

Intel[®] Technology Journal

Wireless Technologies

**Dynamic Wired and Wireless
Networks on Demand**

Dynamic Wired and Wireless Networks on Demand

Amber Sistla, Network Architecture Labs, Intel Corporation
Jeremy Rover, Network Architecture Labs, Intel Corporation
Asha Keddy, Network Architecture Labs, Intel Corporation

Index words: dynamic network configuration, network architecture, pseudo-random scenario generation, automation, testing, wireless, validation, roaming

ABSTRACT

The complexity of use scenarios, user device types, and heterogeneity of networks poses an increasing validation and network management challenge. Current networks combine a wide variety of technologies, protocols, platforms, and device types which, when combined with usage models, leads to a combinational explosion. This paper describes the Dynamic Networks on Demand (DND) framework that automates network creation for wired and wireless network topologies and executes resource transitions (e.g., moving a laptop between subnets) all without ever manually plugging or unplugging a wire or configuring a network device. The uses include automated management of local and remote networks, automated demos for promotion of wireless technology, and automated validation of roaming concepts. These combined pioneer mechanisms enable the DND framework to step up to the challenge of provisioning increasingly varied and dynamic networks.

The paper also describes details of the DND framework. The DND framework is used to create hundreds of networks from limited resources and to execute thousands of pseudo-random test scenarios in each network. Using simple human-readable network requests, the DND framework creates a new network through a series of synchronized steps to reconfigure all portions of the new network: Dynamic Host Control Protocol (DHCP) servers, routers, and logical connections. Resource transitions occur by using the same human-readable network requests. The framework also provides error reporting and logging capabilities. The DND framework uses a single control point so the network changes can be controlled remotely. The framework allows for remote configuration of hardware resources as fast as the hardware can handle. The DND framework is designed to be modular and extensible, and can be used with varied devices from different vendors with minimal changes. The paper also highlights the exponential cost reduction

due to the reduction of manual intervention (the number of tests as well as the network reconfigurations would not be feasible if done manually) and to the reuse of limited resources, achieved through intelligent and innovative automation.

INTRODUCTION

The introduction of mobile technologies such as Intel® Centrino™ mobile technology fuels network evolution at unprecedented rates, thereby fostering new usage models. The demanding consumers of today need to stay connected anywhere, anytime with adequate security models and ease of use. These demands in turn act as catalysts to combine different infrastructures, topologies, technologies, protocols, and network components. Network components in turn include routers, Dynamic Host Control Protocol (DHCP) servers, virtual private networks (VPN), access points (APs), and user stations (desktops, laptops) connecting over wired and wireless (802.11, Global System for Mobile Communications) protocols with services such as single bill roaming through different wireless hotspots.

In addition, each network component requires different configurations depending on the needs of a particular network. User stations and wireless hotspots add more complexity with variable platforms (e.g., Intel Centrino mobile technology), operating systems, network adapters, manufacturers, etc. Network components can also communicate over different protocols (IPv4, IPv6, Mobile IPv4). The wide variety of components and configurations causes a combinational explosion, with heterogeneous networks of many different types being deployed and used.

Intel Centrino is a trademark of Intel Corporation or its subsidiaries in the United States and other countries.

Different combinations of configurations are possible for initial network setup, and the network must continually adapt as changing needs require the addition or removal of network components in any portion of the network. In addition, networks are also expected to be self-aware and resilient as they adjust to dynamically changing resources on the network.

The manual configuration and reconfiguration to address the many network needs can cause significant delays in network turnaround time and network downtime. In-depth and specific knowledge about each network technology and component is also necessary in order to deploy and manage such networks.

An ever-increasing network management and validation challenge results from the dynamic, heterogeneous, and knowledge-intensive nature of real-world networks.

The Switched Roaming¹ technology (also known as “adapter switching”), developed as a part of the Intel Centrino mobile technology effort, required a validation and debugging environment to tackle the dynamic nature of the technology being developed. Performance, mobility, reliability, and the quality of service the user experiences need to be validated.

The Dynamic Networks on Demand (DND) architecture addresses these challenges by streamlining the network setup process to reduce network turnaround time and by making the technology accessible to both experts and non-experts, allowing real-world networks to quickly be created “on demand.”

DYNAMIC NETWORKS ON DEMAND FRAMEWORK OVERVIEW

The Dynamic Networks on Demand (DND) framework is responsible for three unique functions:

1. To create dynamic networks and topologies for known, predefined configurations.
2. To reconfigure networks based on the possible device transitions.
3. To execute network scenarios, perform verification, and report the outcomes from a single control point.

A network scenario for the framework can be defined as these three functions in sequence. The framework creates

¹ Switched Roaming technology: The process of selecting one network interface in the presence of multiple interfaces based on user-defined preferences. This technology is used with today’s network cards to move through network environments without restarting applications.

a network, transitions through various network states, validates the network throughout various stages of transition, then completes the scenario.

To address the three areas of functionality, the DND framework is abstracted into the following four distinct layers: the Control Layer, the Network Management Layer, the Verification and Validation Layer, and the Physical Network Layer (Figure 1). The expandable nature of the architecture allows the addition of components and/or layers at additional levels of the architecture to increase the breadth of functionality.

Control Layer

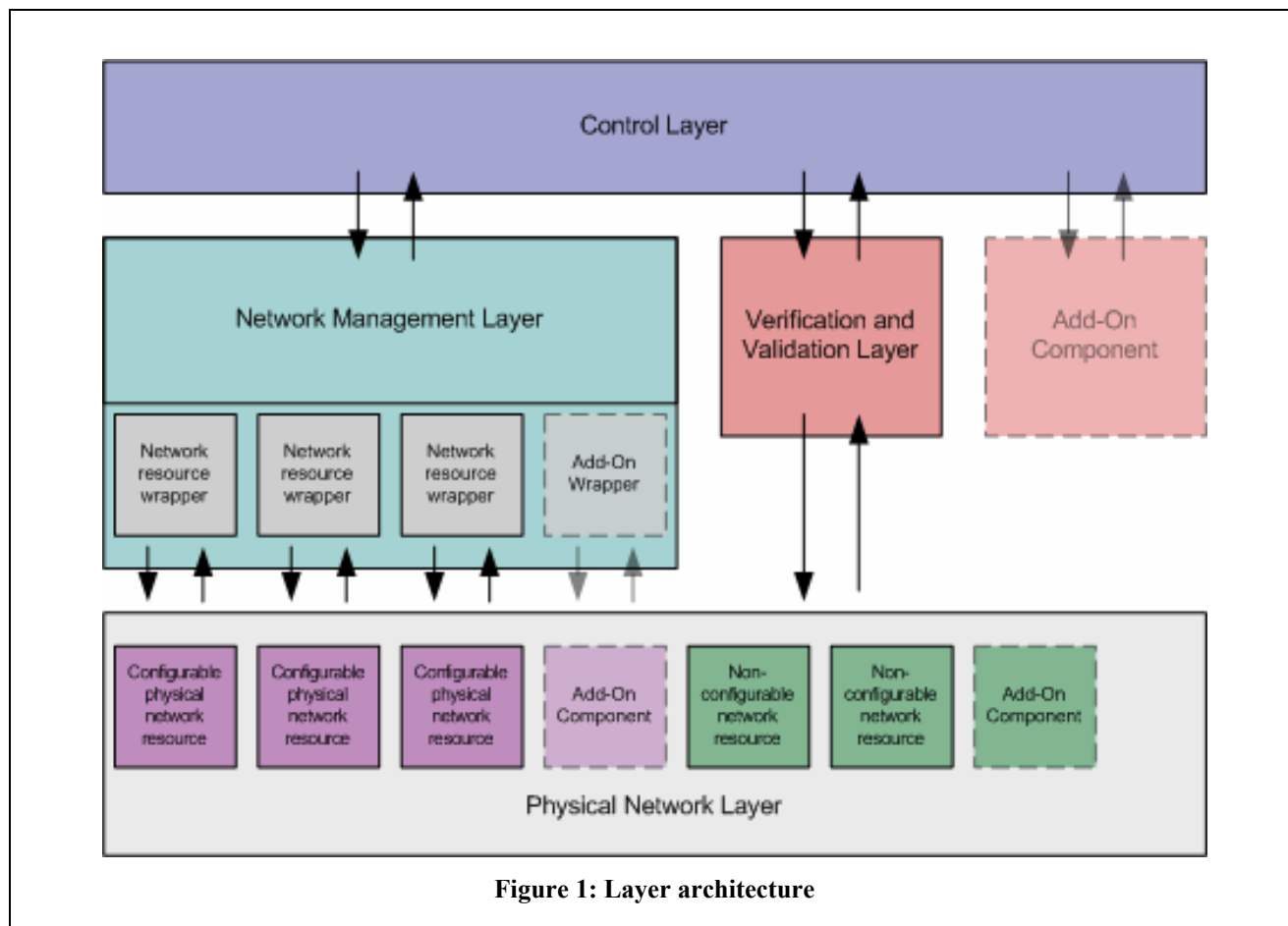
The Control Layer abstracts the inner workings of the framework and allows users to run generated network configurations. Using network scenarios generated by the Control Layer, network creations and transitions can become a secondary concern to the user.

Users can generate scenarios either randomly or based on predefined network configurations. This layer manages these configurations as well as subsets and expansions of network configurations. Randomization features are available in the context of network scenario creation. Each series of network scenarios can be reproduced by supplying a seed logged in past scenarios.

Based on interactions with the Network Management Layer and the Verification and Validation Layer, the Control Layer determines the current physical layout and state of the network. It accesses the physical network configuration through the Network Management Layer to perform network scenarios and transitions on the physical devices of the network. When the physical devices are added or removed from the pool of available network devices, the available device configuration file generally requires only one update to dynamically propagate changes through the entire framework at run-time.

When interactions with the other layers indicate network failures (e.g., device fails to obtain Internet Protocol (IP) address, network is saturated), the DND framework provides mechanisms for performing graceful state recovery by detecting errors from its interactions with other layers to log and restore proper network functionality.

The Control Layer exists as the single control point for the framework. In addition to synchronizing the actions of the layers, another advantage of a single control point is remote management. The functionality of the framework can be accessed from one point through a console directly on a machine in close physical proximity to the network or via a remote login session (e.g., telnet).



Network Management Layer

The Network Management Layer is responsible for all network configuration changes and for maintaining the current physical network state information. This layer was built with reuse and extensibility in mind. All other layers extract network state information from the Network Management Layer.

The Network Management Layer is also a stand-alone component for managing and reconfiguring network devices. This layer can directly create networks and perform transitions when debugging or when networking validation requires manual interaction with devices. In addition, network creation initializes the network state of all devices.

Dynamic network creation does more than just lay out network devices into network topologies. The framework also manages the IP address allocations for all hardware devices on the network and generates a readable text file that reports each IP address to facilitate communication across the network. After every

network creation or transition, the Network Management Layer generates an accurate snapshot of the current network state.

The current network state snapshot file also enables the transitioning of devices through a network. In addition to containing the dynamic IP addresses information, the file also includes a list of possible network transitions that a device can perform based on the device capabilities and the configured network infrastructure (e.g., if a device contains an 802.11a network adapter and an 802.11a access point (AP) exists on the network, the device would be able to make a wireless transition to 802.11a). After a transition, generated transition lists are updated for all devices in the network since some network transitions affect multiple mobile devices.

The framework can also be extended by writing wrappers for any additional required configurable devices such as Dynamic Host Control Protocol (DHCP) servers, routers, etc. Adding non-configurable devices such as hubs, laptops, etc., requires minimal effort. By keeping track of all available components in the network

as well as the current state of the network, this layer provides flexibility to adapt to changing network needs and resources.

Verification and Validation Layer

The Verification and Validation Layer abstracts all devices used to validate the current network configuration. These devices include packet sniffers, traffic generators, and other network validation devices. Third-party verification tools can be added to this layer to provide seamless accessibility to complex network analysis and traffic generation tools like Chariot*, IXIA* and Shomiti*.

A significant feature of the DND framework is the ability to make the validation devices mobile. Network scenario creation puts expensive network validation tools on the particular subnet that requires validation. In traditional networks, dynamically allocating validation devices means running scenarios over and over while manually moving the validation devices between subnets of the network to capture pertinent details of the network operation. The DND framework makes moving hardware just as easy as requesting the hardware at network creation time.

This layer extracts device information such as the dynamically changing IP addresses from the Network Management Layer. The Verification and Validation Layer does not retain network-specific information throughout scenarios.

This layer reports all errors and logs to the Control Layer. At scenario completion, Verification and Validation errors and logs enable the Control Layer to make judgments about whether to continue, to generate a new scenario, and/or to recover from error conditions. In this way, the detection of network errors triggers graceful network state recovery in the Control Layer.

Physical Network Layer

The Physical Network Layer contains all the physical network devices. Adding physical network devices to this layer requires collaboration with the Network Management Layer.

Configuring traditional networks requires in-depth knowledge of each network component's configuration. The DND framework, on the other hand, abstracts all network devices to a resource name. If certain network device functionality is required for a network scenario,

the Control Layer requests that functionality in a network request.

The ability to move network components anywhere on a network is useful for reactive network devices such as clients and for connectivity devices such as APs. However, there is often a need to reconfigure components based on their position and the particular network scenario being executed. The DND framework addresses this necessity with network resource wrappers in the Network Management Layer.

A network resource wrapper is required to configure network components. Objects such as routers, Virtual Local Area Network (VLAN) switches, and DHCP servers require reconfiguration and therefore have network resource wrappers located in the Network Management Layer. Each network resource wrapper maps directly to a configurable physical network resource. When new configurable physical network resources are added, accompanying network resource wrappers are written to abstract the functionality for the Network Management Layer.

Non-configurable network resources can be added and removed from the network at will, as no additional framework support is required to interact with them.

Layer Interactions

The interaction of the layers is shown in [Figure 2](#).

A user activates the Control Layer (not shown). The network scenario then starts with the Control Layer querying the Network Management Layer's network state to determine if the scenario can be performed given the current network configuration.

If the network scenario is supported in the current network state, the sequence skips to the Verification and Validation Layer (described in the following paragraph). If the network scenario is not supported in the current network state, the Control Layer resolves the network scenario into a network configuration and creates a simple, readable text file containing a network request. The network request file contains one or more subnets as well as the starting position on the network for mobile nodes². Using the network request from the Control Layer, the Network Management Layer creates a new network and reports success or failure back to the Control Layer.

Barring failures, the Control Layer triggers the Verification and Validation Layer, which performs any

* Other brands and names are the property of their respective owners.

² Computing devices with one or more network connections.

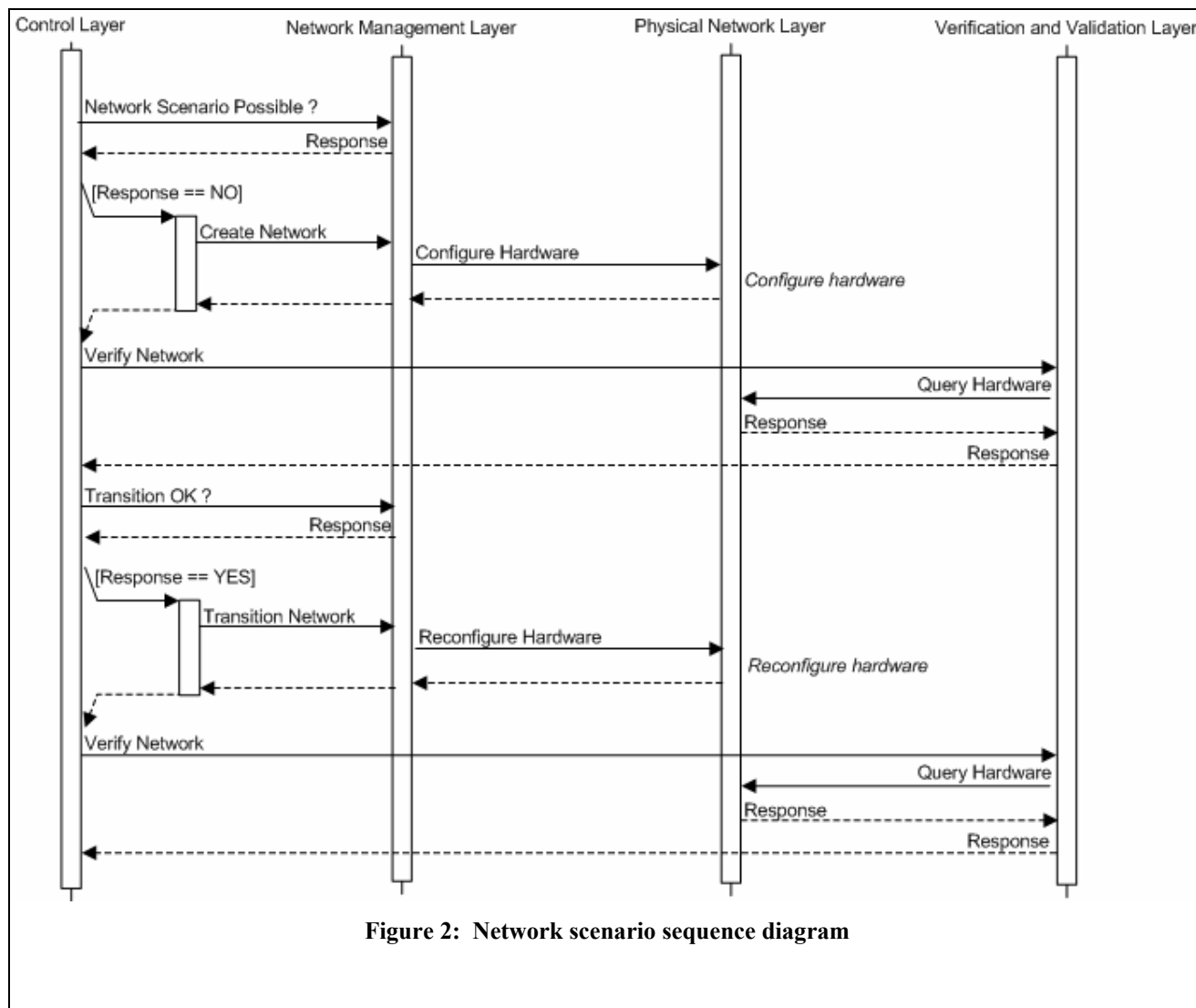


Figure 2: Network scenario sequence diagram

network verification and/or validation tests and reports the findings back to the Control Layer.

If required, the Control Layer queries the Network Management Layer about whether a network transition of a particular network device is supported by the current network configuration. The Control Layer then requests the second half of the network scenario.

For an unsuccessful transition, the Control Layer reports the error and progresses to the end of the sequence as described in [Figure 2](#). For a successful transition, the

REFERENCE IMPLEMENTATION FOR VALIDATION

Mobile technologies introduce new challenges in validation environments, and the Dynamic Networks on Demand (DND) framework addresses some of these problems.

Control Layer again prompts the Verification and Validation Layer to perform network verification and/or validation and report the findings back to the Control Layer.

At this point the sequence can be iterated, enabling multiple network scenarios to occur back-to-back. The sequence can terminate and report findings for the just-completed network scenario, complete with a pseudo-random seed to reproduce the scenario if required.

The implementation of the DND framework addresses validating the mobility features of the Intel Centrino mobile technology platform. The platform ships with multiple network interfaces and exposes users to problems such as deciding which network connection to use as the primary connection. The Switched Roaming technology developed in conjunction with the Intel

Centrino mobile technology effort addresses this problem. The Switched Roaming technology selects which network connection to use when multiple connections are present. The DND framework creates network scenarios where multiple network connections are present and transitions a client throughout a network.

The Switched Roaming technology component includes virtual private networks (VPN) auto-launching functionality. When a client obtains a network connection with a network that requires a VPN, the component automatically launches this software. The DND framework creates network scenarios with VPNs

- *Determine whether the network can be created.* The Network Management Layer must determine the feasibility of a network scenario before it is created. Only valid scenarios will result in a configured network.
- *VLAN switch reconfiguration.* Configuring the VLAN switch (Figure 3, #1) requires a network resource wrapper. The VLAN switch allows the framework to group hubs together programmatically into logical subnets. Also, ports on the switch that contain router interfaces, DHCP interfaces, and other network devices must be placed in the same VLAN to

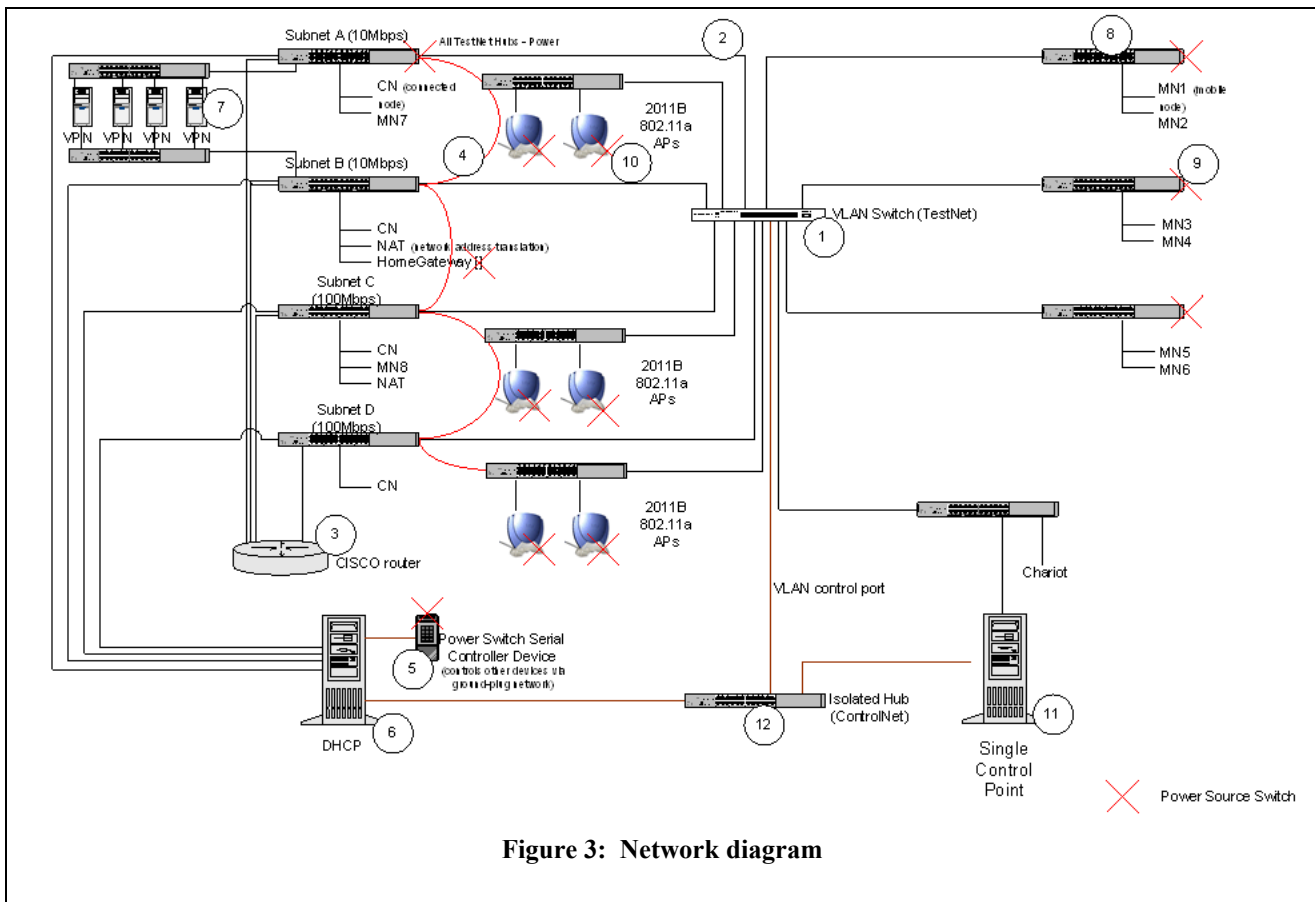


Figure 3: Network diagram

to validate functionality.

The implementation performs multiple functions. Listed below are some of these functions, the corresponding architectural layer responsible for performing the function, and references to the implementation in Figure 3.

1. **Creating networks:** The Network Management Layer is responsible for creating networks. To create a network the following actions must occur:

create an addressable, routable, and valid subnet.

In the example, the router interface, DHCP interface and subnet interface are all located on one hub and plugged directly into the VLAN (Figure 3, #2). The VLAN switch then creates VLANs to join subnets on the left side of Figure 3 to one or more hub(s) on the right side of Figure 3.

- *Create routable subnets.* A router requires a network resource wrapper for configuration (Figure 3, #3). Since the DHCP server communicates the default gateway of each network subnet, the router must have its routing interfaces configured to correspond to the IP addresses that the DHCP server leases to clients. The Network Management Layer creates the associations and configures the routable subnets.
 - *Force clients to renew their IP addresses to realize the new network.* The Network Management Layer uses a programmable power source switch to enable this functionality. The DHCP clients on the network nodes respond to their network link being up or down. When a user unplugs the network cable and then plugs it into another network, the DHCP client realizes the change and acquires a new IP address. To force this same response from the DHCP client, the same message to the DHCP client can be produced by powering off a network node's hub, making the client assume it is unplugged from the network. A power source switch located on the hubs of all DHCP clients creates this response (Figure 3, #4). The Network Management Layer controls the power state of these devices via a serial connection to the power source switch controller device (Figure 3, #5)
 - *Obtain a valid and predictable IP address.* The DHCP server's network resource wrapper allows IP addresses to be specified for each client on the network in every subnet configuration. The Network Management Layer maintains the IP address list and displays only the current IP(s) of the client in any configuration of the network (Figure 3, #6).
 - *Support VPN network configurations.* The main challenge of supporting VPNs is that a number of VPNs require static IP address configurations. Special consideration is given to VPNs in the creation of a network due to their static configuration. If a network requires VPNs, the Network Management Layer creates the specific subnets required for communicating with the statically configured VPN. This is accomplished by configuring the predictable IP addresses on the DHCP server with the required subnet IP addresses that correspond to the specific VPN. It also requires that the router be configured to isolate network traffic on either side of the VPN so that only VPN traffic is routed (Figure 3, #7).
 - *Add/remove network resource.* Adding new devices to test in the scenario and/or removing them throughout the lifetime of a project require the ability to quickly update the current availability of resources. The DND framework allows devices to be easily added and removed from hubs (Figure 3, #8). The Network Management Layer queries the hardware list at run-time in order to use the currently available hardware.
2. **Moving network resources after creation (Roam):** The Network Management Layer is responsible for moving resources after a network is created, which creates its own challenges to overcome. Among these challenges are the following:
- *Change subnets.* Changing which subnet a network resource is located on requires VLAN changes. Changing the port associations of the VLAN allows the network resource to be moved to other subnets.
 - *Obtain a new valid and predictable IP address.* After the VLAN is changed, the network resource will still have its old IP address from the previous subnet. In order to change the IP address, the DHCP client must realize the network connection change. A power-source switch located on the hubs of all DHCP clients creates this response (Figure 3, #9). The Network Management Layer controls the power state of these devices via a serial connection to the power source switch controller device (Figure 3, #5). After the power is restored, the DHCP client picks up the new IP address to complete the move.
 - *Change network topologies (Wired/Wireless) and allow overlapping network interfaces.* Changing which network topology to which network resources are connected requires additional power source device control. For example, when changing from a wired to a wireless network, the boot-time of APs must be considered (Figure 3, #10). The Network Management Layer waits for the AP to finish booting before it powers down the hub to the network resource where it was previously connected. This powering down of the hub forces the network resource to connect to the AP. The time when two network interfaces are available (after booting of the AP but before

powering down the hub) allows a handoff between the two connections to occur.

3. **Run network scenarios.** The Control Layer is responsible for running network scenarios. It combines multiple actions of the Network Management Layer and the Verification and Validation Layer.
 - *Randomly produce and reproduce network scenarios.* The Control Layer or the user can specify a seed to generate a pseudo-random series of network scenarios. The seed from an earlier scenario can be used to recreate that series. The log from a series logs a seed at the start of each network scenario. Any of the logged seeds can be used to reproduce the sequence from that point forward in the log.
 - *Obtain results from multiple layers and combine into one report.* From the Single Control Point (Figure 3, #11), the Control Layer drives all other framework layers and obtains their results to include in the final report.
 - *Control all devices from one location.* The Control Layer communicates with the Validation and Verification Layer and the Network Management Layer (Figure 3, #12) via the multiple interfaces in Single Control Point.
 - *Coverage Analysis.* The Single Control Point reports coverage achieved at scenario completion.

Example Network Scenario Sequence

The following network scenario sequence illustrates one practical instance of the DND framework. To run a network scenario that transitions a mobile client from a wired network to a wireless network on a different subnet, the user would choose from a list of scenarios in the Control Layer (Figure 3, #11). Assuming this network must be created, the Control Layer resolves the particular case into a simple human-readable network request.

Based on the request, the Network Management Layer (Figure 3, #6) appropriately organizes the Physical Network Layer components into the configuration. The physical configuration contains mobile node laptops connected to a wired local area network (LAN). The Validation and Verification Layer confirms that the mobile nodes are reachable.

The Control Layer analyzes the results from the Network Management Layer to determine if the

transition (wired subnet to wireless subnet) is possible. The Control Layer sends commands to the Network Management Layer to transition from wired to wireless. The first command powers on an access point (Figure 3, #10). After the access point has initialized, the hub that joins the mobile nodes to the VLAN is powered down (Figure 3, #9). The Validation and Verification Layer runs validation tests to verify that the mobile nodes are reachable on the wireless subnet. The Control Layer (Figure 3, #11) determines the success or failure of the network scenario and completes the sequence.

Practical Implications

The DND framework implementation discovered memory leak, continuous operation, firmware, roaming, and IP address defects that would have been hard or impossible to discover using traditional manual testing methods. Additionally, the DND framework implementation allowed reproducible test scenarios to aid debugging efforts in several cases, especially where manual testing was unable to reproduce similar scenarios.

The DND framework implementation also increased test coverage. For example, the DND framework implementation ran for 160 hours of continuous operation testing (non-stop) with 14,869 transitions occurring over hundreds of network configurations (approximately five network configurations every 70 minutes) across eight mobile nodes. This averages approximately 1858 transitions per node which approximates over a year (371 days) of usage by a user who averages five transitions a day. To accomplish similar test coverage in a manual environment and assuming two network configurations a day with 60 transitions (an extremely aggressive pace considering the time to manually set up and debug the network, verify results across all clients and collate data), this kind of coverage would require at least 248 days.

BENEFITS AND USES

The flexibility of the architecture allows a wide variety of uses by users with a wide variety of skill levels. IT technicians could use it to deploy new networks or network segments. The Verification and Validation Layer of the architecture could also provide valuable network status information such as network traffic patterns, downed network segments, network saturation, and network health. Marketing personnel could use it for presentations to quickly set up networks and move through different scenarios. Customers out in the field could provide product support engineers with the specifics of their network environment, and the product support engineers could then use the framework to quickly recreate the customer environment to resolve

any issues. Validation engineers could use this architecture to automate test cases for increased testing coverage and log the results of tests on a wide range of real (non-simulated) networks. Validation and development engineers could use the framework to reproduce specific scenarios for debugging work. In each of these cases, no specific network component knowledge is needed to quickly and accurately configure a real-world network according to exact requirements. In addition, the cost in terms of resources, time, the need for network experts, and money is dramatically reduced by the adoption of the Dynamic Networks on Demand (DND) framework.

CHALLENGES AND NEXT STEPS

The next steps in developing the Dynamic Networks on Demand (DND) framework are to identify and extend the framework to handle the wider variety of scenarios required by the evolving needs of wireless networking projects.

For instance, creating interference of multiple signal strengths by access points (APs) would be useful for testing power optimizations and handoffs at boundaries as well as for testing behavior as signal strengths increase/decrease. The proposed solution would integrate an attenuator into the framework to programmatically initiate AP handoffs and increase/decrease signal strengths without actually moving components. The attenuator could also be used to address the challenge of introducing interference onto the wireless network.

Another example, stress testing of the AP with a maximum number of clients, is useful for test scenarios that examine behaviors of networks and clients in over-utilized environments. Solutions are being explored to load the network with a maximum number of clients without actual hardware, by using a simulator.

CONCLUSION

The framework is compelling for a variety of reasons. From a technical standpoint, the layered framework is extensible and reusable, and minimum effort is required to add new resources. Its modular design is easy to understand and enables easy addition of features. It greatly reduces manual operations (plugging and unplugging wires or devices, configuring devices), which saves network configuration turnaround and downtime, and it also allows a maximum number of network configurations to be achieved with a minimum hardware set.

From a non-technical viewpoint, by abstracting the network concepts and configurations, the architecture

allows network operations to become more accessible as networks become deployable, without requiring technical knowledge about all network components.

From either view, the architecture provides support for numerous uses that require accuracy in network tasks while at the same time saving time, money, and resources.

ACKNOWLEDGMENTS

The team acknowledges Allen Sampson for nurturing a quality group that fosters test innovation and automation, Denise Shoup for her work on the Control Layer, Mike Kasun for his test setup and execution, and the support of Mike Andrews, Ranjit Narjala, and Prakash Iyer.

REFERENCES

- [1] Robert V. Binder, *Testing Object-Oriented Systems Models, patterns, and tools*, Addison Wesley Longman Inc., Reading, Massachusetts, (c) 2000.
- [2] [Cem Kaner](#), [Jack Falk](#), [Hung Q. Nguyen](#), *Testing Computer Software, 2nd Edition*, John Wiley & Sons, (c) 1999.
- [3] [Boris Beizer](#), *Black-Box Testing: Techniques for Functional Testing of Software and Systems*, John Wiley & Sons, (c) 1995.

AUTHORS' BIOGRAPHIES

Amber Sistla is a network software quality engineer in the Network Architecture Labs in the Corporate Technology Group at Intel Corporation. Her technical interests include networking, wireless technologies, and test automation. She received her B.S. degree in Computer Science from Brigham Young University. Her e-mail is amber.sistla@intel.com

Jeremy Rover is a network software quality engineer in the Network Architecture Labs in the Corporate Technology Group at Intel Corporation. His current interests include wireless networking, dynamic network reconfiguration, and test automation. He received his B.A. degree in Computer Science from Linfield College. His e-mail is jeremy.rover@intel.com

Asha Keddy is a software quality manager in the Network Architecture Labs in the Corporate Technology Group at Intel Corporation. Her technical interests include network technologies, object-oriented design, software architectures, and innovative test automation. She obtained her B.E. degree in Computer Engineering from Bombay University India and her M.S. degree in Computer Science from Clemson University. Her e-mail is asha.r.keddy@intel.com

Copyright © Intel Corporation 2003. This publication was downloaded from <http://developer.intel.com/>.

Legal notices at <http://developer.intel.com/sites/developer/tradmarx.htm>.

For further information visit:

developer.intel.com/technology/itj/index.htm