Hálózatok építése és üzemeltetése

SDN a gyakorlatban

Mai téma

- Hálózatok “szoftverizálása” - network softwarization
  - control plane szoftverizálása
    - SDN: Software Defined Networking
  - data plane szoftverizálása
    - NFV: Network Function Virtualization
  - hálózati szolgáltatások/alkalmazások szoftverizálása
    - SFC: Service Function Chaining
- Egy konkrét példa: OpenFlow

Kiktől loptam slide-okat?

- Rob Sherwood
  - “GENI Engineering Workshop June 2010”
- Guido Appenzeller
- Nick McKeown
- Guru Parulkar
- Brandon Heller
- Marco Cello
- Seyed Kaveh Fayazbakhsh
- és még sokaktól…

- (Még ez a slide is lopott…)
- (Persze, ők is lopták egymástól…)

SDN koncepció

Marketing

- **Egy-két hír a közelmúltból:**
- “**Google** is using **OpenFlow** on custom-designed hardware for all the internal networks it runs connecting its global **data centers**, said **Urs Holzle**, senior vice president of technology infrastructure at Google”

- “**How Google is using OpenFlow to lower its network costs?** Google is checking out a new form of networking protocol known as OpenFlow, in the communications networks that run between its data centers. The search giant is testing the use of software defined networks in order to lower the cost of delivering a bit of information.” (gigaom.com)

- “Virtualization and cloud infrastructure provider **VMware** (NYSE: VMW), announced this week that it will pay **$1.05 billion** in cash plus approximately $210 million in assumed unvested equity awards to **acquire Nicira**, a software-defined networking (**SDN**) **specialist** and provider of network virtualization for open source initiatives.“ (RCR Wireless News – Americas)

Probléma

- Számítógép-hálózat
  - bonyolult, elosztott rendszer
  - különböző HW eszközökből áll
    - switch, router, middlebox, …
    - zárt, gyártóspecifikus HW, FW, SW
    - bonyolult, elosztott kontroll funkciók (pl. routing protokollok)
    - heterogén eszközök konfigurálása
    - különböző interfészek

Probléma

- Számítógép-hálózat
  - nehéz/költséges tervezés és üzemeltetés
  - konfigurálás ↔ programozás
  - lassú innováció (akadémia problémája)
  - drága (ipar problémája)
    - üzemeltetés
    - fejlesztés
    - új szolgáltatások bevezetése/beüzemelése

Cél

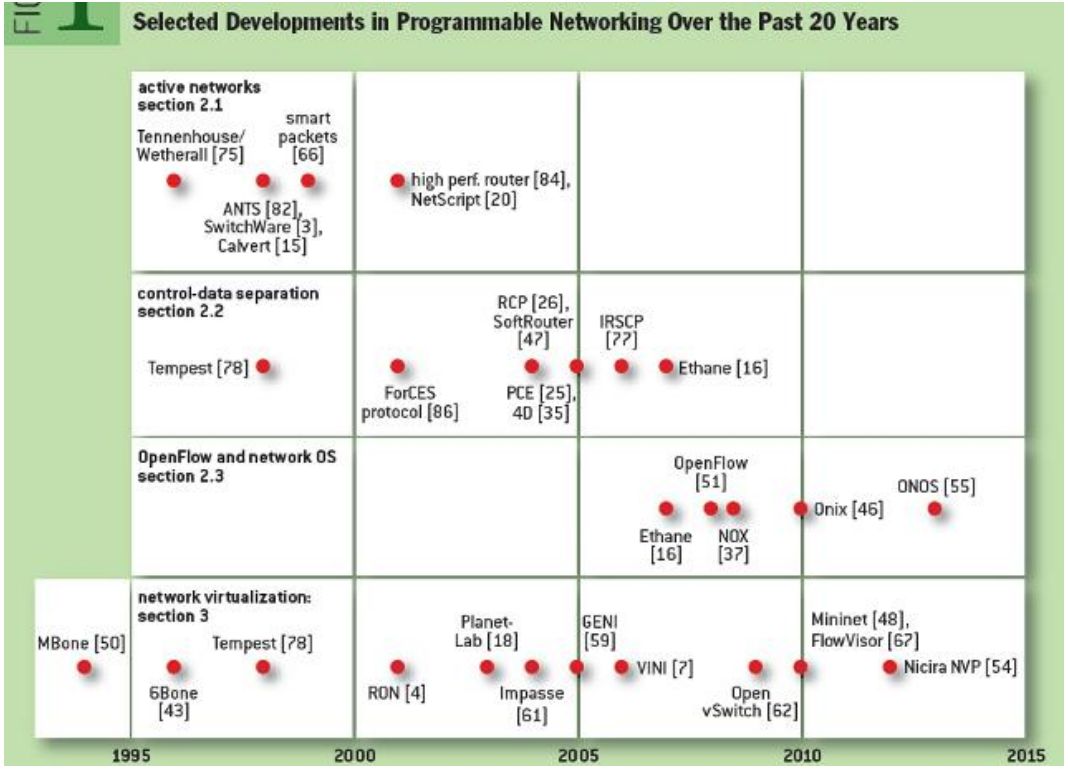
- Számítógép-hálózatok minél “jobb” programozhatósága
- programozhatóság
  - hálózat mint egész működésének meghatározása
  - több mint az egyes elemek működésének befolyásolása
  - konfigurálás ↔ programozás
  - időskála!
- “jobb”
  - könnyebb, gyorsabb
  - flexibilisebb
  - szélesebb körű
  - kevesebb hiba(lehetőség)
  - gyorsabb javíthatóság

(Egy) Megoldás(i irány)

- Software Defined Networking
  - kontrollsík szoftverizálása
  - kontrollsík ↔ adatsík szeparálása
    - kontroll: döntés hogy mi történjen az adott forgalommal
    - adatsík: csomagok továbbítása
  - kontrollsík centralizálása / konszolidálása / egységesítése
  - közöttük: nyílt interfész(ek)
  - korábban: elosztott rendszer, „sok-sok” kapcsolat
  - most: elosztott rendszer, „egy-sok” kapcsolat!
- Nem új ötlet!
  - több évtizedes út
  - sok korábbi ötlet felhasználása
  - pl. PSTN: kontroll- és adatsík szeparálás

Hogyan jutottunk az SDN-ig?

Hogyan jutottunk az SDN-ig?



N. Feamster, J. Rexford, E. Zegura, **The Road to SDN**, *ACM Queue,*, Volume 11, Issue 12, Dec. 2013

Hogyan jutottunk az SDN-ig?

- aktív hálózatok
  - clean-slate architektúra
  - vízió, elrugaszkodott az akkori hálózati valóságtól
  - új adatsík funkció kódja leküldhető az eszközökbe (~ Java kód)
    - kapszula modell (in-band, adatcsomagokban)
    - programozható router/switch model (out-of-band)
  - egyéges végrehajtási környezet az adatsík csomópontjaiban (EE, execution environment)
  - middleboxok egységes kezelése
  - adatsík programozása
  - sok ötlet mára újra előjött!

Hogyan jutottunk az SDN-ig?

- kontroll-adat szeparálás
  - ForCES (Forwarding and Control Element Separation)
    - IETF szabvány
    - nyílt interfész az adatsík felé
    - innováció a kontrollsík szoftvereiben
  - SoftRouter
    - ForCES API-t használja
    - kontroll program forwarding tábla bejegyzéseket tud elhelyezni az adatsík eszközeiben
    - ForCES-t nem fogadták el a nagy router gyártók!
  - RCP (Routing Control Platform)
    - BGP (Border Gateway Protocol) használata
    - flow bejegyzések elhelyezésére
    - meglévő routerekkel működik

Hogyan jutottunk az SDN-ig?

- kontroll-adat szeparálás
  - kontrollsík programozása
  - eszközszintű konfiguráció helyett → hálózat vezérlése

  - Ethane (elődje: SANE)
    - Stanford University, Clean Slate Project
    - centralizált logika
    - magas szintű hálózati policy-k leképzése
    - eszközszintű flow bejegyzésekre
    - OpenFlow közvetlen elődje!

Hogyan jutottunk az SDN-ig?

- MPLS
  - Cél
    - hálózati HW-ek egyszerűsítése
    - hálózati kontroll flexibilitásának javítása
    - (SDN ezt viszi tovább)
  - edge – core szeparáció
    - edge: komplex logika
    - core: hatékony csomagtovábbítás
    - (OpenFlow esetében újra előjött)

Hogyan jutottunk az SDN-ig?

- OpenFlow
  - kontroll-adat szeparálás +
  - hálózati eszköz általánosítása (absztrakció)
  - műveletek általánosítása (bizonyos mértékben)
  - új koncepció: hálózati operációs rendszer

  - SDN (koncepciók) egy népszerű realizációja
  - DE más realizációk is vannak
    - pl. láttuk, BGP-alapú megoldás
    - gyártó specifikus megoldások pl.: Cisco ONE platform, Juniper JunOS SDK

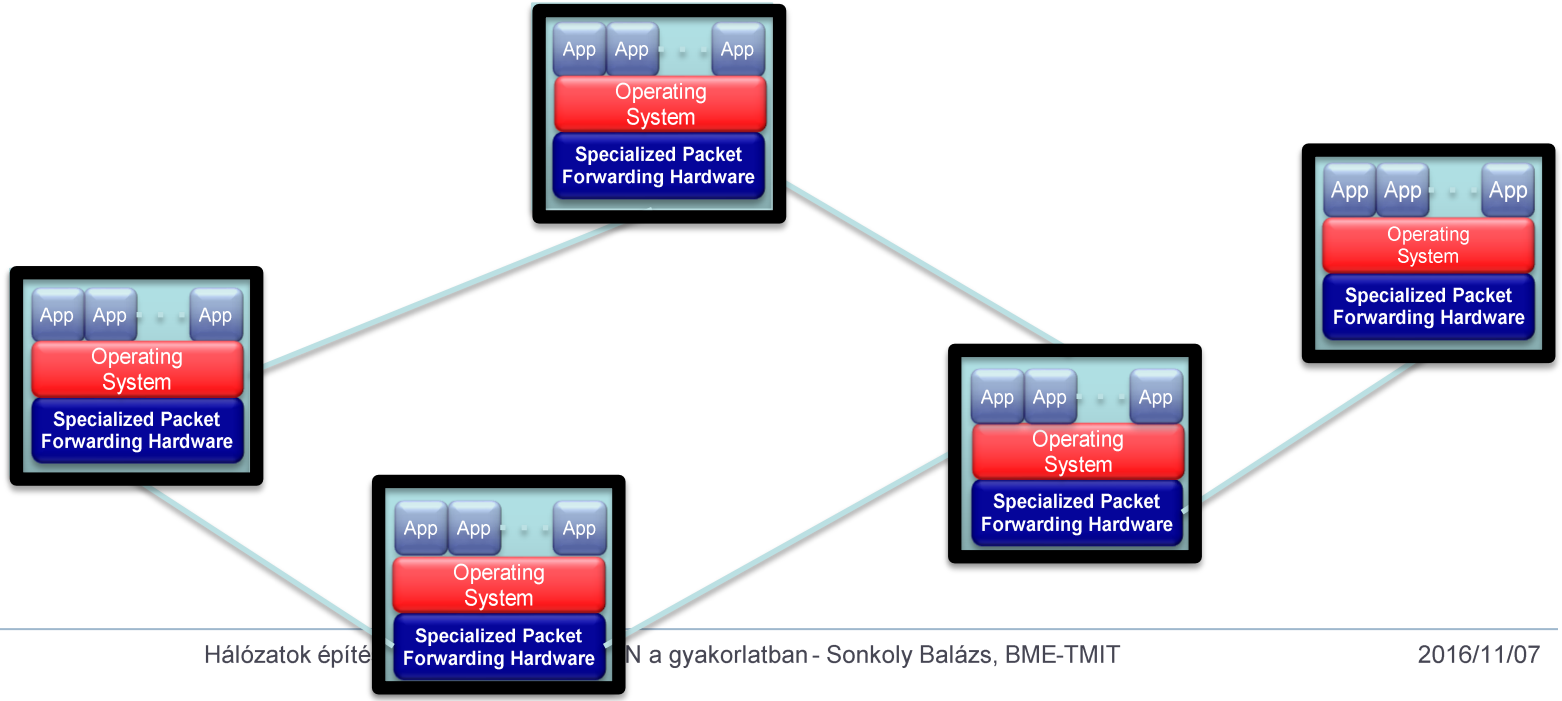
Hogyan jutottunk az SDN-ig?

- OpenFlow
  - Siker oka: sokan álltak mögé
  - Akadémiai szereplők
    - legnagyobb egyetemek (USA, EU)
  - Ipari szereplők
    - gyártók
      - NEC, HP, Cisco, Pronto, Brocade, Broadcom, Ericsson, IBM, …
    - felhő szolgáltatók
      - Amazon, Google, Microsoft, …
    - szolgáltatók / adatközpont üzemeltetők
      - Facebook, …
    - carriers
      - DT, Telecom Italia, Telefonica, NTT, …
  - ma már: szabványosító szervezetek
    - Open Networking Foundation
    - OpenDaylight initiative

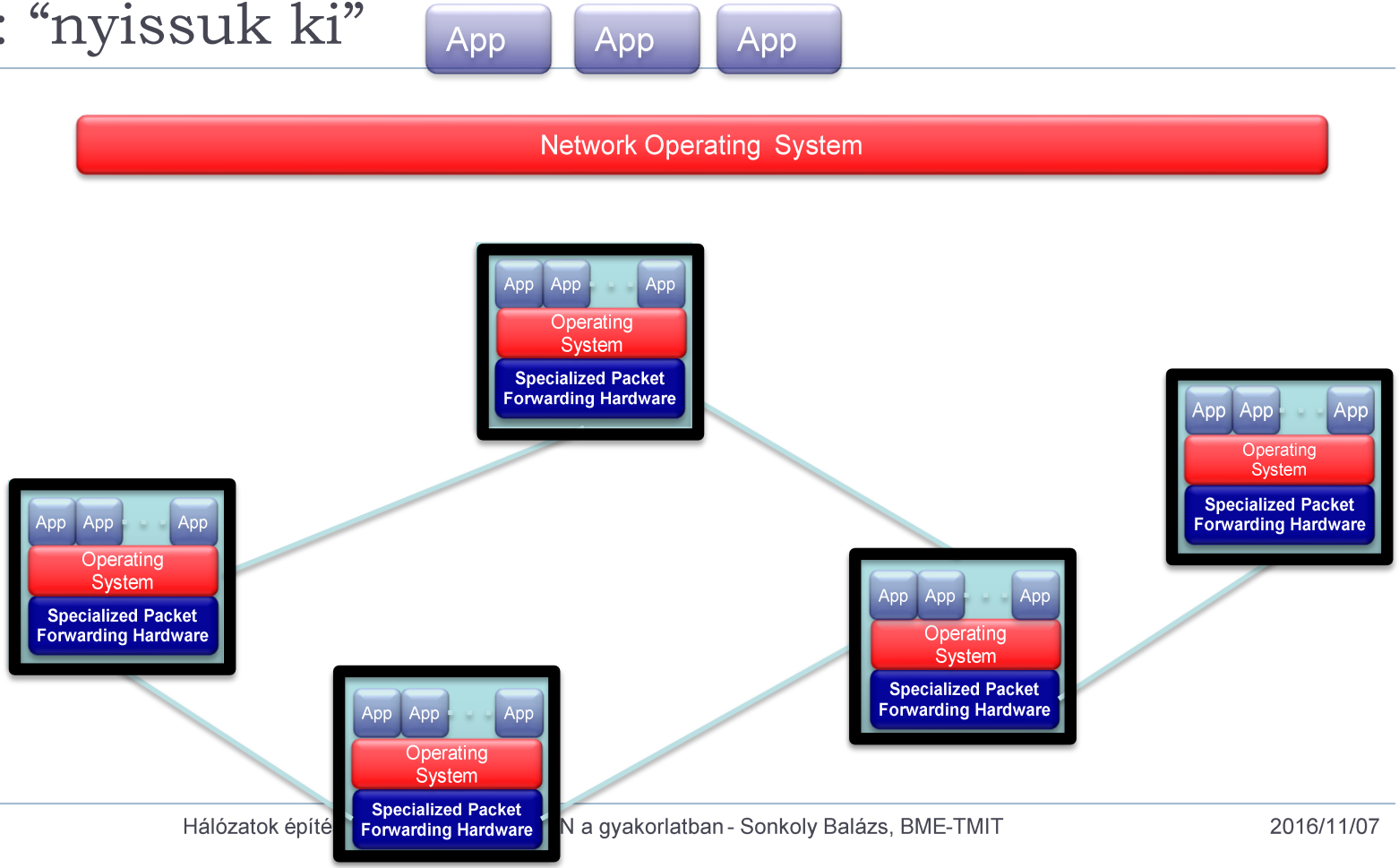
OpenFlow

Alapok

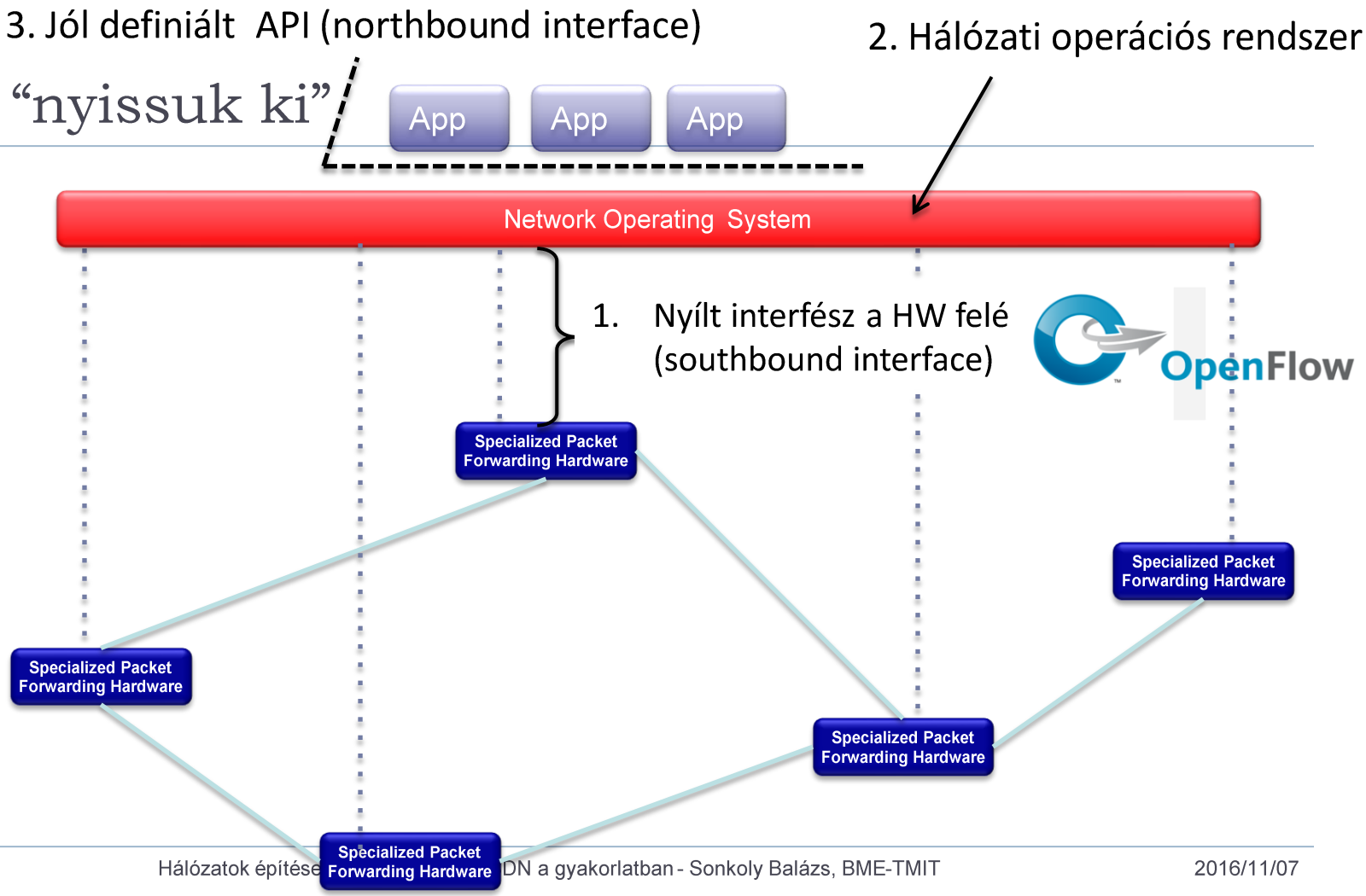
Internet ma: zárt infrastruktúra



SDN: “nyissuk ki”



SDN: “nyissuk ki”



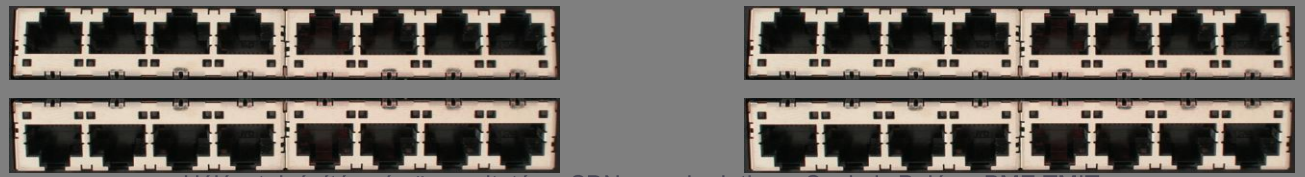
Mi is az az OpenFlow?

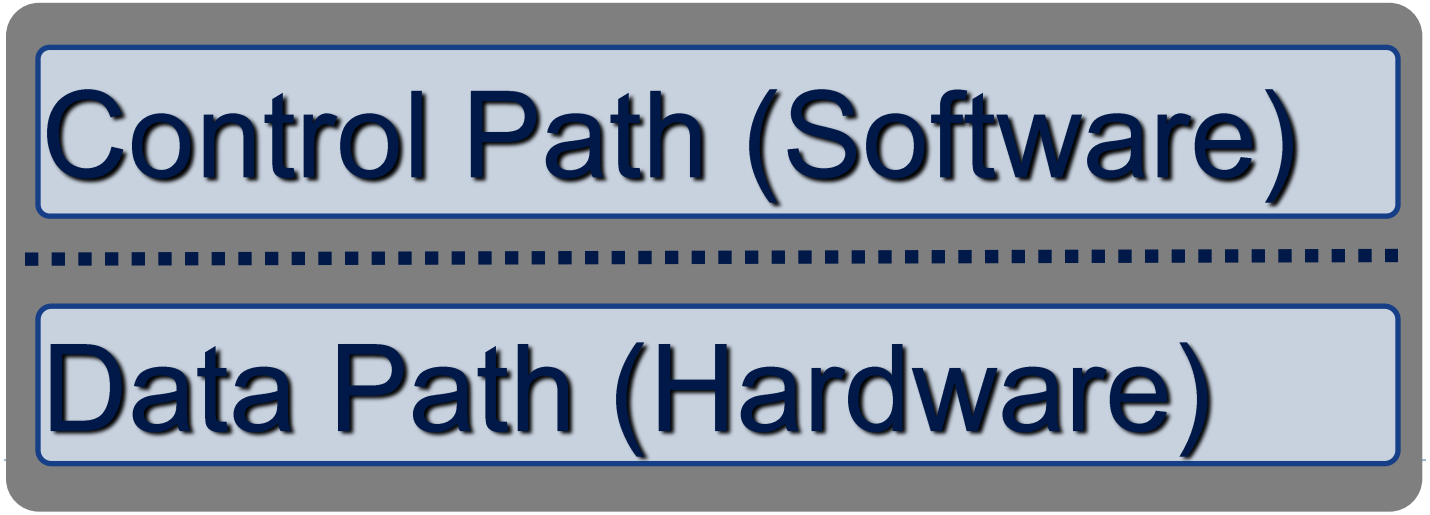
- OpenFlow egy API, interfész

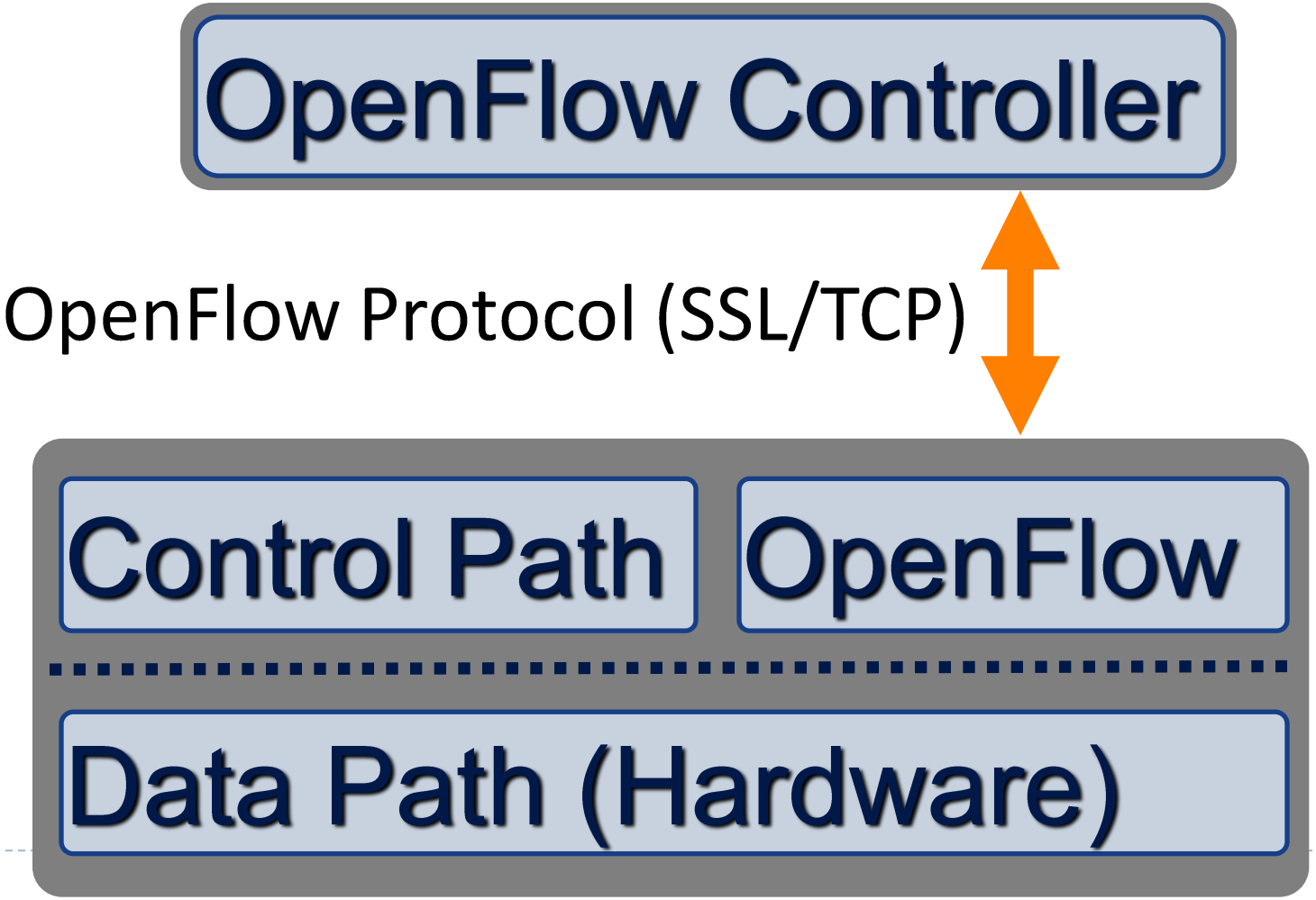
- Ezen keresztül kontrollálható a csomag-továbbítás (forwarding)
- olcsó HW-en is implementálható
- Üzemeltetett hálózat programozható lesz
  - nem csak konfigurálható!
- Egyszerűbb innováció
- (egyszerűbb üzemeltetés, új szolgáltatások bevezetése)

- Fő célok
  - Ne kelljenek speciális testbedek
  - Kísérleti megoldások **valós hálózaton**, **valós forgalom** mellett, **vonali sebesség**en

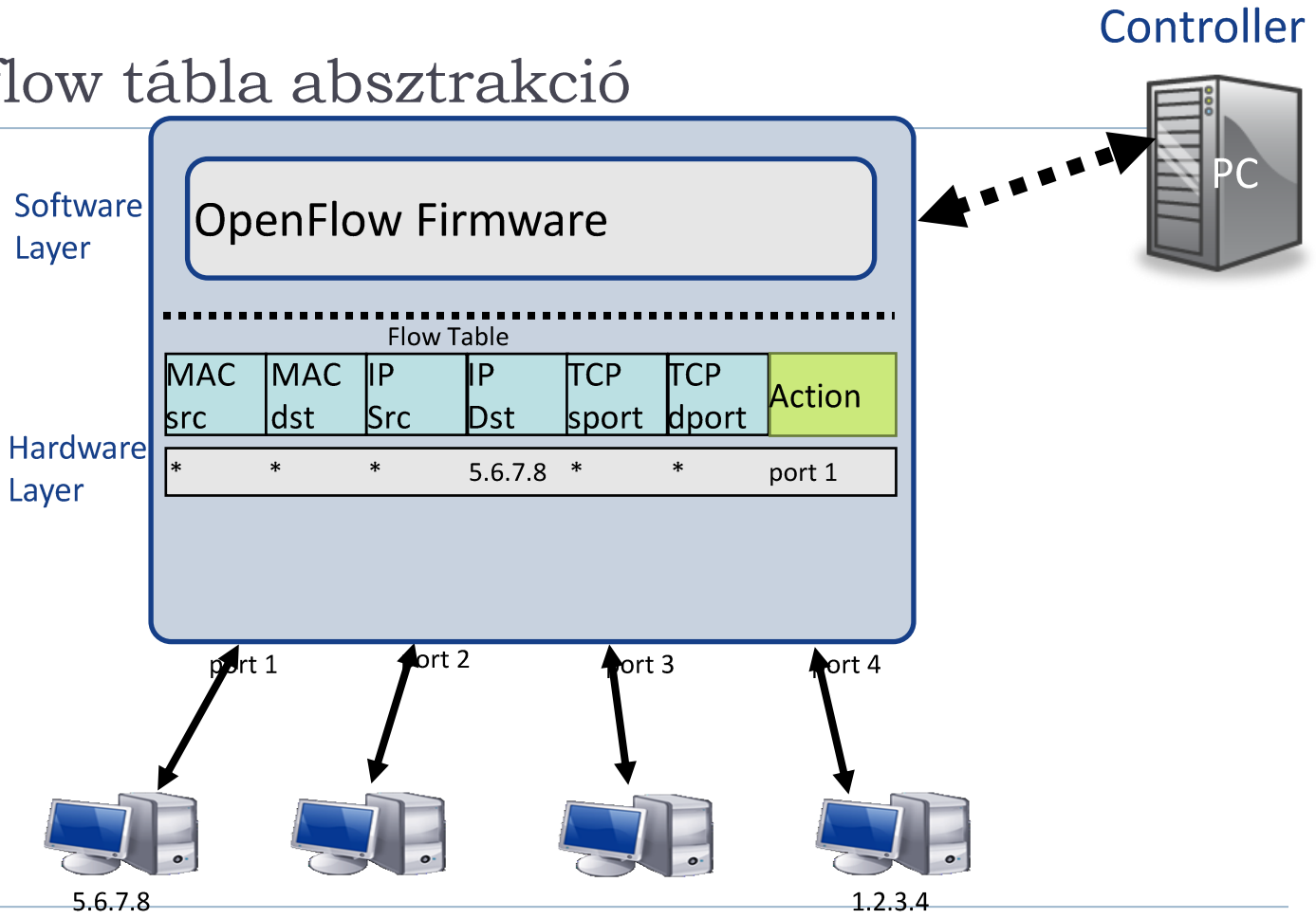
Ethernet Switch



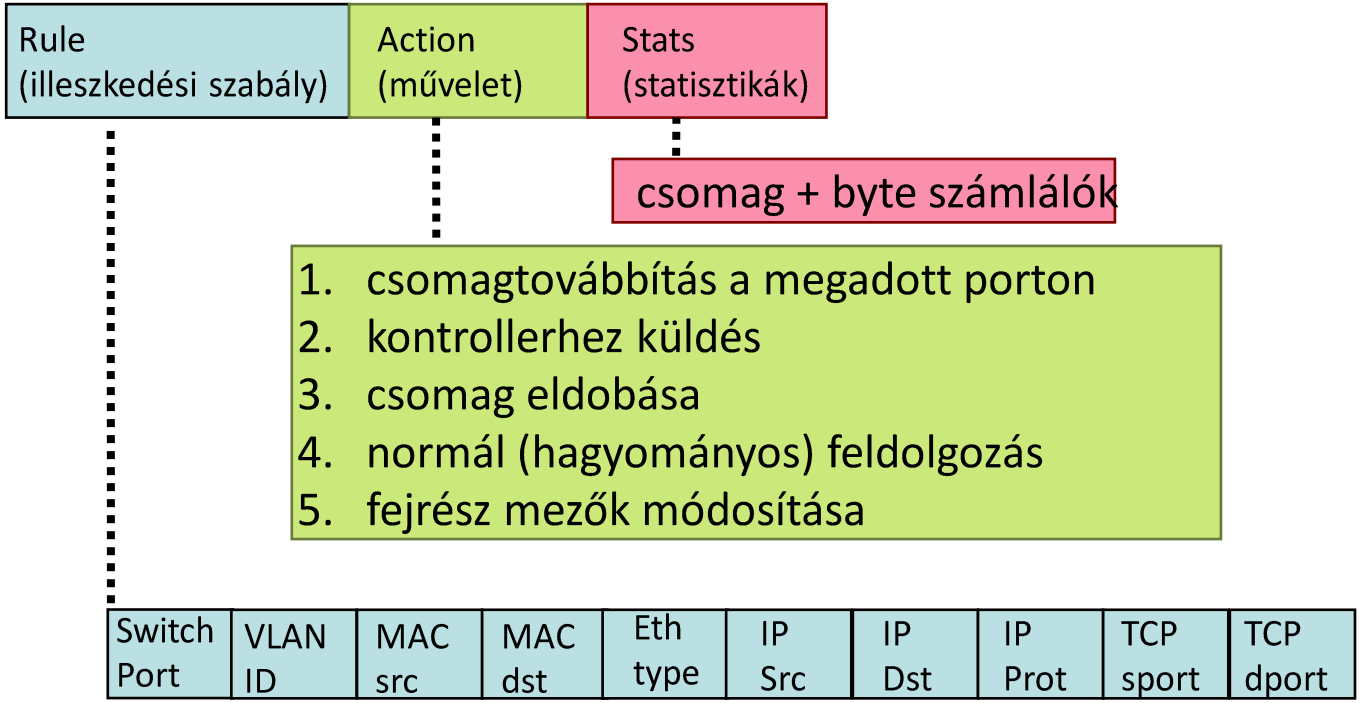

Control Path (Software)
Data Path (Hardware)


OpenFlow Controller
OpenFlow Protocol (SSL/TCP)
Control Path
OpenFlow
Data Path (Hardware)

OpenFlow flow tábla absztrakció



Flow tábla bejegyzések



\+ a nem szükséges mezők maszkolhatók (wildcard)

Flow tábla bejegyzések: példák

Switching (L2 kapcsolás)

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f:..	*	*	*	*	*	*	*	port6

Routing (L3 útvonalválasztás)

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	5.6.7.8	*	*	*	port6

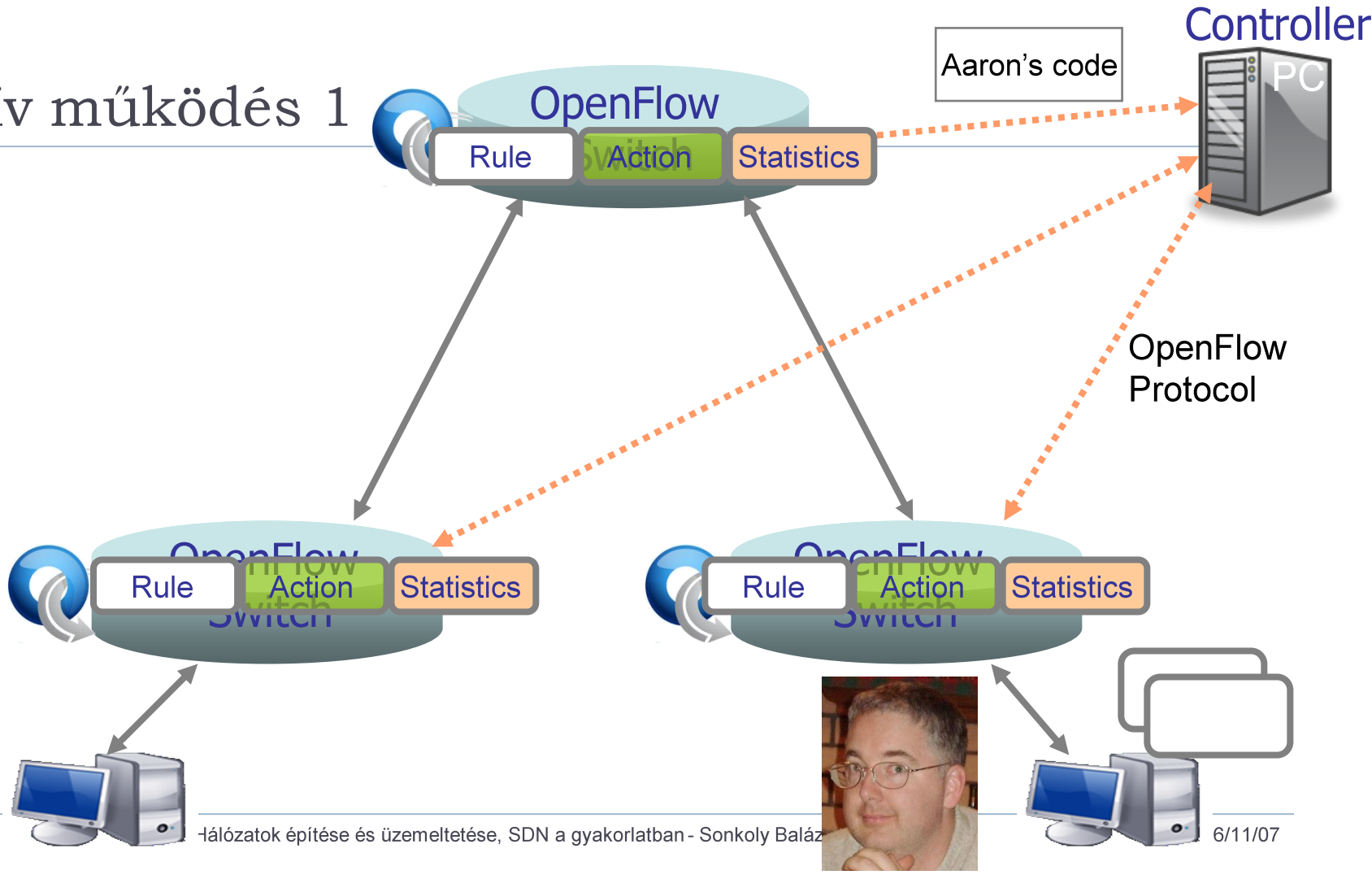
VLAN Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f..	*	vlan1	*	*	*	*	*	port6, port7, port9

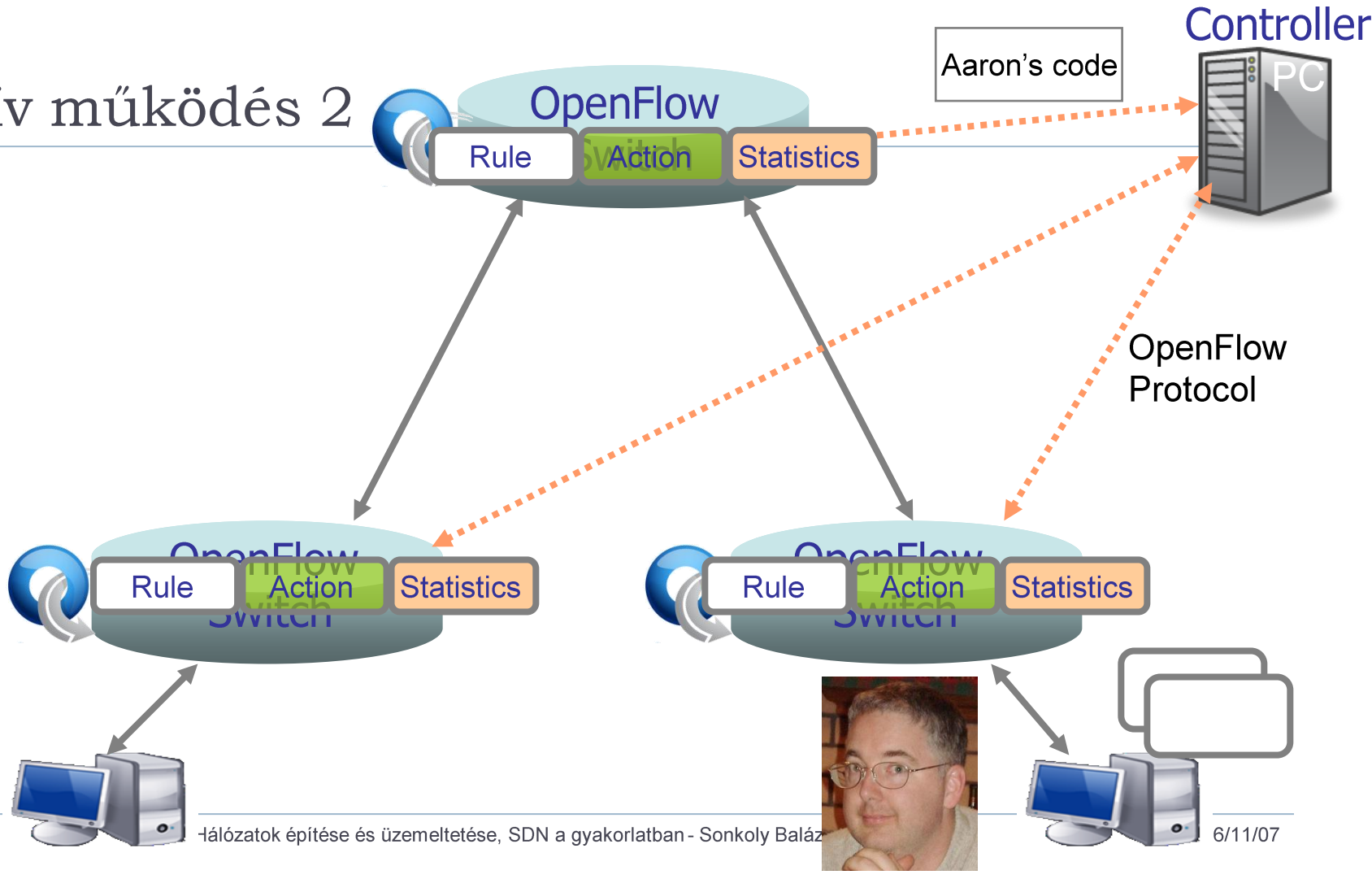
OpenFlow

Működés

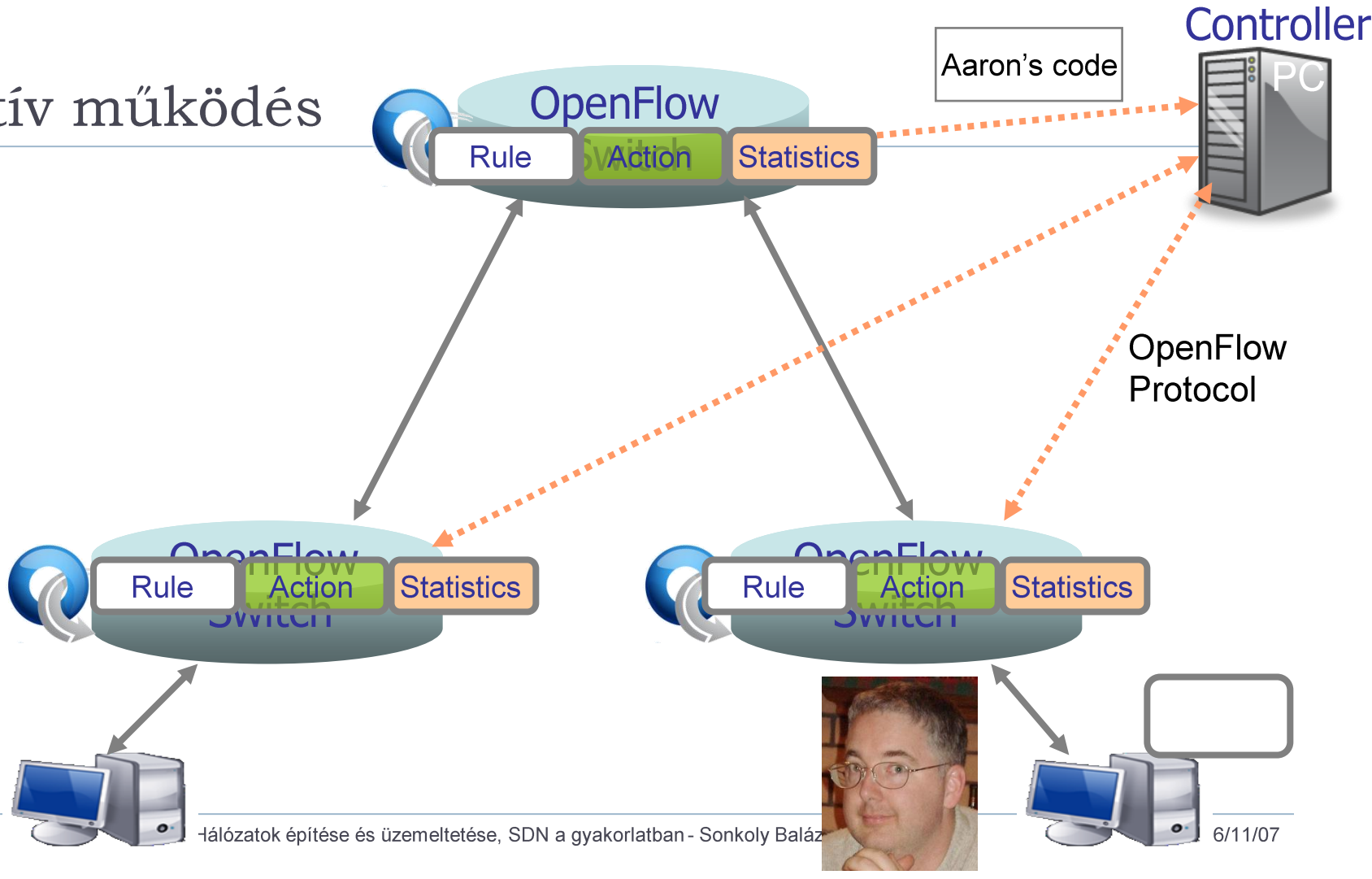
Reaktív működés 1



Reaktív működés 2



Proaktív működés



Új problémák & kihívások

- centralizált kontrollsík
  - skálázhatóság (→ ONOS, ODL)
  - megbízhatóság
- adatsík – kontrollsík szeparálás
  - késleltetés
  - különböző késleltetések
- out-of-band ↔ inband kontroll csatorna
- biztonság
- core-edge szeparáció
  - más elvárások
    - edge: intelligencia, gyors bővíthetőség, SW
    - core: egyszerű, gyors, hatékony, HW
  - OpenFlow a kettő között

OpenFlow evolúciója

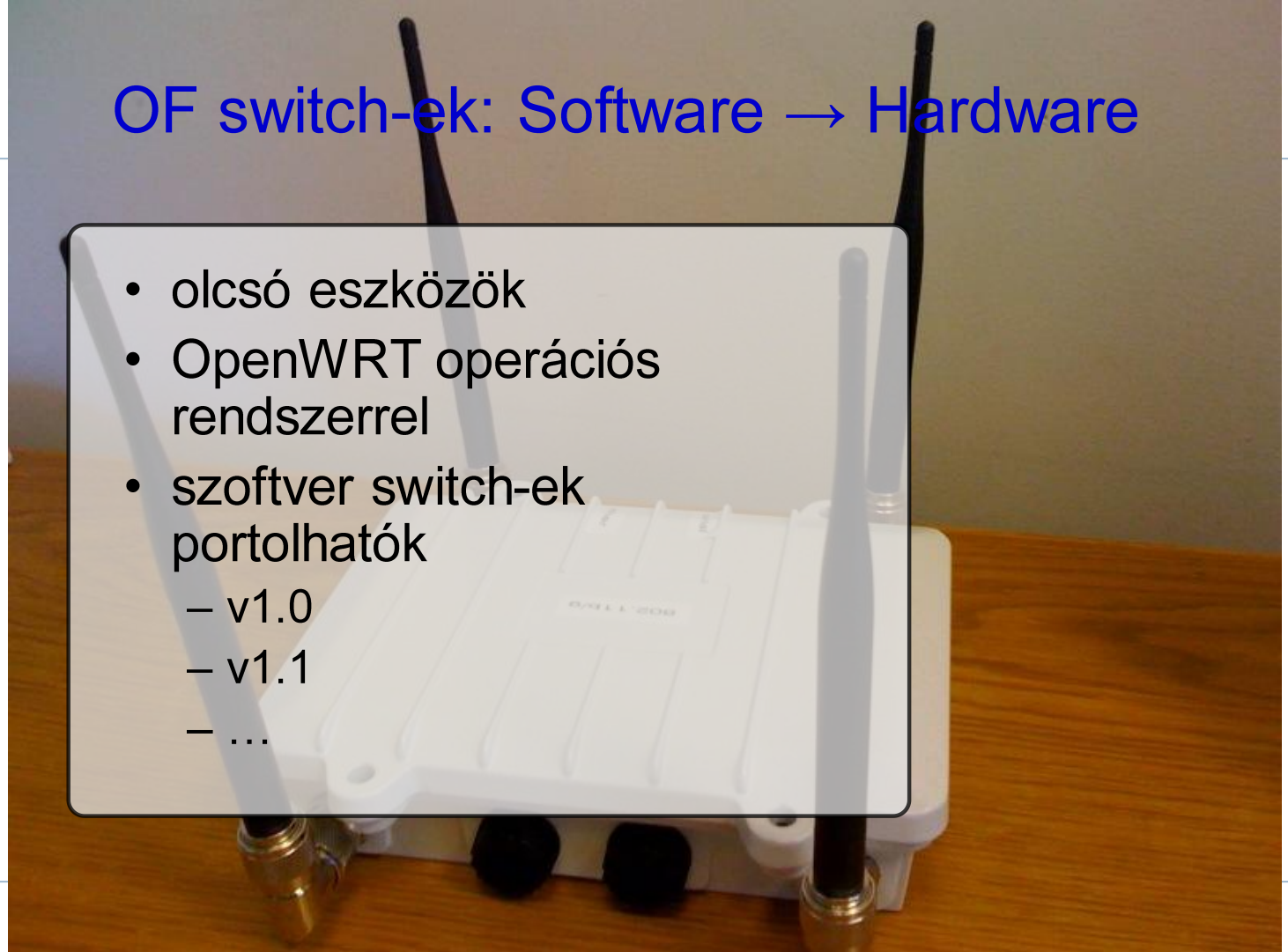
- **OF v1.0**
  - legelterjedtebb
  - HW-ek is támogatják
- OF v1.1
  - WAN kiterjesztések
  - több folyam tábla (pipeline)
- OF v1.2
  - IPv6
  - általánosított matching
- **OF v1.3**
  - már több gyártó eszköze támogatja (bizonyos halmazát)
  - (sokszor Open vSwitch alapon)
- OF v1.4, v1.5…
- OF v2+ : ?
  - általánosított illeszkedés vizsgálat és műveletek
  - általánosított “instruction set” hálózati műveletekhez

OpenFlow switch-ek

OF switch-ek: Software → Hardware

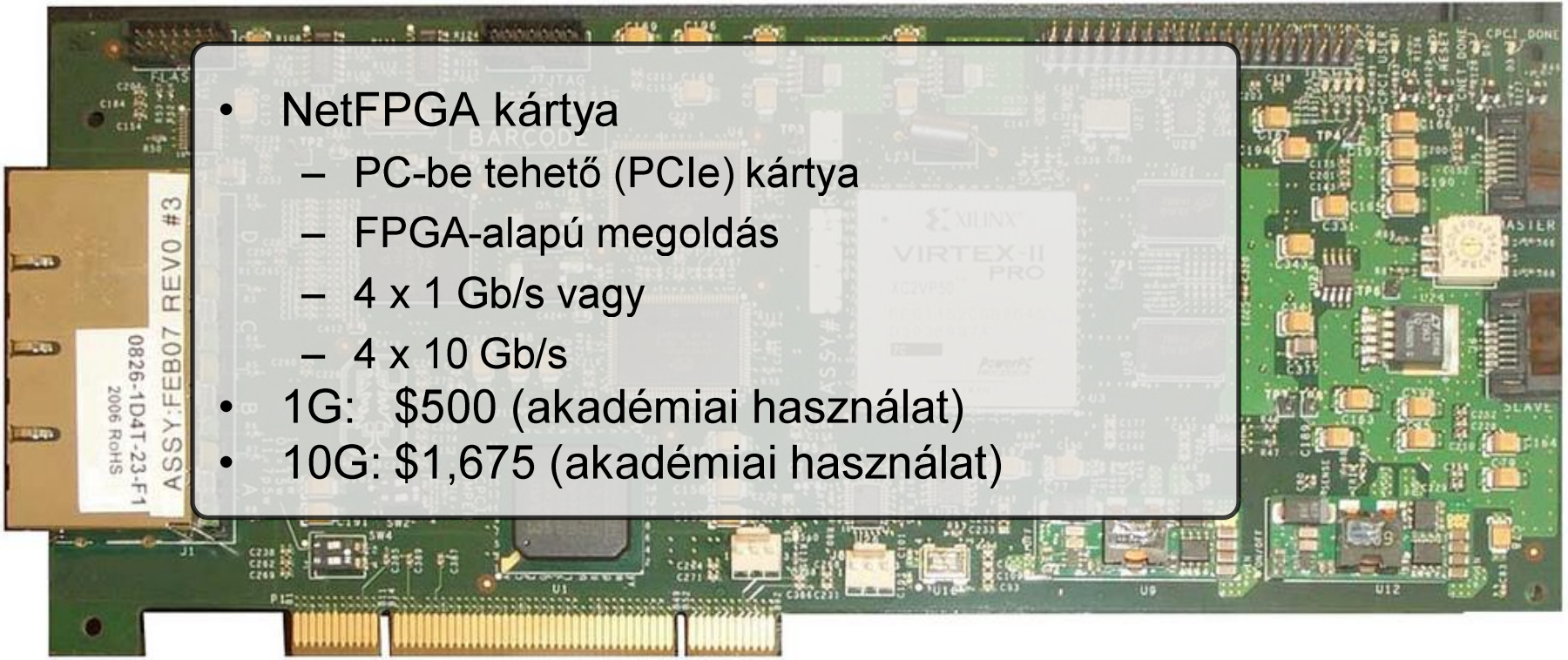
- Stanford Referencia implementáció v1.0
- Ericsson, CPqD implementáció v1.1, v1.2, v1.3, v1.4
  - Linux-alapú **szoftver switch (User Space)**
  - hasznos fejlesztéshez & teszteléshez
  - jó alap az egyéb implementációkhoz
- Open vSwitch
  - Linux-alapú **szoftver switch (Kernel Space)**
  - nem csak egy OF switch, virtuális gépek is használják (VirtualBox, XEN, OpenStack)
  - valós HW-ek firmware-e (SW rétege) sokszor Open vSwitch-re épül

OF switch-ek: Software → Hardware



- olcsó eszközök
- OpenWRT operációs rendszerrel
- szoftver switch-ek portolhatók
  - v1.0
  - v1.1
  - …

OF switch-ek: Software → Hardware



- NetFPGA kártya
  - PC-be tehető (PCIe) kártya
  - FPGA-alapú megoldás
  - 4 x 1 Gb/s vagy
  - 4 x 10 Gb/s
- 1G: $500 (akadémiai használat)
- 10G: $1,675 (akadémiai használat)

OF switch-ek: Software → Hardware

Prototype → Product

Core Router

Cisco Catalyst 6k (prototype)

Juniper MX-series (prototype)

HP ProCurve 5400 and others

Enterprise Campus Data Center

Cisco Catalyst 3750 (prototype)

Arista 7100 series (Q4 2010)

Pronto

NEC IP8800

Circuit Switch

Ciena CoreDirector

Wireless

WiMAX (NEC)

more to follow...

OpenFlow kontrollerek

OF kontrollerek

- Mára számos kontroller platform alakult ki
- programozás
  - különböző szoftver környezetben
  - különböző programozási nyelveken
- különböző célok
- különböző teljesítmény

NOX

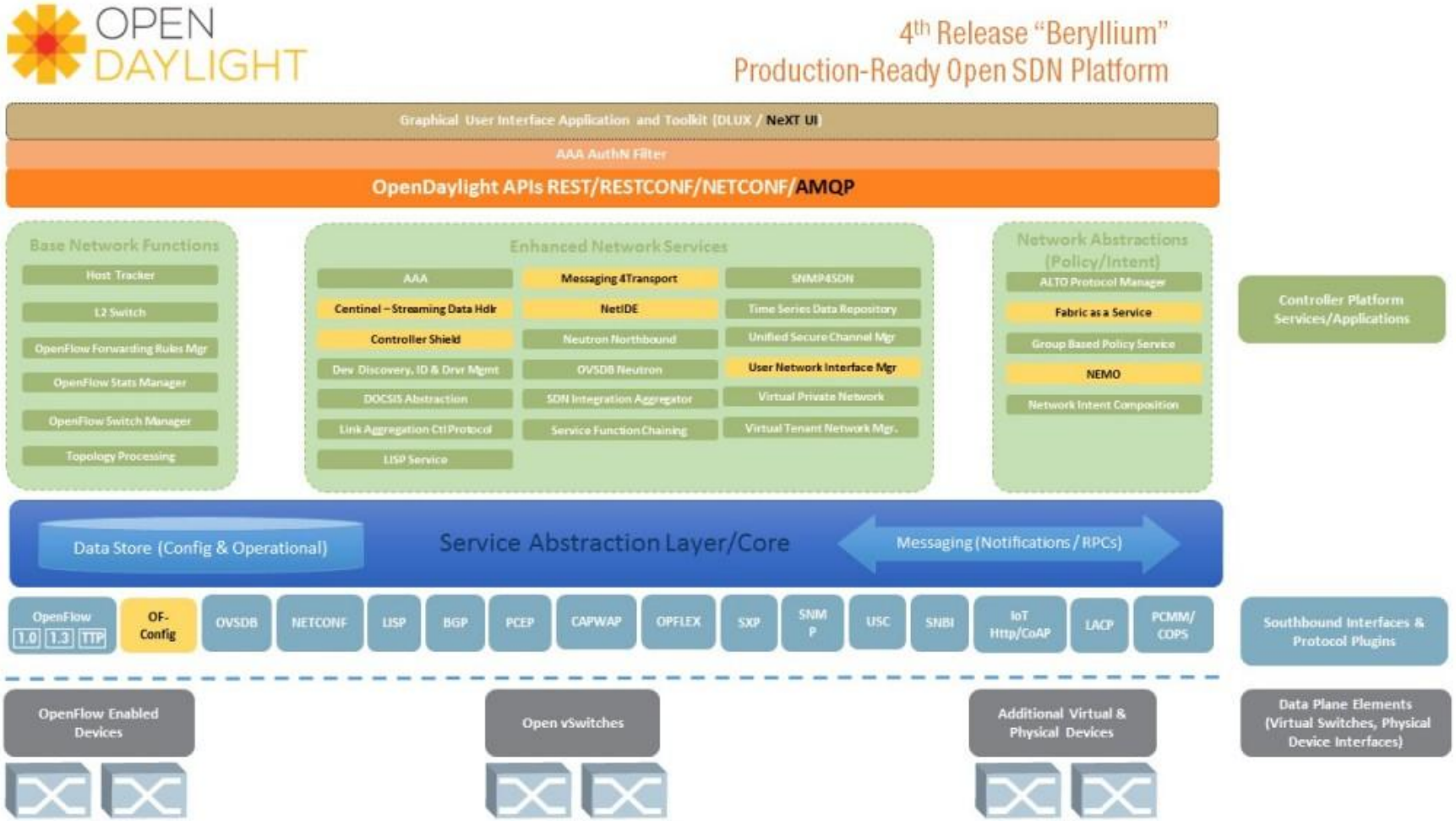
- Egyik legelső kontroller
- célok
  - hatékony működés
  - jó skálázhatóság
  - hálózati operációs rendszer
- C++ alapú
- sima PC-n: több 10 ezer új folyam kezelése másodpercenként
- jól definiált programozói interfész a hálózathoz
- egyszerűbbé válik különböző gyártók eszközeinek, együttes, centralizált vezérlése

POX

- Python nyelven implementált OpenFlow kontroller
- letisztult, egyszerű programozási környezet
- kontroller alkalmazások Python nyelvű fejlesztéséhez
- gyors prototípus implementálás
- oktatási célok
- hátránya: jelenleg csak az 1.0-ás OpenFlow verziót támogatja
- számos “beépített” alkalmazás elérhető
- POX API-n keresztül nagyon sok hasznos funkció elérhető és felhasználható saját alkalmazások implementálása során

Termékek

- Floodlight
  - Big Switch Networks terméke
  - Java alapú
  - Apache licensz
  - northbound API
- OpenDaylight, ONOS
  - “ipari” SDN platformok


OPEN DAYLIGHT
4th Release "Beryllium"
Production-Ready Open SDN Platform
Graphical User Interface Application and Toolkit (DLUX / NeXT UI)
AAA AuthN Filter
OpenDaylight APIs REST/RESTCONF/NETCONF/AMQP
Base Network Functions
Host Tracker
L2 Switch
OpenFlow Forwarding Rules Mgr
OpenFlow Stats Manager
OpenFlow Switch Manager
Topology Processing
Enhanced Network Services
AAA
Messaging4Transport
SNMP4SDN
Centinel – Streaming Data Hdlr
NetIDE
Time Series Data Repository
Controller Shield
Neutron Northbound
Unified Secure Channel Mgr
Dev. Discovery, ID & Drvr Mgmt
OVSDB Neutron
User Network Interface Mgr
DOCSIS Abstraction
SDN Integration Aggregator
Virtual Private Network
Link Aggregation Ctl Protocol
Service Function Chaining
Virtual Tenant Network Mgr.
LISP Service
Network Abstractions (Policy/Intent)
ALTO Protocol Manager
Fabric as a Service
Group Based Policy Service
NEMO
Network Intent Composition
Controller Platform Services/Applications
Data Store (Config & Operational)
Service Abstraction Layer/Core
Messaging (Notifications / RPCs)
OpenFlow 1.0 1.3 TTP
OF-Config
OVSDB
NETCONF
LISP
BGP
PCEP
CAPWAP
OPFLEX
SXP
SNMP
USC
SNBI
IoT Http/CoAP
LACP
PCMM/COPS
Southbound Interfaces & Protocol Plugins
OpenFlow Enabled Devices
Open vSwitches
Additional Virtual & Physical Devices
Data Plane Elements (Virtual Switches, Physical Device Interfaces)

Egyéb megoldások

- Ryu
  - Python alapú
- Trema
  - Ruby, C
- Frenetic, Nettle
  - deklaratív nyelvek (Haskell, OCaml)
- …

Hálózati Funkciók Virtualizálása

NFV, SFC

Middleboxok

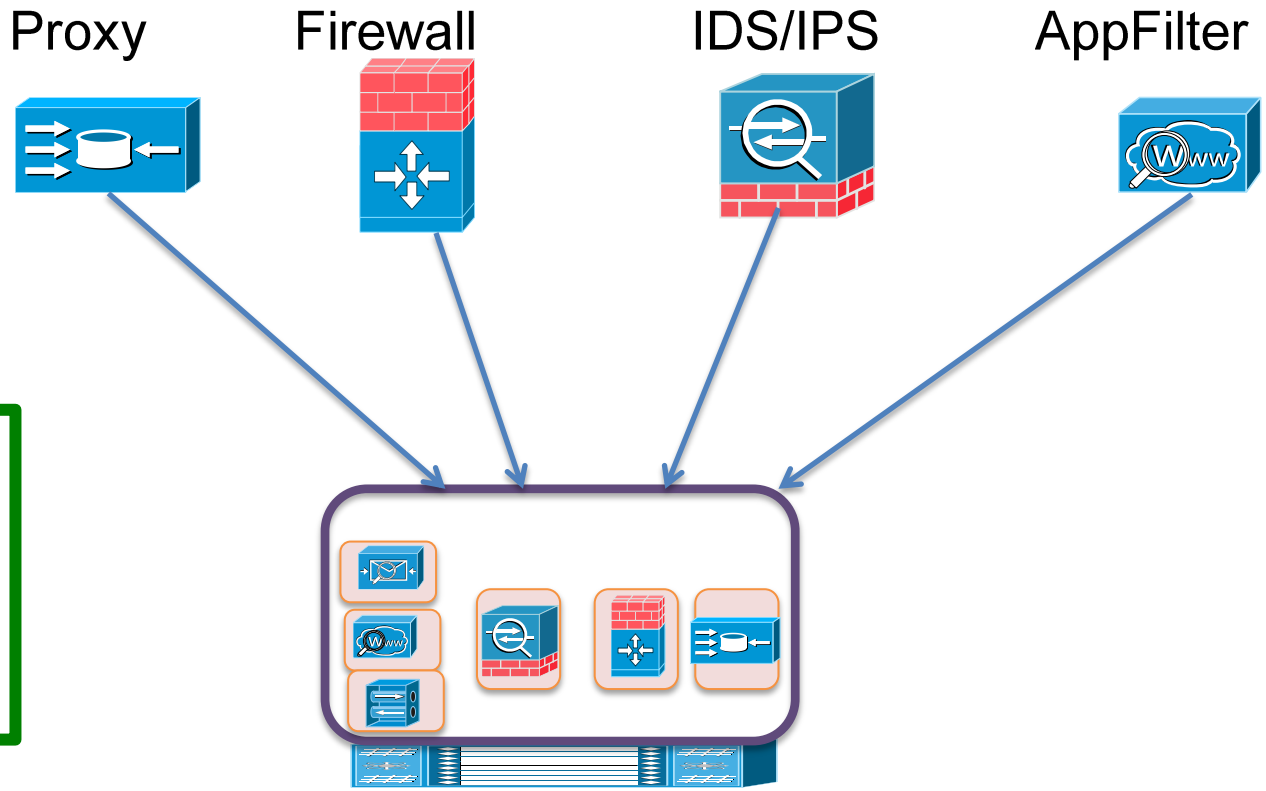
- Mi az a middlebox?
  - minden forgalom/csomag processzáló eszköz, ami nem switch vagy router
  - speciális hálózati funkció
  - pl. NAT, tűzfal, IDS/IPS, DPI, load balancer
- Mennyi van belőlük a hálózatban?
- Felmérés:
  - nagyvállalati hálózati környezet
  - felhasználók >80K
  - telephely n*10

Type of appliance	*Number*
Firewalls	166
NIDS	127
Media gateways	110
Load balancers	67
Proxies	66
VPN gateways	45
WAN Optimizers	44
Voice gateways	11
Total Middleboxes	***636***
Total routers	***~900***

Middleboxok ➡ NFV

Implementáció ma:
- önálló egység
- speciális HW berendezés vagy
- switch/router + extra funkció

Tendencia:
- HW → SW
- általános célú HW-en
- SW komponensek
- NFV: Network Function Virtualization

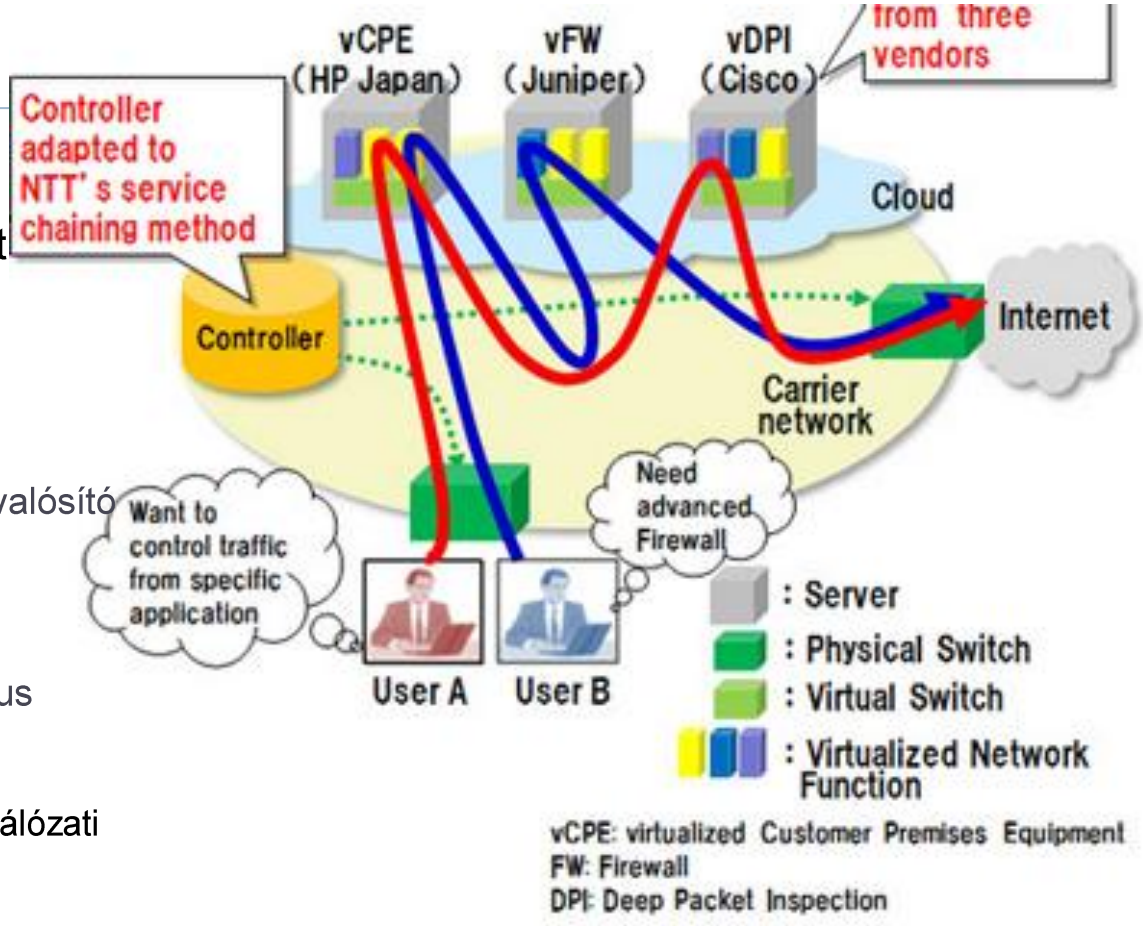


NFV

- Network Function Virtualization
  - vissza az adatsík programozásához
  - aktív hálózatok (lásd korábban)
    - egyéges végrehajtási környezet az adatsík csomópontjaiban (EE, execution environment)
    - akkor nem volt meg a megfelelő HW környezet
  - általános célú szerver HW-ek (pl. x86)
    - hatalmas fejlődés
    - realitás a hatékony csomagfeldolgozás SW alapon!
    - pl. Intel DPDK, Netmap
- NFV-k futtatása
  - cloudban
  - vagy saját adatközpontokban
  - micro/pico adatközpont pl. egy bázisállomásban

SFC

- Service Function Chaining
- Nem új koncepció
- SDN előretörésével fókuszba került
- Tipikus szolgáltatás
  - hálózati funkciók végrehajtása
  - adott sorrendben
  - adott forgalomra
  - csomagok irányítása a funkciót megvalósító blokkok között
- Absztrakció
  - service chain vagy service graph
  - magas szintű szolgáltatások generikus leírására
    - milyen típusú forgalom/felhasználó
    - milyen elemi szolgáltatások (vagy hálózati funkciók)
    - milyen sorrendben



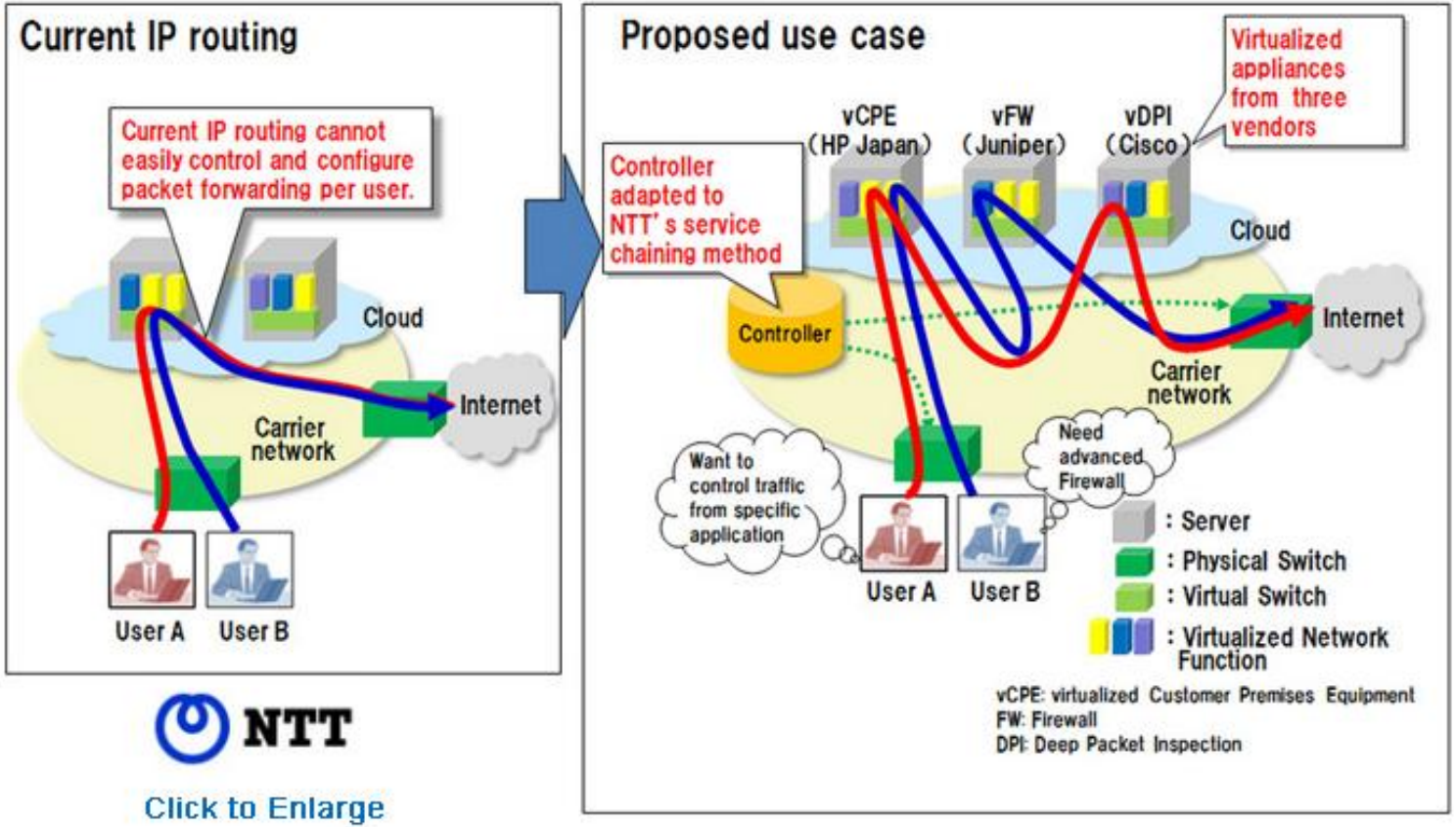
SFC

- Mi a probléma a mai szolgáltatás nyújtással?
  - komoly korlátok
  - erős kötődés a fizikai topológiához
  - speciális képességű, drága middlebox hardverekhez
  - és azok fizikai elhelyezkedéséhez
  - NEM dinamikus
  - NEM flexibilis
  - NEM ad lehetőséget új szolgáltatások gyors bevezetésére
  - NEM jól skálázható
  - NEM garantálható az erőforrások optimális kihasználása
- Következmény
  - service chainek konfigurálása, elhelyezése, üzemeltetése komplex feladat
  - sokszor manuális beavatkozást igényel

Megoldás: SDN+NFV(+cloud)

- SDN
  - forgalom flexibilis irányítása
  - megfelelő funkciók között
  - megfelelő sorrendben
- NFV
  - egy szoftver
  - ami „tetszőleges” környezetben futhat
    - pl. virtuális gép a cloudban
    - Docker konténerben futtatott processz
  - könnyű áthelyezni
  - igény szerint indítani
  - adott szempontok alapján választott fizikai helyen

- Rapid service provision based on user selection of network function
- Verification of service chaining method in multivendor environment



NTT

Click to Enlarge

Egyszerű példák

Live demo

Emulált környezet

- VM: Mininet+Netkit
- Eszközök
  - Mininet
  - POX
  - Wireshark
- Alkalmazási példák
  - hub
  - l2 switch

Hub alkalmazás POX kontrollerben

```
def _handle_ConnectionUp (event):
  """
  Be a proactive hub by telling every connected switch to flood all packets
  """
  msg = of.ofp_flow_mod()
  msg.actions.append(of.ofp_action_output(port = of.OFPP_FLOOD))
  event.connection.send(msg)
  log.info("Hubifying %s", dpidToStr(event.dpid))


def _handle_PacketIn (event):
  """
  Be a reactive hub by flooding every incoming packet
  """
  msg = of.ofp_packet_out()
  msg.data = event.ofp
  msg.actions.append(of.ofp_action_output(port = of.OFPP_FLOOD))
  event.connection.send(msg)


def launch (reactive = False):
  if reactive:
    core.openflow.addListenerByName("PacketIn", _handle_PacketIn)
    log.info("Reactive hub running.")
  else:
    core.openflow.addListenerByName("ConnectionUp", _handle_ConnectionUp)
    log.info("Proactive hub running.")
```

proaktív hub:
- ConnectionUp esemény kezelése
- (switch csatlakozása)
- flow bejegyzés összeállítása & leküldése

reaktív hub:
- PacketIn esemény kezelése
- (csomag sw→ ctrl)
- flow bejegyzés

kétféle üzemmód:
- reaktív
- proaktív