Introduction to IPv6
IPv4 to IPv6 TRANSFORMATION

phd. student Nikolay Milovanov
CCIE SP #20094
http://niau.org

Нов български университет

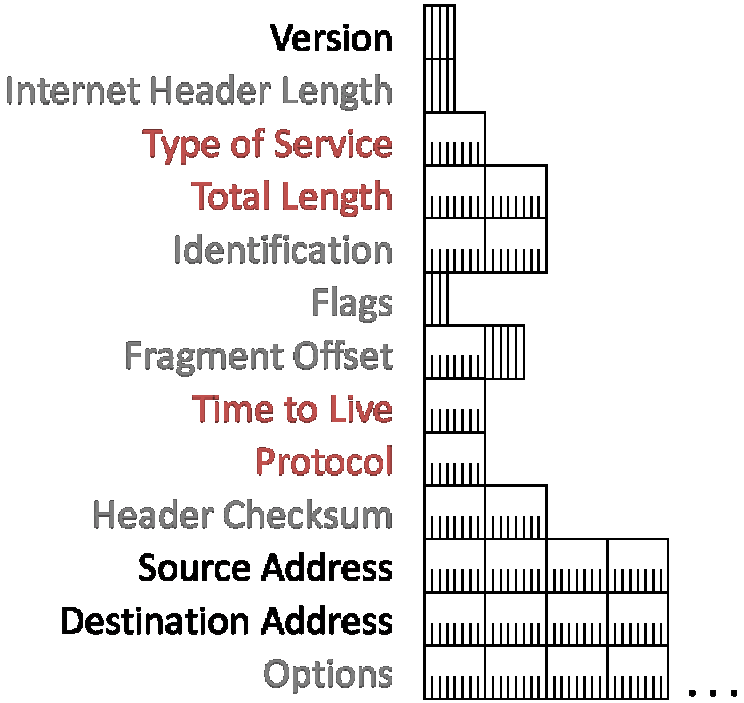
Agenda

- Introduction to IPv6
  - Slides are taken by Bianor's presentation on gogo6.com
- IPv6 pros and cons
- 4TO6TRANS concepts and objectives
  - Solution Architecture
  - Framework Components
  - Transformation Automation
- 4TO6TRANS status
  - Project Support
  - Project Funding
  - Project popularization

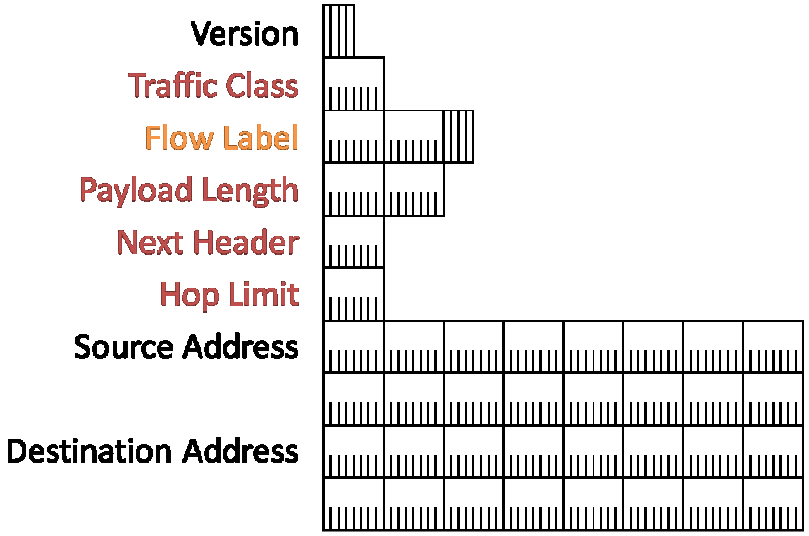
Introduction to IPv6

IPv4 and IPv6 Headers

IPv4 Header



IPv6 Header



IPv6 Address Syntax

- IPv6 address in binary form:

  00100000000000010000000000000000011010001010110000000000000000000000000000000011110001111000010101011100110100001001100001111

- The 128-bit address is divided along 16-bit boundaries:

  0010000000000001 0000000000000000
  0011010001010110 0000000000000000 0000000000000000
  1111000111100000 1010101111001101 0000100110000111

- Each 16-bit block is converted to hexadecimal and delimited with colons:

  2001:0000:3456:0000:0000:F1E0:ABCD:0987

Compressing zeros

- Leading zeros within each 16-bit block can be compressed:
  2001:0000:3456:0000:0000:F1E0:ABCD:0987 becomes
  2001:0:3456:0:0:F1E0:ABCD:987
- Successive fields of 0 can be represented as "::"
  - 2001:0:3456:0:0:F1E0:ABCD:987 becomes
    2001:0:3456::F1E0:ABCD:987
  - FE80:0:0:0:2AA:FF:FE9A:4CA2 becomes FE80::2AA:FF:FE9A:4CA2
  - FF02:0:0:0:0:0:0:2 becomes FF02::2
  - 0:0:0:0:0:0:0:1 becomes ::1
  - 0:0:0:0:0:0:0:0 becomes ::

  A double colon is allowed only once in an IPv6 address!
  - 2001:0:3456:0:0:F1E0:ABCD:987 does not become
    2001::3456::F1E0:ABCD:987

IPv6 Address Prefixes

- Indicates the bits that have fixed values or are the bits of the subnet prefix.
- Also known as Classless Inter-Domain Routing (CIDR) notation for IPv4.
- An IPv6 prefix is written in address/prefix-length notation.
  - 2001:DB8:0:2F3B::/64 is a subnet prefix for a subnet
  - 2001:DB8::/48 is an address prefix for a summarized route
  - FF00::/8 is an address prefix for an address range
- IPv4 uses a dotted decimal representation of the network prefix known as the subnet mask. A subnet mask is not used for IPv6.

Literal IPv6 addresses in URIs

- In a URI the IPv6 address is enclosed in brackets
- Examples:
  - https://[fd00::a00:cd24]/
  - https://[fd00::a00:cd24]:443/
  - https://[fd00::0000:0000:0000:0000:0000:0a00:cd24]:443/

IPv6 supported browsers

- MS IE6 doesn’t support IPv6
- MS IE7 supports IPv6
- Safari supports IPv6
- Mozilla Firefox supports IPv6
- Google Chrome Supports IPv6

Types of IPv6 Addresses

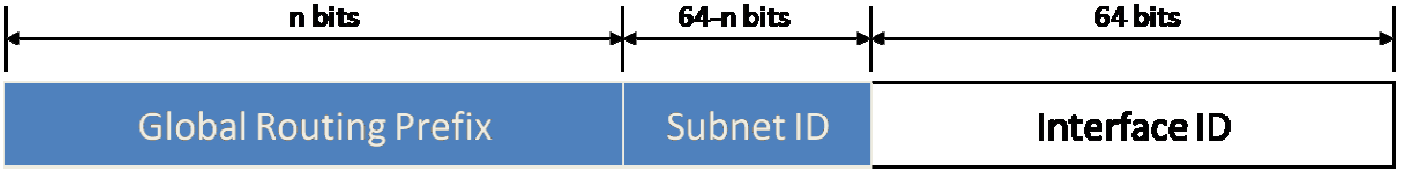
- Unicast
  - Identifies a single interface
  - Delivery to single interface
- Anycast
  - Identifies a set of interfaces that typically belong to different nodes
  - Delivery to a single “nearest” interface in the set
- Multicast
  - Identifies a set of interfaces
  - Delivery to all interfaces in the set
- No more broadcast addresses

Unicast IPv6 addresses

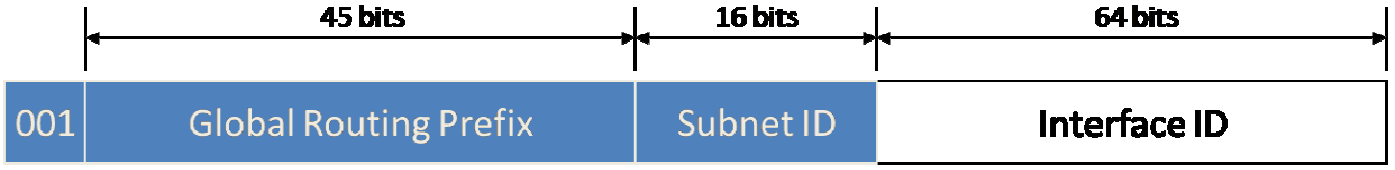
- Global addresses
- Link-local addresses
- Site-local addresses
- Unique local addresses (ULA’s)
- IPv4 mapped IPv6 addresses
- Special unicast addresses

Global unicast addresses

- Address scope is the whole IPv6 Internet
- Equivalent to public IPv4 addresses
- Defined in RFC 3587
- 2001:DB8::/32 – documentation-only prefix

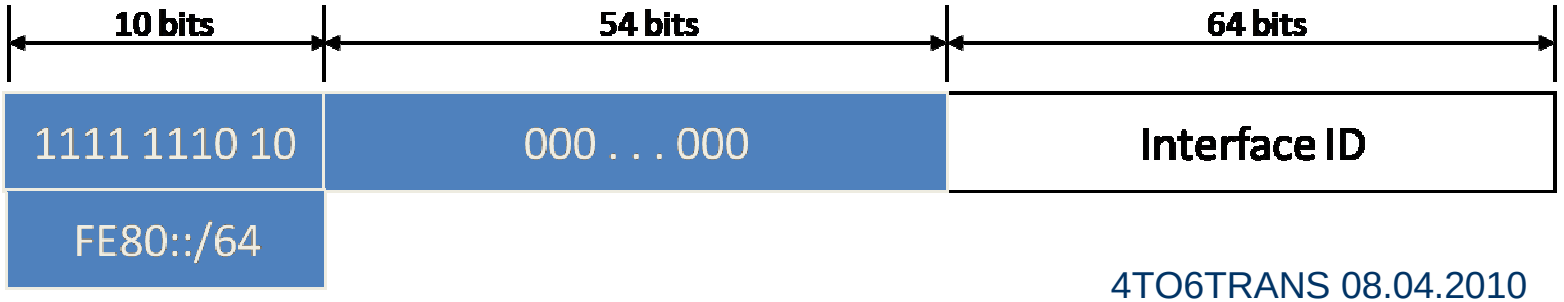


- Currently the following format under the 2000::/3 prefix is delegated by the IANA and recommended in RFC 3177:



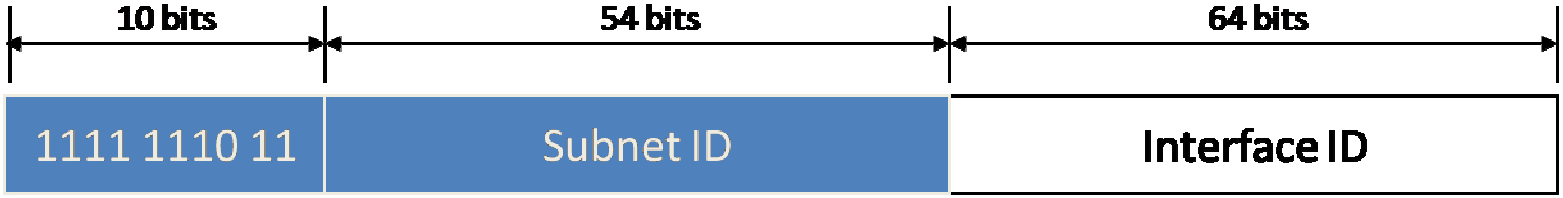
Link-local addresses

- Automatically configured on an interface
- Address scope is limited to the local link
- Usage
  - Single subnet, routerless configurations
  - Neighbor Discovery processes
- Router Discovery processes
- Stateless Autoconfiguration process
- Zone ID is required to identify a specific link



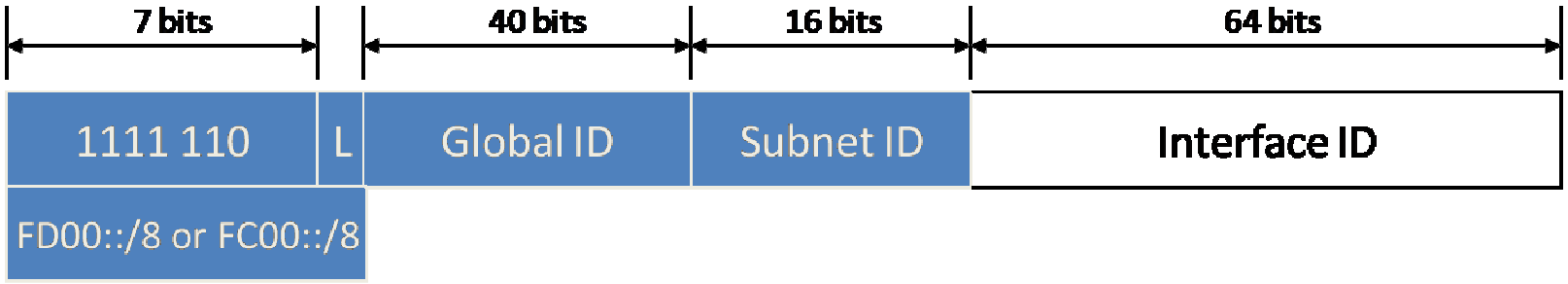
Site-local addresses

- Address scope is a single site
  - Equivalent to private IPv4 addresses
- Zone ID are required to identify a specific site
- Site-local unicast addresses are deprecated (RFC 3879)
- Replaced by unique-local unicast addresses
- Starts with FEC, FED, FEE, FEF
- New implementations must treat them as Global Unicast



Unique local addresses

- Globally unique and are intended for local communications
- Not routable on global Internet, routable within organization
- Replaced the site-local addresses
- Global scope, no zone ID required
- Defined in RFC 4193



IPv4 mapped IPv6 address

- Used by IPv6 only application to be able to deal with IPv4 requests
- Requires dual stack configured on the host
- Defined in RFC 4291
- Example:
  - IPv4-mapped IPv6 address for the IPv4 address 192.168.0.189 is: 0:0:0:0:0:FFFF: 192.168.0.189 = ::FFFF:c0a8:bd

Special unicast addresses

- Loopback unicast address – 0:0:0:0:0:0:0:1 = ::1
  - Similar to IPv4 address 127.0.0.1
  - Used by a node to send an IPv6 packet to itself
  - Should not be assigned to any physical interface

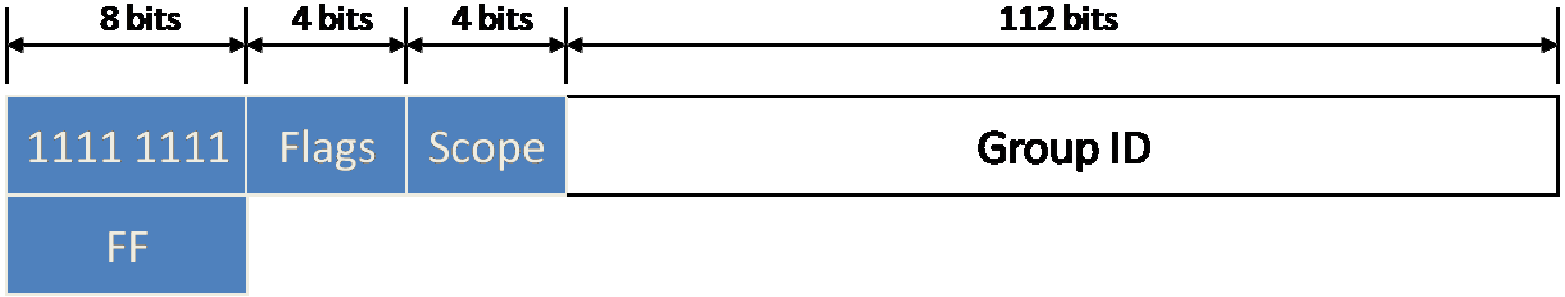
- Unspecified unicast address – 0:0:0:0:0:0:0:0 = ::
  - Similar to the IPv4 address 0.0.0.0
  - Indicates the absence of an address

Anycast IPv6 addresses

- Syntactically the same as a interface unicast address on the link with the interface identifier set to zero
- A packet sent to an anycast address is delivered to one of the interfaces identified by that address - the "nearest" one, according to the routing protocol's measure
- Should be assigned to IPv6 routers only
- Defined in RFC 4291

Multicast IPv6 addresses

- An identifier for a set of interfaces (typically on different nodes)
- Defined in RFC 4291
- Some reserved multicast addresses:
  - FF02::1 (link-local scope, all nodes on the link)
  - FF02::2 (link-local scope, all routers on the link)
  - FF05::2 (site-local scope, all routers in the site)
  - FF02:0:0:0:0:1:FFXX:XXXX (Solicited-node multicast address)

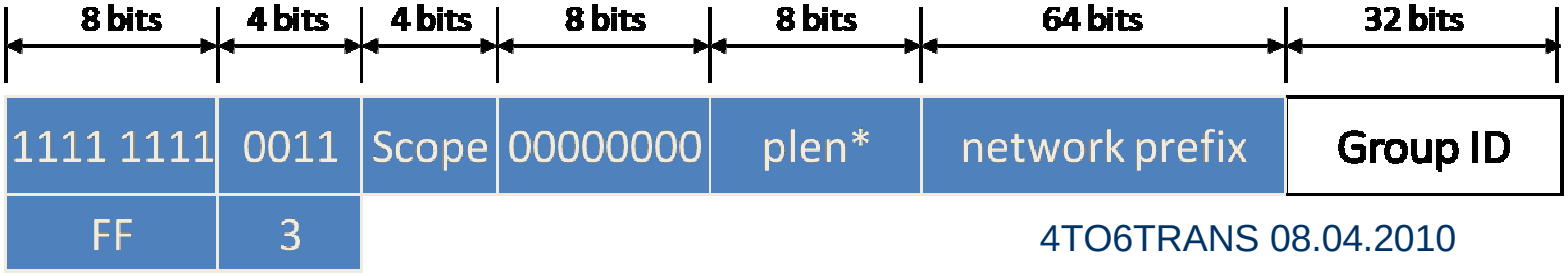


Solicited-node multicast address

- Used by IPv6 Neighbor Discovery protocol
- A multicast address to which Neighbor Solicitation messages are sent
- Formed by taking the low-order 24 bits of an address (unicast or anycast) and appending those bits to the prefix FF02:0:0:0:0:1:FF00::/104
  - FF02:0:0:0:0:1:FFXX:XXXX
- Computed for each unicast and anycast addresses that have been configured for the node's interfaces
- Example:
  - For IPv6 unicast address FD00::abcd:1234:5678 , the corresponding Solicited-node address is FF02::1:FF34:5678

Unicast-Prefix-based Multicast IPv6 addresses

- Make multicast addresses unique between two subnets
- By delegating multicast addresses at the same time as unicast prefixes, network operators will be able to identify their multicast addresses without needing to run an inter-domain allocation protocol.
- Defined in RFC 3306
- See RFC3307 on how to allocate Group IDs
- Example
  - For IPv6 unicast prefix FD00:0:0:abcd::/64, the corresponding unicast-prefix-based multicast prefix with link-local scope is FF32:0040:FD00:0000:0000:abcd/96
  - * “plen” (prefix length) indicates the number of bits in the network prefix field



Obtaining interface identifier for IPv6 address from MAC (IEEE 802) address

Host A has the MAC address of 00-0D-5D-03-F9-CC

- Convert MAC address to EUI-64 (Extended Unique Identifier) format:
  - 00-0D-5D-FF-FE-03-F9-CC
- Complement the seventh bit of first byte:
  - The first byte in binary form is 00000000. When the seventh bit is complemented, it becomes 00000010 (0x02).
  - 02-0D-5D-FF-FE-03-F9-CC
- Convert to colon hexadecimal notation
  - ::020D:5DFF:FE03:F9CC
- The link-local address for the host is:
  - FE80::020D:5DFF:FE03:F9CC
- The solicited-node address is:
  - FE02::1:FF03:F9CC

Neighbor Discovery Protocol

- Replaces ARP (Address Resolution Protocol)
- Used by nodes (hosts and routers)
  - In address resolution process (to determine link-layer addresses)
  - In neighbor unreachability detection
  - Duplicate address detection
- Used by hosts
  - In router discovery process
  - In stateless address autoconfiguration process
- Used by routers
  - Advertise their presence, host configuration parameters, and on-link prefixes
  - Inform hosts of a better next-hop address (redirect)

ICMPv6 packet types used in Neighbor Discovery (1)

- Router Solicitation
  - Send by host when an interface is enabled to request routers to generate Router Advertisements immediately rather than at their next scheduled time
  - Source address is the link-local address of the host
  - Destination address is FF02::2
- Router Advertisement
  - Send by routers periodically or in response to a Router Solicitation message in order to notify their presence and provide information such as: host configuration parameters and on-link prefixes
  - Source address is the link-local address of the sending router
  - Destination address is the unicast address of a node that sent a Router Solicitation or FF02::1
- Redirect
  - Send by routers to inform hosts of a better first hop for a destination

ICMPv6 packet types used in Neighbor Discovery (2)

- **Neighbor Solicitation**
  - Sent by a node to determine the link-layer address of a neighbor, or to verify that a neighbor is still reachable via a cached link-layer address
  - Source address is the link-local address of the node
  - Destination address is the solicited-node multicast address corresponding to the target address, or the target address
  - Also used for Duplicate Address Detection
    - The Target Address field in the Neighbor Solicitation message is set to the IPv6 address for which duplication is being detected
    - The Source Address is set to the unspecified address (::)
- **Neighbor Advertisement**
  - Send by a node in response to a Neighbor Solicitation message
  - A node may also send unsolicited Neighbor Advertisements to announce a link-layer address change
  - Also used for Duplicate Address Detection
    - The Destination Address is set to the link-local scope all-nodes multicast address (FF02::1)

IPv6 Address Autoconfiguration

- Stateless autoconfiguration
  - No manual configuration of hosts is required
  - Hosts can generate their own address by appending its 48 bits MAC address in EUI-64 bits format to the 64 bits of the local link prefix advertised by the router
  - Router advertisement messages contain also lifetime information for each prefix in the advertisement
  - Duplicate address detection

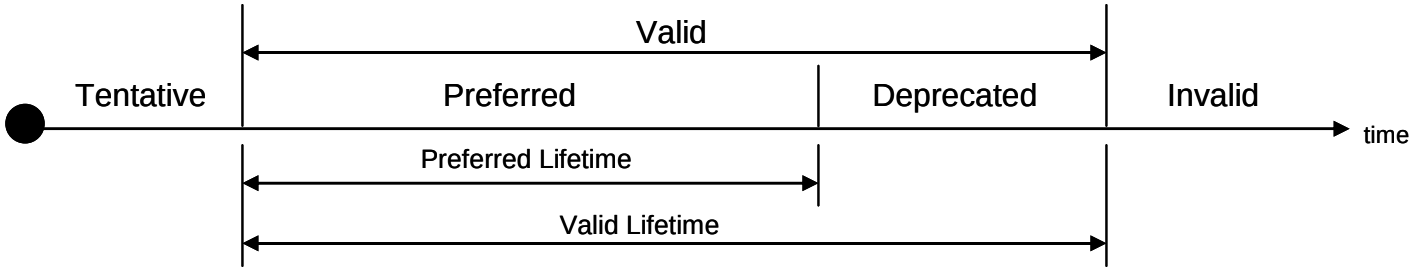
- Stateful autoconfiguration
  - Configuration information is provided to a host by a server such as DHCPv6

IPv6 Address Autoconfiguration Process

- Configure Link-local address
  - Perform duplicate address detection
- Perform router discovery by sending router solicitation messages
- Use Router Advertisement message contents to determine the following items.
  - Configuration parameters
  - Stateless addresses and on-link prefixes
  - Perform duplicate address detection for stateless addresses
  - Whether to use stateful address configuration
  - Specific routes ....

Autoconfiguration address states

- Tentative
  - Accepts only Neighbor Discovery packets related to Duplicate Address Detection for the tentative address
- Valid
  - An address from which unicast traffic can be sent and received
  - Preferred state - uniqueness has been verified, unrestricted use
  - Deprecated state – its use is discouraged, but not forbidden
- Invalid
  - An address from which unicast traffic can no longer be sent and received



Manually configure an IPv6

- On Windows client:
  - netsh interface ipv6 install/uninstall
    - IPv6 is installed and enabled by default on Windows Vista and Windows 2008 Server
  - netsh interface ipv6 add address "Local Area Connection" fd00::c0a8:64
- On Linux client:
  - ip -6 addr add dev eth0 fd00::c0a8:c7/64
- On Mac OS X client:
  - ifconfig en0 inet6 add fd00::c0a8:101 prefixlen 64

IP Auto Configuration

Router Advertisement Daemon – radvd

/etc/init.d/radvd start|stop|restart
/etc/sysconfig/network
IPV6FORWARDING=yes
/etc/radvd.conf

```
interface eth0
{
        AdvSendAdvert on;    #needs to be set to “on” in order the router to send periodic router
                    # advertisements and to respond to router solicitations

        AdvDefaultLifetime 90; #in seconds
        MaxRtrAdvInterval 30; #advertise at least every 30 seconds
        MinRtrAdvInterval 10; #but not less than every 10 seconds
        AdvReachableTime 60000; #in milliseconds

        prefix fd00:0:0:15::/64
        {
                AdvAutonomous on;
                AdvValidLifetime 120; # in seconds (default is 30 days)
                AdvPreferredLifetime 60; #in seconds (default is 7 days)
        };
};
```

DNS support

- AAAA record
  - Maps host name to IPv6 address
  - Equivalent to A record in IPv4
  - Uses the following format:
  - host-ipv6 IN AAAA fd00::c0a8:cd24

- PTR record
  - Maps IPv6 address to host name
  - New reverse domain called IP6.ARPA
  - Uses the following format to store IPv6 addresses:
    - 4.2.d.c.8.a.0.c.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.d.f.ip6.arpa IN PTR host-ipv6.test.net

IPv6 Pros and Cons

Why shall we deploy IPv6? (1)

IPv4 adresses in April 2010 according to RIPE

Mohammad Mahloujian April 09, 2010

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

xx	Allocated in Feb 2010
xx	Allocated in Apr. 2010
xx	Used
	Available
	Not useable

Total	256
Free	20
Percent	7,81%

Why shall we deploy IPv6? (2)

- IPv6 technology is an 'enabler' of new business opportunities. The technology itself is not a 'market driver'.
- IPv6 is NOT a feature. It is about the fundamental IP network layer model developed for end-to-end services and network transparency.
- With the exhaustion of the IPv4 free pool (only about 8.7% left free), IPv6 deployment enables **BUSINESS CONTINUITY**.
- Least but not last IPv6 provides new features
  - virtually unlimited addressing space
  - native support for mobility, security, multicast, etc.
  - Auto configuration (plug & play)
  - No NAT ☺

Why shall we not go towards IPv6

- IPv4 Networks are already too complex
- There are still devices and applications that does not support IPv6.
- IPv4 and IPv6 do not interoperate:
  - IPv4 applications do not work with IPv6
  - IPv4 nodes can not communicate with IPv6 nodes
- Security. Currently we use NAT and the outside hosts does not see the IP addresses of the inside hosts. Only the IP address or the IP addressing pool of the NAT device.
- There are no tools able to reconfigure the Network services that we already use in controlled and automated fashion. So such transition might be a huge mess. Those services might be
  - Internet Access
  - VoIP
  - Business VPN data connectivity
  - Remote Access
  - IPTV
  - And many others …..

Coexistence

It is likely that IPv4 and IPv6 will coexist for a long period of time:

- How to enable communications among IPv6 islands isolated in the IPv4 world?
- How to enable communications between the existing IPv4 world and the new IPv6 world?

Basic transition mechanisms

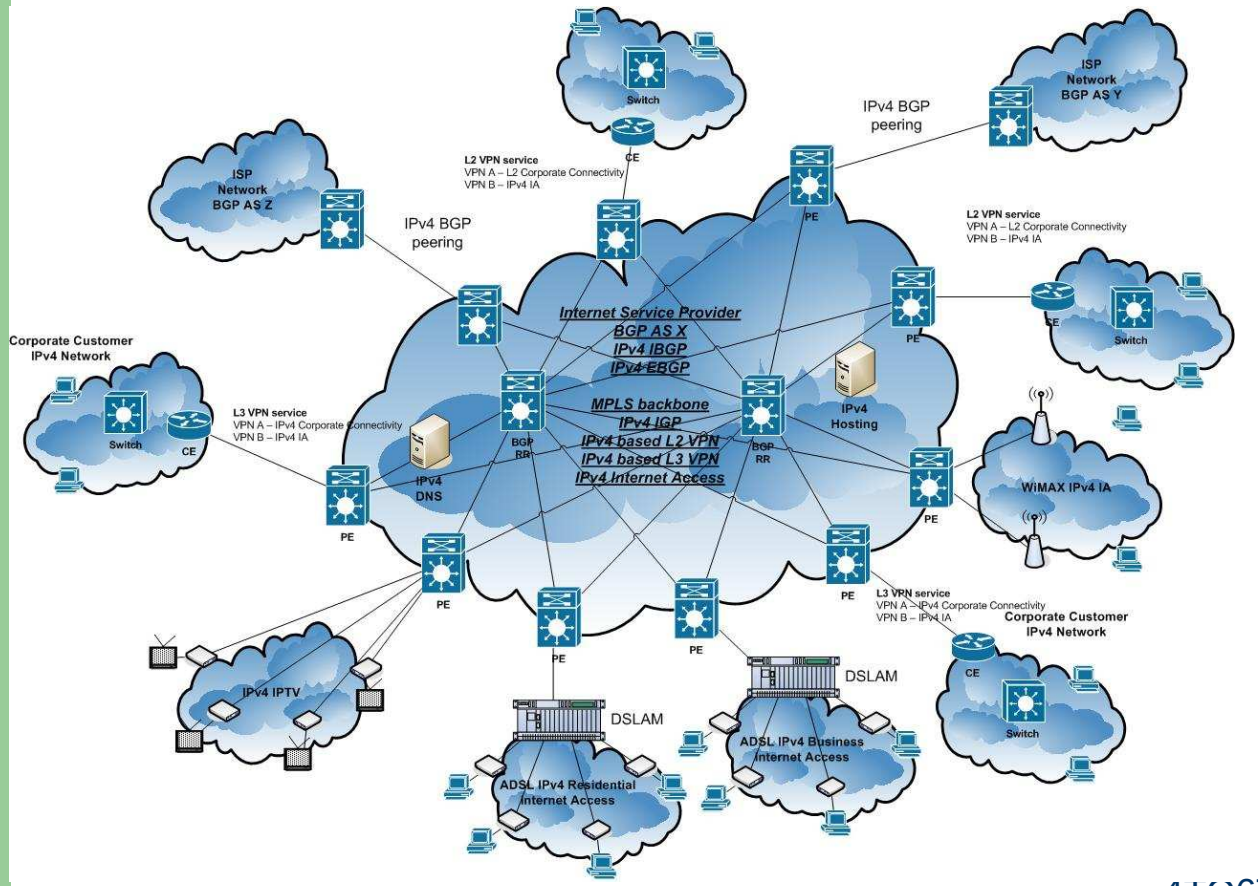
- Dual IP Stack
  - provision of complete support for both IPv4 and IPv6 in hosts and routers
- IPv6 over IPv4 tunneling
  - encapsulation of IPv6 packets within IPv4 headers to carry them over an IPv4 network (e.g. Internet)
  - two types of tunneling: configured and automatic
- NAT-PT

4TO6TRANS concepts and objectives

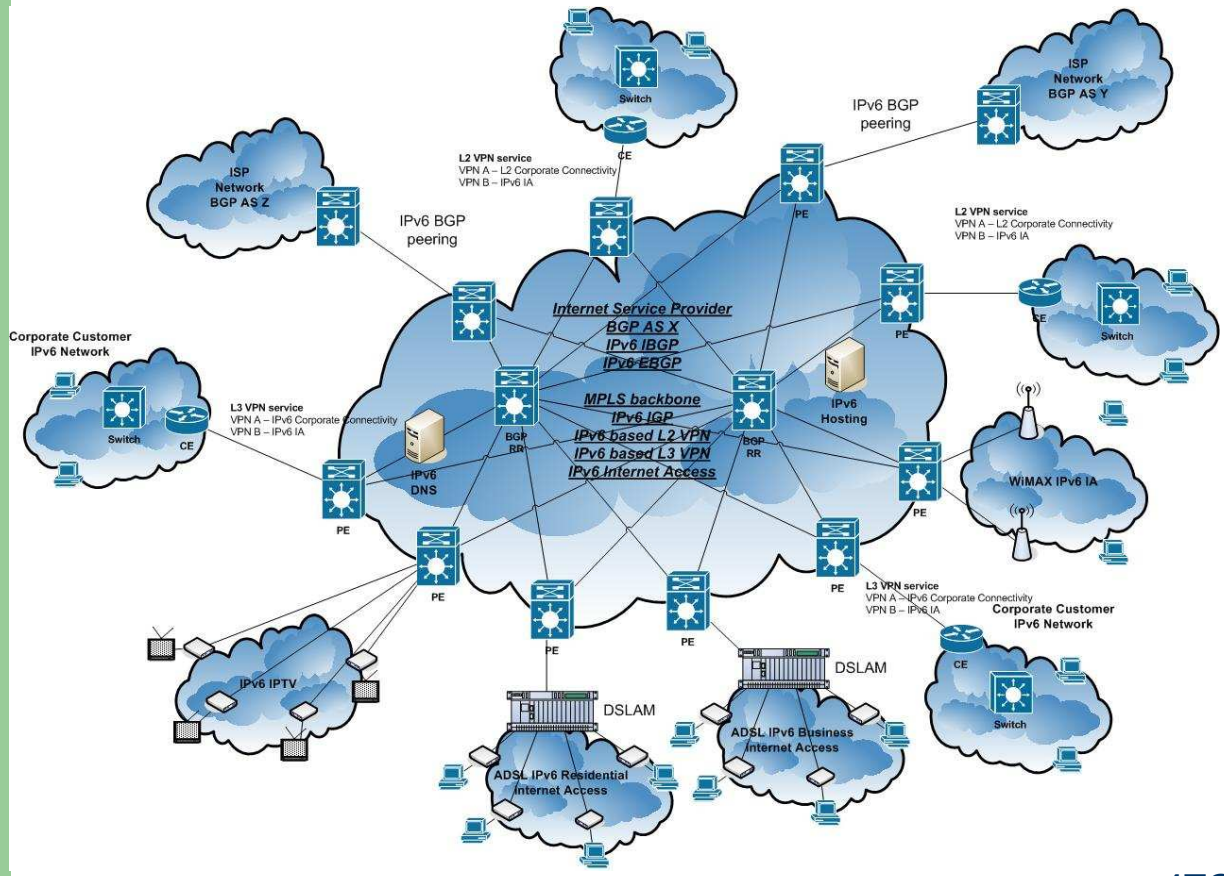
4TO6TRANS project Goals

- The project target is performing the process of transforming IPv4 to an IPv6 service provider infrastructures.
- To go beyond the state of art of the current Operation Support Systems

IPv4 based Service Provider Network will be transformed ….



To an IPv6 based

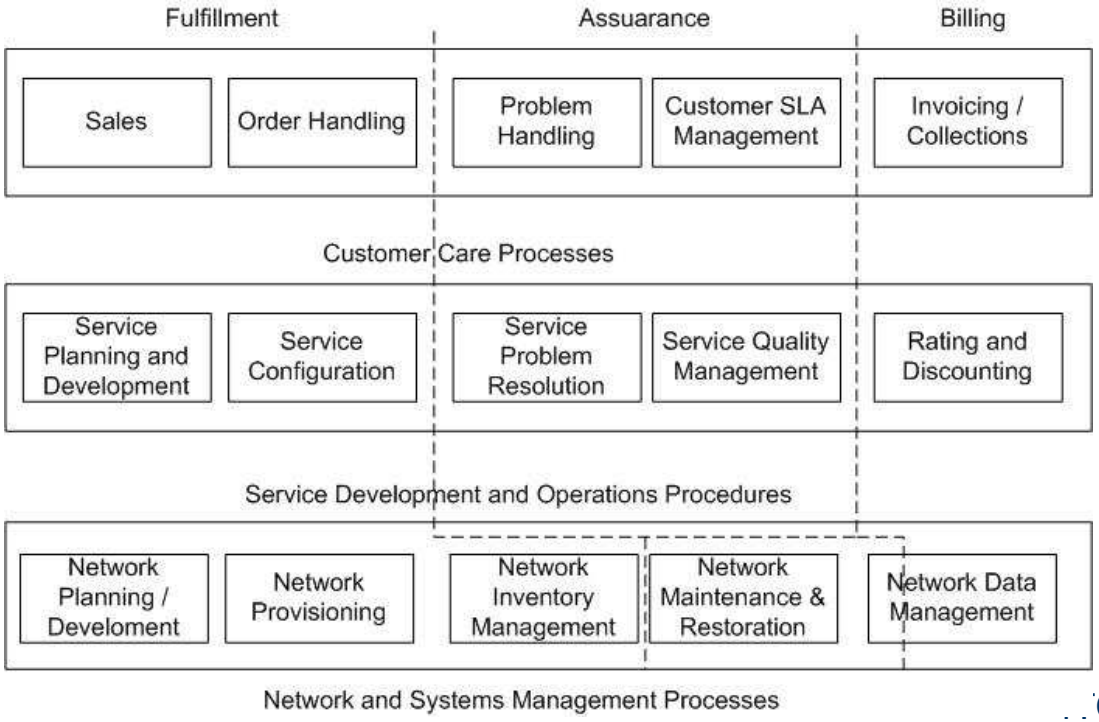


4TO6TRANS project Target

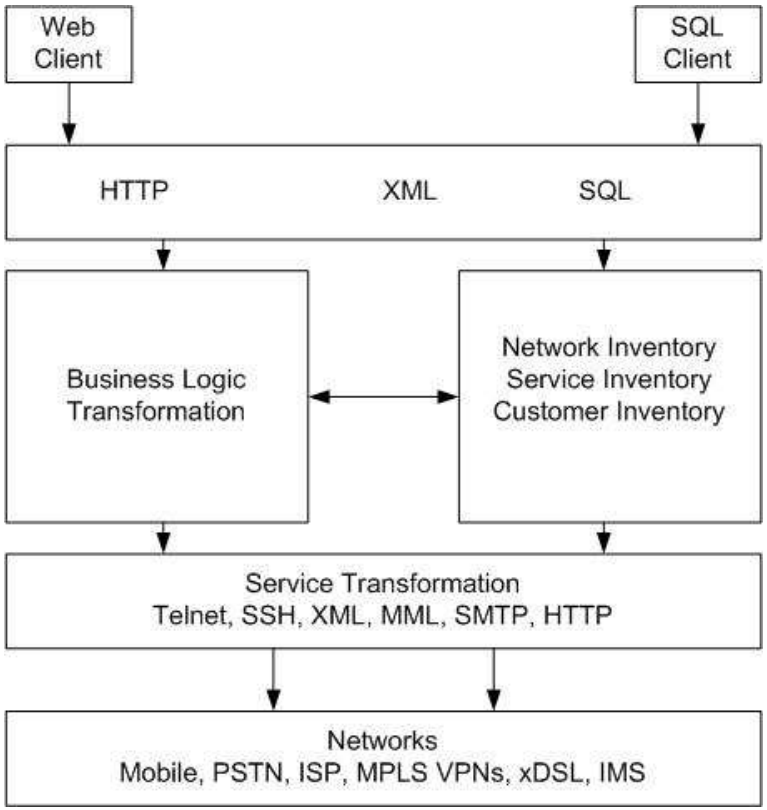
- Creation of 4TO6TRANS framework:
  - having the power and ability to model the current services
  - to “communicate” with the network devices via CLI and SNMP
  - to follow certain business logic during the transformation process
- A similar network migration task being extremely complex, the, 4TO6TRANS promote the idea of an open framework instead of closed platform. That is the only way to handle/have control on the variety and complexity of the current IP networks.
- The framework architecture will consist of several Application Programmable Interfaces build on technologies beyond the current state of art.

Fulfillment, Assurance and Billing (FAB) model

Best practice for ISP OSS architecture recommended by TMFORM part of the OSS eTOM model.



Solution Architecture



- Transformation inputs
- BTL APIs
- Inventory API
- Service transformation
- Network Layer

Transformation Input Layer

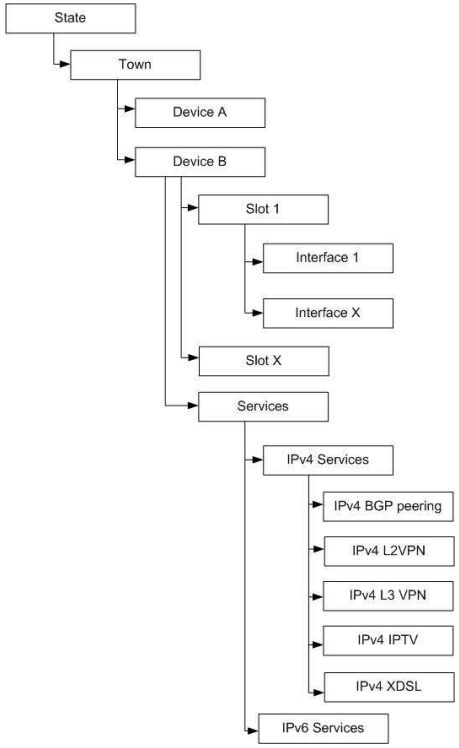
- Service transformation orders may come from different sources
- The most common source will be the Service provider CRM system
- The transformation request might be in HTTP, SQL or preferable XML format
- It shall contain the input data needed for the successful service transformation
- The BTL shall take that input and perform the transformation
- It shall be able to give an intermediate status of the order and also a final result once the transformation process is finished

Network Inventory

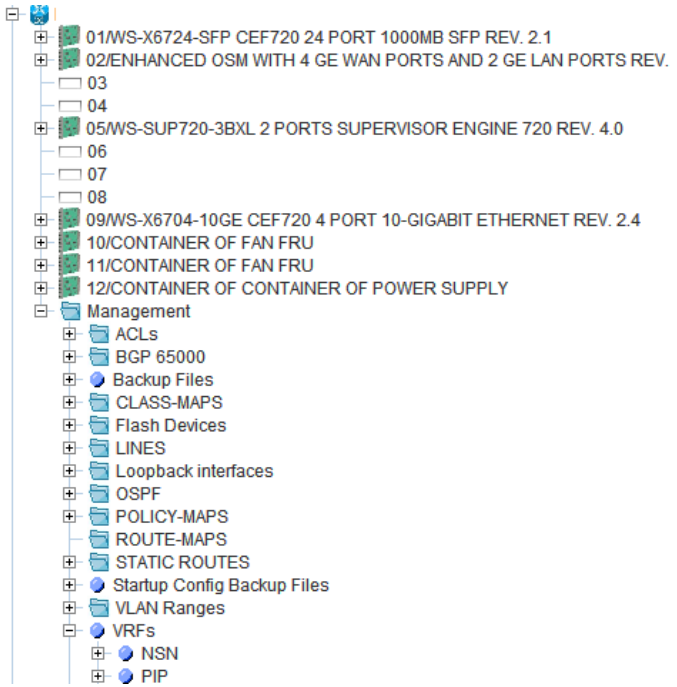
- Contains a logical data model of the network
- Has to be able to model the network devices, their physical and logical structure, the services running on them and the subscribers using those services
- Has to be flexible enough to respond to network changes, extensions and replacements
- The data inside shall be populated in a dynamic and automated way

Inventory data

- Logical Model



- Real device



Inventory Automation

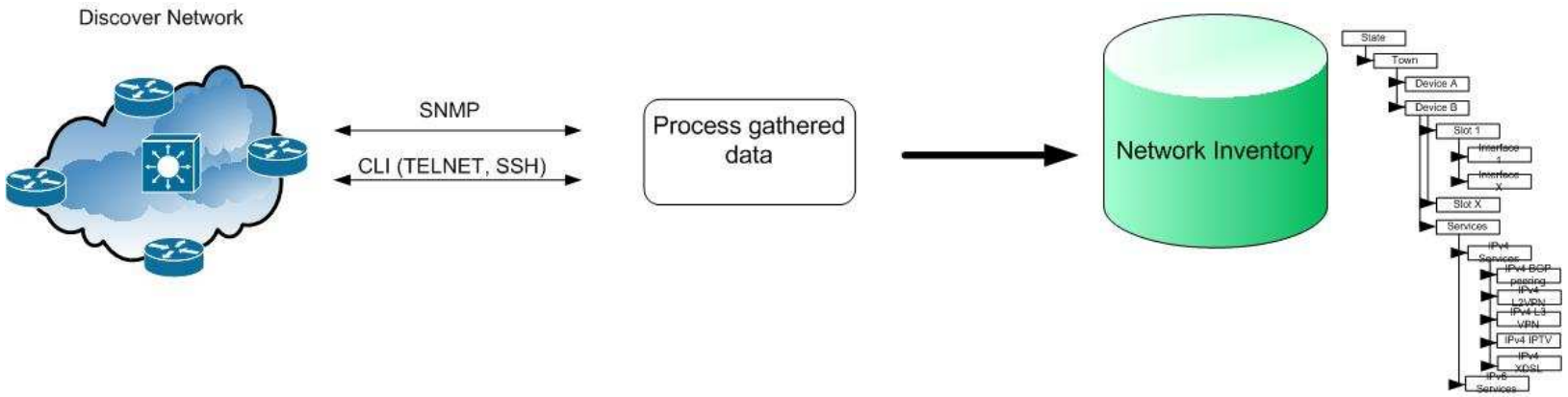
- One of the main goals of the project is to automate the fulfillment process.
- One of the most time consuming steps in that process is filling the inventory with real data from the network.
- To speedup this process two additional functionalities will be developed
  - Device Discoveries
  - Automated Uploads of the discovered devices

Device Discoveries

- Device discovery fulfills an algorithm able spread like a virus thought the network.
- The discovery algorithm needs initial object or IP address to start with.
- Once the initial device is analyzed and identified the algorithm will discover it's immediate neighbors of analyzing various network protocols (MPLS LDP, CDP, routing protocols etc)
- Once certain device is discovered it will be uploaded into the inventory and its neighbors will be also analyzed by the discovery algorithm.
- Discovery STOP criteria has to be defined
  - Already discovered devices shall be distinguished by the discovery algorithm;
  - Discovery might stop if the discovered node is certain type. For example is MPLS Backbone Router or is Customer device.

Device Uploads

- Through the Uploads the data will be filled in into the inventory database.
- Upload Functionality polls the devices through the service transformation API in order to gather the data needed for feeding the Logical model.



Service Automation

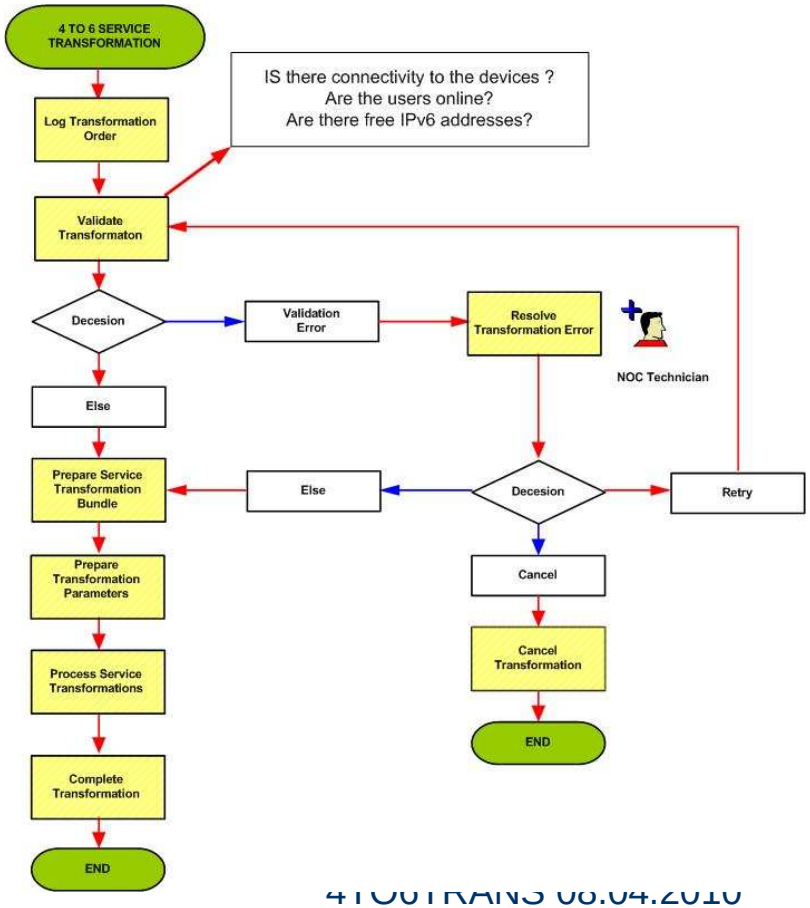
- Once the inventory is populated with the logical model of the network
- The information will be used by the transformation process algorithm.
- The algorithm will be driven by the BTL API

Business Transformation Logic

- Able to model the transformation process
- Able to represent the process in a GANT chart to the process operators

Common transformation algorithm

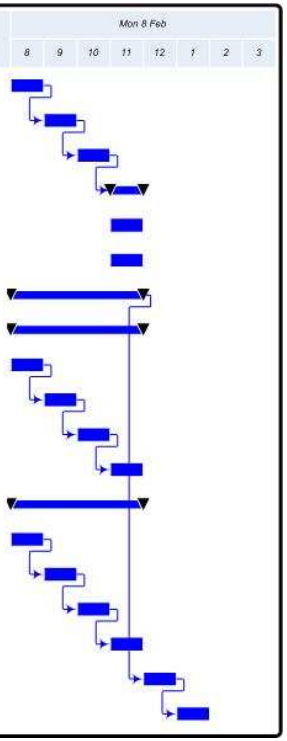
- Transformation algorithm building blocks
  - Log Transformation order
  - **Validation!!!**
  - Prepare service transformation bundle
  - Process service transformation
  - Resolve transformation errors
  - Complete transformation
  - Cancel transformation
- Available for transformation process designers



Algorithm execution

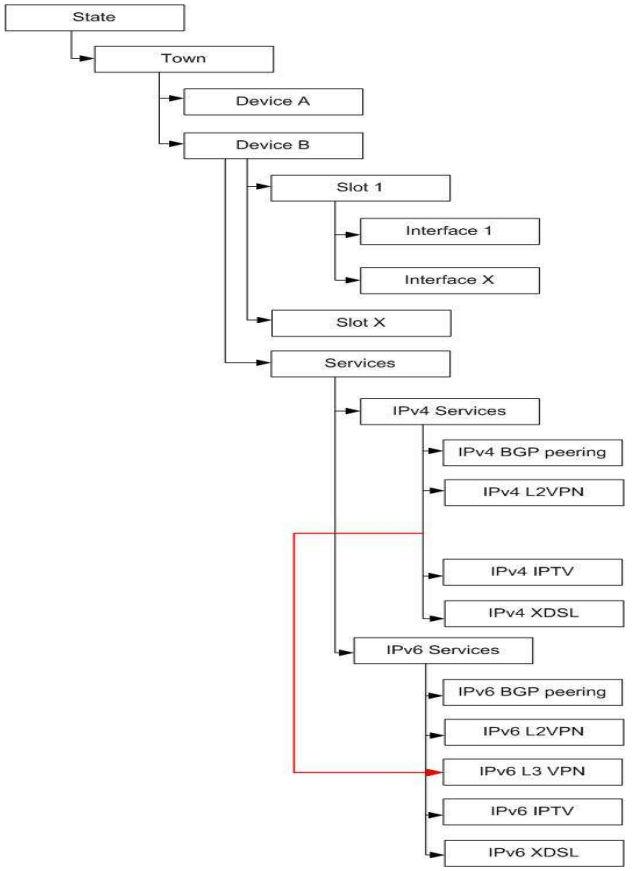
- Algorithm tasks could be manual, automatic or semi-automatic
- Each task could to be processed/stopped controlled by an operator
- GANT chart representation

ID	Task Name	Start	Finish	Duration
1	**Start**	2/8/2010	2/8/2010	1h
2	**Transform Order**	2/8/2010	2/8/2010	1h
3	**ORDER VALIDATION**	2/8/2010	2/8/2010	1h
4	**PREPARE SERVICE TRANSFORMATION BUNDLE**	**2/8/2010**	**2/8/2010**	**1h**
5	Prepare DSL service	2/8/2010	2/8/2010	1h
6	Prepare IPTV service	2/8/2010	2/8/2010	1h
7	**PROCESS ATOMIC TRANSFORMATIIONS**	**2/8/2010**	**2/8/2010**	**4h**
8	**Process DSL Service**	**2/8/2010**	**2/8/2010**	**4h**
9	Reserve IPv6 Address	2/8/2010	2/8/2010	1h
10	Add MAC to the new pool	2/8/2010	2/8/2010	1h
11	Reconfigure BRAS	2/8/2010	2/8/2010	1h
12	Reconfigure Customer Modem	2/8/2010	2/8/2010	1h
13	**Process IPTV Service**	**2/8/2010**	**2/8/2010**	**4h**
14	Reserve IPv6 Address	2/8/2010	2/8/2010	1h
15	Add MAC to the new pool	2/8/2010	2/8/2010	1h
16	Reconfigure IPTV system	2/8/2010	2/8/2010	1h
17	Reconfigure Customer Settop box	2/8/2010	2/8/2010	1h
18	**COMPLETE TRANSFORMATION**	2/8/2010	2/8/2010	1h
19	**END**	2/8/2010	2/8/2010	1h



Successful Transformation

- Once the transformation algorithm has finished
- If we have a successful transformation
- IPv4 services will be moved from the IPv4 service branch
- To the IPv6 service branch



4TO6TRANS project status

Project Development

- The 4TO6TRANS is quite difficult project that needs people with various technical skills:
  - Software Solution Architects
  - Network Solution Architects
  - Database Developers
  - Java Developers
- It’s a job for 10 engineers for 3 years period of time…

Project Funding

- Different funding schemas exists
  - FP7 ICT cooperation
  - FP7 ICT ideas
  - EUREKA's Eurostars
  - Private funding
  - Government funding

Subject Popularization

- Past/Current Projects with similar goals
  - 6INIT
  - 6DISS
  - 6DEPLOY
- None of them targets OSS provider infrastructures!!!
- The subject shall be made more popular

IPv4 to IPv6 TRANSFORMATION

eng. Nikolay Milovanov
CCIE SP# 20094
email: *nmil@niau.org*
http://niau.org

Нов български университет