The Evolution of networking

Status Quo

Network devices are vertically integrated systems with single application hardware components, vendor specific (and often product line specific) operating systems and, more importantly, a closed application ecosystem.
This is a scenario which resembles the initial stage of development that we have seen in other industries such as mainframes/servers and mobile devices. The main economic issue is that these closed vertically integrated systems slow down innovation.
This can be observed in terms of the slow evolution of control applications. For example, IEEE bridging and CLI based provisioning are totally inadequate to the needs of a modern data center but are still the prevalent technologies in use.

The stress factors

Over the last 15 years software development, outside of the networking industry, has changed dramatically. Due to a combination of factors the software development cycle has improved in agility by an order of magnitude. Contributing factors include the extensive use of application frameworks, design patterns, new methodologies, etc.
There has also been a shift in thinking from the big reliable server in the mainframe style to very large clusters of unreliable machines. Over this time period networking technology, as applicable to the data center has remain stagnant, other than in terms of speeds and feeds.

There are two main assumptions to be extracted from this observation:

1) The application development life-cycle is a critical concern to a business. The expectation is that an internet-scale application can go from concept to production in weeks rather than months. This is incompatible with the traditional organization in which networking devices are provisioned manually by a separate group within the company which is not involved in the application development.

2) The scale-out approach with clusters of independent machines and the requirement to support consumer access to applications via any device and any location create a resource management problem: compute, data/content and connectivity must be coordinated. An application that is employing a map/reduce approach needs its compute nodes to be located in a physical environment where they have the required network bandwidth; a consumer application or even an internal I.T application that is serving mobile customers needs a content distribution and caching strategy in order to achieve the desired user experience.

Based on these, one can draw the conclusion that there is a business need to be able to provision the access points of the network in a dynamic way from the application platform. It is the application orchestration layer that is best aware of the resource requirements; it is this layer

that can define the logic network partitions and access control rules. However in order to do its job correctly it also needs to be able to query the network and understand parameters such as available bandwidth, network proximity, etc.

This argues for a strong programmatic API between these two layers that allow information to flow bi-directionally. Logic partitioning, access control and access-point provisioning needs to flow top-down; network state information that is relevant to the application load-balancing decisions must be made available northbound.

These leads us to a rather different vision of how networks are operated: the network core needs to provide a transport service that is well defined; the network access-point needs to be dynamically controlled by the layer above.

The ecosystem

The conclusion at which we arrived above is potentially very disruptive to traditional network vendors and to the traditional network operations group inside the typical I.T. organization. From the business side, the transfer of control from the network to the layer above is a concern. From an engineering side, the typically network OS is rather unsuited to be controlled via an external entity. CLI still dominates anything that is not dynamic routing.

For networking to leave the 80s it is going to be necessary to rethink the network device as an open rather than a closed platform. Many of the required factors are present: there is an active industry of silicon vendors providing off-the-shelf ASIC and network processor solutions; OEMs are capable of putting together the system level design; and both the server and mobile markets have proven that if you give software developers the ability to develop against a platform that the pace of innovation goes up exponentially.

The missing piece in this equation is a software platform that plays the role that Android plays in the mobile space. A platform that enables any OEM and hardware manufacturer to design against it and provide the drivers that enable they proprietary hardware; And that enables application developers to design against a standard set of APIs.

At the technical level, the functionality that is required from this platform is rather distinct from an environment such as Android and more closely resemble the SOA concepts that we find in enterprise software development. In network devices, multiple control applications must cooperate in order to define the treatment that must be applied to specific collections of packets. These control applications must be composable.

A typical technique is to model the treatment of packets on the data plane as a pattern matching graph where special arcs can modify the contents of the packet. This generic graph concept must be mapped to a specific forwarding implementation (ASIC, NPU, CPU, etc).

The task of the platform layer is to provide SOA interfaces to the control applications and map the control application concepts to a agreed upon graph. While the different networking operating systems on the market today have different nomenclature for these functions that majority of them follows the pattern above.

However the industry as a whole is missing an open platform that can achieve this role on a wide variety of systems. It is very likely that where such platform to exist that this would lead to a much faster pace of innovation in networking.