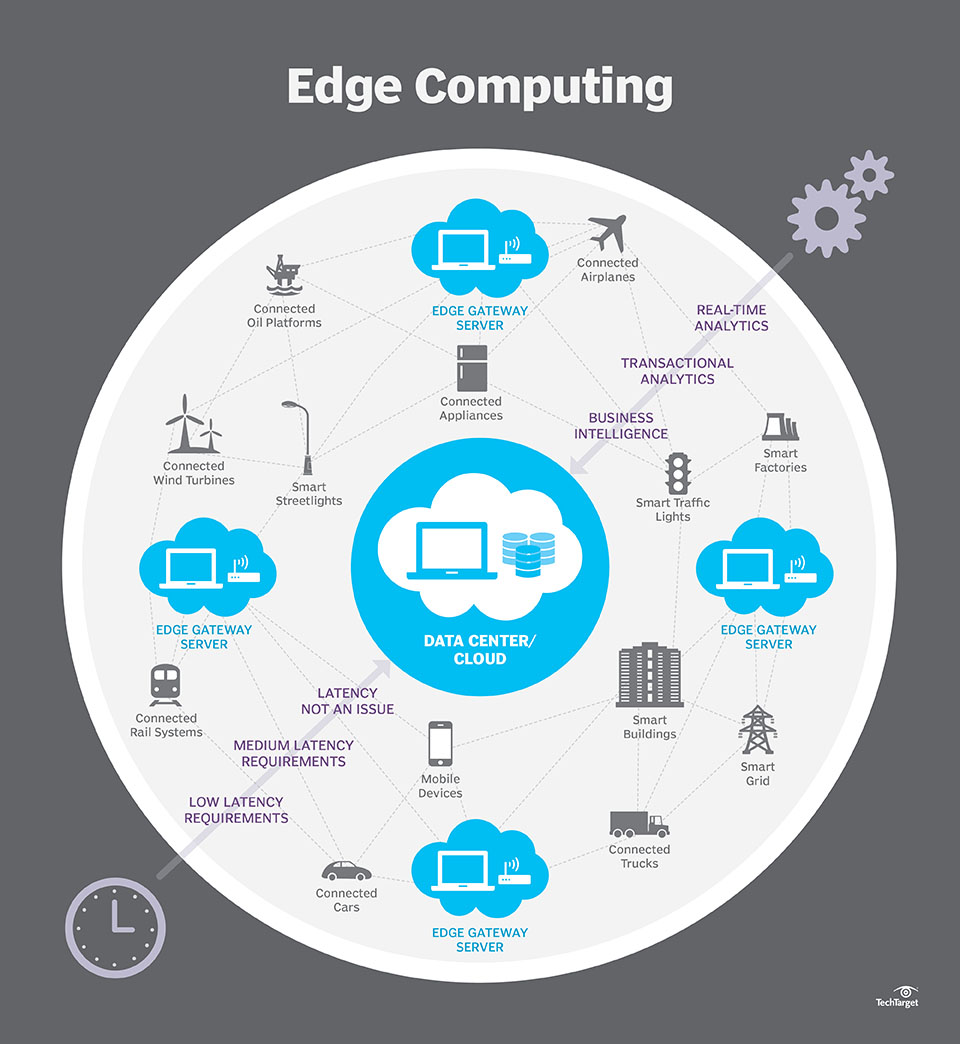
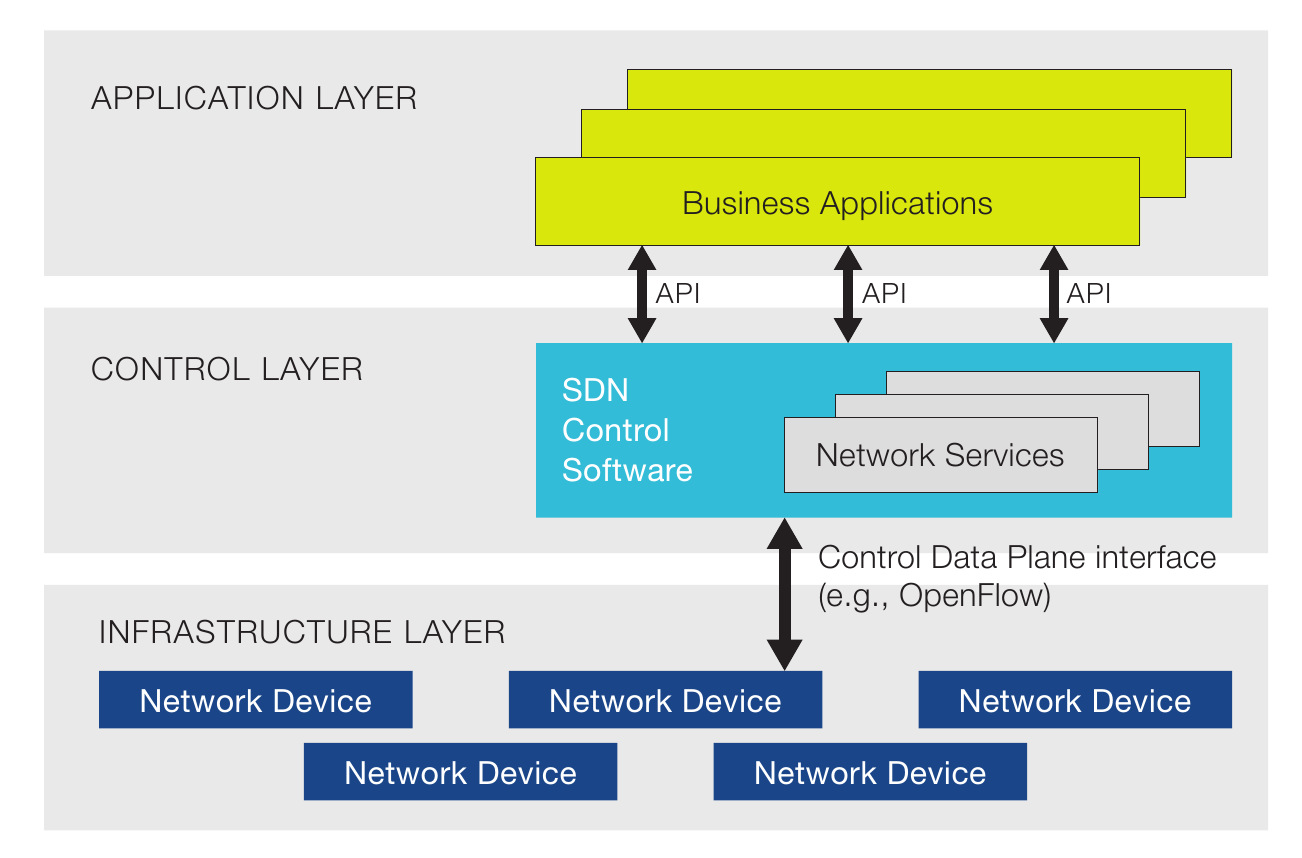
**What is Edge computing?**

Edge computing is a method of performing data processing technology to optimize [cloud computing](https://en.wikipedia.org/wiki/Cloud_computing" \o "Cloud computing) system at edge of network. The operation of the application, data and services, from the network center node to the node of the edge of network logic to deal. The edge computing decompose the large service which originally processed by the central node, and cut them into smaller and more manageable parts, scattered to the edge node to handle. The edge node is closer to the user terminal device, so it can improve the speed about the processing and transmission of the data and reduce the delay. In this architecture, data analysis and data generation are closer to the source of data, therefore, it is more suitable for dealing with large data. Edge Computing covers a wide range of areas including [wireless sensor network](https://en.wikipedia.org/wiki/Wireless_sensor_network" \o "Wireless sensor network), mobile data acquisition, mobile signature analysis and cooperative distributed [peer-to-peer](https://en.wikipedia.org/wiki/Peer-to-peer" \o "Peer-to-peer) ad hoc network.



**What is SDN?**

Software Defined Networking (SDN) is an emerging architecture that is dynamic, manageable, cost-effective, and adaptable, making it ideal for the high-bandwidth, dynamic nature of today`s applications. This architecture has network control and forwarding functions and enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. The OpenFlow protocol is a foundational element for building SDN solutions. Due to these advantages, we believe that SDN will greatly affect on network infrastructure and protocol in the near future.



**Why edge computing needs SDN?**

As the use of mobile devices increases, this brought tremendous pressure to the cloud computing network, and this situation will only be more serious as the use of global mobile devices increases. Creating a unified system of cloud computing and edge computing is an effective way to cope with overloaded resources and latency challenges. However, the integration of cloud computing and edge computing is also facing challenges. There must be a local coordinator to allocate resources real-time in dynamic and unpredictable environments. The system must implement real-time updates to provide the best information about available resources and have an open programmable interface to perform the task in the most efficient way. So we need Software Defined network(SDN).

**Open source SDN switch and controller**

**Open source controller:**

1. OpenDaylight

OpenDaylight led to the open source community, open source framework for implementation of the Java language, to promote innovation and implementation of software-defined network transparency. In the face of SDN network, OpenDaylight as a project core, has a modular, pluggable and extremely flexible controller, also includes a set of modules, to perform the need to quickly complete the network tasks.

1. Floodlight

Floodlight is an OpenFlow controller developed by Big Switch Networks using the apache protocol and the Java language, it is used to work with switches, routers, virtual switches, and other devices that support the OpenFlow standard.

1. RYU

Ryu is a component-based software defined networking framework.

Ryu provides software components with well defined API that make it easy for developers to create new network management and control applications. Ryu supports various protocols for managing network devices, such as OpenFlow, Netconf, OF-config, etc. About OpenFlow, Ryu supports fully 1.0, 1.2, 1.3, 1.4, 1.5 and Nicira Extensions.

All of the code is freely available under the Apache 2.0 license. Ryu is fully written in Python.

1. ONOS

ONOS has been designed to take a disruptive approach to networking, leveraging white box merchant silicon hardware to build carrier-grade solutions. By moving network control into the ONOS cloud controller, innovation is enabled and end-users can easily create new network applications without the need to alter the dataplane systems.

1. MUL

MUL is a multi-threaded architecture of the OpenFlow controller and achieved by C language, used to connect the app of a variety of northbound interface, currently supports openFlow 1.0,1.3 version, mainly for performance and reliability design, is a lightweight and efficient Controller. Support one-button installation, CTL command line management and WEB GUI.

**Open source switch:**

1. Open Vswitch

OpenvSwitch is a production quality, multilayer virtual switch licensed under the open source Apache 2.0 license. It is designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols (e.g. NetFlow, sFlow, IPFIX, RSPAN, CLI, LACP, 802.1ag). In addition, it is designed to support distribution across multiple physical servers similar to VMware's vNetwork distributed vswitch or Cisco's Nexus 1000V.

1. LINC Switch

LINC is a pure OpenFlow software switch written in Erlang. It's implemented in operating system's userspace as an Erlang node. Such approach is not the most efficient one, but it gives a lot of flexibility and allows quick development and testing of new OpenFlow features.

1. Indigo

Indigo is an open source OpenFlow implementation achieved by Big Switch Network based on the OpenFlow reference scheme of Stanford University using C language, it is running on the physical switch, it can support more than 48 high-rate 10G ports, and support for scalable network virtualization applications, as well as the use of OpenFlow controller across the multi-server distributed architecture, similar to VMware's vNetwork, Cisco's Nexus, Open vSwitch.

1. ONetSwitch

ONetSwitch30 is a Quad Gigabit Ethernet Ports SBC based on [Xilinx XC7Z030](http://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf), which combines the software programmability of ARM processors with the hardware programmability of FPGAs. With a FPGA programmable accelerator, five Gigabit Ethernet ports, Up to 3GB DDR3 DRAM, GPIO, and Mini PCIe Slot for WLAN Card or SSD Card.

1. Pantou(OpenWRT)

Pantou turns a commercial wireless router/Access Point to an OpenFlow-enabled switch. OpenFlow is implemented as an application on top of OpenWrt. Pantou is based on the BackFire OpenWrt release (Linux 2.6.32). The OpenFlow module is based on the Stanford reference implementation (userspace).

**Why we choose plan A?**

The intention is to build a SDN switch based on Raspberry Pi, but performance of Raspberry Pi is low compared with PC. Considering the portability and lightweight of the software, the LINC Switch and RYU had been chosen. Meanwhile, a tutorial has realized the configuration of *Plan\_A* in the internet.

**Features of LINC Switch:**

LINC is an open source based on Erlang language and support OpenFlow protocol. Features are as follows:

* Support OpenFlow Protocol 1.2, OpenFlow Protocol 1.3, and OpenFlow Protocol 1.4,
* OpenFlow Capable Switch – able to run multiple logical switches,
* Support for OF-Config 1.1.1 management protocol,
* Modular architecture, easily extensible.

**Why LINC Switch?**

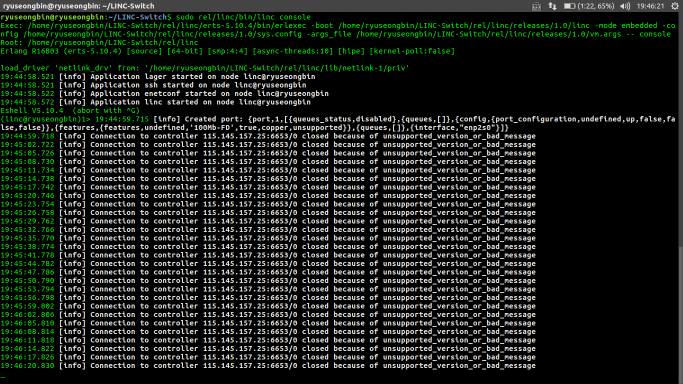
* It provides full support for the OpenFlow feature, which facilitates in-depth understanding of the principles and details of OpenFlow.
* With the support of Erlang, the multi-core processors are fully used, which enhances the processing ability of standard hardware.

**Why RYU controller?**

* Support various protocols for managing network devices, such as OpenFlow, Netconf, OF-config, etc.
* Fully support 1.0, 1.2, 1.3, 1.4, 1.5 and Nicira Extensions.
* RYU controller itself comes with a lot of good prepared APP, through the familiarity of these APP, you can master the ability to write their own APP through RYU.
* The RYU controller is written entirely in the Python language and we can quick to understand and get started.

**The reasons why Plan A is impossible.**

The LINC switch and Ryu controller are successfully installed in separated devices, when a “hello” handshake message is sent from switch to the controller, an error message is shown.



**Reason of failure:**

* Even with powerful performance of emulating SDN switch, the LINC Switch is a open source made by personal developers. It was not updated frequently and the latest version was released almost two years ago.
* The main reason is the communication protocol of the LINC Switch is not compatible with the current communication protocol, as the Ryu controller is updated frequently.