17

SDN-Enabled Energy-Efficient Network Management

Michael Jarschel[1,2], Tobias Hoßfeld[1,3], Franco Davoli[4,5], Raffaele Bolla[4,5], Roberto Bruschi[5] and Alessandro Carrega[4,5]

[1] *University of Würzburg, Communication Networks, Würzburg, Germany*
[2] *Nokia Networks, Munich, Germany*
[3] *University of Duisburg-Essen, Modeling of Adaptive Systems, Essen, Germany*
[4] *Department of Electrical, Electronic and Telecommunications Engineering, and Naval Architecture (DITEN), University of Genoa, Genoa, Italy*
[5] *National Inter-university Consortium for Telecommunications (CNIT), University of Genoa Research Unit, Genoa, Italy*

17.1 Introduction

The continuing growth of network content, applications, and services have ultimately resulted in increasing requirements for data storage capacity and data transfer speed, which raise significantly the energy consumption. In order to cope with the limited resources of user devices, more and more applications and services follow the "cloud" paradigm in order to move most of the computational and storage weight to large-scale powerful datacenters [1, 2]. In 2012 alone, such datacenters required 30 billion watts of electricity worldwide [3], which demonstrates the unsustainability of current network infrastructures to satisfy an ever-growing demand for mass ICT services considering the environmental/economic concern.

Indeed, since the Future Internet is currently shaped, it makes sense to investigate new concepts and analyze key aspects, which have the potential to impact the evolving network design criteria. One of these aspects is energy efficiency.

Achieving energy-efficient network operation requires a more flexible network control and management, since resource allocation becomes more dynamic anticipating a higher degree of adaptability for new applications and network services.

Green Communications: Principles, Concepts and Practice, First Edition.
Edited by Konstantinos Samdanis, Peter Rost, Andreas Maeder, Michela Meo and Christos Verikoukis.

In this context, software defined networking (SDN) [4] and network functions virtualization (NFV) [5] are viable solutions to boost network capacity swiftly and flexibly ensuring smooth connectivity across increasingly complex networks and clouds for global users, who are often on the move. Not only that, but as networks are used for an increasing number of tasks of varying complexity, all of this needs to be done in a more efficient and cost-effective way.

The development of SDN-enabled applications that allow more efficient energy consumption requires the presence of specific functionalities inside the network devices. We refer to these functionalities as "Power Management Primitives" (PMPs). The remainder of this chapter is structured as follows. First we introduce the PMPs as well as global network primitives. Then, we define an SDN-based network architecture for energy-efficient networking and formulate the notion of green abstraction layer (GAL) [6] for the power management of an individual device. Finally, we draw our conclusions.

17.2 Background: Concepts for Network Operation

Two promising concepts to enable energy-efficient networking in the future are SDN and NFV. We briefly introduce both in this section.

17.2.1 Software Defined Networking

The key principle of SDN is the separation of the network elements' control plane to a central external entity. There are several advantages in this concept. The external control plane can be a software program run on commodity hardware. This removes the control plane from a monolithic device and enables development and adaptation of the control plane without the need for new hardware. On the other hand, the data plane has now become interchangeable and can consist of hardware built from standard components. Essentially, both control and data plane hardware are now commodity, and products of different vendors can be combined in a "mix-and-match" approach.

This is possible only because of the open interface, which SDN requires, between control and data plane. This interface is often called the "Southbound-API." The most popular realization of this interface is the OpenFlow protocol [7]. It provides a set of standard messages that allow an external control plane to operate its connected network elements via a management network. On top of the controller, the network functionalities are run as network control modules (NCMs). These modules are freely programmable and can be changed and combined according to the requirements of the network they operate. In addition, these control applications can communicate with the control plane of applications running on top or in conjunction with the network, for example, a cloud orchestration software, to optimize the network according to the applications' requirements. This interface is called the "Northbound-API." Figure 17.1 illustrates the components of the SDN architecture, as well as an OpenFlow-based realization. The OpenFlow switch (=data plane) is connected to the controller (=network control plane) via the OpenFlow protocol. In turn, the controller exchanges information with the network component (=application control plane) of a cloud orchestration software, that is, OpenStack, via a special purpose module. By introducing an abstraction or virtualization layer between controller and switch, it is also possible to change the modules entirely during the operation of the network. This provides the required flexibility for energy-efficient networking.

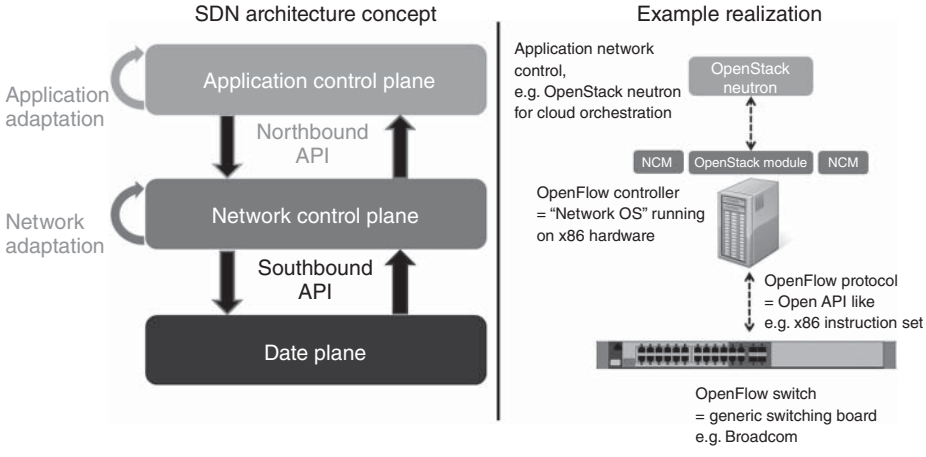

Figure 17.1 SDN concept sketch

17.2.2 Network Functions Virtualization

The concept of NFV is the next logical step in the commoditization process of networking. The central idea is to run network functions, for example, deep packet inspection or intrusion detection, in virtual machines on commodity hardware. Currently, these functions are realized on special purpose hardware, that is, middle boxes. With virtualized network functions the advantages of cloud computing can be leveraged. Network functions can be dynamically instantiated and removed, scaled flexibly, and relocated according to the current demands on the respective function. This way, resources that are not used can be shut down and energy can be saved. There are two key criteria for this concept to be deployed successfully. First, a swift I/O performance between the physical network interfaces of the hardware and the software user-plane in the virtual functions is required to enable sufficiently fast processing. Second, a well-integrated network management and cloud orchestration system is necessary to benefit from the advantages of dynamic resource allocation and to ensure a smooth operation of the NFV-enabled networks. Here, NFV can benefit from being deployed in conjunction with SDN. However, SDN is not a requirement. Figure 17.2 shows an example of such a joint SDN/NFV deployment enabling the flexible composition of multiple network functions.

In this example, an SDN switch is used to selectively redirect a portion of the production traffic to a server running virtualized network functions. This way the server and functions do not need to cope with all production traffic, but only the relevant flows. The virtual switch running inside the server's hypervisor is SDN-enabled and can dynamically redirect traffic flows transparently to an individual network function or to a chain of network functions. This enables a very flexible operation and network management, as functions can be plugged in and out of the service chain at runtime.

17.3 Energy-Efficient Network Management Practices

There is a variety of ways to improve the energy footprint of a network. The focus of this section concentrates on power management as well as on the energy efficiency of devices and

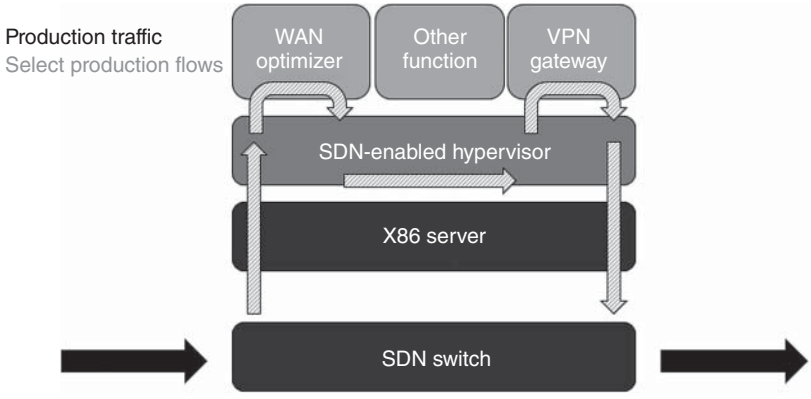

Figure 17.2 SDN/NFV functional composition

network primitives. We define a primitive as an inherent function or feature of a device or network that does not require any additional components to work.

17.3.1 Power Management Primitives

The vast majority of currently deployed network links and devices are designed to operate (and, consequently, to consume power) constantly at their maximum capacity, irrespective of the traffic load and even though their average utilization lies far below the maximum [8–10].

These observations have suggested a profitable power conservation potential for adapting the network offered resources to the actual traffic profiles [11, 12]. Similarly to general purpose computing systems, this ability can be realized by including PMPs into the hardware platforms of networking devices, where energy absorption physically takes place.

PMPs allow a highly dynamic adaptation of the energy consumption in networking devices or some of their components, by putting them into standby states when they are not used, or by decreasing their maximum performance in the presence of low incoming traffic volumes. The best performance is provided when the device operates under no power limitation, while the maximum power saving is obviously obtained when the equipment is completely turned off. There is a whole range of intermediate possibilities between these two extremes.

In principle, the main PMPs can be classified into two categories:

1. Dynamic adaptation (DA)
2. Sleeping/standby

The DA of network/device resources is designed to modulate (i.e., dynamically adapt) capacities of packet processing engines and of network interfaces, to meet actual traffic loads and quality of service (QoS) requirements. This can be performed by using two power-aware capabilities, namely, performance scaling (PS) and idle logic (IL), which both allow the dynamic trade-off between packet service performance and power consumption. PS adapts the processing speed (by changing operating frequency, possibly together with voltage, or throttling the clock), whereas IL exploits idle times (when no processing is required) to put

the processors into low power states and to resume the processing at the chosen speed when new packets arrive (thus, operating at the packet timescale).

Sleeping/Standby approaches are used to smartly and selectively drive unused network /device portions to low standby modes (or to switch them off altogether) and to wake them up only when needed (thus, operating at a much longer timescale with respect to packet processing/transmission times). However, since today's networks, related services, and applications are designed to be continuously and always available, standby modes have to be explicitly supported by using special proxying techniques that are able to maintain the "network presence" of sleeping nodes/components and to reactivate the sleeping device when needed (again, with a sensibly longer timescale than IL-related reactivation).

The PMP-enabled devices need control loops to dynamically tune hardware capabilities to provide the required QoS level for incoming traffic with minimal power consumption. It is worth noting that the PMPs have features locally available in network nodes, and their efficiency may heavily depend on the specific implementation and low-level details of a device's hardware platform; the latter may be quite heterogeneous (in terms of hardware components or firmware), even when considering equipment of the same market segment or vendor.

Owing to these considerations, it is necessary to provide each network device with its own independent control loop, namely, to provide local control policies (LCPs). These control loops may dynamically orchestrate the configuration of internal components (e.g., line-cards, link interfaces, network processors) to meet the desired QoS with the minimum power consumption. However, when each device independently performs energy optimizations, the resulting overall network power conservation is not as high as in the case of cooperation among the participant network nodes. Along this direction, a number of approaches [13–15, 16] have been recently proposed in order to extend current routing and traffic engineering policies beyond conventional network QoS metrics and to also explicitly consider the energy consumption of the entire network.

Figure 17.3 shows the necessary steps in order to optimize the power consumption of the network device and its hardware (HW) components. Internally, a network device can be provided with several LCPs, each of which is designed for a specific goal (e.g., controlling a fan, scaling an operating frequency, to meet given requirements). These LCPs set the energy configuration of the HW components, by using the relative PMPs (a) that, in turn, act directly on the hardware components (b). The critical points of this scenario concern:

1. how the LCPs know the PMPs enabled in the specific devices;
2. how the PMPs can set the HW components considering heterogeneous equipment.

In addition to these problems, it is necessary to take into account that LCPs alone are not sufficient to optimize the energy consumption, while at the same time ensuring a given level of performance. Hence, it is necessary to introduce centralized control policies that consider the network as a whole. We refer to these policies as "Network-wide Control Policies" (NCPs).

Despite their high potential effect compared to LCPs, NCPs applied along suffer from certain drawbacks. First, NCPs can exhibit much higher feedback/convergence delays. Secondly, routing and traffic engineering frameworks generally may not have the ability of distinguishing how logical network entities can be mapped to physical resources, which directly cause energy absorption. Finally, NCPs often represent a network device simply as a node in a graph, whose arcs are the virtual/physical network links.

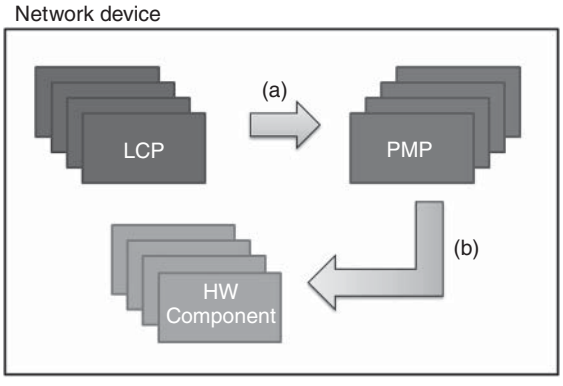

Figure 17.3 LCP–PMP interactions (from LCP control action (a) to its implementation on the physical component (b))

Such a simplistic representation does not allow retaining the knowledge of some hardware peculiarities (e.g., the possible different operating and idle states) that may be very important for reducing energy consumption. Such drawbacks suggest that jointly adopting LCPs and NCPs may optimize the device energy consumption, by following, for example, a hierarchical network management architecture.

Figure 17.4 shows a network example with 5 network devices (ND) connected to each other. Each ND supports LCPs and PMP functionalities, while one ND provides NCPs in order to control the other devices, by considering network-wide constraints. Such a case requires coordinated and well-defined interactions between the NCPs and the other NDs, where NCPs should not experience difficulties in obtaining and setting the energy-aware configuration of each ND.

On the basis of considerations, it is concluded that there is still a significant gap between hardware power management and LCPs/NCPs, as well as certain open issues regarding the adoption of control loops and their effective management, by considering the relationships among multiple local and/or network-wide control loops. Besides devising new energy-aware LCPs and NCPs, an almost necessary condition for their effective development and adoption is the representation of management and control actions, as well as device/network status information, in some standard abstract form, independently of the details of the specific manufacturers' implementations.

So, in order to better exploit the energy-aware functionalities to optimize the network devices considering the energy consumption, it is necessary to define an interface that provides a way to expose green networking capabilities of devices toward the network control plane. We refer to this interface as "Green Abstraction Layer."

17.3.2 Network Primitives

Apart from actively influencing and optimizing the power consumption of intermediate devices and middle boxes locally, the design and operation of the network as a whole can be adapted

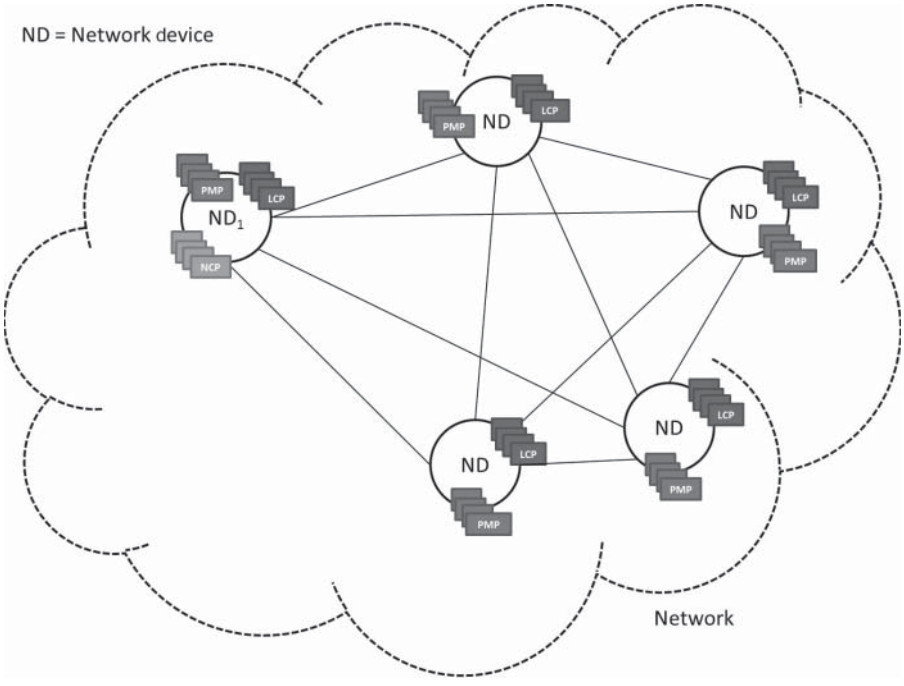

Figure 17.4 LCP–NCP interactions

to be as resource- and, therefore, power-efficient as possible. Hence, we identify two classes of power consumption optimization methods:

1. Local, hardware-specific, and device-based
2. Global, parameter-based, and network-wide

The former approach was discussed in the previous section. In this section we discuss how to make energy-efficient operation and resource management intrinsic to the network, that is, a network primitive.

To enable such a network primitive, the network needs to be able to instantiate and operate network devices in a highly dynamic way [17]. This is necessary because the demands on the network infrastructure change frequently. Technologies that meet the criteria to create such a network are virtualization and the related NFV, as well as SDN. Together they form the foundation for a network substrate based on commodity hardware. At first glance, the advantage of commodity hardware opposed to special purpose devices in terms of energy efficiency may not be obvious. The designers of special purpose hardware can build the equipment for one specific purpose and thus optimize the power usage of each device toward that. However, during the normal operation of a network, not all devices are needed all the time or only at a relatively low level of utilization. This means that even though the devices themselves may be energy efficient, their operation is not.

On commodity hardware, arbitrary network devices can be instantiated as virtual machines on-demand when needed. While there is a performance overhead for an individual device instance, this approach allows a high utilization of available network resources through the consolidation of individual network functions and devices. Therefore, while each virtual device may consume more power under heavy load than its hardware counterpart, the reduction in the amount of underutilized resources in the network yields the benefit that less network equipment is running permanently, which reduces the overall power consumption. Leveraging today's cloud management systems, the instantiation of a virtual machine with a specific network function can be highly automatized and performed very quickly. In the area of NFV, research is conducted to minimize the overhead of virtualization and thus advance this approach further [18].

Virtual devices alone, however, are not sufficient for this kind of efficient network operation. Decisions to reroute traffic to and from a new device instance have to be on pace with the setup of the device. In the traditional distributed routing approach, no single device has a view on the entire network and the flow of traffic. This makes it extremely difficult to influence routing decisions in such a way that the traffic flow can be directed efficiently. Here, SDN offers a way to bring the network up to the speed of the cloud systems. The SDN concept allows the separation of a network device's control plane that can be relocated to a central entity – the SDN controller. Here, monitoring and control information can be aggregated and subsequently be used to make network-wide routing decisions that ensure an energy-efficient operation of the network using NCPs, as depicted in Figure 17.5.

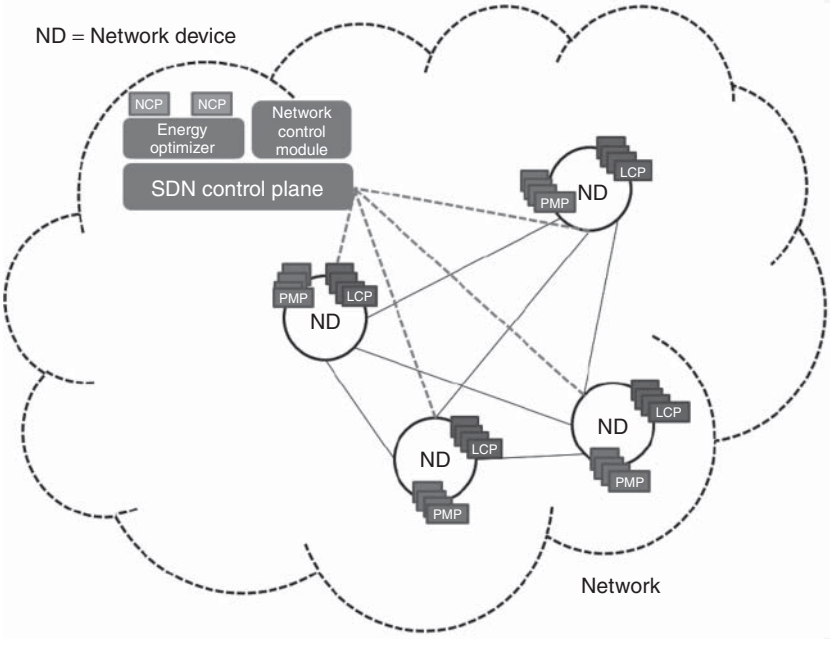

Figure 17.5 SDN-based NCPs

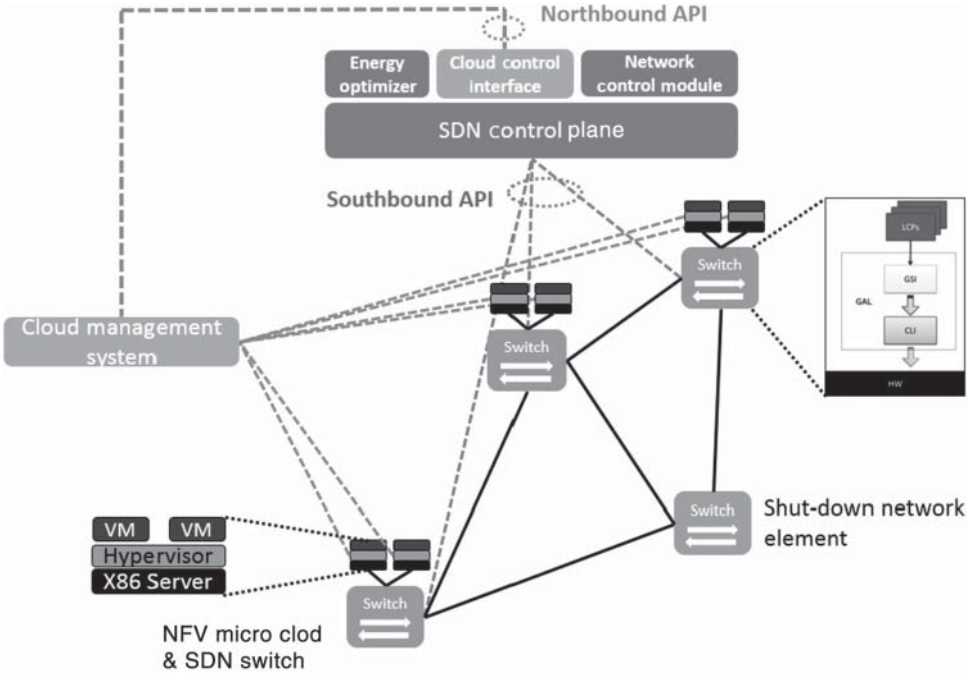

Figure 17.6 SDN/NFV-based energy-efficient network architecture

17.4 Energy-Efficient Network Management Enablers

17.4.1 SDN/NFV-based Energy-Efficient Network Architecture

Figure 17.6 shows a simple example of SDN/NFV-based network architecture, wherein a network element consists of an SDN-enabled switch and an adjacent NFV micro cloud. The SDN switch is responsible for forwarding traffic as dictated by the rules from the SDN control plane. The micro cloud serves to rapidly instantiate network functions as virtual machines on-demand, a process controlled by the cloud management system. The SDN control plane is connected to the cloud management system via a cloud control interface module that represents a realization of SDN's "Northbound-API." Both control planes receive monitoring information from an external network management system not depicted in Figure 17.6 for the purpose of simplicity.

This system is also responsible for the initial configuration of the network elements. The energy optimizer module running on top of the SDN controller uses the monitoring information to calculate the traffic in the network, while it receives load information for the deployed network functions, for example, caches, from the cloud management system via the cloud control interface. Using a scoring system based on, for example, service level agreements (SLAs), the energy optimizer can calculate a network arrangement where the placement of functionality preserves a good trade-off between the maintenance of service quality and energy conservation. Once such a placement is determined, the energy optimizer can then notify the cloud

management system as well as the network control module to instantiate, move, or remove resources, or turn devices off and on, via the network management system. Leveraging the Green Abstraction Layer described in the next section, a more granular approach, that is, shutting down components of devices like ports, is also possible.

17.4.2 Green Abstraction Layer

The GAL, currently under discussion for standardization within the ETSI Environmental Engineering Technical Committee under Work Item DES/EE-0030, is defined with the purpose to simplify the management of the energy-aware characteristics of network devices and their subcomponents. The GAL synthetizes the data related to power management settings into a sort of standard data objects, namely, "Energy-Aware States" (EASes). The GAL manages the EASes through two interfaces:

- Green standard interface (GSI): to exchange power management data among data plane elements and processes realizing control plane strategies (LCPs and NCPs) in a simplified way;
- Convergence layer interface (CLI): to map the GAL commands and data into low-level configuration registers/APIs, which are manufacturer/hardware specific and allow hiding the HW heterogeneity to LCPs/NCPs.

In general, the GAL will be used by three main sets of energy-aware control plane processes (LCPs, NCPs, monitoring and operation, administration & management – OAM).

The energy-aware LCPs aim to optimize the configuration at the device level, in order to achieve the desired trade-off between energy consumption and network performance, according to the incoming traffic load. To this purpose, such processes need to know in detail the internal architecture of the device (or parts thereof), the number, the typology, and the capability of energy-aware elements, as well as to have access to network performance indexes.

Instead, the energy-aware NCPs aim to autonomously control and optimize the network behavior considering a set of devices. Typical examples of this kind of processes are traffic engineering, routing, and signaling algorithms/protocols (e.g., OSPF-TE/RSVP-TE) with "green" extensions.

Finally, the energy-aware OAM processes are used by the operator to control and optimize the behavior of a network (as in network management systems with "green" capabilities).

The adoption of local, network-wide, and OAM policies is not necessarily exclusive, but they are conceived to work with different goals and in different contexts in a complementary way. Notwithstanding these differences, the behavior of local policies cannot be completely independent of network-wide and OAM control frameworks. A simple example of such dependency is given by the fact that the optimal local configuration relies on the traffic load incoming from device links, which, in turn, is influenced by network-wide and OAM policies.

17.4.3 GAL Main Design

The goal of the GAL is to extend and reengineer the ACPI standard [19] for general purpose computing systems and adapt it to architectures, functionalities, and paradigms of network devices, and especially of their data plane components.

It is worth noting that the GAL is a device internal interface, so it does not behave as a network signaling protocol. Control plane processes implement signaling protocols to make network nodes converge to a certain configuration. This is not a direct goal of the GAL, but these control plane processes need to be interfaced with the GAL for acquiring/setting power-related parameters. For example, considering the SDN area, such signaling protocols can be based on the OpenFlow specification.

In other words, the GAL is conceived to make control processes acquiring information on the green capabilities available at the data plane, configuring them, and carrying measurements on the energy consumption.

Thus, unlike the ACPI standard, the specification of the GAL must consider the presence and the main features of multiple and heterogeneous internal components (e.g., link interfaces, multiple chips for packet processing, fans) with energy adaptation and monitoring capabilities, and hide the complexity to the high-level control policies. In addition, it can simplify the management of complex network devices composed of several HW components, by providing methodologies for aggregating all the information from single internal components (e.g., single component, link level, line-card, chassis, and node levels). Obviously, the GAL has to characterize the effect of putting components in a certain energy configuration in terms of power consumption and network performance (e.g., minimum and maximum energy consumption, maximum throughput), by exposing this information to LCPs and NCPs.

The energy-aware configuration is represented by a structure called "Energy Aware State" – EAS. The interaction between NCPs and LCPs takes place via SDN communication by using the primitives provided by the GAL framework.

Figure 17.7 depicts an example, which illustrates how the GAL framework is implemented inside a network device wherein several LCPs are installed, too. These LCPs set and get the

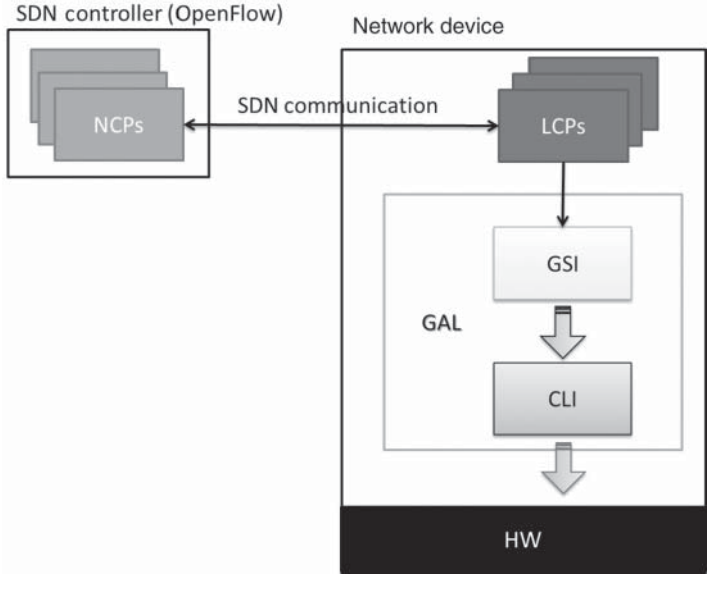

Figure 17.7 GAL–NCP–LCP communications in the SDN framework

energy-aware configuration by means of the EASes and by using the GSI. Inside the GAL framework, each GSI request is translated by the CLI in a specific command for the underlying HW components. The NCPs are installed in a remote device as modules of an SDN controller (in this specific example, we consider OpenFlow). Each interaction between the NCPs and the LCPs is performed according to the OpenFlow specification.

17.4.4 GAL Hierarchical Structure

In this section, we describe in more detail the hierarchical structure provided by the GAL framework.

The GAL architecture is designed as a modular and easily extendable software framework, providing interface capabilities toward heterogeneous HW, as well as multiple hierarchical interfaces toward control processes, in order to set energy configurations at various detailed levels of the internal architecture.

Figure 17.8 shows the case of a multichassis network device including different HW components with energy-aware capabilities, which may be managed by control plane processes. Such device can be represented at various levels: single HW component and/or physical link levels, line-card, chassis, and node levels.

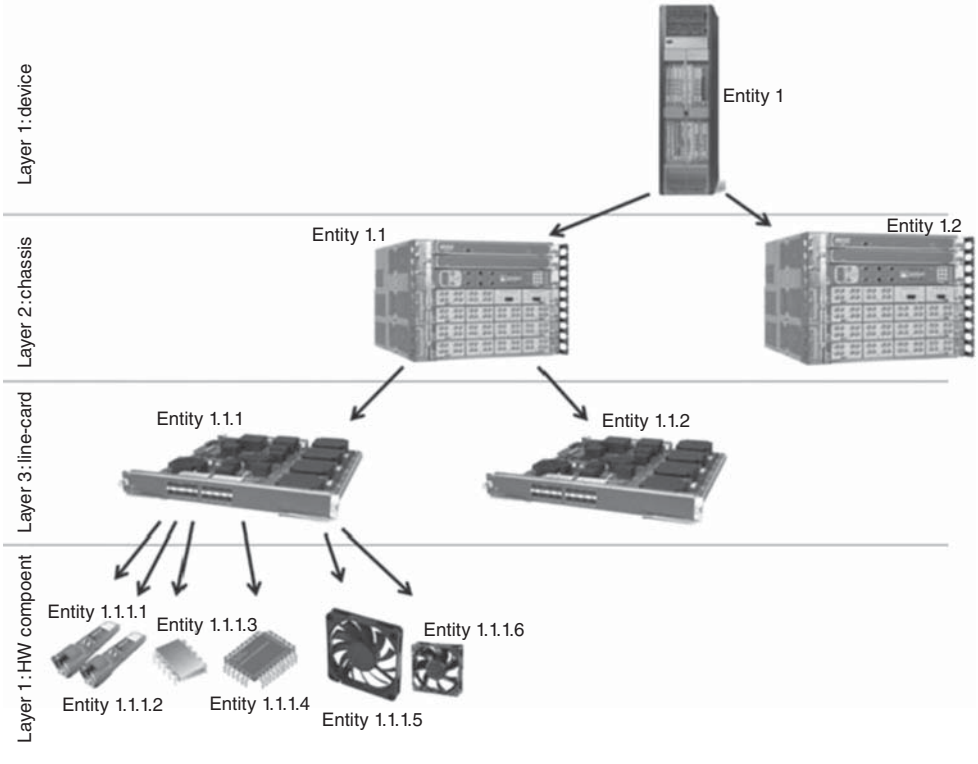

Figure 17.8 The hierarchical architecture of a multichassis network device

The GAL allows the interaction between the power management usually realized inside HW components and network control processes (e.g., routing and traffic engineering signaling protocols) run at the device level. When new device configurations are determined by such control processes, the GAL translates them into specific settings of components at underlying levels. For example, if the network-wide control decides to scale the speed of a link, it has to pass this command to the chassis and then to the line-card hosting the link. The line-card can scale the speed of the physical link, as well as the speed of the related HW (which can be composed of multiple energy-aware components). Thus, a single configuration decided at the device level may impact on multiple underlying components.

Given the fact that power management primitives (PMPs) reside at the lowest levels of the hierarchy, they shall require specific LCPs to directly manage them to achieve the desired operating behavior. For this reason, the GAL is used for interfacing such lowest level energy-aware components with some control plane processes implementing HW-specific optimization strategies.

Then, at intermediated levels (e.g., line-card or chassis), new LCPs (one for each entity at that level) are needed to orchestrate the settings and the operating behaviors of underlying energy-aware components, and to expose a synthetic and aggregate set of operating characteristics and available configurations (of the line-card, or of the chassis) to higher levels.

This process terminates at the device level, where the highest LCP orchestrates the high-level configuration of the device and needs to expose a simplified view of it to network signaling protocols (network-wide and OAM control applications).

The approach pursued here consists of a chain of LCPs, which control from single energy-aware HW components to the entire device. LCPs at different levels need to interoperate, and the highest level LCP needs to interact with processes realizing network signaling protocols.

The result of such hierarchical approach consists of a tree, where root nodes are LCPs and control applications, and leaf nodes correspond to HW elements. The interface among the tree nodes is realized by means of the GAL. Leaf nodes may reside at different levels of the hierarchy for two main reasons:

- Some HW energy-aware component may be accessible at higher hierarchical levels (e.g., fans in a chassis);
- Some manufacturers shall prefer not to expose the internal organization and the subcomponents of a "composite" part of the device (e.g., line-card, chassis).

In the latter case, the "composite" part of the device will be treated as a single HW energy-aware element.

17.5 Conclusions

This chapter focuses on the problem of energy consumption, considering the current network infrastructures, which are not prepared to satisfy an ever-growing demand for mass ICT services with the scalability, flexibility, and effectiveness needed to cope with future technology and traffic trends, as well as environmental and economic challenges. Therefore, it is desired to extend the network design criteria including new different aspects, such as energy efficiency.

The design of energy-efficient networks requires a more flexible approach, making necessary the dynamic allocation of resources that the traditional networks are not able to realize.

The SDN/NFV paradigms can be a viable solution to improve the network functionalities and capacity, by allowing a flexible and dynamic resource allocation. The development of SDN-enabled applications that allow more efficient-energy consumption requires the presence of specific functionalities inside the network devices necessary to manage the energy/performance trade-off.

In this chapter, we introduced the notion of PMPs and also described global network primitives. There is a variety of ways to improve the energy footprint of a network. We focus on providing power management and network primitives, which are inherent functions or features of a device or network that does not require any additional components to operate.

Furthermore, we analyzed an SDN-based network architecture for providing energy-efficient networking, by formulating the notion of a GAL for the power management of individual devices. The architectural considerations and interfaces enabling the GAL have been elaborated, which enable to export data plane energy-aware capabilities from network devices toward the Control Plane. Specifically, the GAL synthetizes the data related to power management settings into a sort of standard data objects, namely, "Energy-Aware States" (EASes).

We have outlined the definition of two interfaces, the Green Standard Interface, which is the "external" interface used to interact with clients and applications and an internal interface (convergence layer).

Finally, we described the hierarchical architecture of the GAL that, with a functionally complete set of primitives, can be very useful and suitable for the management of resources in energy-aware SDN platforms.

References

[1] K. W. Cameron, "Energy oddities, part 2: why green computing is odd," IEEE Comput., vol. 46, no. 3, pp. 90–94, 2013.

[2] P. M. Corcoran, "Cloud computing and consumer electronics: a perfect match or a hidden storm?," IEEE Consumer Electron. Mag., vol. 1, no. 2, pp. 14-19, 2012.

[3] J. Glanz, "The cloud factories – power, pollution and the Internet," The New York Times, Sept. 22, 2012, http://www.nytimes.com/2012/09/23/technology/data-centers-waste-vast-amounts-of-energy-belying-industry-image.html?_r=0.

[4] White paper on "Software Defined Networking", https://www.opennetworking.org/sdn-resources/sdn-library/whitepapers.

[5] White paper on "Network Functions Virtualisation", http://portal.etsi.org/NFV/NFV_White_Paper.pdf.

[6] R. Bolla, R. Bruschi, F. Davoli, L. Di Gregorio, P. Donadio, L. Fialho, M. Collier, A. Lombardo, D. Reforgiato, and T. Szemethy, "The green abstraction layer: a standard power-management interface for next-generation network devices," IEEE Internet Comput., vol. 17, no. 2, pp. 82–86, 2013.

[7] The OpenFlow Specification, URL: http://www.openflow.org.

[8] R. Bolla. R. Bruschi, A. Carrega, F. Davoli, D. Suino, C. Vassilakis, and A. Zafeiropoulos, "Cutting the energy bills of Internet service providers and telecoms trough power management: an impact analysis," Comput. Netw., vol. 56, no. 10, pp. 2320–2342, 2012.

[9] The Climate Group and Global e-Sustainability Initiative, "SMART 2020: enabling the low carbon economy in the information age," http://www.theclimategroup.org/, pp. 1–87, 2008.

[10] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing network energy consumption via sleeping and rate-adaptation," Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation, pp. 323–336, 2008.

[11] R. Bolla, F. Davoli, R. Bruschi, K. Christensen, F. Cucchielti, and S. Singh, "The potential impact of green technologies in next-generation wireline networks: is there room for energy saving optimization?," IEEE Commun. Mag., vol. 49, no. 8, pp. 80–86, 2011.
[12] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, "Energy efficiency in the future Internet: a survey of existing approaches and trends in energy-aware fixed network infrastructures," IEEE Commun. Surveys Tutorials, vol. 13, no. 2, pp. 223–244, 2011.
[13] L. Chiaraviglio, M. Mellia, and F. Neri, "Minimizing ISP network energy cost: formulation and solutions," IEEE/ACM Trans. Netw., vol. 20, no. 2, pp. 463–476, 2011.
[14] A. Cianfrani, V. Eramo, M. Listanti, and M. Polverini, "An OSPF enhancement for energy saving in IP networks," Proceedings of the 2011 IEEE Conference on Computer Communications, pp. 325–330, 2011.
[15] J. C. C. Restrepo, C. G. Gruber, and C. M. Machuca, "Energy profile aware routing," Proceedings of the 2009 IEEE International Conference on Communications, pp. 1–5, 2009.
[16] R. Bolla, R. Bruschi, A. Cianfrani, and M. Listanti, "Enabling backbone networks to sleep," IEEE Netw. Mag., vol. 25, no. 2, pp. 26–31, 2011.
[17] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "ElasticTree: saving energy in data center networks," In NSDI'10 – Proc. 7th USENIX conference on Networked Systems Design and Implementation, San Jose, CA, 2010, vol. 3, pp. 1–16; https://www.usenix.org/legacy/event/nsdi10/tech/full_papers/heller.pdf.
[18] S. Niccolini, "Free Your Middlebox Functions down to the Data Plane with Tiny, Fast Network VMs," Invited talk at European Workshop on Software Defined Networks, Darmstadt, Germany, 2012.
[19] Advanced Configuration & Power Interface (ACPI), URL: http://www.acpi.info.