네트워크 서비스 인터페이스 커넥션 프로토콜(NSI2.0 R117 CS Protocol) 기술 분석

일자: 2014년 11월 25일

부서: 첨단연구망센터 첨단연구망개발팀
제출자: 문정훈
jhmoon@kisti.re.kr

KiSTi www.kisti.re.kr
한국과학기술정보연구원
Korea Institute of Science and Technology Information
305-806 대전광역시 유성구 어은동 52번지
TEL (042)869-0676 / FAX (042)869-0679
www.kisti.re.kr

1.

2. NSI (Network Service Interface)

1) NSI 기본 개념

2) NSI R117

3) NSI 계층

4) NSI 자원 명세

3. NSI CS(Connection Service)

1) NSI CS 요구사항

2) NSI Discovery

3) NSI Life Cycle

4) NSI Service Definition & Service Decoupling

5) NSI CS Messages

1.

기술의 개발과 표준화를 진행하고 있는 OGF(Open Grid Forum)에서는 멀티-도메인 간의 네트워크 자원관리를 위하여 서비스평면, 제어평면 및 전달평면으로 구성되는 연결서비스 프레임워크를 정의하고 있다. NSI(Network Service Interface)는 서비스평면에서 멀티-도메인의 네트워크 서비스 에이젼트들 간에 예약 기반의 연결 관리를 지원한다. NSI 인터페이스에서 예약된 자원의 구성을 위하여 제어 및 관리평면의 NRM(Network Resource Manager)은 도메인 내부에 있는 전달평면의 스위치 장비들을 제어한다. 차세대 연구망은 실시간 자원제어와 자원의 사용 효율성 증대를 위하여, 네트워크 자원의 동적 제어 및 관리가 가능하며 트래픽 엔지니어링과 QoS 기술을 제공하는 GMPLS(Generalized Multi-Protocol Label Switching) 기술을 도입하고 있다. NSI 2.0은 "Open standard for worldwide dynamic circuit service"란 문구와 함께 전 세계를 대상으로 dynamic circuit service를 제공할수 있도록 발전하고 있으며 기존 북미, 유럽, 아시아와 함께 남미(ANSP, RNP), 일본의 SINET4의 추가 참여 등으로 그 규모를 넓혀가고 있다.

2014년에 NSI 2.0은 기술 및 사용자 요구사항을 반영하여 계속하여 리비전되었으며 현재 가장 널리 사용되고 있는 버전은 r117이다.

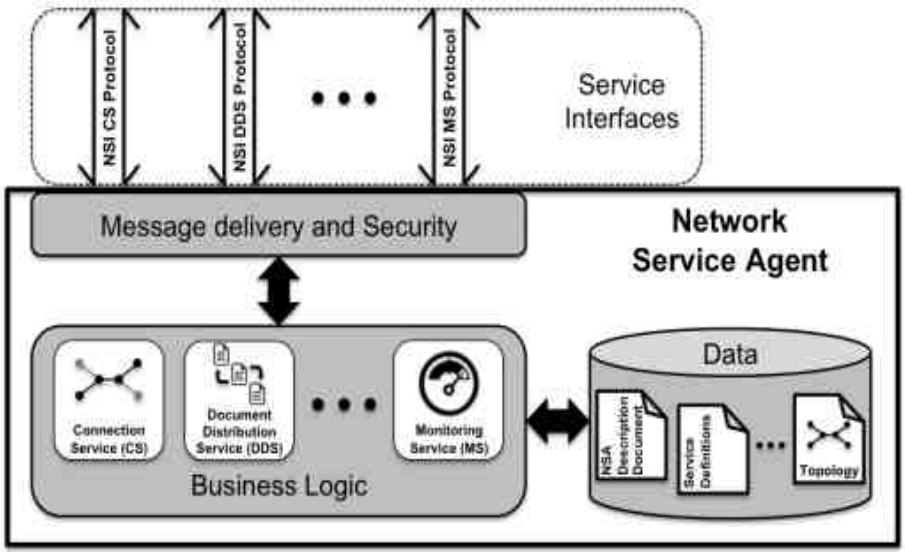
본 문서에는 OGF의 NSI 2.0 r117을 분석하여 기술한다.

2. NSI (Network Service Interface) r117

· NSI 개념

NSI는 네트워크 서비스 인터페이스로서 NSF (Network Service Framework)에 기반한다. 그리고 OGF의 NSI워킹그룹을 통해서 표준화가 진행되고 있다. 기본적인 구조로서는 각각의 네트워크 NRM을 NSA(Network Service Agent)라고 부르며, Originatror, Aggregator, Ultimate Provider로 구분이 된다. NSI에는 글로벌 타이머와 각각의 로컬 네트워크를 관리하는 NRM이 존해한다. 기본적으로 Originator는 최조의 입력을 받는 Requester의 성격을 가지며, Aggregator는 트리 구조상으로 볼 때, 자신의 아래에 자식 노드를 가지는 노드를 말하며, 이때 상 위 노드에서 전달되어 오는 예약을 시작으로 다양한 정보에 대해서 자식노드에게 전달하는 역할을 담당한다. 마지막으로 Ultimate Providier는 더 이상 자식노드가 없는 최종 노드로서 그 노드의 로컬 네트워크 자원에 대해서 NRM을 통해서 자원의 예약 및 메시지를 수행한다.

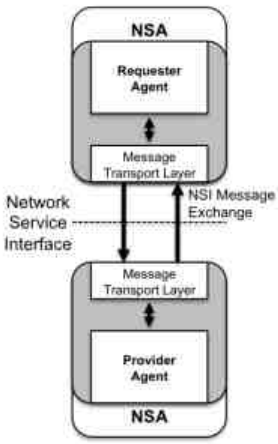
NSI r117에서는 NSI v1.0 / v1.1 / v2.0의 기본 개념을 이으며 네트워크 제어를 위해 모니터링, 네트워크 자원 제어 및 조회 서비스를 제공한다.



그림은 NSI 서비스의 프레임워크를 보여준다. 여러 NSA(Network Service Agent)는 NSI를 이용하여 다른 NSA들과 통신을 하며 인터-도메인간의 네트워크 자원을 연결시켜준다. NSA는 네트워크 자원 제어를 위한 NRM을 가지며 NSI를 사용하여 NSA간 인터페이싱을 하기 위한 모듈을 가진다. NSI 서비스 프레임워크는 확장성을 고려하여 구현되었으며 SDN 및 NFV와의 통합도 고려되고 있다.

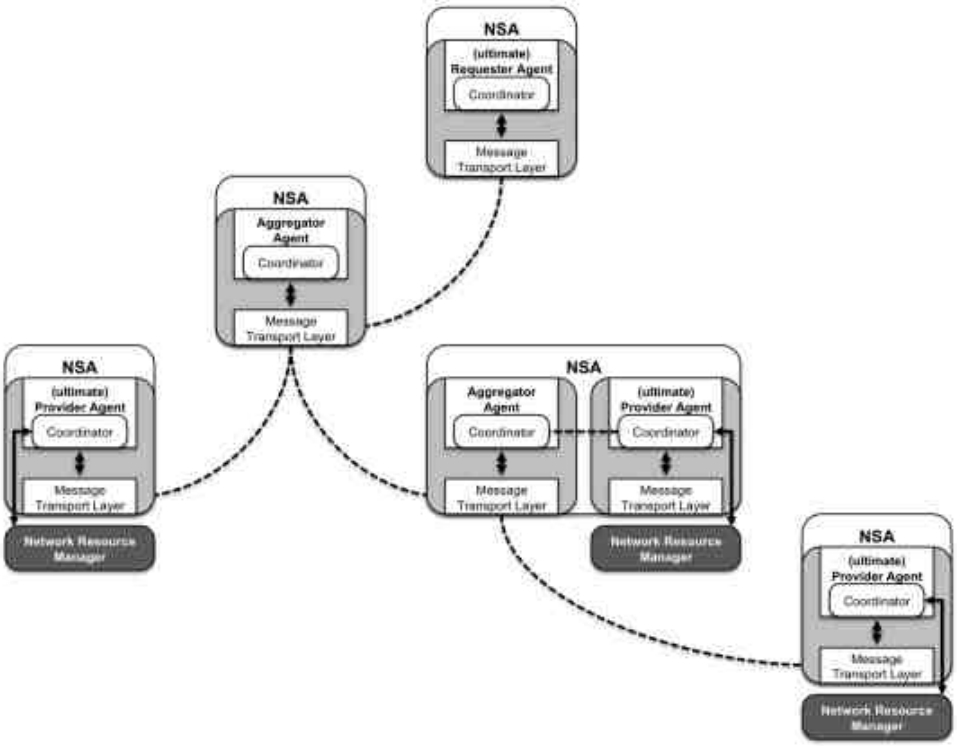
BusinessLogic에 포함되어 있는 NSI CS(Connection Service)는 NSI 서비스의 가장 주요한 키이다. NSI CS는 서로 다른 타입의 커넥션들의 병합 시켜주는 것이 가능하다. NSI DDS(Document Distribution Service)는 다른 NSA와의 정보 공유를 위해 사용된다. 각 NSA의 기능 및 특성, 정보(NSI 프로토콜 버전, NSI 서비스 제공자..)의 명세이다. NSI DDS는 NSI 서비스 중 Service Definition과 Topology Representation에 사용된다.

· NSA(Network Service Agent)



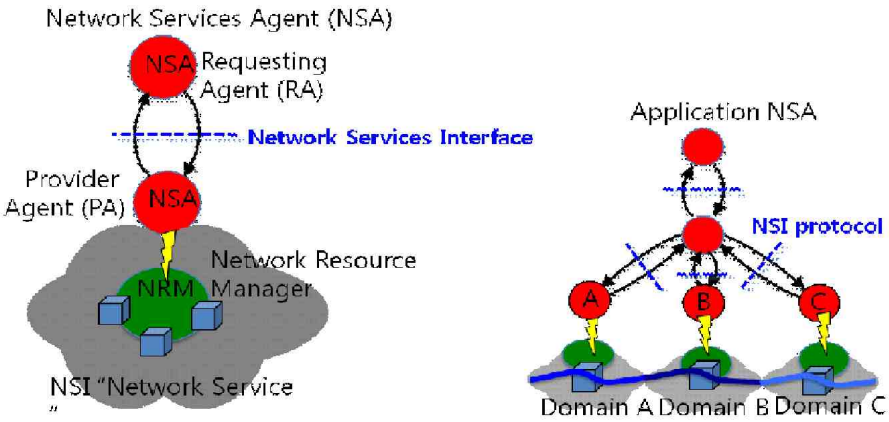
NSI 프레임은 NSA간의 통신을 통하여 이루어진다. NSA 소프트웨어는 NSI 프로토콜에 정의된 상태머신에 의거하여 메시지를 교환하고, 구현되어야 한다. NSA중 서비스를 요청하는 경우를 RA(Request Agent)라 하며 서비스 요청에 응당하는 경우를 PA(Provider Agent)라 한다.

각 NSA는 최대한의 유연성을 가질 수 있도록 설계 되어 있다.



일반적인 네트워크 구성에는 굉장히 많은 네트워크들이 포함되며 그 구성 또한 매우 복잡하다. 이를 반영하기 위한 NSI 메시지는 유연하게 구성되어 있으며 NSA역시 유연하게 구성하게 하는 것이 가능하다.

유연한 NSA 구성을 반영하여 메시지를 전달하기 위해 다음과 같은 모델이 제안 되어있다. 첫 번째로 최초 NSA가 request 요청을 받고 이 요청에 대해서 다음 노드로 전달하게 되고, 이때 멀티도메인 환경으로 진행되고, 요청을 받은 다음 노드는 해당 로컬 NRM을 통해서 로컬 네트워크의 해당 자원에 대한 예약을 진행하게 된다. 그 결과를 confirm 메시지 형태로 역순으로 보내게 되면 최초 요청자에게 확정 메시지가 도창하게 되고 그 다음 provisioning을 진행하게 된다. 따라서 이러한 과정을 거치게 되면 해당 노드들은 인접한 다음 노드들에게 정보를 제공함게 됨으로서 멀티도메인 환경에서도 단일 도메인과 같은 경로의 설정과 메시지에 따른 컨트롤 작업들이 진행된다.



노드들의 메시지 전달 과정에 대해서는 다음과 같이 몇가지 제안들이 있다.

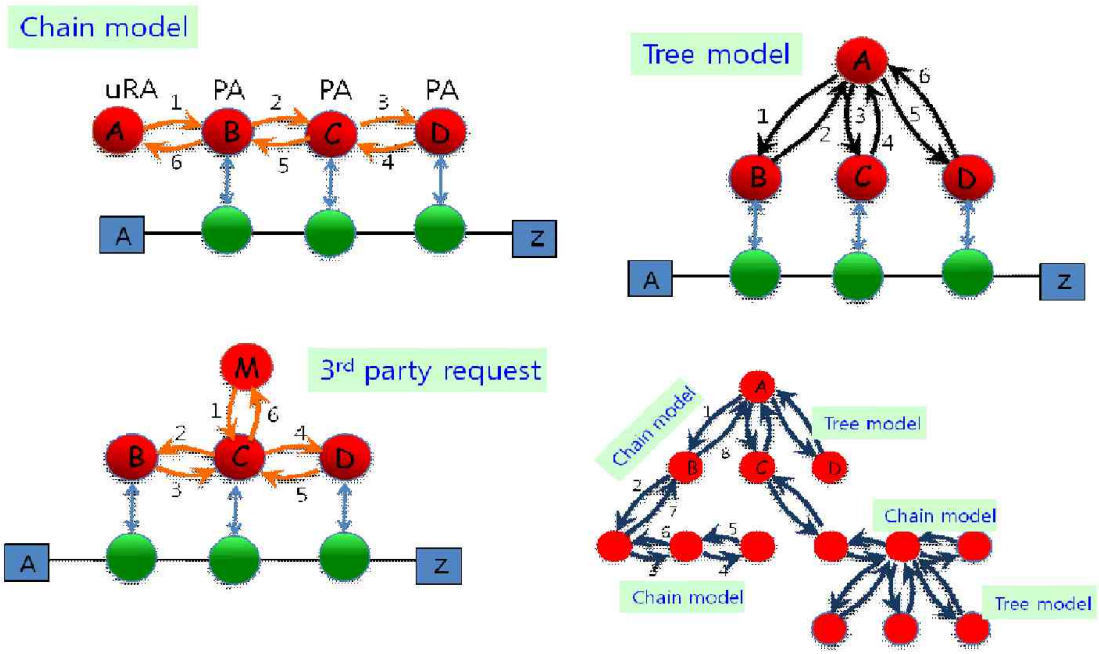
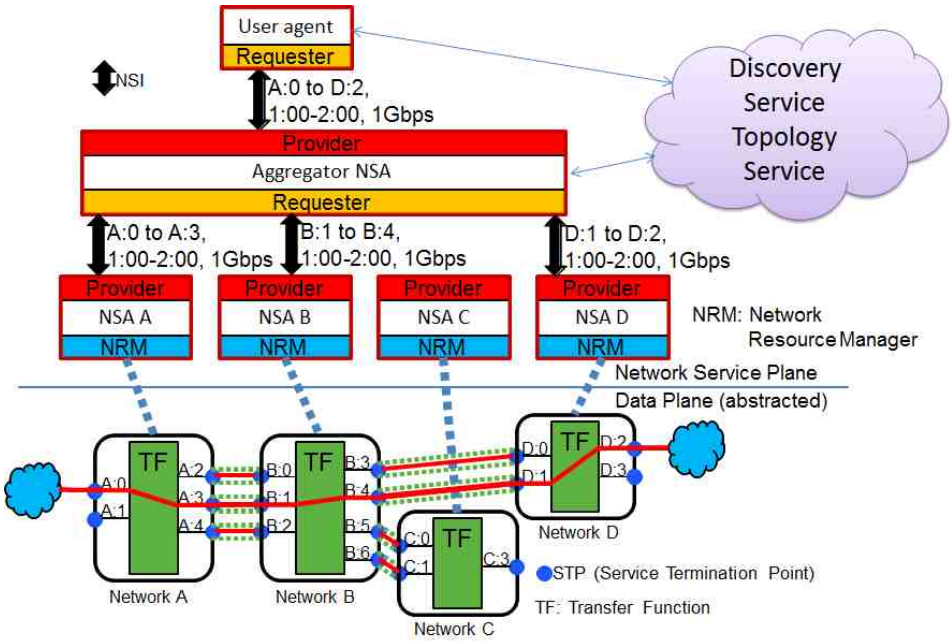


그림 5 Chain, Tree Model

먼저 Chain model로서 A, B, C, D의 노드들은 해당 노드들이 정보를 이웃한 다음 노드들에게 전달하게 되고 전달받은 각각으 노드들은 해당 정보에서 자신의 로컬에 해당하는 자원들에 대해서 로컬 NRM을 통해 예약을 설정한다. 이렇게 D까지 메시지가 전달되게 되면 전체 멀티도메인환경의 크로스 경로가 생성되게 된다. 두 번째 모델로서 Tree Model이 있다. 이 모델의 경우 A, B, C, D의 경우 최초 요처을 받은 A노드가 다른 노드 전체에 개별적인 메시지를 전송함으로서 중앙집중적인 메시지 전송형태를 띠게 되고 다른 노들은 A노드로부터 받은 정보를 통해 Chain Model과 같이 메시지를 전송하고 설정하게 된다. 이외에도 3rd Party request, Complex Model 등이 있으면 각

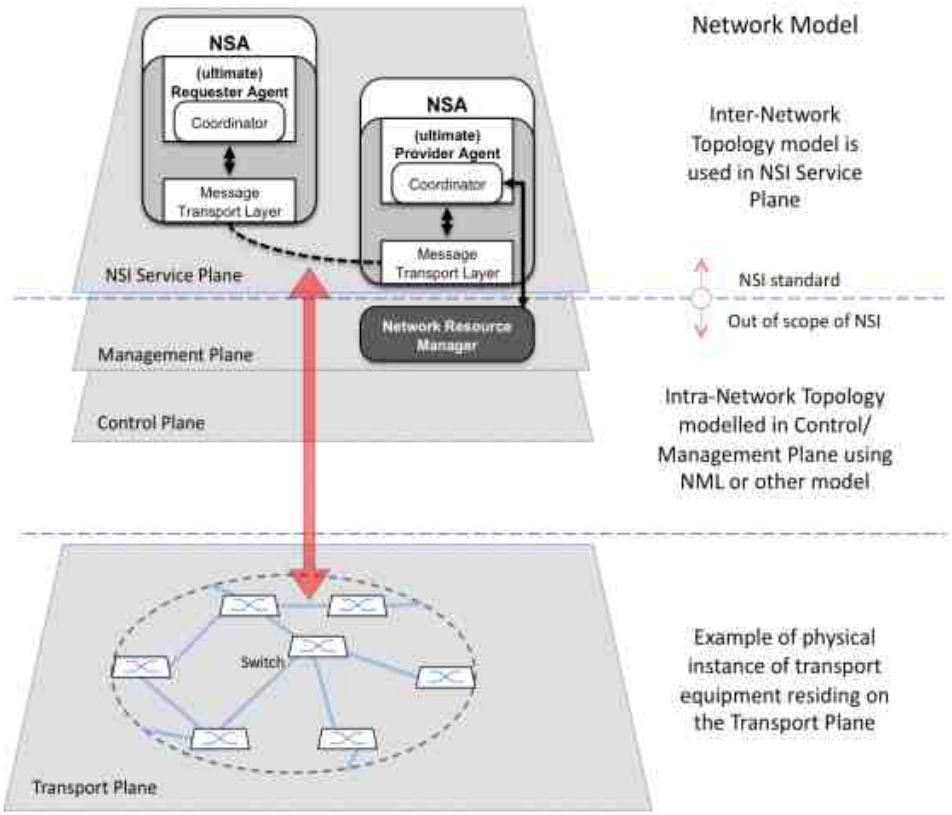
모델을은 글로벌 네트워크 환경의 복잡도를 고려하여 제안된 것으로 해당 토폴로지를 구성하게 된다. 이렇나 토폴로지의 구성은 각각의 노드들이 가지는 로컬 네트워크의 복잡도에 기반하게 되며, 이러한 로컬 네트워크의 효율적인 운영은 NSI 토폴로지 전체의 효율에도 영향을 미치게 된다.

실제 예를 들어볼 경우 다음과 같다. User Agent인 Requester에서 A:0에서 D:2까지 1:00-2:00시까지 1G를 사용하는 경로를 요청하는 경우를 가정하다. 먼저 Requester는 바로 다음에 연동되어 있는 Provider에게 기본적인 요구사항을 Request한다. 이때 첫 번째 Provider는 그 하위에 여러개의 Provider를 가질수 있는 Aggregator이므로 다음번 Provider들에게 Request를 요청할때는 해당 STP(Service Terminate Point)들을 파악하여 첫 번째 Provider에게는 A:0에서 A:3까지 1:00-2:00시까지 1G를 요청한다. 그다음 인접한 두 번째 Provider에게는 B:1에서 B:4까지 1:00-2:00시까지 1G를 요청하고, 다음 인접한 세 번째 Provider에게는 해당 경로에서 없으로므로 지나치고 마지막으로 그다음으로 인접한 네 번째 Provider에게 D:1에서 D2까지 1:00-2:00시까지 1G의 경로를 요청하여 설정하게 한다. 이때 각각의 Data Plane들은 각각의 Provider들은 NRM들에 의해 제어를 받게되고 이렇게 생성되 최종 경로는 A:0에서 D:2까지 1:00-2:00시까지 1G 대역폭에 대한 예약 설정이 완료되게 된다. 이때 User Agent인 Requester와 Aggregator노드에서는 Discoverty Service와 Topology Service를 이용하여 하위 노드들의 STP들을 파악하여 해당 로컬 도메인에 대한 예약을 설정할수 있도록 한다.

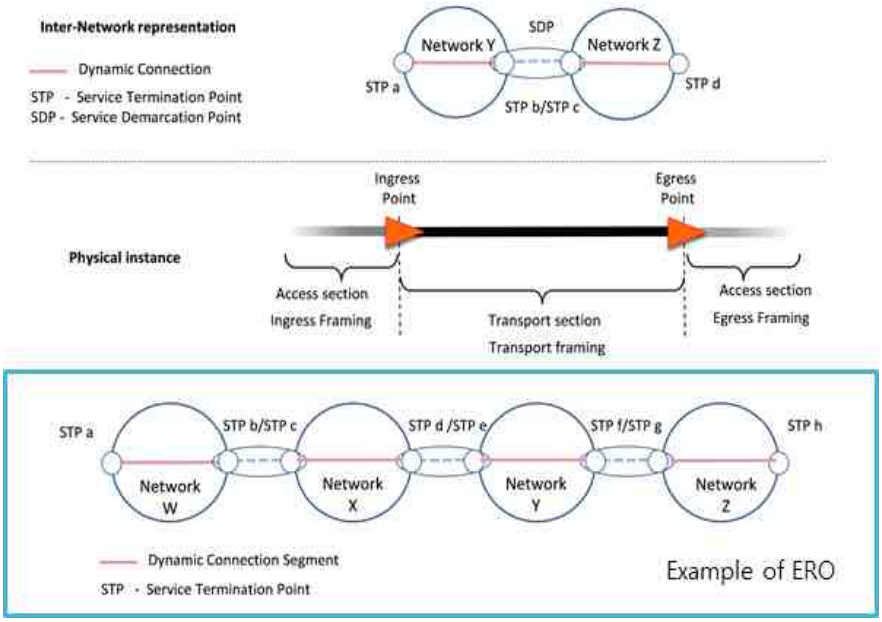


· NSI

그림은 NSI의 계층 모델을 나타낸다. NSI 아키텍쳐는 서비스 계층을 전송 계층으로부터 분리시키고 있으며 NSI 메시지의 사용을 위해 NSA에 특화된 특정한 전송 계층을 필요로 하지 않는다. 이는 NSI가 일반적인 전송계층이 가지는 기능을 신뢰하며 일반적인 전송 계층이 가지는 기능을 변경하지 않는 다는 것을 의미한다.



다음 그림은 NSI의 내부 및 외부 토폴로지 관계를 나타낸다. SDP(Service Demarcation Point)는 네트워크간의 연동포인트를 의미하며 2개의 STP가 그룹으로 묶인 가상의 링크이다. 이러한 여러 개의 링크가 묶여 End to End연결이 연장 될 수 있다. STP(Service Termination Point)는 네트워크의 엣지단을 의미한다.



· **자원 명세**

NSI 네트워는 STP(Service Termination Points)의 구성으로 이루어진다. STP는 각 네트워크의 종단 부분을 의미하며 다른 네트워크와의 접점을 가질 수 있다. STP는 쉽게 생각하여 실제 네트워크 장비(ex : ethernet port)의 포트나 가상 포트(VLAN port)와 같다.

STP는 특정 라벨을 가지고 식별되는 문법을 가지며 라벨에는 STP Group 명과 NML(Network Markup Language)을 기반으로 한 PortGroup이 포함된다.

ex) urn:ogf:network:example.net:2013:A2?vlan=1781

위의 예제는 STP Group으로 urn:ogf:network:example.net:2013:A2를 가지며 PortGroup으로 VLAN 1781을 가진다. STP가 가진 라벨에서 Port Group이 없을 수 있으며 이 경우 단지 STP 라벨은 목적지로 찾아 가기 위한 기능만을 가진다. 사용자는 STP 그룹을 이용해 목적지를 찾고 목적지에서 사용할 수 있는 포트 정보를 조회할 수 있다.

NSI 토폴로지 명세는 NML을 기반으로 정의된다. NSI는 이른 위해 NML 그룹과 협력하고 있다. NSI를 위한 NML 네임스페이스는 다음과 같다

- http://schemas.ogf.org/nsi/2013/03/topology#.

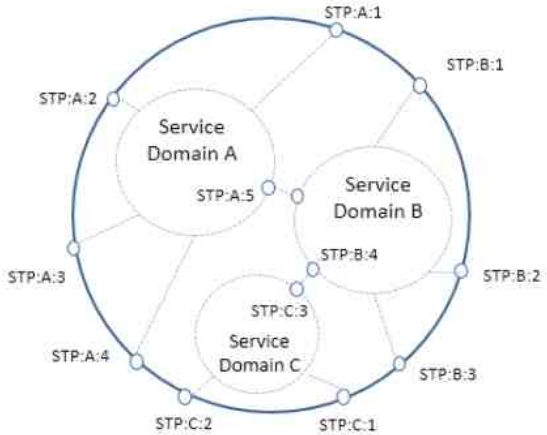
NSI Concept	*Representation*
STP Local ID	2x `nml:Port` / `nml:BidirectionalPort`
Connected To	`nml:isAlias`
NSNetwork	`nml:Topology`
Has STP	`nml:hasPort`
Located at	`nml:locatedAt`
Location	`nml:Location`
GPS coords	`nml:lat`, `nml:long`
NSA	`nsi:NSA`
Network managed by NSA	`nsi:managedBy`
Admin Contact	`nsi:adminContact`
Provider endpoint URL	`nsi:csProviderEndpoint`
Control-plane connections	`nsi:peersWith`

NML을 활용한 NSI Topology 명세의 예제이다.

```
1 @prefix nml: <http://schemas.ogf.org/nml/2013/10/base#> .
2 @prefix nmleth: <http://schemas.ogf.org/nml/2013/10/ethernet#> .
3 @prefix nsi: <http://schemas.ogf.org/nsi/2013/03/topology#> .
4 @prefix exa: <urn:ogf:network:example.com:2013:> .
5 @prefix exb: <urn:ogf:network:example.org:2013:> .
6
7 exa:ExampleCom a nml:Topology ;
8 nml:version "2013012301" ;
9 nml:name "exa" ;
10 nml:locatedAt exa:location ;
11 nml:hasOutboundPort exa:eth0´out ;
12 nml:hasInboundPort exa:eth0´in ;
13 nml:hasOutboundPort exa:eth1´out ;
14 nml:hasInboundPort exa:eth1´in ;
15 nsi:managedBy exa:nsa .
16 exa:location a nml:Location ;
17 nml:lat "55.637"^^<http://www.w3.org/2001/XMLSchema#float> ;
18 nml:long "12.641"^^<http://www.w3.org/2001/XMLSchema#float> .
19 exa:nsa a nsi:NSA ;
20 nsi:csProviderEndpoint "http://nsa.example.com/" ;
21 nsi:peersWith sara:nsa .
22 exa:eth0´out a nml:PortGroup ;
23 nmleth:vlans "1780-1783" ;
24 nml:isAlias exb:if0´in .
25 exa:eth0´in a nml:PortGroup ;
26 nmleth:vlans "1780-1783" ;
27 nml:isAlias exb:if0´out .
```

```
28 exa:eth0-out a nml:PortGroup ;
29 nmleth:vlans "1780-1783" .
30 exa:eth0-in a nml:PortGroup ;
31 nmleth:vlans "1780-1783" .
32
33 exb:nsa a nsi:NSA ;
34 nsi:csProviderEndpoint "http://nsa.example.org"
```

NSI 네트워크 서비스 도메인은 STP 그룹들의 셋으로 이루어져 있다. 네트워크는 하나 이상의 서비스 도메인을 포함할 수 있으며 각 STP는 같은 서비스 도메인 안의 모든 STP에 연결 될 수 있다. 각 서비스 도메인은 서비스 명세(Service Definition)를 통해 서비스 도메인에서 제공하는 내용을 정의할 수 있다.



3. NSI CS(Connection Service)

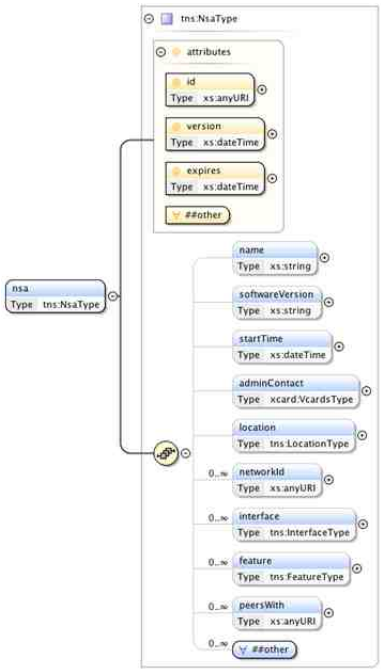
NSI CS 지원하는 NSA는 반드시 다음 사항을 지원하여야 한다.

Requirement	Description	Functional Area
1	MUST be able to describe NSI interfaces and versions of interfaces supported by the NSA.	Schema
2	MUST be able to describe supported protocol features of a specific protocol version supported by the NSA.	Schema
3	MUST be able to describe new protocols, protocol versions, and features without needing to upgrade the schema or associated protocol.	Schema + Protocol
4	MUST provide support for protocol version negotiation, allowing peer NSA to negotiate a mutually supported version of the protocol.	Schema
5	Shall allow bootstrap of peer communications with minimal configuration.	Schema + Protocol
6	Transport of the NSA meta data information MUST have equivalent levels of security as existing NSI protocols.	Protocol
7	The NSA Discovery document MUST be verifiable (e.g. the agent MUST be able to determine that the contents of the NSA Description Document was not altered during delivery).	Protocol
8	MUST support the discovery of multiple independent NSA Discovery Document types (representations).	Schema + Protocol
9	1.1.1.1 MUST support the discovery of multiple versions of the same NSA Discovery document.	Schema + Protocol
10	MUST be able to detect when new documents or new versions of existing documents are available.	Protocol
11	MUST be able to be notified when new	Protocol

	documents or new versions of existing documents are available.	
12	MUST be able to discover the unique NSA identifier of a peer NSA. Will reduce bootstrap provisioning requirements.	Schema + Protocol
13	MUST be able to discover the NSA software type and version running on a peer NSA. This will allow an NSA to adapt behaviors to specific version of NSA when required.	Schema
14	MUST be able to discover the time at which the peer NSA last started to provide uninterrupted service. This is effectively the last restart time of the NSA. A peer discovering a change in this value can initiate recovery procedures.	Schema
15	MUST be able to discover administrative contacts associated with the peer NSA.	Schema
16	MUST be able to discover the physical location of the peer NSA entity. This can be the location of the server hosting the NSA, or some other location related to the service being offered. This is used for visualization applications and troubleshooting.	Schema
17	MUST be able to discover the networks being managed by the peer NSA.	Schema
18	MUST be able to discover compete network control plane topology. This implies discovery of all NSA peering relationships within the network.	Schema
19	MUST be able to determine the peer NSA's CS role within the network (Aggregator, uRA, uPA). This will allow an NSA to find a peer aggregator to service CS requests.	Schema
20	MUST be able to determine the NSA's	Schema

	CS role of all NSA within the network (Aggregator, uRA, uPA). This is required to compute messaging paths in concert with control plane topology (NSA peering).	
21	MUST provide an extensible mechanism to allow additional discovery data to be added to an existing NSA's metadata without needing to upgrade the schema.	Schema

표의 요구사항을 바탕으로 NSI Discovery 서비스에서 사용되는 NSA 명세를 기술할 수 있다. 각 NSA는 반드시 <nsa> 엘리먼트를 포함하고 이를 기술해야 한다.



다음은 위의 NSA Discovery 엘리먼트를 활용한 NSI Discovery 기술 문서의 예시이다.

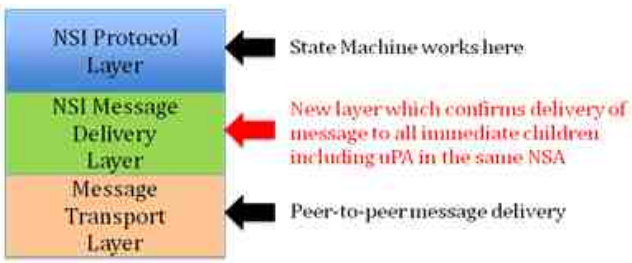
```
<tns:nsa xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns:xcard="urn:ietf:params:xml:ns:vcard-4.0"
         xmlns:tns="http://schemas.ogf.org/nsi/2014/02/discovery/nsa"
    id="urn:ogf:network:example.com:2013:nsa:vixen"
    version="2014-01-04T18:13:51.0Z"
    expires="2014-02-04T18:13:51.0Z">
    <name>Example NSA</name>
    <softwareVersion>ExampleNsa-Version-1.0</softwareVersion>
```

```
    <startTime>2014-01-01T18:13:51.0Z</startTime>
    <adminContact>
        <xcard:vcard>
            <xcard:uid>
                <xcard:uri>http://www.example.com/santa.claus/santa.asc</xcard:uri>
            </xcard:uid>
            <xcard:prodid><xcard:text>OGF          Example          Maker          //
EN</xcard:text></xcard:prodid>

<xcard:rev><xcard:timestamp>20080424T195243Z</xcard:timestamp></xcard:rev>
            <xcard:kind><xcard:text>individual</xcard:text></xcard:kind>
            <xcard:fn><xcard:text>Saint Nicholas</xcard:text></xcard:fn>
            <xcard:n>
                <xcard:surname>Claus</xcard:surname>
                <xcard:given>Santa</xcard:given>
                <xcard:suffix>Saint</xcard:suffix>
            </xcard:n>
            <xcard:tel><xcard:text>+1 555-555-5555</xcard:text></xcard:tel>
            <xcard:email>
                <xcard:text>santa.claus@theworkshop.example.com</xcard:text>
            </xcard:email>
        </xcard:vcard>
    </adminContact>
    <location>
        <name>Santa's Workshop, The North Pole</name>
        <longitude>0.0000</longitude>
        <latitude>90.0000</latitude>
        <altitude>10</altitude>
        <address>
            <xcard:pobox>0001</xcard:pobox>
            <xcard:ext></xcard:ext>
            <xcard:street>1 Top of the world boulevard</xcard:street>
            <xcard:locality>Polar Ice Flows</xcard:locality>
            <xcard:region>The North Pole</xcard:region>
            <xcard:code>CA</xcard:code>
            <xcard:country>Canada</xcard:country>
        </address>
    </location>
    <networkId>urn:ogf:network:example.com:2013:network:theworkshop</networkId>
    <networkId>urn:ogf:network:example.com:2013:network:candycaneforest</networkId>
    <interface>
        <type>application/vnd.ogf.nsi.dds.v1+xml</type>
        <href>https://nsa.example.com/dds</href>
        <describedBy>https://nsa.example.com/dds?wadl</describedBy>
    </interface>
    <interface>
        <type>application/vnd.ogf.nsi.topology.v1+xml</type>
        <href>https://nsa.example.com/topology.xml</href>
    </interface>
    <interface>
```

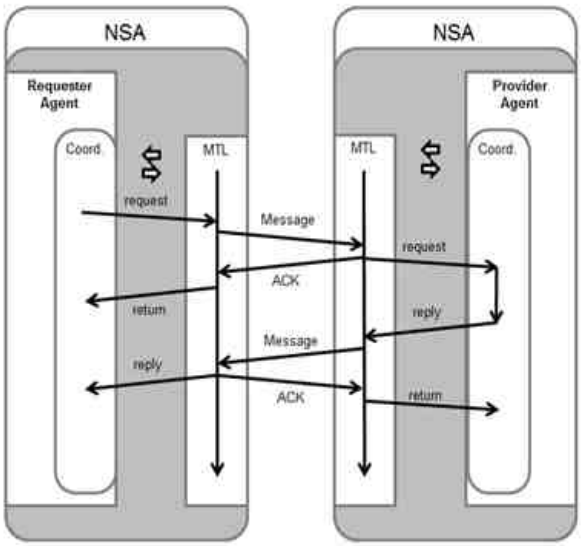
```
        <type>application/vnd.ogf.nsi.cs.v2.provider+soap</type>
        <href>https://nsa.example.com/connectionProvider</href>
        <describedBy>https://nsa.example.com/connectionProvider?wsdl</describedBy>
    </interface>
    <interface>
        <type>application/vnd.ogf.nsi.cs.v2.requester+soap</type>
        <href>https://nsa.example.com/connectionRequester</href>
        <describedBy>https://nsa.example.com/connectionRequester?wsdl</describedBy>
    </interface>
    <feature type="org.ogf.nsi.cs.v2.role.aggregator"/>
    <feature type="org.ogf.nsi.cs.v2.role.uPA"/>
    <feature type="org.ogf.nsi.cs.v2.commitTimeout">120</feature>
    <peersWith>urn:ogf:network:example.com:2013:nsa:dasher</peersWith>
    <peersWith>urn:ogf:network:example.com:2013:nsa:dancer</peersWith>
    <peersWith>urn:ogf:network:example.com:2013:nsa:prancer</peersWith>
</tns:nsa>
```

NSI CS NSI protocol Layar / NSI Message Layer / Message Transport Layer를 정의하고 있다. 각각의 NSA간 메시지 전달체계에 있어서 상호 연동을 통한 안전한 메시지 전달을 위한 메시지 전달 레이어를 따로 만들어 안정한 통신을 유도한다. 이러한 메시지 전달체계는 향후 복잡해지는 글로벌 토폴로지의 반영을 적극 유도할수 있으며, 복잡도의 진화에 따른 시스템간 효과적인 방식이다.

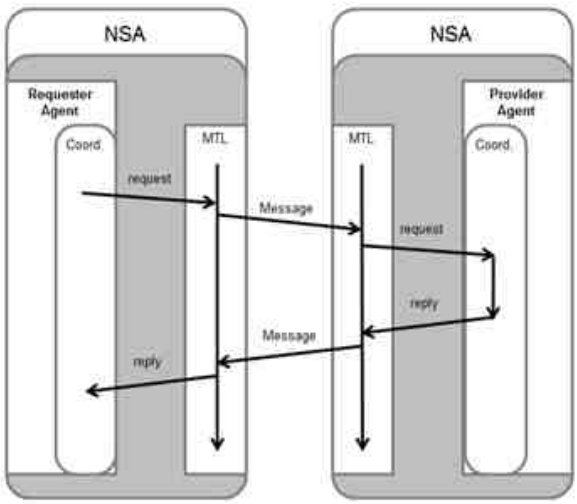


Conceptual Layering in NSA

NSI는 MDL 레이어와 함께 동기(Sync)/비동기(Async) 메시지를 가진다. 대부분의 NSI 메시지는 비동기로 작동하나, 일부 조회(Query)와 알림(Notification) 메시지가 동기적으로 동작한다. 동기적으로 작동하는 오퍼레이션의 경우 오퍼레이션 명에 _Sync가 포함되게 된다. 동기 메시지는 현재 비동기에서의 Call back 메시지 제거 및 방화벽 회피를 위하여 추가적으로 정의되고 있으나 아직까지는 정식으로 사용되고 있지 않다.

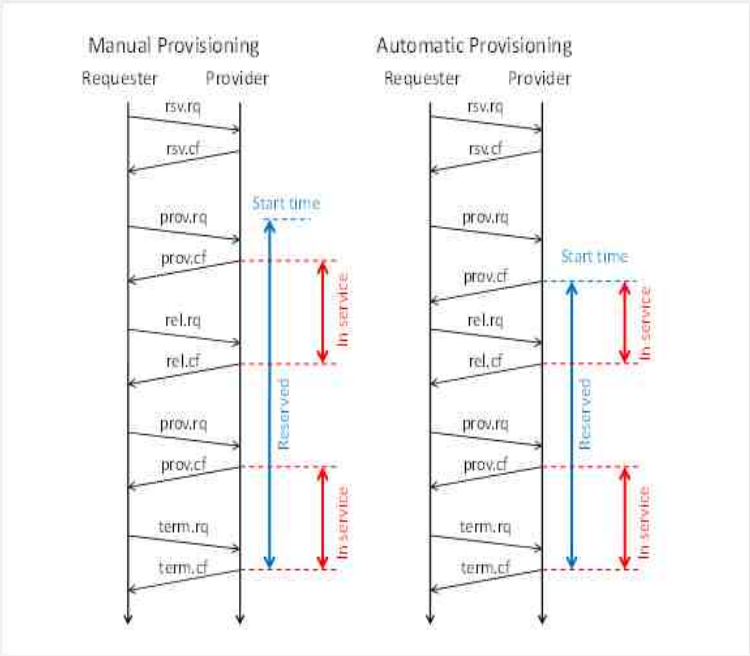


NSI 비동기 메시지



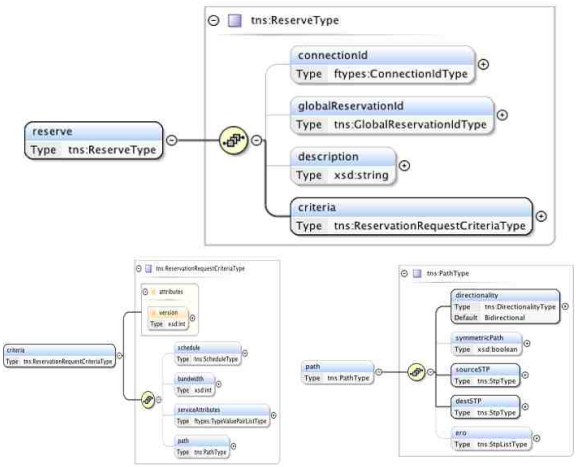
NSI 동기 메시지

NSI CS 자원 예약에서의 편의를 위해 Manual Provisioning과 Automatic Provisioning을 정의하고 있다. 각각의 NSA들은 예약 작업을 진행할 때 Start Time이 실제 provision timeq 앞설 때를 Manual provisioning이라고 하고, Start time이 provisioning time과 같거나 늦을때를 Automatic Provisioning이라고 한다. 이것은 Servic lifecycle에서 중요하며, 각각의 requester가 보내는 메시지의 타이밍에 따른 모드의 변경으로 로컬 nrm의 동작에 영향을 미친다. 이렇게 사용자의 특성과 예약의 특성을 반영하여 NSI의 동작이 이루어진다.

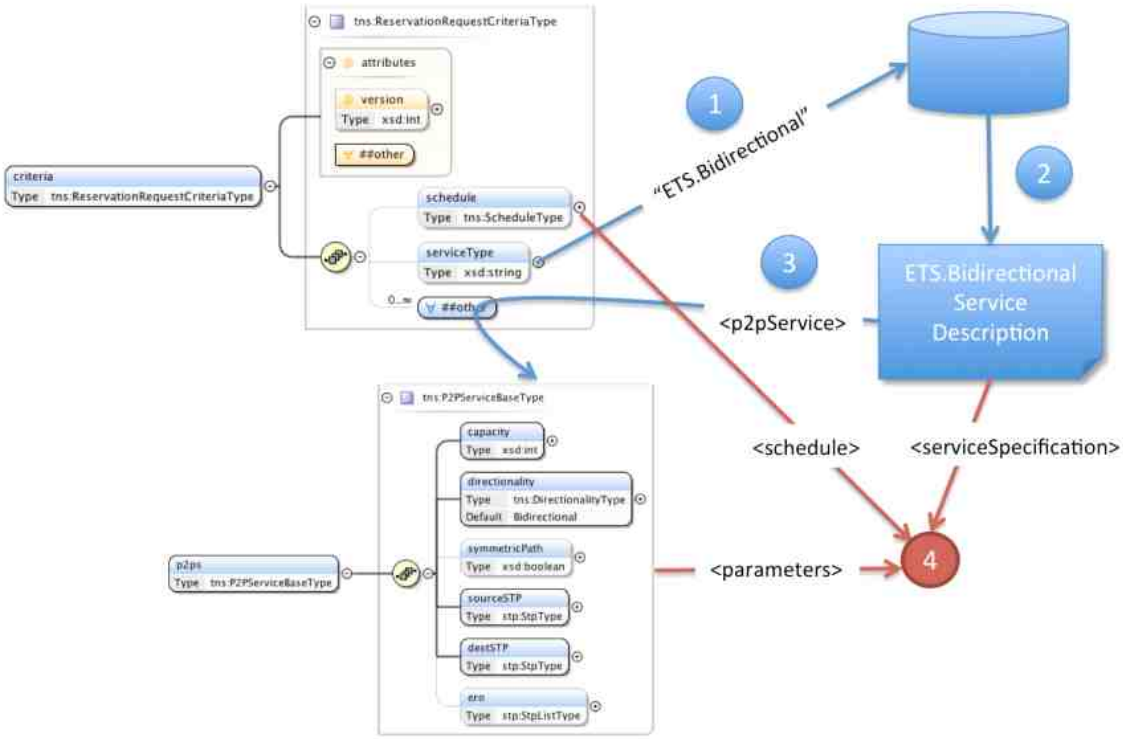


Connection Service lifecycle

NSI 2.0 가장 큰 변화는 바로 Service Decoupling이라 할 수 있다. 이는 NSI CS r117에서도 동일하게 적용되고 있다. 아래 그림은 NSI 1.1에서의 예약 인자이다. 기존 NSI 1.1은 서비스 평면 제어를 위한 NSI 인자 및 전달 평면 제어를 위한 인자를 해당 프레임 워크에 모두 포함하고 있었다. 이로 인해 전달 평면에 새로운 서비스를 제공하기 위해서는 NSI 프레임워크 전체를 다시 컴파일 해야 하는 문제를 내포하고 있다. NSI 2.0에서는 XML 스키마에서 Any 엘리먼트 타입을 이용하여 NSI 프레임워크와 전달망의 서비스 정의의 분리를 시도하고 있다. 간단히 표현하여 NSI 프레임워크 상에 전달망 제어를 위한 인자를 Any 타입으로 정의하며, 전달망 제어 서비스 인자를 별도의 스키마로 정의하여 사용하는 것이다.



NSI v1.1 예약 메시지

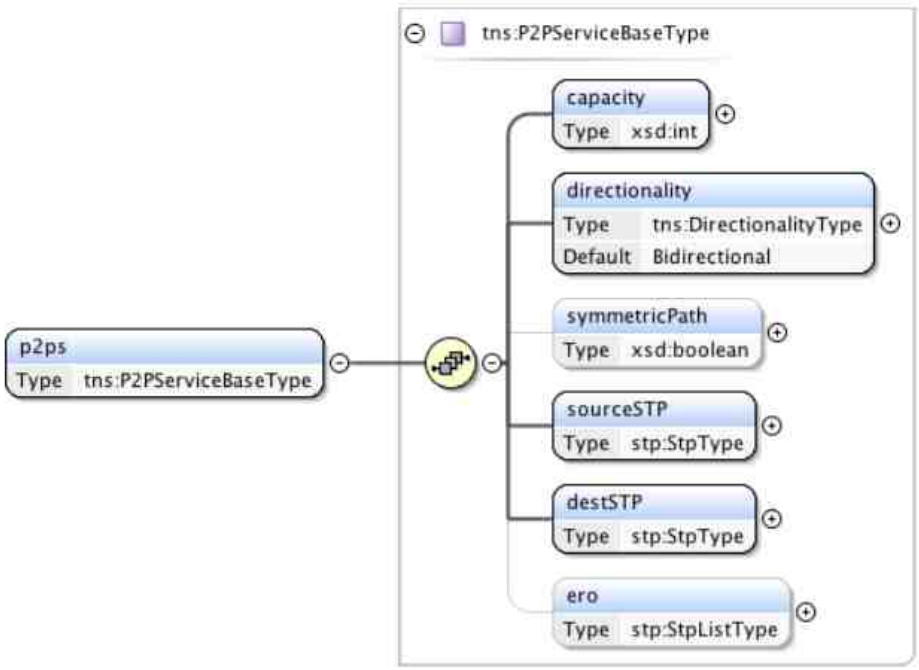


Any Type을 활용한 Service Decoupling

21은 Any Type을 활용한 Service Decoupling 방법과 적용 방법을 보여주고 있다. 순서는 아래와 같다.

1. Extract ServiceType value from incoming reservation request and lookup Service Description corresponding to serviceType.
2. User Service Description to determine the service elements needed for the specific service requested and any other service related parameters.
3. Extract specific services elements from criteria as described in Service Description.
4. Process service request using supplied service parameters and servcie template information.

NSI r117에서는 점차 다양해지는 서비스 명세를 위해 서비스 명세의 정의 및 배포를 좀 더 유연하게 할 수 있도록 많은 토의가 이루어 지고 있으나 현재까지는 NSI v2.0과 동일한 방식으로 Service Decoupling 및 Service Definition을 정의하고 있다. 아래 그림은 Service Description을 위한 스키마 구조 예시이다.



Service Description 예시

NSI CS 1.0에서 시작하여 여러번의 변화가 있었으며 이를 다음과 같이 업데이트 되었다.

Requirement	Description	Functional Area
NSI CS version 1.0	Provider	“application/vnd.ogf.nsi.cs.v1.provider+soap”
NSI CS version 1.0	Requester	“application/vnd.ogf.nsi.cs.v1.requester+soap”
NSI CS version 1.1	Provider	“application/vnd.ogf.nsi.cs.v1-1.provider+soap”
NSI CS version 1.1	Requester	“application/vnd.ogf.nsi.cs.v1-1.requester+soap”
NSI CS version 2.0	Provider	“application/vnd.ogf.nsi.cs.v2.provider+soap”
NSI CS version 2.0	Requester	“application/vnd.ogf.nsi.cs.v2.requester+soap”
NSI Topology version 1.0	Provider	“application/vnd.ogf.nsi.topology.v1+xml”
NSI Topology version 2.0	Document	“application/vnd.ogf.nsi.topology.v2+xml”
NSA Description Document version 1.0	Document	“application/vnd.ogf.nsi.nsa.v1+xml”
NSI Document Distribution Service version 1.0	Requester/ Provider	“application/vnd.ogf.nsi.dds.v1+xml”

표는 NSI CS의 메시지이다.

- RA to PA 메시지

NSI 메시지는 크게 RA to PA 메시지와 PA to RA 메시지로 나눌 수 있다.

NSI CS Message	SM	Synch/Asynch	Description
reserve	RSM	Asynch	요청
reserveCommit	RSM	Asynch	예약 확정
reserveAbort	RSM	Asynch	예약 거절
provision	PSM	Asynch	구성 요청
release	PSM	Asynch	해제 요청
terminate	LSM	Asynch	종결 요청
querySummary	Query	Asynch	조회 요청
queryRecursive	Query	Asynch	조회(Agg) 요청 *1
querySummarySynch	Query	Synch	조회 요청
queryNotification	Query	Asynch	Notification 조회 요청
queryNotificationSynch	Query	Synch	Notification 조회 요청

RA to PA

- PA to RA 메시지

NSI CS Message	SM	Synch/Asynch	Description
reserveConfirmed	RSM	Asynch	예약 성공
reserveFailed	RSM	Asynch	예약 실패
reserveCommitConfirmed	RSM	Asynch	예약 확정 성공
reserveCommitConfirmed	RSM	Asynch	예약 확정 실패
reserveAbortConfirmed	RSM	Asynch	예약 거절 성공
provisionConfirmed	PSM	Asynch	구성 성공

releaseConfirmed	PSM	Asynch	성공
terminateConfirmed	LSM	Asynch	종결 성공
querySummaryConfirmed	Query	Asynch	조회 성공
querySummaryFailed	Query	Asynch	조회 실패
queryRecursiveConfirmed	Query	Asynch	조회(Agg) 성공
queryRecursiveFailed	Query	Asynch	조회(Agg) 실패
querySummarySynch Confirmed	Query	Synch	조회 성공
querySummarySynchFailed	Query	Synch	조회 실패
queryNotificationConfirmed	Query	Asynch	Notification 조회 성공
queryNotificationFailed	Query	Asynch	Notification 조회 실패
queryNotificationSync Confirmed	Query	Synch	Notification 조회 성공
queryNotificationSyncFailed	Query	Synch	Notification 조회 실패
error	Error	Asynch	Error 발생 *2
errorEvent	Notification	Asynch	Error Notification 발생
reserveTimeout	Notification	Asynch	예약을 하였으나 일정 시간내에 Commit 하지 않아 예약이 실패
dataPlaneStatChange	Notification	Asynch	구성 및 해제 요청을 하였으나 전달망에서 에러 발생
messageDeliveryTimeout	Notification	Asynch	요청 메시지를 발송 하였으나 Timeout 발생

PA to RA

*1 QuerySummary QueryRecursive의 차이는 Ultimate Provider 모드와 Aggregator모드의 차이와 동일하다. QuerySummary 요청의 경우 하위 NSA와 관계 없이 자신이 가지고 있는 정보만을 리턴하며 QueryRecursive의 경우 하위 NSA에게 까지 정보를 조회한다.

*2 NSI 2.0에서 에러는 크게 ServiceException을 기반으로 한 에러와 WSDL상에 정의된 에러 메시지로 나누어진다. error의 경우 ServiceException을 통하여 발생하는 에러를 의미한다.