closest server will be server D.

Using mobile ambients, with the proposed domain attribute changes, we are able to express in a concise manner how our client is performing such a DNS lookup query. On the client configuration we have two nameservers 8.8.8.8 and 8.8.4.4. When performing a DNS lookup query the following steps are performed:

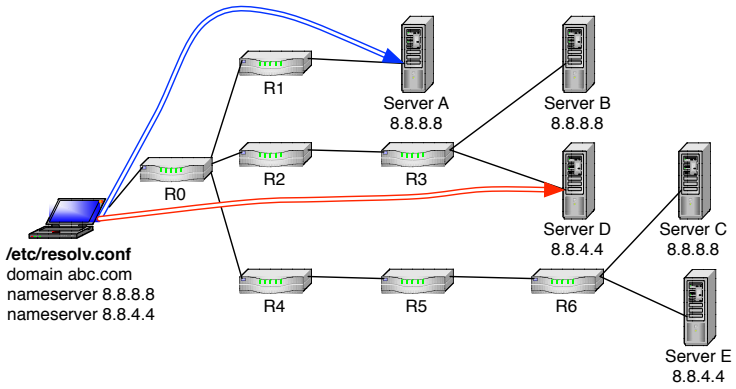

FIGURE 1. DNS lookup using anycast address.

- *Step 1.* - the client is using the 8.8.8.8 address because this is the first configured nameserver, for his DNS lookup query.
- *Step 2.* - the request is processed by $R0$ router. $R0$ knows that server A, B and C all have the 8.8.8.8 anycast address configured, and because the server A is the closest server will forward the DNS lookup to router $R1$.
- *Step 3.* - the router $R1$ is simply forwarding the client request to server A.
- *Step 4.* - server A is processing the client request and responds with a corresponding message.

If one of the above steps the connection is lost, a timeout is reached or the server A is responding with an error, the client restarts the process and performs the same steps using 8.8.4.4 address. Using the second address the $R0$ router is forwarding the request to server D network. If the failed scenarios is repeated in this case too the client will end up deciding that he is not able to resolve the DNS lookup and an error message will be presented to the user [14].

Considering the DNS lookup with mobile ambients we can express the above logic like this:

$$dns\_lookup := (dns\_request^{\Delta t_1}_{(l,h,d^1)}\,[send\_request]\,, process\_error)$$
$$send\_request = send^{\Delta t_2}\,\left[out^{\Delta t_3} client.go^{\Delta t_4}.in^{\Delta t_5} server\right]$$
$$process\_error = (dns\_request^{\Delta t_6}_{(l,h,d^2)}\,[send\_request]\,, display\_error)$$
$$display\_error =! < error\_message >^{\Delta t_7}$$

Where d^1 has the 8.8.8.8 address associated, the first configured nameserver, and d^2 is using 8.8.4.4 address, the failover nameserver, used if an error is received from the first *dns_request* query. If after the second attempt we received an error, a failure message is displayed to the client.

When receiving a request, a router R is performing the following actions:

$$\begin{aligned} process :&= (router^{\Delta t}_{(l,h,d)} \left[forward\_request\right], discard\_request) \\ forward\_request &= send^{\Delta t_2} \left[out^{\Delta t_3} router.go^{\Delta t_4}.in^{\Delta t_5} next\_hop\right] \\ discard\_request &= send^{\Delta t_6} \left[out^{\Delta t_7} router.go^{\Delta t_8}.in^{\Delta t_9} client\right] \\ next\_hop &= send^{\Delta t_{10}} [out^{\Delta t_{11}} router.go^{\Delta t_{12}}.in^{\Delta t_{13}} forward\_request \\ &\quad + server] \end{aligned}$$

The next hop from a router could be the destination server or another router. For each router we are doing the same action: check to see if we can forward the request to one of our servers or to a next router. If the Δt expires, in our case the TTL value id decremented and reaches zero, the router will discard the request and is sending an error code to the client [15].

When everything is working properly and the request is received by the proper server, the service is resolving the request and sends a response back to the client.

$$\begin{aligned} server &= open^{\Delta t_{14}} \left[request^{\Delta t_{15}} | response^{\Delta t_{16}}\right] \\ response &= send^{\Delta t_{17}} \left[out^{\Delta t_{18}} server.go^{\Delta t_{19}}.in^{\Delta t_{20}} client\right] \end{aligned}$$

Because an anycast address is used, on router $R0$, our *process* action is defined in the following way:

$$process := (R0^{\Delta t}_{(l,h,R1+R2+R3)} \left[forward\_request\right], discard\_request)$$

Based on the knowledge that the $R0$ has, it is able to decide that the $R1$ is the "nearest" router that he needs to forward the client request [13]. $R0$ maintains a list of anycast servers, using the routing informations from $R1$, $R2$ and $R3$. In this case the $forward\_request$ action is defined like this:

$$forward\_request = send^{\Delta t_2} \left[out^{\Delta t_3} R0.go^{\Delta t_4}.in^{\Delta t_5} R1\right]$$

4. Conclusion and Further Works

Mobile ambients are a powerful formalism that can be used to express network protocols and all communications between different devices. In this paper we added the composition and choice operations for domain attribute in order to express different addressing and routing schemes. It offers a way

to easily formalize all the routing schemes defined in IPv4, IPv6 and describe routing schemes like anycast, multicast and broadcast. The evolution of ambients is illustrated with an example from real life, DNS routing scheme. In this example we can see how easily is to express the complex operations that are involved when making a DNS request and also how the fallback logic is triggered, when multiple servers are defined. By using this formalism we were able to focus on the big picture and also capture the domain choices.

Using the proposed changes we were also able to write more concise and compact expressions when describing complex network algorithms where the need of multiple domain choices are involved. Having a concise way to express the movement is helping in validating and comparing classic routing algorithms on all type of networks, independent of the used addressing scheme. The formalism can be used to express all sorts of interactions not only the network related ones. Anywhere we have one-to-one, one-to-(one-of-many) or one-to-many relations, between entities that are interacting somehow, we can formalize the conversation.

As further work, we would like to express the interaction between the processes that are happening in a distributed network, like load balancing or map reduce, where multiple queries are sent to different nodes in order to establish the answer, or a partial answer, as quickly as possible. Also we are further considering to tackle and formalize the movement of particles when two object are colliding.

References

[1] L. Cardelli, A. D. Gordon, and G. Ghelli, "Mobility types for mobile ambients," in *ICAL '99: Proceedings of the 26th International Colloquium on Automata, Languages and Programming*. London, UK: Springer-Verlag, 1999, pp. 230–239.

[2] R. Milner, *Communicating and mobile systems: the π-calculus*. New York, NY, USA: Cambridge University Press, 1999.

[3] ——, *Communication and concurrency*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989.

[4] M. Bugliesi, G. Castagna, and S. Crafa, "Boxed ambients," in *In Proc. TACS 2001, LNCS 2215*. Springer-Verlag, 2001, pp. 38–63.

[5] F. Levi and D. Sangiorgi, "Controlling interference in ambients," in *POPL '00: Proceedings of the 27th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. New York, NY, USA: ACM, 2000, pp. 352–364.

[6] D. Teller, P. Zimmer, and D. Hirschkoff, "Using ambients to control resources," *Int. J. Inf. Secur.*, vol. 2, no. 3, pp. 126–144, 2004.

[7] B. Aman and G. Ciobanu, "Timed mobile ambients for network protocols," in *FORTE '08: Proceedings of the 28th IFIP WG 6.1 international conference on Formal Techniques for Networked and Distributed Systems*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 234–250.

[8] G. Ciobanu, "Coordinated mobile agents," *Proceedings of the Romanian Academy Series A-Mathematics Physics Technical Sciences Information Science*, vol. 10, no. 1, pp. 73–79, 2009.
[9] J. Postel *et al.*, "Rfc 791: Internet protocol," 1981.
[10] Z. G. Daniel, D. R. Figueiredo, S. Jaiswal, and L. Gao, "On the hierarchical structure of the logical internet graph," in *in Proc. SPIE ITCOM*, 2001, pp. 208–222.
[11] M. Thorup and U. Zwick, "Compact routing schemes," in *SPAA '01: Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures*. New York, NY, USA: ACM, 2001, pp. 1–10.
[12] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.
[13] S. Sarat, V. Pappas, and A. Terzis, "On the use of anycast in dns," in *SIGMETRICS '05: Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*. New York, NY, USA: ACM, 2005, pp. 394–395.
[14] P. Mockapetris, "Rfc 1034: Domain names: concepts and facilities (november 1987)," *Status: Standard*, 2003.
[15] M. Allman, V. Paxson, and W. Stevens, "Rfc 2581: Tcp congestion control," 1999.

Romanian Academy, Institute of Computer Science, Blvd. King Carol I, Iaşi 700505, Romania
E-mail address: gabriel@info.uaic.ro

Babeş-Bolyai University, Department of Computer Science, 1 M. Kogălniceanu Street, Cluj-Napoca 400084, Romania
E-mail address: dan@cs.ubbcluj.ro