An Alternative Way of Teaching the Advanced Concepts of the Diffusing Update Algorithm for EIGRP

Valentina Trujillo, Jesús Expósito, and Eric Gamess

Abstract — **EIGRP is a routing protocol developed by Cisco Systems as an alternative to traditional distance vector and link state routing protocols. It has been widely accepted since it merges important concepts of these two types of routing protocols in a single one. The Diffusing Update Algorithm (DUAL) is used by EIGRP to compute the shortest path by performing local or diffusing computations. This algorithm was first proposed by Dijkstra and Scholten, and offers a great set of advantages that makes it outperform other loop-free routing algorithms. The teaching and learning of DUAL can be very difficult due to the complex finite state machine involved and the lack of documentation for this algorithm. In this paper, we present an alternative approach to teach advanced concepts of DUAL by using Easy-EIGRP, a didactic implementation of EIGRP developed in Java. Easy-EIGRP includes a powerful module, called DUAL module, which implements the algorithm and offers a set of graphic interfaces for debugging and understanding the processes carried out by DUAL.**

Index Terms — **EIGRP, Easy-EIGRP, DUAL, Cisco Systems, Routing Protocol, Didactic Applications.**

I. Introduction

Routing protocols can be categorized, based on the information they exchange and the way they compute their routing tables, in: distance vector protocols and link state protocols. In a distance vector protocol, each node knows the shortest distance from a neighbor node to every destination network; however it does not know all the nodes between its neighbor and the final destination. This type of protocols sends periodic updates which include every destination entry of its routing table along with the corresponding shortest distance to it. Distance vector protocols use the Bellman-Ford algorithm for shortest path computation. The main disadvantages of this algorithm are routing table loops and counting to infinity. Some of the most important distance vector protocols known are: RIPv1 [10] (Routing Information Protocol v1), RIPv2 [13] (Routing Information Protocol v2), and IGRP [5] (Interior Gateway Routing Protocol).

On the other hand, in link state protocols, every node knows the whole network topology thanks to the update flooding that happens every time a topology change is detected. Based on these updates, each node must calculate the shortest path to a specific destination. Link state protocols use some variant of Dijkstra's algorithm for the shortest path computation, which ensures that the counting to infinity problem is not going to be present; however, an up-to-date version of the entire topology is needed by every node, which may constitute excessive storage and communication overhead in a large, dynamic network [7]. OSPF [2][14] (Open Shortest Path First) and IS-IS [11][15] (Intermediate System to Intermediate System) are some of the most commonly used link state routing protocols.

EIGRP [16] (Enhanced Interior Gateway Routing Protocol) is often categorized as a hybrid protocol since it advertises its routing table to its neighbors as distance vector protocols do, however it uses the hello protocol and creates neighbor relationships, similarly to link state protocols.

In addition, it sends partial updates when a metric or the network topology changes, but it does not send full routing table updates in periodic fashion as distance vector protocols do. EIGRP uses the DUAL (Diffusing Update Algorithm) algorithm, which ensures that there will never be loops, not even temporary, as in the case of distance vector or link state protocols. This algorithm is a new and advanced approach of the distance vector algorithms which includes: loop-free warranty, arbitrary transmissions or delay processing operations, arbitrary positive link assumption and finite time calculations for finding the shortest path towards a destination on a topology change [7]. DUAL uses diffusing computations with the aim of solving the shortest path problem and improving the performance and resource usage of the traditional algorithms.

Easy-EIGRP [6] is a didactic implementation of EIGRP with an integrated graphical viewer for the DUAL finite state machine that allows users to understand, in an easier way, EIGRP's complex processes (including local and diffusing computations). In this paper, we present our alternative way of teaching the advanced concepts of DUAL using Easy-EIGRP.

The rest of the paper is organized as follows: In Section II, the related work is viewed. DUAL is discussed in Section III. Our approach of teaching the DUAL algorithm is presented and justified in Section IV and finally, conclusions and future work are discussed in Section V.

II. Related Work

Currently, there is a wide range of options when it comes to open source projects for TCP/IP based routing protocols. The main goal of these projects is to promote inventive solutions for network routing. Some of the most popular routing software suites are: Zebra [8], Quagga [17] , XORP [9], BIRD [1], Click [12], Vyatta [18], etc.

Manuscript received July 15, 2010. This work was supported in part by the Universidad Central de Venezuela and the CNTI (Centro Nacional de Tecnologías de Información).

V. Trujillo is with the School of Computer Science, Universidad Central de Venezuela, Facultad de Ciencias, Los Chaguaramos, Caracas, Venezuela (e-mail: valentina.trujillo@gmail.com).

J. Expósito is with the DTIC (Dirección de Tecnología de Información y Comunicaciones), Universidad Central de Venezuela, Edificio El Rectorado, Los Chaguaramos, Caracas, Venezuela (e-mail: exposito.j@gmail.com).

E. Gamess is with the Laboratory of Communications and Networks, Universidad Central de Venezuela, Facultad de Ciencias, Caracas, Venezuela (phone: +58-212-6051296; e-mail: egamess@gmail.com).

It is noteworthy that none of the projects mentioned before offer support for EIGRP, furthermore, none of them (not even the Cisco Systems Command Line Interface) provide a graphical view of the DUAL finite state machine which is the main core of the protocol, and its understanding is essential for EIGRP specialists.

For this reason we decided to develop our own didactic EIGRP routing solution which provides a fully graphical environment for the understanding of the protocol. In addition we included a complete and interactive GUI for the DUAL comprehension, stating in that way the innovative quality of Easy-EIGRP.

III. DUAL IN EIGRP

DUAL is a convergence algorithm that replaces the Bellman-Ford algorithm used by other distance vector protocols. DUAL was proposed by E. W. Dijkstra and C. S. Scholten [4]. The main goal of the algorithm is to perform distributed shortest path routing while maintaining the network free from loops at every instant.

The Diffusing Update Algorithm relies on protocols (such as the Hello Protocol and the Reliable Transport Protocol) and data structures (such as the neighbor table and the topology table) to provide consistent information, leading to optimum route selection.

For DUAL to operate correctly, the following conditions must be met [7]:

- Within a finite time, a node detects the existence of a neighbor or the loss of connectivity with a neighbor. This is handled by the Neighbor Discovery/Recovery mechanism implemented by EIGRP.
- All messages transmitted over an operational link are received correctly and in the proper sequence within a finite time. The EIGRP’s RTP (Reliable Transport Protocol) has the responsibility to ensure that this condition is met.
- All messages, changes in the cost of a link, link failures, and new-neighbor notifications are processed one at a time within a finite time and in the order in which they are detected.

EIGRP uses the Hello protocol to discover neighbors and to identify the local router to neighbors; when this happens, EIGRP will attempt to form an adjacency with that neighbor. Once the adjacency is establish, the router will receive updates from its new neighbor which will contain all routes known by the sending router and the metric of those routes. For each neighbor, the router will calculate a distance based on the distance advertised by the neighbor and the cost of the link to it. The lowest distance to a specific destination is called Feasible Distance (FD) [5].

The Feasible Condition (FC) is a condition that is met when a neighbor’s advertised or reported distance (RD) to a destination is strictly lower that the router’s FD to that destination. Any neighbor that meets the FC would be labeled as a Feasible Successor (FS). The FS that provides the lowest distance to a destination would be labeled as a Successor and this would be the next hop that the router would set in order to reach that destination. It is important to mention that there might be more than one Successor and that unequal cost balancing is also allowed by EIGRP.

The FSs and the FC are the elements that ensure that loops will be avoided. Because FSs always have the shortest metric distance to a destination, a router will never choose a path that will lead back through itself (creating a loop), since such path would have a distance larger that the FD, therefore the FC would not be met.

EIGRP works with a topology table, where all the known destinations are recorded. Each destination is registered along with its FD and the corresponding FSs. For each FS, its advertised distance and interface of connectivity will be recorded.

The DUAL's computations can be summarized into two processes (local computations and diffusing computations) as described below.

A. Local Computations

The local computations are carried out by an EIGRP router as long as it can resolve a change in the network topology without querying its neighbors, in other words, if the router can resolve a specific situation locally. For example, if an EIGRP router faces with an increased metric from its Successor and the router has at least one additional FS for the same destination, the action is immediate, the new route is selected and updates are sent to all the neighbors to inform them about the change in the network topology. When performing local computations, the affected route will stay *passive*. It is important to mention that if an EIGRP router can resolve a topology variation with local computations, that does not mean its neighbors are going to have the same opportunity.

B. Diffusing Computations

When an EIGRP router cannot find an alternate route (no alternate route exists, or the new best route still goes through the affected Successor), it starts a diffusing computation by asking all its neighbors about an alternate route. A diffusing computation is performed in a series of steps:

1) The affected route is marked *active* in the topology table.
2) A reply-status table is created to track the replies expected from the neighbors.
3) A query is sent to the neighbors.
4) Responses are collected and stored in the topology table. At the same time, the corresponding entry in the reply-status table is updated.
5) The best response is selected in the topology table and the new best route is installed in the routing table.
6) If necessary, an update is sent to the neighbors to inform them of the changed network topology [16].

Every time a query is sent to a neighbor, an independent timer is started for this neighbor in order to guaranteed network convergence in a reasonable time, which constitutes one of EIGRP principles.

EIGRP ensures one more time that any loop will be avoided thanks to the use of the route’s status flag. If, for example, a router receives a query from a neighbor which is performing a diffusing computation, and the query is about a route that has already being marked as *active*, the router will reply with its current best path and it will stop the query processing, avoiding the creation of a query loop and an upcoming package flooding.

In most cases, after a diffusing computation is complete, the router that initiated the computation must distribute the obtained results.

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

DUAL has a finite state machine (DUAL finite state machine) which controls all the possible states in which a router can be found while performing diffusing computations (if the router is performing local computations, the finite state machine will be executing the IE1 event, leading to the r=0, O=1 state). Because there are multiple types of input events that can cause a route to change its state, some of which might occur while a route is *active*, DUAL defines multiple active states. A *query origin flag* (O) is used to indicate the current state. Figure 1 and Table I show the complete DUAL finite state machine [5][6].

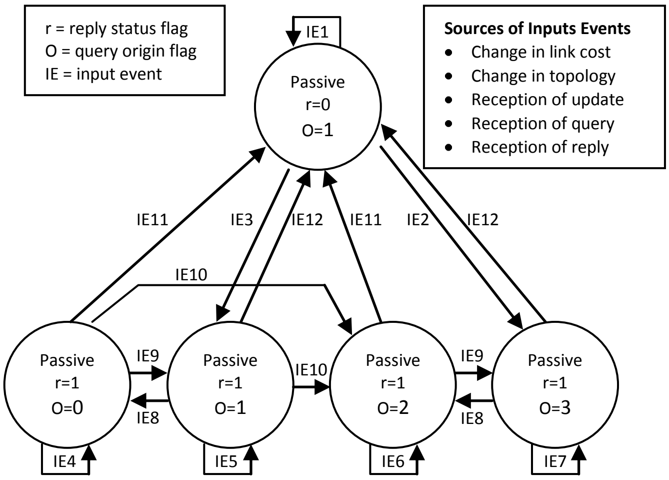

Figure 1: DUAL Finite State Machine

Table I: Input events for the DUAL Finite State Machine

Input Event	Description
IE1	Any input event from which FC is satisfied or the destination is unreachable
IE2	Query received from the Successor; FC not satisfied
IE3	Input event other than a query from the Successor; FC not satisfied
IE4	Input event other than a last reply or a query from the Successor
IE5	Input event other than a last reply, a query from the Successor, or an increase in the distance to destination
IE6	Input event other than a last reply
IE7	Input event other than a last reply or an increase in the distance to destination
IE8	Increase in the distance to destination
IE9	Last reply received; FC not met with current FD
IE10	Query received from Successor
IE11	Last reply received; FC met with current FD
IE12	Last reply received; set FD to infinity

As discussed before, diffusing computation requires that a router receives the replies from all the neighbors it queried in order to select the new best route. However, there are extreme circumstances in which a neighbor might fail to respond a query. Any of those circumstances blocks the router originating the diffusing computation. To prevent these types of deadlock situations, EIGRP contains a built-in safety measure: a maximum amount of time a diffusing computation can take to execute. Whenever a diffusing computation takes longer than the timeout value, the diffusing computation is aborted; the adjacency with any non-responding neighbors is cleared, and the computation proceeds as though these neighbors replied with an infinite metric. The route for which the computation is aborted is said to be SIA (Stuck In Active).

It is clear that the processes describe before are not easy to understand just basing ourselves on the theory, it is necessary a more visual and didactic approach that allows users (students with only basic knowledge of networking on most cases) to understand every single detail of this algorithm. That is the reason why we developed a graphical implementation of the DUAL algorithm which can be found in the DUAL Finite State Machine module of Easy-EIGRP. The main purpose of this module is to provide an intuitive and interactive tool for teaching and learning how EIGRP really works.

IV. DUAL'S IMPLEMENTATION IN EASY-EIGRP

Easy-EIGRP [6] is a didactic application which main goal is the teaching and learning of EIGRP. Easy-EIGRP provides five modules with the aim of allowing users an improved, efficient and easy way to understand the protocol; these modules are: (1) the EIGRP Manager, (2) the DUAL Finite State Machine module, (3) the Partial Network Map Viewer, (4) the EIGRP Tables Viewer and, (5) the Logger module. In this paper we will focus on the DUAL Finite State Machine module, although it is worth to mention that the Logger module plays an important role when it comes to debugging (refer to [6] for further details).

The DUAL implementation is based in a set of Java classes (*DUAL*, *NeighborDiscovery*, *RTP*, etc) which are represented in Figure 2, along with their corresponding relationships. For reasons of space, Figure 2 only shows the most important classes relative to our DUAL's implementation.

The *NeighborDiscovery* class is responsible for discovering neighbors and setting new adjacencies. It also manages EIGRP table's inputs (represented by the *RoutingTable*, the *TopologyTable* and the *NeighborTable* classes) every time a topology change is detected based on the loss (which will be announced by the *HoldTimerRTOThread*) or discovery of a neighbor.

The diagram also shows that the *NeighborTable*, the *RoutingTable* and the *TopologyTable* are composed of *Neighbors*, *Routes* and *Destination* objects respectively. In turn, *Destination* objects are formed of a set of *FeasibleSuccessors* and *Distances*.

The *RTP* (Realiable Transport Protocol) class is the one that will manage EIGRP's packet exchange, guaranteeing the delivery and the ordering of these packets. This class relies on the *ListenerThread*'s operation, because this thread is the entity that will monitor the PC's corresponding interfaces, passing the received packets to the RTP layer, where these packets will be processed.

Finally, the *DUAL* class embodies the decision process for all route computation by tracking all routes advertised by all neighbors. Whenever a local or a diffusing computation

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

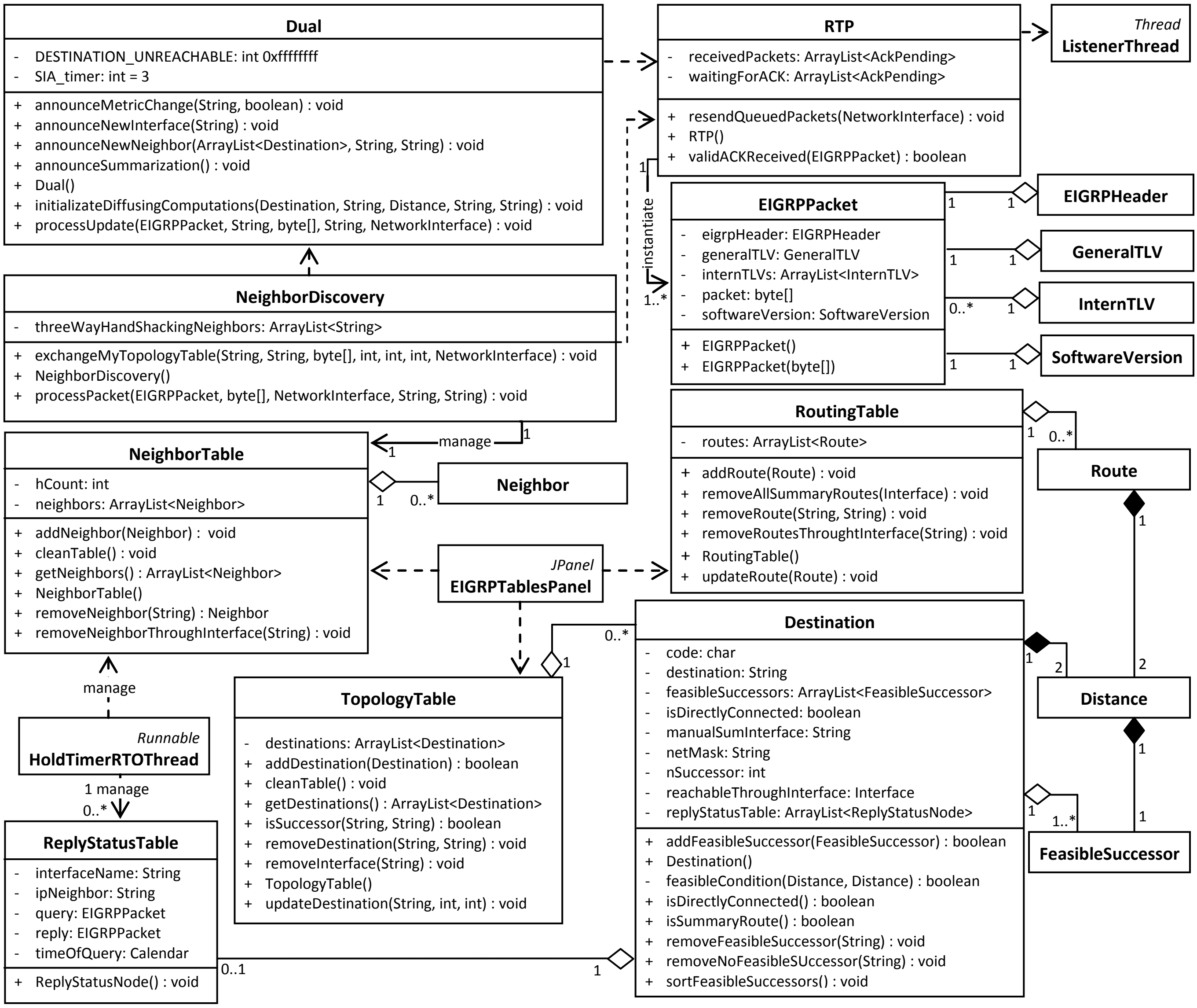

Figure 2: Easy-EIGRP's DUAL Class Diagram

happens, *DUAL* will notify the pertinent changes to the *EIGRPTablesPanel* that eventually will update the EIGRP's tables. If a diffusing computation occurs, *DUAL* will indicate to change the state of a *Destination* on the *TopologyTable* and to assemble its corresponding *replyStatusTable*. Additionally, the *HoldTimerRTOThread* will keep track of the SIA timers in order to prevent possible deadlock situations discussed in Section III.

The interaction of the *DUAL* class can be represented as shown in Figure 3 (took from [16]). Its operation was designed following some basic rules defined in [16]:

1) Whenever a router chooses a new Successor, it informs all its other neighbors about the new RD (Reported Distance).
2) Every time a router selects a Successor, it sends a poison update to its Successor.
3) A poison update is sent to all neighbors on the interface through which the Successor is reachable unless split-horizon is turned off, in which case, it is sent to only the Successor.

Easy-EIGRP also implements a set of methods which are capable of determining and handling both local and diffusing computations which are discussed next.

It is important to mention that, since Easy-EIGRP was developed using reverse engineering, all its processes, operations and rules are the same defined by EIGRP. Easy-EIGRP intends to emulate the EIGRP protocol.

When Easy-EIGRP is not performing diffusing computations, each route is in the *passive* state, as stated by the DUAL's rules of EIGRP.

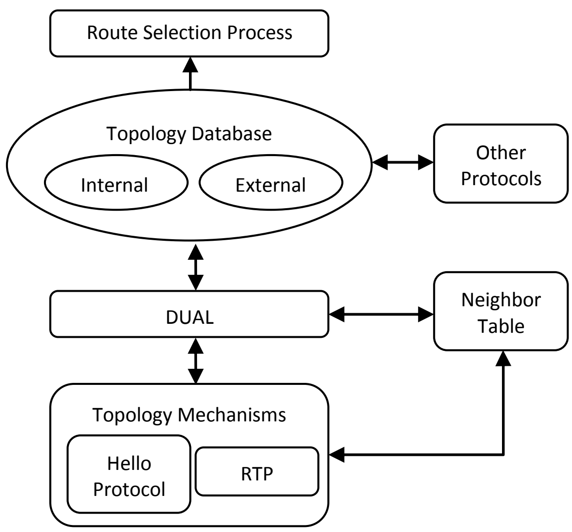

Figure 3: DUAL's Module Interaction

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

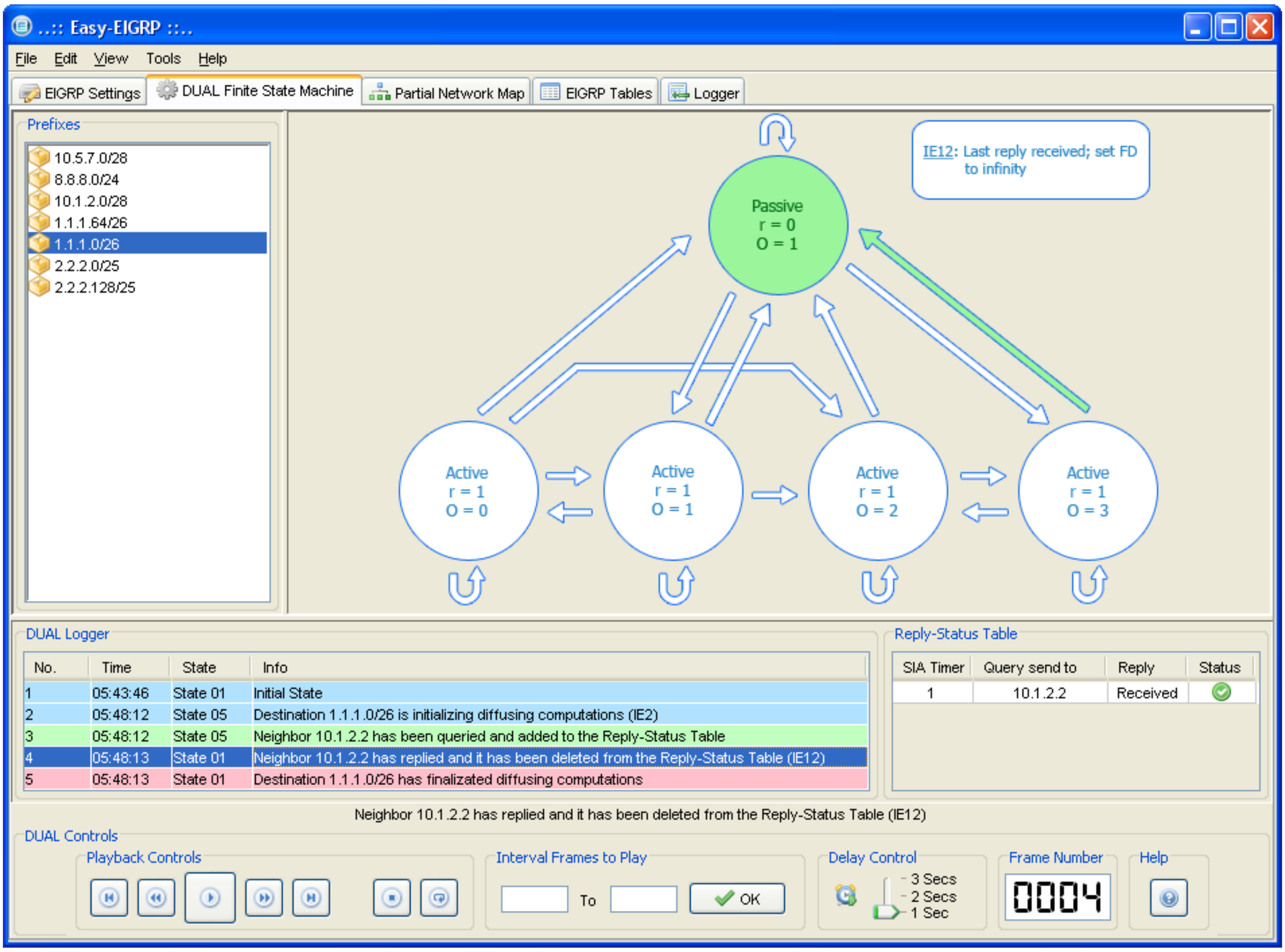

Figure 4: Easy-EIGRP's DUAL Finite State Machine Viewer

Easy-EIGRP will reassess its list of FSs for a route, when a change in the cost or state of a directly connected link is detected or due to the reception of an update, a query or a reply packet.

In case Easy-EIGRP's DUAL module is not able to find a possible Successor (different from the current one) for the affected destination, the application will begin a diffusing computation and it will change the route to the *active* state.

Next, Easy-EIGRP will perform every single step stated in Section III.B.

Easy-EIGRP begins a diffusing computation by sending queries to all its neighbors. Simultaneously, the application will create a data structure which represents EIGRP's reply-status table; this table will contain an input (including the neighbor's IP address, a time stamp for the query sent, a time stamp for the reply received, etc.) for each neighbor queried.

Since every step processed on a diffusing computation corresponds to an input event in the DUAL finite state machine (see Figure 1) and since there is no didactic tool that represents or explain this process, we developed a graphical view that allows users to witness the whole computation (see Figure 4).

Easy-EIGRP's DUAL finite state machine module is composed of 5 sections:

1) The prefix list panel, located on the left upper corner, is responsible for listing and maintaining record of all the prefixes that were handled by the application at some point. It is noteworthy that even if the prefix is lost, this panel will maintain the prefix record for playback.
2) The DUAL finite state machine panel, located on the right upper corner, allows users to view an image of the whole finite state machine used by EIGRP, which can be animated at any time the user decides it.
3) In the middle left, users can find a logger which will provide information about every change registered by the finite state machine. Every time users select an event on the logger, the image of the finite state machine will be updated, showing users the current state of the machine at that point. In some cases, the finite state machine panel will show additional information on its right upper corner about the selected event, generating a correspondence between the event described by the logger and the events establish on Table I. It is important to mention that each prefix owns an independent logger that, no matter what happens, will always keep past events or computations.
4) The reply-status table, located in the middle right is independent for each diffusing computation process and is unique for each prefix. The table is composed of 4 fields: SIA timer, IP address of the queried neighbor, reply status and a field that represents the reply status graphically. The last field can show 3 types of images: a clock (indicating the router is waiting for a reply), a green check mark (indicating the router already received the expected reply) or a red *X* (indicating that the expected reply never arrived and that the SIA timer limit has been reached). Each reply-status table's row has a directly

correspondence with an entry of the *replyStatusTable* (more specifically with a *ReplyStatusNode*) of a *Destination* object (see Figure 2).

5) Finally, there is a media reproduction control panel in the bottom of the module. The main goal of this panel is to provide full flexibility of reproduction (forward, backward, pause, etc.) of past or current diffusing computations. Easy-EIGRP allows users to modify the delay of the reproducing events in order to allow a detailed analysis. This panel also let users specify a range of events on the logger for later reproduction.

It is a fact that it is easier to understand abstract topics, such as the diffusing computation processes and general networking issues, with the help of images, colors and animations, that using traditional command line interfaces or shells. That is the reason why Easy-EIGRP implements these types of tools which provide a more natural way of observing and analyzing the behavior and the decision process of DUAL. Every single section of the DUAL module was designed and developed to improve and facilitate a detailed study of the algorithm, providing an excellent, interactive and friendly interface.

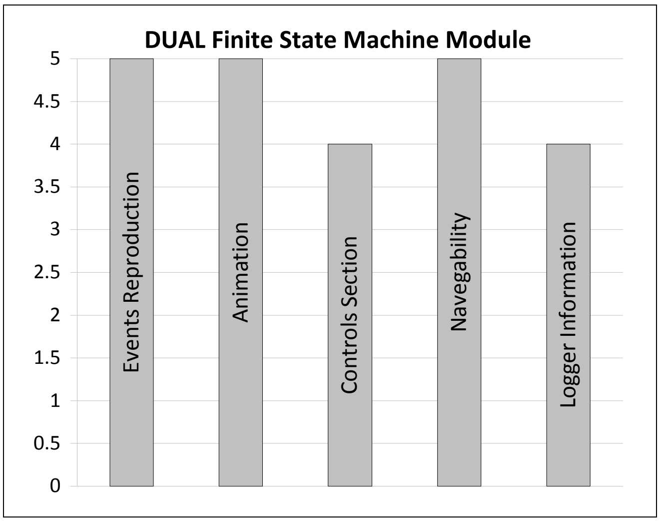

Figure 5: Survey Results

To prove that the benefits of this graphical view are significant, we conducted a set of experimental laboratories where we grouped around 50 computer science students with different levels of networking knowledge. We created two types of laboratories. In the first laboratory, a computer with several network interface cards was connected to Cisco 2811 routers through FastEthernet connections.

In the computer, students installed Easy-EIGRP and configured EIGRP in all the devices to see how routing information was propagated and to understand DUAL. The idea behind the second laboratory is to do a realistic EIGRP practice in a single computer. For that, we did a virtual appliance for Easy-EIGRP, and students created some virtual machines (by cloning the virtual appliance), connected them in a network, configured Easy-EIGRP in these virtual machines, and also studied how routing information was propagated and handled by DUAL.

After the experiment ended, a survey was filled by them. The results are shown in Figure 5, where we can see the high acceptance level of Easy-EIGRP's DUAL module, particularly about characteristics such a impact of the events reproduction, animations features, utility of the controls section, navigability, etc.

V. Conclusions and Future Work

Easy-EIGRP is a limited implementation of the EIGRP protocol which can be used on both Windows and Linux for teaching and learning purposes. This application includes a powerful module which implements DUAL and offers a set of graphical interfaces for debugging and understanding the processes carried out by this complex algorithm.

We had been using Easy-EIGRP to support the teaching and learning of DUAL with a group of networking students. The feedback that we received from these students was very positive. Most of them could easily understand EIGRP's operation; in addition, they claimed that the DUAL Finite State Machine panel provided them a very positive support in order to analyze how the Diffusing Update Algorithm works.

For future work, we plan to further develop Easy-EIGRP to support IPv6 (Internet Protocol version 6) [3], since IPv6 will become the predominant layer-3 protocols in tomorrow's networks.

Since our main goal is to offer Cisco instructors an excellent didactic application for the teaching of EIGRP, we already contacted Cisco Systems for a possible distribution of Easy-EIGRP, and now we are looking forward for an answer.

References

[1] The BIRD Internet Routing Daemon. http://bird.network.cz.
[2] R. Coltun, D. Ferguson, and J. Moy. OSPF for IPv6. RFC 2740. December, 1999.
[3] J. Davies. Understanding IPv6. Microsoft Press. Second Edition. January, 2008.
[4] E.W. Dijkstra and C.S. Scholten. Termination Detection for Diffusing Computations. Information Processing Letters, Vol. 11, No 1. August, 1980.
[5] J. Doyle and J. Carrol. Routing TCP/IP. Volume I. Cisco Press, Second Edition. October, 2005.
[6] J. Expósito, V. Trujillo, and E. Gamess. Easy-EIGRP: A Didactic Application for Teaching and Learning of the Enhanced Interior Gateway Routing Protocol. The Sixth International Conference on Networking and Services (ICNS 2010). Cancun, Mexico. March, 2010.
[7] J. García-Lunes-Aceves. Loop-Free Routing Using Diffusing Computations. IEEE/ACM Transactions on Networking, Vol. 1, No. 1. February, 1993.
[8] GNU Zebra. http://www.zebra.org.
[9] M. Handley, O. Hodson, and E. Kohler. XORP: An Open Platform for Network Research. ACM SIGCOMM Computer Communication Review. October, 2002.
[10] C. Hedrick. Routing Information Protocol. STD 34, RFC 1058. June, 1988.
[11] ISO. Intermediate System to Intermediate System Intra-Domain Routing Exchange Protocol for use in Conjunction with the Protocol for Providing the Connectionless-Mode Network Service. ISO 8473, International Standard 10589:2002, Second Edition.
[12] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. Frans Kaashoek. The Click Modular Router. Laboratory for Computer Science, MIT.
[13] G. Malkin. RIP Version 2. RFC 2453. November, 1998.
[14] J. Moy. The OSPF Specification. RFC 1131. October, 1989.
[15] D. Oran. OSI IS-IS Intra-domain Routing Protocol. RFC 1142. February, 1990.
[16] I. Pepelnjk. EIGRP Network Design Solutions: The Definitive Resource for EIGRP Design, Deployment, and Operation. Cisco Press. January, 2000.
[17] Quagga Routing Suite. http://www.quagga.net.
[18] Vyatta. http://www.vyatta.org.

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)